Package 'swfdr'

August 1, 2017

Title Science-Wise False Discovery Rate

Version 0.1

Description This script reimplements the core idea in David Colquhoun's fascinating paper, ``An investigation of the false discovery rate and the misinterpretation of p-values".The term ``science-wise false discovery rate" is from Jager and Leek's paper, ``An estimate of the science-wise false discovery rate and application to the top medical literature". The base script is swfdr_base.R, a simple implementation that uses base R capabilities only. Other scripts extend the base implementation by providing solutions to some exercises for the reader.

Depends R (>= 3.2.2)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

VignetteBuilder knitr

RoxygenNote 6.0.1

Roxygen list(markdown = TRUE)

URL https://github.com/natgoodman/SWFDR

BugReports https://github.com/natgoodman/SWFDR/issues

R topics documented:

`doit` *Do the work.*

Description

Runs simulation (via `dosim`), interpolates relevant columns of the simulation results at fixed p-values (via `dointerp`), and plots the results and optionally save the plots (via `doplot`).

Usage

```
doit=function()
```

Value

This function is invoked for its side-effect. It has no return value.

`init` *Initialize program parameters.*

Description

Processes parameters and stores them in global variables. Creates parameter grid, called `cases`, containing all combinations of parameters. Creates output directories if they do not exist.

Usage

```
init(prop.true = seq(0.1, 0.9, by = 0.2),
     m = 10000,
     n = 16,
     d = c(0.25, 0.5, 0.75, 1, 2),
     pwr = NA, sig.level = 0.05,
     pval.plot = c(0.001, 0.01, 0.03, 0.05, 0.1),
     scriptname = "swfdr_base",
     datadir = file.path("data", scriptname),
     figdir = file.path("figure", scriptname),
     save = F, save.rdata = save, save.txt = F, save.plot = save,
     clean = F, clean.data = clean, clean.fig = clean)
```

Arguments

`prop.true`	fraction of cases where where there is a real effect.
`m`	number of iterations.
`n`	sample size.
`d`	standardized effect size (aka Cohen's d)
`pwr`	power. if set, program adjusts `d` to achieve power.
`sig.level`	significance level for power calculation
`pval.plot`	p-values for which we plot results
`scriptname`	script name. Used to construct output directory path names.

`datadir`	path name of directory for data files.
`figdir`	path name of directory for plots.
`save`	logical. sets `save.rdata` and `save.plot`
`save.rdata`	if TRUE, save parameters and results (actually, all global variables) in RData format. The output filename is `globals.RData` in directory `datadir`.
`save.txt`	save simulation and interpolation results as tab-delimited text files. The output filenames are `sim.txt` amd `interp.txt` in directory `datadir`. **CAUTION: big & slow!**
`save.plot`	if TRUE, save plots as png files in `figdir`. The output filenames are `plot_byd.png`, `plot_byprop.png`, `plot_vsd.png`, `plot_vsprop.png`
`clean`	logical. sets `clean.data` and `clean.fig`.
`clean.data`	if TRUE, delete contents of datadir and start fresh
`clean.fig`	if TRUE, delete contents of figdir and start fresh

Details

For the default parameters, the `cases` parameter grid expands to 25 cases (5 values of prop.true x 5 values of d; all other parameters have single-valued defaults). We do 10,000 simulations for each case for a total of 250,000 simulations. This takes about 3 minutes on my small Linux server.

This function is usually called by `run`. It may be called directly if the user wishes to perform custom initialization.

Value

The `cases` data frame is returned invisibly.

Examples

```
# initialize parameters with default values
init();

# initialize parameters with default values but save results in directories data/guide01
init(save=T,datadir='data/guide01',figdir='figure/guide01');

# initialize parameters with values requiring less runtime by reducing number of simulati
init(m=1e3,d=c(0.25,0.5,1),prop.true=c(0.3,0.5,0.8));

# specify power directly and let program adjust effect size
init(m=1e3,pwr=c(0.1,0.3,0.8),prop.true=c(0.3,0.5,0.8));
```

`Plot Functions` *Plot the results*

Description

These functions operate on the `sim` and `interp` data frames produced by `dosim` and `dointerp` respectively.

Usage

```
doplot=function(save.plot=F)

plot_byprop(save.plot = F, d1 = 1, sig.level = 0.05)

plot_byd(save.plot = F, prop.true1 = 0.5, sig.level = 0.05)

plot_vsprop(save.plot = F, d1 = 1, sig.level = 0.05)

plot_vsd(save.plot = F, prop.true1 = 0.5, sig.level = 0.05)
```

Arguments

save.plot	if TRUE, save the plot. The output format is PNG. The output filename is the name of the function with .png suffix in directory figdir, eg, figure/plot_byprop.png.
d1	fixed value of d for dimension reduction. If d1 is not in the d vector, it is set d1 to max(d).
sig.level	values of p-value and FDR marked by dashed lines on the plot.
prop.true1	fixed value of prop.true for dimension reduction. If prop.true is not in the prop.true vector, it is set to the first value in prop.true.

Details

doplot is the main plot function. It calls separate functions for each of the four kinds of plot.

- plot_byprop plots FDR by prop.true for one value of d
- plot_byd plots FDR by d for one value of prop.true
- plot_vsprop plots FDR vs prop.true for one value of d at fixed p-values
- plot_vsd plots FDR vs d for one value of prop.true at fixed p-values

Each of the plot_ functions plots a different slice of theoretical and empirical FDR as a function of three variables: FDR=f(prop.true,d,pval) The functions differ in how they reduce four dimensions (FDR and the three variables) to something that can be plotted in two dimensions.

Each function starts by fixing one variable to a single value. Next, the function splits the data into groups based on a second variable. Finally, it plots each group vs. the remaining variable, using different line types (solid vs dashed) to distinguish theoretical and empirical FDR.

plot_byprop and plot_byd operate on sim; plot_vsprop and plot_vsd operate on interp.

---

run *Run the program.*

Description

Top-level function. Sets parameters (via init), does the work (via doit) and optionally saves the results (via saveit)

Usage

```
run=function(...)
```

Arguments

`...`	Parameters passed to `init`

Value

This function is invoked for its side-effect. It has no return value.

See Also

init for more information.

Examples

```
# this code block assumes your working directory is the root of the repository

source("script/swfdr_base.R");
# run default process
run();

# run default process and save results in directories data/guide01 and figure/guide01
run(save=T,datadir='data/guide01',figdir='figure/guide01');

# reduce runtime by reducing number of simulation runs and simulated cases
run(m=1e3,d=c(0.25,0.5,1),prop.true=c(0.3,0.5,0.8));

# specify power directly and let program adjust effect size
run(m=1e3,pwr=c(0.1,0.3,0.8),prop.true=c(0.3,0.5,0.8));
```

Index