Understanding Significance Testing as the Quantification of Contradiction

Conceptual and Practical Foundations of Statistics in the Context of Experimental Research

Thomas E. Gladwin

1. Introduction

This text is meant to help strengthen statistical understanding, leaning strongly on the use of programming simulations rather than formal mathematics. It's not an introductory textbook, but more something to (re-)read alongside your usual textbook during or after undergraduate modules. I'm not going to explain how to calculate a mean or a variance, for instance. Nevertheless, in a sense it is a "pre-statistics" book, in that it's about very basic conceptual foundations. You'll also learn a tiny bit about data analysis and simulation using Python.

A tiny bit of motivating philosophy of science to start. Perhaps you feel that "truth" is a hopelessly outmoded and naive concept; however, I would posit that it's very clear that lies do exist; stupidity exists; gaslighting exists; mistakes exist; and biases exist. If so, the opposite of those things exists, which we might as well call truth: the things that lies and errors take us away from. When we do a scientific experiment, we want to move towards the truth about something. And we need statistics for that to work. Oh really, I hear smug, evil-eyed quasi-scientists ask, why do we can't we just look and use our intuition? Look, we can make up loads of non-statistical rules to follow completely rigorously, and they can involve quite onerous activities, so why aren't those just as valid a method? The reason is that statistics was developed precisely to prevent ourselves from making fools or ourselves because we as humans are absolutely untrustworthy when it comes to interpreting data. We recognize patterns in complete noise, unless you believe that cloud over there actually is a giant shark or there actually are faces looking at you from a huge range of surfaces involving swirly patterns. What a terrifying life you must lead! And of course, we have a history of bitter disagreements between people who have different interpretations or what they see. If you want to base science on subjectivity, did you mean your pure and insightful subjectivity, or that garbage professor Johnson's depraved and moronic subjectivity? Should I then add in my own subjective perspective to decide which of your subjective perspectives is better? But maybe your subjective perspective on which of those subjective perspectives differs from mine… I prefer the alternative: Let's make a good-faith effort to find ways to measure things as objectively as we can and draw conclusions that tend to be valid, i.e., that will move us towards truth in the long term. We'll still be wrong, a lot, but if we work to falsify beliefs that are untrue then at least we can hope that by elimination we'll tend to retain beliefs that are true.

There are lots of ways we can fall short in our attempt to know the truth, and all these ways generate uncertainty. The most basic type of uncertainty, the thing that's absolutely essential for understanding basic statistics, is called ***sampling error***. This is what happens when we ***want*** to know

the truth about a so-called ***population***, but we only have a ***sample*** to draw inferences from. The "sample" is the set of data we've actually collected and can now observe, for instance, questionnaire scores from participants in a study. The "population", in contrast, is everyone who ***could have*** gone into the sample; it's the ***total potential pool*** of participants. Our problem is the gap between the specific sample and the larger population – we have the data of the sample, but we want to ***infer*** something about the population. Every time we draw a different sample from a population, that sample could lead us to draw a different conclusion. For instance, we may urgently need to know the average airspeed velocity of an unladen swallow. If we cannot avoid the question, we could capture a swallow and measure how fast is flies. OK. But maybe we captured a particularly slow or fast bird. Well, let's look at a different one. Oh, this one is faster! Well, we can take their average. But that average would be different if we'd captured two different swallows… We want to know the true average of all unladen swallows in general, not just the two we happen to have in hand!

We're going to be using Python for hands-on exercises (or use a different language like R if you prefer and are happy to translate the code). Please actually do the programming to build skills and avoid only a very superficial grasp of the material. Install Python and, if necessary, follow a quick basic tutorial on Python programming. I'd suggest using the Anaconda distribution for Windows, and working in the Anaconda prompt, using a text editor to make scripts.

Let's ***simulate*** the problem of the airspeed velocity of the unladen swallow. Simulations are the poor man's version of mathematic sophistication, to paraphrase my PhD supervisor; they're incredibly helpful to understand things but also to test and calculate things in practice. We're going to play God and create a population of sparrows, and, being, omniscient, we now know the truth: the airspeed is 11 m/s. But we're going to simulate the capture of random pairs from that population – we'll look at what happens in 10 different captures. Put this code in a Python script (e.g., swallows.py) and run it (`python swallows.py` in the Anaconda prompt) to create the population; ***do this before continuing***:

```
# Install the numpy toolkit for scientific computing, which
has functions for doing data analysis

import numpy as np


# Create the population of faster and slow swallow

# We'll first create a random population generated by sampling
from a standard normal distribution, using randn(). Note this
special case of a sampling procedure is to make our
population; we'll be drawing "samples" in a statistical sense
from our population.

# Then we'll set the population-mean to exactly 11,
representing 11 m/s. First we remove the mean we got after the
first random generation, so we know the mean must be exactly
zero at that point; then we add 11 to all the values.

# Finally we'll output the population to check what's going
on.

Total_number_of_swallows = 50

Speeds_Population = np.random.randn(Total_number_of_swallows)
```

```
Speeds_Population = Speeds_Population -
np.mean(Speeds_Population)

Speeds_Population = Speeds_Population + 11

print(Speeds_Population)

print(np.mean(Speeds_Population))
```

So now we've played God and have a population (abstracted to an array of airspeeds, one per swallow) with an average airspeed velocity of 11 m/s. But now let's take the role of non-gods who don't have omniscience; but who can catch birds. Add this code to represent a random capture of a pair of birds:

```
# Now we're going to draw a random sample of two swallows.

My_Sample = np.random.choice(Speeds_Population, size=2,
replace=False)

print('The captured sparrows had speeds:')

print(np.round(My_Sample, 2))

print('The average speed is this sample was:')

print(np.round(np.mean(My_Sample), 2))
```

Run the program a few times and note how every time you pick a new random sample, the sample-mean changes – but the true mean, i.e., the mean of the population, is set to exactly 11 m/s! ***This fluctuation of the sample-mean around the population-mean is sampling error***. And it means that, if you just look at the sample-mean and say, hah, now I know the population-mean, you're almost always going to be wrong. But statistics will tell you ***how*** wrong you're likely to be.

2. Very basic probability: Infinite imaginary worlds

There's a little word in that last sentence – "likely". All of statistics involves uncertainty and probability. If you dislike the concept of uncertainty, get over it quickly – we can make ***good faith efforts*** to be ***as objective as possible*** and to aim at ***approaching truth***, but we're not an omniscient God except when we're running simulations. But what is probability?

People disagree about how to conceive of probability. The right way is called "frequentism" and is sadly often misunderstood and/or misrepresented by its critics. The essential thing to understand about frequentism is that it's all about infinite imaginary worlds, or the multiverse if you prefer. What we do in frequentism is ***consider all the relevant outcomes that could have happened*** and define the probability of a certain event as the number of outcomes that fall within that event, divided by the number of possible outcomes. An event, for instance, could be that a dice throw gives a value of 3 or lower; the outcomes belonging to that event would be throws 1, 2, or 3; and there are 6 possible outcomes; so the probability of the event would be 3/6 = 0.5. One of the strawmen arguments used against frequentism is that many outcomes couldn't be repeated many times. It's hopefully clear that this is silly, since we're talking about a model of abstract, conceptual, counterfactual outcomes – every time we think in terms of frequentist probability, we open an imaginary multiverse of infinite worlds! This is an incredibly adaptable way to define probability, in combination with the axiomatic definitions from mathematics (not covered here, but highly

recommended you learn enough maths for that). In statistics, our "outcomes" are generally the scores we get in a randomly selected sample, and the probability of an event involves the proportion of outcomes with a certain feature, e.g., having a sample-mean above a certain minimum value.

We generally aren't going to need to do the maths that go into statistical techniques, but mostly for the sake of tradition we need to understand a general point: the probability of a certain kind of event often depends on ***parameters*** called "degrees of freedom" or *df*. This parameter generally has to do with how many random values are being added together to calculate a sample-mean (or a sample statistic in general – a "statistic" is any score that we calculate using data from a random sample). As we'll see later, the *df* affect the kinds of sample-means you're likely to get and therefore what inferences you can draw from an observation.

3. Does a sample-mean "significantly" differ from a hypothesized value?

In this section I'll introduce the core concept of "significance" and "significance testing". This is all about thinking about ***sampling error***.

We're going to expand the swallow simulations. Instead of just taking one sample – i.e., capturing a pair of swallows once – we're going to make the code simulate taking 100 samples. We're just going to put the random sampling in a loop, as follows; note that I've put the number of birds we're going to capture ***in each separate sample*** in a variable. ***Run*** the code below:

```
# Now we're going to simulate what happens when we take a lot
of different samples.

Number_of_samples = 100

Birds_per_sample = 2

for this_sample in range(Number_of_samples):

    My_Sample = np.random.choice(Speeds_Population,
size=Birds_per_sample, replace=False)

    print(np.round(np.mean(My_Sample), 2))
```

Look at the numbers this generated: each number represents the sample-mean (the average of the two birds you captured) of one of the samples you took. Notice how the sample-means vary! What range do the value have, roughly, what minimum and maximum do you see? In my simulations, I'm seeing sample-means vary very roughly between 9 and 12.

Note down the last sample-mean you get. For me that was 11.71. We're going to use that later, as if this was the value we got doing real-life data collection.

Now, in our simulations we're an omniscient God, but as scientists we only have a particular sample to work with (this can lead to bitterness and is why some scientists are atheists). Because of sampling error, we can't just look at our ***sample-mean*** and say, hah, this is the ***population-mean***, and pretend we now indeed know the average airspeed velocity of a sparrow. What we can do, however, is ***test hypotheses*** about the population-mean. We can check whether a certain belief is ***contradicted*** by the observed data. This is pretty much a concrete way of being a Popperian and making scientific progress via falsification: We make a guess about the population and then take samples to check whether what we observe contradicts our guess. We call such a testable guess about the population a ***hypothesis***.

Let's say someone has the hypothesis that the average airspeed velocity of a sparrow is 22 m/s. OK, let's simulate the outcomes (the sample-means, in this case) that we would expect if that were true. We're going to play God again and now use our omnipotence to make the population-mean 22. Find the appropriate line and replace it with the code below and ***run it*** before continuing.

```
Population_mean = 22

Speeds_Population = Speeds_Population + Population_mean
```

Run the code again and look at the list of sample-means. These now cluster around 22, roughly between 21 and 23, right? Now look up the "real-life" value you found when the population-mean was set to be 11. That was 11.71, Do we ever find a value that far away from 22 in all the new sample-means? Nothing like it. That means that when we found our "real-life" value, it completely ***contradicted*** what we would expect to find ***if it were true*** that the population-mean equals 22.

In statistical terminology, we would say: we found a sample-mean that ***significantly differed*** from 22. The "significance" of an effect is the degree to which an observed effect is ***unlikely*** relative to a given hypothesis. We're seeing something surprising, something we would not expect given a belief about the population. We use this contradiction to falsify – it's evidence against the belief. Remember, we always ***falsify***, never ***verify***. At best, as sceptical scientists, we decide not to reject a hypothesis - for now.

Memorize this phrase: *Significance is about how unlikely an observation is, relative to a given hypothesis about the population.*

Memorize this phrase: *If an observation is unlikely if a hypothesis is true, then the observation contradicts the hypothesis.*

But now we need to get a more precise, objective way of measuring this contradiction between a hypothesis and an observation. It's not always going to be as obvious as in the previous example! For example, what if we want to test the belief that the average airspeed velocity of a swallow is 12? Well, adjust ***and run*** the code, again replacing the line omnipotently setting the hypothesized population speed.

```
Population_mean = 12
```

Just by eyeballing the sample-means I'm seeing a range that now easily covers the 11.71 that I observed when I did my "real-life" data collection. Would seeing that 11.71 raise a red flag of contradiction at all? How can we objectively quantify if an observation is unlikely or not?

We do this by defining lower and upper ***critical values*** of sample means, which define a range of sample means that are "suspiciously low" or "suspiciously high". Traditionally, we consider the critical values to be those such that only 5% of the sample means are either below the lower critical value or above the upper critical value. We could, however, use different percentages than 5% to define critical values that define "extremeness" more or less strictly.

Memorize this phrase: *Only a small percentage of all random samples have a sample-mean below the lower critical value or above the upper critical value.*

We can estimate the critical values using our simulations. Find and adjust the relevant code as below to store the simulated sample means in an array, instead of only printing them to screen; then we'll calculate the absolute difference of each sample-mean from the hypothesized population-mean. We want the absolute difference because we consider either abnormally low or abnormally high values

to contradict a hypothesis. We'll then sort these differences from low to high and find which one covers the top 5%. Note how I'm printing variables to screen to have an idea of what's going on.

```
# Now we're going to simulate what happens when we take a lot
of different samples.

# We're going to use this to approximate the critical values.

Number_of_samples = 100

Birds_per_sample = 2

Array_of_sample_means = np.array([]);

for this_sample in range(Number_of_samples):

    My_Sample = np.random.choice(Speeds_Population,
size=Birds_per_sample, replace=False)

    Array_of_sample_means = np.append(Array_of_sample_means,
np.mean(My_Sample))

print('Simulated sample-means:')

print(np.round(Array_of_sample_means, 2))

Array_of_extremeness = np.abs(Array_of_sample_means -
Population_mean)

Sorted_extremeness = np.sort(Array_of_extremeness)

print('Sorted extremeness of sample-means:')

print(np.round(Sorted_extremeness, 2))

Critical_value_index = int(np.floor(0.95 *
len(Sorted_extremeness)))

print("Index in array containing critical extremeness: " +
str(Critical_value_index))

Critical_absolute_difference =
Sorted_extremeness[Critical_value_index]

Critical_value_lower = Population_mean -
Critical_absolute_difference

Critical_value_upper = Population_mean +
Critical_absolute_difference

print("Lower and upper critical values: " +
str(np.round(Critical_value_lower, 2)) + " and " +
str(np.round(Critical_value_upper, 2)))
```

The lower and upper critical values I got were 10.17 and 13.83. Anything lower than 10.17 and anything higher than 13.83 is "extreme", i.e., is something that's less than 5% likely for me to find in a sample. My observed value of 11.71 ***wasn't extreme enough*** to fall in the critical range. So: The observed sample mean of 11.71 ***doesn't significantly differ*** from the hypothesized population mean

of 12. My observed sample-mean doesn't let me reject the hypothesis that the average airspeed velocity of the swallow is 12.

Thinking in terms of critical values that tell us which ranges of extreme observations are unlikely is the basis of understanding statistical significance testing. However, what often happens using modern software seems to be a little different. We don't explicitly set our desired unlikelihood of extreme observations (usually 5%), then calculate critical values, and finally check whether the sample-mean fell in the critical range. The software will give us a so-called *p*-value (this is sometimes labelled "the significance" or "Sig.", as in SPSS). But this *p*-value actually has everything to do with critical ranges. It's telling us what the probability of extreme observations ***would have been***, ***if*** we had used the observed sample-mean (or sample-statistic in general) as the critical value. If the *p*-value is ***below*** 5%, then the critical values would be stricter, covering a smaller part of the tails of the sample-mean distribution: an even smaller percentage of observation than 5% would be considered extreme, with the observed sample-mean as the critical value. So, if the *p*-value is ***lower than 5%***, we know the observed sample-mean would be extreme enough to be ***beyond the critical values*** that would classify 5% of sample-means as extreme. Therefore, when we run a statistical test and we get a *p*-value below .05, the effect being tested is significant.

Memorize this phrase: *The p-value tells us how unlikely the observed sample-statistic is, relative to a given hypothesis.*

Memorize this phrase: *If the p-value is below .05, there is a significant difference from the hypothesized population. This means we reject the hypothesis.*

Sometimes it's unintuitive for students that ***we have to see a small p-value*** to conclude significance, so be very careful you remember and try to understand this! E.g., if *p* = .0043, the effect is significant because .0043 is smaller than .05; but if *p* = .56, the effect is non-significant, because .56 is larger than .05.

Let's estimate the *p*-value of 11.71. We're going to use frequentism to calculate the probability of finding a sample-mean as "extreme" as 11.71, if the population-mean is 12. We can use our sorted extremeness array for this and just count. ***Add and run*** the following code:

```
Observed_sample_mean = 11.71

Observed_extremeness = np.abs(Observed_sample_mean -
Population_mean)

At_least_as_extreme = np.argwhere(Array_of_extremeness >=
Observed_extremeness)

Number_of_at_least_as_extreme = len(At_least_as_extreme)

p = Number_of_at_least_as_extreme / len(Array_of_extremeness)

print("The p-value of " + str(np.round(Observed_sample_mean,
2)) + " is " + str(np.round(p, 4)))

print("There were " + str(Number_of_at_least_as_extreme) + "
more extreme values in the simulated samples.")

print("There were " + str(len(Array_of_extremeness)) + "
simulated values in the array in total.")
```

I got a *p*-value of .63: it's ***non-significant***, the *p*-value is ***too big*** (larger than .05), which agrees with the observed value ***not falling in the extreme critical range***.

This is how we can test whether a sample mean significantly differs from a hypothesized population mean! Of course, we've been using very simple simulations to approximate our probabilities, and proper statistical software uses hardcore maths and does it all far better. But conceptually it's all the same: we want to know how unlikely an observed sample-statistic would be given the hypothesis about the population we're testing. It's very important to remember how we programmed these simulations and what we're looking at to define probabilities. The population-mean is something we control, and the probabilities of sample-means follow from that. This lets us talk about probabilities for sample-means relative to a given hypothetical population-mean; remember the way we calculated the *p*-value for an observed sample-mean above. We do not, however, have a "probability" for the population-mean – we don't have a big array of population means we're randomly selected population-means from! The population-mean is "above" probability; it's something we decide on as the God of the simulation; there is no probability even ***defined*** for the population-mean. All we do in significance testing is say: well, we can for sake of argument assume the hypothesis is true (i.e., we run our simulation with a given value for the population), and then check the probability of finding a sample-statistic as extreme as what we actually observe after data collection. We can't calculate a probability for the divine population mean in the same way we can for the sample-mean, because we don't have a big array of population means to use the argwhere() function on.

Memorize this phrase: *We can only talk about probabilities involving samples; there is no concept of probability defined for the population.*

Memorize this phrase: *The population is where the probabilities of sample outcomes comes from; the population doesn't have a probability itself.*

Of course, outside the strict confines of statistical significance testing we're certainly thinking about whether hypotheses are plausible or not. But we have to keep this strictly separate from the *p*-value, or we'll get very confused. Significance testing plays a central role in experimental research but is by no means the only thing we care about – it's one piece of evidence we can use to try to make good decisions and move towards less-wrong beliefs.

By far the most relevant kind of significance testing in experimental work is ***null hypothesis significance testing***, or NHST. This is where the hypothesis represents some kind of ***absence of an effect***. For instance, let's say that our scores represent a difference between two conditions that each participant has been exposed to, e.g., reaction times on interference versus control blocks of the Stroop task. Hypothesizing that the population-mean of the difference scores is zero formalizes and quantifies the belief that there is no true effect of Stroop interference. We then collect data and find a difference score of, say, 65 ms. Do we then conclude that there is Stroop interference? Well, no, because we know about sampling error. We need to calculate the *p*-value to be able to ***at least*** reject the null hypothesis. Note this phrasing: We don't have to believe the null hypothesis is actually true for NHST to make sense; it's often a priori implausible in real-life for there to be literally, absolutely zero difference. The reason we do NHST is to make sure that we can ***at least*** contradict the skeptic who should always live in our heads reminding us of sampling error. The less plausible the null hypothesis is in reality, the ***more*** relevant it is that we ***can't even*** reject it using our data. ***A non-significant test tells us we shouldn't be drawing conclusions about any apparent***

effects in our sample, no matter how much we might want to. Where we go from there is beyond statistics. Perhaps we try to find a more sensitive test, or we reconsider our measurements, or we check for some source of noise we hadn't considered; or perhaps a competing belief does lead to significant effects, in which case we'll provisionally prefer that one, and science hopefully moves a little further towards truth.

4. Capturing more swallows

Recall the critical values we got in our simulations of a population with a mean airspeed velocity of unladen swallows - 10.17 and 13.83 for me. So anything between 10.17 and 13.83 would be non-significant. That's quite broad, and included a value I know I got from a sample with a different population with a mean of 11 m/s. Ideally, we would have detected that. Is there a way we can "tighten up" the critical values?

First, ***run*** the script again and just have a look at the sample-means you get to get a sense of how much they vary. We can quantify this variation by storing the random sample-means we get per simulated data collection, and taking the standard deviation of all those different sample-means. We already stored the sample-means in an array, so we just have to add this to the bottom of the script:

```
Standard_error = np.std(Array_of_sample_means)

print("The standard error was " + str(np.round(Standard_error,
4)))
```

Run the script again and note down this ***standard error*** – this is the ***standard deviation of sample means*** over replications of the 100 simulated experiments. For me, it was 0.96. Note how the standard error is different from the usual standard deviation of single observations within one experiment. ***The standard error quantifies sampling error***. The bigger the standard error, the more wildly the sample means fluctuate around the population mean, and the further apart will be the lower and upper critical values. And therefore, the bigger the standard deviation, the more difficult it is to get a low-enough *p*-value for an effect to be significant. If we have a very tight distribution of sample-means, then even observed values only a little way away from the population mean could become extreme.

We can reduce the standard error by capturing more birds, since in each simulated experiment we'll be basing our sample-mean on the averaging of a larger number of speeds. It will be more and more unlikely for all the sampled speeds to be very high or very low, this averaging out differences from the population mean. Let's test this claim by changing the number of captured birds in the simulation-loop and ***running*** the script:

```
Birds_per_sample = 20
```

This reduced the standard error to 0.16 and tightened the critical values to 11.65 and 12.35. My 11.71 still wouldn't reach significance but you can see it's getting closer; and its *p*-value was indeed now lower, .1 instead of .63.

The important general point here is that the *p*-value of a certain value of a sample-statistic depended on a ***parameter***; namely, the number of birds captured per experiment, i.e., the sample size. That makes sense: the more birds we capture, the smaller the standard error, and the lower the *p*-value. Now, remember those degrees of freedom, *df*? Those are simply the parameters of a statistical test that determine what the *p*-value of a given observed value is. Just like in our example,

we ***need to know the parameter*** of Birds_per_sample to be able to say what a given sample-mean's *p*-value is – the *p*-value was .63 when Birds_per-sample was 2, but .1 when Birds_per_sample was 20. Why are they called "degrees of freedom"? The relevant parameters for the hardcore maths underlying to tests have to do with how many relevant values depend on random selection and thus "freely vary" over different samples. Unless you go into the mathematic of statistics, *df* are just a kind of nasty exposure of how the sausage is made, but traditionally you have to be able to calculate them per test and report them. They do allow for some quality and sanity checks, but, in my personal opinion, it would be a lot nicer if we reported the basic parameters like sample size and number of groups rather than the quasi-technical *df* (which are completely determined by such understandable parameters). Anyway.

5. Other statistical tests

So in the preceding sections, I've tried to illustrate the statistical concepts of inference from sample to population, sampling error, null hypothesis significance testing and *p*-values within the context of tests of sample-means. This is what a one-sample t-test or paired-sample t-test does, using proper software. The same concepts generalize to all the usual tests, except the null hypothesis – the thing our inner skeptic demands we ***at least*** need to reject to even ***think*** about looking at patterns in our sample – will mean something different. I'll briefly go over the very basic tests, but as an exercise for the reader, if there's one you really want to understand, do the same simulation trick as I went through above.

Between-group t-test: The null hypothesis is that ***two groups*** have the same population means. A significant effect ($p < .05$) means that the size of the difference between the groups is so extreme it contradicts the null hypothesis. Therefore: you can reject the null hypothesis and move forward to interpret the effect you're seeing as if it says something true.

ANOVA: The null hypothesis is that ***two or more groups*** all have the same population means. A significant effect can therefore mean lots and lots of things: Maybe one group differs from all the groups, maybe groups 1 and 2 differ from groups 3, 4, and 5, and so on. You'll need to run so-called post-hoc tests to figure out what pattern caused the significant ANOVA test. What approach makes sense will depend on your context. In experimental work, ***"protected"*** post-hoc tests seem most appropriate – this is where we just test whatever we like, e.g., pairwise t-test, and don't worry about multiple-testing correction. Our aim is ***descriptive*** – we know there's a significant effect already from the ANOVA, we're just using tests to help characterize the pattern. However, if you're really interested in formally testing specific group differences, this of course isn't appropriate and you'll need to use a multiple-testing correction method (see, e.g., NumNomS, https://github.com/thomasgladwin/NumNomS, for my improved version of Bonferroni correction).

Correlation: The null hypothesis is that there is no linear relationship between two variables X and Y. A (two-sided) significant effect means there's either a positive/negative relationship. You're probably fine just visualizing the usual kind of scatterplot here, but it's formally quantified as: over all data points in the population, values that are higher than average on X tend to be proportionally higher/lower than average on Y. (For a nonlinear kind of correlation, see nncorr on https://github.com/thomasgladwin/nncorr)

Regression: Here the essential thing to remember is that there are two kinds of tests in regression: First, we have the *F*-test that tests whether ***the whole model*** explains more sample-variance than expected under the null hypothesis that the model explains nothing in the population. In other

words, the null hypothesis is that the best model of the dependent variable has coefficients of zero for all predictors in the population. In samples there'll always be some spurious, sample-specific relationship that regression will exploit to explain some non-zero amount variance; the *F*-test tells you whether a sufficiently extreme amount of sample-variance is explained. The idea of "explained variance" comes from being able to write raw scores of the dependent variable as the sum of two ***uncorrelated*** components: predicted values and the error. Because in regression these components are uncorrelated, their variances must sum to the variance of the original scores, and we can just calculate the variance of the predicted values and see what its proportion is of the total original variance – this has to be (barring computational foibles perhaps) some proportion from 0 to 1. Second, we have the set of *t*-tests that test whether the coefficient of ***each specific predictor separately*** differs significantly from zero. The main way I use regression is hierarchical regression: I start with a baseline model of predictors unrelated to my hypothesis and use an *F*-test to see whether a nested model with additional hypothesis-related predictors explains a significant amount of additional variance. This lets me test *sets* of predictors. There's often a huge problem with looking at the individual *t*-tests to draw conclusions because of multicollinearity – the specific individual predictors will interfere with each other, unless they're stochastically independent. This is just what regression does: it will by definition find the best linear combination, and that means juggling the whole set of coefficients to suit that aim and that aim alone. People have made up rules that say interpreting individual predictors within a broader model is OK if certain metrics are within certain ranges, but these rules generally seem wrong if you check out the statistical literature.

6. Conclusions

In conclusion, I hope this text has provided some helpful concepts, maybe some useful snippets of Python, and an introduction to simulations as a general trick for understanding things. You may even have thought of ways to make better simulations to actually do significance testing for non-standard analyses. This trick has let me develop analyses for Genome-wide Association Studies that were familywise error-corrected but didn't require obscenely low *p*-values per SNP; tests of time series of heart rate and body sway data; whole-brain tests of activation blobs in fMRI; and tests of white matter tract abnormalities. Whatever the data pattern: If you can simulate it, you can permute it, and then you can significance test it.

You might have noticed that I'm working from the perspective of experiment research, where we expect to do many different studies on a particular topic and our aim is to test ideas. For this kind of research, NHST is perfectly valid: we're likely to try a lot of things that don't work and we want a feedback mechanism telling us whether we're being pointed in the right direction or just getting fooled by fake patterns. ***The reason we use p-values is to stop ourselves following will-o'-the-wisps in the data*****.** Other kinds of research and researchers might well prefer different kinds of statistics, for instance, if instead of smaller studies that ideally should be replicable you have one large, possibly unique dataset; or when instead of testing which ideas are best, you really just want to say something about the actual population values. These are very different aims, related to "***epidemiological" versus "experimental"*** research perhaps. I suspect this is where much of the sometimes really irrationally negative vilification of NHST comes from: Of course NHST isn't the right tool to use if your whole aim is to estimate the actual numeric population values and quantify uncertainty about them. Sure, go for confidence intervals and Bayesian statistics if they suit your purposes better. But don't be a fanatic or a Reviewer #2 about it towards experimentalists, and don't confuse the issue with strawmen. Students, and plenty of academics, are confused enough

about statistics, and the warfare with its associated rhetoric between methodological approaches and preferences often seems more counterproductive than clarifying. All the serious statistical problems related to statistical testing, such as in the replication crisis in psychology, obviously involved the more widespread NHST, but these problems could easily translate to and infect other approaches to statistics. Any metric dependent on the sample is vulnerable to changing the sample inappropriately and all the dishonest methodological and pre-processing tricks to get a significant *p*-value could be used to get a more preferred result in Bayesian analyses. I would argue that, in the debate on how psychologists should analyze their data, conceptual and mathematical simplicity of statistical approaches should be seen as a desirable dimension; obviously in the context of the necessity that results are valid. First, there are limited resources in science and spending them on dealing with unnecessary statistical barriers is unethical; second, overly complicated methods requiring highly specialist expertise to be used reduce the level of competence and responsibility of researchers and hence could well introduce errors. I recall, for example, a “fancy” method that almost everyone used but almost nobody could understand, and that turned out to be wrong only after years and years. For many purposes, NHST is simple and valid and lets experimental researchers get on with all the other elements of their work. In closing, I’d therefore raise the concern that well-meaning but heavy-handed imposition of new statistical rules, e.g., by journals, may be inappropriate and counterproductive. Keep it Simple, Statistically.