Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. doi: 10.18637/jss.v000.i00

causalglm: A tlverse R package for robust generalized linear models and interpretable causal inference for heterogeneous treatment effects using targeted machine-learning

Lars van der Laan

Abstract

Generalized linear models are well-established and interpretable methods for learning heterogeneous treatment effects. However, conventional parametric generalized linear models make strong parametric assumptions on components of the data-generating distribution that are not directly of interest for the causal question at hand. Notably, in real-world settings, the relation between confounders and the outcome is almost always unknown and may be quite complex. As a result, causal inference based on parametric generalized linear models can be very biased and even misleading. Moreover, in high dimensional settings, generalized linear models and their inference breaks down and alternative methods like lasso and elastic-net regression do not easily provide inference. In this article, we present the R package **causalglm** that implements robust semiparametric (*spglm*) and nonparametric (*npglm*) generalized linear models and causal inference for heterogeneous treatment effects in both low and high dimensions. These methods utilize targeted maximum likelihood estimation and therefore can leverage adaptive machine-learning algorithms for estimation and variable selection of nonparametric nuisance components of the data-distribution. The methods allow for the interpretable and familiar coefficient-based inference while significantly relaxing the assumptions needed for correct inference through the use of machine-learning. This package supports semiparametric and nonparametric estimation and inference for user-specified parametric models of the estimands: the conditional average treatment effect, the conditional relative risk, the conditional odds ratio, and much more. The nonparametric methods view the user-specified parametric model as a causal approximation (i.e. working-model) and provide interpretable and correct causal inference even when the parametric model is incorrect (i.e. misspecified). Robust causal inference for marginal structural models based on nonparametric projections onto working models are implemented in the function *msmglm*. A custom lasso-based version of these methods are implemented in the function *causalglmnet*, allowing for robust post-confounder-selection causal inference in high dimensions. These methods are implemented using the powerful **tlverse** machine-learning (**sl3**) and generalized targeted learning (**tmle3**) ecosystem.

Keywords: interpretable-machine-learning, targeted-learning, generalized-linear-models, causal-inference, relative-risk-regression, heterogeneous-treatment-effects, high-dimensional-inference, robust-statistics, semiparametric, nonparametric, double-robust .

1. Introduction to causalglm

1.1. Semiparametric and nonparametric generalized linear models and interpretable causal inference for heterogeneous treatment effects using targeted maximum likelihood estimation

It is possible to get robust and efficient inference for causal quantities using machine-learning. In the search for answers to causal questions, assuming parametric models can be dangerous. With even a seemingly small amount of confounding and misspecificaton, they can give biased answers. One way of mitigating this challenge is to instead assume a parametric model for only the feature of the data-generating distribution that you care about. That is, assume a semiparametric model! Let the data speak for itself and use machine-learning to model the nuisance features of the data that are not directly related to your causal question. Why worry about things that don't matter for your question? It is not worth the risk of being wrong.

In this package, we utilize targeted machine-learning to generalize the parametric generalized linear models commonly used for heterogeneous treatment effect estimation (e.g. the R package glm) to the world of semi and nonparametric models. There is little-to-no loss in precision/p-values/confidence-interval-widths with these semiparametric methods relative to parametric generalized linear models, but the bias reduction from these methods can be substantial. Simulations suggest that these methods can work well with small sample sizes. All methods utilize targeted maximum likelihood estimation (TMLE) (van der Laan, Rose, 2011).

Each estimand considered in this package can be modeled with a user-specified parametric model that is either assumed correct ('spglm' and 'causalglmnet') or as an approximation, i.e. working model, of the nonparametric true estimand ('npglm'). The former approach provides interpretable estimates and correct inference only when the parametric model is correct, and the latter approach provides interpretable estimates and nonparametrically correct inference even when the parametric model is incorrect.

By default, the Highly Adaptive Lasso (HAL) and its semiparametric variants are used for estimation. See Benkeser et al. (2016) for an overview of HAL and its statistical performance. For the theoretical analysis of HAL including its fast rate of convergence in high dimensions, see Bibaut et al. (2019) and van der Laan (2017) . We implement the HAL estimators using the R package hal9001 (Coyle et al., 2021).

Noticable features supported:

- Efficent semiparametric and nonparametric interpretable inference for user-specified parametric working-models of conditional treatment-effect functions with 'spglm' and 'npglm'.
- Efficient nonparametric inference for marginal structural models for a number of conditional treatment-effect estimands with 'npglm'.
- Designed-easy-to-use interface and built-in machine-learning routines for diverse settings and immediate use.
- General machine-learning tools with the tlverse/sl3 generalized machine-learning ecosystem (Coyle et al., 2021).

- Semiparametric inference with high dimensional covariates and adaptive variable selection of confounders with the wrapper function 'causalglmnet'.

All functions give the outputs:

- Coefficient estimates (and their exponential transform for the OR and RR estimands)
- 95% confidence intervals for the coefficients
- Z-scores and p-values
- Estimand point-wise predictions and 95% confidence (prediction) intervals can be extracted with the 'predict' function and argument 'data'.

1.2. spglm: semiparametric causal inference for generalized linear models

This function supports semiparametric efficient estimation of following point-treatment estimands:

- Conditional average treatment effect (CATE). (Causal semiparametric linear regression)
- Conditional odds ratio (OR) between two binary variables. (Causal semiparametric logistic regression)
- Conditional relative risk (RR) for nonnegative outcomes and a binary treatment. (Causal semiparametric log-linear relative-risk regression)

1.3. npglm: nonparametric causal inference for generalized linear models

The function npglm supports nonparametric efficient working-model-based estimation of following point-treatment estimands:

- Conditional average treatment effect (CATE).
- Conditional average treatment effect among the treated (CATT)
- Conditional treatment-specific mean (TSM)
- Conditional odds ratio (OR) between two binary variables.
- Conditional relative risk (RR) for nonnegative outcomes and a binary treatment.

This function also automatically supports marginal structural models by specifying lower dimensional formulas/working-models for the estimands: CATE, CATT, TSM and RR. For ease-of-use, we provide a separate function called "msmglm" for marginal structural model estimation.

1.4. msmglm: nonparametric causal-inference for marginal structural models

This function follows user-specified marginal structural models (based on nonparametric projections onto working models) for the following estimands:

- Marginal structural models for the conditional average treatment effect (CATE).
- Marginal structural models for the conditional average treatment effect among the treated (CATT).
- Marginal structural models for the conditional treatment-specific mean (TSM).
- Marginal structural models for the conditional relative risk (RR).

1.5. causalglmnet: high dimensional causal inference for generalized linear models

The function causalglmnet supports causal inference with high dimensional confounders for the following estimands:

- Conditional average treatment effect (CATE).
- Conditional odds ratio (OR) between two binary variables.
- Conditional relative risk (RR) for nonnegative outcomes and a binary treatment.

Note that the function causalglmnet is a custom wrapper function around the function spglm.

1.6. npcoxph: assumption-lean inference for the conditional hazard ratio

The function npcoxph supports the following survival estimands:

- Nonparametric inference for a user-specified working-model for the conditional hazard ratio between two treatments with 'npcoxph'.
- The estimands supported by 'npcoxph' based on lower dimensional formulas can immediately be interpreted as marginal structural models for the hazard ratio.

2. Data-Structure and treatment-effect estimands

We will mainly consider the point-treatment data-structure $O = (W, A, Y)$ where W represents a vector of baseline covariates (i.e. possible confounders), A is a binary treatment assignment, and Y is some outcome variable. As an example, for a given observation O, W could be measurements: age, sex, a risk-score, location, income; A could take the value 1 if the individual receives the treatment and 0 if they do not receive the treatment; and Y is a binary or continuous variable that captures the effect of the treatment. For the goal of assessing heterogeneity of the treatment effect, there are a number of popular estimands:

The conditional average treatment effect (CATE):

$$CATE(w) := E[Y|A = 1, W = w] - E[Y|A = 0, W = w], \tag{1}$$

which is an additive measure of the effect of the treatment ($A = 1$) relative to no treatment ($A = 0$).

The conditional odds ratio (OR) for when Y is binary:

$$OR(w) := \frac{P(Y=1|A=1,W=w)/P(Y=0|A=1,W=w)}{P(Y=1|A=0,W=w)/P(Y=0|A=0,W=w)} \tag{2}$$

The conditional relative risk (RR) for when Y is nonnegative (e.g. a binary or count variable):

$$RR(w) := \frac{E[Y|A=1,W=w]}{E[Y|A=0,W=w]}, \tag{3}$$

which is a relative measure of the effect of the treatment ($A=1$) relative to no treatment ($A=0$).

In some application, non-contrast measures like the conditional treatment-specific mean (TSM) may be of interest:

$$TSM_a(w) := E[Y|A=a,W=w]. \tag{4}$$

2.1. Conventional estimators using parametric generalized linear models

In order to estimate the estimands of the previous section, parametric generalized linear models are often employed (e.g. the R package *glm*). For the CATE,the following linear regression model is often used:

$$E[Y|A,W=w] = \beta_0 A + \beta_1^T w \cdot A + \beta_2^T w.$$

This model is equivalent to assuming both the nuisance linear model

$$E[Y|A=0,W=w] = \beta_2^T w$$

and target linear model

$$CATE(w) = \beta_0 + \beta_1^T w.$$

Thus, the coefficient in front of the treatment interactions can be directly interpreted as a measure of the conditional average treatment effect when the parametric model is correct. However, we see that very strong parametric assumptions are made on the orthogonal nuisance function $w \mapsto E[Y|A=0,W=w]$, which has little to do with the CATE.

Next, for the conditional odds ratio, the following logistic regression model is often used

$$\text{logit}\left\{P(Y=1|A,W=w)\right\} = \beta_0 A + \beta_1^T w \cdot A + \beta_2^T w.$$

This model is equivalent to assuming both the nuisance logistic model

$$\text{logit}\left\{P(Y=0|A=0,W=w)\right\} = \beta_2^T w$$

and the target log-linear model

$$\log OR(w) = \beta_0 + \beta_1^T w.$$

Once again, we see that the conventional logistic regression model makes strong parametric assumptions on the orthogonal nuisance parameter $P(Y=0|A=0,W=w)$.

Finally, for the conditional relative-risk, the poisson or log-linear regression model is a well-known approach:

$$\log\{E[Y|A, W = w]\} = \beta_0 A + \beta_1^T w \cdot A + \beta_2^T w.$$

This is equivalent to assuming the nuisance log-linear model

$$\log\{E[Y|A = 0, W = w]\} = \beta_2^T w$$

and the target log-linear model

$$\log\{RR(w)\} = \beta_0 + \beta_1^T w.$$

Besides the strong parametric assumptions on the nuisance parameter $E[Y|A = 0, W = w]$, another issue with this approach is that conventional methods only provide inference when the outcome is Poisson distributed, which is not useful if the outcome is binary. Log-link binomial generalized-linear-models are one way to overcome this limitation.

Under the parametric assumptions, standard generalize linear model software can be used to obtain estimates and inference for the coefficients in the above models for the treatment-effect estimands. However, these methods make much stronger assumptions than are needed. In particular, the parametric assumptions on $E[Y|A = 0, W]$ provides little-to-no benefit in interpretability since we are interested in the coefficients for the treatment interaction terms, and it comes at a possibly substantial cost in bias due to model misspecification. Additionally, these methods do not allow for any adaptive estimation of $E[Y|A = 0, W]$ (e.g. using the LASSO, variable selection, or machine-learning) and therefore do not perform well in both estimation and inference in high dimensions. In the next section, we consider a partial relaxation of the parametric models through so-called partially-linear generalized linear models.

3. Semiparametric glms for treatment effect estimation with spglm

In this section, we give an overview of semiparametric treatment-effect estimation in partially-linear generalized linear models which allows for adaptive estimation of nuisance parameters of the data-generating distribution that are not directly relevant for the problem at end. Semiparametric models are statistical models in which one component of the data-generating distribution is parametric and the remaining components are nonparametric. For background on semiparametric models and estimators in causal inference, we refer to Bickel et al. (1993) and van der Laan, Robins (2003).

These methods allow for:

1. Interpretable (coefficient-based) inference for user-specified parametric models for conditional treatment effect estimands.

2. Adaptive machine-learning and variable selection methods including generalized additive models, LASSO, MARS and gradient-boosting can be used to estimate nuisance parameters nonparametrically, thereby substantially relaxing assumptions for valid inference although still assuming a parametric model for the conditional estimand of interest.

3.1. Conditional average treatment effect (CATE) and partially-linear least-squares regression

Let $\underline{f}(w)$ be an arbitrary known vector-valued function of the covariates and consider the linear parametric model $\beta^T \underline{f}(w)$ for $CATE(w)$. The partially-linear least-squares model is of the form:

$$E[Y|A, W = w] = \beta^T \underline{f}(w) \cdot A + h_0(w),$$

where $h_0(w) := E[Y|A = 0, W = w]$ is an unspecified nuisance function that is to be learned from the data nonparametrically. The parametric component of the model is the coefficient vector β. This model is equivalent to *only* assuming:

$$CATE(w) = \beta^T \underline{f}(w).$$

Thus, this semiparametric model only makes assumptions that directly relate the estimand of interest!

A concrete model is choosing $\underline{f}(W) = (1, W)$ which gives the linear model

$$CATE(w) = \beta_0 + \beta_1^T w.$$

Estimates and inference for the coefficient vector in this semiparametric model can be obtained by applying the R function *spglm* with the option 'estimand = "CATE"'. We employ machine-learning for initial estimation of the relevant components of the data-generating distribution and then use targeted maximum likelihood estimation for bias-correction, thereby allowing for valid efficient inference. The estimand is estimated using targeted maximum likelihood estimation and the theory and pseudo-code for the method can be found in the working paper, van der Laan (2009).

3.2. Conditional odds ratio (OR) and partially-linear logistic regression

Let $\underline{f}(w)$ be an arbitrary known vector-valued function of the covariates and consider the parametric model $\beta^T \underline{f}(w)$ for $\log OR(w)$. The partially-linear logistic model is given by:

$$\text{logit}\left\{E[Y|A, W = w]\right\} = \beta^T \underline{f}(w) \cdot A + h_0(w),$$

where $h_0(w) := \text{logit}\left\{E[Y|A = 0, W = w]\right\}$ is an unspecified nuisance function that is to be learned from the data nonparametrically. This model is equivalent to *only* assuming:

$$\log OR(w) = \beta^T \underline{f}(w).$$

Estimates and inference for the coefficient vector in this semiparametric model can be obtained by applying the R function *spglm* with the option 'estimand = "OR"'. The estimand is estimated using targeted maximum likelihood estimation and the theory and pseudo-code for the method can be found in the working paper, van der Laan (2009). For a similar estimator based on estimating equations and additional background on the semiparametric logistic regression model, see Tchetgen Tchetgen (2010).

3.3. Conditional relative-risk (RR) and partially-linear log-linear regression

Let $\underline{f}(w)$ be an arbitrary known vector-valued function of the covariates and consider the parametric model $\beta^T \underline{f}(w)$ for $\log RR(w)$. The partially-linear log-linear model is of the form:

$$\log \{E[Y|A, W = w]\} = \beta^T \underline{f}(w) \cdot A + h_0(w),$$

where $h_0(w) := \log \{E[Y|A = 0, W = w]\}$ is an unspecified nuisance function that is to be learned from the data nonparametrically. This model is equivalent to *only* assuming:

$$\log RR(w) = \beta^T \underline{f}(w).$$

Estimates and inference for the coefficient vector in this semiparametric model can be obtained by applying the R function *spglm* with the option 'estimand = "RR"'. The estimand is estimated using targeted maximum likelihood estimation and the theory and pseudo-code for the method can be found in the working paper, Tuglus et al. (2011).

3.4. spglm in action

Let us generate a mock dataset that has a constant CATE of value 1.

```
> library(causalglm)
> n <- 250
> W <- runif(n, min = -1, max = 1)
> A <- rbinom(n, size = 1, prob = plogis(W))
> Y <- rnorm(n, mean = A + W, sd = 0.3)
> data <- data.frame(W, A, Y)
```

We specify the parametric form of the CATE through the formula argument. In this case, we will use the intercept-only formula which is equivalent to assuming the CATE is constant. The output consists of: a coefficient estimate for the intercept, lower and upper confidence intervals, an asymptotic standard error estimate for the estimator, a Z-score and p-value. The argument 'W' should be a character vector of variable names in data for which to adjust, 'A' should be the name of a treatment variable, and 'Y' should be the name of an outcome variable. We set the argument 'estimand = "CATE"' to estimate the conditional average treatment effect.

```
> formula <- ~1
> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     verbose = FALSE
+   )
> summary(output)
```

```
A causalglm fit object obtained from spglm for the estimand CATE with formula:
CATE(W) = 0.967 * (Intercept)

Coefficient estimates and inference:
   type       param  tmle_est         se     lower    upper  Z_score p_value
1: CATE (Intercept) 0.9668451 0.04063697 0.8871981 1.046492 376.1886       0

>
```

We can also do a much more complex model with higher dimensional treatment effect interactions. The following data distribution has $CATE(w) = 1 + w$.

```
> Y <-
+   rnorm(n,
+     mean = A * W + A + poly(W, degree = 3) + sin(4 * W),
+     sd = 0.4
+   )
> data <- data.frame(W, A, Y)
> formula <- ~ 1 + W
> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand CATE with formula:
CATE(W) = 0.935 * (Intercept) + 1.01 * W

Coefficient estimates and inference:
   type       param  tmle_est         se     lower    upper  Z_score p_value
1: CATE (Intercept) 0.9347039 0.04595963 0.8446247 1.024783 321.5641       0
2: CATE           W 1.0125533 0.07483231 0.8658846 1.159222 213.9433       0

>
```

We can also obtain predictions and confidence intervals thereof at observations.

```
> head(predict(output, data = data))

  (Intercept)          W   CATE(W)        se   CI_left  CI_right   Z-score
1           1 -0.1094781 0.8238515 0.7422615 0.7318398 0.9158632 17.549390
2           1  0.2619617 1.1999541 0.7807333 1.1031734 1.2967348 24.301435
3           1  0.3341857 1.2730847 0.8159637 1.1719368 1.3742326 24.669279
```

```
4           1 -0.6112963 0.3157339 1.0417692 0.1865949 0.4448729  4.792032
5           1  0.5444318 1.4859701 0.9553485 1.3675439 1.6043963 24.593383
6           1  0.5751882 1.5171126 0.9793800 1.3957074 1.6385178 24.492695
    p-value
1 0.000e+00
2 0.000e+00
3 0.000e+00
4 1.651e-06
5 0.000e+00
6 0.000e+00
```

Currently, the nonparametric learning is performed using the partially-linear first-order Highly Adaptive Lasso (HAL) implemented using the R package tlverse/hal9001. HAL is an adaptive piece-wise linear regression spline estimator and the custom implementation performs the risk minimization entirely within the semiparametric model. It is implemented using LASSO regression with the parametric treatment interactions (as specified by the formula argument) unpenalized and a rich penalized spline basis for the nonparametric component of the model. This method performs risk minimization entirely within the semiparametric model. There are a number of other built-in learning options: c("HAL", "SuperLearner", "glm", "glmnet", "gam", "mars", "ranger", "xgboost")

```
> Y <- rnorm(n, mean = A * W + A + W, sd = 0.4)
> data <- data.frame(W, A, Y)
> # generalized additive models:
> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     learning_method = "gam",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand CATE with formula:
CATE(W) = 1.06 * (Intercept) + 1.03 * W

Coefficient estimates and inference:
   type       param tmle_est         se     lower    upper  Z_score p_value
1: CATE (Intercept) 1.058124 0.04950326 0.9610992 1.155148 337.9658       0
2: CATE           W 1.029996 0.08387380 0.8656064 1.194386 194.1687       0

> # multivariate adaptive regression splines:
> output <-
+   spglm(
+     formula,
```

```
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     learning_method = "mars",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand CATE with formula:
CATE(W) = 1.06 * (Intercept) + 1.03 * W

Coefficient estimates and inference:
   type       param tmle_est         se     lower    upper  Z_score p_value
1: CATE (Intercept) 1.055222 0.04986641 0.9574859 1.152959 334.5845       0
2: CATE           W 1.028903 0.08480928 0.8626797 1.195126 191.8231       0

> # gradient-boosting with xgboost: :
> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     learning_method = "xgboost",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand CATE with formula:
CATE(W) = 1.04 * (Intercept) + 1.1 * W

Coefficient estimates and inference:
   type       param tmle_est         se     lower    upper  Z_score p_value
1: CATE (Intercept) 1.039763 0.05161865 0.9385918 1.140933 318.4913       0
2: CATE           W 1.099206 0.08564441 0.9313457 1.267066 202.9317       0
```

The default learning algorithm "HAL" can be customized with the HAL_args_Y0W argument (see the arguments in hal9001 for more description). 'max_degree' = 1 corresponds with estimating the nuisance function $E[Y|A=0,W]$ with an additive model. 'num_knots' specifies for each interaction degree how many variable knot points are used to generate the tensor product interaction basis functions.

```
> HAL_args_Y0W <-
+   list(
+     smoothness_orders = 1,
+     max_degree = 2,
```

```
+     num_knots = c(10, 5, 1)
+   )
> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     learning_method = "HAL",
+     HAL_args_Y0W = HAL_args_Y0W,
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand CATE with formula:
CATE(W) = 1.06 * (Intercept) + 1.03 * W

Coefficient estimates and inference:
   type       param tmle_est         se     lower    upper  Z_score p_value
1: CATE (Intercept) 1.059258 0.04955137 0.9621389 1.156377 337.9994       0
2: CATE           W 1.030605 0.08319826 0.8675399 1.193671 195.8611       0
```

It is also possible to employ custom learners using the tlverse/sl3 framework and the sl3_Learner_Y (to estimate $E[Y|A = 1, W]$ and $E[Y|A = 0, W]$) and sl3_Learner_A (to estimate $P(A = 1|W)$) argument. Take a look at the argument append_interaction_matrix to understand the design matrix that is given as input to the learner sl3_Learner_Y. In particular, it is important to note that sl3_Learner_Y will be sent to Lrnr_glm_semiparametric.

```
> library(sl3)
> lrnr_glmnet <- Lrnr_glmnet$new()
> lrnr_xgboost <- Lrnr_xgboost$new(max_depth = 4)
> lrnr_earth <- Lrnr_earth$new()
> lrnr_stack <-
+   make_learner(Stack, lrnr_glmnet, lrnr_xgboost, lrnr_earth)
> lrnr_cv <- Lrnr_cv$new(lrnr_stack, full_fit = TRUE)
> # A custom superlearner
> lrnr_sl <- make_learner(Pipeline, lrnr_cv, Lrnr_cv_selector$new())
> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     sl3_Learner_A = lrnr_sl,
+     sl3_Learner_Y = lrnr_sl,
+     verbose = FALSE
```

```
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand CATE with formula:
CATE(W) = 1.06 * (Intercept) + 1.04 * W

Coefficient estimates and inference:
   type       param tmle_est         se     lower    upper  Z_score p_value
1: CATE (Intercept) 1.062483 0.04942073 0.9656203 1.159346 339.9249       0
2: CATE           W 1.037883 0.08500852 0.8712689 1.204496 193.0438       0
```

That's all there is to it! spglm also supports the RR and OR estimands which are run in a completely analagous way. Use the option 'estimand = "OR"' to estimate the conditional odds ratio, and use the option 'estimand = "rR"' to estimate the conditional relative risk. Note that the parametric model specified by formula is actually for the log odds ratio and log relative risk (i.e. at the log scale). Thus, the coefficients returned are for the log-transformed OR and RR. We also provide the exponential-transformed coefficients and confidence intervals, which may be more interpretable as measures of the OR and RR.

```
> n <- 250
> W <- runif(n, min = -1, max = 1)
> A <- rbinom(n, size = 1, prob = plogis(W))
> # OR
> Y <- rbinom(n, size = 1, prob = plogis(A + A * W + W + sin(5 * W)))
> data <- data.frame(W, A, Y)
> formula ~ 1 + W

formula ~ 1 + W

> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "OR",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand OR with formula:
log OR(W) = 1.15 * (Intercept) + 1.14 * W

Coefficient estimates and inference:
   type       param tmle_est        se      lower    upper  psi_exp lower_exp
1:   OR (Intercept) 1.149833 0.2993617 0.56309447 1.736571 3.157664  1.756098
2:   OR           W 1.142400 0.5470256 0.07024998 2.214551 3.134283  1.072776
```

```
   upper_exp  Z_score p_value
1:  5.677839 60.73072       0
2:  9.157296 33.02028       0

> # RR
> Y <- rpois(n, lambda = exp(A + A * W + sin(5 * W)))
> data <- data.frame(W, A, Y)
> formula ~ 1 + W

formula ~ 1 + W

> output <-
+   spglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "RR",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from spglm for the estimand RR with formula:
log RR(W) = 0.928 * (Intercept) + 0.849 * W

Coefficient estimates and inference:
   type       param  tmle_est        se     lower    upper  psi_exp lower_exp
1:   RR (Intercept) 0.9283380 0.0978336 0.7365876 1.120088 2.530300  2.088796
2:   RR           W 0.8485581 0.1590271 0.5368707 1.160246 2.336276  1.710645
   upper_exp   Z_score p_value
1:  3.065125 150.03344       0
2:  3.190717  84.36852       0
```

3.5. Risk minimization in semiparametric models and an optional argument

In this section, we consider the general partially-linear generalized linear model

$$\theta(E[Y|A,W]) = A \cdot \beta^T \underline{f}(W) + \theta(E[Y|A=0,W]).$$

The identity link $\theta(x) := x$ corresponds with the partially-linear least-squares regression model, $\theta(x) = \text{logit}(x)$ the partially-linear logistic regression model, and $\theta(x) = \log(x)$ corresponds with the partially-linear log-linear regression model. A critical component of the methods implemented in spglm is the initial estimation of both β and $E[Y|A=0,W]$ using machine-learning. The function spglm supports two different ways of performing this first initial estimation through the boolean argument 'append_interaction_matrix'.

The first approach corresponds with the option 'append_design_matrix = FALSE' and utilizes a two-stage initial estimation approach. Specifically, we do the following:

1. Estimate $E[Y|A=0,W]$ with machine-learning separately by regressing Y on W using only the observations with $A=0$.
2. Estimate β by performing offsetted standard glm regression of Y on $A \cdot \underline{f}(W)$ using as offset the initial estimator of $\theta(E[Y|A=0,W])$ from the previous step and all observations.

This approach is very flexible since any machine-learning algorithm can be used to estimate $E[Y|A=0,W]$. However, one issue with this approach is that, by separating the tasks, the machine-learning is not able to pool across treatment arms and therefore does not fully utilize the simpler form of the regression function. For example, if there are far fewer observations with $A=0$ than $A=1$ then this method may perform badly. This method may also run into issues in finite samples.

The second approach performs the estimation pooled across all treatment arms and corresponds with 'append_design_matrix = TRUE' (which is the default option). We do as follows:

1. Create a design matrix obtained by appending the user-specified treatment interaction design matrix corresponding with the term $A \cdot \underline{f}(W)$ to the baseline variable design matrix for W.
2. Perform the regression of Y onto (A,W) using this design matrix and any machine-learning algorithm. In other words, regress Y on $(A \cdot \underline{f}(W), W)$. This gives an initial estimator of $E[Y|A=0,W]$ that leveraged the smoothness across the treatment arms.
3. Project the resulting estimator of $E[Y|A=1,W]$ onto the semiparametric model by performing the parametric glm regression of the estimator of $E[Y|A=1,W]$ onto $\underline{f}(W)$ using as offset $\theta(E[Y|A=0,W])$. This pseudo-outcome regression step gives an estimate of the coefficient vector β.

This approach has a number of desirable properties. Firstly, additive regression methods like glm, glmnet or gam can perform well since the treatment interaction terms are included in the design matrix. Also, linear estimators like glm and glmnet combined with this method will immedietly respect the semiparametric model without the projection. The projection is needed for non-linear algorithms like gam and gradient-boosting that may not inherently respect the parametric component of the semiparametric model. Additionally using this method, we can construct custom machine-learning algorithms that generate, for instance, a rich penalized spline basis for the W part of the original design matrix and leaves the treatment interaction terms untransformed and unpenalized. Such an algorithm would fully respect the constraints of the semiparametric statistical model throughout the optimization procedure. We note that the default 'learning_method = "HAL"' option for the function spglm does exactly this. A custom Highly Adaptive Lasso estimator is used that performs risk minimization fully within the semiparametric model by using a rich variation-norm-penalized spline basis for the nonparametric component and leaves the parametric component untransformed and unpenalized. This HAL estimator is implemented using the glmnet implementation of LASSO through hal9001.

parametric glms for conditional and marginal structural treatment effects with npgl

In the previous section, we considered semiparametric generalized linear models where the

data distribution component of interest (the estimand) is modeled parametrically and nuisance components are modeled nonparametrically. While this is much more robust than typical parametric methods, the parametric assumption on the estimand of interest can still be quite strong. It is therefore of interest to develop fully nonparametric methods that provide correct and interpretable estimates and inference under no parametric assumptions. To do so, we will still utilize user-specified parametric models for the estimand of interest, however, we will not assume that these parametric models are correct. We will treat these parametric models as interpretable approximations of the true nonparametric estimand. That is, we utilize the parametric model as a "working-model" that is only used to derive an interesting nonparametric estimand.

It turns out that many of these working models have desirable properties. In particular, by specifying parametric models/formulas that depend on $V \subset W$ where V is a subvector of baseline covariates, we can actually learn marginal structural models. Notably, the intercept working model that approximates the true estimand by a constant often corresponds with a marginal causal estimand like the average treatment effect (ATE or ATT), the marginal treatment-specific mean, or the marginal relative risk. For these reasons, these nonparametrics method allow for the estimation of an even more rich class of parameters than the analagous semiparametric methods.

Nonparametric working-model based estimators for the estimands are implemented in the function 'npglm' and 'msmglm' is a wrapper function that focuses on marginal structural model estimation but simply calls 'npglm' behind the scenes. These methods have:

- Interpretable coefficient-based estimates and inference
- Nonparametric and causal estimates and inference even when the parametric model is incorrect
- Many of the estimands for 'npglm' correspond with marginal structural models when lower dimensional working-models are used.

We refer to van der Laan, Robins (2003), Neugebauer, van der Laan (2008), and Robins et al. (1994) for background on marginal structural models. See also the working paper, van der Laan (2009), for more on marginal structural models in the context of TMLE.

4.1. Conditional average treatment effect (CATE) estimation with a linear working-model

To define a nonparametric approximation of the true CATE with a parametric linear-working model, we utilize the least-squares projection. Let $\underline{f}(w)$ be an arbitrary known vector-valued function of the covariates and consider the linear parametric working-model $\beta^T \underline{f}(w)$ for $CATE(w)$.

Define the risk function,

$$R_{CATE}(\beta) = E\left\{CATE(W) - \beta^T \underline{f}(W)\right\}^2$$

Our estimand of interest is given by β^* which is defined as the minimizer of R_{CATE}.

Consider the simple working model $\beta^T \underline{f}(W) := \beta_0 + \beta_1^T W$. The risk function then reduces to

$$R_{CATE}(\beta_0, \beta_1) = E\left\{CATE(W) - \beta_0 - \beta_1^T W\right\}^2$$

which can be viewed as the ordinary least-squares regression of the true estimand $CATE(W)$ onto W. By specifying a lower dimensional working model $\underline{f}(W) := V$ for some $V \subset W$, the risk function further reduces to

$$R_{CATE}(\beta_0, \beta_1) = E\left\{E[CATE(W)|V] - \beta_0 - \beta_1^T V\right\}^2.$$

The risk minimizer is now the least-squares projection of the true marginal structural CATE model $E[CATE(W)|V]$ onto the linear working model. Note if $E[CATE(W)|V] = \beta_0 + \beta_1^T V$, so that the working model is correct, then this estimand can be directly interpreted as a marginal structural CATE function.

If we use the intercept model then the risk function reduces to

$$R_{CATE}(\beta_0) = E\left\{E[CATE(W)] - \beta_0\right\}^2.$$

and the risk minimizer is exactly given by the nonparametric ATE $E[CATE(W)]$! Thus, the intercept model can be used for marginal ATE estimation. Quite remarkably, this estimands based on such least-squares working model projections automatically reduce to marginal structural model parameters when lower dimensional working models are used. No user or developer intervention is needed for this to happen!

These estimators and estimands are inspired by Chambaz et al. (2012) and is based on discussions with Prof. Mark van der Laan.

4.2. Conditional average treatment effect among the treated (CATT) estimation with a linear working-model

We now define an alternative working model for the CATE that focuses on the treatment effect among the treated. Again, let $\underline{f}(w)$ be an arbitrary known vector-valued function of the covariates and consider the linear parametric working-model $\beta^T \underline{f}(w)$ for $CATE(w)$.

Define the risk function,

$$R_{CATT}(\beta) = E\left\{E[Y|A, W] - A \cdot \beta^T \underline{f}(W) - E[Y|A = 0, W]\right\}^2$$

Our estimand of interest is β^* which is defined as the minimizer of R_{CATT}. This working model can be viewed as the least-squares regression of the true conditional mean $E[Y|A, W]$ onto the interaction model $A \cdot \beta^T \underline{f}(W)$ using as offset the true placebo conditional mean $E[Y|A = 0, W]$. It turns out that we can rewrite this risk function as

$$R_{CATT}(\beta) = E\left\{A\left[CATE(W) - \beta^T \underline{f}(W)\right]\right\}^2,$$

which is the least-squares projection of the true CATE onto the linear working model using only the observations with $A = 1$ (the treated). Because of this, we call estimands based on this risk function measures of the conditional average treatment effect among the treated (CATT).

This method can also be used to estimate marginal structural models for treatment effects among the treated. Specifically, by specifying a lower dimensional working model $\underline{f}(W) := V$ for some $V \subset W$, the risk function further reduces to

$$R_{CATT}(\beta_0, \beta_1) = E\left\{A\left[E[CATE(W)|V, A=1] - \beta_0 - \beta_1^T V\right]^2\right\}.$$

The risk minimizer is now the least-squares projection of the true marginal structural CATT model $E[CATE(W)|V, A=1]$ onto the linear working model. Next, if we use the intercept model then the risk function reduces to

$$R_{CATT}(\beta_0) = E\left\{A \cdot [E[CATE(W)|A=1] - \beta_0]^2\right\}.$$

and the risk minimizer is exactly given by the nonparametric ATT $E[CATE(W)|A=1]$! Thus, the intercept model can be used for marginal ATT estimation.

These estimators and estimands are directly due to Chambaz et al. (2012).

4.3. Conditional treatment-specific mean (TSM) estimation with a linear working-model

A similar working model-based estimand can be constructed for the conditional treatment specific mean. Let a be a level of a categorical treatment assignment A. Again, let $\underline{f}(w)$ be an arbitrary known vector-valued function of the covariates and consider the linear parametric working-model $\beta^T \underline{f}(w)$ for $CATE(w)$.

Define the risk function,

$$R_{TSM}(\beta) = E\left\{E[Y|A=a, W] - \beta^T \underline{f}(W)\right\}^2$$

Our estimand of interest is β^* which is defined as the minimizer of R_{TSM}. This estimand corresponds with the least-squares projection of $E[Y|A=a, W=w]$ onto the linear working model.

By specifying a lower dimensional working model $\underline{f}(W) := V$ for some $V \subset W$, the risk function further reduces to

$$R_{TSM}(\beta_0, \beta_1) = E\left\{E[E[Y|A=a, W]|V] - \beta_0 - \beta_1^T V\right\}^2.$$

The risk minimizer is now the least-squares projection of the true marginal structural TSM model E[E[Y|A=a,W]|V] onto the linear working model. If we use the intercept model then the risk function reduces to

$$R_{TSM}(\beta_0) = E\left\{E[E[Y|A=a, W]] - \beta_0\right\}^2.$$

and the risk minimizer is exactly given by the nonparametric TSM $E[E[Y|A=a, W]]$! Thus, the intercept model can be used for marginal TSM estimation.

These estimators and estimands are inspired by Chambaz et al. (2012) and is based on discussions with Prof. Mark van der Laan.

4.4. Conditional odds-ratio (OR) estimation with a logistic working-model

Define the working model

$$P_\beta(Y=1|A, W) := \text{expit}\left\{A \cdot \beta^T \underline{f}(W) + \text{logit}(P(Y=0|A, W))\right\},$$

which is not assumed correct.

Consider the log-likelihood projection risk function

$$R_{OR}(\beta) = E\left\{P(Y=1|A,W)\log(P_\beta(Y=1|A,W)) + P(Y=0|A,W)\log(P_\beta(Y=0|A,W))\right\}.$$

We define the nonparametric OR estimand as the risk minimizer β^* of R_{OR}. This estimand unfortnately does not reduce to a marginal structural model estimand when $\underline{f}(W)$ lower dimensional.

4.5. Conditional relative-risk regression (RR) with a log-linear working-model

Define the log-linear multiplicative working model

$$E_\beta[Y|A=1,W] := \exp\left\{\beta^T \underline{f}(W)\right\} E[Y|A=0,W],$$

which is not assumed correct and $E_\beta[Y|A=0,W] := E[Y|A=0,W]$ is left correctly specified. We define the projection using the log-linear generalized linear model,

$$R_{RR}(\beta) = E\left\{E[Y|A=0,W]\exp\left\{\beta^T \underline{f}(W)\right\} - E[Y|A=1,W]\beta^T \underline{f}(W)\right\}.$$

We define the nonparametric RR estimand as the risk minimizer β^* of R_{RR}.

By specifying a lower dimensional working model $\underline{f}(W) := V$ for some $V \subset W$, the risk function further reduces to

$$R_{RR}(\beta) = E\left\{E[E[Y|A=0,W]|V]\exp\left\{\beta^T V\right\} - E[E[Y|A=1,W]|V]\beta^T V\right\}.$$

The risk minimizer is now the projection of the true marginal structural RR model $\frac{E[E[Y|A=1,W]|V]}{E[E[Y|A=0,W]|V]}$ onto the log-linear working model. Thus, if the working model is correctly specified, the estimand is $\frac{E[E[Y|A=1,W]|V]}{E[E[Y|A=0,W]|V]}$. If we use the intercept model then the risk function reduces to

$$R_{RR}(\beta) = E\left\{E[E[Y|A=0,W]]\exp\left\{\beta\right\} - E[E[Y|A=1,W]]\beta\right\}.$$

and the risk minimizer is exactly given by the nonparametric marginal relative risk $\frac{E[E[Y|A=1,W]]}{E[E[Y|A=0,W]]}$! Thus, the intercept model can be used for marginal RR estimation.

4.6. npglm and msmglm in action

npglm operates in almost exactly the same way as spglm (with a few less optional arguments). The main difference is that it supports new estimands like the CATT and TSM. All previous discussion on learner customization operates in the exact same way but it should be noted that nuisance functions are now estimated nonparametrically (and no longer semiparametrically) so it is important to use learners that include interactions. The following block of the code runs npglm for each supported estimand.

```
> n <- 250
> W <- runif(n, min = -1, max = 1)
> A <- rbinom(n, size = 1, prob = plogis(W))
> # CATE
> Y <- rnorm(n, mean = A + A * W + W + sin(5 * W), sd = 0.5)
> data <- data.frame(W, A, Y)
> formula ~ 1 + W
```

```
formula ~ 1 + W

> # Use the formula_Y argument to specify the design matrix for glm, glmnet or mars (not s
> output <-
+   npglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATE",
+     learning_method = "glm",
+     formula_Y = ~ . + . * A,
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from npglm for the estimand CATE with formula:
CATE(W) = 1.11 * (Intercept) + 0.676 * W

Coefficient estimates and inference:
   type       param  tmle_est        se     lower    upper   Z_score p_value
1: CATE (Intercept) 1.1104671 0.1191302 0.8769763 1.343958 147.38523       0
2: CATE           W 0.6758914 0.1929926 0.2976329 1.054150  55.37405       0

> # CATT
> Y <- rnorm(n, mean = A + A * W + W + sin(5 * W), sd = 0.5)
> data <- data.frame(W, A, Y)
> formula ~ 1 + W

formula ~ 1 + W

> output <-
+   npglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "CATT",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from npglm for the estimand CATT with formula:
CATT(W) = 0.999 * (Intercept) + 1.01 * W

Coefficient estimates and inference:
   type       param tmle_est         se     lower    upper  Z_score p_value
1: CATT (Intercept) 0.998788 0.06598156 0.8694665 1.128109 239.3430       0
2: CATT           W 1.009127 0.10956382 0.7943859 1.223868 145.6293       0
```

```
> # TSM
> Y <- rnorm(n, mean = A + A * W + W + sin(5 * W), sd = 0.5)
> data <- data.frame(W, A, Y)
> formula ~ 1 + W

formula ~ 1 + W

> output <-
+   npglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "TSM",
+     learning_method = "mars",
+     formula_Y = ~ . + . * A,
+     verbose = FALSE
+   )
> # TSM returns a list of causalglm objects for each level
> summary(output[[1]])

A causalglm fit object obtained from npglm for the estimand TSM with formula:
TSM(W) = 0.121 * E[Y_{A=0}]: (Intercept) + 0.733 * E[Y_{A=0}]: W

Coefficient estimates and inference:
   type                   param  tmle_est         se        lower     upper
1:  TSM E[Y_{A=0}]: (Intercept) 0.1209241 0.06455376 -0.005598968 0.2474471
2:  TSM           E[Y_{A=0}]: W 0.7326545 0.11073053  0.515626675 0.9496824
     Z_score p_value
1:  29.61838       0
2: 104.61691       0

> summary(output[[2]])

A causalglm fit object obtained from npglm for the estimand TSM with formula:
TSM(W) = 1.08 * E[Y_{A=1}]: (Intercept) + 1.75 * E[Y_{A=1}]: W

Coefficient estimates and inference:
   type                   param tmle_est         se     lower    upper  Z_score
1:  TSM E[Y_{A=1}]: (Intercept) 1.084399 0.06728185 0.9525289 1.216269 254.8362
2:  TSM           E[Y_{A=1}]: W 1.750770 0.12618478 1.5034525 1.998088 219.3775
   p_value
1:       0
2:       0

> # OR
> Y <- rbinom(n, size = 1, prob = plogis(A + A * W + W + sin(5 * W)))
> data <- data.frame(W, A, Y)
> formula ~ 1 + W
```

```
formula ~ 1 + W

> output <-
+   npglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "OR",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from npglm for the estimand OR with formula:
log OR(W) = 0.887 * (Intercept) + 1.73 * W

Coefficient estimates and inference:
   type       param  tmle_est        se     lower    upper  psi_exp lower_exp
1:   OR (Intercept) 0.8872691 0.3300549 0.2403734 1.534165 2.428489  1.271724
2:   OR           W 1.7292531 0.5564935 0.6385459 2.819960 5.636442  1.893725
   upper_exp  Z_score p_value
1:  4.637451 42.50492       0
2: 16.776183 49.13246       0

> # RR
> Y <- rpois(n, lambda = exp(A + A * W + sin(5 * W)))
> data <- data.frame(W, A, Y)
> formula ~ 1 + W

formula ~ 1 + W

> output <-
+   npglm(
+     formula,
+     data,
+     W = "W", A = "A", Y = "Y",
+     estimand = "RR",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from npglm for the estimand RR with formula:
log RR(W) = 0.915 * (Intercept) + 1.27 * W

Coefficient estimates and inference:
   type       param  tmle_est         se     lower    upper  psi_exp lower_exp
1:   RR (Intercept) 0.9145938 0.09314506 0.7320328 1.097155 2.495761  2.079303
```

```
2:   RR           W 1.2679611 0.23008908 0.8169948 1.718927 3.553600  2.263687
   upper_exp   Z_score p_value
1:  2.995631 155.25243       0
2:  5.578542  87.13245       0
```

```
> head(predict(output, data = data))
```

```
  (Intercept)          W     RR(W)       se   CI_left  CI_right    Z-score
1           1  0.5556887 5.0489758 2.320777 3.7867149  6.731998 11.0314625
2           1 -0.1479096 2.0689684 1.640151 1.6883215  2.535435  7.0089087
3           1  0.9121139 7.9336938 3.428151 5.1870273 12.134792  9.5524569
4           1 -0.7609298 0.9510053 3.320180 0.6301424  1.435249  0.2392327
5           1  0.5936813 5.2981548 2.429914 3.9202024  7.160458 10.8494582
6           1  0.5275654 4.8721054 2.242058 3.6898940  6.433088 11.1673035
     p-value
1 0.0000e+00
2 2.4019e-12
3 0.0000e+00
4 8.1093e-01
5 0.0000e+00
6 0.0000e+00
```

Marginal structural models can be learned with ‘msmglm‘. The only difference with ‘npglm‘ is that the marginal covariate of interest ‘V‘ must be provided. ‘V‘ can be a vector of covariates but plotting is only supported for univariate ‘V‘.

```
> n <- 250
> V <- runif(n, min = -1, max = 1)
> W <- runif(n, min = -1, max = 1)
> A <- rbinom(n, size = 1, prob = plogis(W))
> # CATE
> Y <- rnorm(n, mean = A * (1 + V + 2*V^2) + W + V + sin(5 * W), sd = 0.5)
> data <- data.frame(V,W, A, Y)
> formula_msm = ~ poly(V, degree = 2, raw = TRUE) # A second degree polynomial
> output <-
+   msmglm(
+     formula_msm,
+     data,
+    V = "V",
+     W = c("V","W"), A = "A", Y = "Y",
+     estimand = "CATE",
+     learning_method = "glm",
+     formula_Y = ~ . + . * A,
+     verbose = FALSE
+   )
> summary(output)
```

```
A causalglm fit object obtained from msmglm for the estimand CATE with formula:
E[CATE(W)|V] = 0.706 * (Intercept) + 1.01 * poly(V, degree = 2, raw = TRUE)1 + 2.23 * poly

Coefficient estimates and inference:
   type                              param  tmle_est        se     lower    upper
1: CATE                         (Intercept) 0.7061484 0.1899408 0.3338712 1.078426
2: CATE poly(V, degree = 2, raw = TRUE)1 1.0112121 0.2300606 0.5603016 1.462123
3: CATE poly(V, degree = 2, raw = TRUE)2 2.2263337 0.4814923 1.2826260 3.170041
    Z_score p_value
1: 58.78244       0
2: 69.49762       0
3: 73.10900       0

> plot_msm(output)
> # CATT
> Y <- rnorm(n, mean = A * (1 + V + 2*V^2) + W + V + sin(5 * W), sd = 0.5)
> data <- data.frame(V,W, A, Y)
> formula_msm = ~ poly(V, degree = 2, raw = TRUE)
> output <-
+   msmglm(
+     formula_msm,
+     data,
+     V = "V",
+     W = c("V","W"), A = "A", Y = "Y",
+     estimand = "CATT",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from msmglm for the estimand CATT with formula:
E[CATE(W)|V, A=1] = 1.18 * (Intercept) + 0.969 * poly(V, degree = 2, raw = TRUE)1 + 1.91 *

Coefficient estimates and inference:
   type                              param  tmle_est         se     lower
1: CATT                         (Intercept) 1.1805707 0.09477092 0.9948231
2: CATT poly(V, degree = 2, raw = TRUE)1 0.9686454 0.11483994 0.7435633
3: CATT poly(V, degree = 2, raw = TRUE)2 1.9073548 0.19282243 1.5294298
      upper  Z_score p_value
1: 1.366318 196.9640       0
2: 1.193728 133.3650       0
3: 2.285280 156.4026       0

> # TSM
> Y <- rnorm(n, mean = A * (1 + V + 2*V^2) + W + V, sd = 0.5)
> data <- data.frame(V,W, A, Y)
> formula_msm = ~ poly(V, degree = 2, raw = TRUE)
> output <-
```

```
+   msmglm(
+     formula_msm,
+     data,
+     V = "V",
+     W = c("V","W"), A = "A", Y = "Y",
+     estimand = "TSM",
+     learning_method = "mars",
+     formula_Y = ~ . + . * A,
+     verbose = FALSE
+   )
> summary(output[[1]])

A causalglm fit object obtained from msmglm for the estimand TSM with formula:
E[TSM(W)|V] = -0.258 * E[Y_{A=0}]: (Intercept) + 1.08 * E[Y_{A=0}]: poly(V, degree = 2, ra

Coefficient estimates and inference:
   type                                          param   tmle_est         se
1:  TSM                       E[Y_{A=0}]: (Intercept) -0.2583340 0.08694949
2:  TSM E[Y_{A=0}]: poly(V, degree = 2, raw = TRUE)1  1.0825909 0.09425945
3:  TSM E[Y_{A=0}]: poly(V, degree = 2, raw = TRUE)2  0.5343227 0.17913666
        lower       upper   Z_score p_value
1: -0.4287519 -0.08791617  46.97693       0
2:  0.8978458  1.26733605 181.59734       0
3:  0.1832213  0.88542415  47.16167       0

> summary(output[[2]])

A causalglm fit object obtained from msmglm for the estimand TSM with formula:
E[TSM(W)|V] = 1.07 * E[Y_{A=1}]: (Intercept) + 1.87 * E[Y_{A=1}]: poly(V, degree = 2, raw

Coefficient estimates and inference:
   type                                          param tmle_est         se
1:  TSM                       E[Y_{A=1}]: (Intercept) 1.068007 0.08983411
2:  TSM E[Y_{A=1}]: poly(V, degree = 2, raw = TRUE)1 1.874266 0.10648762
3:  TSM E[Y_{A=1}]: poly(V, degree = 2, raw = TRUE)2 1.832246 0.20181490
       lower    upper  Z_score p_value
1: 0.8919354 1.244079 187.9762       0
2: 1.6655537 2.082978 278.2928       0
3: 1.4366961 2.227796 143.5491       0

>  plot_msm(output[[1]])
> # RR
> Y <- rpois(n, lambda = exp( A * (1 + V + 2*V^2)  + sin(5 * W)))
> data <- data.frame(V,W, A, Y)
> formula_msm = ~ poly(V, degree = 2, raw = TRUE)
> output <-
+   msmglm(
```

```
+     formula_msm,
+     data,
+     V = "V",
+     W = c("V","W"), A = "A", Y = "Y",
+     estimand = "RR",
+     verbose = FALSE
+   )
> summary(output)

A causalglm fit object obtained from msmglm for the estimand RR with formula:
log E[RR(W)|V] = 0.979 * (Intercept) + 0.918 * poly(V, degree = 2, raw = TRUE)1 + 1.94 * p

Coefficient estimates and inference:
   type                             param  tmle_est        se     lower    upper
1:   RR                       (Intercept) 0.9791400 0.1306146 0.7231401 1.235140
2:   RR poly(V, degree = 2, raw = TRUE)1 0.9184511 0.1195502 0.6841370 1.152765
3:   RR poly(V, degree = 2, raw = TRUE)2 1.9394218 0.1871540 1.5726068 2.306237
    psi_exp lower_exp upper_exp  Z_score p_value
1: 2.662166  2.060894  3.438860 118.5286       0
2: 2.505407  1.982060  3.166938 121.4718       0
3: 6.954728  4.819194 10.036584 163.8488       0

> plot_msm(output)
```

5. High dimensional semiparametric glms using the LASSO with causalglmnet

In high dimensional settings (e.g. dim(W) >= 50-1000), conventional machine-learning algorithms may be computationally expensive or poorly behaved. In such scenarios, we can utilize lasso-penalized regression (Tibshirani, 1994) to estimate the nuisance parameters, allowing for adaptive variable selection and adjusting of confounders. The function causalglmnet is a specialized wrapper for spglm that uses the lasso implementation provided by the state-of-the-art R package glmnet (Friedman, 2010) for estimation of all nuisance parameters. Its use is exactly the same as spglm except learners no longer need to be specified.

```
> n <- 200
> W <- replicate(100, runif(n, min = -1, max = 1))
> colnames(W) <- paste0("W", 1:100)
> beta <- runif(10, -1, 1) / 20
> A <- rbinom(n, size = 1, prob = plogis(W[, 10 * (1:10)] %*% beta))
> # CATE
> Y <- rnorm(n, mean = A + W[, 10 * (1:10)] %*% beta, sd = 0.5)
> data <- data.frame(W, A, Y)
> formula <- ~1
> output <-
+   causalglmnet(
```

```
+     formula,
+     data,
+     W = colnames(W), A = "A", Y = "Y",
+     estimand = "CATE",
+     verbose = FALSE
+   )
> summary(output)
> # OR
> Y <- rbinom(n, size = 1, prob = plogis(A + W[, 10 * (1:10)] %*% beta))
> data <- data.frame(W, A, Y)
> formula <- ~1
> output <-
+   causalglmnet(
+     formula,
+     data,
+     W = colnames(W), A = "A", Y = "Y",
+     estimand = "OR",
+     verbose = FALSE
+   )
> summary(output)
> # RR
> Y <- rpois(n, lambda = exp(A + W[, 10 * (1:10)] %*% beta))
> data <- data.frame(W, A, Y)
> formula <- ~1
> output <-
+   causalglmnet(
+     formula,
+     data,
+     W = colnames(W), A = "A", Y = "Y",
+     estimand = "RR",
+     verbose = FALSE
+   )
> summary(output)
> head(predict(output, data = data))
>
```

6. Robust nonparametric inference for the hazard ratio with npcoxph

This is in development.

References

Benkeser D, van der Laan M (2016). "The Highly Adaptive Lasso Estimator." *International Conference on Data Science and Advanced Analytics*, pp. 689–696. doi:10.1109/DSAA.2016.93.

Bibaut AF, van der Laan MJ (2019). "Fast rates for empirical risk minimization over càdlàg functions with bounded sectional variation norm." 1907.09244.

Bickel PJ, Klaassen CA, Ritov Y, Wellner J (1993). *Efficient and adaptive estimation for semiparametric models*, volume 4. Johns Hopkins University Press Baltimore.

Chambaz A, Neuvial P, van der Laan MJ (2012). "Estimation of a non-parametric variable importance measure of a continuous exposure." *Electronic Journal of Statistics*, **6**(none), 1059 – 1099. doi:10.1214/12-EJS703. URL https://doi.org/10.1214/12-EJS703.

Coyle JR, Hejazi NS, Malenica I, Sofrygin O (2021a). *sl3: Modern Pipelines for Machine Learning and Super Learning*. doi:10.5281/zenodo.1342293. R package version 1.4.2, URL https://doi.org/10.5281/zenodo.1342293.

Coyle JR, Hejazi NS, van der Laan MJ (2021b). *hal9001: The scalable highly adaptive lasso*. doi:10.5281/zenodo.3558313. R package version 0.2.7, URL https://github.com/tlverse/hal9001.

Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, **33**(1), 1–22. URL https://www.jstatsoft.org/v33/i01/.

Hejazi NS, Coyle JR, van der Laan MJ (2020). "hal9001: Scalable highly adaptive lasso regression in R." *Journal of Open Source Software*. doi:10.21105/joss.02526. URL https://doi.org/10.21105/joss.02526.

Laan MJVD, Rubin D (2006). "Targeted Maximum Likelihood Learning." *The International Journal of Biostatistics*, **2**(1). doi:10.2202/1557-4679.1043.

Neugebauer R, van der Laan M (2007). "Nonparametric causal effects based on marginal structural models." *Journal of Statistical Planning and Inference*, **137**(2), 419–434. ISSN 0378-3758. doi:https://doi.org/10.1016/j.jspi.2005.12.008. URL https://www.sciencedirect.com/science/article/pii/S0378375806000334.

Robins JM, Rotnitzky A, Zhao LP (1994). "Marginal Structural Models and Causal Inference in Epidemiology." *Journal of the American statistical Association*, **89**(427), 846–866.

Tchetgen Tchetgen E, Robins J, Rotnitzky A (2010). "On doubly robust estimation in a semiparametric odds ratio model." *Biometrika*, **97**, 171–180.

Tibshirani R (1994). "Regression Shrinkage and Selection Via the Lasso." *Journal of the Royal Statistical Society, Series B*, **58**, 267–288.

Tuglus C, Porter K, van der Laan M (2011). "Targeted Maximum Likelihood Estimation of Conditional Relative Risk in a Semi-parametric Regression Model." *U.C. BERKELEY DIVISION OF BIOSTATISTICS WORKING PAPER SERIES*, **Working paper 283**.

van der Laan M (2009). "Readings in Targeted Maximum Likelihood Estimation." *Biostatistics Working Paper Series Working Paper 253*, pp. 621–622,626–629.

van der Laan M (2017). "A Generally Efficient Targeted Minimum Loss Based Estimator based on the Highly Adaptive Lasso." *The international journal of biostatistics*, **13**(2). doi:/10.1515/ijb-2015-0097. URL https://doi.org/10.1515/ijb-2015-0097.

van der Laan MJ, Polley EC, Hubbard AE (2007). "Super learner." *Statistical Applications in Genetics and Molecular Biology*, **6**, number 1.

van der Laan MJ, Robins JM (2003). *Unified Methods for Censored Longitudinal Data and Causality*. Springer.

van der Laan MJ, Rose S (2011). *Targeted Learning: Causal Inference for Observational and Experimental Data*. Springer, New York.

Affiliation:

Lars van der Laan
Department of Statistics
University of Washington, Seattle
E-mail: vanderlaanlars@yahoo.com

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd