# Significance Testing of Rq3 Groups

Jeffrey M. Young

15 May 2020

Hello, in this walk-through we'll performing the significance testing on the financial and automotive datasets in light of the non-normality results from the `rq1_rq3` walk through. We are assuming you have read that walk-through first.

First, libraries, note that I have silenced the shadowing warnings from R:

```
library(ggplot2) #plotting
library(dplyr)   #dataframe manipulation
library(tidyr)   #dataframe manipulation
library(broom)   #for the tidy function
library(scales)  #for scientific function
```

Let's load the data, we immediately pipe this with dplyr for conveniences like manipulating the arrow to look nice and making factors. I drop the Name column because it is large and separated to the other columns. We use the name `Config` to stand for version variants throughout the scripts.

```
finRawFile <- "../../data/fin_rq3_singletons.csv"
autoRawFile <- "../../data/auto_rq3_singletons.csv"

finSingData <- read.csv(file=finRawFile) %>%
  mutate(Algorithm = as.factor(Algorithm), Config = as.factor(Config)) %>%
  mutate(Algorithm = gsub("-->", "\U27f6", Algorithm), data = "Financial") %>%
  group_by(Algorithm, Config) %>%
  mutate(TimeCalc = time -append(0,head(time, -1))) %>% filter(TimeCalc > 0)


autoSingData <- read.csv(file=autoRawFile) %>%
  mutate(Algorithm = as.factor(Algorithm), Config = as.factor(Config)) %>%
  mutate(Algorithm = gsub("-->", "\U27f6", Algorithm), data = "Auto") %>%
  group_by(Algorithm, Config) %>%
  mutate(TimeCalc = time -append(0,head(time, -1))) %>% filter(TimeCalc > 0)
```

## RQ3: Statistically meaningful overhead of the variational solver

In the last walk-through we found that the data was not normally distributed, and variance was not homogeneous about the sample groups. Thus, we cannot soundly perform a two-way ANOVA and trust the results. To overcome this we must use a `non-parametric` test for statistically significant comparison between groups. We choose to use a Kruskal-Wallis test, as this is commonly accepted as a `non-parametric` hypothesis test, unfortunately the Kruskal-Wallis test is only one-way. We begin with `financial` and then perform the test on `auto`. In the ANOVA analysis we were able to specify the model *with* an interaction: `TimeCalc ~ Config * Algorithm`. For the Kruskal-Wallis this is not the case, rather we must reproduce the analysis manually.

```r
## performing the test for Algorithm being significant
kruskal.test(TimeCalc ~ Algorithm, finSingData)
```

**Financial**

```
##
##  Kruskal-Wallis rank sum test
##
## data:  TimeCalc by Algorithm
## Kruskal-Wallis chi-squared = 316.61, df = 3, p-value < 2.2e-16
```

Algorithms statistically explain variance in the dataset, as expected.

```r
## and for versions
kruskal.test(TimeCalc ~ Config, finSingData)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  TimeCalc by Config
## Kruskal-Wallis chi-squared = 234.59, df = 9, p-value < 2.2e-16
```

And so do versions, again as expected.

```r
## We must manually construct the interaction
fin.inters <- interaction(finSingData$Algorithm, finSingData$Config)
kruskal.test(TimeCalc ~ fin.inters, finSingData)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  TimeCalc by fin.inters
## Kruskal-Wallis chi-squared = 580.46, df = 39, p-value < 2.2e-16
```

Interactions are also meaningful. Significant pairs are found with a Wilcox test:

```r
## perform the test
fin.pairs <- pairwise.wilcox.test(finSingData$TimeCalc, fin.inters,
                                  ## choose a bonferroni p-value adjustment for familywise
                                  ## error rate bonferroni is widely accepted, and very
                                  ## conservative, alternatives exist but are left to
                                  ## the interested
                                  p.adj="bonf", exact=FALSE,
                                  paired=FALSE) %>%
  ## cleanup
  tidy %>%
  separate(group1, sep=c(3,4), into = c("AlgLeft", "Dump", "ConfigLeft")) %>%
  separate(group2, sep=c(3,4), into = c("AlgRight", "Dump2", "ConfigRight")) %>%
  select(-Dump, -Dump2) %>%
  ## filter to version variants who were compared with themselves
  filter(ConfigRight == ConfigLeft) %>%
  ## add data column as `Financial` and sort by p-value
  mutate(data = "Financial") %>%
  arrange(p.value)

## view the data frame
fin.pairs
```

```
## # A tibble: 60 x 6
##    AlgLeft ConfigLeft AlgRight ConfigRight     p.value data
##    <chr>   <chr>      <chr>    <chr>             <dbl> <chr>
##  1 v v     V10        p p      V10         0.000000302 Financial
##  2 v v     V9         p p      V9          0.000000352 Financial
##  3 v v     V3         p p      V3          0.00000207  Financial
##  4 v v     V9         p v      V9          0.00000274  Financial
##  5 v v     V10        v p      V10         0.00000426  Financial
##  6 v v     V4         p p      V4          0.00000443  Financial
##  7 v v     V9         v p      V9          0.00000556  Financial
##  8 v v     V3         p v      V3          0.00000671  Financial
##  9 v v     V4         v p      V4          0.0000117   Financial
## 10 v v     V4         p v      V4          0.0000125   Financial
## # ... with 50 more rows
```

We see in the snippet of the `fin.pairs` data frame that all of the statistically significant comparisons involve `v-->v` (`vsat`), which confirm our observations from the rq3 violin plot in the `rq1_rq3.Rmd` walk-through. We now perform the same analysis on `auto`:

```r
## Algorithms are significant
kruskal.test(TimeCalc ~ Algorithm, autoSingData)
```

**Automotive**

```
##
##  Kruskal-Wallis rank sum test
##
## data:  TimeCalc by Algorithm
## Kruskal-Wallis chi-squared = 7.9673, df = 3, p-value = 0.04669
```

Algorithms are significant, but just by a small margin as shown by the `p-value`'s proximity to 0.05. Thus, in a more constrictive hypothesis test they would be considered not significant for the `auto` dataset. This would imply that `v-->v` is *not* statistically different from other algorithms for the `auto` dataset.

```r
## Versions are significant as expected
kruskal.test(TimeCalc ~ Config, autoSingData)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  TimeCalc by Config
## Kruskal-Wallis chi-squared = 124.56, df = 3, p-value < 2.2e-16
```

```r
## Interaction, also significant as expected
auto.inters <- interaction(autoSingData$Algorithm, autoSingData$Config)
kruskal.test(TimeCalc ~ auto.inters, autoSingData)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  TimeCalc by auto.inters
## Kruskal-Wallis chi-squared = 137.17, df = 15, p-value < 2.2e-16
```

For convenience we synthesize the `adj.p.value`'s into a dataframe:

```r
## Auto pairs which are significant
auto.pairs <- pairwise.wilcox.test(autoSingData$TimeCalc, auto.inters,
```

```r
                                    p.adj="bonf", method="holm"
                                  , exact=TRUE, paired=FALSE) %>%
  ## cleanup
  tidy %>%
              separate(group1, sep=c(3,4),
                       into = c("AlgLeft", "Dump", "ConfigLeft")) %>%
              separate(group2, sep=c(3,4),
                       into = c("AlgRight", "Dump2", "ConfigRight")) %>%
             select(-Dump, -Dump2) %>%
             ## restricting comparisons to the same versino variant
             filter(ConfigRight == ConfigLeft) %>%
             ## add data column as `Auto`, then sort
             mutate(data = "Auto") %>%
             arrange(p.value)

## observe the dataset
auto.pairs
```

```
## # A tibble: 24 x 6
##    AlgLeft ConfigLeft AlgRight ConfigRight p.value data
##    <chr>   <chr>      <chr>    <chr>         <dbl> <chr>
##  1 v v     V1         p v      V1          0.00260 Auto
##  2 v p     V1         p v      V1          0.0247  Auto
##  3 v v     V2         p v      V2          0.0870  Auto
##  4 v v     V3         v p      V3          0.0870  Auto
##  5 v v     V4         v p      V4          0.0870  Auto
##  6 v v     V1         p p      V1          0.251   Auto
##  7 v p     V3         p p      V3          0.466   Auto
##  8 v p     V1         p p      V1          0.624   Auto
##  9 v v     V2         p p      V2          0.821   Auto
## 10 p v     V1         p p      V1          1       Auto
## # ... with 14 more rows
```

We see that only two comparisons (v–>v, p–>v), and (v–>p, p–>v) for $V_1$ are meaningful in the `auto` dataset. This is good evidence the `v-->v` is not statistically worse for the `auto` dataset. The exact cause behind this result requires a more robust dataset, ideally, a dataset composed of several product lines which have a distribution of sharing ratios. Frankly, more data is needed to assess the signal. Finally, combine the datasets and synthesize a `p-value` matrix to visualize significant comparisons:

```r
options(scipen = 999)
rq3pvDF <- rbind(auto.pairs, fin.pairs) %>%
  mutate(Significance = case_when(p.value <= 0.05 ~ "Significant",
                                  TRUE ~ "Not Significant"),
         SigColor = paste(AlgLeft,":",Significance,sep=""),
         Version = factor(ConfigLeft, levels = c("V1", "V2", "V3", "V4", "V5", "V6", "V7", "V8", "V9",



ggplot(rq3pvDF, aes(x=AlgLeft, y=AlgRight, size=(1-p.value),
                                shape=SigColor, color=Significance)) +
  geom_point() +
  geom_jitter() +
  theme_classic() +
  facet_grid(data ~ Version, scales="free") +
  scale_size_continuous(range=c(2,4)) +
```

```r
scale_shape_manual(values = c(6,5,18,2,17)) +
guides(size=FALSE, shape=FALSE) +
ggtitle("RQ3: Statistical significance comparison matrix") +
ylab("Algorithm") +
theme(panel.grid.major.y = element_line(color = "lightgrey", linetype="dashed"),
      axis.text.x = element_text(angle = 90, hjust = 1),
      axis.title.x = element_blank(),
      legend.position = "bottom")
```