

# Package ‘swfdr’

October 26, 2024

**Title** Estimation of the science-wise false discovery rate and the false discovery rate conditional on covariates

**Version** 1.31.0

**Author** Jeffrey T. Leek, Leah Jager, Simina M. Boca, Tomasz Konopka

**Maintainer** Simina M. Boca <smb310@georgetown.edu>, Jeffrey T. Leek <jtleek@gmail.com>

**Description** This package allows users to estimate the science-wise false discovery rate from Jager and Leek, “Empirical estimates suggest most published medical research is true,” 2013, *Biostatistics*, using an EM approach due to the presence of rounding and censoring. It also allows users to estimate the false discovery rate conditional on covariates, using a regression framework, as per Boca and Leek, “A direct approach to estimating false discovery rates conditional on covariates,” 2018, *PeerJ*.

**Depends** R (>= 3.4)

**Imports** methods, splines, stats4, stats

**License** GPL (>= 3)

**URL** <https://github.com/leekgroup/swfdr>

**BugReports** <https://github.com/leekgroup/swfdr/issues>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**Suggests** dplyr, ggplot2, BiocStyle, knitr, qvalue, reshape2, rmarkdown, testthat

**VignetteBuilder** knitr

**biocViews** MultipleComparison, StatisticalMethod, Software

**git\_url** <https://git.bioconductor.org/packages/swfdr>

**git\_branch** devel

**git\_last\_commit** 1ba4d3b

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.20

**Date/Publication** 2024-10-25

## Contents

BMI_GIANT_GWAS_sample . . . . .	2
calculateSwfdr . . . . .	3
get_number_decimals . . . . .	4
journals_pVals . . . . .	4
lm_pi0 . . . . .	5
lm_qvalue . . . . .	6
<b>Index</b>	<b>8</b>

---

BMI\_GIANT\_GWAS\_sample *Subset of SNPs from meta-analysis of BMI GWAS study.*

---

### Description

A dataset containing 50,000 SNPs and results for their associations with BMI.

### Usage

```
data(BMI_GIANT_GWAS_sample)
```

### Format

A data frame with 50,000 rows and 9 variables:

**SNP** ID for SNP (single nucleotide polymorphism)

**A1** Allele 1 for SNP

**A2** Allele 2 for SNP

**Freq\_MAF\_Hapmap** Frequency of minor allele (MAF) in Hapmap project

**b** Estimated beta for association between SNP and BMI

**se** Estimated standard error (se) for association between SNP and BMI

**p** P-value for association between SNP and BMI

**N** Total sample size considered for association of SNP and BMI

**Freq\_MAF\_Int\_Hapmap** Three approximately equal intervals for the Hapmap MAFs

### Value

Object of class `tbl_df`, `tbl`, `data.frame`.

### Source

[https://www.broadinstitute.org/collaboration/giant/index.php/GIANT\\_consortium\\_data\\_files#GWAS\\_Anthropometric\\_2015\\_BMI](https://www.broadinstitute.org/collaboration/giant/index.php/GIANT_consortium_data_files#GWAS_Anthropometric_2015_BMI)

---

calculateSwfdr	<i>Calculate the science-wise FDR (swfdr)</i>
----------------	---

---

**Description**

Calculate the science-wise FDR (swfdr)

**Usage**

```
calculateSwfdr(
  pValues,
  truncated,
  rounded,
  pi0 = 0.5,
  alpha = 1,
  beta = 50,
  numEmIterations = 100
)
```

**Arguments**

pValues	Numerical vector of p-values
truncated	Vector of 0s and 1s with indices corresponding to those in pValues; 1 indicates that the p-values is truncated, 0 that it is not truncated
rounded	Vector of 0s and 1s with indices corresponding to those in pValues; 1 indicates that the p-values is rounded, 0 that it is not rounded
pi0	Initial prior probability that a hypothesis is null (default is 0.5)
alpha	Initial value of parameter alpha from Beta(alpha, beta) true positive distribution (default is 1)
beta	Initial value of parameter beta from Beta(alpha, beta) true positive distribution (default is 50)
numEmIterations	The number of EM iterations (default is 100)

**Value**

pi0	Final value of prior probability - estimated from EM - that a hypothesis is null, i.e. estimated swfdr
alpha	Final value of parameter alpha - estimated from EM - from Beta(alpha, beta) true positive distribution
beta	Final value of parameter beta - estimated from EM - from Beta(alpha, beta) true positive distribution
z	Vector of expected values of the indicator of whether the p-value is null or not - estimated from EM - for the non-rounded p-values (values of NA represent the rounded p-values)
n0	Expected number of rounded null p-values - estimated from EM - between certain cutpoints (0.005, 0.015, 0.025, 0.035, 0.045, 0.05)
n	Number of rounded p-values between certain cutpoints (0.005, 0.015, 0.025, 0.035, 0.045, 0.05)

**Examples**

```
pVals <- runif(100)
tt <- rr <- rep(0, 100)
resSwfdr <- calculateSwfdr(pValues = pVals, truncated = tt, rounded = rr, numEmIterations=100)
```

---

```
get_number_decimals    Get number of decimals (i.e. return total number of digits after decimal point) for any vector of numbers in [0,1) if number of decimals <= 6
```

---

**Description**

Get number of decimals (i.e. return total number of digits after decimal point) for any vector of numbers in [0,1) if number of decimals <= 6

**Usage**

```
get_number_decimals(x)
```

**Arguments**

x                      Numerical vector where all elements are in [0,1)

**Value**

Vector giving the number of decimals for each element in x if the number is <= 6; otherwise return 7 with a warning

**Examples**

```
#get_number_decimals(c(0.0006, 0.0750, 0.0420, 0.0031, 0.0001, 0.0100))
#get_number_decimals(c(6*10^-4, 7.5*10^-2, 4.2*10^-2, 3.1*10^-3, 10^-4, 10^-2))
#get_number_decimals(c(6.5*10^-4, 0.0100))
#get_number_decimals(c(6.5e-4, 0.0100))
#get_number_decimals(c(0.00065, 0.0100))
#get_number_decimals(c(10^-7, 10e-7, 10e-3))
```

---

```
journals_pVals    P-values from abstracts from articles in 5 biomedical journals (American Journal of Epidemiology, BMJ, JAMA, Lancet, New England Journal of Medicine), over 11 years (2000-2010).
```

---

**Description**

A dataset containing 15,653 p-values.

**Usage**

```
journals_pVals
```

**Format**

A tbl data frame with 15,653 rows and 5 variables:

**pvalue** P-value

**pvalueTruncated** Equals to 1 if the p-value is truncated, 0 otherwise

**pubmedID** Pubmed ID of the article

**year** Year of publication

**journal** Journal

**Value**

Object of class `tbl_df`, `tbl`, `data.frame`.

**Source**

Code for extracting p-values at: [inst/script/getPvalues.R](#)

---

lm_pi0	<i>Estimation of <math>\pi_0</math>, proportion of p-values consistent with a null hypothesis</i>
--------	---

---

**Description**

Estimation of  $\pi_0$ , proportion of p-values consistent with a null hypothesis

**Usage**

```
lm_pi0(
  p,
  lambda = seq(0.05, 0.95, 0.05),
  X,
  type = c("logistic", "linear"),
  smooth.df = 3,
  threshold = TRUE,
  smoothing = c("unit.spline", "smooth.spline")
)
```

**Arguments**

<code>p</code>	numeric vector, p-values
<code>lambda</code>	numeric vector, thresholds used to bin pvalues, must be in [0,1).
<code>X</code>	numeric matrix, covariates that might be related to p values (one test per row, one variable per column).
<code>type</code>	character, type of regression used to fit features to pvalues
<code>smooth.df</code>	integer, degrees of freedom when estimating $\pi_0(x)$ with a smoother.
<code>threshold</code>	logical, if TRUE, all estimates are thresholded into unit interval; if FALSE, all estimates are left as they are computed
<code>smoothing</code>	character, type of smoothing used to fit $\pi_0$

**Value**

object of class 'lm\_pi0', which is a list with several components

call	matched function call
lambda	numeric vector of thresholds used in calculating pi0.lambda
X.names	character vector of covariates used in modeling
pi0.lambda	numeric matrix of estimated pi0(x) for each value of lambda. The number of columns is the number of tests, the number of rows is the length of lambda.
pi0	numerical vector of smoothed estimate of pi0(x). The length is the number of rows in X.
pi0.smooth	(only output with smoothing="smooth.spline") Matrix of fitted values from the smoother fit to the pi0(x) estimates at each value of lambda (same number of rows and columns as pi0.lambda)

**Examples**

```
# define a covariate
X <- seq(-1,2,length=1000)
# set probability of being null
pi0 <- 1/4*X + 1/2
# generate null/alternative p-values
nullI <- rbinom(1000,prob=pi0,size=1)> 0
# vector of p-values
pValues <- rep(NA,1000)
pValues[nullI] <- runif(sum(nullI)) # from U(0,1)
pValues[!nullI] <- rbeta(sum(!nullI),1,2) # from Beta
pi0x <- lm_pi0(pValues, X=X)
```

---

lm\_qvalue

*Estimation of qvalues conditioned on covariates*


---

**Description**

The recipe for turning pvalues into qvalues is adapted from package 'qvalue' and articles by Storey, Tibshirani, Taylor, Siegmund.

**Usage**

```
lm_qvalue(
  p,
  X,
  pfdR = FALSE,
  pi0 = NULL,
  smoothing = c("unit.spline", "smooth.spline"),
  ...
)
```

**Arguments**

p	numeric vector of p-values
X	matrix of covariates (can be missing if pi0 is specified instead)
pfdr	logical, making estimates robust for small p-values and a small sample size
pi0	list with pi0 estimates from lm_pi0. If this is not provided, pi0 is estimated using function lm_pi0.
smoothing	character, type of smoothing used to fit pi0. Note the default in this function is different than in lm_pi0.
...	other parameters (passed on to lm_pi0 if pi0 is not provided)

**Value**

	object of class 'lm_qvalue', which is a list with several components
call	matched function call
pvalues	numeric vector of original p-values
qvalues	numeric vector of q-values other list elements transferred from pi0

**Examples**

```
# define a covariate
X <- rep(c(0, 1), each=1000)
# generate p-values, randomly for group 0 and with low values for group 1
pVal <- c(runif(1000), rbeta(1000, 0.2, 1))
# compute qvalues, using the covariate
qVal <- lm_qvalue(pVal, X=X)
```

# Index

## \* datasets

BMI\_GIANT\_GWAS\_sample, [2](#)

journals\_pVals, [4](#)

BMI\_GIANT\_GWAS\_sample, [2](#)

calculateSwfdr, [3](#)

get\_number\_decimals, [4](#)

journals\_pVals, [4](#)

lm\_pi0, [5](#)

lm\_qvalue, [6](#)