Almost all researchers have experienced the tingling feeling of suspense that arises right before they take a look at long-awaited data: Will they support their favored hypothesis? Will they yield interesting or even groundbreaking results? In a perfect world (especially one without publication bias), the cause of this suspense should be nothing else but scientific curiosity. However, the world, and specifically the incentive system in science, is not perfect. A lot of pressure rests on researchers to produce statistically significant results. For many researchers, statistical significance is the cornerstone of their academic career, so non-significant results in an important study can not only question their scientific convictions but also crash their hopes of professional promotion..

As many people may have heard, *p*-hacking works because it exploits a process called *alpha error accumulation* which is covered in most introductory statistics classes (but also easily forgotten again). Basically, alpha error accumulation means that as one conducts more and more hypothesis tests, the probability increases that one makes a wrong test decision at least once. Specifically, this wrong test decision is a false positive decision or *alpha error*, which means that you proclaim the existence of an effect although, in fact, there is none. Speaking in statistical terms, an alpha error occurs when the test yields a significant result although the null hypothesis (“There is no effect”) is true in the population. This means that *p*-hacking leads to the publication of an increased rate of false positive results, that is, studies that claim to have found an effect although, in fact, their result is just due to the randomness of the data. Such studies will never replicate.

At this point, the blog post could be over. *P*-Hacking exploits alpha error accumulation and fosters the publication of false positive results which is bad for science. However, we want to take a closer look at *how* bad it really is. In fact, some *p*-hacking techniques are worse than others (or, if you like the unscrupulous science villain perspective: some *p*-hacking techniques work better than others). As a showcase, we want to introduce two researchers: The *HARKer* takes existing data and conducts multiple independent hypothesis tests (based on multiple uncorrelated variables in the data set) with the goal to publish the ones that become significant. For example, the HARKer tests for each possible correlation in a large data set whether it differs significantly from zero. On the other hand, the *Accumulator* uses optional stopping. This means that he collects data for a single research question test in a sequential manner until either statistical significance or a maximum sample size is reached. For simplicity, we assume that they use neither other *p*-hacking techniques nor other questionable research practices.

**The HARKer’s *p*-hacking strategy**

Let us start with the HARKer: Since the conducted hypothesis tests in our defined scenario are essentially independent, the situation can be seen as a problem of multiple testing. This means, it is comparatively easy to determine the exact probability that the HARKer will end up with at least one false-positive result given a certain number of hypothesis tests. Assuming no effects in the population (for example, no correlation between the variables), one can picture the situation as a decision tree: At each branch level stands a hypothesis test which can either result in a non-significant result with 95% probability or in a (spurious) significant result with 5% probability, which is the level.

No matter how many hypothesis tests the HARKer conducts, there will only be one condition in the all-null scenario where no error occurs, that is, where all hypothesis tests yield non-significant results. The probability that this occurs can be calculated by , with being the number of conducted hypothesis tests. The probability that at least one of the hypothesis tests is significant is the probability of the complementary event, that is . For example, when the HARKer computes hypothesis with an alpha level of , the overall probability to obtain at least one false positive result is . Of course, the formula can be adjusted for other suggested alpha levels, such as or . We show this general formula in the R-code chunk below.

[cc lang=”rsplus” escaped=”true”]

# The problem of HARKing

harker <- function(x, alpha){1-(1-alpha)^x}  
[/cc]

**The Accumulator’s *p*-hacking strategy**

The Accumulator has a different tactic: Instead of conducting multiple hypothesis tests on different variables of one data set, he repeatedly conducts the same hypothesis test on the same variables in a growing sample. Starting with a minimum sample size, the Accumulator looks at the results of the analysis – if these are significant, data collection is stopped, if not, more data is collected until (finally) the results become significant, or a maximum sample size is reached. This strategy is also called *Optional Stopping*. Of course, the more often a researcher peeks at the data, the higher is the probability to obtain a false positive result at least once. However, this overall probability is not the same as the one obtained through HARKing. The reason is that the hypothesis tests are not independent in this case. Why is that? The same hypothesis test is repeatedly conducted on only slightly different data. In fact, the data that were used in in the first hypothesis test are used in every single of the subsequent hypothesis tests so that there is a spillover effect of the first test to every other hypothesis test in the set. Imagine, your initial sample contains an outlier: This outlier will affect the test results in any other test. With multiple testing, in contrast, the outlier will affect only the test in question but none of the other tests in the set.

So does this dependency make optional stopping more or less effective than HARKing? Of course, people have been wondering about this for quite a while. A paper by [Armitage et al. (1969)](https://www.jstor.org/stable/2343787) demonstrates error accumulation in optional stopping for three different tests. We can replicate their results for the *z*-test with a small simulation (a more flexible simulation can be found at the end of the blog post): We start by drawing a large number of samples (iter) with the maximum sample size (n.max) from the null hypothesis. Then we conduct a sequential testing procedure on each of the samples, starting with a minimum sample size (n.min) and going up in several steps (step) up to the maximum sample size. The probability to obtain a significant result at least once up to a certain step can be estimated through the percentage of iterations that end up with at least one significant result at that point.

[cc lang=”rsplus” escaped=”true”]  
# The Problem of optional stopping

accumulator <- function(n.min, n.max, step, alpha=0.05, iter=10000){

# Determine places of peeks  
peeks <- seq(n.min, n.max, by=step)

# Initialize result matrix (non-sequential)  
res <- matrix(NA, ncol=length(peeks), nrow=iter)  
colnames(res) <- peeks

# Conduct sequential testing (always until n.max, with peeks at pre-determined places)  
for(i in 1:iter){  
sample <- rnorm(n.max, 0, 1)  
res[i,] <- sapply(peeks, FUN=function(x){sum(sample[1:x])/sqrt(x)})  
}

# Create matrix: Which tests are significant?  
signif <- abs(res) >= qnorm(1-alpha)

# Create matrix: Has there been at least one significant test in the trial?  
seq.signif <- matrix(NA, ncol=length(peeks), nrow=iter)

for(i in 1:iter){  
for(j in 1:ncol(signif)){  
seq.signif[i,j] <- TRUE %in% signif[i, 1:j]  
}  
}

# Determine the sequential alpha error probability for the sequential tests  
seq.alpha <- apply(seq.signif, MARGIN = 2, function(x){sum(x)/iter})

# Return a list of individual test p-values, sequential significance and sequential alpha error probability  
return(list(seq.alpha = seq.alpha))

}  
[/cc]

For example, the researcher conducts a two-sided one-sample *z*-test with an overall level of .05 in a sequential way. He starts with 10 observations, then always adds another 10 if the result is not significant, up to 100 observations at maximum. This means, he has 10 chances to peek at the data and end the data collection if the hypothesis test is significant. Using our simulation function, we can determine the probability to have obtained at least one false positive result at any of these steps:

[cc lang=”rsplus” escaped=”true”]  
set.seed(1234567)  
res.optstopp <- accumulator(n.min=10, n.max=100, step=10, alpha=0.025, iter=10000)  
print(res.optstopp[[1]])  
[/cc]  
[cc]  
[1] 0.0492 0.0824 0.1075 0.1278 0.1431 0.1574 0.1701 0.1788 0.1873 0.1949  
[/cc]

We can see that with one single evaluation, the false positive rate is at the nominal 5%. However, when more in-between tests are calculated, the false positive rate rises to roughly 20% with ten peeks. This means that even if there is no effect at all in the population, the researcher would have stopped data collection with a signficant result in 20% of the cases.

**A comparison of the HARKer’s and the Accumulator’s strategy**

Let’s compare the false positive rates of HARKing and optional stopping: Since the researcher in our example above conducts one to ten dependent hypothesis tests, we can compare this to a situation where a HARKer conducts one to ten independent hypothesis tests. The figure below shows the results of both *p*-hacking strategies:

[cc lang=”rsplus” escaped=”true”]  
# HARKing False Positive Rates

HARKs <- harker(1:10, alpha=0.05)  
[/cc]

We can see that HARKing produces higher false positive rates than optional stopping with the same number of tests. This can be explained through the dependency on the first sample in the case of optional stopping: Given that the null hypothesis is true, this sample is not very likely to show extreme effects in any direction (however, there is a small probability that it does). Every extension of this sample has to “overcome” this property not only by being extreme in itself but also by being extreme enough to shift the test on the overall sample from non-significance to significance. In contrast, every sample in the multiple testing case only needs to be extreme in itself. Note, however, that false positive rates in optional stopping are not only dependent on the number of interim peeks, but also on the size of the initial sample and on the step size (how many observations are added between two peeks?). The difference between multiple testing and optional stopping which you see in the figure above is therefore only valid for this specific case. Going back to the two researchers from our example, we can say that the HARKer has a better chance to come up with significant results than the Accumulator, if both do the same number of hypothesis tests.

**Practice HARKing and Optional Stopping yourself**

You can use the interactive *p*-hacker app to experience the efficiency of both *p*-hacking strategies yourself: You can increase the number of dependent variables and see whether one of them gets significant (HARing), or you can got to the “Now: p-hack!” tab and increase your sample until you obtain significance. Note that the DVs in the p-hacker app are not completely independent as in our example above, but rather correlate with *r* = .2, assuming that the DVs to some extent measure at least related constructs.

**Conclusion**

To conclude, we have shown how two *p*-hacking techniques work and why their application is bad for science. We found out that *p*-hacking techniques based on multiple testing typically end up with higher rates of false positive results than *p*-hacking techniques based on optional stopping, if we assume the same number of hypothesis tests. We want to stress that this does *not* mean that naive optional stopping is okay (or even okay-ish) in frequentist statistics, even if it does have a certain appeal. For those who want to do guilt-free optional stopping, there are ways to control for the error accumulation in the frequentist framework (see for example [Wald, 1945](https://projecteuclid.org/download/pdf_1/euclid.aoms/1177731118), [Chow & Chang, 2008](https://ojrd.biomedcentral.com/articles/10.1186/1750-1172-3-11), [Lakens, 2014](https://psyarxiv.com/9yegd/)) and sequential Bayesian hypothesis tests (see for example [our paper](https://www.researchgate.net/profile/Felix_Schoenbrodt/publication/286971067_Sequential_Hypothesis_Testing_With_Bayes_Factors_Efficiently_Testing_Mean_Differences/links/5673d52408aee7a427458f83.pdf) on sequential hypothesis testing with Bayes factors or [Rouder, 2014](https://pdfs.semanticscholar.org/0f7b/f686ed6a052fca9465698acec9e26d7be7e2.pdf)).

**Alternative Simulation Code (also including one-sided tests and t-tests)**

[cc lang=”rsplus” escaped=”true”]

sim.optstopping <- function(n.min, n.max, step, alpha=0.05, test=”z.test”, alternative=”two.sided”, iter=10000){

match.arg(test, choices=c(“t.test”, “z.test”))  
match.arg(alternative, choices=c(“two.sided”, “directional”))

# Determine places of peek  
peeks <- seq(n.min, n.max, by=step)

# Initialize result matrix (non-sequential)  
res <- matrix(NA, ncol=length(peeks), nrow=iter)  
colnames(res) <- peeks

# Conduct sequential testing (always until n.max, with peeks at pre-determined places)  
for(i in 1:iter){  
sample <- rnorm(n.max, 0, 1)  
if(test==”t.test”){res[i,] <- sapply(peeks, FUN=function(x){mean(sample[1:x])/sd(sample[1:x])\*sqrt(x)})}  
if(test==”z.test”){res[i,] <- sapply(peeks, FUN=function(x){sum(sample[1:x])/sqrt(x)})}  
}

# Create matrix: Which tests are significant?  
if(test==”z.test”){  
ifelse(alternative==”two.sided”, signif <- abs(res) >= qnorm(1-alpha), signif <- res <= qnorm(alpha))  
}else if (test==”t.test”){  
n <- matrix(rep(peeks, iter), nrow=iter, byrow=T)  
ifelse(alternative==”two.sided”, signif <- abs(res) >= qt(1-alpha, df=n-1), signif <- res <= qt(alpha, df=n-1))  
}

# Create matrix: Has there been at least one significant test in the trial?  
seq.signif <- matrix(NA, ncol=length(peeks), nrow=iter)

for(i in 1:iter){  
for(j in 1:ncol(signif)){  
seq.signif[i,j] <- TRUE %in% signif[i, 1:j]  
}  
}

# Determine the sequential alpha error probability for the sequential tests  
seq.alpha <- apply(seq.signif, MARGIN = 2, function(x){sum(x)/iter})

# Return a list of individual test p-values, sequential significance and sequential alpha error probability  
return(list(p.values = res,  
seq.significance = seq.signif,  
seq.alpha = seq.alpha))

}

[/cc]