Vignette for phack package

Sebastian Kranz, Ulm University

2022-09-17

This vignette heavily relies on the main analysis and code from G. Elliott, N. Kudrin and K. Wuthrich (2022). Responsible for any adaption and interpretation errors is the author Sebastian Kranz.

This vignette explains the `phack` package. Similar to Application 1 in Elliot et al. (2022), we apply different tests for p-hacking on the data set collected by Brodeur et al. (2016). As also Elliot et al (2022) do, this vignette particular stresses the fact that rounding problems in your p-values can invalidate the proposed tests and *must* be adressed.

Let us load the data set and have a look at it.

```
library(phack)
dat = phack::starwars
head(dat)
```

```
##   artid                  journal year issue article_page           p
## 1   424 American Economic Review 2010 100.1           35 9.921964e-02
## 2   424 American Economic Review 2010 100.1           35 1.520992e-07
## 3   424 American Economic Review 2010 100.1           35 3.550389e-02
## 4   424 American Economic Review 2010 100.1           35 1.917037e-01
## 5   424 American Economic Review 2010 100.1           35 4.550026e-02
## 6   424 American Economic Review 2010 100.1           35 7.451811e-01
##      p_deround table_panel column model field     field_2
## 1 9.970194e-02           4      3   yes Micro Development
## 2 2.638804e-06           5      1   yes Micro Development
## 3 3.494197e-02           4      2   yes Micro Development
## 4 1.927735e-01           4      3   yes Micro Development
## 5 4.823421e-02           4      3   yes Micro Development
## 6 7.511159e-01         A_2      5   yes Micro Development
```

Brodeur et al. (2016) looked at the regression tables from many economic articles and extracted the coefficients and standard errors of the main variables of interest. They then computed from those numbers the t-statistics and the p-values of the corresponding t-test with the null hypothesis that the true coefficient is zero. (Since degrees of freedom were often not provided in the regression tables Brodeur et al. computed the p-values assuming that the t-statistics are normally distributed).

The original data set contains over 130 columns, that e.g. provide background information about the authors or detailed information of the significant digits of the extracted coefficients and so on. To save space, I just stored a small number of columns that suffice for the analysis in this vignette. I also applied some correction of coding errors that Elliot et al. (2022) perform in their code supplement. The complete data set is available in the code and data supplement of Brodeur et al. (2016).

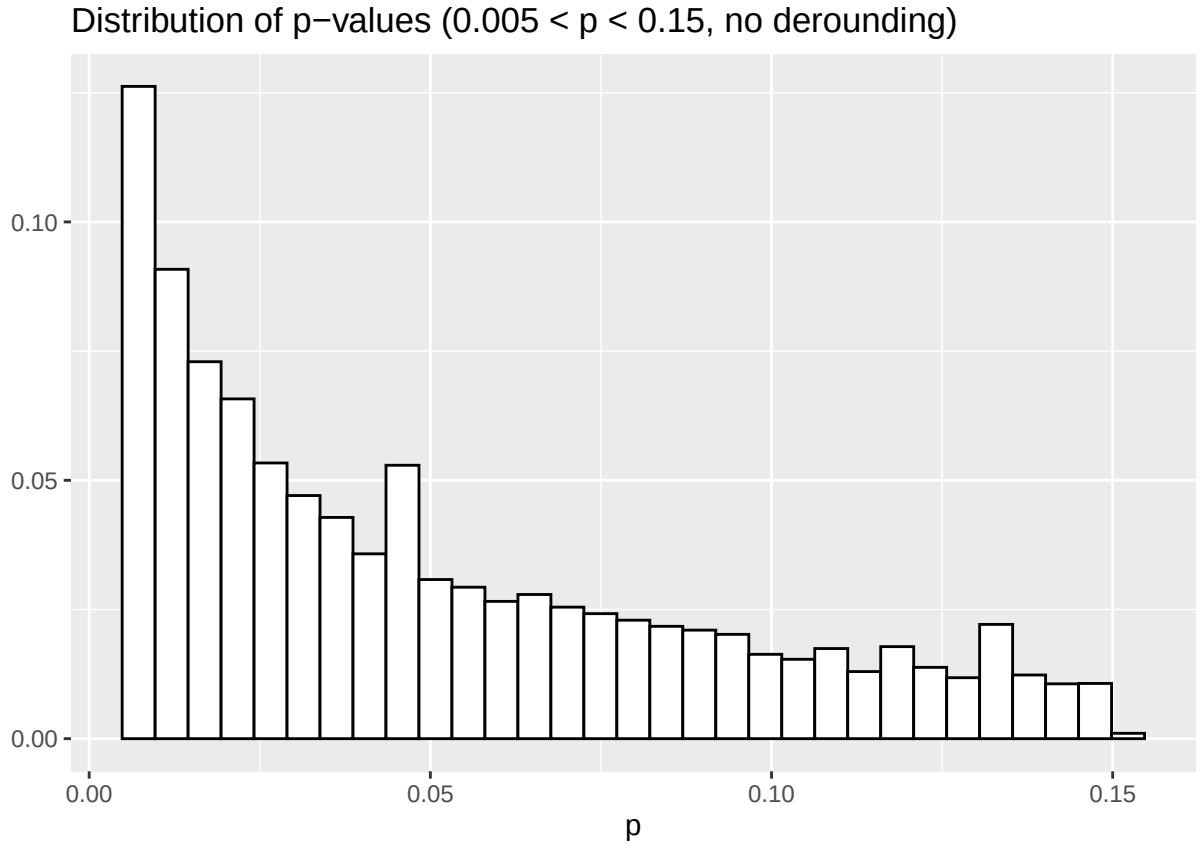
The p-curve and the rounding problem

Let us first plot the distribution of p-values:

```
library(dplyr)
# Considered range of p-values for plots and tests
p_min = 0.005
p_max = 0.15
d = filter(dat, p >= p_min, p <= p_max)

# Number of histogram bins
J = 30

ggplot(d, aes(x = p, y = stat(count / NROW(d))))+
  geom_histogram(binwidth = (p_max-p_min)/J, boundary = 0,color="black", fill="white") +
  ggtitle("Distribution of p-values (0.005 < p < 0.15, no derounding)") +
  ylab("")
```



Note that this observed distribution of p-values is actually a sample from a mixture distribution. Each considered regression coefficient / t-value has its own true effect size. Elliot et al. call the density of this mixture distribution the *p-curve*.

We know that in different contexts mixture distributions can have quite complicated structures. E.g. the density of a mixture distribution often has multiple local maxima. Yet, Elliot et al. (2022) derive a great result in their Theorem 1. Let me state the key implication here somewhat informally:

> Given some (rather weak) regulatory conditions the density function of p-values (p-curve) across tests is non-increasing if
>
> 1. there is no p-hacking and publication bias and
> 2. p-values are observed without rounding errors

In the plotted histogram above, we see that just below the 5% threshold, the empirical density of p-values increases substantially. There could be three different reasons for why the empirical density of p-values is not everywhere non-increasing:

A: There is systematic p-hacking or publication bias.

B: The increase in the empirical density is just due to random sampling variation.

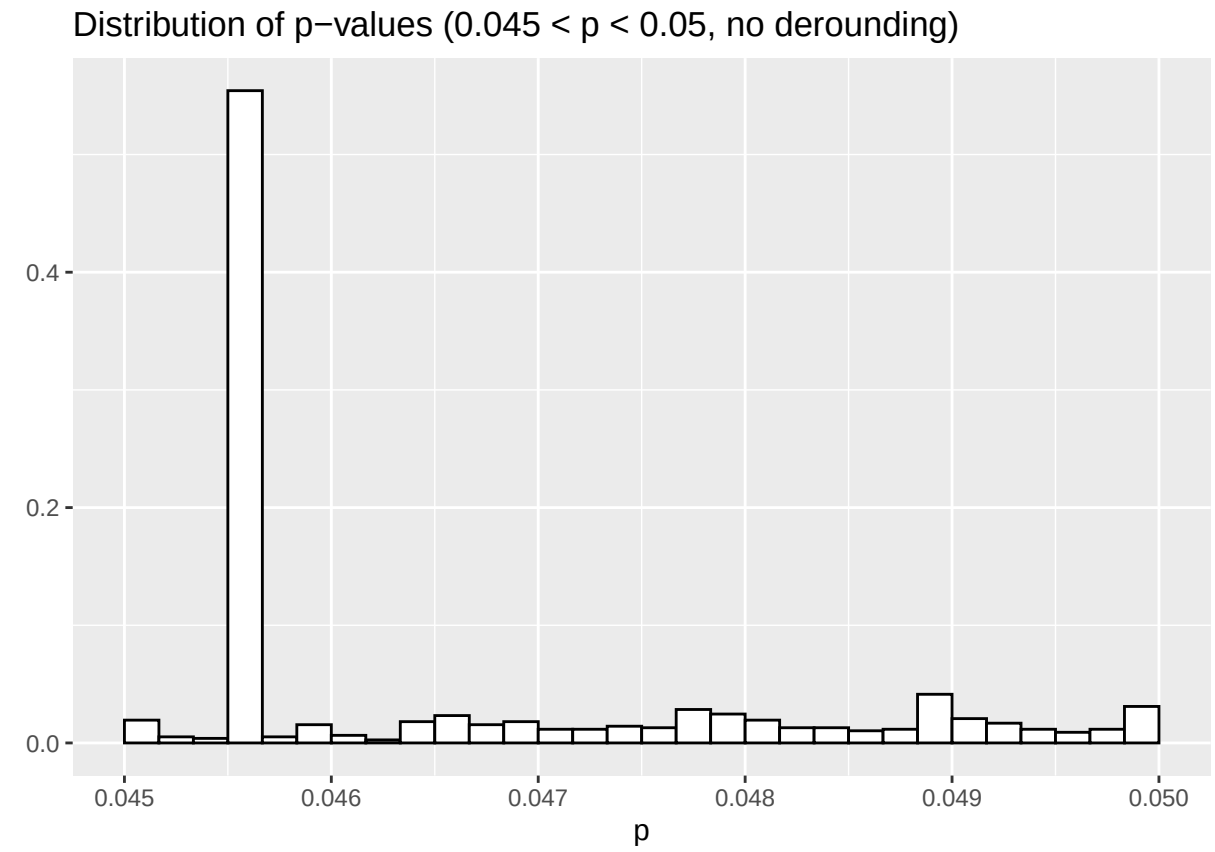
C: The increase in the density of p-values is due to rounding errors in the collected p-values.

To check whether there is systematic p-hacking (or publication bias) or whether the irregularities in the empirical density of p-values are just due to sampling variation, Elliot et al. (2022) propose and implement several different statistical tests. We will look at them further below.

But first, we have to make sure that the increase in p-values before 0.05 is not due to rounding errors in the collected p-values.

Let us zoom into the histogram and look in more detail at the distribution of p-values before the 5% threshold.

```
p_min = 0.045; p_max = 0.05
d = dat %>% filter(p >= p_min, p <= p_max)
ggplot(d, aes(x = p, y = stat(count / NROW(d))))+
  geom_histogram(binwidth = (p_max-p_min)/J, boundary = 0,color="black", fill="white") +
  ggtitle("Distribution of p-values (0.045 < p < 0.05, no derounding)") +
  ylab("")
```



We see that there is massive bunching of p-values. Indeed 427 p-values have a value of 0.045500264:

```
sort(table(d$p), decreasing=TRUE)[1]
```

```
## 0.045500264
```

```
##         427
```

What is the reason? It turns out that this is the p-value corresponding to a t-statistic of exactly `t=2` (using the normal approximation):

```
t = 2
2*(1-pnorm(t))
```

```
## [1] 0.04550026
```

The likely main reason for this bunching is coarse rounding of the extracted coefficients and standard errors in the published regression tables.

For example, assume the reported coefficient is `beta.hat = 0.04` and the reported standard error is `se = 0.02`. Then Brodeur et al (2016), would compute a t-statistic of `t = 0.04 / 0.02 = 2` for this test.

However, before rounding `beta.hat` could have been any number in the interval `[0.035,0.045)` and the standard error could have been any number in the interval `[0.015,0.025)`. This implies that the actual p-values could have taken any value between 0.00270 and 0.161 if regression results were observed without rounding:

```
# Range of possible p-values in example
# before rounding
t.min = 3.5 / 2.5;
t.max = 4.5 / 1.5;
c(2*(1-pnorm(t.max)),2*(1-pnorm(t.min)))
```

```
## [1] 0.002699796 0.161513318
```

Elliot et al. (2022) thus emphasize that one should not ignore rounding problems when applying their tests. In a related manner, Kranz and Puetz (2022) illustrate how ignored rounding problems can cause a lot of falsely significant findings also in other tests for p-hacking that were used in a well published article.

There are different ways to correct for rounding. One simple procedure, discussed by Brodeur et al. (2016) is to generate for each observation one single de-rounded p-value. In our example, we would compute a derounded p-value as follows:

```
set.seed(123)
# Reported coefficient and se
beta.hat = 0.04; se = 0.02;

# Deround by drawing uniformly from all feasible
# values before rounding
beta.hat.deround = runif(1, 0.035, 0.045)
se.deround = runif(1, 0.015, 0.025)

# Compute t-statistic and p-value for derounded
# coefficient and stand errors
t.deround = beta.hat.deround / se.deround
p.deround = 2*(1-pnorm(t.deround))
p.deround
```

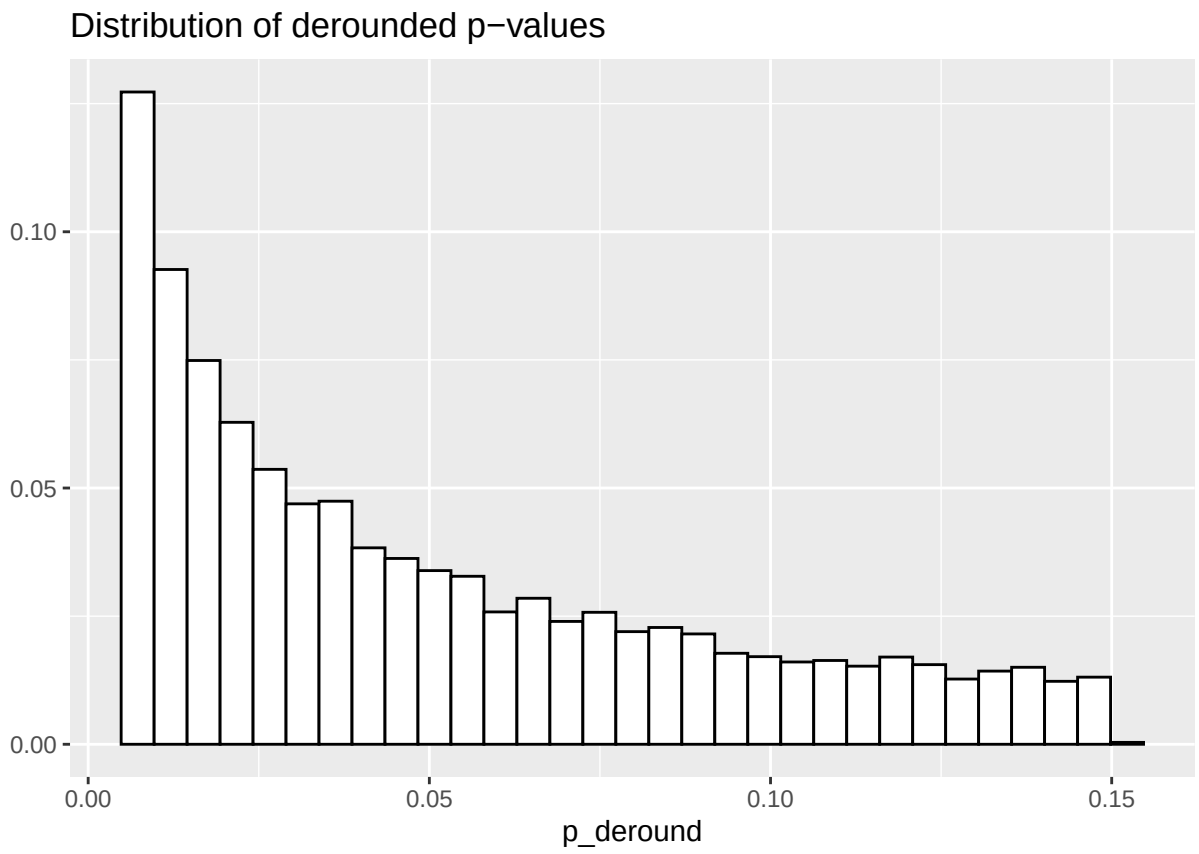
```
## [1] 0.09788616
```

The column `p_deround` in our data set contains a single draw of derounded p-values using the procedure above. (An earlier working paper version of Elliot et al. shows in more detail how de-rounded p-values will satisfy their monotonicity result absent p-hacking. Kranz and Puetz (2022) compare for different tests different de-rounding approaches using Monte-Carlo studies.)

Let us plot the empirical p-curve for those derounded p-values:

```r
# Considered range of p-values for plots and tests
p_min = 0.005; p_max = 0.15
d = filter(dat, p_deround >= p_min, p_deround <= p_max)

ggplot(d, aes(x = p_deround, y = stat(count / NROW(d))))+
  geom_histogram(binwidth = (p_max-p_min)/J, boundary = 0,color="black", fill="white") +
  ggtitle("Distribution of derounded p-values") +
  ylab("")
```



The derounded p-values don't show anymore the large density increase before 0.05. Yet, still the empirical density is not completely monotonically decreasing. So let's move on to the formal tests for p-hacking.

Tests for p-hacking

A first look at the binomial test

The following code runs a binomial test for p-hacking using our derounded p-values and returns the p-value of the test:

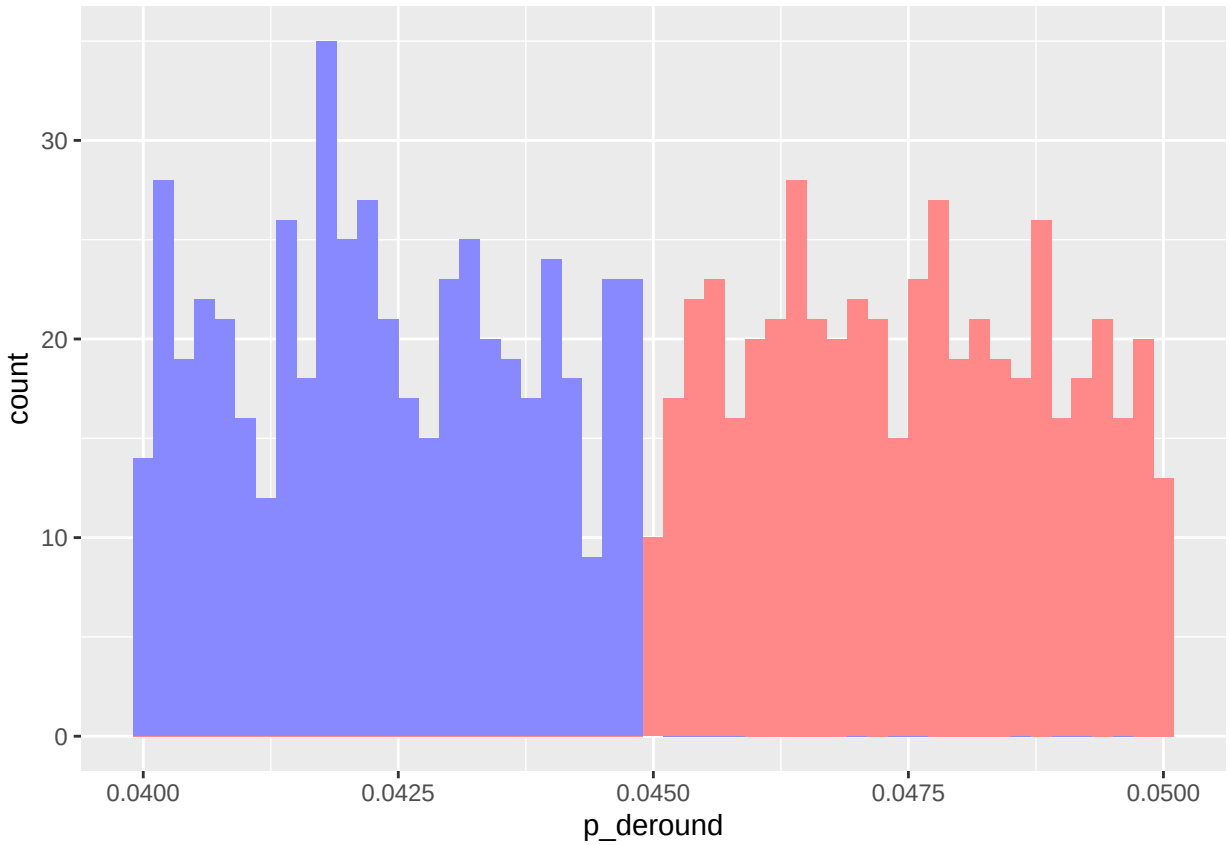
```r
phack_test_binomial(dat$p_deround, p_min = 0.04, p_max = 0.05)
```

```
## [1] 0.6790704
```

This test delivers no significant evidence. The test above considers all derounded p-values in the interval from `p_min = 0.04` to `p_max = 0.05` shown in the following histogram:

```r
ggplot(dat, aes(x = p_deround))+
  geom_histogram(data=filter(dat, p_deround>=0.04, p_deround<=0.045),
```

```
                binwidth = 0.0002, fill="#8888ff") +
  geom_histogram(data=filter(dat, p_deround>0.045, p_deround<=0.05),
                binwidth = 0.0002, fill="#ff8888")
```



The test splits this interval in the middle and then tests with a binomial test, whether the share of p-values in the right half (red) is significantly larger than in the left half (blue). That would violate the property of a non-increasing p-curve and thus suggest p-hacking or publication bias. But here that is not the case.

You would typically run such a binomial test using an interval left of a typical significance threshold (1%, 5% or 10%). Let us repeat the binomial test before the 10% threshold:

```
phack_test_binomial(dat$p_deround, p_min = 0.09, p_max = 0.10)
```

```
## [1] 0.8209841
```

Again no evidence for p-hacking.

What happens if we run the binomial test on the raw p-values that were subject to rounding errors?

```
phack_test_binomial(dat$p, 0.09,0.10)
```

```
## Warning in warn_p_rounding(P, min_bunch):
## Your p-values seem to face an unsolved rounding problem.
## For example, you have 111 p-values that all have the value 0.095580712.
## A likely reason for such bunching are rounding errors.
## (The rounding errors might be present in the
## corresponding t-values or reported coefficients and standard errors
##  from which p-values were computed.).
##  This test for p-hacking is not valid if p-values face rounding errors.
```

```
## Please first apply an appropriate de-rounding procedure for your p-values.
```

```
## [1] 0.00374467
```

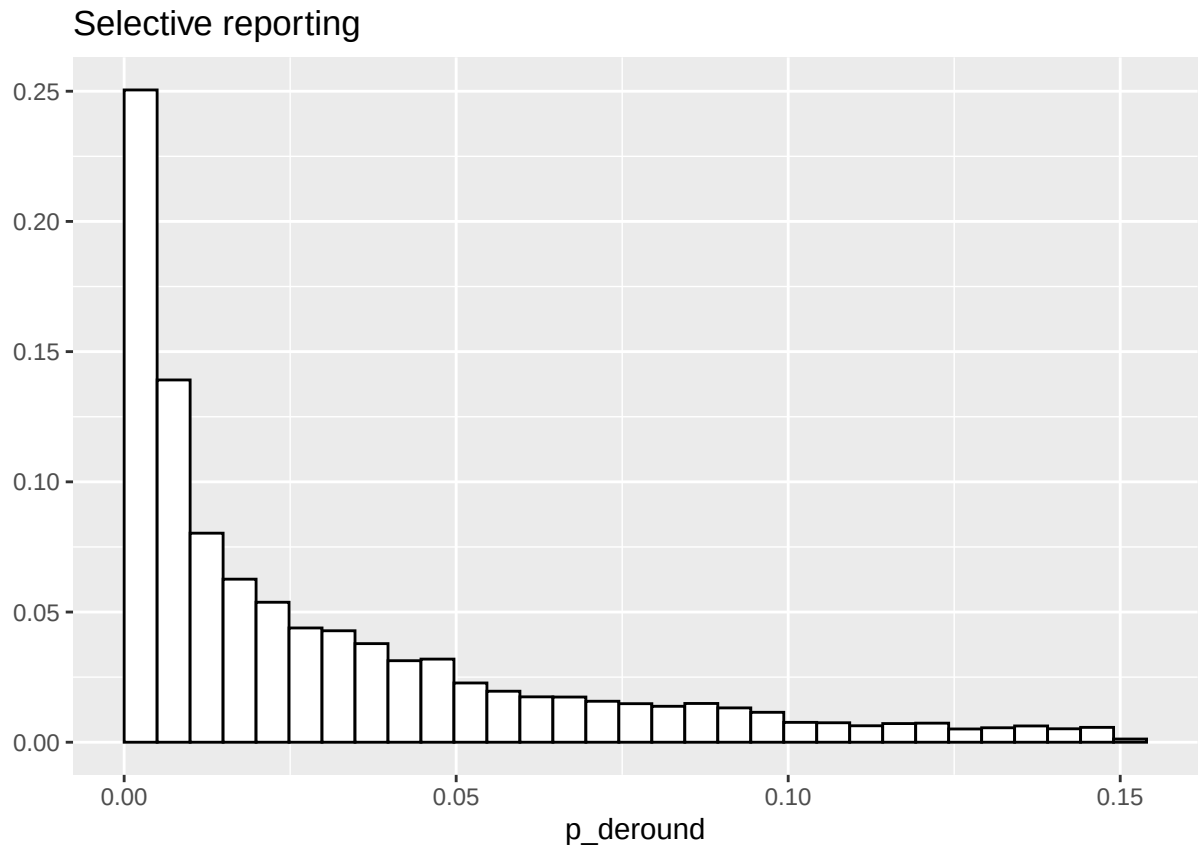
We now get a very small p-value, but also a warning that several p-values are bunched on the same number suggesting a rounding problem. Recall that unresolved rounding problems can lead to false findings if we test for p-hacking.

Simulate data sets with p-hacking / publication bias

There are several other tests implemented in the package and studied by Elliot et al. (2022). Before running them, let us generate some artifically modified data sets to reflect some p-hacking or publication bias.

Let us first assume that authors are less likely to report some tests if the corresponding p-value is above a certain significance thresholds. We assume that a test with p >= 10% is thrown out with 60% probability, a test with p >= 5% with 40% probability and a test with p >= 1% with 20% probability. If each article would consist only of a single test this *selective reporting* could be also interpreted as publication bias. In our simulation, we assume that `p_deround` corresponds to the actual p-values.

```
set.seed(123)
dat_sel = dat %>%
  mutate(
    rand_num = runif(n(),0,1),
    remove_prob = case_when(
      p_deround > 0.1 ~ 0.6,
      p_deround > 0.05 ~ 0.4,
      p_deround > 0.01 ~ 0.2,
      TRUE ~ 0
    )
  ) %>%
  filter(rand_num > remove_prob)
p_min = 0.001; p_max = 0.15
d = filter(dat_sel, p_deround>= p_min, p_deround <= p_max)
ggplot(d, aes(x = p_deround, y = stat(count / NROW(d))))+
  geom_histogram(binwidth = (p_max-p_min)/J, boundary = 0,
                 color="black", fill="white") +
  ggtitle("Selective reporting") +
  ylab("")
```



You might already correctly predict that our binomial test won't be able to detect this selective reporting / publication bias, as it does not induce an increase in the p-curve. The p-curve rather becomes more decreasing when crossing a significance threshold. Let us verify this intuition:

```
phack_test_binomial(dat_sel$p_deround, p_min = 0.04, p_max = 0.05)
```

```
## [1] 0.4310329
```

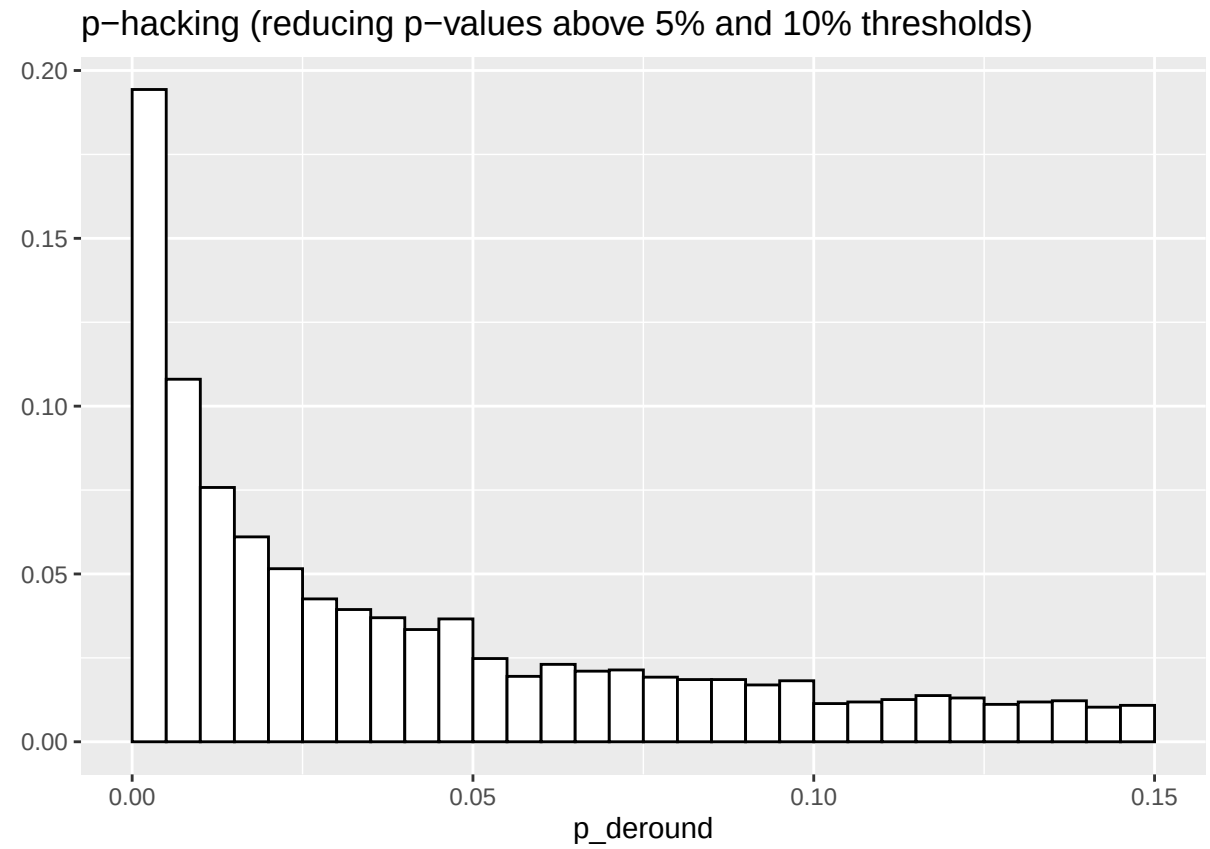
Yep, that binomial test found no significance evidence for p-hacking or publication bias in our data set with selective reporting. We will later look whether other tests are more successful for this data set.

Let us now instead assume that if a p-value is originally above a significance threshold of 10% or 5%, some authors tweak the specification (e.g. by modifying the set of control variables) in order to reduce the p-value. Here is a simple implemenation to simulate such behavior:

```
set.seed(123)
dat_hack = dat %>%
  mutate(
    rand_num = runif(n(),0,1),
    # Only 25% of p-values slightly above
    # threshold will be hacked
    do_hack = runif(n(),0,1) <= case_when(
      p_deround > 0.1 & p_deround < 0.11 ~ 0.25,
      p_deround > 0.05 & p_deround < 0.06 ~ 0.25,
      TRUE ~ 0
    ),
    # Amount that can be reduced is somewhat random
    reduce_by = runif(n(),0.001,0.01)*do_hack,
```

```
    p_deround = p_deround - reduce_by
  )

p_min = 0.001; p_max = 0.15
d = filter(dat_hack, p_deround>= p_min, p_deround <= p_max)
ggplot(d, aes(x = p_deround, y = stat(count / NROW(d))))+
  geom_histogram(binwidth = 0.005, boundary = 0,
                 color="black", fill="white") +
  ggtitle("p-hacking (reducing p-values above 5% and 10% thresholds)") +
  ylab("")
```



This p-hacking now causes the p-curve to be increasing before the 5% and 10% significance thresholds. Let's see whether the binomial tests can detect it:

```
phack_test_binomial(dat_hack$p_deround, p_min = 0.04, p_max = 0.05)
```

```
## [1] 0.06461619
```

```
phack_test_binomial(dat_hack$p_deround, p_min = 0.09, p_max = 0.10)
```

```
## [1] 0.2049562
```

Well, we find weak evidence for p-hacking before the 5% threshold. But maybe other tests are more powerful.

Fisher test

Instead of a binomal test, we could also run a Fisher test, which extends the idea of the binomial test. I won't explain the idea here but refer the reader to Simonsohn et al. (2014) for more background.

```
phack_test_fisher(dat$p_deround, p_min = 0.04, p_max = 0.05)
```

```
## [1] 0.70284
```

```
phack_test_fisher(dat_sel$p_deround, p_min = 0.04, p_max = 0.05)
```

```
## [1] 0.5169509
```

```
phack_test_fisher(dat_hack$p_deround, p_min = 0.04, p_max = 0.05)
```

```
## [1] 0.0538295
```

For the p-hacked data set, the Fisher test yields a lower p-value than the binomial test. But the test still does not detect the selective reporting.

Let me state the obvious: Of course, one should determine in advance which tests to report, not just pick ex-post the one with the lowest p-value.

Histogram Test based on Cox-Shi conditional chi-squared test

In Section 4.1 Elliot et al. (2022) also propose a Cox-Shi conditional chi-squared test that looks at the complete histogram in a larger range of p-values. The test allows for cluster-robust variance estimators that will be clustered on article level. Let us run it on the derounded p-values in our original data set:

```
dat$artid = paste0(dat$journal,"_",dat$issue,"_",dat$article_page)
phack_test_cox_shi(dat$p_deround,dat$artid, p_min=0, p_max=0.15, J=30)
```

```
## Warning in warn_p_rounding(P, min_bunch):
## Your p-values seem to face an unsolved rounding problem.
## For example, you have 97 p-values that all have the value 2.220446e-16.
## A likely reason for such bunching are rounding errors.
## (The rounding errors might be present in the
## corresponding t-values or reported coefficients and standard errors
##  from which p-values were computed.).
##  This test for p-hacking is not valid if p-values face rounding errors.
## Please first apply an appropriate de-rounding procedure for your p-values.
```

```
## [1] 0.4916875
```

Nothing significant. Yet, we also get a warning suggesting that our derounding was not completely succesful for p-values very close to zero. So we will ignore those, very small p-values when running the test on all three data sets:

```
phack_test_cox_shi(dat$p_deround,dat$artid, p_min=0.0001, p_max=0.15, J=30)
```

```
## [1] 0.6899854
```

```
phack_test_cox_shi(dat_sel$p_deround,dat_sel$artid, p_min=0.00001, p_max=0.15, J=30)
```

```
## [1] 0.9951807
```

```
phack_test_cox_shi(dat_hack$p_deround,dat_hack$artid, p_min=0.00001, p_max=0.15, J=30)
```

```
## [1] 0.1956939
```

OK, this test did not detect any problems in any data set.

More general monotonicity and bounds test for p-curve from t-tests

If all p-values correspond to t-tests, Elliot et al. (2022) derive additional theoretical implications on bounds of the p-curves and a feature called complete monotonicity. They use these results to implement more powerful

test that are also based on the Cox-Shi conditional chi-squared test. Details are given in Sections 3, 4.3 and Appendix A in Elliot et al. (2022). Let us just run these test using the same specification that Elliot et al. used in their application:

```
phack_test_cox_shi(dat$p_deround,dat$artid, p_min=0.0001, p_max=0.15,
                   J=30, K=2, use_bound=TRUE)
```

```
## [1] 0.2747358
```

```
phack_test_cox_shi(dat_sel$p_deround,dat_sel$artid,  p_min=0.0001, p_max=0.15,
                   J=30, K=2, use_bound=TRUE)
```

```
## [1] 0.0008844484
```

```
phack_test_cox_shi(dat_hack$p_deround,dat_hack$artid,  p_min=0.0001, p_max=0.15,
                   J=30, K=2, use_bound=TRUE)
```

```
## [1] 3.25446e-09
```

Great! We find no evidence for p-hacking for the derounded p-values in our original data set, but the test detects both the selected reporting and p-hacking in our modified data set. While I don't really understand what the tests do (I am both ignorant and somewhat lazy to dig too deep into complicated theoretical econometrics), they seem to work pretty well!

My impression from reading the article is that Elliot et al. indeed do recommend this test if your p-curve comes purely from t-tests.

Discontinuity test

While we could actually stop here given the great performance of the previous test, let us look at two more tests studied by Elliot et al. They show that absent p-hacking, publication bias and rounding errors, the p-curve should be continous. The following test (described in their Section 4.2) allows to test for a discontinouty at a particular point. One typically would test for a discontinouty at a signifance threshold like 5% or 10%.

```
phack_test_discontinuity(dat$p_deround,0.05,min_bunch = 100)
```

```
## [1] 0.794516
```

```
phack_test_discontinuity(dat_sel$p_deround,0.05,min_bunch = 100)
```

```
## [1] 0.07303213
```

```
phack_test_discontinuity(dat_hack$p_deround,0.05,min_bunch = 100)
```

```
## [1] 0.5187237
```

The discontinouty test seems to have some power to detect the selective reporting, but does not see any problem in our `dat_hack` data set.

LCM Test

In Section 4.1 Elliot et al. also propose a test on the concavity of the cdf of p-values (the integral of the p-curve). It is a test based on the least concave majorant (LCM). I have no clue what that is. For further information, best check the literature cited in Elliot et al. (2022). Let us just run the test.

```
# The first time the test is run
# it generates some numbers using a Monte-Carlo simulation
set.seed(123)
phack_test_lcm(dat$p_deround, p_min=0.0001, p_max=0.15)
```

```
## [1] 1
```

```
phack_test_lcm(dat_sel$p_deround, p_min=0.0001, p_max=0.15)
```

```
## [1] 1
```

```
phack_test_lcm(dat_hack$p_deround, p_min=0.0001, p_max=0.15)
```

```
## [1] 1
```

Well here, the LCM tests take long to run and detect nothing…

References:

Brodeur, A., Le, M., Sangnier, M., and Zylberberg, Y. (2016a). Replication data for: Star wars: The empirics strike back. Nashville, TN: American Economic Association [publisher], 2016. Ann Arbor, MI: Inter-university Consortium for Political and Social Research [distributor], 2019-10-12. https://doi.org/10.3886/E113633V1

Brodeur, A., Le, M., Sangnier, M., and Zylberberg, Y. (2016b). Star wars: The empirics strike back. American Economic Journal: Applied Economics, 8(1):1–32.

Elliott, G., Kudrin, N., & Wüthrich, K. (2022). Detecting p-Hacking. Econometrica, 90(2), 887-906.

Kranz, S., & Pütz, P. (2022). Methods matter: P-hacking and publication bias in causal analysis in economics: Comment. American Economic Review, 112(9), 3124-36.

Simonsohn, U., Nelson, L. D., & Simmons, J. P. (2014). P-curve: a key to the file-drawer. Journal of experimental psychology: General, 143(2), 534.