

# Package: PoSIAdjRSquared (via r-universe)

August 27, 2024

**Title** Post-Selection Inference for Adjusted R Squared

**Version** 0.0.0.1

**Maintainer** Sarah Pirenne <sarah.pirenne@kuleuven.be>

**Description** Conduct post-selection inference for regression coefficients in linear models after they have been selected by adjusted R squared. The p-values and confidence intervals are valid after model selection with the same data. This allows the user to use all data for both model selection and inference without losing control over the type I error rate. The provided tests are more powerful than data splitting, which bases inference on less data since it discards all information used for selection.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Imports** lmf, stats, utils, VGAM

**NeedsCompilation** no

**Author** Sarah Pirenne [aut, cre]

(<<https://orcid.org/0000-0003-1787-2035>>), Gerda Claeskens [aut]

**Repository** CRAN

**Date/Publication** 2024-08-26 13:00:02 UTC

## Contents

compute_ci_with_specified_interval . . . . .	2
construct_adj_r_squared . . . . .	3
construct_selection_event . . . . .	4
construct_test_statistic . . . . .	5
datagen.norm . . . . .	6
datagen.norm.intercept . . . . .	7
equal_tailed_interval . . . . .	8

f . . . . .	9
find_root . . . . .	10
fit_all_subset_linear_models . . . . .	11
fit_specified_size_subset_linear_models . . . . .	12
pivot_with_specified_interval . . . . .	13
postsel_p_value_specified_interval . . . . .	14
solve_selection_event . . . . .	16
<b>Index</b>	<b>18</b>

---

compute\_ci\_with\_specified\_interval

*Compute post-selection confidence interval with specified interval*

---

### Description

This function inverts a post-selection p-value to a confidence interval.

### Usage

```
compute_ci_with_specified_interval(z_interval, etaj, etajTy, Sigma, tn_mu, alpha)
```

### Arguments

z_interval	The intervals of type "list" where the OLS estimator gets selected: can be obtained from function "solve_selection_event"
etaj	Vector of type "matrix" and dimension nx1: useful in orthogonal decomposition of y (see Lemma 1 for details)
etajTy	The OLS estimator of the j'th selected coefficient in the selected model of type "matrix" and dimension 1x1
Sigma	The variance covariance matrix of dimension nxn of the error in the model
tn_mu	Integer for the mean of the truncated sampling distribution of the test statistic under the null hypothesis: for example, if you want to test $\beta_j=0$ , specify 0 for the mean
alpha	Integer for the desired significance level of the confidence interval

### Value

ci	The two-sided (1-alpha)% confidence interval valid after model selection
----	--

### References

Pirrenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

**Examples**

```

# Generate data
n <- 100
Data <- datagen.norm(seed = 7, n, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y

# Select model
result <- fit_all_subset_linear_models(y, X, intercept=FALSE)
phat <- result$phat
X_M_phat <- result$X_M_phat
k <- result$k
R_M_phat <- result$R_M_phat
kappa_M_phat <- result$kappa_M_phat
R_M_k <- result$R_M_k
kappa_M_k <- result$kappa_M_k

# Estimate Sigma from residuals of full model
full_model <- lm(y ~ 0 + X)
sigma_hat <- sd(resid(full_model))
Sigma <- diag(n)*(sigma_hat)^2

# Construct test statistic
Construct_test <- construct_test_statistic(j = 5, X_M_phat, y, phat, Sigma, intercept=FALSE)
a <- Construct_test$a
b <- Construct_test$b
etaj <- Construct_test$etaj
etajTy <- Construct_test$etajTy

# Solve selection event
Solve <- solve_selection_event(a,b,R_M_k,kappa_M_k,R_M_phat,kappa_M_phat,k)
z_interval <- Solve$z_interval

# Post-selection confidence interval
compute_ci_with_specified_interval(z_interval, etaj, etajTy, Sigma, tn_mu = 0, alpha = 0.05)

```

---

construct\_adj\_r\_squared

*Construct adjusted R squared*

---

**Description**

This function computes the adjusted R squared and returns some useful matrices from this computation.

**Usage**

```
construct_adj_r_squared(X, k, y, n, intercept = c(TRUE, FALSE), sst)
```

**Arguments**

X	Design matrix of type "matrix" and dimension nxp
k	Index set included in model k
y	Response vector of type "matrix" and dimension nx1
n	An integer for the sample size
intercept	Logical value: TRUE if fitted models should contain intercept, FALSE if not
sst	An integer for the total sum of squares

**Value**

X_M_k	The design matrix of model k
P_M_k	The projection matrix of model k
R_M_k	The orthogonal projection matrix of model k
kappa_M_k	Adjustment factor for model complexity kappa of model k
adj_r_squared	The adjusted R squared value of model k

**References**

Pirrenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

**Examples**

```
# Generate data
n <- 100
k <- 1:10
Data <- datagen.norm(seed = 7, n, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y
sst <- sum((y-mean(y))^2)

construct_adj_r_squared(X, k, y, n, intercept=FALSE, sst)
```

---

construct\_selection\_event

*Construct selection event*

---

**Description**

This function constructs the selection event by computing  $c_k$ ,  $d_k$  and  $e_k$  which are the constants in the quadratic inequalities which characterize the model selection event. The function is used internally by the function `solve_selection_event`, which returns the intervals of the OLS estimator where the selection event takes place.

**Usage**

```
construct_selection_event(a,b,R_M_k,kappa_M_k,R_M_phat,kappa_M_phat)
```

**Arguments**

a	Residual vector of type "matrix" and dimension nx1 (see Lemma 1 for details)
b	Vector of type "matrix" and dimension nx1: useful in orthogonal decomposition of y (see Lemma 1 for details)
R_M_k	The orthogonal projection matrix of model k
kappa_M_k	Adjustment factor for model complexity kappa of model k
R_M_phat	The orthogonal projection matrix of the selected model
kappa_M_phat	Adjustment factor for model complexity kappa of the selected model

**Value**

c_k	Constant c_k in the quadratic inequality $c_k Z^2 + d_k Z + e_k \geq 0$ which characterizes the model selection event of the selected model compared to model k (see Lemma 1 for details)
d_k	Constant d_k in the quadratic inequality $c_k Z^2 + d_k Z + e_k \geq 0$ which characterizes the model selection event of the selected model compared to model k (see Lemma 1 for details)
e_k	Constant e_k in the quadratic inequality $c_k Z^2 + d_k Z + e_k \geq 0$ which characterizes the model selection event of the selected model compared to model k (see Lemma 1 for details)

**References**

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

---

construct\_test\_statistic

*Construct test statistic*

---

**Description**

This function constructs the OLS estimator of the j'th selected coefficient in the selected model. The functions also returns some useful vectors for post-selection inference (a and b).

**Usage**

```
construct_test_statistic(j, X_M_phat, y, phat, Sigma, intercept)
```

**Arguments**

j	The index of type "integer" of the regression coefficient
X_M_phat	The design matrix in the selected model
y	Response vector of type "matrix" and dimension nx1
phat	Index set included in the selected model
Sigma	The variance covariance matrix of dimension nxn of the error in the model
intercept	Logical value: TRUE if the selected model contains an intercept, FALSE if not

**Value**

etaj	Vector of type "matrix" and dimension nx1: useful in orthogonal decomposition of y (see Lemma 1 for details)
etajTy	The OLS estimator of the j'th selected coefficient in the selected model of type "matrix" and dimension 1x1
a	Residual vector of type "matrix" and dimension nx1 (see Lemma 1 for details)
b	Vector of type "matrix" and dimension nx1: useful in orthogonal decomposition of y (see Lemma 1 for details)

**References**

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

**Examples**

```
# Generate data
n <- 100
Data <- datagen.norm(seed = 7, n, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y

# Select model
result <- fit_all_subset_linear_models(y, X, intercept=FALSE)
phat <- result$phat
X_M_phat <- result$X_M_phat

# Estimate Sigma from residuals of full model
full_model <- lm(y ~ 0 + X)
sigma_hat <- sd(resid(full_model))
Sigma <- diag(n)*(sigma_hat)^2

# Construct test statistic
construct_test_statistic(j = 5, X_M_phat, y, phat, Sigma, intercept=FALSE)
```

---

datagen.norm

*Data generation normal*


---

**Description**

Function to generate data according to the linear model of the form  $Y = X \cdot \beta + \epsilon$  where the noise  $\epsilon$  follows a standard normal distribution.

**Usage**

```
datagen.norm(seed, n, p, rho, beta_vec)
```

**Arguments**

seed	Integer for seed
n	Integer for sample size
p	Integer for number of variables in the design matrix
rho	Integer for correlation between variables in the design matrix
beta_vec	True regression coefficient vector of length p

**Value**

X	Design matrix of type "matrix" and dimension nxp
y	Response vector of type "matrix" and dimension nx1
true_y	True response vector, i.e. without the noise, of type "matrix" and dimension nx1

**Examples**

```
datagen.norm(seed = 7, n = 100, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
```

---

```
datagen.norm.intercept
```

*Data generation normal with intercept*

---

**Description**

Function to generate data according to the linear model of the form  $Y = X \cdot \beta + \epsilon$  where the noise  $\epsilon$  follows a standard normal distribution and the first column of  $X$  consists of 1's such that an intercept is included in the model.

**Usage**

```
datagen.norm.intercept(seed, n, p, rho, beta_vec)
```

**Arguments**

seed	Integer for seed
n	Integer for sample size
p	Integer for number of variables in the design matrix
rho	Integer for correlation between variables in the design matrix
beta_vec	True regression coefficient vector of length p

**Value**

X	Design matrix of type "matrix" and dimension nxp
y	Response vector of type "matrix" and dimension nx1
true_y	True response vector, i.e. without the noise, of type "matrix" and dimension nx1

**Examples**

```
datagen.norm.intercept(seed = 7, n = 100, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
```

---

```
equal_tailed_interval Equal tailed interval
```

---

**Description**

This function inverts a post-selection p-value to a confidence interval.

**Usage**

```
equal_tailed_interval(z_interval, etajTy, alpha, tn_mu, tn_sigma)
```

**Arguments**

z_interval	The intervals of type "list" where the OLS estimator gets selected: can be obtained from function "solve_selection_event"
etajTy	The OLS estimator of the j'th selected coefficient in the selected model of type "matrix" and dimension 1x1
alpha	Integer for the desired significance level of the confidence interval
tn_mu	Integer for the mean of the truncated sampling distribution of the test statistic under the null hypothesis: for example, if you want to test $\beta_j=0$ , specify 0 for the mean
tn_sigma	Integer for the variance of the truncated sampling distribution of the test statistic

**Value**

L	lower bound
U	upper bound

**References**

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

**Examples**

```
# Generate data
n <- 100
Data <- datagen.norm(seed = 7, n, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y

# Select model
result <- fit_all_subset_linear_models(y, X, intercept=FALSE)
phat <- result$phat
X_M_phat <- result$X_M_phat
```



```

k <- result$k
R_M_phat <- result$R_M_phat
kappa_M_phat <- result$kappa_M_phat
R_M_k <- result$R_M_k
kappa_M_k <- result$kappa_M_k

# Estimate Sigma from residuals of full model
full_model <- lm(y ~ 0 + X)
sigma_hat <- sd(resid(full_model))
Sigma <- diag(n)*(sigma_hat)^2

# Construct test statistic
Construct_test <- construct_test_statistic(j = 5, X_M_phat, y, phat, Sigma, intercept=FALSE)
a <- Construct_test$a
b <- Construct_test$b
etaj <- Construct_test$etaj
etajTy <- Construct_test$etajTy

# Solve selection event
Solve <- solve_selection_event(a,b,R_M_k,kappa_M_k,R_M_phat,kappa_M_phat,k)
z_interval <- Solve$z_interval

# Post-selection confidence interval
tn_sigma <- sqrt((t(etaj)**Sigma)**etaj)
equal_tailed_interval(z_interval, etajTy, alpha = 0.05, tn_mu = 0, tn_sigma)

```

f

f

---

## Description

Function used internally by `compute_ci_with_specified_interval` for calculating valid confidence intervals post-selection.

## Arguments

<code>z_interval</code>	The intervals of type "list" where the OLS estimator gets selected: can be obtained from function "solve_selection_event"
<code>etajTy</code>	The OLS estimator of the j'th selected coefficient in the selected model of type "matrix" and dimension 1x1
<code>mu</code>	Integer for the mean of the truncated sampling distribution of the test statistic (updated iteratively in <code>compute_ci_with_specified_interval</code> )
<code>tn_sigma</code>	Integer for the variance of the truncated sampling distribution of the test statistic

## Value

The cumulative distribution function of a truncated gaussian distribution evaluated in the observed test statistic

**References**

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

---

 find\_root

*Find root*


---

**Description**

This function is used internally by the function `compute_ci_with_specified_interval` for inverting post-selection p-values to confidence intervals.

**Usage**

```
find_root(z_interval, etajTy, tn_sigma, y, lb, ub, tol=1e-6)
```

**Arguments**

<code>z_interval</code>	The intervals of type "list" where the OLS estimator gets selected: can be obtained from function "solve_selection_event"
<code>etajTy</code>	The OLS estimator of the j'th selected coefficient in the selected model of type "matrix" and dimension 1x1
<code>tn_sigma</code>	Integer for the variance of the truncated sampling distribution of the test statistic
<code>y</code>	For example 1.0-0.5*alpha for finding the lower bound of a (1-alpha)% confidence interval, and 0.5*alpha for finding the upper bound of a (1-alpha)% confidence interval
<code>lb</code>	Lower bound in current iteration
<code>ub</code>	Upper bound in current iteration
<code>tol</code>	Tolerance parameter: default set to 1e-6

**Value**

Returns confidence interval bound

**References**

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

---

```
fit_all_subset_linear_models
```

*Fit all subset linear models*

---

### Description

This function fits all possible combinations of linear models and returns the selected model based on adjusted  $R^2$ .

### Usage

```
fit_all_subset_linear_models(y, X, intercept)
```

### Arguments

y	Response vector of type "matrix" and dimension $n \times 1$
X	Design matrix of type "matrix" and dimension $n \times p$
intercept	Logical value: TRUE if fitted models should contain intercept, FALSE if not

### Value

k	Index set included in model k
best_model	The selected model fit (lm object)
phat	Index set included in the selected model
X_M_phat	The design matrix in the selected model
best_adj_r_squared	The adjusted $R^2$ value of the selected model
R_M_phat	The orthogonal projection matrix of the selected model
kappa_M_phat	Adjustment factor for model complexity kappa of the selected model
R_M_k	The orthogonal projection matrix of model k
kappa_M_k	Adjustment factor for model complexity kappa of model k

### References

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

### Examples

```
# Generate data
Data <- datagen.norm(seed = 7, n = 100, p = 3, rho = 0, beta_vec = c(1,0.5,0))
X <- Data$X
y <- Data$y

# Select model
fit_all_subset_linear_models(y, X, intercept=FALSE)
```

---

```
fit_specified_size_subset_linear_models
```

*Fit all linear models of a specified size*

---

### Description

This function fits all possible combinations of a pre-specified size of linear models and returns the selected model based on adjusted  $R^2$ .

### Usage

```
fit_specified_size_subset_linear_models(y, X, size, intercept)
```

### Arguments

y	Response vector of type "matrix" and dimension $n \times 1$
X	Design matrix of type "matrix" and dimension $n \times p$
size	Size of type "integer" of the fitted models
intercept	Logical value: TRUE if fitted models should contain intercept, FALSE if not

### Value

k	Index set included in model k
best_model	The selected model fit (lm object)
phat	Index set included in the selected model
X_M_phat	The design matrix in the selected model
best_adj_r_squared	The adjusted $R^2$ value of the selected model
R_M_phat	The orthogonal projection matrix of the selected model
kappa_M_phat	Adjustment factor for model complexity kappa of the selected model
R_M_k	The orthogonal projection matrix of model k
kappa_M_k	Adjustment factor for model complexity kappa of model k

### References

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

### Examples

```
# Generate data
Data <- datagen.norm(seed = 7, n = 100, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y

# Select model
fit_specified_size_subset_linear_models(y, X, size = 9, intercept=FALSE)
```

---

```

pivot_with_specified_interval
      Pivot with specified interval

```

---

## Description

This function returns the value of the cumulative distribution function of a truncated gaussian distribution evaluated in the observed test statistic. Its output is used by the function `postselp_value_specified_interval` by taking  $2 \cdot \min(\text{output}, 1 - \text{output})$  for a p-value for a two-sided test.

## Usage

```

pivot_with_specified_interval(z_interval, etaj, etajTy, tn_mu, tn_sigma)

```

## Arguments

<code>z_interval</code>	The intervals of type "list" where the OLS estimator gets selected: can be obtained from function "solve_selection_event"
<code>etaj</code>	Vector of type "matrix" and dimension $n \times 1$ : useful in orthogonal decomposition of $y$ (see Lemma 1 for details)
<code>etajTy</code>	The OLS estimator of the $j$ 'th selected coefficient in the selected model of type "matrix" and dimension $1 \times 1$
<code>tn_mu</code>	Integer for the mean of the truncated sampling distribution of the test statistic under the null hypothesis: for example, if you want to test $\beta_j = 0$ , specify 0 for the mean
<code>tn_sigma</code>	Integer for the variance of the truncated sampling distribution of the test statistic

## Value

The cumulative distribution function of a truncated gaussian distribution evaluated in the observed test statistic

## References

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

## Examples

```

# Generate data
n <- 100
Data <- datagen.norm(seed = 7, n, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y

# Select model
result <- fit_all_subset_linear_models(y, X, intercept=FALSE)
phat <- result$phat

```

```

X_M_phat <- result$X_M_phat
k <- result$k
R_M_phat <- result$R_M_phat
kappa_M_phat <- result$kappa_M_phat
R_M_k <- result$R_M_k
kappa_M_k <- result$kappa_M_k

# Estimate Sigma from residuals of full model
full_model <- lm(y ~ 0 + X)
sigma_hat <- sd(resid(full_model))
Sigma <- diag(n)*(sigma_hat)^2

# Construct test statistic
Construct_test <- construct_test_statistic(j = 5, X_M_phat, y, phat, Sigma, intercept=FALSE)
a <- Construct_test$a
b <- Construct_test$b
etaj <- Construct_test$etaj
etajTy <- Construct_test$etajTy

# Solve selection event
Solve <- solve_selection_event(a,b,R_M_k,kappa_M_k,R_M_phat,kappa_M_phat,k)
z_interval <- Solve$z_interval

# Post-selection inference for beta_j=0
tn_sigma <- sqrt((t(etaj)%*%Sigma)%*%etaj)
pivot_with_specified_interval(z_interval, etaj, etajTy, tn_mu = 0, tn_sigma)

```

---

postselp\_value\_specified\_interval

*Post-selection p-value specified interval*

---

## Description

This function returns a p-value for the test whether the regression coefficient equals  $tn\_mu$  (e.g. 0) with a two-sided alternative. The p-value is valid given the model selection, because it conditions on the specified intervals of the OLS estimator where the regression coefficient actually gets selected. The intervals contained in the object "z\_interval" can be obtained from the function "solve\_selection\_event".

## Usage

```
postselp_value_specified_interval(z_interval, etaj, etajTy, tn_mu, tn_sigma)
```

## Arguments

z_interval	The intervals of type "list" where the OLS estimator gets selected: can be obtained from function "solve_selection_event"
etaj	Vector of type "matrix" and dimension $n \times 1$ : useful in orthogonal decomposition of $y$ (see Lemma 1 for details)

etajTy	The OLS estimator of the $j$ 'th selected coefficient in the selected model of type "matrix" and dimension $1 \times 1$
tn_mu	Integer for the mean of the truncated sampling distribution of the test statistic under the null hypothesis: for example, if you want to test $\beta_j=0$ , specify 0 for the mean
tn_sigma	Integer for the variance of the truncated sampling distribution of the test statistic

**Value**

p_value	The p-value for a two-sided test which is valid after model selection
---------	---

**References**

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.

**Examples**

```
# Generate data
n <- 100
Data <- datagen.norm(seed = 7, n, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y

# Select model
result <- fit_all_subset_linear_models(y, X, intercept=FALSE)
phat <- result$phat
X_M_phat <- result$X_M_phat
k <- result$k
R_M_phat <- result$R_M_phat
kappa_M_phat <- result$kappa_M_phat
R_M_k <- result$R_M_k
kappa_M_k <- result$kappa_M_k

# Estimate Sigma from residuals of full model
full_model <- lm(y ~ 0 + X)
sigma_hat <- sd(resid(full_model))
Sigma <- diag(n)*(sigma_hat)^2

# Construct test statistic
Construct_test <- construct_test_statistic(j = 5, X_M_phat, y, phat, Sigma, intercept=FALSE)
a <- Construct_test$a
b <- Construct_test$b
etaj <- Construct_test$etaj
etajTy <- Construct_test$etajTy

# Solve selection event
Solve <- solve_selection_event(a,b,R_M_k,kappa_M_k,R_M_phat,kappa_M_phat,k)
z_interval <- Solve$z_interval

# Post-selection inference for beta_j=0
tn_sigma <- sqrt((t(etaj)%*%Sigma)%*%etaj)
postselp_value_specified_interval(z_interval, etaj, etajTy, tn_mu = 0, tn_sigma)
```

---

solve\_selection\_event *Solve selection event*

---

### Description

This function solves the selection event by calculating the intervals of the OLS estimator where the regression coefficient actually gets selected.

### Usage

```
solve_selection_event(a,b,R_M_k,kappa_M_k,R_M_phat,kappa_M_phat,k)
```

### Arguments

a	Residual vector of type "matrix" and dimension nx1 (see Lemma 1 for details)
b	Vector of type "matrix" and dimension nx1: useful in orthogonal decomposition of y (see Lemma 1 for details)
R_M_k	The orthogonal projection matrix of model k
kappa_M_k	Adjustment factor for model complexity kappa of model k
R_M_phat	The orthogonal projection matrix of the selected model
kappa_M_phat	Adjustment factor for model complexity kappa of the selected model
k	Index set included in model k

### Value

intervals_list	The intervals of the OLS estimator for which the inequality in Lemma 1 holds
z_interval	The intersection of the intervals of the OLS estimator for which the inequality in Lemma 1 holds: post-selection inference is conditioned on those intervals

### Note

This function will give the error message: "Error in if (D >= 0.001): missing value where TRUE/FALSE needed" if it is run for a coefficient which is not part of the selected model. Only run solve\_selection\_event for selected indices.

### References

Pirenne, S. and Claeskens, G. (2024). Exact Post-Selection Inference for Adjusted R Squared.



**Examples**

```
# Generate data
n <- 100
Data <- datagen.norm(seed = 7, n, p = 10, rho = 0, beta_vec = c(1,0.5,0,0.5,0,0,0,0,0,0))
X <- Data$X
y <- Data$y

# Select model
result <- fit_all_subset_linear_models(y, X, intercept=FALSE)
phat <- result$phat
X_M_phat <- result$X_M_phat
k <- result$k
R_M_phat <- result$R_M_phat
kappa_M_phat <- result$kappa_M_phat
R_M_k <- result$R_M_k
kappa_M_k <- result$kappa_M_k

# Estimate Sigma from residuals of full model
full_model <- lm(y ~ 0 + X)
sigma_hat <- sd(resid(full_model))
Sigma <- diag(n)*(sigma_hat)^2

# Construct test statistic
Construct_test <- construct_test_statistic(j = 5, X_M_phat, y, phat, Sigma, intercept=FALSE)
a <- Construct_test$a
b <- Construct_test$b

# Solve selection event
solve_selection_event(a,b,R_M_k,kappa_M_k,R_M_phat,kappa_M_phat,k)
```

# Index

## \* **adjusted R squared**

- compute\_ci\_with\_specified\_interval, 2
- construct\_adj\_r\_squared, 3
- construct\_selection\_event, 4
- construct\_test\_statistic, 5
- equal\_tailed\_interval, 8
- f, 9
- find\_root, 10
- fit\_all\_subset\_linear\_models, 11
- fit\_specified\_size\_subset\_linear\_models, 12
- pivot\_with\_specified\_interval, 13
- postselp\_value\_specified\_interval, 14
- solve\_selection\_event, 16

## \* **datagen**

- datagen.norm, 6
- datagen.norm.intercept, 7

## \* **htest**

- compute\_ci\_with\_specified\_interval, 2
- construct\_adj\_r\_squared, 3
- construct\_selection\_event, 4
- construct\_test\_statistic, 5
- equal\_tailed\_interval, 8
- f, 9
- find\_root, 10
- pivot\_with\_specified\_interval, 13
- postselp\_value\_specified\_interval, 14
- solve\_selection\_event, 16

## \* **model selection**

- compute\_ci\_with\_specified\_interval, 2
- construct\_adj\_r\_squared, 3
- construct\_selection\_event, 4
- construct\_test\_statistic, 5
- equal\_tailed\_interval, 8

- f, 9
- find\_root, 10
- fit\_all\_subset\_linear\_models, 11
- fit\_specified\_size\_subset\_linear\_models, 12
- pivot\_with\_specified\_interval, 13
- postselp\_value\_specified\_interval, 14
- solve\_selection\_event, 16

## \* **models**

- compute\_ci\_with\_specified\_interval, 2
- construct\_adj\_r\_squared, 3
- construct\_selection\_event, 4
- construct\_test\_statistic, 5
- datagen.norm, 6
- datagen.norm.intercept, 7
- equal\_tailed\_interval, 8
- f, 9
- find\_root, 10
- fit\_all\_subset\_linear\_models, 11
- fit\_specified\_size\_subset\_linear\_models, 12
- pivot\_with\_specified\_interval, 13
- postselp\_value\_specified\_interval, 14
- solve\_selection\_event, 16

## \* **post-selection inference**

- compute\_ci\_with\_specified\_interval, 2
- construct\_adj\_r\_squared, 3
- construct\_selection\_event, 4
- construct\_test\_statistic, 5
- equal\_tailed\_interval, 8
- f, 9
- find\_root, 10
- pivot\_with\_specified\_interval, 13
- postselp\_value\_specified\_interval, 14

solve\_selection\_event, 16

\* **regression**

- compute\_ci\_with\_specified\_interval, 2
- construct\_adj\_r\_squared, 3
- construct\_selection\_event, 4
- construct\_test\_statistic, 5
- datagen.norm, 6
- datagen.norm.intercept, 7
- equal\_tailed\_interval, 8
- f, 9
- find\_root, 10
- fit\_all\_subset\_linear\_models, 11
- fit\_specified\_size\_subset\_linear\_models, 12
- pivot\_with\_specified\_interval, 13
- postselp\_value\_specified\_interval, 14
- solve\_selection\_event, 16

compute\_ci\_with\_specified\_interval, 2

construct\_adj\_r\_squared, 3

construct\_selection\_event, 4

construct\_test\_statistic, 5

datagen.norm, 6

datagen.norm.intercept, 7

equal\_tailed\_interval, 8

f, 9

find\_root, 10

fit\_all\_subset\_linear\_models, 11

fit\_specified\_size\_subset\_linear\_models, 12

pivot\_with\_specified\_interval, 13

postselp\_value\_specified\_interval, 14

solve\_selection\_event, 16