Non-Parametric Tests Materials

Brier Gallihugh, M.S.

2024-02-23

Table of contents

```r
set.seed(10311993)
library(tidyverse)
library(psych)
library(palmerpenguins)

data <- palmerpenguins::penguins %>% dplyr::select(island,body_mass_g) %>%  ①
  dplyr::filter(island == "Biscoe" | island == "Dream") %>% na.omit()       ②

data <- droplevels(data)                                                    ③
```

① Take the `palmerpenguins` data set and use the `select` function to isolate the island and body_mass_g variables

② Use the `filter()` function to select for only observations where the island variable is "Biscoe" or "Dream" and omit any missing data after using the `na.omit()` function.

③ Use the `droplevels()` function to ensure factor levels no longer present are excluded from analysis.

Independent Sample T Test (i.e., Wilcoxon's Rank-Sum Test)[1]

Tip

1. This is computationally the same as the Mann-Whitley test (Field et al, 2012)

Caution

Non-parametric tests are NOT less powerful than their parametric counterpart as long as the distribution of interest is not normally distributed (Field et al, 2012)

```
# Two-Tailed (one-tailed "greater" diff is + OR "lesser" diff is -)
# Correction for p value (TRUE by default, but can change to FALSE correction = FALSE)
ind_t_nonpar <- wilcox.test(body_mass_g ~ island, data = data)                  ①
ind_t_nonpar                                                                      ②

# Effect Size (from Field et al, 2012)

rFromWilcox<-function(wilcoxModel, N)                                             ③
{
z<- qnorm(wilcoxModel$p.value/2)
r<- z/ sqrt(N)
return(r)
}

rFromWilcox(ind_t_nonpar,291)                                                     ④

# Typical reporting for non-parametric involves median values
descriptive_stats <- data %>%                                                     ⑤
  group_by(island) %>%
  summarize(group_size = n(),                                                     ⑥
            median = median(body_mass_g),
            mean = round(mean(body_mass_g, na.rm = TRUE),2),
            sd = round(sd(body_mass_g, na.rm = TRUE),2))

print(descriptive_stats)                                                          ⑦
```

① Initiate a non parametric independent samples t test using the `wilcox.test()` function.
② Show the output of the independent sample non-parametric t test
③ Create a pseudo effect size measure for non parametric tests courtesy of Field et al (2012).
④ Generate an effect size measure r using the function defined above. The function requires you provide the wilcox.test object and the sample size

⑤ Generate some simple descriptive statistics for reporting purposes grouped by island affiliation
⑥ Several statistics included such as sample size, mean, median, and standard deviation.
⑦ Show the output of the above calculations using the `print()` function.

```
    Wilcoxon rank sum test with continuity correction

data:  body_mass_g by island
W = 17741, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0

[1] -0.610081
# A tibble: 2 x 5
  island group_size median  mean    sd
  <fct>       <int>  <dbl> <dbl> <dbl>
1 Biscoe        167  4775  4716.  783.
2 Dream         124  3688. 3713.  417.
```

Dependent Sample T Test (i.e., Wilcoxon Signed-Rank Test)

```
data <- psych::sat.act %>%                                                    ①
  mutate(condition = rep(c(1,2),350),
         ACT_2 = if_else(age < 39, ACT + 3, ACT -1))

data_condition1 <- data %>%                                                   ②
  filter(condition == 1)

data_condition2 <- data %>%                                                   ③
  filter(condition == 2)

condition1_test <- wilcox.test(data_condition1$ACT,                           ④
                               data_condition1$ACT_2,
                               paired = TRUE,
                               correct = FALSE)

condition2_test <- wilcox.test(data_condition2$ACT,                           ⑤
                               data_condition2$ACT_2,
                               paired = TRUE,
                               correct = FALSE)
```

```
print(condition1_test)                                          ⑥
print(condition2_test)

# Effect Size
rFromWilcox(condition1_test,350)                                ⑦
rFromWilcox(condition2_test,350)
```

① Create a dependent samples t test data set
② Filter data by condition variable (`condition = 1`) using the `filter()` function
③ Filter data by condition variable (`condition = 2`) using the `filter()` function
④ Run a `wilcox.test()` function for `condition = 1`. You want to include the time1 and time2 measures. The `paired = TRUE` tells R that this is a dependent sample t test
⑤ Do the same for the `condition = 2` condition.
⑥ The `print()` function will give you the results for the non parametric dependent sample t test
⑦ Use the function shown earlier by Fields et al (2012) to calculate an effect size. It takes the same arguments as before.

```

	Wilcoxon signed rank test

data:  data_condition1$ACT and data_condition1$ACT_2
V = 666, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0


	Wilcoxon signed rank test

data:  data_condition2$ACT and data_condition2$ACT_2
V = 666, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0

[1] -0.9363034
[1] -0.9363034
```

One Way ANOVA (i.e., Kruskal–Wallis Test)

```
data2 <- psych::sat.act[1:600,] %>%                             ①
  mutate(condition = as.factor(rep(c(1:3),200)),
         rank = rank(ACT))
```

```
# Condition 3 is control

kruskal.test(ACT ~ condition, data = data2)                                      ②

pgirmess::kruskalmc(ACT ~ condition, data = data2)                               ③

# Maybe we want to see the ranked means

by(data2$rank,data2$condition,mean)                                              ④

# Maybe we just want to compare each exp group to the control (Lowers Odds of Type II error)
# We can do that. But we need it to be the first level factor

data2$condition <- relevel(data2$condition,3)                                    ⑤

levels(data2$condition)                                                          ⑥

pgirmess::kruskalmc(ACT ~ condition, data = data2, cont = "two-tailed")          ⑦
```

① Create a dummy data set
② Run a non parametric ANOVA using the `kruskal.test()` function
③ Run a posthoc multiple comparisons test using the `kruskalmc()` function
④ Show "means" using the `by()` function. It takes the rank and condition variables as well as the `mean()` function
⑤ Sort levels so that the main comparison condition is first using the `relevel()` function. It takes your condition variable as well as the factor you want first (in this case 3).
⑥ The 'levels() function confirms that our ordering is correct (i.e., 3,1,2)
⑦ Use the `kuskalmc()` function with the `cont` argument set to "two-tailed" for planned comparison

```

	Kruskal-Wallis rank sum test

data:  ACT by condition
Kruskal-Wallis chi-squared = 2.4489, df = 2, p-value = 0.2939

Multiple comparison test after Kruskal-Wallis
alpha: 0.05
Comparisons
    obs.dif critical.dif stat.signif
1-2 20.7875     41.49949       FALSE
1-3 25.4050     41.49949       FALSE
2-3  4.6175     41.49949       FALSE
```

```
data2$condition: 1
[1] 285.1025
------------------------------------------------------------
data2$condition: 2
[1] 305.89
------------------------------------------------------------
data2$condition: 3
[1] 310.5075
[1] "3" "1" "2"
Multiple comparison test after Kruskal-Wallis, treatments vs control (two-tailed)
alpha: 0.05
Comparisons
    obs.dif critical.dif stat.signif
3-1 25.4050     38.85457       FALSE
3-2  4.6175     38.85457       FALSE
```

Repeated Measures ANOVA (i.e., Friedman's ANOVA)

```
data3 <- data2 %>%                                                          ①
  mutate(ACT_2 = if_else(age < 39, ACT + 3, ACT -1)) %>% select(ACT,ACT_2)

# Test
friedman.test(as.matrix(data3))                                             ②

# Post Hoc
pgirmess::friedmanmc(as.matrix(data3))                                      ③
```

① Create a repeated measures ANOVA dummy data set

② Run a repeated measures non parametric ANOVA using the `friedman.test()` function. Note that the `as.matrix()` function is required for the test to run properly.

③ Run a posthoc test using the `friedmanmc()` function.

```

    Friedman rank sum test

data:  as.matrix(data3)
Friedman chi-squared = 377.63, df = 1, p-value < 2.2e-16

Multiple comparisons between groups after Friedman test
alpha: 0.05
Comparisons
    obs.dif critical.dif stat.signif     p.value
```

```
1-2     476     48.00912        TRUE 1.22687e-83
```

Tip

For the Friedman test, the function requires that ONLY the variables needed for the analysis are included in the data frame