Package 'sffdr'

March 31, 2026

Type Package

Title Surrogate Functional False Discovery Rates for Genome-Wide Association Studies

Version 1.1.2

Maintainer Andrew Bass <ab3105@cam.ac.uk>

Description Pleiotropy-informed significance analysis of genome-wide association studies with surrogate functional false discovery rates (sfFDR). The sfFDR framework adapts the fFDR to leverage informative data from multiple sets of GWAS summary statistics to increase power in study while accommodating for linkage disequilibrium. sfFDR provides estimates of key FDR quantities in a significance analysis such as the functional local FDR and q-value, and uses these estimates to derive a functional p-value for type I error rate control and a functional local Bayes' factor for post-GWAS analyses (e.g., fine mapping and colocalization).

URL https://github.com/ajbass/sffdr

License LGPL

Encoding UTF-8

VignetteBuilder knitr

LazyData true

Depends R (>= 4.1.0)

Imports locfit, splines, ggplot2 (>= 3.5.1), patchwork (>= 1.3.0), qvalue, fastglm, withr, Rcpp

LinkingTo Rcpp

Suggests testthat (>= 3.0.0), knitr, rmarkdown

RoxygenNote 7.3.3

NeedsCompilation yes

Author Andrew Bass [aut, cre], Chris Wallace [aut]

Repository CRAN

Date/Publication 2026-03-31 15:20:02 UTC

Contents

bmi . . . 2
decorrelate_informative . . . 3
discover_empirical_nulls . . . 4
estimate_rho_overlap . . . 4
fpi0est . . . 5
fpvalues . . . 7
kernelEstimator . . . 8
monoSmooth_conditional . . . 10
monoSmooth_pava . . . 11
overlap_null_density . . . 11
pi0_model . . . 12
plot.sffdr . . . 13
run_empirical_overlap_correction . . . 14
sffdr . . . 15

Index **19**

`bmi` *Subset of p-values from the UK Biobank*

Description

A dataset containing a subset of p-values from the UK Biobank.

Format

A data frame with 10,000 rows and 4 columns:

bmi Body mass index

bfp Body fat percentage

cholesterol Cholesterol

triglycerides Triglycerides

Examples

```
# Import data
data(bmi)

# Separate main p-values and informative p-values
p <- sumstats$bmi
z <- as.matrix(sumstats[, -1])
```

`decorrelate_informative` *Soft whitening of informative trait z-scores*

Description

Removes overlap-induced correlation from informative trait z-scores (‘z_y‘) without altering primary p-values. The correction is scaled by the primary trait's local false discovery rate (lfdr) to preserve true biological signal while removing shared noise.

Usage

```
decorrelate_informative(z_x, z_y, p_x, p_y, rho_o = NULL, lfdr_x = NULL)
```

Arguments

`z_x`	Numeric vector; primary trait z-scores (used for weighting).
`z_y`	Numeric vector; informative trait z-scores (transformed).
`p_x`	Numeric vector; primary trait p-values.
`p_y`	Numeric vector; informative trait p-values.
`rho_o`	Numeric scalar; sample overlap correlation. If NULL, estimated via `estimate_rho_overlap`. Supplying LDSC intercepts is recommended.
`lfdr_x`	Numeric vector; pre-computed marginal lfdr for primary trait. If NULL, computed internally via `qvalue::lfdr`.

Value

A list containing:

z2 Numeric vector; decorrelated informative trait z-scores.

p2 Numeric vector; decorrelated informative trait p-values.

lfdr_x Numeric vector; marginal lfdr values used for weighting.

rho_o Numeric scalar; overlap correlation used.

See Also

`estimate_rho_overlap`, `sffdr`

Examples

```
## Not run:
dec    <- decorrelate_informative(z_x, z_y, p_x, p_y)
mpi0   <- pi0_model(as.matrix(dec$p2))
fpi0   <- fpi0est(p_x, pi0_model_obj = mpi0)
result <- sffdr(p_x, fpi0 = fpi0$fpi0, surrogate = dec$p2)

## End(Not run)
```

discover_empirical_nulls *Discover Empirical Negative Controls*

Description

Uses marginal local FDRs to dynamically identify genes that are highly likely to be null in both traits, serving as empirical negative controls for batch effect/overlap correlation estimation.

Usage

```
discover_empirical_nulls(p1, p2, threshold = 0.95, min_genes = 500)
```

Arguments

p1	Numeric vector; primary trait p-values.
p2	Numeric vector; surrogate trait p-values.
threshold	Numeric; the lfdr threshold for confidence (default 0.95).
min_genes	Integer; the minimum number of nulls required for a stable covariance estimate.

Value

Integer vector of indices representing empirical null genes.

estimate_rho_overlap *Estimate sample overlap correlation*

Description

Computes the empirical null correlation between two traits due to sample overlap. Whenever possible, supplying an external rho_o (e.g., from LDSC intercepts) is recommended over empirical estimation.

Usage

```
estimate_rho_overlap(
  z1,
  z2,
  p1,
  p2,
  lfdry = NULL,
  method = "double_null",
  thresh = 0.5
)
```

Arguments

z1, z2	Numeric vectors; z-scores for traits 1 and 2.
p1, p2	Numeric vectors; p-values for traits 1 and 2.
lfdry	Numeric vector; marginal lfdr for trait 2 (required only for method = "weighted_cov").
method	Character; the estimation strategy to use: • "weighted_cov": Weighted least squares via lfdr. • "double_null": Pearson correlation strictly on the double-null stratum. • "mom_scaled": Method of moments (scaled lambdas). • "mom_rank": Method of moments (rank-normalised).
thresh	Numeric scalar; lfdr threshold for defining the double-null stratum in the "double_null" method. Default is 0.5.

Value

Numeric scalar representing the estimated overlap correlation.

---

fpi0est *Estimate Functional Proportion of Null Tests*

Description

Estimates the functional proportion of null tests (pi0) using a GLM approach with a constrained binomial family. This function fits models across multiple lambda thresholds and selects the optimal estimate via MISE minimization.

Usage

```
fpi0est(
  p,
  pi0_model_obj = NULL,
  z = NULL,
  pi0_model = NULL,
  indep_snps = NULL,
  weights = NULL,
  lambda = seq(0.05, 0.95, 0.05),
  constrained.p = TRUE,
  tol = 1e-09,
  maxit = 200,
  ncores = 1L,
  verbose = TRUE,
  ...
)
```

Arguments

p	Numeric vector of p-values.
pi0_model_obj	Optional list object returned by pi0_model. Must contain fmod (formula) and zt (rank-transformed covariates). If provided, z and pi0_model are extracted automatically. Default is NULL.
z	Optional data frame or matrix of rank-transformed covariates. Required if pi0_model_obj is not provided. Default is NULL.
pi0_model	Optional formula (as character string or formula object). Required if pi0_model_obj is not provided. Default is NULL.
indep_snps	Optional logical vector indicating independent SNPs for model fitting. Default is NULL (all SNPs used).
weights	Optional numeric vector of weights for density estimation. For GWAS data, this should be inverse LD scores. If these are available then you don't have to use the indep_snps argument, as the weights will effectively prioritize independent SNPs in the fitting process.
lambda	Numeric vector of lambda thresholds. Default is seq(0.05, 0.95, 0.05).
constrained.p	Logical; use constrained binomial family. Default is TRUE.
tol	Numeric; convergence tolerance. Default is 1e-9.
maxit	Integer; maximum iterations. Default is 200.
ncores	Integer; number of cores for parallel lambda fitting via parallel::mclapply. Fork-based parallelism (Unix/macOS only) - on Windows this falls back to sequential execution. Default is 1L.
verbose	Logical; print progress messages. Default is TRUE.
...	Additional arguments passed to fastglm.

Details

Algorithm:

1. For each lambda threshold, fit a binomial GLM: $P(p \geq \lambda | z)$
2. Use constrained binomial family to ensure predictions in (0, 1)
3. Select optimal lambda via MISE minimization

Usage Patterns:

Pattern 1 (Recommended): Use output from pi0_model

```
mpi0 <- pi0_model(z)
# Clean syntax:
fpi0_out <- fpi0est(p, mpi0)
```

Pattern 2: Manually specify formula and covariates

```
fpi0_out <- fpi0est(p, z = z_matrix, pi0_model = formula_obj)
```

Value

An object of class `fpi0` (a list) containing:

fpi0 Numeric vector of functional pi0 estimates for each test.

tableLambda A data frame summarizing results for each lambda value.

MISE The Mean Integrated Squared Error (MISE) for the chosen model.

lambda The selected optimal lambda value.

Examples

```
# Import data
data(bmi)

# Separate main p-values and conditioning p-values
p <- sumstats$bmi
z <- as.matrix(sumstats[, -1])

# Apply pi0_model to create model (uses adaptive knot selection)
fmod <- pi0_model(z)

# Estimate functional pi0
fpi0_out <- fpi0est(p, fmod)
fpi0 <- fpi0_out$fpi0

# Apply sffdr
sffdr_out <- sffdr(p, fpi0)
```

---

fpvalues *Functional p-values*

Description

Calculate functional p-values from functional local FDRs.

Usage

```
fpvalues(lfdr, p = NULL)
```

Arguments

`lfdr`	A vector of functional local FDRs.
`p`	A vector of p-values for ranking purposes. Default is NULL.

Value

A list containing:

fp	Functional p-values.
fq	Functional q-values.

`kernelEstimator` *Kernel Density Estimation for GWAS P-values*

Description

Performs kernel density estimation on p-values (univariate) or joint p-value/covariate pairs (bivariate) using local regression on the probit scale. The estimator is optimized for GWAS data with linkage disequilibrium (LD) and uses adaptive downsampling to prioritize signal-rich regions while maintaining computational efficiency.

Usage

```
kernelEstimator(
  x,
  eval.points = x,
  epsilon = .Machine$double.xmin,
  epsilon.max = 1 - 1e-04,
  maxk = 5e+05,
  maxit = 200,
  target_null = 1e+05,
  trim = 0,
  nn = NULL,
  tail_threshold = -2,
  weights = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

x	Numeric vector of p-values (for 1D density) or a 2-column matrix where the first column contains an informative covariate/surrogate and the second column contains the p-values. All p-values must be in (0, 1) and the covariate/surrogate must be rank-transformed to be (0,1].
eval.points	Points at which to evaluate the density estimate. Defaults to `x`. For custom evaluation points, must match the dimensionality of `x` (vector or 2-column matrix).
epsilon	Lower bound for p-values to prevent numerical issues. P-values below this are clamped to `epsilon`. Default: `.Machine$double.xmin`.
epsilon.max	Upper bound for p-values. P-values above this are clamped to `epsilon.max`. Default: `1 - 1e-4`.

maxk	Maximum number of fitting points passed to locfit. Increase for very large datasets. Default: 500000.
maxit	Maximum number of iterations for local regression fitting. Default: 200.
target_null	Maximum number of null SNPs to include in the weighted fit (bivariate case only). SNPs in the signal-enriched tail (defined by tail_threshold) are always retained; null SNPs are downsampled to this target and upweighted accordingly. Default: 100000.
trim	Numeric in [0, 1); if > 0, flattens the density for p-values in (1 - trim, 1) to reduce boundary artifacts from p-values artificially clumped near 1. Only applies to the p-value dimension. Default is 0 (no trimming).
nn	Nearest-neighbor bandwidth parameter for locfit, expressed as a fraction of the data. If NULL (default), automatically determined based on effective sample size to span approximately 5000 neighbors (which corresponds to multiple LD blocks in GWAS). Larger values increase smoothing.
tail_threshold	Z-score threshold on the probit-transformed covariate scale (bivariate case only). SNPs with z < tail_threshold are treated as signal-enriched and prioritized in the adaptive fit. Default: -2 (approximately 2.3% of standard normal distribution).
weights	Optional numeric vector of weights for density estimation. For GWAS data, this should be inverse LD scores.
verbose	Logical; whether to print progress messages during fitting. Default is TRUE.
...	Additional arguments passed to lp for controlling local polynomial fitting (e.g., deg, kern).

Details

The function implements a multi-stage density estimation procedure:

1. **Probit transformation**: P-values are transformed to the normal scale via qnorm to stabilize variance and handle extreme values.
2. **Adaptive downsampling (bivariate only)**: To handle large GWAS datasets efficiently, the null region (where the covariate suggests low signal) is downsampled to target_null SNPs, with inverse-probability weighting to preserve the density. Signal-enriched SNPs (tail) are always retained.
3. **Cascade fitting**: Multiple fitting strategies are attempted in sequence, with decreasing resolution and increasing robustness, until a valid fit is obtained.
4. **Jacobian correction**: Density estimates are transformed back to the original p-value scale using the Jacobian of the probit transformation.

The nearest-neighbor bandwidth nn controls smoothing and LD robustness. By targeting ~5000 neighbors (default), the estimator naturally averages over multiple LD blocks (~30-50 blocks in European ancestry populations), reducing spurious local structure while preserving true signal-covariate relationships.

Value

A data frame with columns:

x Evaluation points (original scale).

fx Estimated density at evaluation points (original scale).

s Evaluation points on probit scale (`qnorm(x)`).

fs Estimated density on probit scale.

The returned object has an attribute `"lfit"` containing the fitted `locfit` object for diagnostics.

See Also

`locfit`, `sffdr`

Examples

```
## Not run:
# 1D density estimation
p <- runif(10000, 0, 1)
dens <- kernelEstimator(p)
plot(dens$x, dens$fx, type = "l")

# 2D density with informative covariate
p <- runif(10000, 0, 1)
z <- runif(10000)  # rank-norm transformed covariate
x_mat <- cbind(p, z)
dens <- kernelEstimator(x_mat)

## End(Not run)
```

`monoSmooth_conditional`

Fast Conditional Monotonic Smoothing

Description

Enforces strictly non-increasing density conditional on surrogate bins. Note: Data must be sorted by (group, pvalue) before passing.

Usage

```
monoSmooth_conditional(pvalue, density, group)
```

Arguments

`pvalue`	Numeric vector of p-values.
`density`	Numeric vector of estimated densities.
`group`	Integer vector of surrogate bins.

Value

A numeric vector of smoothed densities.

`monoSmooth_pava` *Fast Conditional PAVA (Decreasing)*

Description

Fast Conditional PAVA (Decreasing)

Usage

```
monoSmooth_pava(pvalue, density, group)
```

Arguments

`pvalue`	A numeric vector of p-values. Must be sorted ascendingly within each group.
`density`	A numeric vector of initial density estimates (e.g., from locfit).
`group`	An integer vector denoting the surrogate bin/stratum for each observation.

Value

A numeric vector of smoothed, monotonically non-increasing densities.

`overlap_null_density` *Compute conditional null density under sample overlap*

Description

Computes the conditional null density f(z1 | H1=0, z2) for a primary trait (z1) given a surrogate trait (z2) when the two studies share samples.

Usage

```
overlap_null_density(
  z1,
  z2,
  lfdry,
  rho_o = NULL,
  p1 = NULL,
  p2 = NULL,
  se2 = NULL,
  empirical_null_var = TRUE,
  ...
)
```

Arguments

z1	Numeric vector; z-scores for the primary trait.
z2	Numeric vector; z-scores for the surrogate trait.
lfdry	Numeric vector; marginal local FDR of the surrogate trait.
rho_o	Numeric scalar; expected sample overlap correlation. If NULL (default), it is estimated internally via estimate_rho_overlap.
p1	Numeric vector; primary trait p-values. Required if rho_o = NULL.
p2	Numeric vector; surrogate trait p-values. Required if rho_o = NULL.
se2	Numeric vector; standard errors of the surrogate trait effects. If provided, enables per-variant Empirical Bayes shrinkage (recommended for GWAS). If NULL, uses global shrinkage (recommended for RNA-seq).
empirical_null_var	Logical; if TRUE (default), estimates null variance empirically from the double-null center to absorb genomic inflation.
...	Additional arguments passed to estimate_rho_overlap.

Value

A numeric vector representing the conditional null density for each observation. Pass this to the f0 argument of sffdr.

---

pi0_model *Build Model for Functional Proportion of Null Tests (fpi0)*

Description

Generates a natural spline model for the functional proportion of null tests (fpi0) by adaptively selecting knots based on an FDR threshold.

Usage

```
pi0_model(
  z,
  indep_snps = NULL,
  weights = NULL,
  fdr_threshold = 0.25,
  min_discoveries = 150,
  min_snps_per_knot = 100,
  n_knots = 5,
  verbose = TRUE,
  seed = 2026
)
```

Arguments

z	Matrix/data.frame of p-values (rows=tests, cols=traits).
indep_snps	Logical, numeric, or character vector indicating independent SNPs (training subset). If NULL, uses all tests. Default: NULL.
weights	Optional numeric vector of weights (e.g., inverse LD scores) for weighted spline fitting. Default: NULL.
fdr_threshold	FDR threshold for signal definition. Default: 0.25.
min_discoveries	Min (LD-independent) significant hits required to include trait. Default: 150.
min_snps_per_knot	Base minimum significant SNPs per knot interval. Default: 100. Dynamically scales up for larger datasets to prevent overfitting.
n_knots	Target knot count. Default: 5. Automatically reduced if insufficient discoveries (via min_snps_per_knot) or capped by max_knots.
verbose	Print selection details. Default: TRUE.
seed	Optional seed for reproducibility. Default: 2026.

Details

Independent SNPs determine knot placement; all SNPs train the model to capture signal shape. For smaller datasets (<100K), consider reducing min_snps_per_knot and min_discoveries to improve knot placement.

Value

List containing:

fmod	Model formula (using splines::ns)
zt	Data frame of globally rank-transformed p-values

---

plot.sffdr *Plotting function for sffdr object*

Description

Graphical display of the sffdr object

Usage

```
## S3 method for class 'sffdr'
plot(x, rng = c(0, 5e-08), ...)
```

Arguments

x	A sffdr object.
rng	Significance region to show. Optional.
...	Additional arguments. Currently unused.

Value

Nothing of interest.

Author(s)

Andrew J. Bass

See Also

sffdr

Examples

```
## Not run:
# import data
data(bmi)

# Separate main p-values and conditioning p-values
p <- sumstats$bmi
z <- as.matrix(sumstats[, -1])

# Apply pi0_model to create model
fmod <- pi0_model(z)

# Estimate functional pi0
fpi0_out <- fpi0est(p, z = fmod$zt, pi0_model = fmod$fmod)
fpi0 <- fpi0_out$fpi0

# Apply sffdr
sffdr_out <- sffdr(p, fpi0)

# Plot significance results
plot(sffdr_out, rng = c(0, 5e-4))

## End(Not run)
```

---

run_empirical_overlap_correction

Run Overlap Correction using Data-Driven Nulls

Description

Run Overlap Correction using Data-Driven Nulls

Usage

```
run_empirical_overlap_correction(z1, z2, p1, p2)
```

Arguments

z1	Numeric vector; primary trait z-scores.
z2	Numeric vector; surrogate trait z-scores.
p1	Numeric vector; primary trait p-values.
p2	Numeric vector; surrogate trait p-values.

Value

A list containing the estimated rho_o, the conditional f0 density, and the indices of the empirical nulls used.

---

sffdr *Surrogate Functional False Discovery Rate Analysis*

Description

Estimate functional p-values, q-values, and local false discovery rates (lfdr) for GWAS data leveraging summary statistics from related traits. Functional p-values map from the functional q-value (FDR-based measure) to a p-value for type I error rate control, accounting for pleiotropy that impacts the prior probability of association.

Usage

```
sffdr(
  p.value,
  fpi0,
  surrogate = NULL,
  weights = NULL,
  epsilon = .Machine$double.xmin,
  nn = NULL,
  monotone = TRUE,
  monotone_method = c("min", "pava"),
  fp_ties = TRUE,
  seed = 2026,
  verbose = TRUE,
  ...
)
```

Arguments

p.value	Numeric vector of p-values to analyze.
fpi0	Numeric vector of functional pi0 estimates, obtained from fpi0est. Values must be in [0, 1].
surrogate	Optional numeric vector (same length as p.value) used as a surrogate variable for compression. If NULL (default), uses fpi0 as the surrogate.
weights	Optional numeric vector of weights for density estimation. For GWAS data, this should be inverse LD scores.
epsilon	Numeric; lower bound for p-value clamping during density estimation. Default is .Machine$double.xmin.
nn	Numeric; nearest-neighbor bandwidth for kernelEstimator. If NULL (default), automatically selected as ~5000 neighbors.
monotone	Logical; if TRUE, enforces monotonicity of the estimated density with respect to p-values within bins of the surrogate variable. Default is TRUE.
monotone_method	Character; method for enforcing monotonicity. Options are "min" (running minimum) or "pava" (Pool Adjacent Violators Algorithm). Default is "min". We do not recommend "pava" for GWAS data sets due to LD, but it may be better for expression data.
fp_ties	Logical; whether to break ties in functional p-values using the original p-value ordering. Default is TRUE.
seed	Integer; random seed for reproducibility of rank tie-breaking. Default is 2026.
verbose	Logical; print progress messages. Default is TRUE.
...	Additional arguments passed to kernelEstimator.

Details

The surrogate functional FDR (sfFDR) methodology extends the functional FDR framework to leverage multiple informative variables (e.g., functional annotations, GWAS summary statistics) for increased power while controlling false discovery rates.

Workflow:

1. Estimate functional pi0 (proportion of nulls) using fpi0est
2. Call sffdr() with p-values and estimated functional pi0
3. Use returned functional p-values/q-values/local FDRs for significance testing

Surrogate Variable: If not specified, the estimated functional pi0 is used as the surrogate variable.

Interpretation note: Functional p-values reflect the global ranking of all SNPs by the functional local FDR. A SNP with a small functional p-value but large flfdr (> 0.5) has weak individual evidence but ranks favorably relative to null SNPs. This happens when there is strong prior evidence (fpi0) of the SNP being non-null. Thus, users should also consider flfdr (and fqvalues) alongside fpvalues when assessing individual SNP evidence. SNPs with flfdr > 0.5 should be interpreted cautiously regardless of their functional p-value.

Value

An S3 object of class "sffdr" containing:

call	The function call.
pvalues	Original p-values.
fpvalues	Functional p-values.
fqvalues	Functional q-values.
flfdr	Functional local false discovery rates.
fpi0	Functional pi0 estimates.
fx	Joint density estimates at observed (p-value, surrogate) pairs.

Author(s)

Andrew J. Bass

See Also

fpi0est, plot.sffdr, kernelEstimator

Examples

```
## Not run:
# Import data
data(bmi)

# Separate main p-values and conditioning p-values
p <- sumstats$bmi
z <- as.matrix(sumstats[, -1])

# Apply pi0_model to create model
# (note: use indep_snps argument to specify independent SNPs for training)
fmod <- pi0_model(z)

# Estimate functional pi0
# (note: use indep_snps argument to specify independent SNPs for training)
fpi0_out <- fpi0est(p, fmod)
fpi0 <- fpi0_out$fpi0

# Apply sffdr
sffdr_out <- sffdr(p, fpi0)

# Plot significance results
plot(sffdr_out)

# Extract functional quantities
fp <- sffdr_out$fpvalues
fq <- sffdr_out$fqvalues
flfdr <- sffdr_out$flfdr

## End(Not run)
```

Index

∗ **datasets**
bmi, 2
∗ **plot**
plot.sffdr, 13
∗ **sffdr**
sffdr, 15

bmi, 2

decorrelate_informative, 3
discover_empirical_nulls, 4

estimate_rho_overlap, *3*, 4

fastglm, *6*
fpi0est, 5, *16, 17*
fpvalues, 7

kernelEstimator, 8, *16, 17*

locfit, *9, 10*
lp, *9*

monoSmooth_conditional, 10
monoSmooth_pava, 11

overlap_null_density, 11

pi0_model, *6*, 12
plot, *(*plot.sffdr*)*, 13
plot.sffdr, 13, *17*

run_empirical_overlap_correction, 14

sffdr, *3*, *10*, *14*, 15
sumstats *(*bmi*)*, 2