Package ‘fcfdr’

February 7, 2022

Title Flexible cFDR

Version 1.0.0

Description Provides functions to implement the Flexible cFDR (Hutchinson et al. (2021) <doi:10.1371/journal.pgen.1009853>) and Binary cFDR (Hutchinson et al. (2021) <doi:10.1101/2021.10.21.465274>) methodologies to leverage auxiliary data from arbitrary distributions, for example functional genomic data, with GWAS p-values to generate re-weighted p-values.

Imports locfdr, MASS, ggplot2, cowplot, fields, dplyr, spatstat.geom, polyCub, hexbin, bigsplines, data.table, grDevices, Hmisc

Suggests stats, knitr, rmarkdown, digest, testthat (>= 3.0.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

VignetteBuilder knitr

Depends R (>= 3.5.0)

Config/testthat/edition 3

 NeedsCompilation no

Author Anna Hutchinson [aut, cre] (<https://orcid.org/0000-0002-9224-4410>), Chris Wallace [aut], Thomas Willis [ctb, aut], James Liley [ctb]

Maintainer Anna Hutchinson <annahutchinson1995@gmail.com>

Repository CRAN

Date/Publication 2022-02-07 09:00:05 UTC

R topics documented:

  binary_cfdr ............................................................. 2
corr_plot ................................................................. 3
**binary Cfdr**

**Perform cFDR leveraging binary auxiliary covariates**

**Description**

Perform cFDR leveraging binary auxiliary covariates

**Usage**

`binary_cfdr(p, q, group)`

**Arguments**

- **p**  
  p-values for principal trait (vector of length n)

- **q**  
  binary auxiliary data values (vector of length n)

- **group**  
  group membership of each SNP for leave-one-out procedure (vector of length n) (e.g. chromosome number or LD block)

**Value**

data.frame of p, q and v values

**Examples**

```r
# In this example, we generate some p-values (representing GWAS p-values)  
# and some arbitrary auxiliary data values (e.g. representing functional genomic data).  
# We use the parameters_in_locfdr() function to extract the parameters estimated by  
# the locfdr function.

# generate p
set.seed(2)  
n <- 1000  
n1p <- 50  
zp <- c(rnorm(n1p, sd=5), rnorm(n-n1p, sd=1))  
p <- 2*pnorm(-abs(zp))

# generate q
q <- rbinom(n, 1, 0.1)
```
group <- c(rep("A", n/2), rep("B", n/2))
binary_cfdr(p, q, group)

---
corr_plot  Violin plot of p-values for quantiles of q

Description
Violin plot of p-values for quantiles of q

Usage
corr_plot(p, q, ylim = c(0, 1.5))

Arguments
- **p**: p values for principal trait (vector of length n)
- **q**: auxiliary data values (vector of length n)
- **ylim**: y-axis limits (-log10)

Details
Can be used to investigate the relationship between p and q
If this shows a non-monotonic relationship then the cFDR framework should not be used
(because e.g. cFDR cannot simultaneously shrink v-values for high p and low p)

Value
ggplot object

Examples

```r
# In this example, we generate some p-values (representing GWAS p-values)
# and some arbitrary auxiliary data values (e.g. representing functional genomic data).
# We use the corr_plot() function to visualise the relationship between p and q.

# generate p
set.seed(1)
n <- 1000
nlp <- 50
zp <- c(rnorm(nlp, sd=5), rnorm(n-nlp, sd=1))
p <- 2*pnorm(-abs(zp))

# generate q
```
mixture_comp1 <- function(x) rnorm(x, mean = -0.5, sd = 0.5)
mixture_comp2 <- function(x) rnorm(x, mean = 2, sd = 1)
q <- c(mixture_comp1(n1p), mixture_comp2(n-n1p))
corr_plot(p, q)

flexible_cfdr  
Perform Flexible cFDR

Description

Performs Flexible cFDR for continuous auxiliary covariates

Usage

flexible_cfdr(
  p,
  q,
  indep_index,
  res_p = 300,
  res_q = 500,
  nxbin = 1000,
  gridp = 50,
  splinecorr = TRUE,
  dist_thr = 0.5,
  locfdr_df = 10,
  plot = TRUE,
  maf = NULL,
  check_indep_cor = TRUE,
  enforce_p_q_cor = TRUE
)

Arguments

p  p-values for principal trait (vector of length n)
q  continuous auxiliary data values (vector of length n)
indep_index  indices of independent SNPs
res_p  number of grid points in x-direction (p) for KDE estimation
res_q  number of grid points in y-direction (q) for KDE estimation
nxbin  number of bins in x-direction (p) for hex-binning
gridp  number of data points required in a KDE grid point for left-censoring
splinecorr  logical value for whether spline correction should be implemented
dist_thr  distance threshold for spline correction
locfdr_df  df parameter in locfdr function
flexible_cfdr

plot logical value for whether to produce plots to assess KDE fit
maf minor allele frequencies for SNPs to which p and q relate (optional and used to perform MAF matching)
check_indep_cor check that sign of the correlation between p and q is the same in the independent subset as in the whole
enforce_p_q_cor if p and q are negatively correlated, flip the sign on q values

Details
If maf is specified, then the independent SNPs will be down-sampled to match the minor allele frequency distribution.

Value
List of length two: (1) data.frame of p-values, q-values and v-values (2) data.frame of auxiliary data (q_low used for left censoring, how many data-points were left censored and/or spline corrected)

Examples

# this is a long running example

# In this example, we generate some p-values (representing GWAS p-values) and some arbitrary auxiliary data values (e.g. representing functional genomic data). We use the flexible_cfdr() function to generate v-values using default parameter values.

# generate p
set.seed(1)
n <- 1000
n1p <- 50
zp <- c(rnorm(n1p, sd=5), rnorm(n-n1p, sd=1))
p <- 2*pnorm(-abs(zp))

# generate q
mixture_comp1 <- function(x) rnorm(x, mean = -0.5, sd = 0.5)
mixture_comp2 <- function(x) rnorm(x, mean = 2, sd = 1)
q <- c(mixture_comp1(n1p), mixture_comp2(n-n1p))
n_indep <- n

flexible_cfdr(p, q, indep_index = 1:n_indep)
log10pv_plot

Plot -$\log_{10}(p)$ against -$\log_{10}(v)$ and colour by q

Description
Plot -$\log_{10}(p)$ against -$\log_{10}(v)$ and colour by q

Usage
log10pv_plot(p, q, v, axis_lim = c(0, 20))

Arguments
- p: p values for principal trait (vector of length n)
- q: auxiliary data values (vector of length n)
- v: v values from cFDR
- axis_lim: Optional axis limits

Details
Can be used to visualise the results from Flexible cFDR

Value
ggplot object

Examples
# this is a long running example

# In this example, we generate some p-values (representing GWAS p-values) 
# and some arbitrary auxiliary data values (e.g. representing functional genomic data). 
# We use the flexible_cfdr() function to generate v-values and then the log10pv_plot() function 
# to visualise the results.

# generate p
set.seed(1)
n <- 1000
nlp <- 50
zp <- c(rnorm(nlp, sd=5), rnorm(n-nlp, sd=1))
p <- 2*pnorm(-abs(zp))

# generate q
mixture_compl <- function(x) rnorm(x, mean = -0.5, sd = 0.5)
mixture_compz <- function(x) rnorm(x, mean = 2, sd = 1)
q <- c(mixture_compl(nlp), mixture_compz(n-nlp))
match_ind_maf

n_indep <- n
res <- flexible_cfdr(p, q, indep_index = 1:n_indep)
log10pv_plot(p = res[[1]]$p, q = res[[1]]$q, v = res[[1]]$v)

match_ind_maf

Function to downsample independent SNPs to match MAF distribution of whole set.

Description

Matches MAF distribution of independent set of SNPs to MAF distribution of whole set of SNPs to avoid MAF-based confounding.

Usage

match_ind_maf(maf, indep_index)

Arguments

maf
minor allele frequencies of (all) SNPs

indep_index
indices of independent SNPs

Details

Must supply maf values from the whole data set, not just the independent SNPs.

Value

indices of independent SNP in chosen in sample

parameters_in_locfdr

Description

parameters_in_locfdr
Usage

parameters_in_locfdr(
  p,
  q,
  indep_index,
  res_p = 300,
  res_q = 500,
  maf = NULL,
  check_indep_cor = TRUE,
  enforce_p_q_cor = TRUE
)

Arguments

p  p values for principal trait (vector of length n)
q  continuous auxiliary data values (vector of length n)
indep_index  indices of independent SNPs
res_p  resolution for p
res_q  resolution for q
maf  minor allele frequencies for SNPs to which p and q relate (optional and used to perform MAF matching)
check_indep_cor  check that sign of the correlation between p and q is the same in the independent subset as in the whole
enforce_p_q_cor  if p and q are negatively correlated, flip the sign on q values

Value

list of values used as input into locfdr::locfdr function intrinsically in flexible_cfdr

Examples

# In this example, we generate some p-values (representing GWAS p-values)
# and some arbitrary auxiliary data values (e.g. representing functional genomic data).
# We use the parameters_in_locfdr() function to extract the parameters estimated by # the locfdr function.

# generate p
set.seed(1)
n <- 1000
nlp <- 50
zp <- c(rnorm(nlp, sd=5), rnorm(n-nlp, sd=1))
p <- 2*pnorm(-abs(zp))

# generate q
mixture_comp1 <- function(x) rnorm(x, mean = -0.5, sd = 0.5)
mixture_comp2 <- function(x) rnorm(x, mean = 2, sd = 1)
q <- c(mixture_comp1(n1p), mixture_comp2(n-n1p))
n_indep <- n
parameters_in_locfdr(p, q, indep_index = 1:n_indep)

---

**pv_plot**

*Plot p against v and colour by q*

**Description**

Plot p against v and colour by q

**Usage**

`pv_plot(p, q, v, axis_lim = c(0, 1))`

**Arguments**

- `p`: p values for principal trait (vector of length n)
- `q`: auxiliary data values (vector of length n)
- `v`: v values from cFDR
- `axis_lim`: Optional axis limits

**Details**

Can be used to visualise the results from Flexible cFDR

**Value**

ggplot object

**Examples**

```
# this is a long running example

# In this example, we generate some p-values (representing GWAS p-values)
# and some arbitrary auxiliary data values (e.g. representing functional genomic data).
# We use the flexible_cfdr() function to generate v-values and then the pv_plot() function
# to visualise the results.

# generate p
set.seed(1)
n <- 1000
```
```r
n1p <- 50
zp <- c(rnorm(n1p, sd=5), rnorm(n-n1p, sd=1))
p <- 2*pnorm(-abs(zp))

# generate q
mixture_comp1 <- function(x) rnorm(x, mean = -0.5, sd = 0.5)
mixture_comp2 <- function(x) rnorm(x, mean = 2, sd = 1)
q <- c(mixture_comp1(n1p), mixture_comp2(n-n1p))

n_indep <- n
res <- flexible_cfdr(p, q, indep_index = 1:n_indep)
pv_plot(p = res[[1]]$p, q = res[[1]]$q, v = res[[1]]$v)
```

---

**stratified_qqplot**  
*Stratified Q-Q plot.*

**Description**
Stratified Q-Q plot.

**Usage**
```r
stratified_qqplot(
data_frame,
prin_value_label,
cond_value_label = NULL,
thresholds = c(1, 0.1, 0.01, 0.001, 1e-04)
)
```

**Arguments**
- `data_frame`  
data.frame containing p-values and auxiliary data values
- `prin_value_label`  
label of principal p-value column in `data_frame`
- `cond_value_label`  
label of conditional trait column in `data_frame`
- `thresholds`  
threshold values to define strata

**Details**
Can be used to investigate the relationship between p and q  
Note that this function does not do the heavy lifting of styling the plot’s aesthetics.
Description

A data frame containing the rsID, chromosome (CHR19) and base pair position (BP19) in hg19, reference allele (REF), alternative allele (ALLT), type 1 diabetes GWAS p-value (T1D_pval), minor allele frequency (MAF), LDAK weight (LDAK_weight), rheumatoid arthritis GWAS p-value (RA_pval), binary regulatory factor binding site overlap (DGF), average H3K27ac fold change value in T1D-relevant cell types (H3K27ac) for 113,543 SNPs in the T1D GWAS (https://www.nature.com/articles/ng.3245)

Usage

`T1D_application_data`

Format

A data frame with 113543 rows and 11 variables:

Details

Minor allele frequencies estimated from the CEU sub-population samples in the 1000 Genomes Project Phase 3 data set. Missing values were replaced by drawing samples from the empirical distribution of MAFs
Index

* datasets
  T1D_application_data, 11

binary_cfdr, 2
corr_plot, 3
flexible_cfdr, 4
log10pv_plot, 6
match_ind_maf, 7
parameters_in_locfdr, 7
pv_plot, 9
stratified_qqplot, 10
T1D_application_data, 11