STA303/1002 Portfolio

An exploration of linear mixed models and common misconceptions in statistics

Xiaotang Zhou

2022-02-17

Contents

List of Figures

Introduction

Across the three main sections of Statistical Skills Sample, Writing Sample, and Reflection in this portfolio, the applications of many different statistical methods are showcased and discussed. From a communicative standpoint, this portfolio seeks to demonstrate highly important core transferable skills in the context of several applicable settings.

In the Statistical Skills Sample section of this portfolio, focus is put on setting up and formatting a document that not only examines several statistical methods, but can also be reproduced and easily understood by both a statistically proficient and non-statistically proficient audience. These methods include loading the appropriate libraries and modules for use throughout the course of the document, fitting, interpreting, and differentiating between different types of linear models, creating interactives that interpret often incorrectly understood and misconstrued statistical objects such as confidence intervals and p-values, creating reprexes (reproducible examples) of code that can aid in debugging and editing, and examining the effect that sampling from different distributions can have on hypothesis tests.

Next, in the Writing Sample section, an analysis and evaluation of Motulsky's "Common Misconceptions About Data Analysis and Statistics" is conducted, with a primary focus on what factors fuel the research reproducibility crisis from the scope of p-values and hypothesis testing, and how the commonly held belief among statisticians, scientists, and researchers that a large sample size is never a bad thing could be worsening instead of helping to alleviate this crisis.

Finally, in the Reflection section, the accomplishments, possible applications, and possible shortcomings and improvements that were realized during the creation of this portfolio are discussed.

Statistical skills sample

Task 1: Setting up libraries and seed value

```
# Code for keeping warnings off R Markdown document
knitr::opts_chunk$set(warning = FALSE, message = FALSE)

# Load required libraries
library(tidyverse)
library(lme4)
library(reprex)

# Create the last3digplus object and assign the last 3 digits
# of my student number plus 100 to it
last3digplus <- 859 + 100
```

Task 2a: Return to Statdew Valley: exploring sources of variance in a balanced experimental design (teaching and learning world)

Growing your (grandmother's) strawberry patch

```
# Load grow_my_strawberries() function
source("grow_my_strawberries.R")

# Create the my_patch object and assign the output of the
# grow_my_strawberries(last3digplus) function to it
my_patch <- grow_my_strawberries(seed = last3digplus)

# Turn the treatment variable into a factor variable with the
# given ordering of "No netting < Netting < Scarecrow"
my_patch$treatment <- fct_relevel(my_patch$treatment, "No netting", after = 0)
```

Plotting the strawberry patch

```
# From top to bottom, left to right: set x and y data, set background colours,
# set plot as scatterplot with triangle-shaped points, set colour of points in
# accordance with treatment variable, add caption
my_patch %>%
  ggplot(aes(x = patch, y = yield, color = treatment, fill = treatment)) +
  geom_point(pch = 25) +
  scale_color_manual(values = c("#78BC61", "#E03400", "#520048")) +
  scale_fill_manual(values = c("#78BC61", "#E03400", "#520048")) +
  theme_minimal() +
  labs(caption = "Created by Xiaotang Zhou in STA303/1002, Winter 2022")
```

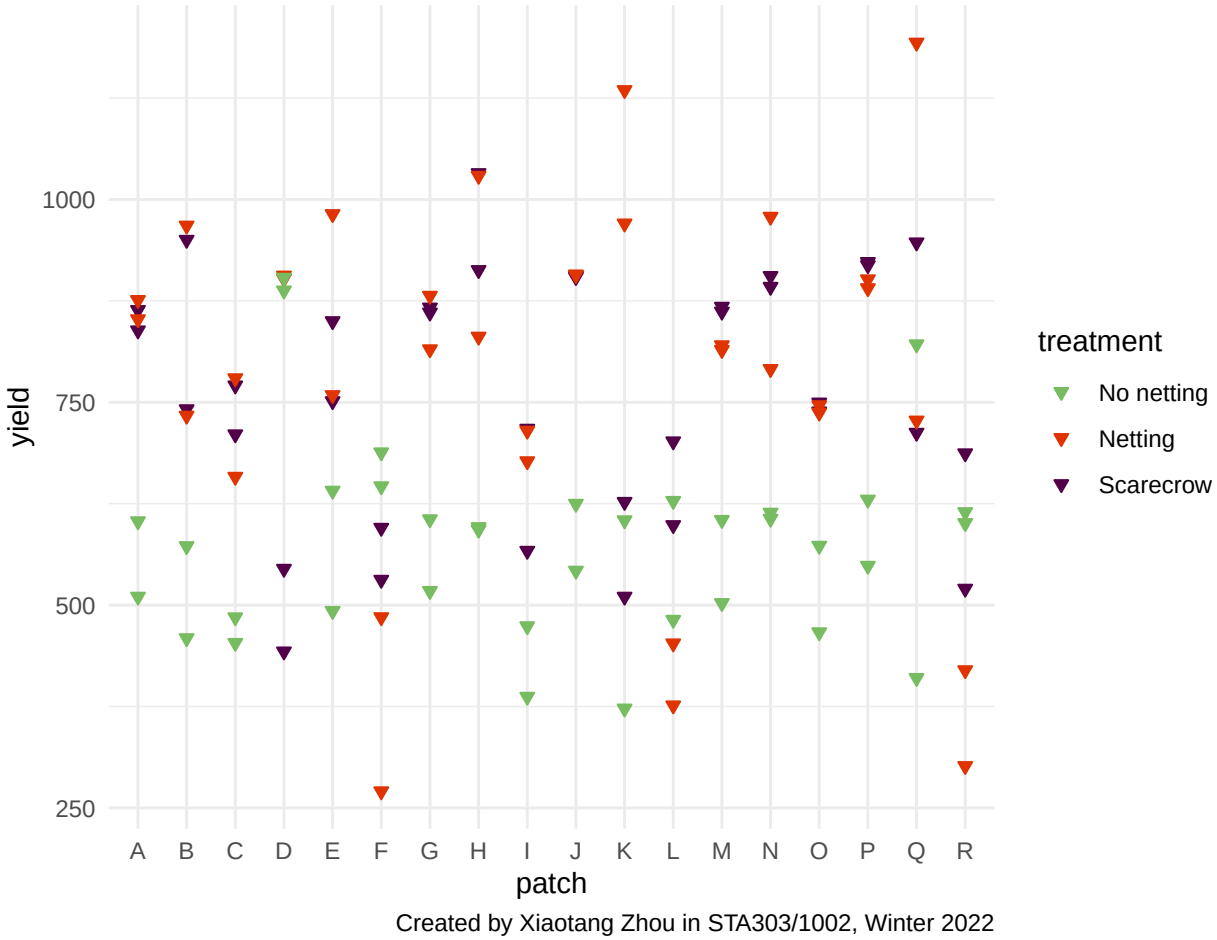

Figure 1: The associations between different patches on the strawberry farm and crop yield, subject to the treatments applied to keep birds away

Demonstrating calculation of sources of variance in a least-squares modelling context

Model formula The model formula is written mathematically as follows:

$$y_{ijk} = \mu + \alpha_i + b_j + (\alpha b)_{ij} + \epsilon_{ijk}$$

where:

- y_{ijk} is the amount of yield from patch j using treatment i the kth time, where $k \in \{1, 2\}$, $i \in \{$"No Netting", "Netting", "Scarecrow"$\}$, and $j \in \{A, B, ..., Q, R\}$
- μ is the grand mean of yield
- α_i are the fixed effects for treatment
- b_j are the random effects for patch with distribution $b_j \sim N(0, \sigma_b^2)$
- $(\alpha b)_{ij}$ are the random effects for the interaction between treatment i and patch j with distribution $(\alpha b)_{ij} \sim N(0, \sigma_{\alpha b}^2)$
- ϵ_{ijk} are the residual error components with distribution $\epsilon_{ijk} \sim N(0, \sigma^2)$

Now, we will create some objects that will aid our analysis of this model. First, we will create some new datasets from our existing one:

```
# Create the agg_patch object by assigning to it a new dataset where each
# patch is given 1 row and their average yields are recorded
agg_patch <- my_patch %>%
  group_by(patch) %>%
  summarise(yield_avg_patch = mean(yield), .groups = "drop")

# Create the agg_int object by assigning to it a new dataset where each
# patch-treatment combination is given 1 row and their average yields are
# recorded
agg_int <- my_patch %>%
  group_by(patch, treatment) %>%
  summarise(yield_avg_int = mean(yield), .groups = "drop")
```

Next, we will create some secondary models using both the original dataset and the new ones we have created:

```
# Create the int_mod object by assigning it the interaction model to predict
# yield based on patch and treatment
int_mod <- lm(yield ~ patch * treatment, data = my_patch)

# Create the patch_mod object by assigning it the intercept-only model
# predicting average yield by patch
patch_mod <- lm(yield_avg_patch ~ 1, data = agg_patch)

# Create the agg_mod object by assigning it the main effects model to predict
# average yield by both patch and treatment based on patch and treatment
agg_mod <- lm(yield_avg_int ~ patch + treatment, data = agg_int)
```

Finally, we will assign some objects some numbers corresponding to the above models:

```
# Create the var_patch object by assigning to it the variance in yield
# explained by patch to patch variance. We divide the second term by 3
# because we are averaging over the number of treatments, which is 3
var_patch <- summary(patch_mod)$sigma^2 - (summary(agg_mod)$sigma^2) / 3

# Create the var_int object by assigning to it the residual variance after
# fitting the version of this linear model with the most parameters
var_int <- summary(int_mod)$sigma^2

# Create the var_ab object by assigning to it the variance in yield explained by
# the interaction of treatment and patch. We divide by 2 because there are
# 108 total observations and 18 * 3 = 54 interactions, so 108 / 54 = 2
var_ab <- summary(agg_mod)$sigma^2 - var_int / 2
```

```
# Create a table detailing each of the possible sources of variation in the
# model, the corresponding variances that were previously calculated, and the
# proportion of variance in yield accounted for by each source
tibble(Source = c("treatment:patch", "patch", "residual"),
  Variance = c(var_ab, var_patch, var_int),
  `Proportion of variance explained` = c(
    round(var_ab / (var_ab + var_patch + var_int), 2),
    round(var_patch / (var_ab + var_patch + var_int), 2),
    round(var_int / (var_ab + var_patch + var_int), 2))) %>%
  knitr::kable(caption = "Summary of the possible sources of variation in
  the model, and how much variation each source accounts for")
```

Table 1: Summary of the possible sources of variation in the model, and how much variation each source accounts for

Source	Variance	Proportion of variance explained
treatment:patch	15415.866	0.54
patch	3374.343	0.12
residual	9897.597	0.35

Task 2b: Applying linear mixed models for the strawberry data (practical world)

First, we will create the three models that we will be working with:

```
# Create the mod0 object by assigning it the simple linear model to predict
# yield based on treatment
mod0 <- lm(yield ~ treatment, data = my_patch)

# Create the mod1 object by assigning it the linear mixed model
# predicting yield with treatment as a fixed effect and patch as
# a random effect
mod1 <- lmer(yield ~ treatment + (1 | patch), data = my_patch)

# Create the mod2 object by assigning it the linear mixed model
# predicting yield with treatment as a fixed effect, patch as
# a random effect, and the interaction between treatment and patch as
# a random effect
mod2 <- lmer(yield ~ treatment + (1|patch) + (1|treatment:patch), data=my_patch)
```

Next, we will run likelihood ratio tests between each model to compare them with each other:

```
# Test the hypothesis that the simple linear model is just as good as the linear
# mixed model with treatment as a fixed effect and patch as a random effect, and
# output the p-value for this test
lmtest::lrtest(mod0, mod1)[[5]][2]
```

```
## [1] 3.697603e-09
```

```
# Test the hypothesis that the simple linear model is just as good as the linear
# mixed model with treatment as a fixed effect, patch as a random effect, and the
# interaction between treatment and patch as a random effect, and output the
# p-value for this test
lmtest::lrtest(mod0, mod2)[[5]][2]
```

```
## [1] 6.242011e-13
```

```
# Test the hypothesis that the linear mixed model with treatment as a fixed
# effect and patch as a random effect is just as good as the linear mixed
# model with treatment as a fixed effect, patch as a random effect, and the
# interaction between treatment and patch as a random effect, and output the
# p-value for this test
lmtest::lrtest(mod1, mod2)[[5]][2]
```

```
## [1] 3.67397e-06
```

For the above models and tests comparing them, the restricted maximum likelihood (REML) variation of maximum likelihood estimation was used. As we are comparing nested models that have different random effects and our goal is to estimate various statistics about the models such as the coefficients and variances, REML can be justified to be the better option over ML.

Justification and interpreation

Through constructing and comparing the three models previously, we conclude that the linear mixed model with treatment as a fixed effect, patch as a random effect, and the interaction between treatment and patch as a random effect (mod2) is the most appropriate final model. This is because by looking at the p-values for the three likelihood ratio tests conducted previously,

we can see in all three tests that there is very strong statistical evidence against the hypothesis that the simpler model is just as good as the more complex one. This allows us to take the most complex model mod2 as our final model.

Now, having chosen mod2 as our final model, first consider the coefficients of its fixed effects:

```
# Output the coefficients of the fixed effects for mod2
summary(mod2)$coefficients
```

```
##                    Estimate Std. Error   t value
## (Intercept)        576.8292   36.31582 15.883688
## treatmentNetting   206.3150   47.56836  4.337231
## treatmentScarecrow 187.6428   47.56836  3.944697
```

We can interpret these coefficients as saying that on average, when netting is the treatment used on the patch to stop birds from eating the strawberries the yield is about 206 higher, while when a scarecrow is the treatment the yield is about 188 higher on average.

Next, to analyze the proportion of variance not explained by the fixed effects, consider the variances of the random effects:

```
# Output the variances of the random effects of mod2
print(VarCorr(mod2), comp = "Variance")
```

```
##  Groups          Name        Variance
##  treatment:patch (Intercept) 15416.0
##  patch           (Intercept)  3374.4
##  Residual                     9897.6
```

As we can see above, the interaction between treatment and patch accounts for $\frac{15416}{15416+3374.4+9897.6} = \frac{15416}{28688} \approx 0.54 = 54\%$ of the variation in yield unexplained by the fixed effects, patch to patch variation accounts for $\frac{3374.4}{15416+3374.4+9897.6} = \frac{3374.4}{28688} \approx 0.12 = 12\%$ of the variation in yield unexplained by the fixed effects, and the residual variance accounts for $\frac{9897.6}{15416+3374.4+9897.6} = \frac{9897.6}{28688} \approx 0.35 = 35\%$ of the variation in yield unexplained by the fixed effects. This matches exactly with the results obtained in Task 2a, which means the final model was selected correctly.

Task 3a: Building a confidence interval interpreter

```
# A function that returns an interpretation of a confidence interval
# given its lower bound, upper bound, confidence level, and a description
# of the statistic of interest
interpret_ci <- function(lower, upper, ci_level, stat){

  # Produces a warning if stat isn't
  # a character string
  if(!is.character(stat)) {
    warning("
    Warning:
    stat should be a character string that describes the statistics of
    interest.")
  }

  # Produces a warning if lower isn't numeric
  else if(!is.numeric(lower)) {
    warning("
    Warning:
    lower should be a numeric value corresponding to the lower
    bound of your confidence interval.")
  }

  # Produces a warning if upper isn't numeric
  else if(!is.numeric(upper)) {
    warning("
    Warning:
    upper should be a numeric value corresponding to the upper
    bound of your confidence interval.")
  }

  # Produces a warning if ci_level isn't numeric or is a number
  # below 0 or is a number above 100
  else if(!is.numeric(ci_level) | ci_level < 0 | ci_level > 100) {
    warning("
    Warning:
    ci_level should be a numeric value corresponding to the confidence
    level of your confidence interval, and thus it must take a value of
    at least 0 and no more than 100.")
  }
```

```
  # Print interpretation
  else{
    str_c("We are ", ci_level, "% confident that the true value
          of the ", stat,
          " lies between ", lower, " and ", upper, ".")
  }
}

# Test 1
ci_test1 <- interpret_ci(10, 20, 99, "mean number of shoes owned by students")

# Test 2
ci_test2 <- interpret_ci(10, 20, -1, "mean number of shoes owned by students")

# Test 3
ci_test3 <- interpret_ci(10, 20, -1, tibble(stat = 3))
```

CI function test 1: We are 99% confident that the true value of the mean number of shoes owned by students lies between 10 and 20.

CI function test 2: Warning: ci_level should be a numeric value corresponding to the confidence level of your confidence interval, and thus it must take a value of at least 0 and no more than 100.

CI function test 3: Warning: stat should be a character string that describes the statistics of interest.

Task 3b: Building a p value interpreter

```
# A function that returns an interpretation of a p-value
# given the p-value itself and a description of the null hypothesis
interpret_pval <- function(pval, nullhyp){

  # Produces a warning if nullhyp isn't a character string
  if(!is.character(nullhyp)) {
    warning("
            Warning:
            nullhyp should be a character string that describes
            the null hypothesis.")
  }
```

```
# Produces a warning if pval isn't numeric
else if(!is.numeric(pval)) {
  warning("
          Warning:
          pval should be a numeric value corresponding to the
          p-value that is to be interpreted.")
}

# Produces a warning if pval is a value greater than 1
else if(pval > 1 | pval < 0) {
  warning("
          Warning:
          pval should be a numeric value corresponding to the
          p-value that is to be interpreted, and thus it cannot take
          a value less than 0 or greater than 1.")
}

# Prints the interpretation for a pval value larger than 0.1
else if(pval > 0.1){
  str_c("As the p-value of ", round(pval, 3),
        " is larger than 0.1 and also larger than the threshold of 0.05,
        we conclude that there is no evidence against the hypothesis
        that ", nullhyp, ".")
}

# Prints the interpretation for a pval value less than
# or equal to 0.1 but larger than 0.05
else if(pval <= 0.1 & pval > 0.05){
  str_c("As the p-value of ", round(pval, 3),
        " is less than or equal to 0.1 but also larger
        than the threshold of 0.05, we conclude that there is
        weak evidence against the hypothesis that ", nullhyp, ".")
}

# Prints the interpretation for a pval value less than
# or equal to 0.05 but larger than 0.01
else if(pval <= 0.05 & pval > 0.01){
  str_c("As the p-value of ", round(pval, 3),
        " is larger than 0.01 but also less than or equal to
        the threshold of 0.05, we conclude that there is some
        evidence against the hypothesis that ", nullhyp, ".")
```

```
  }

  # Prints the interpretation for a pval value less than
  # or equal to 0.01 but larger than 0.001
  else if(pval <= 0.01 & pval > 0.001){
    str_c("As the p-value of ", round(pval, 3),
          " is larger than 0.001 but also less than or equal to 0.01
          and is thus less than the threshold of 0.05,
          we conclude that there strong evidence against the hypothesis
          that ", nullhyp, ".")
  }

  # Prints the interpretation for a pval value less than 0.001
  else if(pval < 0.001){
    str_c("Since the p-value is <0.001 and thus much less
          than the threshold of 0.05, we conclude that there is
          very strong evidence against the hypothesis that ", nullhyp, ".")
  }

}

pval_test1 <- interpret_pval(0.0000000003,
                             "the mean grade for statistics students is the same as
                             ↪  for non-stats students")

pval_test2 <- interpret_pval(0.0499999,
                             "the mean grade for statistics students is the same as
                             ↪  for non-stats students")

pval_test3 <- interpret_pval(0.050001,
                             "the mean grade for statistics students is the same as
                             ↪  for non-stats students")

pval_test4 <- interpret_pval("0.05", 7)
```

p value function test 1: Since the p-value is <0.001 and thus much less than the threshold of 0.05, we conclude that there is very strong evidence against the hypothesis that the mean grade for statistics students is the same as for non-stats students.

p value function test 2: As the p-value of 0.05 is larger than 0.01 but also less than or equal to the threshold of 0.05, we conclude that there is some evidence against the hypothesis that the mean grade for statistics students is the same as for non-stats students.

p value function test 3: As the p-value of 0.05 is less than or equal to 0.1 but also larger than the threshold of 0.05, we conclude that there is weak evidence against the hypothesis that the mean grade for statistics students is the same as for non-stats students.

p value function test 4: Warning: nullhyp should be a character string that describes the null hypothesis.

Task 3c: User instructions and disclaimer

Instructions

To use the confidence interval interpreter, enter the following arguments in this exact order into the function call: the lower bound of your confidence interval, the upper bound of your confidence interval, the confidence level of the interval, and a worded description of the statistic you are estimating in quote marks. For example, if you want to interpret a 95% confidence interval that states that the average number of steps taken per day by a person living in Toronto is between 4500 and 6500, your lower bound would be 4500, your upper bound would be 6500, your confidence level would be 95, your description of the statistic would be "mean number of steps taken per day by a Toronto resident", and your function call would look like the following:

interpret_ci(4500, 6500, 95, "mean number of steps taken per day by a Toronto resident")

To give some further insight into what exactly is being interpreted here, note that in the example given above, the population of interest would be everyone living in the city of Toronto, and the number we are trying to gain information on, also known as the population parameter, is the average number of steps that the people living in Toronto take on a daily basis. The hypothetical confidence interval above therefore gives plausible estimates of this population parameter. On the subject of interpreting confidence intervals, it also cannot be stressed enough that when interpreting confidence intervals, the confidence level should **never** be stated to be the probability that the population parameter lies between the lower and upper bounds, as given a set of lower and upper bounds, the population parameter is either between these bounds or isn't.

Next, to use the p-value interpreter, enter the following arguments in this exact order into the function call: the p-value that you want to interpret and a worded description of the null hypothesis of the hypothesis test from which your p-value was derived from in quote marks. For example, if you want to interpret a p-value of 0.004 that was obtained by testing the null hypothesis that the average number of steps taken in a day by residents of the city of Toronto is 5000, your p-value would be 0.004, your description of the null hypothesis would be "the mean number of steps taken in a day by residents of the city of Toronto is 5000", and your function call would look like the following:

interpret_pval(0.004, "the mean number of steps taken in a day by residents of the city of Toronto is 5000")

To give some further insight into what exactly is being interpreted here, note that in the example given above, the null hypothesis states that the average number of steps taken daily by a Toronto resident is 5000, and so the corresponding hypothetical p-value is a quantifier of the stength of the evidence **against** the null hypothesis that was seen when conducting the hypothesis test. On the subject of null hypotheses, it is important to always remember that although a hypothesis test always has a null hypothesis and an alternate hypothesis, the major difference between the two is that while an alternate hypothesis can take on many forms (for example, the alternate hypothesis in the example above could be that the average number of steps taken daily by a Toronto resident is less than, greater than, or simply not equal to 5000), the form of a null hypothesis never changes in that it always states that the parameter(s) of interest **is/are** a certain value (a good way to think of this in a mathematical sense is that a null hypothesis will always and only involve an equal sign). Given this fact, a foolproof way to word null hypotheses is to always have an equating word (i.e. is/are, equals, etc.) between the parameter of interest and the hypothesized value.

Disclaimer

Like a lot of statistical analysis and testing, there will always exist a variable amount of reliability or accuracy when it comes to interpreting results due to many factors such as assumptions about the population, faulty sampling, and others, and the results of a hypothesis test are no different which is why the language surrounding the interpretation of a p-value is a lot more cautious and ambiguous than expected as seen through the output of the function. That is, even when we decide that there is significant statistical evidence against the null hypothesis, we will only ever say we "fail to reject" the null hypothesis, while we will never say we "accept" it. To put this in a more understandable context, imagine that a hypothesis test is a murder trial and the prosecutors have asked for the death penalty where the null hypothesis is that the defendant is NOT guilty and the alternate hypothesis is that the defendant IS guilty. As we know, the justice system relies on being able to prove guilt beyond a reasonable doubt, and a p-value serves as a measure of how much the presented evidence crosses that threshold of reasonable doubt. As we also know, however, sometimes the evidence that is presented could be faulty or have key pieces missing, which directly impacts whether the defendant is declared guilty or not guilty and can lead to incorrect judgement to be handed down, which could result in either an innocent person getting the death penalty or the guilty person escaping accountability. With these factors in mind, it is easy to see why a defendant is only declared "guilty" or "not guilty", which is analogous to our language of "rejecting" or "failing to reject" the null hypothesis.

Task 4: Creating a reproducible example (reprex)

In this section, we are faced with the hypothetical problem of trying to find the individual group averages of a dataset that is stratified into said groups, only to have our code not run properly. To ask for help online to see why the code isn't working, we create what is called a "reproducible example" (reprex for short), defined as a remake of the faulty code that produces the same error that we are dealing with, which someone else can read and directly run in order to see exactly what error the code is causing. To produce this reprex, I used the tool that is built into R with the same code that was causing the error. I also had to consider whether or not to add the "library(tidyverse)" line, and ultimately decided in favour of doing so as someone who tries to run the reprex but doesn't have tidyverse installed will only see errors pertaining to functions not being defined and will thus be unable to help solve the error I am actually dealing with.

```
library(tidyverse)

my_data <- tibble(group = rep(1:10, each=10),
                  value = c(16, 18, 19, 15, 15, 23, 16, 8, 18, 18, 16, 17, 17,
                            16, 37, 23, 22, 13, 8, 35, 20, 19, 21, 18, 18, 18,
                            17, 14, 18, 22, 15, 27, 20, 15, 12, 18, 15, 24, 18,
                            21, 28, 22, 15, 18, 21, 18, 24, 21, 12, 20, 15, 21,
                            33, 15, 15, 22, 23, 27, 20, 23, 14, 20, 21, 19, 20,
                            18, 16, 8, 7, 23, 24, 30, 19, 21, 25, 15, 22, 12,
                            18, 18, 24, 23, 32, 22, 11, 24, 11, 23, 22, 26, 5,
                            16, 23, 26, 20, 25, 34, 27, 22, 28))
my_summary <- my_data %>%
  summarize(group_by = group, mean_val = mean(value))
glimpse(my_summary)
#> Rows: 100
#> Columns: 2
#> $ group_by <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3...
#> $ mean_val <dbl> 19.67, 19.67, 19.67, 19.67, 19.67, 19.67, 19.67, 19.67, 19.67...
```

Task 5: Simulating p-values

Setting up simulated data

```
# Set the seed to the last3digplus value created in task 1
set.seed(last3digplus)

# Create the gr object by assigning it a vector of 100000 entries, where each
# integer from 1 to 1000 is repeated 100 times. We will use this to generate
# the next 3 datasets
gr <- rep(1:1000, each = 100)

# Create the sim1 object by assigning it a dataset of 2 columns, where the val
# column contains observations drawn from an N(0, 1) distribution and the
# group column classifies the val values as belonging to a certain group
sim1 <- tibble(group = gr, val = rnorm(n = 100000, mean = 0, sd = 1))

# Create the sim2 object by assigning it a dataset of 2 columns, where the val
# column contains observations drawn from an N(0.2, 1) distribution and the
# group column classifies the val values as belonging to a certain group
sim2 <- tibble(group = gr, val = rnorm(n = 100000, mean = 0.2, sd = 1))

# Create the sim3 object by assigning it a dataset of 2 columns, where the val
# column contains observations drawn from an N(1, 1) distribution and the
# group column classifies the val values as belonging to a certain group
sim3 <- tibble(group = gr, val = rnorm(n = 100000, mean = 1, sd = 1))

# Create the all_sim object by combining sim1, sim2, and sim3 together, where
# a new column sim denotes which observations belong to which simulation
all_sim <- bind_rows(sim1, sim2, sim3, .id = "sim")

# Create the sim_description dataset to merge with improved simulation names
sim_description <- tibble(sim = 1:4, desc = c("N(0, 1)",
                                              "N(0.2, 1)",
                                              "N(1, 1)",
                                              "Pois(5)"))

# Update the all_sim object by merging it with sim_description in order
# to include a column desc that indicates from which distribution each
# value in the val column was drawn from
all_sim <- merge(x = all_sim, y = sim_description)
```

```r
# From top to bottom, left to right: pipe the all_sim data in, only take the
# first 3 groups of data, set x and y axes in ggplot, set plot as a histogram
# with 40 bins, create three rows of three histograms where each row corresponds
# to a different distribution and each column corresponds to a different group,
# set the theme of the plot, add a caption
all_sim %>%
  filter(group <= 3) %>%
  ggplot(aes(x = val)) +
  geom_histogram(bins = 40) +
  facet_wrap(desc~group, nrow = 3) +
  theme_minimal() +
  labs(caption = "Created by Xiaotang Zhou in STA303/1002, Winter 2022")
```

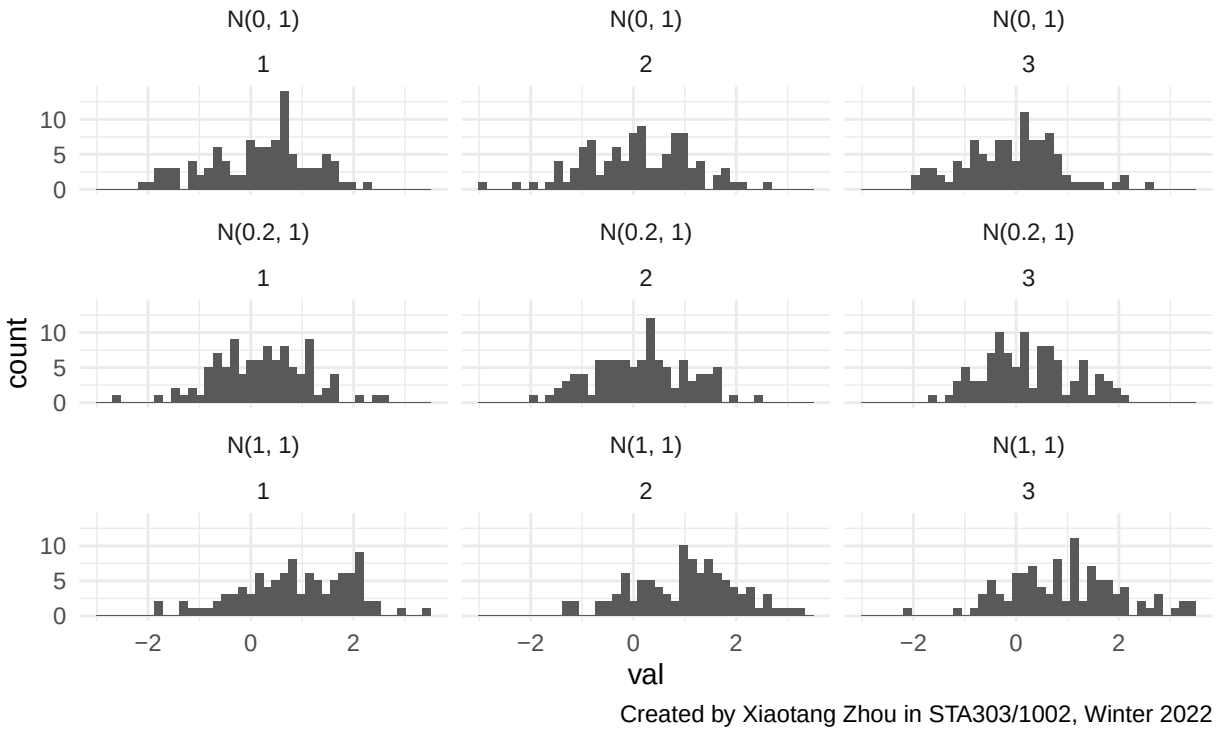

Figure 2: The various distributions for 3 randomly created groups of observations, where each row of histograms pertains to the visualization of 100 randomly sampled observations from the titled distribution

Calculating *p* values

```
# From top to bottom, left to right: save to pvals the following created object:
# pipe the all_sim data in, group the data first by the category describing the
# distribution from which the observation was drawn then by the group number,
# add a new column pval that contains the p-value that results from conducting a
# one-sample t-test under the null hypothesis that the population mean is equal
# to 0 for each group and each distribution
pvals <- all_sim %>%
  group_by(desc, group) %>%
  summarise(pval = t.test(val, mu = 0)$p.value, .groups = "drop")
```

```
# From top to bottom, left to right: pipe the pvals data in, set plot as a
# histogram with appropriate parameters, create a histogram for each distribution
# that was sampled from that shows the distribution of the p-values for each of
# the 100 groups of samples, set the x-axis to be bounded between x = 0 and
# x = 1, set the theme of the plot, add a caption
pvals %>%
  ggplot(aes(x = pval)) +
  geom_histogram(boundary=0, binwidth=0.05, fill = "grey", color = "black") +
  facet_wrap(~desc, scales = "free_y") +
  xlim(0, 1) +
  theme_minimal() +
  labs(caption = "Created by Xiaotang Zhou in STA303/1002, Winter 2022")
```

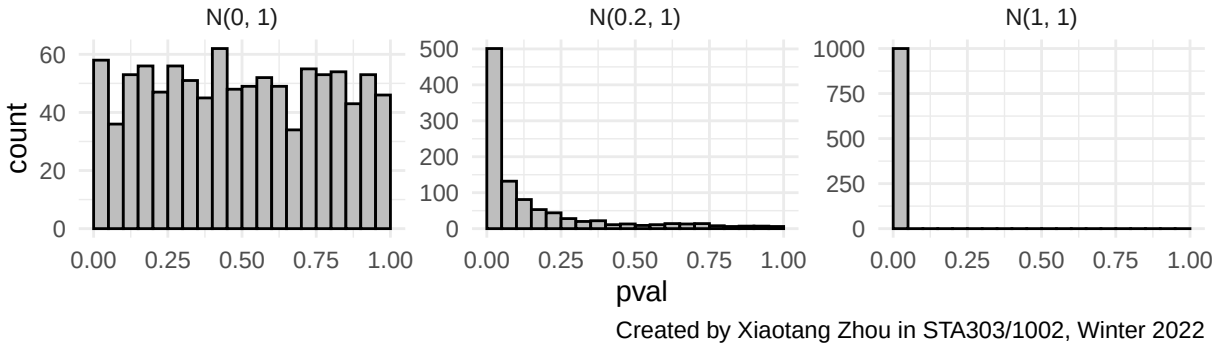

Figure 3: The distributions of the p-values for the one-sample t-tests (with the null hypothesis that the population mean is 0) run on each of the sampled groups of observations, where each histogram corresponds to a different distribution from which the observations were sampled

Drawing Q-Q plots

```
# From top to bottom, left to right: pipe the pvals data in, set plot as a
# QQ plot with appropriate axes and quantile distribution, embed a line with
# slope 1 and y-intercept 0 that will allow us to assess normality, create a QQ
# plot for each distribution that was sampled from that shows the distribution
# of the p-values for each of the 100 groups of samples, set the theme of the
# plot, add a caption
pvals %>%
  ggplot(aes(sample = pval)) +
  geom_qq(distribution = qunif) +
  geom_abline(intercept = 0, slope = 1) +
  facet_wrap(~desc) +
  theme_minimal() +
  labs(caption = "Created by Xiaotang Zhou in STA303/1002, Winter 2022")
```

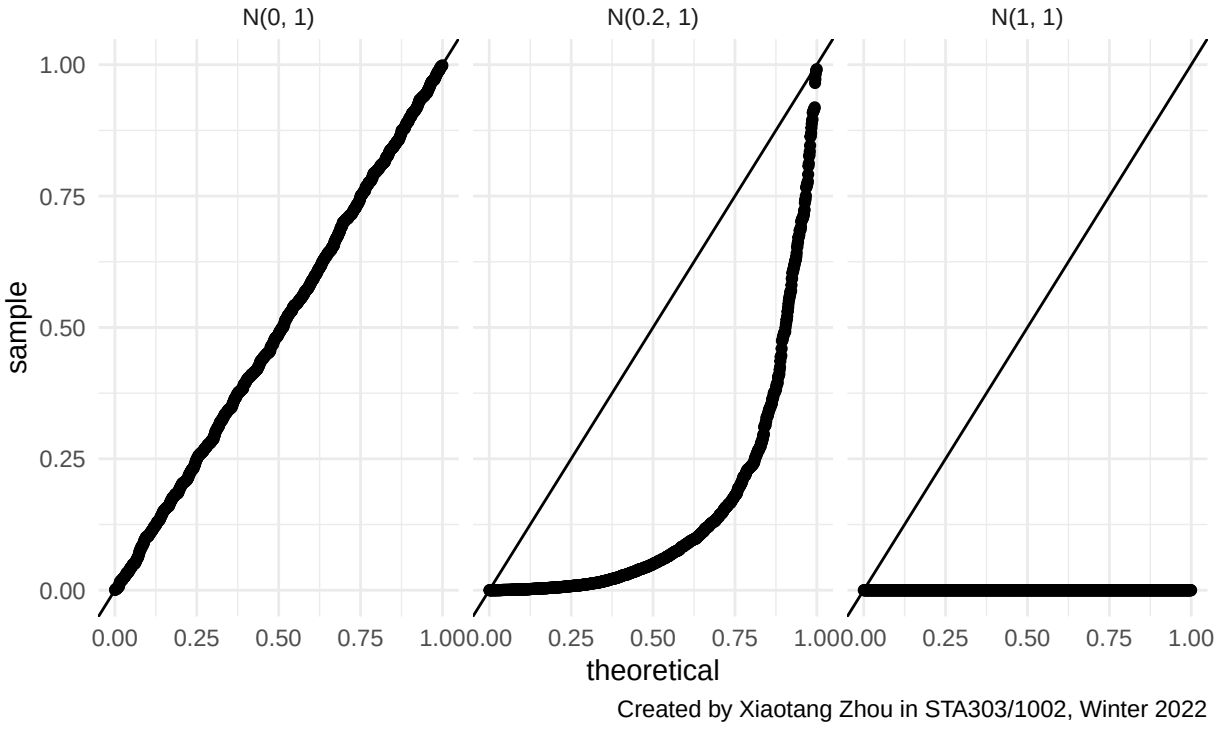

Figure 4: QQ plots that assess the uniformity of the distributions of the p-values for the one-sample t-tests (with the null hypothesis that the population mean is 0) run on each of the sampled groups of observations, where each plot corresponds to a different distribution from which the observations were sampled

Conclusion and summary

In this section, we are given the task of visualizing the behaviour of p-values that are derived from hypothesis tests run on samples pulled from various Normal distributions (with different means but constant variance) with the null hypothesis that the population mean is equal to 0. As seen in the histograms that we drew for each of the three distributions, when the null hypothesis is actually true (i.e. when the population mean is actually 0) the distribution of the p-values seems to be uniform with no clear peak or skew, while when the null hypothesis is false the distribution of the p-values is highly skewed to the right.

This is expected and connects to the definition of a p-value in that when the null hypothesis is actually true, we are much more likely to find little to no statistical evidence against the null hypothesis by drawing samples from that population, which will cause the p-value to increase and take larger values because the probability of finding data at least as extreme as the data we found under the assumption that the null hypothesis is true, which is the exact definition of a p-value, is much greater in this scenario. This also relates to the 16th question on the pre-knowledge check in that when the null hypothesis is actually correct, the p-values will tend toward a roughly uniform distribution, so it follows that around 10% of them will be between 0.9 and 1, 10% of them will be between 0.8 and 0.9, and so on.

On the other hand, when the population mean is not 0 and the null hypothesis is false, we are more likely to find statistically significant evidence against the null hypothesis since the probability of finding data as extreme as we did under the assumption that the null hypothesis is true becomes smaller and smaller the further away the actual population mean is from 0. We can see this very clearly on the histograms corresponding to the samples drawn from $N(0.2, 1)$ and $N(1, 1)$, as the distribution corresponding to the samples drawn from $N(0.2, 1)$ still contain some roughly visible bumps to the right of the towering bin on the far left, while the distribution corresponding to the samples drawn from $N(1, 1)$ contain almost no bins aside from the towering bin on the far left.

Writing sample

In "Common misconceptions about data analysis and statistics" by Harvey J. Motulsky, five of the many fallacies committed by scientists on the topic of reproducibility of results in research are discussed, with a focus on those involving hypothesis testing and p-values. From a personal perspective, the idea of accidentally introducing bias by increasing the sample size only when the results are not statistically significant or otherwise desirable struck me as the most impactful, as it shines a light on flaws that I believe should be prioritized when considering what elements in the world of reproducibility need to be fixed or changed the most.

This idea that bias is introduced into the research when the sample size is increased after finding undesirable results struck me first as an incorrect assertion, but quickly began to make more sense when exploring the dangers of what Motulsky terms ad hoc sample size selection. This is grouped with the many possible ways of p-hacking, defined as essentially changing the settings of the research, such as the model being used, the stratification of the sample, and many other possible factors, until a statistically significant or otherwise desired result is achieved (Motulsky, 2014). Based on my own initial reaction to this idea, I feel that the unintended consequences of ad hoc sample size selection lie in the unintuitive nature of claiming that increasing sample size can be a bad thing. As statisticians, we are so conditioned to believe that a large sample size is never a bad thing that our knee-jerk reaction to a "bad" result is to say we didn't sample enough from the population and thus should do so, while not realizing that only doing this when faced with a "bad" result taints the research. This is the main reason I believe this idea should be prioritized when considering what changes need to be made to fix the growing problem of reproducibility, as in my opinion, this fallacy is so closely related to such a popular belief among statisticians that stopping the spread of this fallacy may require a complete restructuring of how much emphasis is placed on larger sample sizes being a catchall that can solve any problem.

All in all, after reading Motulsky's article, I have come to the conclusion that introducing bias by increasing sample size after obtaining an undesirable result is one of the most pressing issues that fuels the reproducibility crisis. I feel this is the case because this fallacy is hidden behind the commonly held belief by statisticians that a large sample size is a good thing no matter what, and that to stop the spread of this fallacy could entail a complete overhaul of how large sample sizes are viewed by statisticians, researchers, and scientists alike.

Word count: 458 words

References:

Motulsky, H. J. (2014). Common misconceptions about data analysis and statistics. *Naunyn-Schmiedeberg's Archives of Pharmacology*, *387*(11), 1017–1023. https://doi.org/10.1007/s00210-014-1037-6

Reflection

What is something specific that I am proud of in this portfolio?

One thing that I am proud of in this portfolio is my newfound ability to more quickly identify the exact strategy that I will be required to use prior to starting a task in order to complete it quickly. I feel that this is an extension and even a direct consequence of what I was proud of when working on the mini-portfolio, in that while creating the mini-portfolio I was able to improve my ability to write clean and readable code, and by applying those skill and experience I was able to spend more time closely analyzing the problem and deciding what strategy, such as what R function to use, I would need to properly complete the task instead of constantly worrying about how clean or readable the code would look.

How might I apply what I've learned and demonstrated in this portfolio in future work and study, after STA303/1002?

There are many ways in which I feel I can apply what I've learned and demonstrated while working on this portfolio in future work and study, but the most important skill that I have learned in my opinion is how to create reprexes. I believe the skill of being able to create reprexes that someone else can read, run, and instantly understand is something that will help immensely in the future, as reproducibility of code will be essential in allowing others to edit, debug, and suggest changes to it, which is critical when asking for help from people who are not involved in the project and don't have the necessary background information to be able to know right away where the problem might be.

What is something I'd do differently next time?

If I could do this portfolio all over again, one thing I would do differently is to fully analyze a task and make sure I have the right tools to complete it before ever starting it. Learning from my reflection that I wrote for my mini-portfolio, while working on this portfolio I committed to focusing on one task at a time in order to not cause any unnecessary confusion by working on multiple tasks at the same time. This was a good strategy until I realized that I was missing the critical step of reading over the entire task completely before attempting it. In other words, I found that although I had committed to finishing a single task before moving to the next, by not fully reading the instructions for every part before starting the task, I was still wasting a lot of time by having to stop and clarify over and over again what I needed to do next while working on the sub-parts of a task.