Day 1, Practical 1, Hely's solution

Helene Charlotte Wiese Rytgaard

September 28, 2021

1 Simulating data

Task 1.

```
library(data.table)

sim.fun <- function(n) {

    X1 <- runif(n, -2, 2)
    X2 <- rnorm(n, 0, 0.2)
    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))

}
```

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
```

2 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 0.000000000000113 ***
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 0.000000001180179 ***
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;
```

```r
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.

```r
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.

```r
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
```

```r
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 7.

```r
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
```

3 Simulation study

Task 8.

```
fit.g.glm1 <- list()
fit.g.glm2 <- list()
fit.g.rf <- list()
fit.ipw <- list()
fit.tmle <- list()
fit.one.step <- 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)

    ##-- g-formula (section 3.1);
    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)]

    ##-- g-formula (section 3.2);
    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)]

    ##-- g-formula based on RF (section 3.3);
    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 (section 3.1);
    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 (section 3.1);
    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

    ##-- one-step estimator
    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.one.step[[m]] <- sim.data[, mean((A/pred.pi1 - (1-A)/(1-pred.pi1))*(Y - pred.f
    ) +
                     pred.f1 - pred.f0)]

}
```

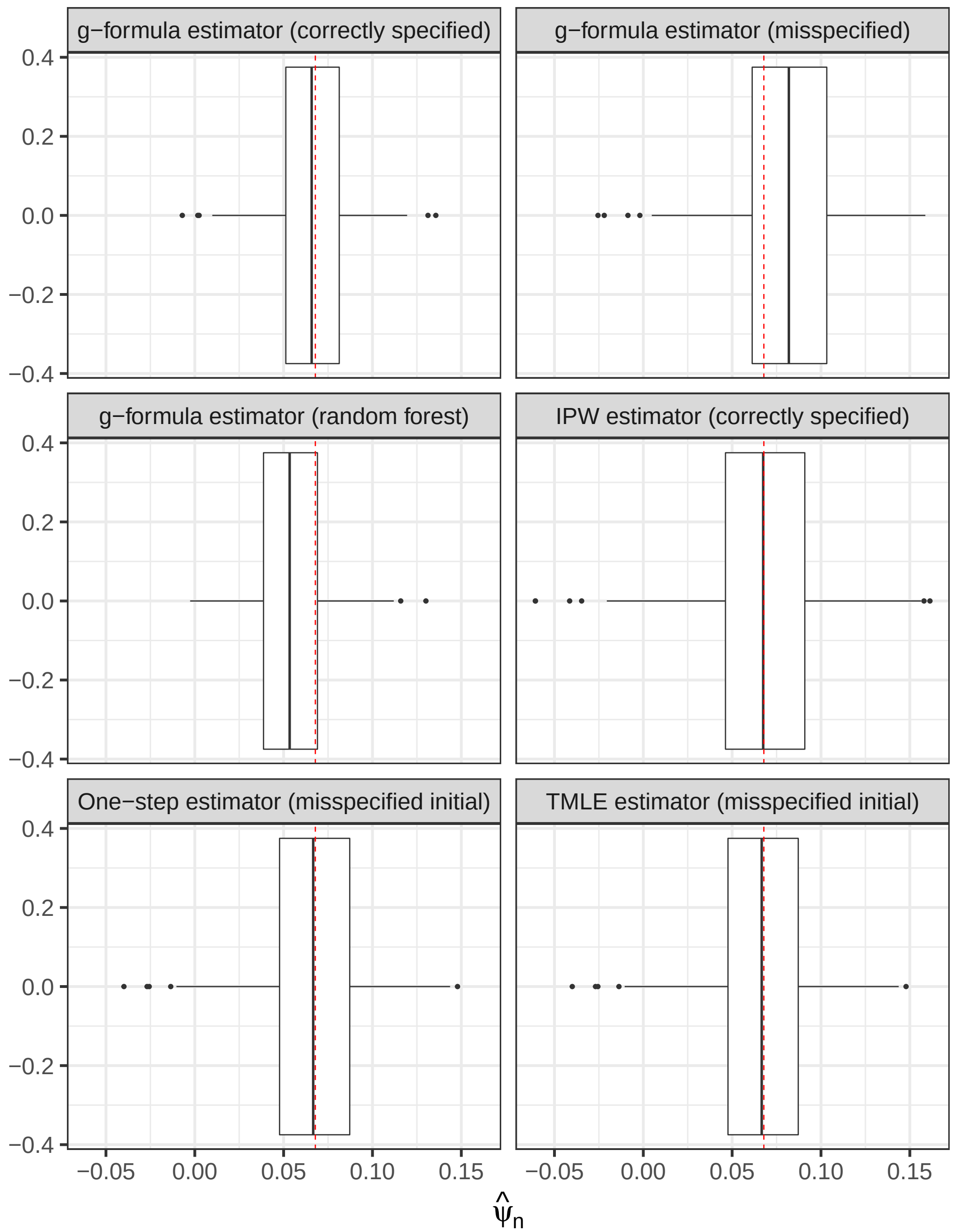
Task 9. See Figure 1. Note that I have added the one-step estimator to show equivalence to TMLE.

```
setwd("~/Undervisning/TMLE/beamer/day1/")
library(ggplot2)

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

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

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



g−formula estimator (correctly specified)
g−formula estimator (misspecified)
g−formula estimator (random forest)
IPW estimator (correctly specified)
One−step estimator (misspecified initial)
TMLE estimator (misspecified initial)
0.4
0.2
0.0
−0.2
−0.4
−0.05
0.00
0.05
0.10
0.15
$\hat{\psi}_n$

Figure 1

4 Simulation study with positivity issues

Extra for task 9. Same as **Task 9** but changing the distribution of A as follows: $\mathbb{E}[A \mid X_1, X_2, X_3] = \text{logit}(-0.25 + 2.8X_1 + 0.25X_3)$. See Figure 3. Note that I have added the one-step estimator to show better finite-sample distribution of TMLE.

```
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))
    }
}
```

```
fit.g.glm1 <- list()
fit.g.glm2 <- list()
fit.g.rf <- list()
fit.ipw <- list()
fit.tmle <- list()
fit.one.step <- 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)

    ##-- g-formula (section 3.1);
    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)]

    ##-- g-formula (section 3.2);
    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)]
```

```r
    ##-- g-formula based on RF (section 3.3);
    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 (section 3.1);
    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 (section 3.1);
    tmle.fit <- tmle(Y=sim.data$Y, A=sim.data$A,
             cbind(X1=sim.data$X1,X2=sim.data$X2,X3=sim.data$X3),
             gform=A~X1+X2+X3, ## treatment model
             Qform=Y~A+X1+X2+X3, ## outcome model
             family="binomial",
             gbound=0,
             cvQinit=FALSE)
    fit.tmle[[m]] <- tmle.fit$estimates$ATE$psi

    ##-- one-step estimator
    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.one.step[[m]] <- sim.data[, mean((A/pred.pi1 - (1-A)/(1-pred.pi1))*(Y - pred.f
    ) +
                     pred.f1 - pred.f0)]


}
```

```r
setwd("~/Undervisning/TMLE/beamer/day1/")
library(ggplot2)

pdat <- data.table(estimator=c(rep("g-formula estimator (misspecified)",
                   length(fit.g.glm1)),
                   rep("g-formula estimator (correctly specified)",
                   length(fit.g.glm2)),
                   rep("g-formula estimator (random forest)",
                   length(fit.g.rf)),
                   rep("IPW estimator (correctly specified)",
                   length(fit.ipw)),
                   rep("TMLE estimator (misspecified initial)",
                   length(fit.tmle)),
                   rep("One-step estimator (misspecified initial)",
                   length(fit.one.step))),
           est=c(unlist(fit.g.glm1),
             unlist(fit.g.glm2),
             unlist(fit.g.rf),
             unlist(fit.ipw),
             unlist(fit.tmle),
```

```r
            unlist(fit.one.step)))

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

```r
setwd("~/Undervisning/TMLE/beamer/day1/")
library(ggplot2)

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

ggplot(pdat[estimator %in% c("TMLE estimator (misspecified initial)",
                 "One-step estimator (misspecified initial)")]) +
    theme_bw(base_size=25) +
    geom_boxplot(aes(x=est)) +
    facet_wrap(. ~ estimator, ncol=2) +
    geom_vline(aes(xintercept=ATE), linetype="dashed", color="red") +
    xlab(expression(hat(psi)[n])) + ylab("")
```

```r
message(paste0("mse tmle: ", mean((unlist(fit.tmle) - mean(unlist(fit.tmle)))^2)))
message(paste0("mse one step: ", mean((unlist(fit.one.step) - mean(unlist(fit.one.step
    )))^2)))
message(paste0("variance tmle: ", var(unlist(fit.tmle))))
message(paste0("variance one step: ", var(unlist(fit.one.step))))
message(paste0("bias tmle: ", mean(unlist(fit.tmle)-ATE)))
message(paste0("bias one step: ", mean(unlist(fit.one.step)-ATE)))
```

```
mse tmle: 0.00504335124157197
mse one step: 0.0055914632575187
variance tmle: 0.00505345815788774
variance one step: 0.00560266859470812
bias tmle: 0.00498946280696468
bias one step: 0.00403961703692303
```

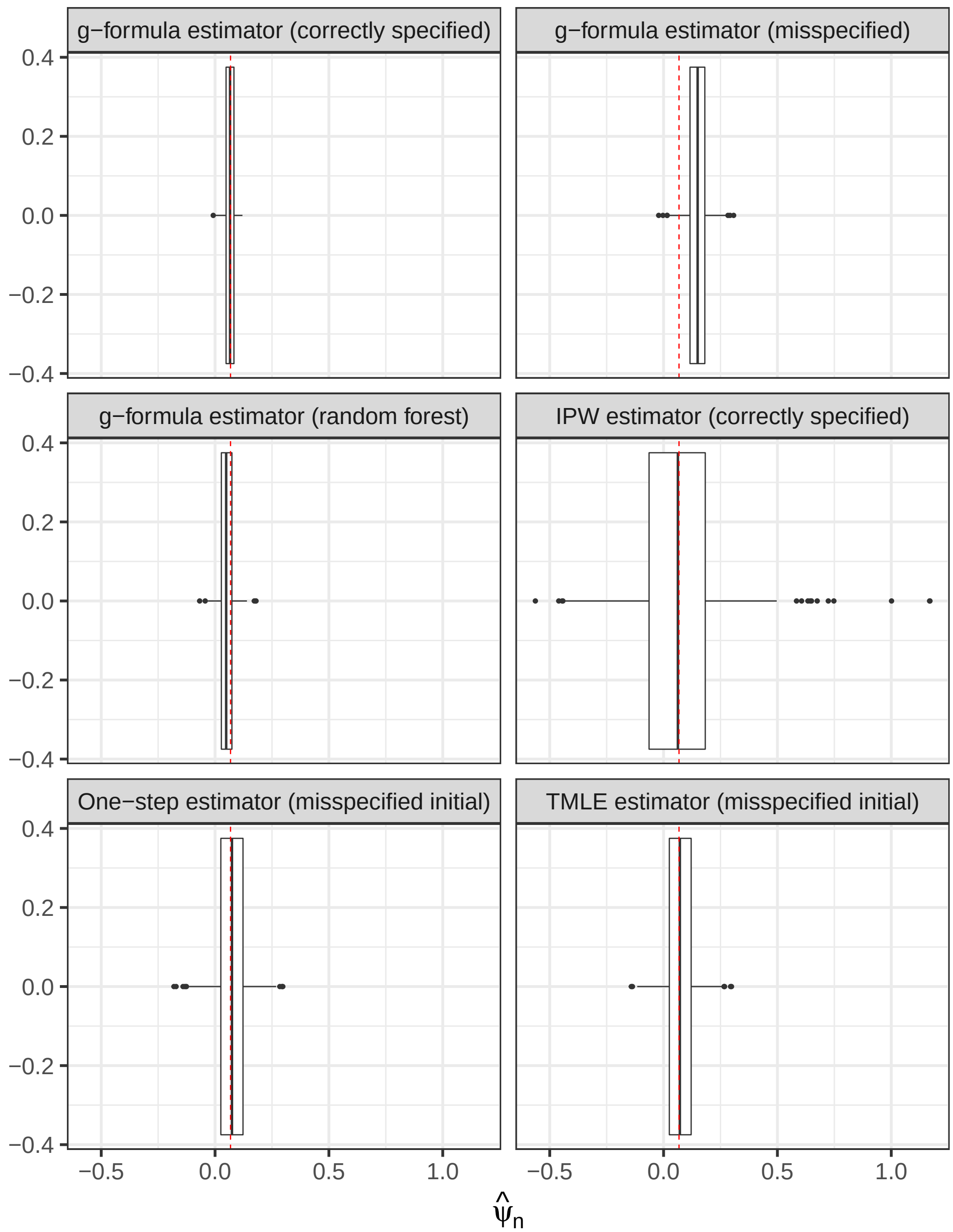

g−formula estimator (correctly specified)
g−formula estimator (misspecified)
g−formula estimator (random forest)
IPW estimator (correctly specified)
One−step estimator (misspecified initial)
TMLE estimator (misspecified initial)
0.4
0.2
0.0
−0.2
−0.4
−0.5
0.0
0.5
1.0
$\hat{\psi}_n$

Figure 2

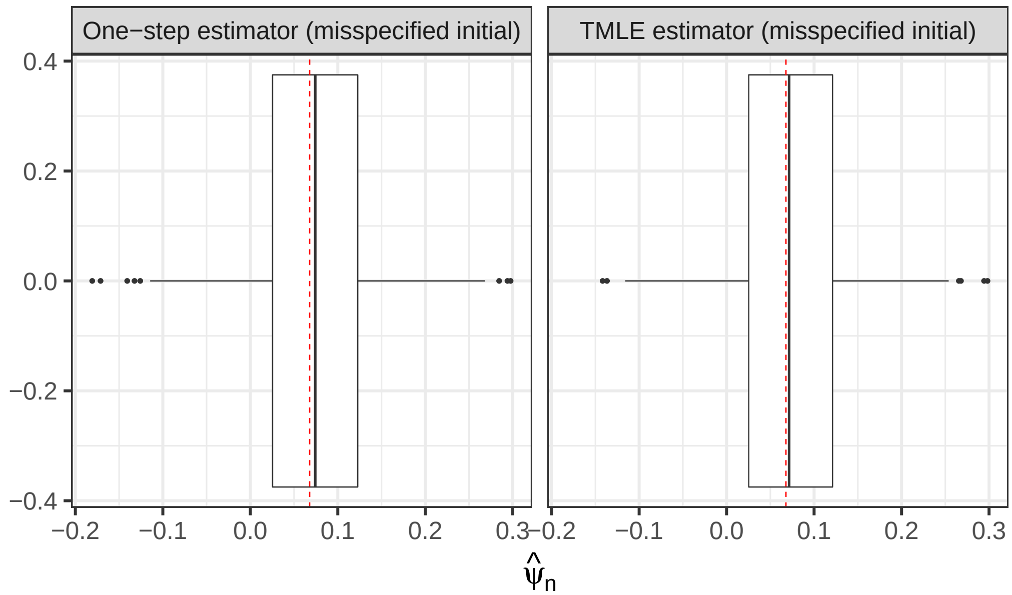


Figure 3

5 Simulation study: Tuning the random forest

```
setwd("~/Undervisning/TMLE/beamer/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")))])
}
```

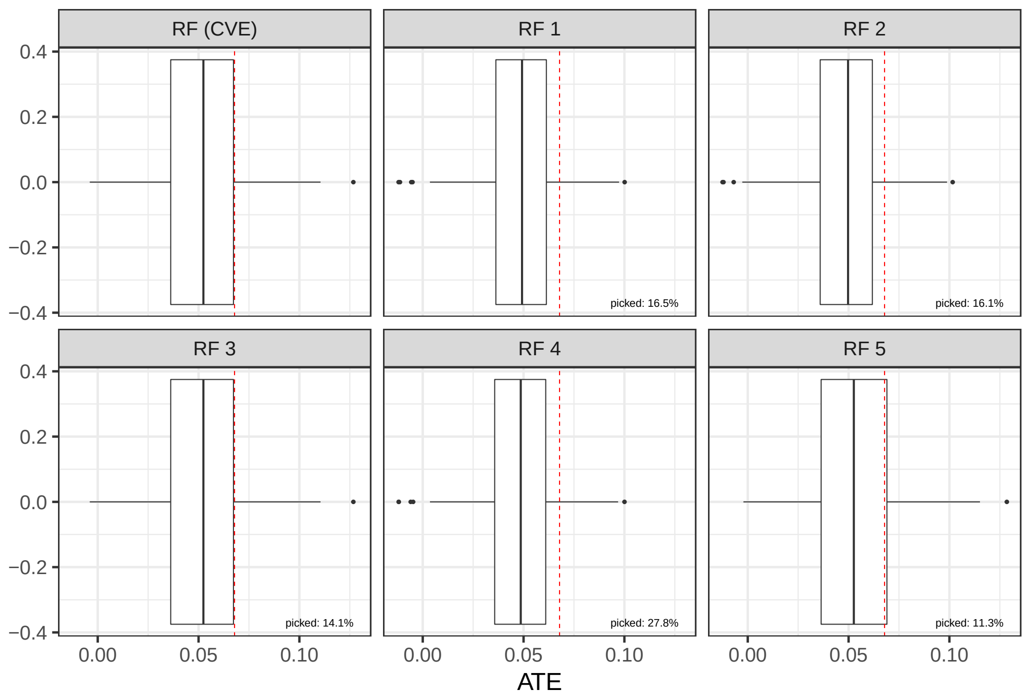

RF (CVE)
RF 1
RF 2
RF 3
RF 4
RF 5
picked: 16.5%
picked: 16.1%
picked: 14.1%
picked: 27.8%
picked: 11.3%
0.4
0.2
0.0
−0.2
−0.4
0.00
0.05
0.10
ATE

Figure 4