



---

# *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. 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. Robust causal inference for marginal structural models based on nonparametric projections onto such working models are implemented in the function *msmgglm*. A custom lasso-based version of *spglm* is 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 misspecification, 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:

- Efficient 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 “msmgglm” for marginal structural model estimation.

## 1.4. **msmgglm: 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):

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

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:

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

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

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

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:

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

## 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

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

This model is equivalent to assuming both the nuisance logistic model

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

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  $\beta$ . 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:

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

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

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

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:  
 $\text{CATE}(W) = 1.02 * (\text{Intercept})$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE	(Intercept)	1.023336	0.03977735	0.9453736	1.101298	406.7732	0

>

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

```
> 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:  
 $\text{CATE}(W) = 1.07 * (\text{Intercept}) + 1.07 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE	(Intercept)	1.072966	0.04803212	0.9788248	1.167107	353.2029	0
2:	CATE	W	1.066057	0.08714895	0.8952481	1.236866	193.4141	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	-0.3200201	0.7318065	0.9323055	0.61623667	0.8473762	12.411035
2	1	-0.3614329	0.6876580	0.9671893	0.56776401	0.8075521	11.241675
3	1	0.7423726	1.8643775	1.1814966	1.71791769	2.0108374	24.950048

```

4          1 -0.7733488 0.2485322 1.3960979 0.07547016 0.4215943 2.814731
5          1 -0.6234698 0.4083118 1.2270695 0.25620270 0.5604209 5.261297
6          1  0.6229954 1.7371146 1.0597156 1.60575094 1.8684783 25.918459
      p-value
1 0.0000e+00
2 0.0000e+00
3 0.0000e+00
4 4.8818e-03
5 1.4304e-07
6 0.0000e+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:  
 $\text{CATE}(W) = 1.01 * (\text{Intercept}) + 0.971 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE (Intercept)		1.0094527	0.05524539	0.9011738	1.117732	288.9083	0
2:	CATE	W	0.9713103	0.11353205	0.7487916	1.193829	135.2725	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:  
 $\text{CATE}(W) = 1 * (\text{Intercept}) + 0.993 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE (Intercept)		1.0043078	0.05642633	0.8937142	1.114901	281.4200	0
2:	CATE	W	0.9931916	0.10541455	0.7865829	1.199800	148.9713	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:  
 $\text{CATE}(W) = 1.03 * (\text{Intercept}) + 0.955 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE (Intercept)		1.0300669	0.06473633	0.9031860	1.156948	251.5865	0
2:	CATE	W	0.9551231	0.12538508	0.7093729	1.200873	120.4435	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_YOW = HAL_args_YOW,
+     verbose = FALSE
+   )
> summary(output)

```

A causalglm fit object obtained from spglm for the estimand CATE with formula:  
 $\text{CATE}(W) = 1.01 * (\text{Intercept}) + 0.965 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE (Intercept)		1.0113548	0.05521624	0.903133	1.119577	289.6054	0
2:	CATE	W	0.9647065	0.11313498	0.742966	1.186447	134.8243	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:  
 $\text{CATE}(W) = 0.999 * (\text{Intercept}) + 0.965 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE (Intercept)	0.9994168	0.05722684	0.8872543	1.111579	276.1321		0
2:	CATE	W	0.9652990	0.10999462	0.7497136	1.180885	138.7588	0

That's all there is to it! spglm also supports the RR and OR estimands which are run in a completely analogous 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:  
 $\log \text{OR}(W) = 0.756 * (\text{Intercept}) + 1.85 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	psi_exp	lower_exp
1:	OR (Intercept)	0.7564983	0.3033517	0.1619399	1.351057	2.130802	1.175790	
2:	OR	W	1.8526012	0.5574324	0.7600537	2.945149	6.376384	2.138391

```

      upper_exp  Z_score p_value
1:  3.861504 39.43043      0
2: 19.013487 52.54843      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:  
 $\log \text{RR}(W) = 1.11 * (\text{Intercept}) + 1.23 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	psi_exp	lower_exp
1:	RR (Intercept)		1.109086	0.1075168	0.8983564	1.319815	3.031585	2.455564
2:	RR	W	1.230396	0.1800059	0.8775913	1.583201	3.422586	2.405100

  

	upper_exp	Z_score	p_value
1:	3.742727	163.1018	0
2:	4.870524	108.0758	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  $\beta$  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  $\beta$  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  $\beta$ .

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 immediately 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 npglm

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 'msmgglm' 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,

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

Our estimand of interest is given by  $\beta^*$  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

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

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

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

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

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

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,

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

Our estimand of interest is  $\beta^*$  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

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

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

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

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  $\beta^*$  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

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

We define the nonparametric OR estimand as the risk minimizer  $\beta^*$  of  $R_{OR}$ . This estimand unfortunately 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

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

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,

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

We define the nonparametric RR estimand as the risk minimizer  $\beta^*$  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

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

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

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

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 msglm 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:  
 $\text{CATE}(W) = 1.01 * (\text{Intercept}) + 1.03 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATE (Intercept)		1.005377	0.1073578	0.7949597	1.215794	148.06945	0
2:	CATE	W	1.028498	0.2016641	0.6332435	1.423752	80.63894	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:  
 $\text{CATT}(W) = 1.02 * (\text{Intercept}) + 1.11 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score	p_value
1:	CATT (Intercept)		1.024894	0.06111499	0.9051107	1.144677	265.1558	0
2:	CATT	W	1.112873	0.12519085	0.8675034	1.358243	140.5539	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:  
 $TSM(W) = 0.101 * E[Y_{\{A=0\}}]: (Intercept) + 0.818 * E[Y_{\{A=0\}}]: W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper
1:	TSM	$E[Y_{\{A=0\}}]: (Intercept)$	0.1011973	0.06090594	-0.01817619	0.2205707
2:	TSM	$E[Y_{\{A=0\}}]: W$	0.8180376	0.11906683	0.58467095	1.0514043
		Z_score	p_value			
1:		26.27115	0			
2:		108.63068	0			

```
> summary(output[[2]])
```

A causalglm fit object obtained from npglm for the estimand TSM with formula:  
 $TSM(W) = 0.973 * E[Y_{\{A=1\}}]: (Intercept) + 1.84 * E[Y_{\{A=1\}}]: W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	Z_score
1:	TSM	$E[Y_{\{A=1\}}]: (Intercept)$	0.9734988	0.0607090	0.8545113	1.092486	253.5434
2:	TSM	$E[Y_{\{A=1\}}]: W$	1.8437310	0.1098418	1.6284450	2.059017	265.3993
		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:  
 $\log \text{OR}(W) = 1.68 * (\text{Intercept}) + 2.12 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	psi_exp	lower_exp
1:	OR	(Intercept)	1.677225	0.4241868	0.8458345	2.508616	5.350690	2.329921
2:	OR	W	2.124445	0.6383311	0.8733387	3.375550	8.368248	2.394893

  

	upper_exp	Z_score	p_value
1:	12.28792	62.51788	0
2:	29.24038	52.62225	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:  
 $\log \text{RR}(W) = 0.905 * (\text{Intercept}) + 0.855 * W$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper	psi_exp	lower_exp
1:	RR	(Intercept)	0.9054026	0.08250482	0.7436962	1.067109	2.472927	2.103697

```

2:   RR           W 0.8550550 0.17395465 0.5141102 1.196000 2.351504 1.672150
   upper_exp   Z_score p_value
1:  2.906964 173.51317      0
2:  3.306862  77.71915      0

```

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

```

(Intercept)      W   RR(W)      se CI_left CI_right  Z-score
1           1 -0.03470096 2.400630 1.300707 2.043153 2.820654 10.645390
2           1 -0.32953613 1.865690 1.530534 1.543271 2.255469  6.442506
3           1 -0.67719976 1.385914 2.190811 1.056312 1.818361  2.355382
4           1 -0.71359406 1.343449 2.272138 1.013676 1.780506  2.054524
5           1  0.05899105 2.600863 1.326817 2.206416 3.065826 11.390575
6           1  0.38319001 3.431684 1.738605 2.766357 4.257027 11.213729
      p-value
1 0.0000e+00
2 1.1752e-10
3 1.8504e-02
4 3.9925e-02
5 0.0000e+00
6 0.0000e+00

```

Marginal structural models can be learned with ‘msmgglm’. 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 msglm for the estimand CATE with formula:

$E[CATE(W)|V] = 1.22 * (\text{Intercept}) + 1.03 * \text{poly}(V, \text{degree} = 2, \text{raw} = \text{TRUE})1 + 1.89 * \text{poly}($

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper
1: CATE		(Intercept)	1.215806	0.1644811	0.8934289	1.538183
2: CATE poly(V, degree = 2, raw = TRUE)1			1.029166	0.2341862	0.5701699	1.488163
3: CATE poly(V, degree = 2, raw = TRUE)2			1.885597	0.4325959	1.0377247	2.733469

  

	Z_score	p_value
1: 116.87410	0	
2: 69.48551	0	
3: 68.91861	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 <-
+   msglm(
+     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 msglm for the estimand CATT with formula:

$E[CATE(W)|V, A=1] = 1.24 * (\text{Intercept}) + 0.916 * \text{poly}(V, \text{degree} = 2, \text{raw} = \text{TRUE})1 + 1.63 * \text{poly}($

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper
1: CATT		(Intercept)	1.236324	0.0868406	1.0661194	1.406528
2: CATT poly(V, degree = 2, raw = TRUE)1			0.916062	0.1240648	0.6728994	1.159225
3: CATT poly(V, degree = 2, raw = TRUE)2			1.630790	0.2008528	1.2371256	2.024454

  

	Z_score	p_value
1: 225.1020	0	
2: 116.7471	0	
3: 128.3778	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:

$E[TSM(W)|V] = -0.015 * E[Y_{\{A=0\}}]: (Intercept) + 1.02 * E[Y_{\{A=0\}}]: poly(V, degree = 2, raw = TRUE)$

Coefficient estimates and inference:

	type	param	tmle_est	se
1:	TSM	$E[Y_{\{A=0\}}]: (Intercept)$	-0.01501465	0.0896871
2:	TSM	$E[Y_{\{A=0\}}]: poly(V, degree = 2, raw = TRUE)1$	1.01614922	0.1139134
3:	TSM	$E[Y_{\{A=0\}}]: poly(V, degree = 2, raw = TRUE)2$	0.10243993	0.1975088

  

	lower	upper	Z_score	p_value
1:	-0.1907981	0.1607688	2.647008	8.1207e-03
2:	0.7928830	1.2394155	141.043314	0.0000e+00
3:	-0.2846702	0.4895500	8.200736	2.2204e-16

```

> summary(output[[2]])

```

A causalglm fit object obtained from msmglm for the estimand TSM with formula:

$E[TSM(W)|V] = 0.921 * E[Y_{\{A=1\}}]: (Intercept) + 2.04 * E[Y_{\{A=1\}}]: poly(V, degree = 2, raw = TRUE)$

Coefficient estimates and inference:

	type	param	tmle_est	se
1:	TSM	$E[Y_{\{A=1\}}]: (Intercept)$	0.9211587	0.08289779
2:	TSM	$E[Y_{\{A=1\}}]: poly(V, degree = 2, raw = TRUE)1$	2.0445230	0.11080504
3:	TSM	$E[Y_{\{A=1\}}]: poly(V, degree = 2, raw = TRUE)2$	2.0829169	0.20326867

  

	lower	upper	Z_score	p_value
1:	0.7586821	1.083635	175.6959	0
2:	1.8273491	2.261697	291.7444	0
3:	1.6845177	2.481316	162.0211	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:

$\log E[RR(W)|V] = 0.991 * (\text{Intercept}) + 1.08 * \text{poly}(V, \text{degree} = 2, \text{raw} = \text{TRUE})1 + 2.36 * \text{poly}(V, \text{degree} = 2, \text{raw} = \text{TRUE})2$

Coefficient estimates and inference:

	type	param	tmle_est	se	lower	upper
1:	RR	(Intercept)	0.9912139	0.1801569	0.6381129	1.344315
2:	RR poly(V, degree = 2, raw = TRUE)1		1.0849137	0.1634155	0.7646252	1.405202
3:	RR poly(V, degree = 2, raw = TRUE)2		2.3600278	0.3942551	1.5873020	3.132754
	psi_exp	lower_exp	upper_exp	Z_score	p_value	
1:	2.694503	1.892905	3.835558	86.99344	0	
2:	2.959184	2.148189	4.076351	104.97163	0	
3:	10.591245	4.890536	22.937050	94.64764	0	

```

> plot_msm(output)

```

## 5. High dimensional semiparametric glms using the LASSO with causalglmnet

In high dimensional settings (e.g.  $\dim(W) \geq 50-1000$ ), 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](mailto:vanderlaanlars@yahoo.com)