

# Package: swaglm (via r-universe)

May 17, 2026

**Title** Fast Sparse Wrapper Algorithm for Generalized Linear Models and Testing Procedures for Network of Highly Predictive Variables

**Version** 0.0.1

**Description** Provides a fast implementation of the SWAG algorithm for Generalized Linear Models which allows to perform a meta-learning procedure that combines screening and wrapper methods to find a set of extremely low-dimensional attribute combinations. The package then performs test on the network of selected models to identify the variables that are highly predictive by using entropy-based network measures.

**License** AGPL-3

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**LinkingTo** Rcpp, RcppArmadillo

**Imports** Rcpp, fastglm, stats, igraph, gdata, plyr, progress, DescTools, scales, fields

**Suggests** knitr, MASS, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Lionel Voirol [aut, cre] (ORCID: <https://orcid.org/0000-0003-1696-1407>), Yagmur Ozdemir [aut]

**Maintainer** Lionel Voirol <lionelvoiroi@hotmail.com>

**Config/pak/sysreqs** cmake libglpk-dev make libuv1-dev libxml2-dev libssl-dev libx11-dev zlib1g-dev

**Repository** <https://lionelvoiroi.r-universe.dev>

**Date/Publication** 2025-09-18 08:50:02 UTC

**RemoteUrl** <https://github.com/cran/swaglm>

**RemoteRef** HEAD

**RemoteSha** 3a4e82447856aec0426377a487bb2f05eed2eeb9

## Contents

compute_network . . . . .	2
plot.swaglm_network . . . . .	3
predict.swaglm . . . . .	4
print.swaglm . . . . .	7
print.swaglm_test . . . . .	8
summary.swaglm . . . . .	9
swaglm . . . . .	10
swaglm_test . . . . .	12

<b>Index</b>	<b>14</b>
--------------	-----------

---

compute_network	<i>compute_network</i>
-----------------	------------------------

---

### Description

Compute a network representation of the selected models from a swaglm object

### Usage

```
compute_network(x, mode = "undirected")
```

### Arguments

x	An object of class swaglm.
mode	Character string specifying the mode of the network. Default is "undirected".

### Value

A list of class swaglm\_network.

### Examples

```
set.seed(12345)
n <- 2000
p <- 100

# create design matrix and vector of coefficients
Sigma <- diag(rep(1/p, p))
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)
beta = c(-15,-10,5,10,15, rep(0,p-5))

# ----- generate from logistic regression with an intercept of one
z <- 1 + X%*%beta
pr <- 1/(1 + exp(-z))
y <- as.factor(rbinom(n, 1, pr))
y = as.numeric(y)-1
```

```
# define swag parameters
quantile_alpha = .15
p_max = 20
swag_obj = swaglm::swaglm(X=X, y = y, p_max = p_max, family = stats::binomial(),
alpha = quantile_alpha, verbose = TRUE, seed = 123)
names(swag_obj)
swag_network = swaglm::compute_network(swag_obj)
names(swag_network)
```

---

```
plot.swaglm_network  plot.swaglm_network
```

---

## Description

Visualizes a SWAG network with discretized vertex size, optional edge width scaling, and edge coloring based on a correlation matrix.

## Usage

```
## S3 method for class 'swaglm_network'
plot(x, bins = 5, scale_edge = NULL, size_range = c(8, 30), ...)
```

## Arguments

x	An object of class swaglm_network
bins	Number of bins for vertex size discretization (default = 5)
scale_edge	Logical; whether to scale the edge width
size_range	A numeric vector of length 2 specifying the range of node sizes.
...	Additional arguments passed to igraph::plot

## Value

None.

## Examples

```
set.seed(12345)
n <- 2000
p <- 100

# create design matrix and vector of coefficients
Sigma <- diag(rep(1/p, p))
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)
beta = c(-15,-10,5,10,15, rep(0,p-5))

# ----- generate from logistic regression with an intercept of one
z <- 1 + X*%beta
pr <- 1/(1 + exp(-z))
```

```

y <- as.factor(rbinom(n, 1, pr))
y = as.numeric(y)-1

# define swag parameters
quantile_alpha = .15
p_max = 20
swag_obj = swaglm::swaglm(X=X, y = y, p_max = p_max, family = stats::binomial(),
                          alpha = quantile_alpha, verbose = TRUE, seed = 123)

names(swag_obj)
swag_network = swaglm::compute_network(swag_obj)
names(swag_network)
plot(swag_network)

```

---

<code>predict.swaglm</code>	<i>predict.swaglm</i>
-----------------------------	-----------------------

---

## Description

Predict method for a `swaglm` object

## Usage

```

## S3 method for class 'swaglm'
predict(object, newdata, ...)

```

## Arguments

<code>object</code>	An object of class <code>swaglm</code> returned by <a href="#">swaglm</a> .
<code>newdata</code>	A <code>data.frame</code> or matrix containing the same predictors (columns) as used in the original training dataset.
<code>...</code>	Further arguments passed to or from other methods.

## Details

Computes predictions from a fitted `swaglm` object on new data. The function returns the linear predictors ( $\eta$ ) for each selected model and the corresponding predicted responses using the model's inverse link function.

The function performs the following steps:

- Checks that `newdata` has the same number of columns as the design matrix used in the fitted `swaglm` object.
- Computes the linear predictors ( $\eta = X\hat{\beta}$ ) for each selected model in the `swaglm` object.
- Applies the inverse of the model's link function to compute predicted responses.

**Value**

A list with two elements:

**mat\_eta\_prediction** A numeric matrix containing the linear predictors ( $\eta = X\hat{\beta}$ ) for each observation in `newdata` for all selected models. Each row corresponds to an observation in `newdata`, and each column corresponds to one of the selected models.

**mat\_response\_prediction** A numeric matrix containing the predicted responses, obtained by applying the inverse link function to the elements in `mat_eta_prediction`. The dimensions are identical to `mat_eta_prediction`, so that for each observation (row) the predicted responses for all selected models are aligned per column.

**Examples**

```
set.seed(12345)
n <- 2000
p <- 100
# create design matrix and vector of coefficients
Sigma <- diag(rep(1/p, p))
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)
beta = c(10, rep(0,p-1))
sigma2=2
y <- 1 + X%*%beta + rnorm(n, mean = 0, sd = sqrt(sigma2))

# subset data
n_data_train = n-5
X_sub = X[1:n_data_train, ]
y_sub = y[1:n_data_train]

# plot train data
plot(X_sub[,1], y_sub)
abline(a=1, b=beta[1])

# run swag
swaglm_obj = swaglm(X = X_sub, y = y_sub, p_max = 15, family = gaussian(),
                    method = 0, alpha = .15, verbose = TRUE)

# compute prediction
X_to_predict = X[(n_data_train+1):(dim(X)[1]), ]
y_pred = predict(swaglm_obj, newdata = X_to_predict)
n_selected_model = dim(y_pred$mat_eta_prediction)[2]

# in that case mat_eta_prediction and mat_reponse_prediction are the same
all.equal(y_pred$mat_eta_prediction, y_pred$mat_reponse_prediction)

# compute average prediction (accross selected models)
y_pred_mean = apply(y_pred$mat_reponse_prediction, MARGIN = 1, FUN = mean)

# plot
y_to_predict = y[(n_data_train+1):(dim(X)[1])]
plot(X_to_predict[,1], y_to_predict, col="red", ylim=c(-4,4), ylab="y", xlab="X")
# add all prediction
```

```

for(i in seq(dim(X_to_predict)[1])){
  # i=1
  points(x = rep(X_to_predict[i,1],n_selected_model ),
         y = y_pred$mat_reponse_prediction[i,])
}
points(x =X_to_predict[,1], y = y_pred_mean, col="orange")
abline(a=1, b=beta[1])
legend(
  "topleft",
  legend = c(
    "True values (test set)",
    "Predictions (all selected models)",
    "Average prediction",
    "True regression line"
  ),
  col = c("red", "black", "orange", "black"),
  pch = c(1, 1, 1, NA),
  lty = c(NA, NA, NA, 1),
  cex = 0.8,
  bty="n"
)

# ----- logistic regression

set.seed(12345)
n <- 2000
p <- 100
# create design matrix and vector of coefficients
Sigma <- diag(rep(1/p, p))
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)
beta = c(-15,-10,5,10,15, rep(0,p-5))

# ----- generate from logistic regression with an intercept of one
z <- 1 + X%*%beta
pr <- 1/(1 + exp(-z))
y <- as.factor(rbinom(n, 1, pr))
y = as.numeric(y)-1

# subset data
n_data_train = n-200
X_sub = X[1:n_data_train, ]
y_sub = y[1:n_data_train]

swaglm_obj = swaglm::swaglm(X=X_sub, y = y_sub, p_max = 20, family = stats::binomial(),
                           alpha = .15, verbose = TRUE, seed = 123)

# compute prediction
X_to_predict = X[(n_data_train+1):(dim(X)[1]), ]
y_pred = predict(swaglm_obj, newdata = X_to_predict)
y_pred_majority_class <- ifelse(rowMeans(y_pred$mat_reponse_prediction) >= 0.5, 1, 0)
y_to_predict = y[(n_data_train+1):(dim(X)[1])]

```

```
# tabulate  
table(True = y_to_predict, Predicted = y_pred_majority_class)
```

---

`print.swaglm`                    *print.swaglm*

---

### Description

Print a swaglm object

### Usage

```
## S3 method for class 'swaglm'  
print(x, ...)
```

### Arguments

`x`                    An object of class swaglm.  
`...`                Additional arguments

### Value

None.

### Examples

```
n <- 2000  
p <- 50  
# create design matrix and vector of coefficients  
Sigma <- diag(rep(1/p, p))  
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)  
beta <- c(-15,-10,5,10,15, rep(0,p-5))  
z <- 1 + X%*%beta  
pr <- 1/(1 + exp(-z))  
y <- as.factor(rbinom(n, 1, pr))  
y <- as.numeric(y)-1  
quantile_alpha <- .15  
p_max <- 20  
swag_obj <- swaglm::swaglm(X=X, y = y, p_max = p_max, family = stats::binomial(),  
                          alpha = quantile_alpha, verbose = TRUE, seed = 123)  
print(swag_obj)
```

---

print.swaglm\_test      *print.swaglm\_test*

---

## Description

Print a swaglm\_test object

## Usage

```
## S3 method for class 'swaglm_test'  
print(x, ...)
```

## Arguments

x                    An object of class swaglm\_test.  
...                   Additional arguments

## Value

None.

## Examples

```
n <- 2000  
p <- 50  
  
# create design matrix and vector of coefficients  
Sigma <- diag(rep(1/p, p))  
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)  
beta = c(-15,-10,5,10,15, rep(0,p-5))  
z <- 1 + X%%beta  
pr <- 1/(1 + exp(-z))  
y <- as.factor(rbinom(n, 1, pr))  
y = as.numeric(y)-1  
quantile_alpha = .15  
p_max = 20  
swag_obj = swaglm::swaglm(X=X, y = y, p_max = p_max, family = stats::binomial(),  
                          alpha = quantile_alpha, verbose = TRUE, seed = 123)  
swag_test = swaglm::swaglm_test(swag_obj, B = 10, verbose = TRUE)  
swag_test
```

---

summary.swaglm	<i>summary.swaglm</i>
----------------	-----------------------

---

## Description

summary.swaglm

## Usage

```
## S3 method for class 'swaglm'
summary(object, ...)
```

## Arguments

object	An object of class swaglm.
...	Additional parameters

## Value

A list of class summary\_swaglm with five elements:

**mat\_selected\_model** A matrix where each row represents a selected model. Columns give the indices of the variables included in that model. Models are padded with NA to match the largest model size.

**mat\_beta\_selected\_model** A matrix containing the estimated regression coefficients (including intercept) of the selected models, stacked across all dimensions. Only models with AIC less than or equal to the **lowest median AIC across all dimensions** are included. Each row corresponds to a model, columns correspond to the coefficients. Rows are padded with NA to match the largest model size.

**mat\_p\_value\_selected\_model** A matrix containing the p-values associated with the estimated regression coefficients in beta\_selected\_model, stacked in the same order. Each row corresponds to a model, columns correspond to the coefficients. Rows are padded with NA to match the largest model size.

**vec\_aic\_selected\_model** A numeric vector containing the AIC values of all models in mat\_selected\_model, stacked across dimensions. These are the AIC values for the selected models that passed the threshold described above.

**lst\_estimated\_beta\_per\_variable** A named list where each element corresponds to a variable (named V<index>). Each element is a numeric vector containing all estimated beta coefficients for that variable across all selected models in which it appears. This summarizes the distribution of effects for each variable across the selected models.

**lst\_p\_value\_per\_variable** A named list where each element corresponds to a variable (named V<index>). Each element is a numeric vector containing all estimated p-values for that variable across all selected models in which it appears.

**Model selection criterion:** For each model dimension (number of variables in the model), the median AIC across all models of that dimension is computed. The **smallest median AIC across all dimensions** is identified. Then, **all models with AIC less than or equal to this value** are selected. This ensures that only relatively well-performing models across all dimensions are retained for summarization.

## Examples

```

set.seed(12345)
n <- 2000
p <- 100
# create design matrix and vector of coefficients
Sigma <- diag(rep(1/p, p))
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)
beta = c(-15,-10,5,10,15, rep(0,p-5))

# ----- generate from logistic regression with an intercept of one
z <- 1 + X%*%beta
pr <- 1/(1 + exp(-z))
y <- as.factor(rbinom(n, 1, pr))
y = as.numeric(y)-1

# define swag parameters
quantile_alpha = .15
p_max = 20
swaglm_obj = swaglm::swaglm(X=X, y = y, p_max = p_max, family = stats::binomial(),
                           alpha = quantile_alpha, verbose = TRUE, seed = 123)

swaglm_obj
swaglm_summ = summary(swaglm_obj)
swaglm_summ$mat_selected_model
swaglm_summ$mat_beta_selected_model
swaglm_summ$mat_p_value_selected_model
swaglm_summ$vec_aic_selected_model
swaglm_summ$lst_estimated_beta_per_variable

# plot distribution of estimated beta with respect to true value
for(i in seq_along(swaglm_summ$lst_estimated_beta_per_variable)){
  # get variable name

  var_name_i = names(swaglm_summ$lst_estimated_beta_per_variable)[i]
  var_index_i = as.numeric(substr(var_name_i, 2, nchar(var_name_i)))
  boxplot(swaglm_summ$lst_estimated_beta_per_variable[[i]],
          main=paste0(var_name_i, " ", "true value=", beta[i]))
}

```

**Description**

Run the SWAG algorithm on Generalized Linear Models specified by a family object and using the fastglm library.

**Usage**

```
swaglm(
  X,
  y,
  p_max = 2L,
  family = NULL,
  method = 0L,
  alpha = 0.3,
  verbose = FALSE,
  seed = 123L
)
```

**Arguments**

X	A numeric matrix of predictors.
y	A numeric vector of responses.
p_max	An integer specifying the maximum dimension to explore
family	A family object. Default is binomial.
method	An integer scalar with value 0 for the column-pivoted QR decomposition, 1 for the unpivoted QR decomposition, 2 for the LLT Cholesky, or 3 for the LDLT Cholesky. See ?fastglm::fastglm
alpha	A double specifying the quantile of the criterion used to select models which are employed to construct models to explore at the next dimension
verbose	A boolean used to control verbose
seed	An integer that is the random seed used when creating the set of model to explore for the next dimension

**Value**

An object of class swaglm structured as a List containing:

- `lst_estimated_beta`: A List that contain the estimated coefficients for each estimated model. Each entry of this List is a matrix where in each rows are the estimated coefficients for the model.
- `lst_p_value`: A List that contain the p-value associated with each estimated coefficients for each estimated model. Each entry of this List is a matrix where in each rows are the p-value for the model.
- `lst_AIC`: A List that contains the AIC values for each model at each dimension. Each entry of this list correspond to the AIC values for the models explored at this dimension.
- `lst_var_mat`: A List that that contain in each of its entries, a matrix that specify for each row a combination of variables that compose a model.

- `lst_selected_models` A List that contain the selected models at each dimension.
- `lst_index_selected_models` A List that contain the index of the rows corresponding to the selected models at each dimension.
- `vec_selected_variables_dimension_1` A vector that contain the index of the selected variables at the screening step.
- `y` The response vector used in the estimation.
- `X` The predictor matrix used in the estimation.
- `p_max` The maximum dimension explored by the algorithm.
- `alpha` The selection quantile used at each step.
- `family` The GLM family used in the estimation (e.g. `binomial()`).
- `method` The method used by `fastglm` for estimation.

## Examples

```
# Parameters for data generation
set.seed(12345)
n <- 2000
p <- 100
# create design matrix and vector of coefficients
Sigma <- diag(rep(1/p, p))
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)
beta = c(-15,-10,5,10,15, rep(0,p-5))

# ----- generate from logistic regression with an intercept of one
z <- 1 + X%*%beta
pr <- 1/(1 + exp(-z))
y <- as.factor(rbinom(n, 1, pr))
y = as.numeric(y)-1

# define swag parameters
quantile_alpha = .15
p_max = 20
swag_obj = swaglm::swaglm(X=X, y = y, p_max = p_max, family = stats::binomial(),
alpha = quantile_alpha, verbose = TRUE, seed = 123)
str(swag_obj)
```

---

swaglm\_test

*swaglm\_test*


---

## Description

Compute significance of identified set of variables

## Usage

```
swaglm_test(swag_obj, B = 50, verbose = FALSE)
```

**Arguments**

swaglm_obj	An object of class swaglm.
B	a integer specifying the number of swag procedures to generate a distribution of the network statistics under the null.
verbose	A boolean used to control verbose

**Value**

A swaglm\_test object.

**Examples**

```
n <- 2000
p <- 50

# create design matrix and vector of coefficients
Sigma <- diag(rep(1/p, p))
X <- MASS::mvrnorm(n = n, mu = rep(0, p), Sigma = Sigma)
beta = c(-15,-10,5,10,15, rep(0,p-5))
z <- 1 + X%%beta
pr <- 1/(1 + exp(-z))
y <- as.factor(rbinom(n, 1, pr))
y = as.numeric(y)-1
quantile_alpha = .15
p_max = 20
swaglm_obj = swaglm::swaglm(X=X, y = y, p_max = p_max, family = stats::binomial(),
                           alpha = quantile_alpha, verbose = TRUE, seed = 123)
swaglm::swaglm_test(swaglm_obj, B = 10, verbose = TRUE)
```

# Index

`compute_network`, [2](#)

`plot.swaglm_network`, [3](#)

`predict.swaglm`, [4](#)

`print.swaglm`, [7](#)

`print.swaglm_test`, [8](#)

`summary.swaglm`, [9](#)

`swaglm`, [4](#), [10](#)

`swaglm_test`, [12](#)