STA303/1002 Portfolio

An exploration of linear mixed models and common misconceptions in statistics

Ruizhe Huang

2022-02-03

Contents

List of Figures

Introduction

STA303/1002 is a course about how to wrangle and explore a dataset; create appropriate data visualizations; describe ethical considerations in data analysis; understand the assumptions and appropriate use cases for linear mixed models, generalized linear models, generalized linear mixed models, and generalized additive models; write and execute R code for the model types covered in this course; accurately and appropriately interpret the results of the model types.
In this portfolio, I will first perform the statistical skills such as setting up libraries and seed value; exploring sources of variance in a balanced experimental design (teaching and learning world); applying linear mixed models for the strawberry data (practical world); building a confidence interval interpreter; building a p-value interpreter; user instructions and disclaimer; creating a reproducible example (reprex); simulating p values. Secondly, paraphrase and comment the article *Common misconceptions about data analysis and statistics* with the goal of giving advice to future self. In the end, I will include a reflection section to discuss something specific I'm proud of in this portfolio; apply what I've learned and demonstrated in this portfolio in future work and study, after STA303/1002; something I'd do differently next time.

Statistical skills sample

Task 1: Setting up libraries and seed value

```r
# load the tidyverse library
library(tidyverse)
# 100 + the last three digits of student ID number.
last3digplus <- 100 + 331
```

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

Growinng your (grandmother's) strawberry patch

```r
# Sourcing it makes a function available
source("grow_my_strawberries.R")

# Load grow_my_strawberries() function
my_patch <- grow_my_strawberries(seed = last3digplus)
# Alter the the levels
my_patch <- my_patch %>%
  mutate(treatment = fct_relevel(treatment, "No netting", after = 0))
```

Plotting the strawberry patch

```r
# Create a plot where the x-xis has each plot (alphabetical) and the y-axis represents
↪  yield.
my_patch %>%
  ggplot(aes(x = patch, y = yield, fill = treatment, color = treatment)) +
  geom_point(pch = 25) +
  scale_fill_manual(values = c("#78BC61", "#E03400", "#520048")) +
  scale_color_manual(values = c("#78BC61", "#E03400", "#520048")) +
  theme_minimal() +
  labs(caption = "Created by Ruizhe Huang in STA303/1002, Winter 2022")
```

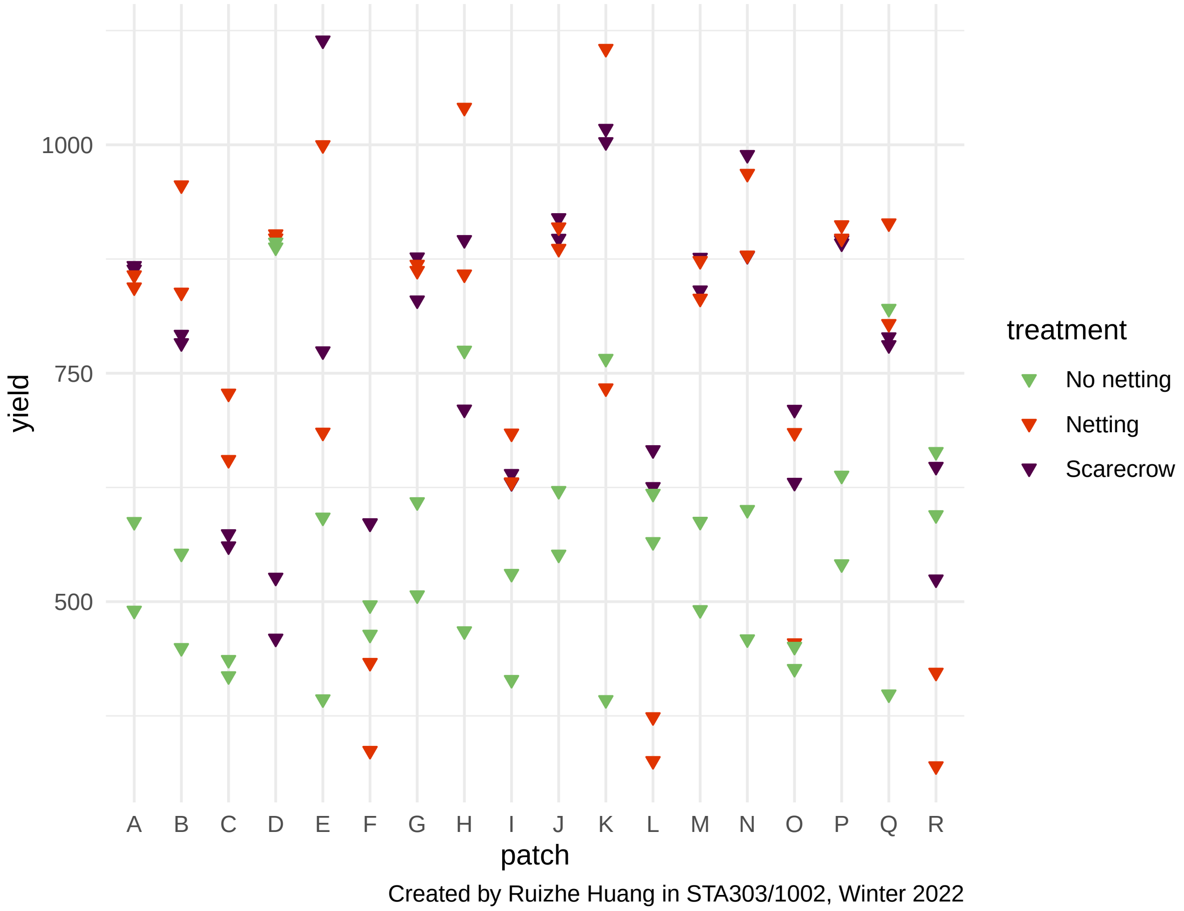


Figure 1: Plotting the strawberry patch for each treatment

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

Model formula

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

where:

- y_{ijk} is the amount of strawberry yields (in kgs) in k^{th} harvest with treatment i at patch j.
- μ is the grand mean of strawberry yields (in kgs).
- α_i are the $i = 1, 2, 3$ fixed effects for treatment.
- b_j are the random effects for patch $j = 1, 2, 3, \ldots, 18$.

- $(\alpha b)_{ij}$ are the $3 \times 18 = 54$ interaction terms for the interaction between the the treatment and the patch.

- ϵ_{ijk} is the error term.
- $(\alpha b)_{ij} \sim N(0, \sigma^2_{\alpha b}), b_k \sim N(0, \sigma^2_b)$ and $\epsilon_{ijk} \sim N(0, \sigma^2)$.
  All the random effects are mutually independent random variables.

```
# interaction model including main effects
int_mod <- lm(yield ~ patch * treatment, data = my_patch)
# residual variance after fitting the version of this linear model with this most
↪  parameters
var_int <- summary(int_mod)$sigma^2

# group by both patch and treatment and then summarize to create a new variable
agg_int <- my_patch %>%
  group_by(patch, treatment) %>%
  summarise(yield_avg_int = mean(yield), .groups = "drop")
# aggregated model with data aggregated across both patch and treatment
agg_mod <- lm(yield_avg_int ~ patch + treatment, data = agg_int)
# variance in yield explained by the interaction between patch and treatment, after
↪  accounting
var_ab <- summary(agg_mod)$sigma^2 - var_int / 2

# group by patch only
agg_patch <- my_patch %>%
  group_by(patch) %>%
  summarise(yield_avg_patch = mean(yield), .groups = "drop")
# intercept only model
patch_mod <- lm(yield_avg_patch ~ 1, data = agg_patch)
# variance in average yield patch-to-patch
var_patch <- summary(patch_mod)$sigma^2 - (summary(agg_mod)$sigma^2)/3
```

```
#  Create a table
tibble(`Source of variation` = c("treatment:patch",
                                 "patch",
                                 "residual"),
       Variance = c(var_ab, var_patch, var_int),
       Proportion = 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 = "proportion of variance in yield explained by the sources")
```

Table 1: proportion of variance in yield explained by the sources

Source of variation	Variance	Proportion
treatment:patch	12823.942	0.42
patch	7537.211	0.24
residual	10494.861	0.34

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

```r
# load the necessary library
library(lme4)
# Fit 3 models
mod0 <- lm(yield ~ treatment, data = my_patch)
mod1 <- lmer(yield ~ treatment + (1|patch), data = my_patch)
mod2 <- lmer(yield ~ treatment + (1|patch) + (1|treatment:patch), data = my_patch)
# Set up likelihood ratio test between these models
lmtest::lrtest(mod1, mod2)
```

```
## Likelihood ratio test
##
## Model 1: yield ~ treatment + (1 | patch)
## Model 2: yield ~ treatment + (1 | patch) + (1 | treatment:patch)
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   5 -687.10
## 2   6 -678.87  1 16.463   4.96e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The main difference between the maximum likelihood (ML) and restricted maximum likelihood (REML) is how they estimate the variance parameters. **We are using restricted maximum likelihood(REML) here**. Since we prefer REML when there are many parameters or the main goal is estimates of our model parameters (random and fixed). Moreover, since the fixed effect for these models is the same, we only compare the difference of the random effect. Due to these reasons, I chose the REML.

```
summary(mod2)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: yield ~ treatment + (1 | patch) + (1 | treatment:patch)
##    Data: my_patch
##
## REML criterion at convergence: 1357.7
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.00553 -0.39289  0.09994  0.35429  2.11426
##
## Random effects:
##  Groups          Name        Variance Std.Dev.
##  treatment:patch (Intercept) 12824    113.24
##  patch           (Intercept)  7537     86.82
##  Residual                    10495    102.44
## Number of obs: 108, groups:  treatment:patch, 54; patch, 18
##
## Fixed effects:
##                    Estimate Std. Error t value
## (Intercept)          559.79      37.72  14.841
## treatmentNetting     200.60      44.81   4.477
## treatmentScarecrow   208.41      44.81   4.651
##
## Correlation of Fixed Effects:
##             (Intr) trtmnN
## trtmntNttng -0.594
## trtmntScrcr -0.594  0.500
```

Justification and interpreation

H_0 : the model 1 (simpler model) is as good as model 2 (complex model)
H_A : the model 1 (simpler model) is not as good as model 2 (complex model)
Since we get a small p-value, so we have very strong evidence against the null hypothesis that the model 1 (simper model) with no treatment/patch interaction is as good as model 2 (complex model). Therefore, **model 2 is the most appropriate final model**. The intercept of the

final model: 559.79 kgs is the average strawberry yield for no netting. Netting has a higher average strawberry yield than no netting by an average of 200.60 kgs. Scarecrow has a higher average strawberry yield than no netting by an average of 208.41 kgs.
There is a 42% variance explained by the interaction term (treatment:patch) in the model. There is a 24% variance explained by patch in the model. There is 34% variance that is explained by residual in the model.

Task 3a: Building a confidence interval interpreter

```r
interpret_ci <- function(lower, upper, ci_level, stat){
  if(!is.character(stat)) {
    # produce a warning if the statement of the parameter isn't a character string
    warning("
    Warning: stat should be a character string that describes the statistics of
↪   interest.")
  } else if(!is.numeric(lower)) {
    # produce a warning if lower isn't numeric
    warning("
    Warning: lower should be a numeric number that describes the lower bound of the
↪   confidence interval.")
  } else if(!is.numeric(upper)) {
    # produce a warning if upper isn't numeric
    warning("
    Warning: upper should be a numeric number that describes the upper bound of the
↪   confidence interval.")
  } else if(!is.numeric(ci_level) | ci_level < 0 | ci_level > 100) {
    # produce a warning if ci_level isn't appropriate
    warning("
    Warning: ci_level should be a numeric number between 0 and 100 that describes the
↪   confidence level this interval was calculated")
  } else{
    # print interpretation
  str_c("We are ", ci_level, "% confident that the population ", stat, " is between ",
  ↪  lower, " and ", upper, ".")

  }
}

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

```r
# 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 population mean number of shoes owned by students is between 10 and 20.

CI function test 2: Warning: ci_level should be a numeric number between 0 and 100 that describes the confidence level this interval was calculated

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

Task 3b: Building a p value interpreter

```r
interpret_pval <- function(pval, nullhyp){
  if(!is.character(nullhyp)) {
    # produce a warning if the null hypothesis isn't a character string
    warning("
            Warning: nullhyp should be a character string that describes the null
↪   hypothesis.")
  } else if(!is.numeric(pval)) {
    # produce a warning if p value isn't numeric
    warning("Warning: You p value should be a number.")
  }  else if(pval > 1) {
    # produce a warning if p value is greater than 1
    warning("
            Warning: p value should be a numeric number that is less or equal to 1.")
  } else if(pval < 0){
    # produce a warning if p value is smaller than 0
    warning("
            Warning: p value should be a numeric number that is non-negative.")
  } else if(pval > 0.1){
    # print interpretation
    str_c("The p value is ", round(pval, 3),
                ", no evidence against the null hyothesis that ", nullhyp, ".")
  } else if(pval > 0.05){
    # print interpretation
    str_c("The p value is ", round(pval, 3),
                ", weak evidence against the null hyothesis that ", nullhyp, ".")
  } else if(pval > 0.01){
    # print interpretation
    str_c("The p value is ", round(pval, 3),
                ", moderate evidence against the null hyothesis that ", nullhyp, ".")
  } else if(pval > 0.001){
    # print interpretation
    str_c("The p value is ", round(pval, 3),
                ", strong evidence against the null hyothesis that ", nullhyp, ".")
  } else {
    # print interpretation
    str_c("The p value is <.001, very strong evidence against the null hyothesis 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: The p value is <.001, very strong evidence against the null hyothesis that the mean grade for statistics students is the same as for non-stats students.

p value function test 2: The p value is 0.05, moderate evidence against the null hyothesis that the mean grade for statistics students is the same as for non-stats students.

p value function test 3: The p value is 0.05, weak evidence against the null hyothesis 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

The confidence interval is used to give the likely range of the population parameter. If the confidence interval does not catch the null hypothesis, the results are statistically significant. Some common misconceptions in interpreting frequentist confidence intervals include: there is a certain probability that the population parameter will fall between the confidence interval. This is not true. The population parameter is a constant, not a random variable. The probability of a constant falling within any given range is always 0% or 100%.
The p-value is under the assumption that the null hypothesis is true, a result that is at least as extreme as the observed result of the hypothesis test is obtained. If the p-value is larger than the pre-specified alpha level(commonly 0.05), we have weak/no evidence against the null hypothesis. If the p-value is smaller than the alpha level, we have moderate/strong evidence against the null hypothesis.
The population parameters are the value that describes the entire population, which is unknown and constant. For example, the population mean is a population parameter. When wording the null hypothesis, we should mention which population parameter we hypothesize. For example, we want to investigate whether the population mean grade for statistics students is the same as for non-stats students. Then, the null hypothesis could be that there is no difference between the population mean grade for statistics students and non-stats students.

Disclaimer

The p-value should be a number between 0 and 1, and the null hypothesis should be a character string that describes the null hypothesis. The smaller the p-value, the stronger the evidence against the null hypothesis. We should never make claims in favor/against the alternative hypothesis. The statistical claims are always about the null. A common threshold for rejecting or failing to reject the null hypothesis is 0.05. This is mostly from habit/convention. Many statisticians—especially in light of the reproducibility crisis and poor public, and even sometimes researcher, understanding of p values—prefer to make statements about the strength of evidence, not just reject/fail to reject.

p value range	Strength comment
> 0.1	No evidence against the null
$0.05 <$ p value < 0.1	Weak evidence against the null hypothesis
$0.01 <$ p value < 0.05	Moderate/some evidence against the null hypothesis

p value range	Strength comment
$0.001 <$ p value < 0.01	Strong evidence against the null hypothesis
< 0.001	Very strong evidence against the null hypothesis

In addition, if given the null hypothesis is true, but we conclude that reject the null, then it is a type I error. If given the null hypothesis is false, but we conclude that in favour the null, then it is a type II error.

Task 4: Creating a reproducible example (reprex)

reprex is a reproducible example, which is an example that someone else can reproduce and help replicate bugs or other behaviors that we want to show them. We need the `reprex`, an R package, and `reprex::reprex()`, an R function in reprex to produce the reprex.

```
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
set.seed(last3digplus)
# Create 3 simulated data sets
sim1 <- tibble(group = rep(1:1000, each = 100),
               val = rnorm(100000, 0, 1))
sim2 <- tibble(group = rep(1:1000, each = 100),
               val = rnorm(100000, 0.2, 1))
sim3 <- tibble(group = rep(1:1000, each = 100),
               val = rnorm(100000, 1, 1))
# Stack all datasets into one new dataset
all_sim <- bind_rows(sim1, sim2, sim3, .id = "sim")
# Create 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)"))
# all_sim join on the dataset sim_description
all_sim <- all_sim %>%
  mutate(sim = as.numeric(sim)) %>%
  left_join(sim_description, by = "sim")
```

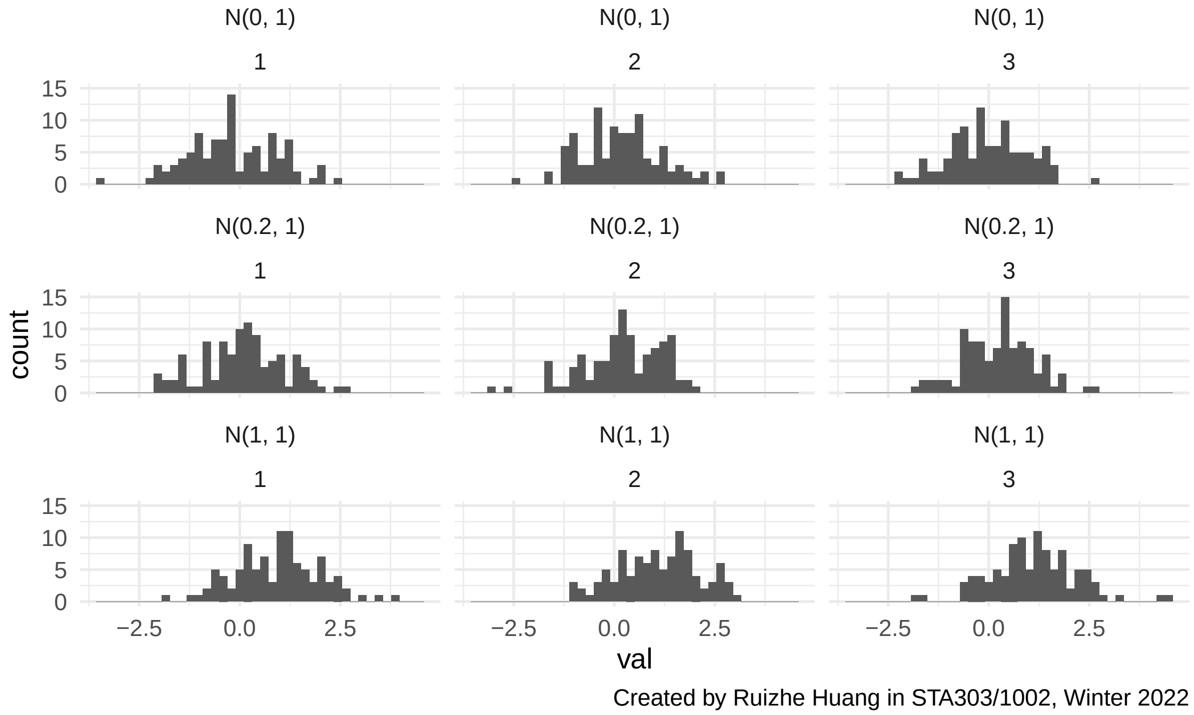


Figure 2: Histograms for the first three groups for each simulated dataset

Calculating *p* values

```
# Create a new dataset called pvals
pvals <- all_sim %>% group_by(desc, group) %>%
  summarize(pval = t.test(val, mu = 0)$p.value, .groups = "drop")
```

```
# Plot histograms of the p values, faceted by desc
pvals %>% ggplot(aes(x = pval)) +
  geom_histogram(boundary = 0, binwidth = 0.05, fill = "grey", color = "black") +
  xlim(0, 1) +
  facet_wrap(~desc, scales = "free_y") +
  theme_minimal() +
  labs(caption = "Created by Ruizhe Huang in STA303/1002, Winter 2022")
```

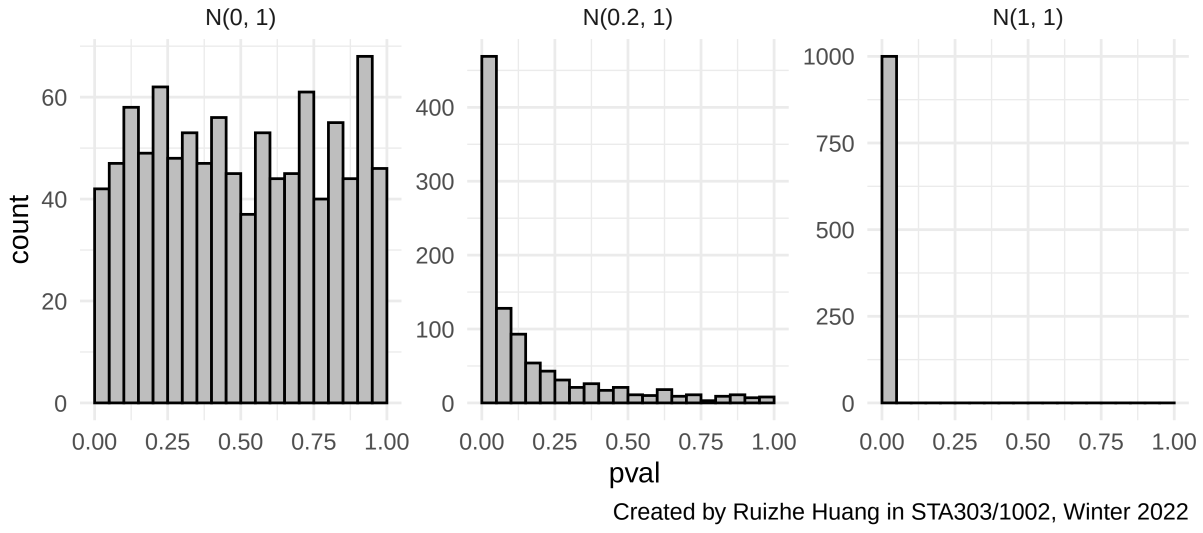
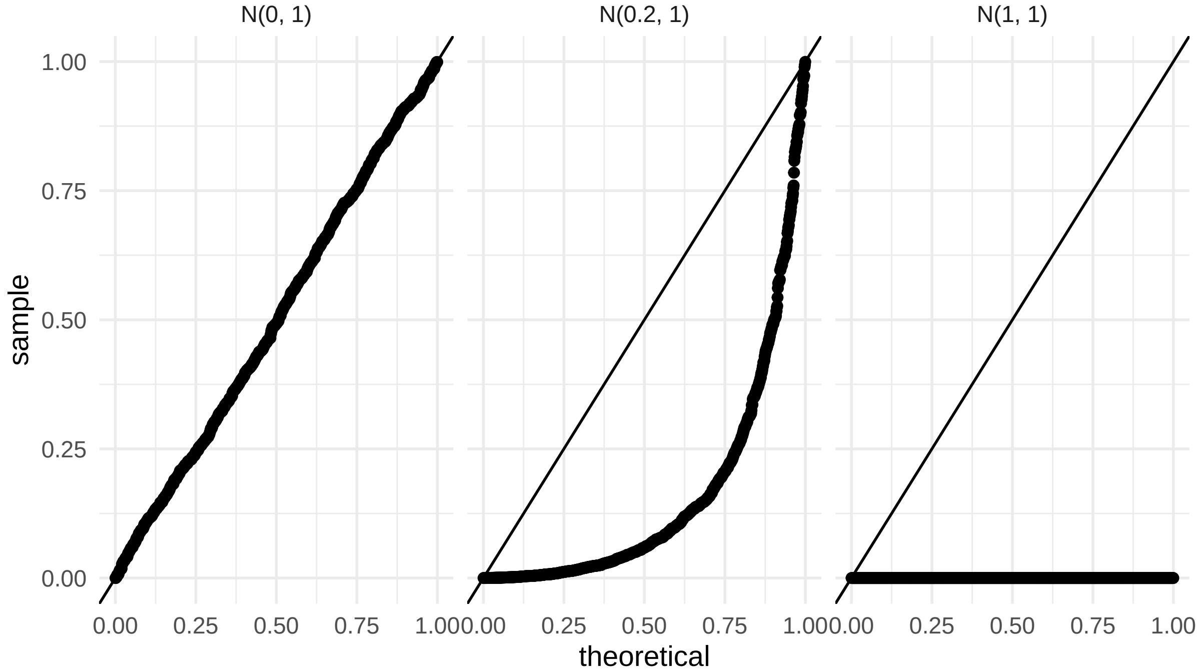


Figure 3: Histograms of the p values

Drawing Q-Q plots

```r
# Create a figure with QQ plots for each simulation
pvals %>%
  ggplot(aes(sample = pval)) +
  geom_qq(distribution = qunif) +
  geom_abline(intercept = 0, slope = 1) +
  facet_wrap(~desc) +
  theme_minimal() +
  labs(caption = "Created by Ruizhe Huang in STA303/1002, Winter 2022")
```



Created by Ruizhe Huang in STA303/1002, Winter 2022

Figure 4: QQ plots for each simulation

Conclusion and summary

Since the p-value assumes that the null hypothesis is true, a result that is at least as extreme as the observed result of the hypothesis test is obtained. The first histogram is the p-values assuming that the null hypothesis $\mu = 0$ of $N(0, 1)$ is true.
From the uniform Q-Q plot, most points fall on the qq line. Given the null hypothesis $H_0 : \mu = 0$ is true, the distribution of p values follows $Uniform(0, 1)$. Therefore, question 16 from the pre-knowledge check should be approximately 10% of p values will be between 0.9 and 1.

Writing sample

Introduction

Misconceptions in researchers' understanding of statistical concepts resulted in the reproducibility of many published results were questioned. Especially, p-hacking is a common mistake the researchers may make. Therefore, the following is the advice that you should not forget.

P-hacking is not OK (Motulsky, 2014).

Based on the statistical results, you cannot infer if you do not follow the planned experiment design. For example, if you have a result that is not statistically significant as you followed the experiment design, however, because you favor the result, you try a different method such as collecting more data to reanalyze the data. Alternatively, other approaches until you get a statistically significant result. The problem is that if you only try a different approach or collect more data when the p-value is not statistically significant, you won't know the initial p-value that is significant may turn to not significant after using a different approach or collecting more data.

Moreover, if you did not choose the sample size at the beginning, keep going until you favor the results. It is called Ad hoc sample size selection, an example of p-hacking, leading to a misleading result. Another example of p-hacking is that make hypothesis after the result is known. You use different approaches to analyze the data and find out a relationship, then hypothesize based on the relationship as it is supposed to be stated before collecting data.

I suggest that you do not forget to claim the planned experiment design, chosen sample size, and the hypothesis at the very beginning of the analysis. Also, the conclusion should be preliminary if you use p-hacking, which is not limited to the above.

Conclusion

Some of the published results that were not reproducible may result from p-hacking, making adjustments to the original experiment design to get favorable or significant results. All in all, do not forget to document and report every step you take in the analysis, including methodology and result (Motulsky, 2014).

Word count: 332 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?

I am proud to give instructions to interpret confidence intervals and p values. There was something about interpreting the p values that I did not know before. For example, I do not know we can comment on the strength of evidence, not only reject/fail to reject. Moreover, I know that we should not make claims in favor/against the alternative hypothesis. In addition, I am proud to summarize the main point of the article *Common misconceptions about data analysis and statistics* by Motulsky and give useful advice to future myself. I am also proud of this reflection part; I can make reflections about the entire portfolio thoroughly.

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

I learned that we could not claim that there is a certain probability that the population parameter will fall between the confidence interval since the population parameter is a constant, not a random variable. The probability of a constant falling within any given range is always 0% or 100%. In my future work and study life, I will remember how to appropriately interpret the confidence interval results. Moreover, I learned p hacking is not ok, and I should document and report every step I take in the analysis, including methodology and result in the future. In this portfolio, I also demonstrated writing skills critical for future work and study.

What is something I'd do differently next time?

I will plan ahead and start writing the portfolio earlier next time. I will try to understand the material more in-depth and summarize more essential points of the article to give more advice to future myself. Moreover, I will collect more possible pitfalls of the concept or interpretation of the confidence interval and p-value. Next time, I will be more creative and try my own cover page style to make it more aesthetically pleasing. In addition, I will try to write the R function of task 3 without the template. I also want to neatly format my code with `styler` package.