

# SimulationComparisonWithCompetitors

Disclaimer: These simulations are not necessarily representative of the performance of these methods in the real-world. The simulation models are all randomly generated main-term parametric models and there are little-to-no positivity issues. Because of this, glm/glmnet will do better than other machine-learning algorithms in the below simulations. To make the simulations more difficult, take a look at the arguments to the function `sim.causalGLM`. The main point of these simulations is show the robust performance of `causalGLM` in small sample sizes and settings where parametric glm would do well. We hope these simulations convince you that semiparametric methods need not come at a cost in power/robustness relative to glm in regimes with small sample sizes, even when the truth is a simple parametric model.

## Random examples of simulation datasets

```
library(causalGLM)
```

```
## Loading required package: sl3
```

```
## Loading required package: hal9001
```

```
## Loading required package: Rcpp
```

```
## hal9001 v0.4.0: The Scalable Highly Adaptive Lasso
```

```
## note: fit_hal defaults have changed. See ?fit_hal for details
```

```
## Loading required package: data.table
```

```
## Loading required package: R6
```

```
n <- 50
```

```
p <- 5
```

```
data_sim <- sim.CATE(n=n, p=p, formula_estimand = ~1 + W1 + W2 + W3 + W4, formula_A = ~., formula_YOW
```

```
# simulated data and true nuisance functions
```

```
head(data_sim$data)
```

```
##           W1           W2           W3           W4           W5 A           Y
## 1 -0.8634133  0.31147395 -0.05437368 -0.61653509  0.4533747 0 -0.4925576
## 2  0.2864373  0.09586197 -0.71619320  0.70548316  0.7058047 1 -1.1984018
## 3  0.1782600  0.45242165  0.69143438  0.06521152 -0.1342753 0  0.2544379
## 4 -0.6274913  0.70606042  0.34102032 -0.14845691 -0.9075634 1  0.3326014
## 5  0.4455131  0.15039213 -0.30658243 -0.53477437 -0.1093241 1 -0.1299451
## 6  0.6992097 -0.34650912 -0.77933084 -0.61473479 -0.7531079 1 -0.6845901
##           pA1           pY           sd           CATE
## 1 0.4540912 -0.17070952 0.17194  0.4455508
## 2 0.4469395 -0.82637811 0.17194 -0.3847293
## 3 0.5039092  0.36157431 0.17194 -0.3910504
## 4 0.5618906  0.31248048 0.17194  0.3345186
## 5 0.3590636 -0.09979905 0.17194 -0.2736096
## 6 0.3438602 -0.72472878 0.17194 -0.5439853
```

```
# True coefs
```

```
data_sim$beta_CATE
```

```
## [1] -0.3479335 -0.5189985 0.6026423 -0.3012645 -0.2291611
```

```
n <- 50
p <- 5
data_sim <- sim.RR(n=n, p=p, formula_estimand = ~1 + W1 + W2 + W3 + W4, formula_A = ~., formula_YOW = ~.,
# simulated data and true nuisance functions
head(data_sim$data)
```

```
##           W1           W2           W3           W4           W5 A Y           pA1
## 1  0.06526936  0.66166882 -0.768259513 -0.8531918 -0.7007032 1 3 0.4512356
## 2 -0.97643909 -0.03088573  0.431270692  0.8634789  0.3810071 1 2 0.3065994
## 3  0.93569961  0.98790277 -0.885422965  0.7749886  0.9947198 0 2 0.2543651
## 4 -0.48088661 -0.34247833 -0.003025467  0.3075676  0.7094941 0 1 0.3670288
## 5 -0.99317582  0.67335508 -0.972927345  0.9955734 -0.7278970 0 0 0.2945736
## 6  0.01970777  0.95942055  0.843189764  0.2931848  0.5659914 0 1 0.2777202
##           pY           RR
## 1 1.0646165 1.3581593
## 2 0.6085169 0.9170453
## 3 0.9040728 0.9808035
## 4 0.7538457 1.1259601
## 5 0.4096432 0.9239375
## 6 1.1828821 0.9012761
```

```
# True coefs
data_sim$beta_logRR
```

```
## [1] 0.15142684 0.03322117 -0.12826578 -0.08871756 -0.19836866
```

```
n <- 50
p <- 5
data_sim <- sim.OR(n=n, p=p, formula_estimand = ~1 + W1 + W2 + W3 + W4, formula_A = ~., formula_YOW = ~.,
# simulated data and true nuisance functions
head(data_sim$data)
```

```
##           W1           W2           W3           W4           W5 A Y           pA1
## 1 -0.5547048  0.2392246 -0.9551891 -0.49233436 -0.850171624 1 0 0.5662029
## 2 -0.4628820  0.5295956 -0.3259557 -0.08873086 -0.008091807 1 0 0.5268962
## 3  0.2247846 -0.6521099 -0.7158842  0.67663684  0.341291513 0 0 0.5136655
## 4  0.1229099 -0.3243321  0.3549943 -0.95301823 -0.624408116 1 0 0.5622219
## 5  0.4189643  0.7171637 -0.7671800 -0.32055329  0.568748479 1 0 0.6327392
## 6  0.3531201  0.8213957 -0.9700102 -0.34761860  0.039099015 0 1 0.6337371
##           pY           pY1           pY0           OR
## 1 0.2736606 0.2736606 0.3992686 0.5668757
## 2 0.3024170 0.3024170 0.4667250 0.4953365
## 3 0.4013451 0.5288561 0.4013451 1.6743360
## 4 0.2775260 0.2775260 0.4368610 0.4951695
## 5 0.3947370 0.3947370 0.6521919 0.3477988
## 6 0.5905057 0.3232773 0.5905057 0.3312747
```

```
# True coefs
data_sim$beta_logOR
```

```
## [1] -0.3922651 -0.1860321 -0.7615094 -0.1579696 0.5022238
```

## Simulation Comparison with estimating-equation competitors

The `sim.R` function contains customizable functions that randomly generate test data for both the CATE, RR, and OR functions. The functions `sim.causalGLM` and `sim.causalGLMwithLasso` internally call these functions and run `nsims` number of simulations and report the proportion of estimated 95% confidence intervals that contain the true coefficient values (as determined from the simulation). The confidence interval coverage for the competitor, estimating equation-based estimators, is also reported. Both methods are fit on the same simulation data with the same nuisance estimators and same variance estimator. Therefore, the randomness is only in the difference between the two methods. We will employ these methods to compare the TMLE method implemented in this package, `causalGLM`, with competitors.

We will focus on small  $n$  with a 4-dimensional covariate model for the estimand. In these settings, differences are more pronounced. The simulation data distributions are linear main-term parametric models, so that `glm` and `glmnet` are correctly specified. This may be unrealistic in some settings but it is an important benchmark. We would like these methods to do just as well as parametric `glm` in coverage when the assumptions are true.

Note sometimes due to monte-carlo randomness one method may by chance perform better than another. For most reliable and fair results, set “`nsims`” as high as possible, e.g. `nsims = 1000` or `2500`.

Note the competitor is asymptotically equivalent to `causalGLM`.

```
library(causalGLM)
seed <- 12345678

nsims <- 500
```

### CATE

The estimating equation for CATE is fairly simple (it is linear). Thus, estimating equation estimators should perform fairly well for the CATE relative to non-linear estimating equation estimators (like for RR). `### n = 50, p=5`

```
set.seed(seed)
n <- 50
p <- 5
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.

out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "DML")

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coefficient obtained by causalGLM. These
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W1      W2      W3      W4
##      0.968      0.934      0.942      0.922      0.948
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so far: "
## [1] 0.882 0.832 0.834 0.806 0.826
### You should see causalGLM do better than the estimating equation estimator.
```

`n = 100, p=5`

```
set.seed(seed)
n <- 100
```

```

p <- 5
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "DML")

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coeficient obtained by causalGLM. These are the
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W1      W2      W3      W4
##      0.978      0.980      0.954      0.964      0.972
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so far: "
## [1] 0.908 0.882 0.856 0.866 0.858

### You should see causalGLM do better than the estimating equation estimator.

```

n = 200, p=5

```

set.seed(seed)
n <- 200
p <- 5
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "DML")

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coeficient obtained by causalGLM. These are the
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W1      W2      W3      W4
##      1.000      1.000      1.000      0.996      1.000
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so far: "
## [1] 0.962 0.878 0.922 0.914 0.904

### You should see both methods do fairly well.

```

## RR

The estimating equation for RR is highly non-linear. This should lead to worse performance for the estimating equation in certain settings. ### n = 50, p=4

```

set.seed(seed)
n <- 50
p <- 4
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "DML")

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coeficient obtained by causalGLM. These are the
out$report()

```

```

out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W1      W2      W3      W4
##      0.868      0.876      0.860      0.898      0.874
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so
## [1] 0.850 0.820 0.794 0.830 0.794

### You should see causalGLM do quite a bit better than the estimating equation estimator. Even so, bot

```

n = 100, p=4

```

set.seed(seed)
n <- 100
p <- 4
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coeficient obtained by causalGLM. These
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W1      W2      W3      W4
##      0.920      0.916      0.928      0.908      0.920
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so
## [1] 0.912 0.892 0.892 0.878 0.868

### You should see causalGLM do better than the estimating equation estimator. The >0.95 coverage is li

```

n = 200, p=4

```

set.seed(seed)
n <- 200
p <- 4
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coeficient obtained by causalGLM. These
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W1      W2      W3      W4
##      0.942      0.926      0.932      0.942      0.922
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so
## [1] 0.930 0.910 0.902 0.910 0.912

### You should see both methods perform similarly.

```

OR

n = 50, p=5

```
set.seed(seed)
n <- 50
p <- 4
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "DML")

# Both do great here!

x <- sim.OR(formula_estimand = ~1 + W1 + W2 + W3 + W4, n=n, p=p)
x$data[, -c(1:5)]

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coefficient obtained by causalGLM. These are the
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## [1] 0.802 0.710 0.776 0.756 0.718
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so far: "
## [1] 0.864 0.888 0.910 0.886 0.902

# Both methods do great here!
```

n = 100, p=5

```
set.seed(seed)
n <- 100
p <- 4
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "DML")

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coefficient obtained by causalGLM. These are the
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## [1] 0.848 0.824 0.834 0.808 0.864
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so far: "
## [1] 0.876 0.886 0.900 0.894 0.906

### Similar performance again
```

n = 250, p=5

```
#set.seed(seed)
n <- 1000
p <- 4
#n=50 is sample size
#p=4 is number of covariates in W
```

```
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.
out <- sim.causalGLM(cross_fit = F, formula = ~1 + W1 + W2 + W3 + W4, n=n, p = p, learning_method = "DML")

# The report function summarizes the coverage
# The first row of values is the coverage probability for each coefficient obtained by causalGLM. These are the
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## [1] 0.916 0.920 0.920 0.930 0.934
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so far: "
## [1] 0.888 0.896 0.912 0.932 0.926

### You should see causalGLM do better than the competing estimating equation method.
```

## High dimensional comparison with causalGLMwithLASSO

```
nsims <- 50

set.seed(seed)
n <- 50
p <- 200
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.

bigformula <- ~ 1 + W12 + + W2 + W3 + W112 + W123 + W154 + W45 + W34 + W121 + W63 + W164 + W162 + W63
smallformula <- ~ 1 + W12 + W50 + W99 + W14

out <- sim.causalGLMwithLasso(formula = smallformula, n = n, p = p, formula_A = bigformula, formula_Y = Y)

out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W12      W50      W99      W14
##          0.88      0.96      0.94      0.98      0.92
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so far: "
## [1] 0.88 0.84 0.84 0.90 0.80

### You should see causalGLM do substantially better than the competing estimating equation method.
```

```
set.seed(seed)
n <- 100
p <- 200
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.

bigformula <- ~ 1 + W12 + W2 + W3 + W112 + W123 + W154 + W45 + W34 + W121 + W63 + W164 + W162 + W63
smallformula <- ~ 1 + W12 + W50 + W99 + W14

out <- sim.causalGLMwithLasso(formula = smallformula, n = n, p = p, formula_A = bigformula, formula_Y
```

```

out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W12      W50      W99      W14
##      0.90      0.96      0.90      0.96      0.94
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so
## [1] 0.88 0.92 0.88 0.92 0.90
### You should see causalGLM do substantially better than the competing estimating equation method.

set.seed(seed)
n <- 250
p <- 200
#n=50 is sample size
#p=4 is number of covariates in W
### This will print updates of coverage per iteration to your consol. It should run in a minute or so.

bigformula <- ~ 1 + W12 + + W2 + W3 + W112 + W123 + W154 + W45 + W34 + W121 + W63 + W164 + W162 + W63 +
smallformula <- ~ 1 + W12 + W50 + W99 + W14
# Cross-fitting is turned off for speed
out <- sim.causalGLMwithLasso(formula = smallformula, n = n, p = p, formula_A = bigformula, formula_Y =
out$report()

## [1] "Coverage probability of 95% confidence intervals of causalGLM so far: "
## (Intercept)      W12      W50      W99      W14
##      0.96      0.92      0.94      0.96      0.94
## [1] "Coverage probability of 95% confidence intervals of the estimating equation (DML) competitor so
## [1] 0.96 0.88 0.94 0.92 0.94
### The two methods perform similar, but it looks like causalGLM has a slight edge.

```