

# Package: sffdr (via r-universe)

October 20, 2026

**Type** Package

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

**Version** 1.1.4

**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** MIT + file LICENSE

**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

**Config/pak/sysreqs** libicu-dev

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

**Date/Publication** 2026-10-20 18:45:47 UTC

**RemoteUrl** <https://github.com/ajbass/sffdr>  
**RemoteRef** HEAD  
**RemoteSha** 0153e7017651b06798bf01f40087ebfe4a04c219

**Contents**

|                                   |           |
|-----------------------------------|-----------|
| bmi . . . . .                     | 2         |
| decorrelate_informative . . . . . | 3         |
| estimate_overlap . . . . .        | 4         |
| fpi0est . . . . .                 | 5         |
| fpvalues . . . . .                | 7         |
| kernelEstimator . . . . .         | 8         |
| monoSmooth_conditional . . . . .  | 11        |
| monoSmooth_pava . . . . .         | 11        |
| pi0_model . . . . .               | 12        |
| plot.sffdr . . . . .              | 13        |
| sffdr . . . . .                   | 14        |
| <b>Index</b>                      | <b>18</b> |

---

|     |                                               |
|-----|-----------------------------------------------|
| bmi | <i>Subset of p-values from the UK Biobank</i> |
|-----|-----------------------------------------------|

---

**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

*Decorrelate the informative studies from the primary study*


---

## Description

When the primary study shares samples with the informative studies, their null z-scores are correlated. Used as covariates in [pi0\\_model](#), the informative p-values then predict the primary's null tests, which inflates false discoveries. `decorrelate_informative()` removes the primary's noise from each informative study,

$$u_k = (z_k - r_k z_1) / \sqrt{1 - r_k^2},$$

where  $r_k$  is the null correlation of study  $k$  with the primary. Under the primary's null,  $u$  is independent of  $z_1$  whatever the informative studies' effects, so the primary's p-values are analysed unchanged with the decorrelated p-values as covariates.

## Usage

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

## Arguments

|                         |                                                                                                                                                                                    |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>z_x</code>        | Numeric vector; the primary study's z-scores.                                                                                                                                      |
| <code>z_y</code>        | Numeric vector or matrix; signed z-scores of the informative studies (one column per study), in the same order as <code>z_x</code> .                                               |
| <code>p_x, p_y</code>   | Ignored; kept for compatibility with <code>sffdr</code> 1.1.2.                                                                                                                     |
| <code>rho_o</code>      | Optional overlap correlations of the primary with each informative study (one number per study). If <code>NULL</code> (default), estimated with <a href="#">estimate_overlap</a> . |
| <code>lfdr_x</code>     | Ignored. <code>sffdr</code> 1.1.2 scaled the correction by the primary's local FDR, which left part of the overlap in the covariates.                                              |
| <code>indep_snps</code> | Optional logical or integer vector selecting approximately independent SNPs, used to estimate <code>rho_o</code> .                                                                 |

## Details

The correction needs only the overlap correlation of each informative study with the primary: cross-trait LD score regression intercepts are the recommended source for GWAS; otherwise they are estimated by [estimate\\_overlap](#). Errors in them enter at second order: in simulations, a correlation of 0.5 given as 0.35 or 0.65 left the FDR within 0.005 of its value at the true correlation.

**Value**

A list with

**z2** decorrelated z-scores of the informative studies (vector or matrix, as **z\_y**)

**p2** their p-values, for use as covariates

**rho\_o** the overlap correlations used

**See Also**

[estimate\\_overlap](#)

**Examples**

```
set.seed(1)
n <- 20000
e1 <- rnorm(n)
e2 <- 0.5 * e1 + sqrt(1 - 0.5^2) * rnorm(n)
z_x <- e1 + ifelse(runif(n) < 0.05, 3, 0)      # primary
z_y <- e2 + ifelse(runif(n) < 0.1, 3, 0)      # informative, same samples
dec <- decorrelate_informative(z_x, z_y, rho_o = 0.5)

p_x <- 2 * pnorm(-abs(z_x))
mpi0 <- pi0_model(as.matrix(dec$p2))
fpi0 <- fpi0est(p_x, pi0_model_obj = mpi0)
res <- sffdr(p_x, fpi0 = fpi0$fpi0)
```

---

estimate\_overlap

*Estimate the overlap correlation between studies*

---

**Description**

Estimates the null correlation of z-scores between studies that share samples, for every pair of `cbind(z, z_informative)`. For each pair it uses the mean product of z-scores inside the box  $|z_1|, |z_2| < t$ , which is unaffected by signal present in only one of the two studies, and inverts it through the exact truncated bivariate normal moment. Cross-trait LD score regression intercepts are preferable for GWAS when available.

**Usage**

```
estimate_overlap(z, z_informative, indep_snps = NULL, threshold = 1.5)
```

**Arguments**

|                            |                                                                                                                 |
|----------------------------|-----------------------------------------------------------------------------------------------------------------|
| <code>z</code>             | Numeric vector; the primary study's z-scores.                                                                   |
| <code>z_informative</code> | Numeric vector or matrix of the informative studies' z-scores.                                                  |
| <code>indep_snps</code>    | Optional logical or integer vector of approximately independent SNPs to use.                                    |
| <code>threshold</code>     | Numeric; the box half-width $t$ (default 1.5). Smaller boxes are less affected by moderate effects and noisier. |

**Value**

The  $(K + 1) \times (K + 1)$  correlation matrix, primary first.

**See Also**

[decorrelate\\_informative](#)

---

fpi0est

---

*Estimate Functional Proportion of Null Tests*


---

**Description**

Estimates the functional proportion of null tests ( $\pi_0$ ) 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**

|                            |                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>p</code>             | Numeric vector of p-values.                                                                                                                                                                                                                                                                                                                                     |
| <code>pi0_model_obj</code> | Optional list object returned by <code>pi0_model</code> . Must contain <code>fmod</code> (formula) and <code>zt</code> (rank-transformed covariates). If provided, <code>z</code> and <code>pi0_model</code> are extracted automatically. Default is <code>NULL</code> .                                                                                        |
| <code>z</code>             | Optional data frame or matrix of rank-transformed covariates. Required if <code>pi0_model_obj</code> is not provided. Default is <code>NULL</code> .                                                                                                                                                                                                            |
| <code>pi0_model</code>     | Optional formula (as character string or formula object). Required if <code>pi0_model_obj</code> is not provided. Default is <code>NULL</code> .                                                                                                                                                                                                                |
| <code>indep_snps</code>    | Optional independent SNPs for model fitting: a logical vector with one entry per p-value, row indices, or SNP ids matched against <code>rownames(z)</code> . Default is <code>NULL</code> (all SNPs used).                                                                                                                                                      |
| <code>weights</code>       | Optional numeric vector of weights in (0, 1], one per p-value, for the model fit. For GWAS data, this should be inverse LD scores (values above 1 are clamped to 1 with a warning). If these are available then you don't have to use the <code>indep_snps</code> argument, as the weights will effectively prioritize independent SNPs in the fitting process. |
| <code>lambda</code>        | Numeric vector of lambda thresholds in (0, 1). Default is <code>seq(0.05, 0.95, 0.05)</code> .                                                                                                                                                                                                                                                                  |
| <code>constrained.p</code> | Logical; use constrained binomial family. Default is <code>TRUE</code> .                                                                                                                                                                                                                                                                                        |
| <code>tol</code>           | Numeric; convergence tolerance. Default is <code>1e-9</code> .                                                                                                                                                                                                                                                                                                  |
| <code>maxit</code>         | Integer; maximum iterations. Default is 200.                                                                                                                                                                                                                                                                                                                    |
| <code>ncores</code>        | Integer; number of cores for parallel lambda fitting via <code>parallel::mclapply</code> . Fork-based parallelism (Unix/macOS only) - on Windows this falls back to sequential execution. Default is 1L.                                                                                                                                                        |
| <code>verbose</code>       | Logical; print progress messages. Default is <code>TRUE</code> .                                                                                                                                                                                                                                                                                                |
| <code>...</code>           | Additional arguments passed to <code>fastglm</code> .                                                                                                                                                                                                                                                                                                           |

**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  $\pi_0$  estimates for each test, floored at 0.01. Tests with a missing p-value or covariate get 1.

**tableLambda** A data frame with one row per lambda value: the MISE, the mean fitted  $\pi_0$  and whether it was chosen.

**MISE** The Mean Integrated Squared Error (MISE) at each lambda.

**lambda** The selected optimal lambda value.

**model** The fitted model at the selected lambda (class `fastglm2`; `predict()` gives  $P(p \geq \lambda|z)$ ).

**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 in  $[0, 1]$  (values above 1 are clamped to 1).

**p** A vector of p-values for ranking purposes. Default is `NULL`.

## Details

Tests are ranked by l<sub>fdr</sub>, with ties broken by p. The functional q-value of a test is the mean l<sub>fdr</sub> of the tests ranked at or above it, and its functional p-value is the l<sub>fdr</sub> mass ranked at or above it as a fraction of the total mass. Tests with identical l<sub>fdr</sub> and p are one set: each receives the set's values, so the result does not depend on the order of the input. When p is NULL, ties are broken at random.

## Value

A list containing:

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

---

|                 |                                                    |
|-----------------|----------------------------------------------------|
| kernelEstimator | <i>Kernel Density Estimation for GWAS P-values</i> |
|-----------------|----------------------------------------------------|

---

## 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 = 5000,
  maxit = 200,
  target_null = 1e+05,
  trim = 0,
  nn = NULL,
  tail_threshold = -2,
  weights = NULL,
  verbose = TRUE,
  ...
)
```

**Arguments**

|                             |                                                                                                                                                                                                                                                                                                                                         |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>              | 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].                                                       |
| <code>eval.points</code>    | Points at which to evaluate the density estimate. Defaults to <code>x</code> . For custom evaluation points, must match the dimensionality of <code>x</code> (vector or 2-column matrix).                                                                                                                                               |
| <code>epsilon</code>        | Lower bound for p-values to prevent numerical issues. P-values below this are clamped to <code>epsilon</code> . Default: <code>.Machine\$double.xmin</code> .                                                                                                                                                                           |
| <code>epsilon.max</code>    | Upper bound for p-values. P-values above this are clamped to <code>epsilon.max</code> . Default: <code>1 - 1e-4</code> .                                                                                                                                                                                                                |
| <code>maxk</code>           | Space allocated for <code>locfit</code> 's evaluation structure. It does not affect the fit: when it is too small, <code>locfit</code> stops with "out of vertex space" and the fit is repeated with ten times the space. Default: 5000 (about 35 MB; versions before 1.1.4 allocated 2.7 GB).                                          |
| <code>maxit</code>          | Maximum number of iterations for the bivariate local likelihood fit. Default: 200. The univariate fit uses at most 20 ( <code>locfit</code> 's default), because more iterations let it diverge on tied values.                                                                                                                         |
| <code>target_null</code>    | Maximum number of null SNPs to include in the weighted fit (bivariate case only). SNPs in the signal-enriched tail (defined by <code>tail_threshold</code> ) are always retained; null SNPs are downsampled to this target and upweighted accordingly. Default: 100000.                                                                 |
| <code>trim</code>           | 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).                                                                                                                |
| <code>nn</code>             | Nearest-neighbor bandwidth parameter for <code>locfit</code> , expressed as a fraction of the rows the density is fitted on. If NULL (default), chosen so that the window spans approximately 5000 of those rows (which corresponds to multiple LD blocks in GWAS); see Details. Larger values increase smoothing.                      |
| <code>tail_threshold</code> | Threshold on the probit-transformed p-values (bivariate case only). SNPs with $qnorm(p) < tail\_threshold$ form the signal-enriched tail, which is never downsampled. Default: -2 (p-values below about 0.023).                                                                                                                         |
| <code>weights</code>        | Optional numeric vector of prior weights in (0, 1] for the fit (values above 1 are clamped to 1).                                                                                                                                                                                                                                       |
| <code>verbose</code>        | Logical; whether to print progress messages during fitting. Default is TRUE.                                                                                                                                                                                                                                                            |
| <code>...</code>            | Additional arguments passed to <code>locfit</code> (for example <code>scale</code> ). The arguments <code>nn</code> , <code>h</code> , <code>deg</code> , <code>kern</code> , <code>ev</code> , <code>maxk</code> , <code>maxit</code> , <code>weights</code> and <code>data</code> are set by the estimator and cannot be passed here. |

**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 (p-values above the tail threshold) 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 default the window spans about 5000 of the rows the density is fitted on (the tail plus up to `target_null` null SNPs), so the estimator averages over multiple LD blocks (~30-50 blocks in European ancestry populations), reducing spurious local structure while preserving true signal-covariate relationships. Versions before 1.1.4 took the fraction relative to all SNPs, which left about 600 neighbours at one million SNPs.

In the univariate case, tied values are collapsed into one point with the corresponding multiplicity before fitting.

### 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: the covariate (rank-transformed) first, the p-values second
p <- runif(10000, 0, 1)
z <- runif(10000)
x_mat <- cbind(z, p)
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.

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**

|                                |                                                                                                                                                                                                                                            |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>z</code>                 | Numeric matrix/data.frame of p-values (rows=tests, cols=traits).                                                                                                                                                                           |
| <code>indep_snps</code>        | Independent SNPs (the training subset): a logical vector with one entry per row of <code>z</code> , row indices, or SNP ids matched against <code>rownames(z)</code> . If <code>NULL</code> , uses all tests. Default: <code>NULL</code> . |
| <code>weights</code>           | Optional numeric vector of weights in (0, 1] (e.g., inverse LD scores) for weighted spline fitting. Default: <code>NULL</code> .                                                                                                           |
| <code>fdr_threshold</code>     | FDR threshold for signal definition. Default: 0.25.                                                                                                                                                                                        |
| <code>min_discoveries</code>   | Min (LD-independent) significant hits required to place a trait's knots from its discoveries; with fewer, the trait keeps a single fallback knot. Default: 150.                                                                            |
| <code>min_snps_per_knot</code> | Base minimum significant SNPs per knot interval. Default: 100. Dynamically scales up for larger datasets to prevent overfitting.                                                                                                           |
| <code>n_knots</code>           | Target knot count. Default: 5. Automatically reduced if insufficient discoveries (via <code>min_snps_per_knot</code> ).                                                                                                                    |
| <code>verbose</code>           | Print selection details. Default: <code>TRUE</code> .                                                                                                                                                                                      |
| <code>seed</code>              | 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. Every column must hold p-values: the knots of a column are placed at the quantiles of its discoveries (`qvalue` at `fdr_threshold`), and a column with values outside [0, 1] gets the fallback knot with a warning. A null anchor knot at rank 0.1 is always added.

**Value**

List containing:

|                   |                                                  |
|-------------------|--------------------------------------------------|
| <code>fmod</code> | Model formula (using <code>splines::ns</code> )  |
| <code>zt</code>   | Data frame of globally rank-transformed p-values |

---

|                         |                                           |
|-------------------------|-------------------------------------------|
| <code>plot.sffdr</code> | <i>Plotting function for sffdr object</i> |
|-------------------------|-------------------------------------------|

---

**Description**

Graphical display of the sffdr object

**Usage**

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

**Arguments**

|                  |                                                                                                                                                           |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>   | A sffdr object.                                                                                                                                           |
| <code>rng</code> | Significance region to show (functional p-value range). Optional. If no functional p-value falls in it, the range is widened to the smallest one percent. |
| <code>...</code> | 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)
```

---

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

|                              |                                                                                                                                                                                                                                                       |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>p.value</code>         | Numeric vector of p-values to analyze, in $[0, 1]$ . Exact zeros are clamped to the smallest positive double with a warning; negative values are an error.                                                                                            |
| <code>fpi0</code>            | Numeric vector of functional $\pi_0$ estimates, obtained from <code>fpi0est</code> . Values must be in $[0, 1]$ .                                                                                                                                     |
| <code>surrogate</code>       | Optional numeric vector (same length as <code>p.value</code> ) used as a surrogate variable for compression. If NULL (default), uses <code>fpi0</code> as the surrogate.                                                                              |
| <code>weights</code>         | Ignored, with a warning (kept for compatibility with <code>sfldr</code> 1.1.2); see Details.                                                                                                                                                          |
| <code>epsilon</code>         | Numeric; lower bound for p-value clamping during density estimation. Default is <code>.Machine\$double.xmin</code> .                                                                                                                                  |
| <code>nn</code>              | Numeric; nearest-neighbor bandwidth for <code>kernelEstimator</code> , a fraction of the rows the density is fitted on. If NULL (default), the window spans ~5000 of those rows.                                                                      |
| <code>monotone</code>        | Logical; if TRUE, enforces monotonicity of the estimated density with respect to p-values within bins of the surrogate variable. Default is TRUE.                                                                                                     |
| <code>monotone_method</code> | 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. |
| <code>fp_ties</code>         | Logical; whether to break ties in functional p-values using the original p-value ordering. Default is TRUE.                                                                                                                                           |
| <code>seed</code>            | Integer; random seed for reproducibility of rank tie-breaking, the jitter and the null downsampling in the density fit. Default is 2026.                                                                                                              |
| <code>verbose</code>         | Logical; print progress messages. Default is TRUE.                                                                                                                                                                                                    |
| <code>...</code>             | Additional arguments passed to <code>kernelEstimator</code> .                                                                                                                                                                                         |

### 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  $\pi_0$  (proportion of nulls) using `fpi0est`
2. Call `sfldr()` with p-values and estimated functional  $\pi_0$
3. Use returned functional p-values/q-values/local FDRs for significance testing

Surrogate Variable: If not specified, the estimated functional  $\pi_0$  is used as the surrogate variable.

Weights and LD: The joint density is estimated over all tested SNPs, which is the set the functional p-values count errors in, so `sfldr()` no longer reweights it. Inverse-LD weights change the density

to that of independent signals; because associated SNPs have larger LD scores, this biases the local FDR (conservative with inverse-LD weights, anti-conservative with weights that are larger at associated SNPs). LD scores carry information about association, so their place is the functional `pi0` model, as a covariate (see the vignette); as inverse weights they remain available in `pi0_model` and `fpi0est` in place of an LD-pruned training set.

Perfect-LD copies: SNPs with identical p-values are copies of one signal (perfect LD). Each set of copies contributes one point to the density fit, and the density is then evaluated for every SNP, so copies do not narrow the effective window in the tail (which inflated functional p-values below  $1e-3$  before 1.1.4). Rounded p-values tie between unrelated SNPs too, almost only in the bulk of the distribution where values are dense per significant digit, and collapsing those would distort the density; so identical p-values count as copies only when they are at least half as common below  $p = 0.023$  as above it, which holds for LD copies and fails for rounding. The window itself spans about 5,000 of the rows the density is fitted on; see `kernelEstimator`.

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:

|                       |                                                                 |
|-----------------------|-----------------------------------------------------------------|
| <code>call</code>     | The function call.                                              |
| <code>pvalues</code>  | The p-values as analysed (exact zeros clamped).                 |
| <code>fpvalues</code> | Functional p-values.                                            |
| <code>fqvalues</code> | Functional q-values.                                            |
| <code>flfdr</code>    | Functional local false discovery rates.                         |
| <code>fpi0</code>     | Functional <code>pi0</code> estimates.                          |
| <code>fx</code>       | 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, [14](#)
- bmi, [2](#)
- decorrelate\_informative, [3](#), [5](#)
- estimate\_overlap, [3](#), [4](#), [4](#)
- fastglm, [6](#)
- fpi0est, [5](#), [15](#), [16](#)
- fpvalues, [7](#)
- kernelEstimator, [8](#), [15](#), [16](#)
- locfit, [9](#), [10](#)
- monoSmooth\_conditional, [11](#)
- monoSmooth\_pava, [11](#)
- pi0\_model, [3](#), [6](#), [12](#), [16](#)
- plot, (plot.sffdr), [13](#)
- plot.sffdr, [13](#), [16](#)
- qvalue, [13](#)
- sffdr, [10](#), [13](#), [14](#)
- sumstats (bmi), [2](#)