Mistakes of Significance

Nathan (Nat) Goodman

January 16, 2019

A collection of R scripts and documents exploring mistakes made by significance testing. The only content at present is a blog post entitled "When You Select Significant Findings, You're Selecting Inflated Estimates" whose main point is that significance testing is a biased procedure that overestimates effect size.

THE SOFTWARE IS STILL ROUGH and SOFTWARE DOCUMENTATION NONEXISTENT. PLEASE GET IN TOUCH IF YOU NEED HELP

Overview

The program explores the bias of significance testing and why it inflates effect size. The software simulates *studies* for conditions of interest. The studies are simple two group comparisons parameterized by sample size n and population effect size d_{pop} ($d_{pop} \geq 0$). For each study, I generate two groups of random numbers, each of size n. One group, *group0*, comes from a standard normal distribution with $mean = 0$; the other, *group1*, is from a standard normal distribution with $mean = d_{pop}$. The effect size statistic is standardized difference, aka *Cohen's d*, defined as the mean of *group1* minus the mean of *group0* divided by the pooled standard deviation of the two groups. P-values are from the t-test.

> **Technical note**: The pairing of Cohen's d with the t-test is mathematically natural because both are *standardized*, meaning both are relative to the sample standard deviation. In fact, Cohen's d and the t-statistic are essentially the same statistic, related by the identities $d = t\sqrt{2/n}$ and $t = d\sqrt{n/2}$ (for my simulation scenario).

The program performs two types of simulations.

1. sim_{rand}. Randomly selects a large number of d_{pop}s and simulates one study per d_{pop}.
2. sim_{fixd}. Fixes d_{pop} to a few values of interest, sets n to a range of values, and simulates many studies for each d_{pop} and n.

The program uses sim_{fixd} to estimate the mean significant observed effect size for each d_{pop} and n. It also caculates these values analytically.

After running the simulations and calculating the mean significant observed effect sizes, the program generates the figures that appear in the blog post, as well as the specific results mentioned in the post. It stores the latter in a table for future reference.

Notation

in code	in text	meaning
`n`	n	sample size
`d.pop`	d_{pop}	population effect size; this arises in the text but not the software
`d.sdz`	d_{sdz}	standardized observed effect size, aka Cohen's d
`d`	d	variously means either d_{pop} or d_{sdz}; hopefully clear from context
`sim.rand`	sim_{rand}	simulation that randomly selects a large number of d_{pop}s and simulates one study per d_{pop}.
`sim.fixd`	sim_{fixd}	simulation that fixes d_{pop} to a few values of interest, sets n to a range of values, and simulates many studies for each d_{pop} and n
`meand.empi`	$meand_{empi}$	mean significant observed effect size estimated from simulation (sim_{fixd})
`meand.theo`	$meand_{theo}$	mean significant observed effect size calculated analytically
`pval`		p-value

in code	in text	meaning
`df`		degrees of freedom
`t`	t	t-statistic
`d0`		noncentrality parameter expressed in standardized d units; this is the presumed population effect size
`ncp`		noncentrality parameter expressed in t units
`sig.level`	α	significance level

Installation and Usage

The software is **not a package** and cannot be installed by `devtools::install_github` or related. Sorry. The simplest way to get the software is to download or clone the entire repo.

The code mostly uses base R capabilities but has a few dependencies: `RColorBrewer`, `akima`, `knitr`, and `pryr`. Since it's not a package, you have to manually install these packages if you don't already have them.

The recommended way to run the program is to `source` the file `R/run.R` into your R session; `R/run.R will source the rest. Once loaded, you can run the program by executing the statement`run()‘ as shown below.

```
## This code block assumes your working directory is the root of the distribution.

source('R/run.R');
run();
```

This runs the program in a demo-like mode that quickly generates the data and produces the figures that appear in this README document. The default computation simulates 2,500 replications and produces 22 figures and one table. The data generation takes about 30 seconds on my small Linux server; the figures take about a minute, much of which is spent rendering the plots over a remote X11 connection.

The code that creates the outout (figures and table) for this README document is in `R/doc_readme.R`.

To rerun the program from scratch you must specify `clean=T`, since by default the program reuses existing data.

```
## This code block assumes your working directory is the root of the distribution
## and you've already sourced R/readme.R into your session

run(clean=T);                 # delete data from previous run and rerun program
```

You can run each part separately by running one of the statements below.

```
## This code block assumes your working directory is the root of the distribution
## and you've already sourced R/readme.R into your session

init(doc='readme',clean=T);  # you MUST specify doc to run the program in pieces
dosim();                     # run simulations
domeand();                   # calculate mean significant observed effect sizes
dodoc();                     # generate data and figures
dodoc(save.out=F);           # generate data and figures without saving them
dodoc(figscreen=F);          # generate and save figures without plotting to screen. faster!
```

The program can also generate the data and figures for the blog post associated with the project. To generate these, execute `run()` with a suitable `doc` argument as shown below.

```
## This code block assumes your working directory is the root of the distribution.
```

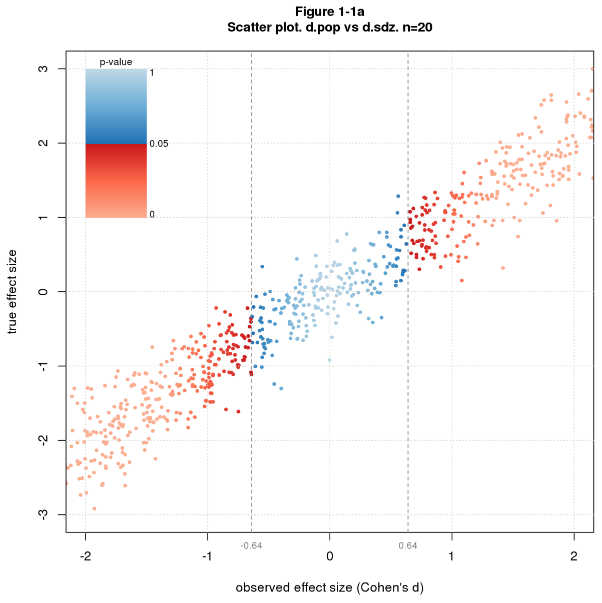
```r
source('R/run.R');
run(doc='ovrfx');                # generate data and figures for blog post
```

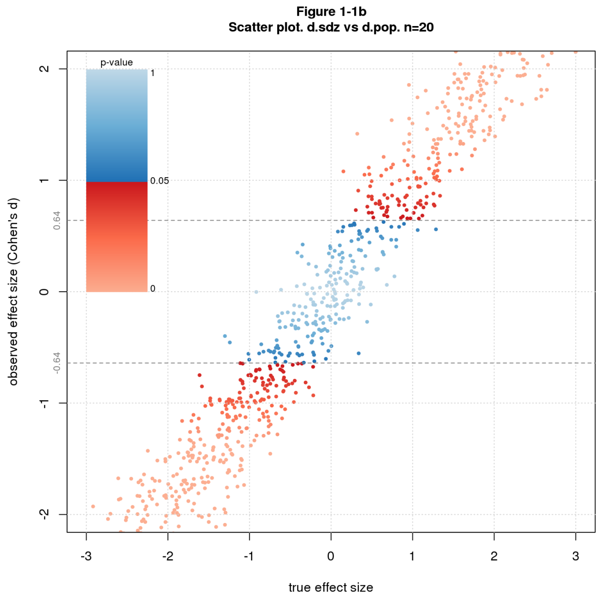
Figures

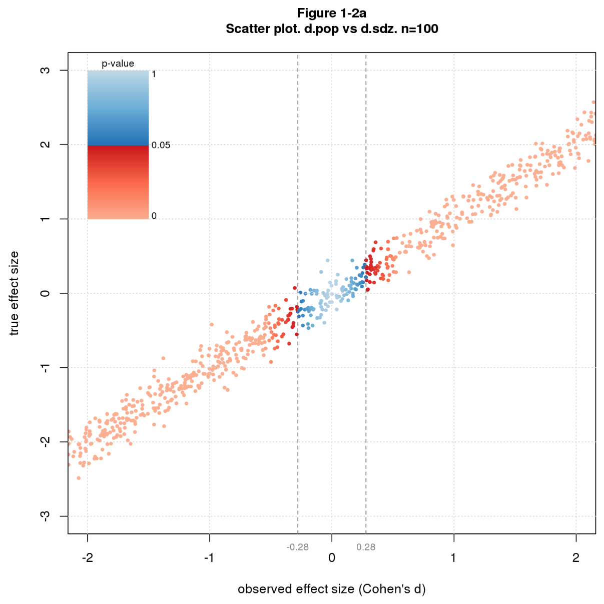
The default mode produces figures that illustrate the kinds of graphs the program can produce.

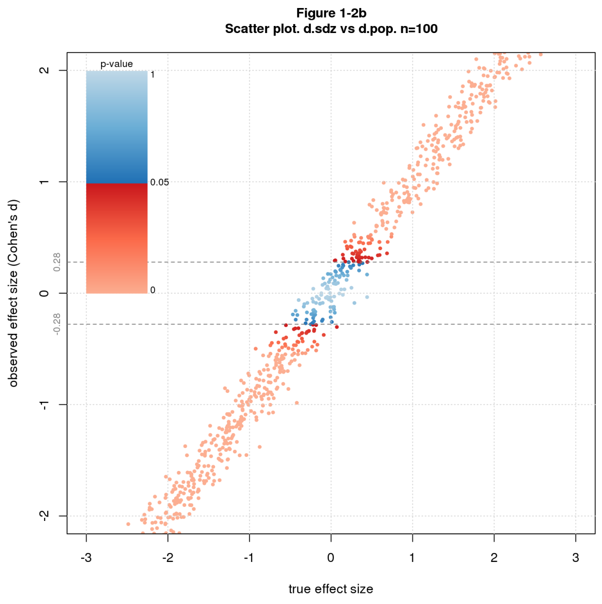
1. d vs. d scatter plot colored by p-value; by default d_{pop} vs. d_{sdz}
2. histogram typically of d_{sdz} colored by p-value
3. probability distribution vs. d_{sdz} colored by p-value; can compute probability density or cumulative probability internally, or you can pass the distribution into the function
4. multiple line plot; my adaptation of R's `matplot`

The first block of figures show d vs. d scatter plots for $n = 20$ and $n = 100$. You may notice that the p-value color legends are of different sizes. This is a limitation of the code that draws these legends.

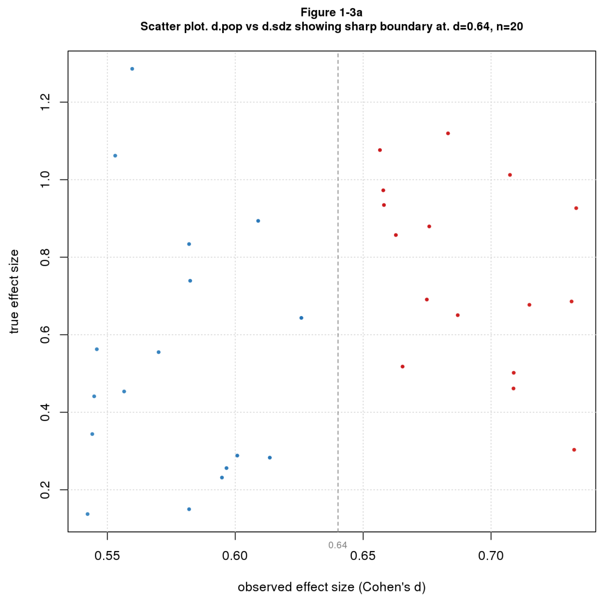


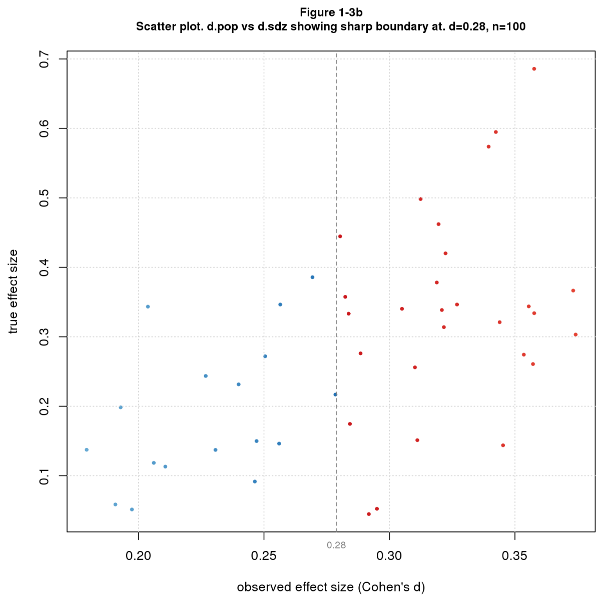




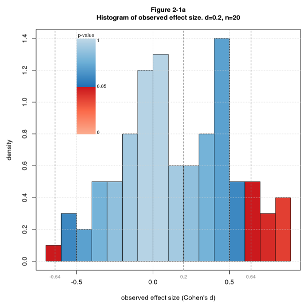


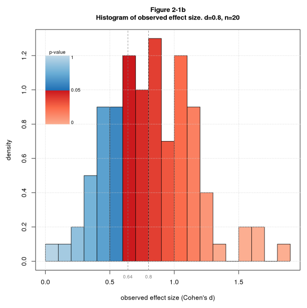
The next block are similar but zoom in to the critical region where p-values switch from nonsignificant to significant.

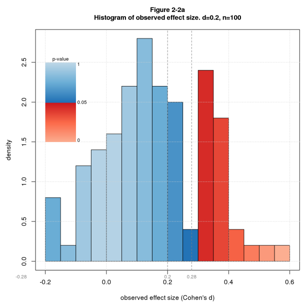


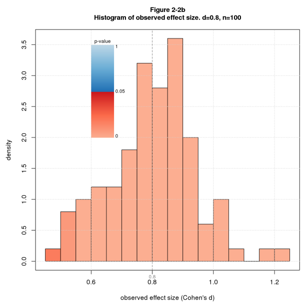


Next are four histograms for $d = 0.2$ and $d = 0.8$ by $n = 20$ and $n = 100$. You may notice that the p-value color legends are in different places and have different sizes. Again, these are limitations of the code.

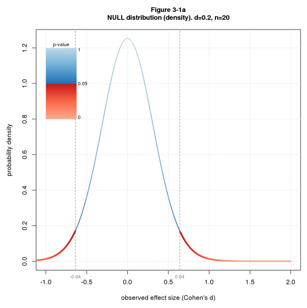


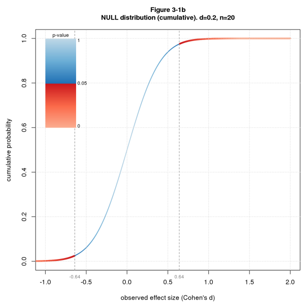


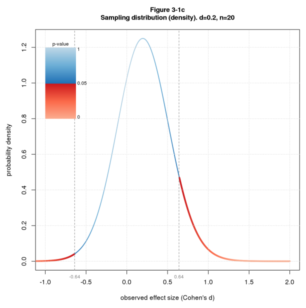


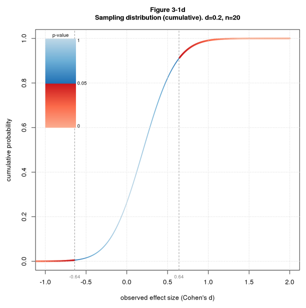


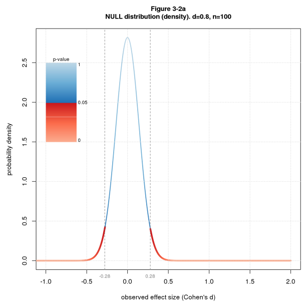
The next block are NULL and sampling distributions for $d = 0.2$, $n = 20$, and $d = 0.8$, $n = 100$.

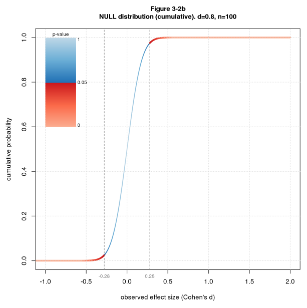


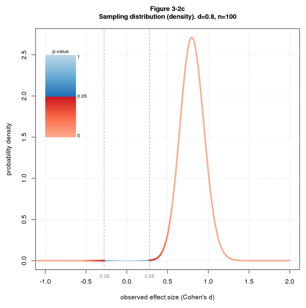


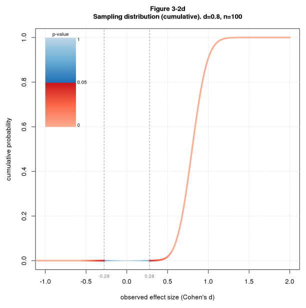




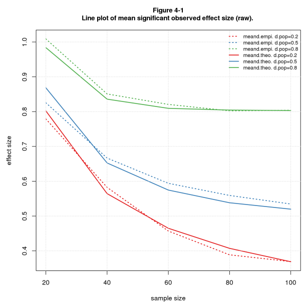


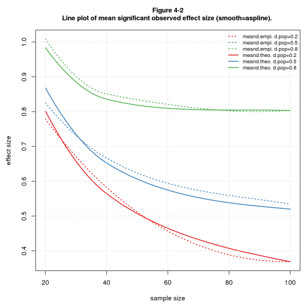


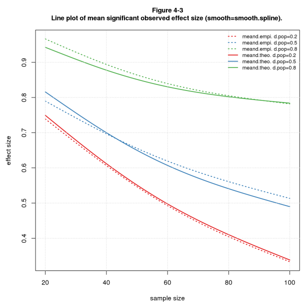


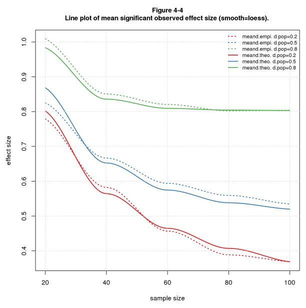


The final block are line plots of mean significant observed effect size ($meand_{empi}$, $meand_{theo}$) with various smoothing functions applied/









Finally, here is a table of supporting data.

Table 2: Table of supporting data

d.crit	meand20_0.2	meand20_0.5	meand20_0.8	over20_0.2	over20_0.5	over20_0.8	nover_0.2	nover_0
0.6401696	0.7927474	0.8590826	0.975697	3.963737	1.718165	1.219621	166.5173	46.5957

Comments Please!

Please post comments on Twitter or Facebook, or contact me by email natg@shore.net.

Please report bugs, other software problems, and feature requests using the GitHub Issue Tracker. I will be notified, and you'll be apprised of progress. As already noted, the software is still rough and software documentation nonexistent.

Copyright & License