Ontology-based expert diagnosis reporting to reduce no-fault found scenarios in Industry 4.0

Analysis Report

```r
knitr::opts_chunk$set(echo = TRUE)
```

Abstract

This document presents the experimental results and their statistical analysis that contribute to the research validation of *"Ontology-based diagnosis reporting and monitoring to improve fault finding in Industry 4.0"*. This document presents the different analyses conducted using R to support that research validation. These analyses include: ontology expert interviews, ontology structural analysis, reporting usability surveys, monitoring efficiency experiments and monitoring usability surveys.

The following packages have been used for data analysis within the R code presented in this document.

```r
# Functions to manage and analyse data
library(dplyr)
library(tidyr)
library(car)
# Functions to work with plots
library(ggplot2)
library(grid)
library(ggpubr)
# Functions to theme graph colours
library(knitr)
library(kableExtra)
# Declare colour palettes
c03Palette = c("#1A406A","#7F7F7F","#0D1930")
c06Palette = c("#3F97C0","#1A406A","#9EBF43","#C2446F","#F2BC41","#5D4184")
c09Palette = c("#3F97C0","#1A406A","#9EBF43","#C2446F","#F2BC41","#5D4184",
               "#D32D40","#7F7F7F","#0D1930")
c12Palette = c("#D32D40","#F2BC41","#9EBF43","#3F97C0","#5D4184","#C2446F",
               "#791C24","#C77F3A","#617628","#2A386B","#402D55","#782A43")
# Declare plot theme
plotTheme <-
    theme(panel.background = element_rect(colour= "gray90", fill = "white"),
          strip.background = element_rect(colour = "gray90", fill = "white"),
          panel.grid.major = element_line(colour = "gray90", size = 0.35),
          panel.grid.minor = element_line(colour = "gray90", size = 0.175),
          axis.ticks = element_blank(),
          text = element_text(size = 10))
```

Data pre-processing: collection and formatting

Each data set has been prepared in R-readable formats (long tables) for further treatment.

First, read design results and format relevant columns.

```
# Format data as data frame
design <- read.csv("Data/1-OntologyInterviews.csv")
design$Interviewee <- as.factor(design$Interviewee)
# Visualise data frame
str(design)
```

```
## 'data.frame':    18 obs. of  5 variables:
##  $ Interviewee: Factor w/ 9 levels "1","2","3","4",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ Company    : Factor w/ 2 levels "Organisation A",..: 1 1 1 1 1 1 1 1 2 2 ...
##  $ Change     : Factor w/ 2 levels "Applied","Proposed": 2 1 2 1 2 1 2 1 2 1 ...
##  $ Number     : int  24 21 16 6 2 0 2 2 18 10 ...
##  $ Percentage : Factor w/ 8 levels "0%","100%","33%",..: 2 8 2 4 2 1 2 2 2 7 ...
```

Second, read assessment results and format relevant columns.

```
# Format data as data frame
assessment <- read.csv("Data/2-OntologyAssessment.csv")
assessment$Paper <- as.factor(assessment$Paper)
# Visualise data frame
str(assessment)
```

```
## 'data.frame':    33 obs. of  5 variables:
##  $ Paper   : Factor w/ 11 levels "1","2","3","4",..: 1 1 1 2 2 2 3 3 3 4 ...
##  $ Ontology: Factor w/ 11 levels "AHMK","AI2MS",..: 3 3 3 9 9 9 11 11 11 10 ...
##  $ DOI     : Factor w/ 11 levels "10.1016/j.aei.2014.10.001",..: 9 9 9 11 11 11 10 10 10 4 ...
##  $ Measure : Factor w/ 3 levels "AR","IR","RR": 3 1 2 3 1 2 3 1 2 3 ...
##  $ Result  : num  0.636 0.577 0.769 0.5 0.36 0.76 0.917 0.182 0.091 0.5 ...
```

Third, read interviews results and format relevant columns.

```
# Format data as data frame
interviews <- read.csv("Data/3-ReportingSurveys.csv")
interviews$Interviewee <- as.factor(interviews$Interviewee)
# Visualise data frame
str(interviews)
```

```
## 'data.frame':    108 obs. of  5 variables:
##  $ Interviewee: Factor w/ 9 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Company    : Factor w/ 2 levels "Organisation A",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Aspect     : Factor w/ 3 levels "Context","Structure",..: 3 3 3 3 2 2 2 2 1 1 ...
##  $ Criterion  : Factor w/ 4 levels "Accuracy","Completeness",..: 1 2 3 4 1 2 3 4 1 2 ...
##  $ Response   : int  7 5 7 7 7 5 7 7 7 5 ...
```

Fourth, read experiments results and format relevant columns.

```
# Format data as data frame
experiments <- read.csv("Data/4-MonitoringExperiments.csv")
experiments$Tester <- as.factor(experiments$Tester)
experiments$Solution <- ordered(experiments$Solution, levels = c("None","KRD","KRE"))
# Visualise data frame
str(experiments)
```

```
## 'data.frame':    48 obs. of  6 variables:
```

```
## $ Tester   : Factor w/ 48 levels "6","7","8","9",..: 1 2 3 4 5 6 7 8 9 10 ...
## $ Failure  : Factor w/ 2 levels "CNN","TEM": 2 1 2 2 2 2 1 1 1 1 ...
## $ Solution : Ord.factor w/ 3 levels "None"<"KRD"<"KRE": 1 1 1 3 1 1 3 3 2 1 ...
## $ Expertise: Factor w/ 2 levels "IT","NOIT": 2 1 1 2 2 1 2 2 1 2 ...
## $ Seconds  : int  72 48 90 16 70 60 10 12 10 100 ...
## $ Errors   : int  2 0 1 0 2 1 0 1 0 3 ...
```

Finally, read surveys results and format relevant columns.

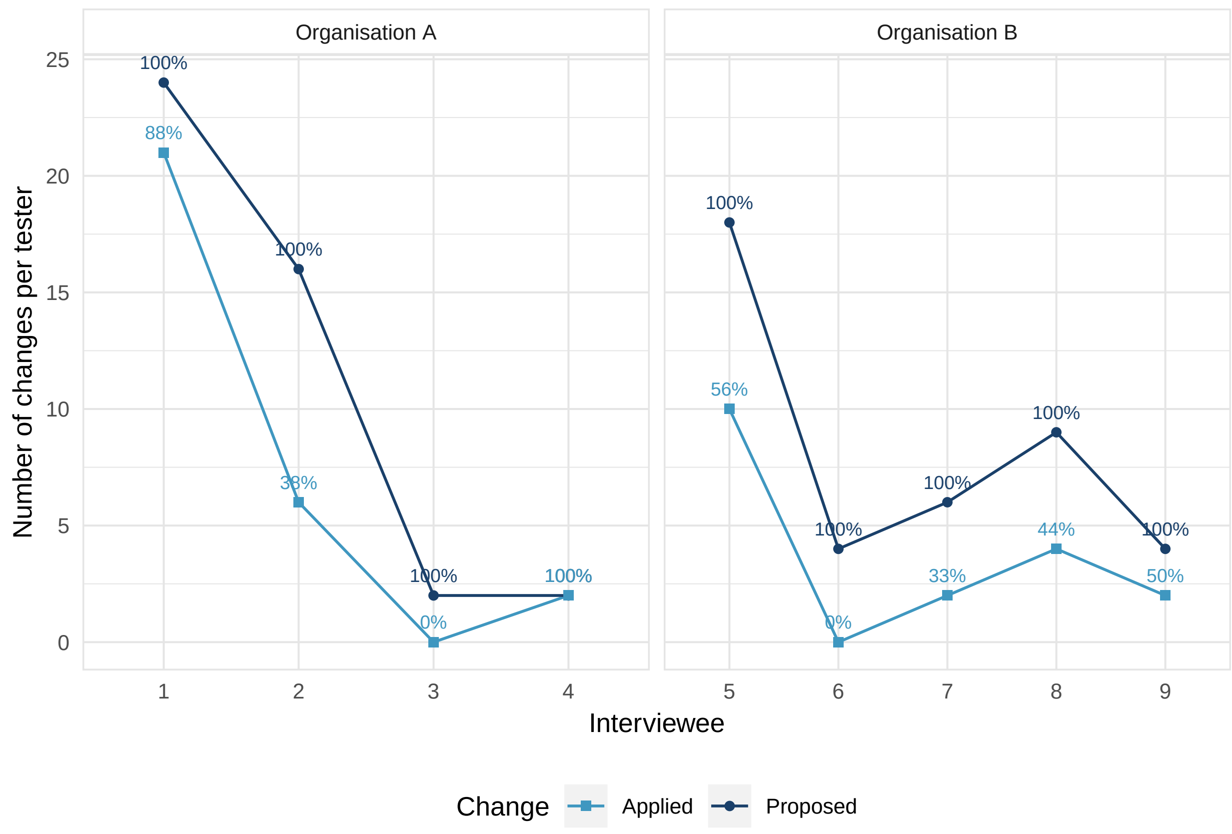
```r
# Format data as data frame
surveys <- read.csv("Data/5-MonitoringSurveys.csv")
surveys$Tester <- as.factor(surveys$Tester)
surveys$Solution <- ordered(surveys$Solution, levels = c("KRD","KRE"))
# Visualise data frame
str(surveys)
```

```
## 'data.frame':    192 obs. of  7 variables:
## $ Tester   : Factor w/ 32 levels "9","12","13",..: 1 1 1 1 1 1 2 2 2 2 ...
## $ Failure  : Factor w/ 2 levels "CNN","TEM": 2 2 2 2 2 2 1 1 1 1 ...
## $ Solution : Ord.factor w/ 2 levels "KRD"<"KRE": 2 2 2 2 2 2 2 2 2 2 ...
## $ Expertise: Factor w/ 2 levels "IT","NOIT": 2 2 2 2 2 2 2 2 2 2 ...
## $ Criterion: Factor w/ 2 levels "EASE-OF-USE",..: 1 1 1 2 2 2 1 1 1 2 ...
## $ Question : Factor w/ 6 levels "I found easy to understand the data presented by the monitoring too
## $ Response : int  5 5 5 5 5 5 5 5 5 5 ...
```

Ontology expert interviews

Percentage of accepted changes by tester are calculated in data formatting. Plot total changes, accepted changes and percentages by tester and organisation.

```
# Plot lines using abovementioned rationale and prepared theme
designPlot <-
  ggplot(design, aes(x = Interviewee, y = Number, group = Change,
                     colour = Change, shape = Change)) +
  geom_line() +
  geom_point() +
  geom_text(aes(label = Percentage), size = 2.5, vjust = -1,
            show.legend = FALSE) +
  facet_grid(. ~ Company, scales = "free_x") +
  scale_shape_manual(values = c(15,16,17,18)) +
  scale_colour_manual(values=c06Palette) +
  labs(x = "Interviewee", y = "Number of changes per tester") +
  theme(legend.position = "bottom") +
  plotTheme
# Visualise plot
designPlot
```



Tabulate means and standard deviations for changes proposed and accepted by interviewee.

```
# Calculate using group_by_ function from dplyr
designStats <- group_by(design, Change) %>%
  summarise(count=n(), mean = mean(Number,na.rm = TRUE),
            sd = sd(Number,na.rm = TRUE))
```

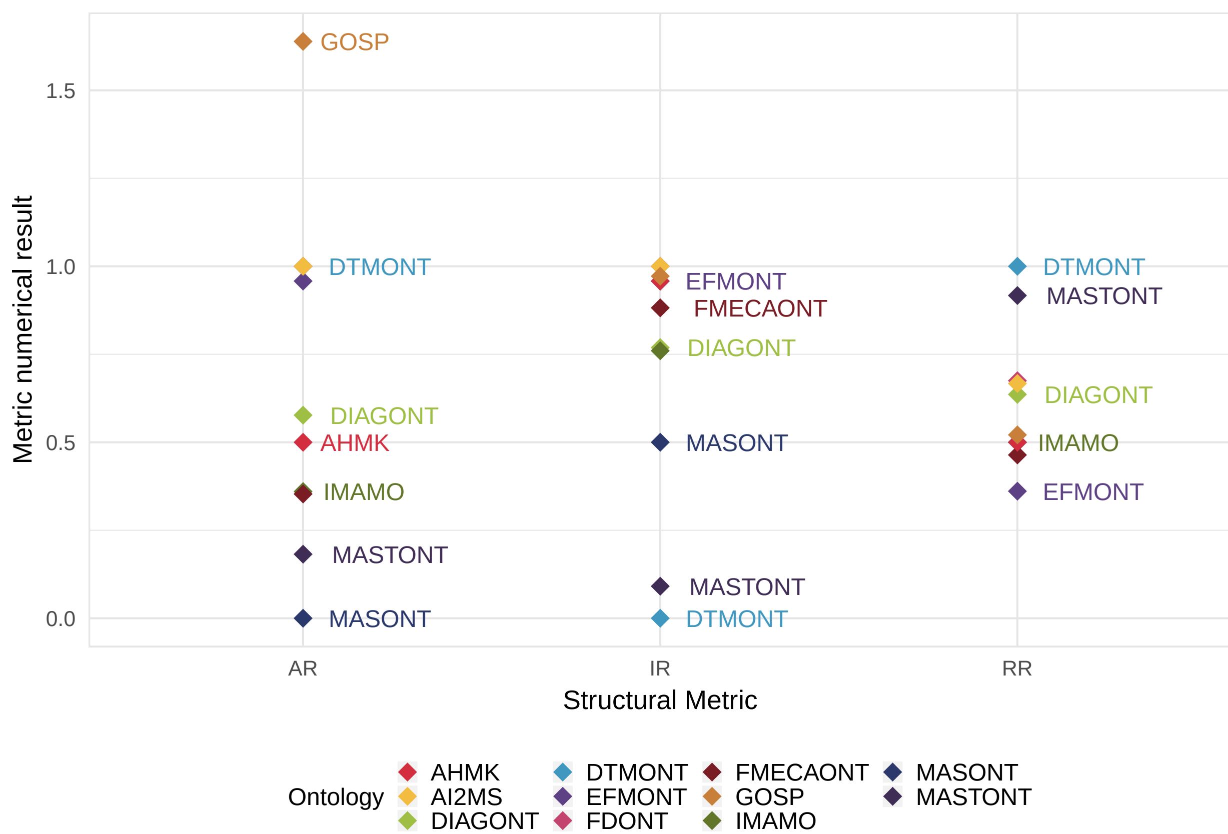
```r
# Visualise tabulated result
kable(designStats, format = "latex", booktabs = TRUE)
```

Change	count	mean	sd
Applied	9	5.222222	6.704062
Proposed	9	9.444444	7.986099

Ontology structural analysis

Structural measures calculated in data formatting. Plot structural measurements (RR,AR,IR) by ontology and criterion.

```
# Plot points using abovementioned rationale and prepared theme
assessmentPlot <-
    ggplot(assessment, aes(x = Measure, y = Result, colour = Ontology)) +
    geom_point(shape = 18, size = 3) +
    geom_text(aes(label = Ontology), size = 3, hjust = -0.25,
              check_overlap = TRUE, show.legend = FALSE) +
    scale_colour_manual(values = c12Palette) +
    labs(x = "Structural Metric", y = "Metric numerical result") +
    theme(legend.position = "bottom", legend.key.size = unit(0.3, "cm"),
          legend.title = element_text(size = 9),
          legend.text = element_text(size = 9)) +
    guides(colour = guide_legend(nrow = 3)) +
    plotTheme
# Visualise plot
assessmentPlot
```



Calculate mean and standard deviation for each structural measure.

```
# Calculate using group_by_ function from dplyr
assessmentStats <- group_by(assessment, Measure) %>%
  summarise(count=n(), mean = mean(Result,na.rm = TRUE),
            sd = sd(Result,na.rm = TRUE))
# Visualise tabulated result
kable(assessmentStats, format = "latex", booktabs = TRUE)
```

Measure	count	mean	sd
AR	11	0.6880909	0.4760448
IR	11	0.7172727	0.3641698
RR	11	0.6128182	0.1954353

Tabulate ontology and doi by structural measure. Calculate diagont ranking by structural measure.

```
# Transform assessment data from long to wide format using tidyr functions
assessmentWide <- spread(assessment,Measure,Result)
# Calculate ontology rankings for each structural measure
# For descending order, negate the column of values to order
assessmentWide$RRRank <- rank(-assessmentWide$RR, ties.method = "random")
assessmentWide$ARRank <- rank(-assessmentWide$AR, ties.method = "random")
assessmentWide$IRRank <- rank(-assessmentWide$IR, ties.method = "random")
# Tabulate ordering structural measure and its ontology ranking
kable(assessmentWide[, c(1, 2, 6, 7, 4, 8, 5, 9, 3)],
      format = "latex", booktabs = TRUE) %>%
        kable_styling(latex_options = "scale_down")
```

Paper	Ontology	RR	RRRank	AR	ARRank	IR	IRRank	DOI
1	DIAGONT	0.636	5	0.577	6	0.769	7	10.17862/cranfield.rd.12279152
2	IMAMO	0.500	8	0.360	8	0.760	8	10.3233/ao-2012-0112
3	MASTONT	0.917	2	0.182	10	0.091	10	10.3182/20120523-3-RO-2023.00124
4	MASONT	0.500	9	0.000	11	0.500	9	10.1016/j.jnca.2012.11.004
5	FMECAONT	0.464	10	0.353	9	0.882	6	10.1016/j.aei.2014.10.001
6	DTMONT	1.000	1	1.000	2	0.000	11	10.1016/j.compind.2013.03.001
7	EFMONT	0.361	11	0.958	5	0.958	5	10.1016/j.knosys.2014.02.002
8	FDONT	0.675	3	1.000	3	1.000	1	10.1109/TSMC.2013.2281963
9	AI2MS	0.667	4	1.000	4	1.000	2	10.1109/indin.2014.6945616
10	AHMK	0.500	7	0.500	7	0.958	4	10.1109/icqr2mse.2012.6246302
11	GOSP	0.521	6	1.639	1	0.972	3	10.1016/j.jlp.2012.10.001

Reporting usability surveys

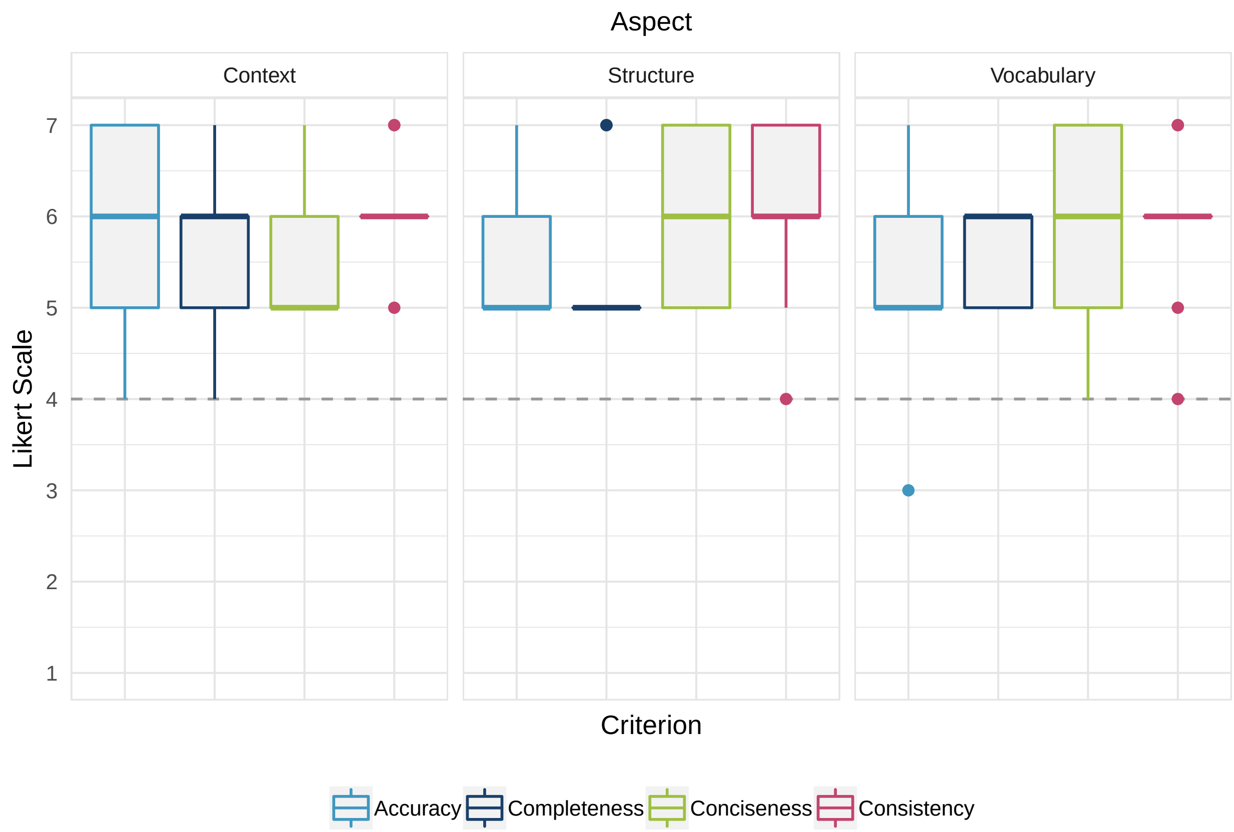
Calculate usability statistics and present survey responses.

```
# Calculate using group_by_ function from dplyr
interviewsStats <- group_by(interviews, Aspect, Criterion) %>%
    summarise(count=n(), mean = mean(Response,na.rm = TRUE),
              sd = sd(Response,na.rm = TRUE))
# Visualise tabulated result
kable(interviewsStats, format = "latex", booktabs = TRUE)
```

Aspect	Criterion	count	mean	sd
Context	Accuracy	9	5.888889	1.0540926
Context	Completeness	9	5.555556	0.8819171
Context	Conciseness	9	5.666667	0.8660254
Context	Consistency	9	6.111111	0.6009252
Structure	Accuracy	9	5.666667	0.8660254
Structure	Completeness	9	5.444444	0.8819171
Structure	Conciseness	9	5.888889	0.9279607
Structure	Consistency	9	6.000000	1.0000000
Vocabulary	Accuracy	9	5.444444	1.2360331
Vocabulary	Completeness	9	5.555556	0.5270463
Vocabulary	Conciseness	9	5.888889	1.0540926
Vocabulary	Consistency	9	5.888889	0.9279607

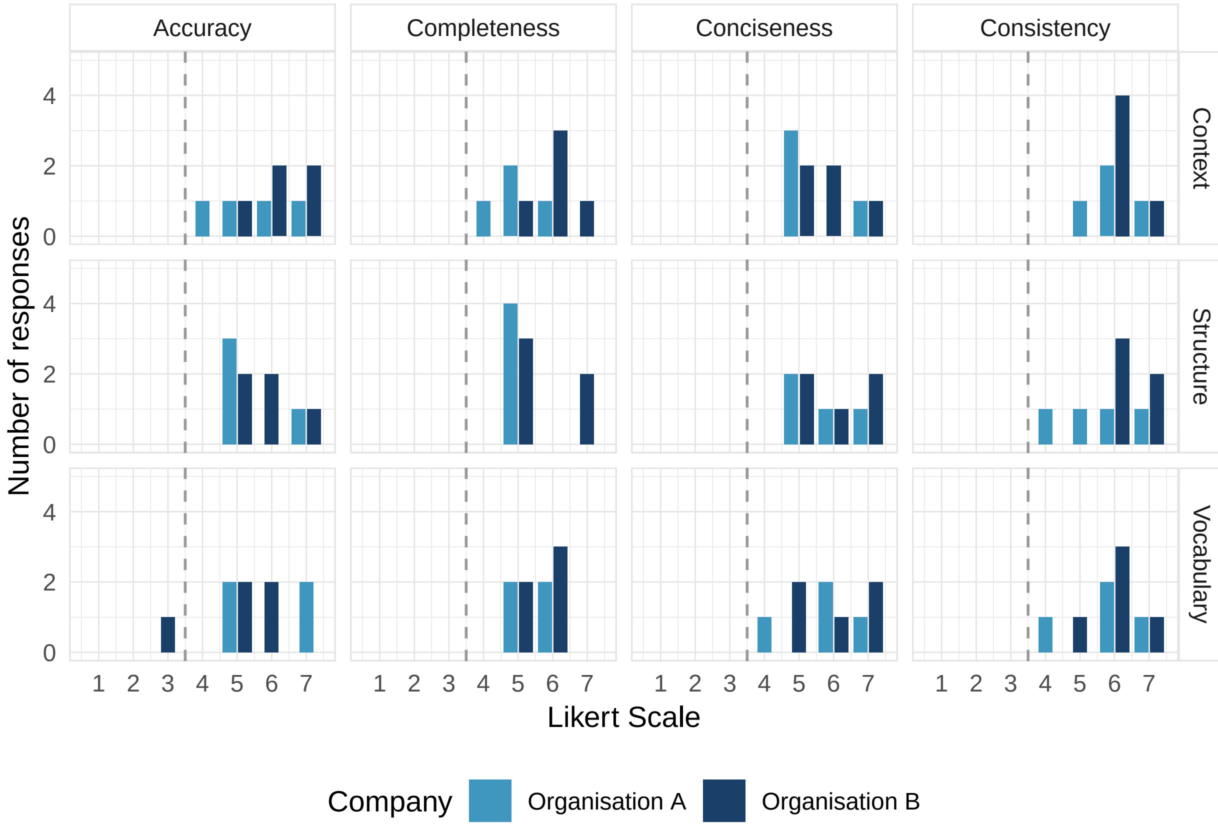
Plot usability statistics by criterion and aspect.

```
# Plot box and whiskers using abovementioned rationale and prepared theme
interviewsPlotStats <-
  ggplot(interviews, aes(x = Criterion, y = Response, colour = Criterion)) +
  geom_hline(yintercept = 4, colour = "gray60", linetype = "dashed") +
  geom_boxplot(fill = "gray95", lwd = 0.5) +
  facet_grid(. ~ Aspect) +
  scale_colour_manual(values = c06Palette) +
  scale_y_continuous(limits = c(1,7), breaks = c(1,2,3,4,5,6,7)) +
  labs(y = "Likert Scale", title = "Aspect") +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, size = 10),
        axis.text.x = element_blank(),
        legend.title = element_blank()) +
  plotTheme
# Visualise plot
interviewsPlotStats
```



Plot responses count by criterion, aspect and organisation.

```
# Plot bars using abovementioned rationale and prepared theme
interviewsPlotCount <- ggplot(interviews, aes(x = Response, fill = Company)) +
    geom_bar(position = position_dodge2(preserve = "single"), width = 0.9) +
    geom_vline(xintercept = 3.5, linetype = "dashed", color = "gray60") +
    facet_grid(Aspect ~ Criterion) +
    scale_fill_manual(values=c06Palette) +
    scale_y_continuous("Number of responses", limits = c(0,5),
                       breaks = c(0,2,4)) +
    scale_x_continuous("Likert Scale", limits = c(0.5,7.5),
                       breaks = c(1,2,3,4,5,6,7)) +
    theme(legend.position = "bottom",
          panel.background = element_rect(colour = "gray90", fill = "white"),
          panel.grid.major = element_line(colour = "gray90", size = 0.25),
          panel.grid.minor = element_line(colour = "gray90", size = 0.125),
          axis.ticks = element_blank(),
          strip.background = element_rect(colour = "gray90", fill = "white"),
          text = element_text(size = 11))
# Visualise plot
interviewsPlotCount
```


Accuracy
Completeness
Conciseness
Consistency
Context
Structure
Vocabulary
Number of responses
0
2
4
1 2 3 4 5 6 7
Likert Scale
Company
Organisation A
Organisation B

Monitoring efficiency experiments

Calculate summary statistics for experiments results.

```
summary(experiments)
```

```
##      Tester    Failure  Solution  Expertise    Seconds
##  6      : 1   CNN:24   None:16   IT  :24   Min.   :  5.00
##  7      : 1   TEM:24   KRD :16   NOIT:24   1st Qu.: 15.00
##  8      : 1            KRE :16             Median : 30.00
##  9      : 1                                Mean   : 37.19
##  10     : 1                                3rd Qu.: 50.00
##  11     : 1                                Max.   :105.00
##  (Other):42
##      Errors
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :1.0000
##  Mean   :0.9375
##  3rd Qu.:1.2500
##  Max.   :3.0000
##
```

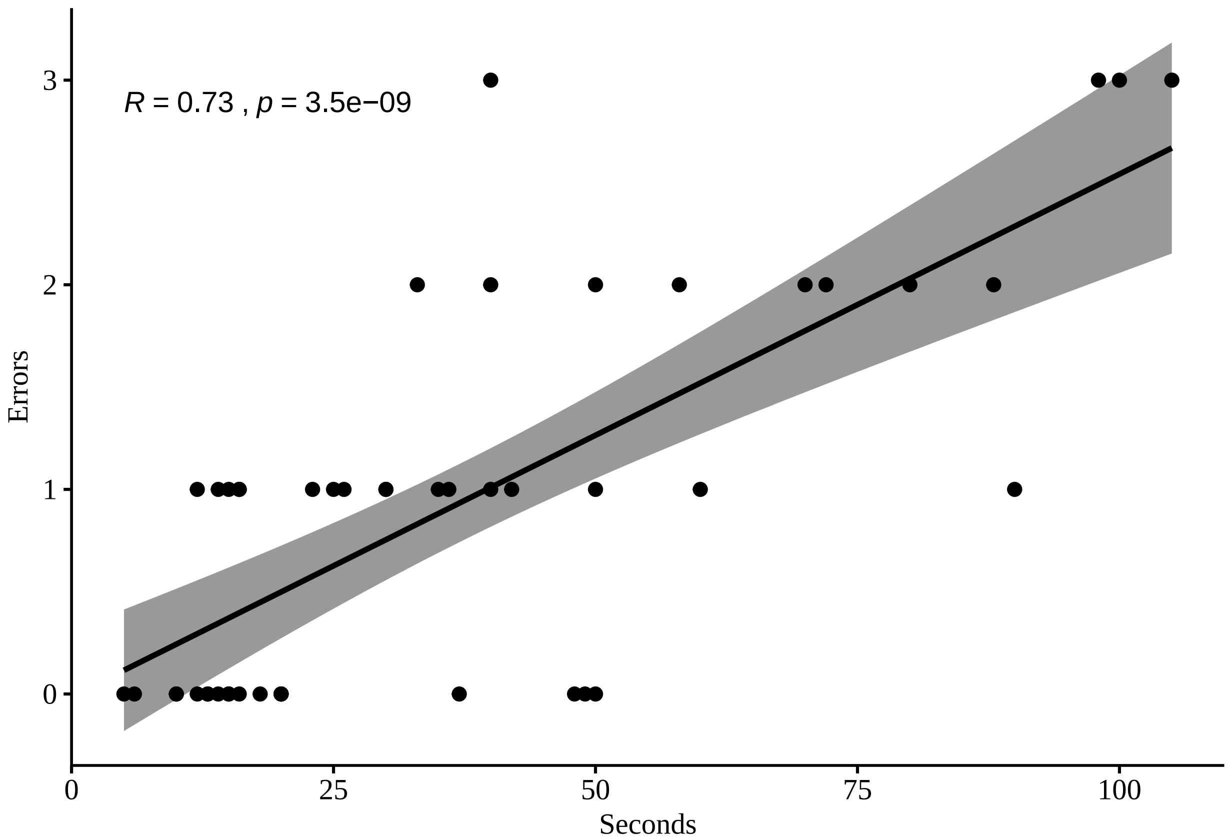
Evaluate correlation between response variables

```
# Evaluate correlation coefficient using Pearson's method
cor.test(experiments$Seconds, experiments$Errors, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  experiments$Seconds and experiments$Errors
## t = 7.2766, df = 46, p-value = 3.511e-09
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5647637 0.8408737
## sample estimates:
##       cor
## 0.7315132
```

Plot correlation between response variables errors and seconds.

```
# Plot points and line using ggscatter from ggpubr
efficiencyPlotCor <- ggscatter(experiments, x = "Seconds", y = "Errors",
                               add = "reg.line", conf.int = TRUE,
                               cor.coef = TRUE, cor.method = "pearson") +
                     font("axis.title", size = 11, family = "Times") +
                     font("axis.text", size = 11, family = "Times")
# Visualise plot
efficiencyPlotCor
```



R = 0.73 , p = 3.5e−09
Errors
3
2
1
0
0
25
50
75
100
Seconds

Stop watch errors study

Exploratory analysis

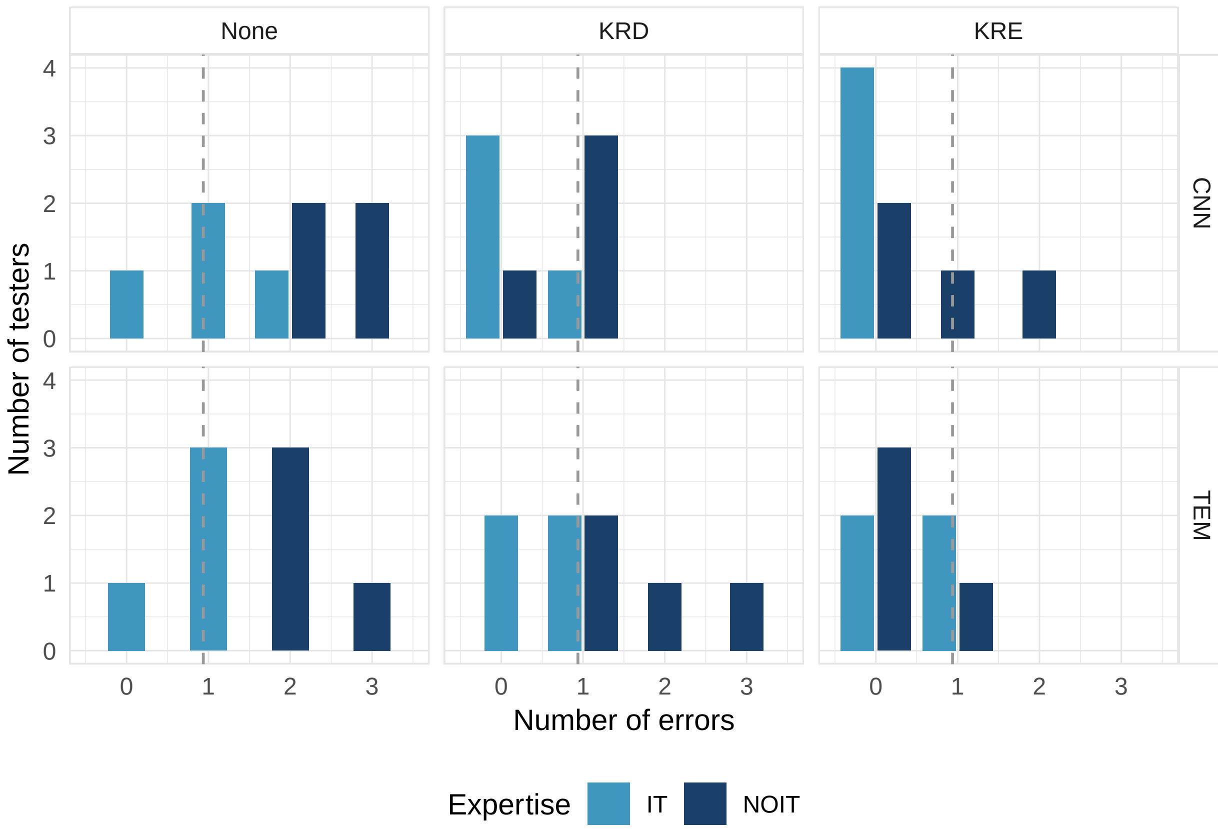
Calculate errors statistics by solution.

```
# Calculate using group_by_ function from dplyr
errorsStatsSolution <- group_by(experiments, Solution) %>%
    summarise(count=n(), mean = mean(Errors, na.rm = TRUE),
              sd = sd(Errors, na.rm = TRUE))
# Visualise tabulated result
kable(errorsStatsSolution, format = "latex", booktabs = TRUE)
```

Solution	count	mean	sd
None	16	1.6250	0.9574271
KRD	16	0.8125	0.8341663
KRE	16	0.3750	0.6191392

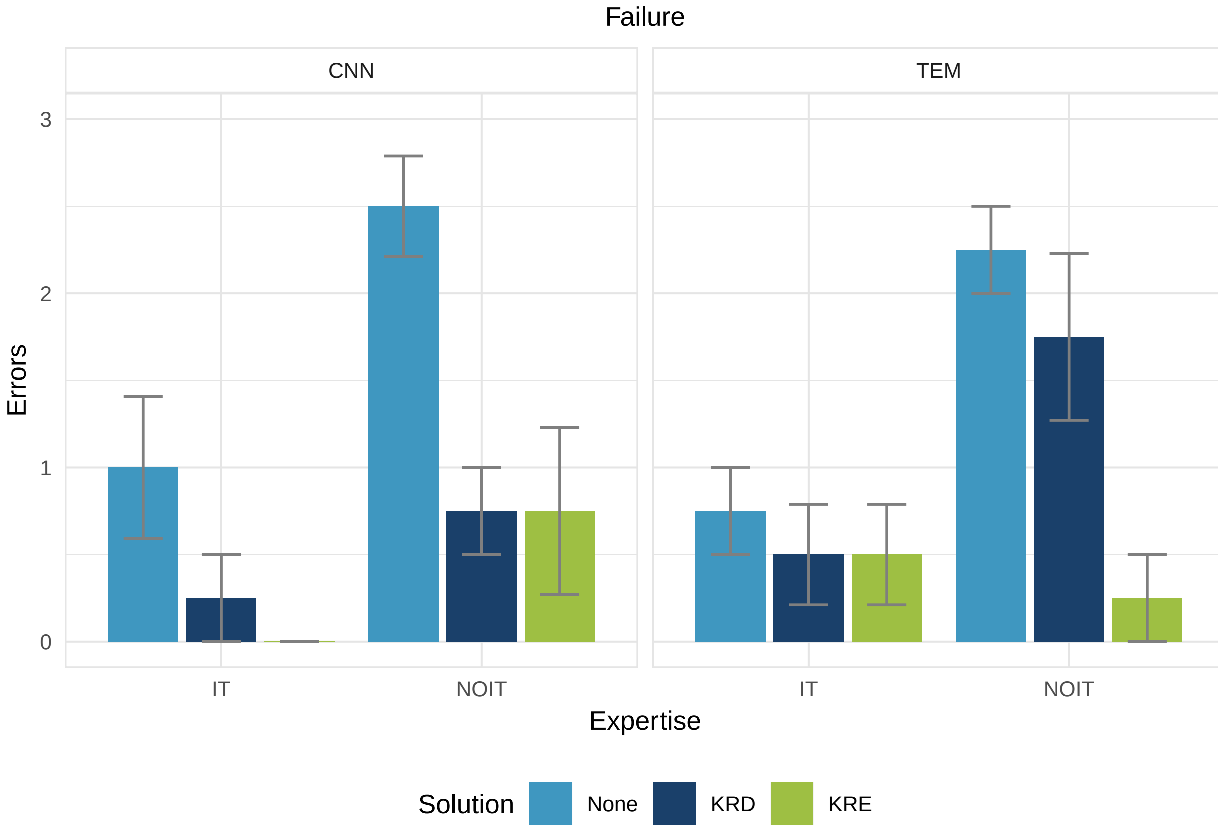
Calculate errors count with mean by failure, solution and expertise.

```
# Plot bars using abovementioned rationale and prepared theme
errorsPlotCount <- ggplot(experiments, aes(x = Errors, fill = Expertise)) +
    geom_bar(position = position_dodge2(preserve = "single"), width = 0.9) +
    geom_vline(xintercept = 0.9375, linetype = "dashed", color = "gray60") +
    facet_grid(Failure ~ Solution) +
    scale_fill_manual(values=c06Palette) +
    scale_y_continuous("Number of testers", limits = c(0,4),
                       breaks = c(0,1,2,3,4)) +
    scale_x_continuous("Number of errors", limits = c(-0.5,3.5),
                       breaks = c(0,1,2,3)) +
    theme(legend.position = "bottom",
          panel.background = element_rect(colour = "gray90", fill = "white"),
          panel.grid.major = element_line(colour = "gray90", size = 0.25),
          panel.grid.minor = element_line(colour = "gray90", size = 0.125),
          axis.ticks = element_blank(),
          strip.background = element_rect(colour = "gray90", fill = "white"),
          text = element_text(size = 11))
# Visualise plot
errorsPlotCount
```



Plot errors statistics by solution, failure and expertise.

```
# Plot box and whiskers using abovementioned rationale and prepared theme
errorsPlotStats <-
  ggplot(experiments, aes(x = Expertise, y = Errors, fill = Solution)) +
  geom_bar(stat = "summary", fun.y = "mean",
           position = position_dodge2(preserve = "single")) +
  geom_errorbar(stat = "summary", fun.data = "mean_se",
                position = position_dodge2(width = 0.5, padding = 0.5),
                colour = "gray50", lwd = 0.5) +
  facet_grid(. ~ Failure, scales = "free_x") +
  scale_y_continuous(limits = c(0,3), breaks = c(0,1,2,3)) +
  scale_fill_manual(values = c06Palette) +
  labs(y = "Errors", title = "Failure") +
  theme (legend.position = "bottom",
         plot.title = element_text(size = 10, hjust = 0.5)) +
  plotTheme
# Visualise plot
errorsPlotStats
```



Evaluate errors variability with expertise using t-test.

```
t.test(Errors ~ Expertise, data = experiments)
```

```
##
##  Welch Two Sample t-test
##
## data:  Errors by Expertise
## t = -3.5452, df = 36.086, p-value = 0.001107
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.3755113 -0.3744887
## sample estimates:
##   mean in group IT mean in group NOIT
##              0.500              1.375
```

Evaluate errors variability with failure using t-test.

```
t.test(Errors ~ Failure, data = experiments)
```

```
##
##  Welch Two Sample t-test
##
## data:  Errors by Failure
## t = -0.44983, df = 45.826, p-value = 0.655
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.6844076  0.4344076
## sample estimates:
```

```
## mean in group CNN mean in group TEM
##             0.875            1.000
```

Evaluate errors variability with solution using one-way anova.

```r
summary(aov(Errors ~ Solution, data = experiments))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Solution     2  12.88   6.438   9.676 0.00032 ***
## Residuals   45  29.94   0.665
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Two effects (expertise and solution) seem relevant to errors (interaction) variability. Hence, it seems necessary to do a multi-variate analysis and test its assumptions.

Correlation assumptions testing

Prepare errors data for multivariate analysis. Remove outliers from outside quartiles using box and whiskers plot. Fit clean data to linear model for conducting multi-variate anova in expertise, solution and failure. Calculate residuals, predicted values and squared predicted values for assumptions testing.

```r
# Calculate outliers using box and whiskers plot
subset(experiments,experiments$Errors %in% boxplot.stats(experiments$Errors)$out)
```

```
## [1] Tester    Failure   Solution  Expertise Seconds   Errors
## <0 rows> (or 0-length row.names)
```

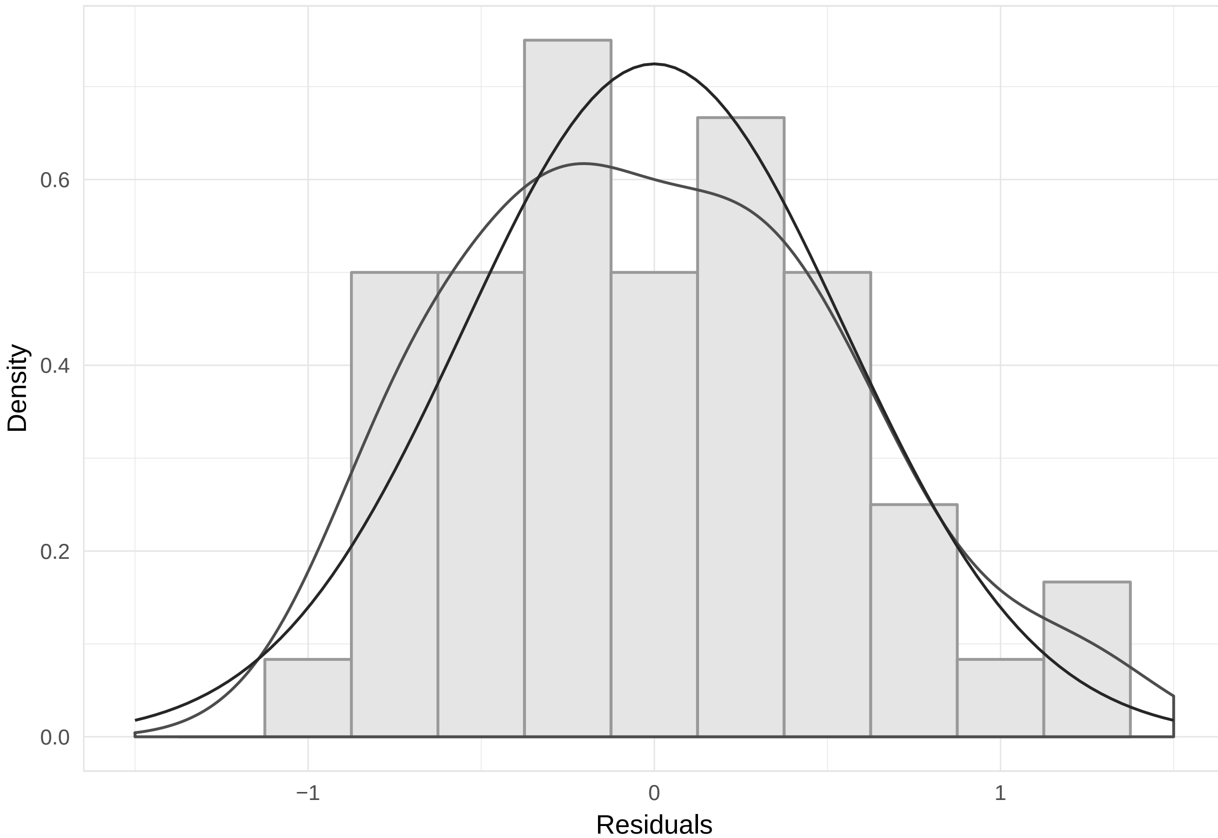
```r
# No outliers found
# Subset data removing non-relevant variables
errors <- within(experiments, rm(Seconds))
# Fit data to a linear model to run anova considering interactions of effects
errorsLM <- lm(Errors ~ Solution*Failure*Expertise, errors)
# Calculate residuals, predicted values and squared predicted values
errors$Residuals <- residuals(errorsLM)
errors$Predicted <- predict(errorsLM)
errors$SqrdPredicted <- predict(errorsLM)^2
```

Plot errors as histogram and errors residuals as line to evaluate normality. Include line of normal distribution for further comparison.

```r
# Plot residuals histogram and overlay normal distribution shape
errorsPlotNorm <- ggplot(errors) +
  geom_histogram(aes(x = Residuals, y = ..density..), binwidth = 0.25,
                 fill = "gray90", colour = "gray60") +
  geom_density(aes(x = Residuals, y = ..density..), colour = "gray30") +
  stat_function(fun = function(x,mean,sd,n){
    dnorm(x = x, mean = mean, sd = sd)
  }, args = with(errors,
                 c(mean = mean(Residuals), sd = sd(Residuals),
                   n = length(Residuals))), colour = "gray15") +
  xlim(-1.5, 1.5) + scale_y_continuous("Density") +
  theme(panel.background = element_rect(colour = "gray90", fill = "white"),
        panel.grid.major = element_line(colour = "gray90", size = 0.25),
        panel.grid.minor = element_line(colour = "gray90", size = 0.125),
        axis.ticks = element_blank(),
        strip.background = element_rect(colour = "gray90", fill = "white"),
```

```r
        text = element_text(size = 10))
# Visualise plot
errorsPlotNorm
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



Calculate Shapiro test to evaluate normality.

```r
# Run shapiro test to evaluate normality
shapiro.test(errors$Residuals)
```

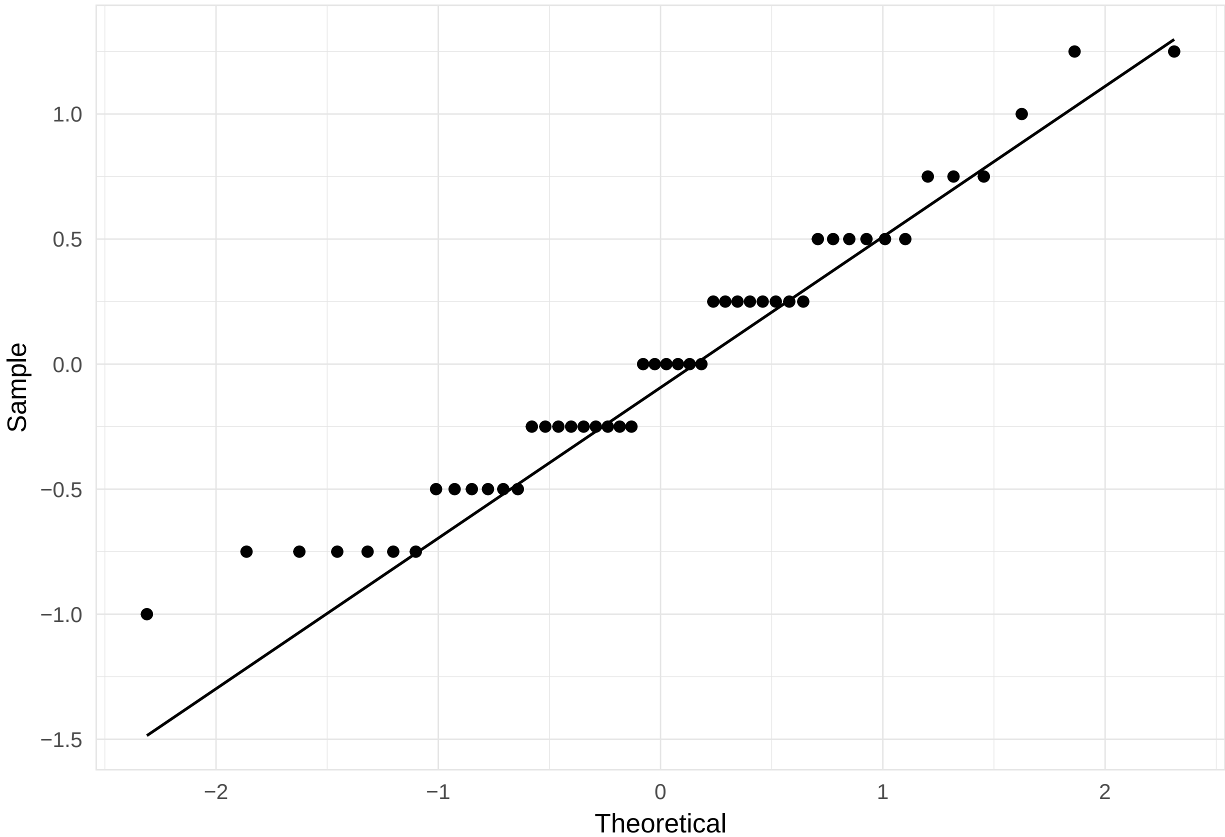
```
##
##  Shapiro-Wilk normality test
##
## data:  errors$Residuals
## W = 0.96126, p-value = 0.1136
```

```r
# Data is distributed normally if p-value is below 0.05
```

Plot errors lm qq line to evaluate linearity.

```r
# Plot residuals using ggplot stats
errorsPlotLin <- ggplot(errors, aes(sample = Residuals)) +
  stat_qq() + stat_qq_line() +
  scale_x_continuous("Theoretical") + scale_y_continuous("Sample") +
  theme(panel.background = element_rect(colour = "gray90", fill = "white"),
        panel.grid.major = element_line(colour = "gray90", size = 0.25),
        panel.grid.minor = element_line(colour = "gray90", size = 0.125),
        axis.ticks = element_blank(),
        strip.background = element_rect(colour = "gray90", fill = "white"),
```

```
        text = element_text(size = 10))
# Visualise plot
errorsPlotLin
```



Calculate Bartlett test to evaluate homogeinity.

```
# Run Bartlett test to evaluate homogeinity
bartlett.test(Errors ~ interaction(Solution, Failure, Expertise), data = errors)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  Errors by interaction(Solution, Failure, Expertise)
## Bartlett's K-squared = Inf, df = 11, p-value < 2.2e-16
```

Correlation analysis

Calculate three-way anova for effects (solution, failure and expertise) and interaction (errors) multi-variate analysis. Tabulate summarised test results.

```
# Run anova test on cleaned results with outliers removed
errorsAnova <- aov(Errors ~ Solution*Failure*Expertise, errors)
# Visualise tabulated results
summary(errorsAnova)
```

```
##                             Df Sum Sq Mean Sq F value   Pr(>F)
## Solution                     2 12.875   6.438  16.263 9.29e-06 ***
## Failure                      1  0.188   0.188   0.474   0.4957
```

```
## Expertise                   1  9.188   9.188  23.211 2.62e-05 ***
## Solution:Failure            2  1.625   0.812   2.053   0.1432
## Solution:Expertise          2  3.125   1.563   3.947   0.0282 *
## Failure:Expertise           1  0.021   0.021   0.053   0.8198
## Solution:Failure:Expertise  2  1.542   0.771   1.947   0.1574
## Residuals                  36 14.250   0.396
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Calculate Tukey HDS test results to post-hoc compare groups means.

```
# Run post-hoc pairwise t-test comparisons
TukeyHSD(aov(Errors ~ Solution:Failure:Expertise, errors))
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Errors ~ Solution:Failure:Expertise, data = errors)
##
## $`Solution:Failure:Expertise`
##                                   diff         lwr         upr
## KRD:CNN:IT-None:CNN:IT      -7.500000e-01 -2.30276818  0.80276818
## KRE:CNN:IT-None:CNN:IT      -1.000000e+00 -2.55276818  0.55276818
## None:TEM:IT-None:CNN:IT     -2.500000e-01 -1.80276818  1.30276818
## KRD:TEM:IT-None:CNN:IT      -5.000000e-01 -2.05276818  1.05276818
## KRE:TEM:IT-None:CNN:IT      -5.000000e-01 -2.05276818  1.05276818
## None:CNN:NOIT-None:CNN:IT    1.500000e+00 -0.05276818  3.05276818
## KRD:CNN:NOIT-None:CNN:IT    -2.500000e-01 -1.80276818  1.30276818
## KRE:CNN:NOIT-None:CNN:IT    -2.500000e-01 -1.80276818  1.30276818
## None:TEM:NOIT-None:CNN:IT    1.250000e+00 -0.30276818  2.80276818
## KRD:TEM:NOIT-None:CNN:IT     7.500000e-01 -0.80276818  2.30276818
## KRE:TEM:NOIT-None:CNN:IT    -7.500000e-01 -2.30276818  0.80276818
## KRE:CNN:IT-KRD:CNN:IT       -2.500000e-01 -1.80276818  1.30276818
## None:TEM:IT-KRD:CNN:IT       5.000000e-01 -1.05276818  2.05276818
## KRD:TEM:IT-KRD:CNN:IT        2.500000e-01 -1.30276818  1.80276818
## KRE:TEM:IT-KRD:CNN:IT        2.500000e-01 -1.30276818  1.80276818
## None:CNN:NOIT-KRD:CNN:IT     2.250000e+00  0.69723182  3.80276818
## KRD:CNN:NOIT-KRD:CNN:IT      5.000000e-01 -1.05276818  2.05276818
## KRE:CNN:NOIT-KRD:CNN:IT      5.000000e-01 -1.05276818  2.05276818
## None:TEM:NOIT-KRD:CNN:IT     2.000000e+00  0.44723182  3.55276818
## KRD:TEM:NOIT-KRD:CNN:IT      1.500000e+00 -0.05276818  3.05276818
## KRE:TEM:NOIT-KRD:CNN:IT      7.216450e-16 -1.55276818  1.55276818
## None:TEM:IT-KRE:CNN:IT       7.500000e-01 -0.80276818  2.30276818
## KRD:TEM:IT-KRE:CNN:IT        5.000000e-01 -1.05276818  2.05276818
## KRE:TEM:IT-KRE:CNN:IT        5.000000e-01 -1.05276818  2.05276818
## None:CNN:NOIT-KRE:CNN:IT     2.500000e+00  0.94723182  4.05276818
## KRD:CNN:NOIT-KRE:CNN:IT      7.500000e-01 -0.80276818  2.30276818
## KRE:CNN:NOIT-KRE:CNN:IT      7.500000e-01 -0.80276818  2.30276818
## None:TEM:NOIT-KRE:CNN:IT     2.250000e+00  0.69723182  3.80276818
## KRD:TEM:NOIT-KRE:CNN:IT      1.750000e+00  0.19723182  3.30276818
## KRE:TEM:NOIT-KRE:CNN:IT      2.500000e-01 -1.30276818  1.80276818
## KRD:TEM:IT-None:TEM:IT      -2.500000e-01 -1.80276818  1.30276818
## KRE:TEM:IT-None:TEM:IT      -2.500000e-01 -1.80276818  1.30276818
## None:CNN:NOIT-None:TEM:IT    1.750000e+00  0.19723182  3.30276818
## KRD:CNN:NOIT-None:TEM:IT     0.000000e+00 -1.55276818  1.55276818
```

```
## KRE:CNN:NOIT-None:TEM:IT    -6.661338e-16 -1.55276818  1.55276818
## None:TEM:NOIT-None:TEM:IT    1.500000e+00 -0.05276818  3.05276818
## KRD:TEM:NOIT-None:TEM:IT     1.000000e+00 -0.55276818  2.55276818
## KRE:TEM:NOIT-None:TEM:IT    -5.000000e-01 -2.05276818  1.05276818
## KRE:TEM:IT-KRD:TEM:IT        1.665335e-15 -1.55276818  1.55276818
## None:CNN:NOIT-KRD:TEM:IT     2.000000e+00  0.44723182  3.55276818
## KRD:CNN:NOIT-KRD:TEM:IT      2.500000e-01 -1.30276818  1.80276818
## KRE:CNN:NOIT-KRD:TEM:IT      2.500000e-01 -1.30276818  1.80276818
## None:TEM:NOIT-KRD:TEM:IT     1.750000e+00  0.19723182  3.30276818
## KRD:TEM:NOIT-KRD:TEM:IT      1.250000e+00 -0.30276818  2.80276818
## KRE:TEM:NOIT-KRD:TEM:IT     -2.500000e-01 -1.80276818  1.30276818
## None:CNN:NOIT-KRE:TEM:IT     2.000000e+00  0.44723182  3.55276818
## KRD:CNN:NOIT-KRE:TEM:IT      2.500000e-01 -1.30276818  1.80276818
## KRE:CNN:NOIT-KRE:TEM:IT      2.500000e-01 -1.30276818  1.80276818
## None:TEM:NOIT-KRE:TEM:IT     1.750000e+00  0.19723182  3.30276818
## KRD:TEM:NOIT-KRE:TEM:IT      1.250000e+00 -0.30276818  2.80276818
## KRE:TEM:NOIT-KRE:TEM:IT     -2.500000e-01 -1.80276818  1.30276818
## KRD:CNN:NOIT-None:CNN:NOIT  -1.750000e+00 -3.30276818 -0.19723182
## KRE:CNN:NOIT-None:CNN:NOIT  -1.750000e+00 -3.30276818 -0.19723182
## None:TEM:NOIT-None:CNN:NOIT -2.500000e-01 -1.80276818  1.30276818
## KRD:TEM:NOIT-None:CNN:NOIT  -7.500000e-01 -2.30276818  0.80276818
## KRE:TEM:NOIT-None:CNN:NOIT  -2.250000e+00 -3.80276818 -0.69723182
## KRE:CNN:NOIT-KRD:CNN:NOIT   -6.661338e-16 -1.55276818  1.55276818
## None:TEM:NOIT-KRD:CNN:NOIT   1.500000e+00 -0.05276818  3.05276818
## KRD:TEM:NOIT-KRD:CNN:NOIT    1.000000e+00 -0.55276818  2.55276818
## KRE:TEM:NOIT-KRD:CNN:NOIT   -5.000000e-01 -2.05276818  1.05276818
## None:TEM:NOIT-KRE:CNN:NOIT   1.500000e+00 -0.05276818  3.05276818
## KRD:TEM:NOIT-KRE:CNN:NOIT    1.000000e+00 -0.55276818  2.55276818
## KRE:TEM:NOIT-KRE:CNN:NOIT   -5.000000e-01 -2.05276818  1.05276818
## KRD:TEM:NOIT-None:TEM:NOIT  -5.000000e-01 -2.05276818  1.05276818
## KRE:TEM:NOIT-None:TEM:NOIT  -2.000000e+00 -3.55276818 -0.44723182
## KRE:TEM:NOIT-KRD:TEM:NOIT   -1.500000e+00 -3.05276818  0.05276818
##                                 p adj
## KRD:CNN:IT-None:CNN:IT      0.8629391
## KRE:CNN:IT-None:CNN:IT      0.5285663
## None:TEM:IT-None:CNN:IT     0.9999862
## KRD:TEM:IT-None:CNN:IT      0.9913907
## KRE:TEM:IT-None:CNN:IT      0.9913907
## None:CNN:NOIT-None:CNN:IT   0.0662756
## KRD:CNN:NOIT-None:CNN:IT    0.9999862
## KRE:CNN:NOIT-None:CNN:IT    0.9999862
## None:TEM:NOIT-None:CNN:IT   0.2183862
## KRD:TEM:NOIT-None:CNN:IT    0.8629391
## KRE:TEM:NOIT-None:CNN:IT    0.8629391
## KRE:CNN:IT-KRD:CNN:IT       0.9999862
## None:TEM:IT-KRD:CNN:IT      0.9913907
## KRD:TEM:IT-KRD:CNN:IT       0.9999862
## KRE:TEM:IT-KRD:CNN:IT       0.9999862
## None:CNN:NOIT-KRD:CNN:IT    0.0006797
## KRD:CNN:NOIT-KRD:CNN:IT     0.9913907
## KRE:CNN:NOIT-KRD:CNN:IT     0.9913907
## None:TEM:NOIT-KRD:CNN:IT    0.0034676
## KRD:TEM:NOIT-KRD:CNN:IT     0.0662756
## KRE:TEM:NOIT-KRD:CNN:IT     1.0000000
```

```
## None:TEM:IT-KRE:CNN:IT      0.8629391
## KRD:TEM:IT-KRE:CNN:IT       0.9913907
## KRE:TEM:IT-KRE:CNN:IT       0.9913907
## None:CNN:NOIT-KRE:CNN:IT    0.0001272
## KRD:CNN:NOIT-KRE:CNN:IT     0.8629391
## KRE:CNN:NOIT-KRE:CNN:IT     0.8629391
## None:TEM:NOIT-KRE:CNN:IT    0.0006797
## KRD:TEM:NOIT-KRE:CNN:IT     0.0162500
## KRE:TEM:NOIT-KRE:CNN:IT     0.9999862
## KRD:TEM:IT-None:TEM:IT      0.9999862
## KRE:TEM:IT-None:TEM:IT      0.9999862
## None:CNN:NOIT-None:TEM:IT   0.0162500
## KRD:CNN:NOIT-None:TEM:IT    1.0000000
## KRE:CNN:NOIT-None:TEM:IT    1.0000000
## None:TEM:NOIT-None:TEM:IT   0.0662756
## KRD:TEM:NOIT-None:TEM:IT    0.5285663
## KRE:TEM:NOIT-None:TEM:IT    0.9913907
## KRE:TEM:IT-KRD:TEM:IT       1.0000000
## None:CNN:NOIT-KRD:TEM:IT    0.0034676
## KRD:CNN:NOIT-KRD:TEM:IT     0.9999862
## KRE:CNN:NOIT-KRD:TEM:IT     0.9999862
## None:TEM:NOIT-KRD:TEM:IT    0.0162500
## KRD:TEM:NOIT-KRD:TEM:IT     0.2183862
## KRE:TEM:NOIT-KRD:TEM:IT     0.9999862
## None:CNN:NOIT-KRE:TEM:IT    0.0034676
## KRD:CNN:NOIT-KRE:TEM:IT     0.9999862
## KRE:CNN:NOIT-KRE:TEM:IT     0.9999862
## None:TEM:NOIT-KRE:TEM:IT    0.0162500
## KRD:TEM:NOIT-KRE:TEM:IT     0.2183862
## KRE:TEM:NOIT-KRE:TEM:IT     0.9999862
## KRD:CNN:NOIT-None:CNN:NOIT  0.0162500
## KRE:CNN:NOIT-None:CNN:NOIT  0.0162500
## None:TEM:NOIT-None:CNN:NOIT 0.9999862
## KRD:TEM:NOIT-None:CNN:NOIT  0.8629391
## KRE:TEM:NOIT-None:CNN:NOIT  0.0006797
## KRE:CNN:NOIT-KRD:CNN:NOIT   1.0000000
## None:TEM:NOIT-KRD:CNN:NOIT  0.0662756
## KRD:TEM:NOIT-KRD:CNN:NOIT   0.5285663
## KRE:TEM:NOIT-KRD:CNN:NOIT   0.9913907
## None:TEM:NOIT-KRE:CNN:NOIT  0.0662756
## KRD:TEM:NOIT-KRE:CNN:NOIT   0.5285663
## KRE:TEM:NOIT-KRE:CNN:NOIT   0.9913907
## KRD:TEM:NOIT-None:TEM:NOIT  0.9913907
## KRE:TEM:NOIT-None:TEM:NOIT  0.0034676
## KRE:TEM:NOIT-KRD:TEM:NOIT   0.0662756
```

Calculate errors statistics by relevant factors: solution and expertise.

```r
# Calculate using group_by_ function from dplyr
errorsStats <- group_by(experiments, Expertise, Solution) %>%
    summarise(count=n(), mean = mean(Errors, na.rm = TRUE),
              sd = sd(Errors, na.rm = TRUE))
# Visualise tabulated result
kable(errorsStats, format = "latex", booktabs = TRUE)
```

Expertise	Solution	count	mean	sd
IT	None	8	0.875	0.6408699
IT	KRD	8	0.375	0.5175492
IT	KRE	8	0.250	0.4629100
NOIT	None	8	2.375	0.5175492
NOIT	KRD	8	1.250	0.8864053
NOIT	KRE	8	0.500	0.7559289

Stop watch time study

Exploratory analysis

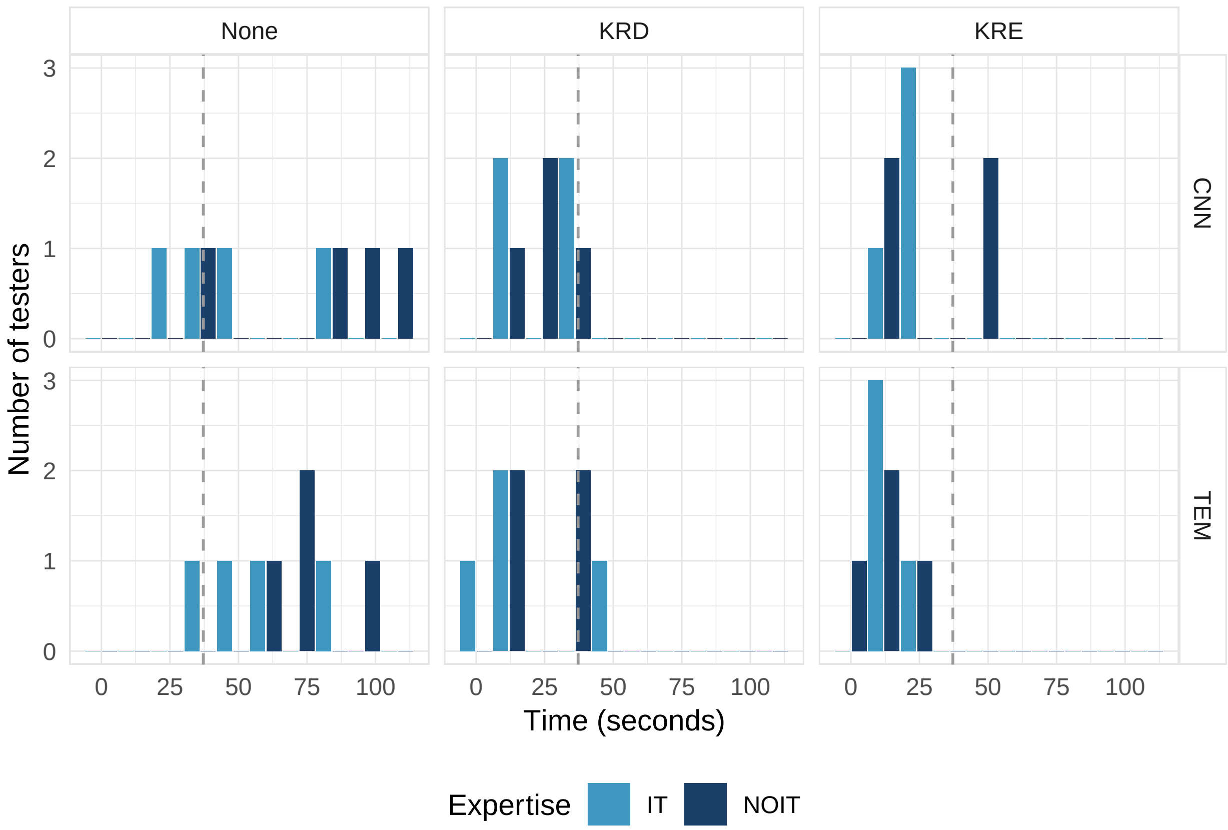
Calculate seconds statistics by solution.

```
# Calculate using group_by_ function from dplyr
secondsStatsSolution <- group_by(experiments, Solution) %>%
    summarise(count=n(), mean = mean(Seconds, na.rm = TRUE),
              sd = sd(Seconds, na.rm = TRUE))
# Visualise tabulated result
kable(secondsStatsSolution, format = "latex", booktabs = TRUE)
```

Solution	count	mean	sd
None	16	66.0000	25.63071
KRD	16	25.6250	13.57878
KRE	16	19.9375	12.55106

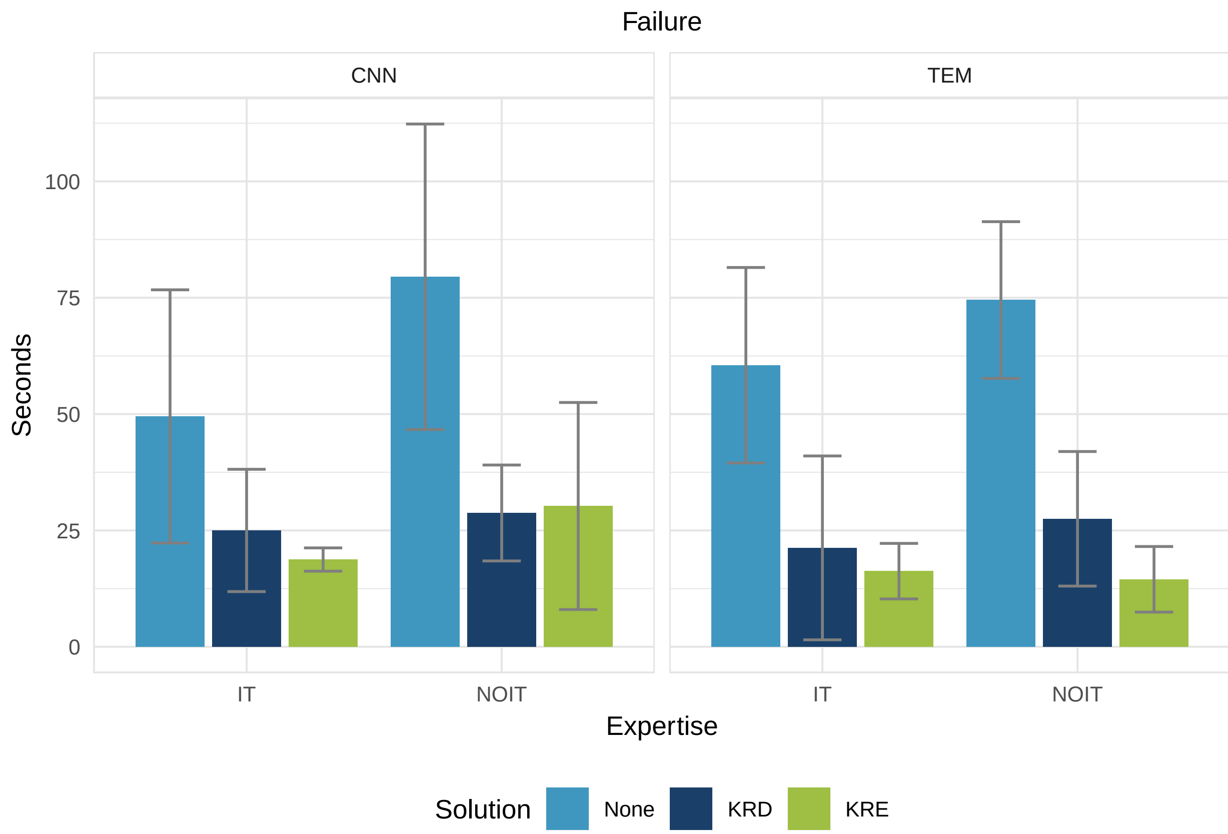
Calculate seconds histogram with mean by failure, solution and expertise

```
# Plot histogram using abovementioned rationale and prepared theme
secondsPlotHist <- ggplot(experiments, aes(x = Seconds, fill = Expertise)) +
    geom_histogram(binwidth = 12, position = position_dodge2(preserve = "single")) +
    geom_vline(xintercept = 37.19, linetype = "dashed", color = "gray60") +
    facet_grid(Failure ~ Solution) +
    scale_fill_manual(values=c06Palette) +
    scale_y_continuous("Number of testers", limits = c(0,3),
                       breaks = c(0,1,2,3)) +
    scale_x_continuous("Time (seconds)", breaks = c(0,25,50,75,100)) +
    theme(legend.position = "bottom",
          panel.background = element_rect(colour = "gray90", fill = "white"),
          panel.grid.major = element_line(colour = "gray90", size = 0.25),
          panel.grid.minor = element_line(colour = "gray90", size = 0.125),
          axis.ticks = element_blank(),
          strip.background = element_rect(colour = "gray90", fill = "white"),
          text = element_text(size = 11))
# Visualise plot
secondsPlotHist
```



Plot seconds statistics by solution, failure and expertise.

```
# Plot box and whiskers using abovementioned rationale and prepared theme
secondsPlotStats <-
  ggplot(experiments, aes(x = Expertise, y = Seconds, fill = Solution)) +
  geom_bar(stat = "summary", fun.y = "mean",
           position = position_dodge2(preserve = "single")) +
  geom_errorbar(stat = "summary", fun.data = "mean_sd",
                position = position_dodge2(width = 0.5, padding = 0.5),
                colour = "gray50", lwd = 0.5) +
  facet_grid(. ~ Failure, scales = "free_x") +
  scale_y_continuous(breaks = c(0,25,50,75,100)) +
  scale_fill_manual(values = c06Palette) +
  labs(y = "Seconds", title = "Failure") +
  theme (legend.position = "bottom",
         plot.title = element_text(size = 10, hjust = 0.5)) +
  plotTheme
# Visualise plot
secondsPlotStats
```



Evaluate seconds variability with expertise using t-test.

```
t.test(Seconds ~ Expertise, data = experiments)
```

```
##
##  Welch Two Sample t-test
##
## data:  Seconds by Expertise
## t = -1.3576, df = 42.67, p-value = 0.1817
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -26.41179   5.16179
## sample estimates:
##   mean in group IT mean in group NOIT
##             31.875             42.500
```

Evaluate seconds variability with fature using t-test.

```
t.test(Seconds ~ Failure, data = experiments)
```

```
##
##  Welch Two Sample t-test
##
## data:  Seconds by Failure
## t = 0.36071, df = 45.975, p-value = 0.72
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -13.16861  18.91861
## sample estimates:
```

```
## mean in group CNN mean in group TEM
##            38.625            35.750
```

Evaluate errors variability with solution using one-way anova.

```r
summary(aov(Seconds ~ Solution, data = experiments))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Solution     2  20183   10091   30.31 4.6e-09 ***
## Residuals   45  14983     333
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Only one effect (solution) seem relevant to seconds (interaction) variability. However, it seems reasonable to perform a multi-variate analysis and test its assumptions.

Correlation assumptions testing

Prepare seconds data for multivariate analysis. Remove outliers from outside quartiles using box and whiskers plot. Fit clean data to linear model for conducting multi-variate anova in expertise, solution and failure. Calculate residuals, predicted values and squared predicted values for assumptions testing.

```r
# Calculate outliers using box and whiskers plot
subset(experiments,experiments$Seconds %in% boxplot.stats(experiments$Seconds)$out)
```

```
##    Tester Failure Solution Expertise Seconds Errors
## 30     35     CNN     None      NOIT     105      3
```

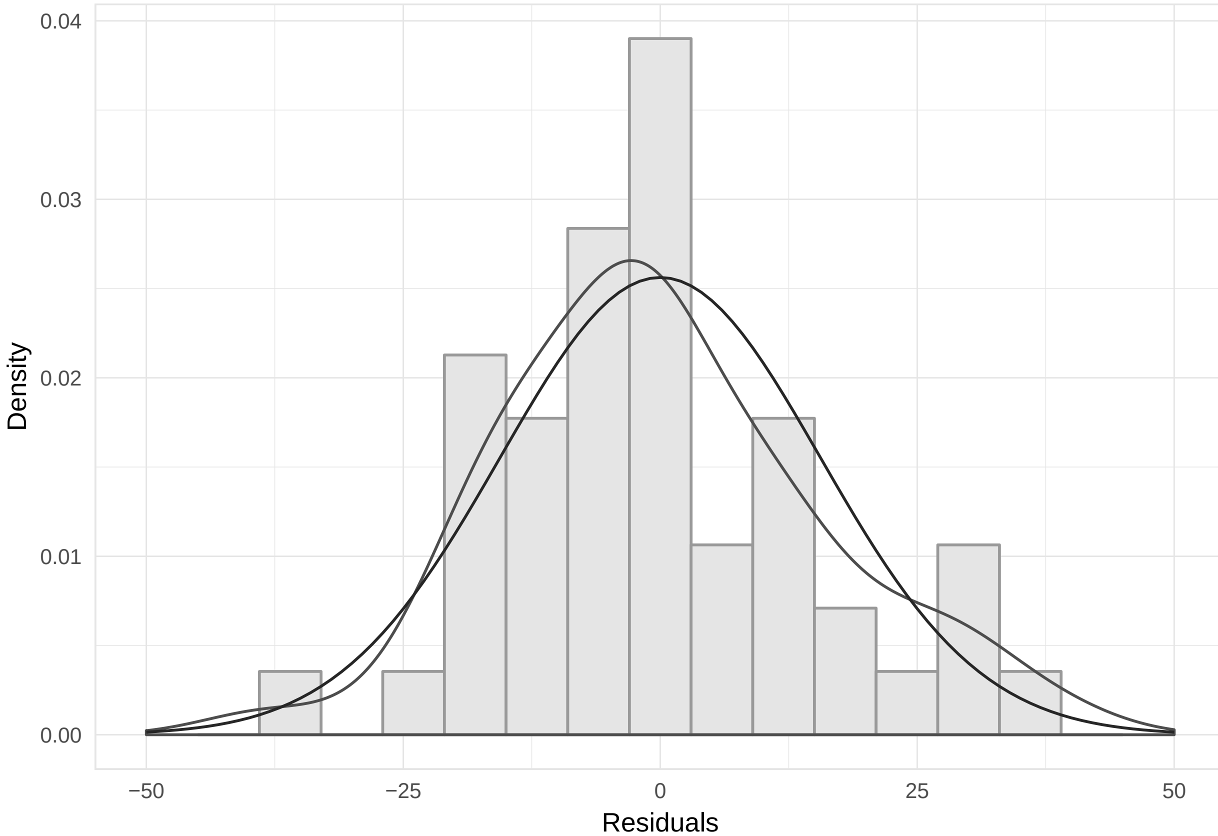
```r
# One outlier found (Tester 35)
seconds <- subset(experiments, !(experiments$Tester %in% c(35)))
# Subset data removing non-relevant variables
seconds <- within(seconds, rm(Errors))
# Fit data to a linear model to run anova considering interactions of effects
secondsLM <- lm(Seconds ~ Solution*Failure*Expertise, seconds)
# Calculate residuals, predicted values and squared predicted values
seconds$Residuals <- residuals(secondsLM)
seconds$Predicted <- predict(secondsLM)
seconds$SqrdPredicted <- predict(secondsLM)^2
```

Plot seconds as histogram and seconds residuals as line to evaluate normality. Include line of normal distribution for further comparison.

```r
# Plot residuals histogram and overlay normal distribution shape
secondsPlotNorm <- ggplot(seconds) +
  geom_histogram(aes(x = Residuals, y = ..density..), binwidth = 6,
                 fill = "gray90", colour = "gray60") +
  geom_density(aes(x = Residuals, y = ..density..), colour = "gray30") +
  stat_function(fun = function(x,mean,sd,n){
    dnorm(x = x, mean = mean, sd = sd)
  }, args = with(seconds,
                 c(mean = mean(Residuals), sd = sd(Residuals),
                   n = length(Residuals))), colour = "gray15") +
  xlim(-50, 50) + scale_y_continuous("Density") +
  theme(panel.background = element_rect(colour = "gray90", fill = "white"),
        panel.grid.major = element_line(colour = "gray90", size = 0.25),
        panel.grid.minor = element_line(colour = "gray90", size = 0.125),
        axis.ticks = element_blank(),
```

```
       strip.background = element_rect(colour = "gray90", fill = "white"),
       text = element_text(size = 10))
# Visualise plot
secondsPlotNorm
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



Calculate Shapiro test to evaluate normality.

```
# Run shapiro test to evaluate normality
shapiro.test(seconds$Residuals)
```

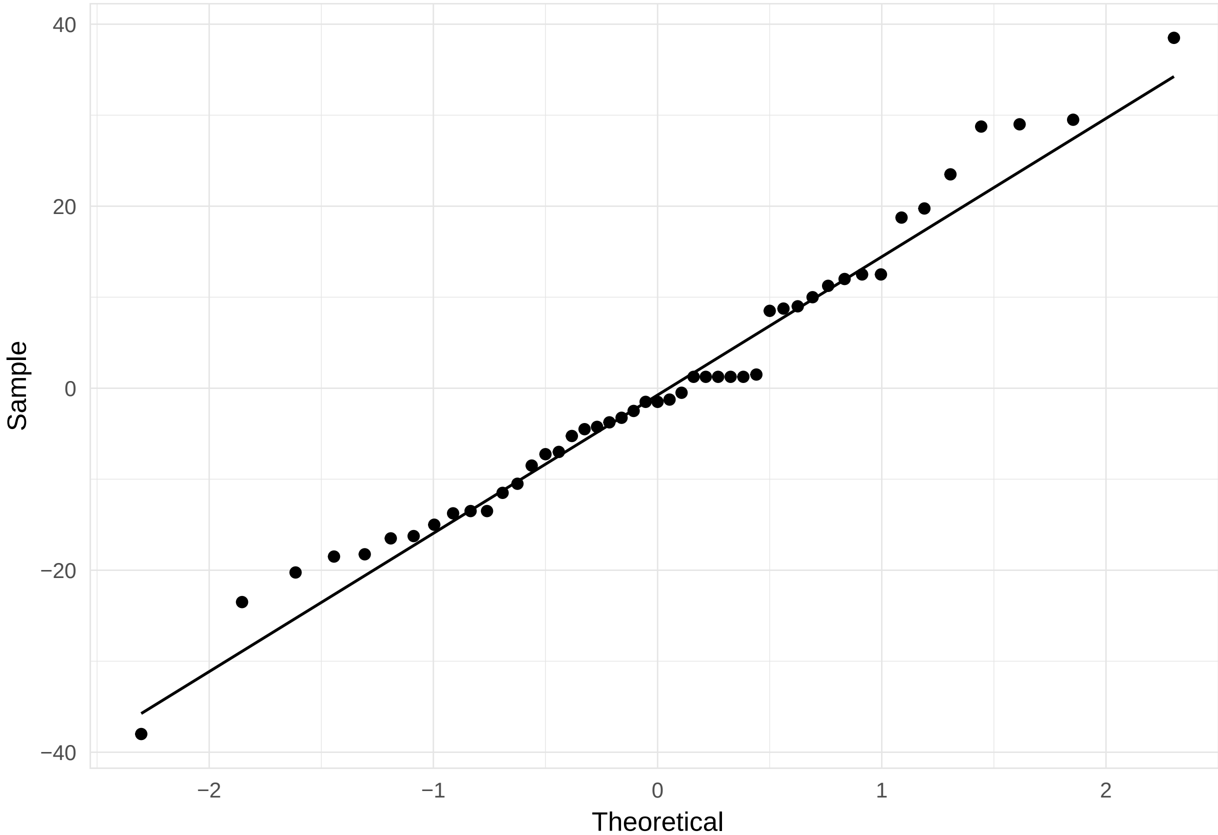
```
##
##  Shapiro-Wilk normality test
##
## data:  seconds$Residuals
## W = 0.97707, p-value = 0.4772
```

```
# Data is distributed normally if p-value is below 0.05
```

Plot seconds lm qq line to evaluate linearity.

```
# Plot residuals using ggplot stats
secondsPlotLin <- ggplot(seconds, aes(sample = Residuals)) +
  stat_qq() + stat_qq_line() +
  scale_x_continuous("Theoretical") + scale_y_continuous("Sample") +
  theme(panel.background = element_rect(colour = "gray90", fill = "white"),
        panel.grid.major = element_line(colour = "gray90", size = 0.25),
        panel.grid.minor = element_line(colour = "gray90", size = 0.125),
        axis.ticks = element_blank(),
```

```r
        strip.background = element_rect(colour = "gray90", fill = "white"),
        text = element_text(size = 10))
# Visualise plot
secondsPlotLin
```



Calculate Bartlett test to evaluate homogeinity.

```r
# Run Bartlett test to evaluate homogeinity
bartlett.test(Seconds ~ interaction(Solution, Failure, Expertise), data = seconds)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  Seconds by interaction(Solution, Failure, Expertise)
## Bartlett's K-squared = 20.118, df = 11, p-value = 0.04374
```

Correlation analysis

Calculate three-way anova for effects (solution, failure and expertise) and interaction (seconds) multi-variate analysis. Tabulate summarised test results.

```r
# Run anova test on cleaned results with outliers removed
secondsAnova <- aov(Seconds ~ Solution*Failure*Expertise, seconds)
# Visualise tabulated results
summary(secondsAnova)
```

```
##                             Df Sum Sq Mean Sq F value   Pr(>F)
## Solution                     2  17109    8554  26.846 8.58e-08 ***
```

```
## Failure                    1     16      16   0.050   0.8240
## Expertise                  1    976     976   3.064   0.0888 .
## Solution:Failure           2    583     292   0.915   0.4097
## Solution:Expertise         2    398     199   0.624   0.5415
## Failure:Expertise          1    107     107   0.335   0.5662
## Solution:Failure:Expertise  2    127      63   0.199   0.8204
## Residuals                 35  11153     319
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Calculate Tukey HDS test results to post-hoc compare groups means.

```r
# Run post-hoc pairwise t-test comparisons
TukeyHSD(aov(Seconds ~ Solution:Failure:Expertise, seconds))
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Seconds ~ Solution:Failure:Expertise, data = seconds)
##
## $`Solution:Failure:Expertise`
##                              diff         lwr         upr     p adj
## KRD:CNN:IT-None:CNN:IT      -24.50  -68.6384319  19.6384319 0.7268635
## KRE:CNN:IT-None:CNN:IT      -30.75  -74.8884319  13.3884319 0.4100527
## None:TEM:IT-None:CNN:IT      11.00  -33.1384319  55.1384319 0.9990222
## KRD:TEM:IT-None:CNN:IT      -28.25  -72.3884319  15.8884319 0.5352772
## KRE:TEM:IT-None:CNN:IT      -33.25  -77.3884319  10.8884319 0.2999397
## None:CNN:NOIT-None:CNN:IT    21.50  -26.1749553  69.1749553 0.9059860
## KRD:CNN:NOIT-None:CNN:IT    -20.75  -64.8884319  23.3884319 0.8804830
## KRE:CNN:NOIT-None:CNN:IT    -19.25  -63.3884319  24.8884319 0.9232110
## None:TEM:NOIT-None:CNN:IT    25.00  -19.1384319  69.1384319 0.7024893
## KRD:TEM:NOIT-None:CNN:IT    -22.00  -66.1384319  22.1384319 0.8361131
## KRE:TEM:NOIT-None:CNN:IT    -35.00  -79.1384319   9.1384319 0.2349812
## KRE:CNN:IT-KRD:CNN:IT        -6.25  -50.3884319  37.8884319 0.9999962
## None:TEM:IT-KRD:CNN:IT       35.50   -8.6384319  79.6384319 0.2183766
## KRD:TEM:IT-KRD:CNN:IT        -3.75  -47.8884319  40.3884319 1.0000000
## KRE:TEM:IT-KRD:CNN:IT        -8.75  -52.8884319  35.3884319 0.9998871
## None:CNN:NOIT-KRD:CNN:IT     46.00   -1.6749553  93.6749553 0.0668120
## KRD:CNN:NOIT-KRD:CNN:IT       3.75  -40.3884319  47.8884319 1.0000000
## KRE:CNN:NOIT-KRD:CNN:IT       5.25  -38.8884319  49.3884319 0.9999994
## None:TEM:NOIT-KRD:CNN:IT     49.50    5.3615681  93.6384319 0.0172060
## KRD:TEM:NOIT-KRD:CNN:IT       2.50  -41.6384319  46.6384319 1.0000000
## KRE:TEM:NOIT-KRD:CNN:IT     -10.50  -54.6384319  33.6384319 0.9993615
## None:TEM:IT-KRE:CNN:IT       41.75   -2.3884319  85.8884319 0.0778210
## KRD:TEM:IT-KRE:CNN:IT         2.50  -41.6384319  46.6384319 1.0000000
## KRE:TEM:IT-KRE:CNN:IT        -2.50  -46.6384319  41.6384319 1.0000000
## None:CNN:NOIT-KRE:CNN:IT     52.25    4.5750447  99.9249553 0.0216853
## KRD:CNN:NOIT-KRE:CNN:IT      10.00  -34.1384319  54.1384319 0.9995946
## KRE:CNN:NOIT-KRE:CNN:IT      11.50  -32.6384319  55.6384319 0.9985409
## None:TEM:NOIT-KRE:CNN:IT     55.75   11.6115681  99.8884319 0.0045066
## KRD:TEM:NOIT-KRE:CNN:IT       8.75  -35.3884319  52.8884319 0.9998871
## KRE:TEM:NOIT-KRE:CNN:IT      -4.25  -48.3884319  39.8884319 0.9999999
## KRD:TEM:IT-None:TEM:IT      -39.25  -83.3884319   4.8884319 0.1204206
## KRE:TEM:IT-None:TEM:IT      -44.25  -88.3884319  -0.1115681 0.0489505
## None:CNN:NOIT-None:TEM:IT    10.50  -37.1749553  58.1749553 0.9996895
```

```
## KRD:CNN:NOIT-None:TEM:IT    -31.75  -75.8884319  12.3884319 0.3637682
## KRE:CNN:NOIT-None:TEM:IT    -30.25  -74.3884319  13.8884319 0.4341589
## None:TEM:NOIT-None:TEM:IT    14.00  -30.1384319  58.1384319 0.9921991
## KRD:TEM:NOIT-None:TEM:IT    -33.00  -77.1384319  11.1384319 0.3100740
## KRE:TEM:NOIT-None:TEM:IT    -46.00  -90.1384319  -1.8615681 0.0348901
## KRE:TEM:IT-KRD:TEM:IT        -5.00  -49.1384319  39.1384319 0.9999996
## None:CNN:NOIT-KRD:TEM:IT     49.75    2.0750447  97.4249553 0.0344840
## KRD:CNN:NOIT-KRD:TEM:IT       7.50  -36.6384319  51.6384319 0.9999755
## KRE:CNN:NOIT-KRD:TEM:IT       9.00  -35.1384319  53.1384319 0.9998515
## None:TEM:NOIT-KRD:TEM:IT     53.25    9.1115681  97.3884319 0.0077793
## KRD:TEM:NOIT-KRD:TEM:IT       6.25  -37.8884319  50.3884319 0.9999962
## KRE:TEM:NOIT-KRD:TEM:IT      -6.75  -50.8884319  37.3884319 0.9999916
## None:CNN:NOIT-KRE:TEM:IT     54.75    7.0750447 102.4249553 0.0134275
## KRD:CNN:NOIT-KRE:TEM:IT      12.50  -31.6384319  56.6384319 0.9969701
## KRE:CNN:NOIT-KRE:TEM:IT      14.00  -30.1384319  58.1384319 0.9921991
## None:TEM:NOIT-KRE:TEM:IT     58.25   14.1115681 102.3884319 0.0025827
## KRD:TEM:NOIT-KRE:TEM:IT      11.25  -32.8884319  55.3884319 0.9988018
## KRE:TEM:NOIT-KRE:TEM:IT      -1.75  -45.8884319  42.3884319 1.0000000
## KRD:CNN:NOIT-None:CNN:NOIT  -42.25  -89.9249553   5.4249553 0.1231703
## KRE:CNN:NOIT-None:CNN:NOIT  -40.75  -88.4249553   6.9249553 0.1547740
## None:TEM:NOIT-None:CNN:NOIT   3.50  -44.1749553  51.1749553 1.0000000
## KRD:TEM:NOIT-None:CNN:NOIT  -43.50  -91.1749553   4.1749553 0.1010717
## KRE:TEM:NOIT-None:CNN:NOIT  -56.50 -104.1749553  -8.8250447 0.0095234
## KRE:CNN:NOIT-KRD:CNN:NOIT     1.50  -42.6384319  45.6384319 1.0000000
## None:TEM:NOIT-KRD:CNN:NOIT   45.75    1.6115681  89.8884319 0.0366433
## KRD:TEM:NOIT-KRD:CNN:NOIT    -1.25  -45.3884319  42.8884319 1.0000000
## KRE:TEM:NOIT-KRD:CNN:NOIT   -14.25  -58.3884319  29.8884319 0.9910068
## None:TEM:NOIT-KRE:CNN:NOIT   44.25    0.1115681  88.3884319 0.0489505
## KRD:TEM:NOIT-KRE:CNN:NOIT    -2.75  -46.8884319  41.3884319 1.0000000
## KRE:TEM:NOIT-KRE:CNN:NOIT   -15.75  -59.8884319  28.3884319 0.9805089
## KRD:TEM:NOIT-None:TEM:NOIT  -47.00  -91.1384319  -2.8615681 0.0286179
## KRE:TEM:NOIT-None:TEM:NOIT  -60.00 -104.1384319 -15.8615681 0.0017397
## KRE:TEM:NOIT-KRD:TEM:NOIT   -13.00  -57.1384319  31.1384319 0.9957693
```

Calculate seconds statistics by relevant factors: expertise and solution.

```
# Calculate using group_by_ function from dplyr
secondsStats <- group_by(experiments, Expertise, Solution) %>%
    summarise(count=n(), mean = mean(Seconds, na.rm = TRUE),
              sd = sd(Seconds, na.rm = TRUE))
# Visualise tabulated result
kable(secondsStats, format = "latex", booktabs = TRUE)
```

Expertise	Solution	count	mean	sd
IT	None	8	55.000	23.250192
IT	KRD	8	23.125	15.661030
IT	KRE	8	17.500	4.440077
NOIT	None	8	77.000	24.301675
NOIT	KRD	8	28.125	11.642748
NOIT	KRE	8	22.375	17.443275

Monitoring usability surveys analysis

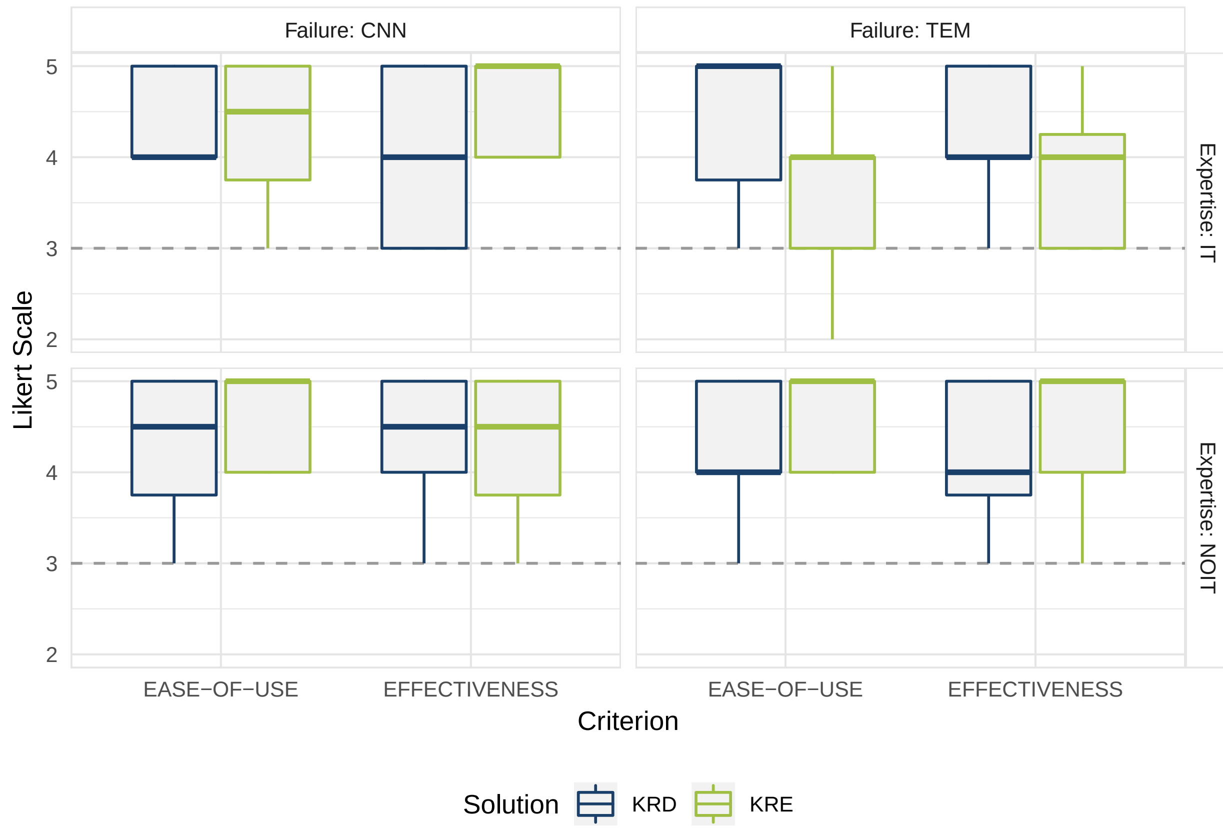
Calculate usability statistics and present survey responses.

```
# Calculate using group_by_ function from dplyr
surveysStats <- group_by(surveys, Expertise, Failure, Criterion, Solution) %>%
    summarise(count=n(), mean = mean(Response,na.rm = TRUE),
              sd = sd(Response,na.rm = TRUE))
# Visualise tabulated result
kable(surveysStats, format = "latex", booktabs = TRUE)
```

Expertise	Failure	Criterion	Solution	count	mean	sd
IT	CNN	EASE-OF-USE	KRD	12	4.416667	0.5149287
IT	CNN	EASE-OF-USE	KRE	12	4.250000	0.8660254
IT	CNN	EFFECTIVENESS	KRD	12	4.000000	0.9534626
IT	CNN	EFFECTIVENESS	KRE	12	4.666667	0.4923660
IT	TEM	EASE-OF-USE	KRD	12	4.416667	0.9003366
IT	TEM	EASE-OF-USE	KRE	12	3.666667	0.7784989
IT	TEM	EFFECTIVENESS	KRD	12	4.250000	0.7537784
IT	TEM	EFFECTIVENESS	KRE	12	3.916667	0.7929615
NOIT	CNN	EASE-OF-USE	KRD	12	4.250000	0.8660254
NOIT	CNN	EASE-OF-USE	KRE	12	4.583333	0.5149287
NOIT	CNN	EFFECTIVENESS	KRD	12	4.416667	0.6685579
NOIT	CNN	EFFECTIVENESS	KRE	12	4.250000	0.8660254
NOIT	TEM	EASE-OF-USE	KRD	12	4.333333	0.6513389
NOIT	TEM	EASE-OF-USE	KRE	12	4.583333	0.5149287
NOIT	TEM	EFFECTIVENESS	KRD	12	4.083333	0.7929615
NOIT	TEM	EFFECTIVENESS	KRE	12	4.416667	0.7929615

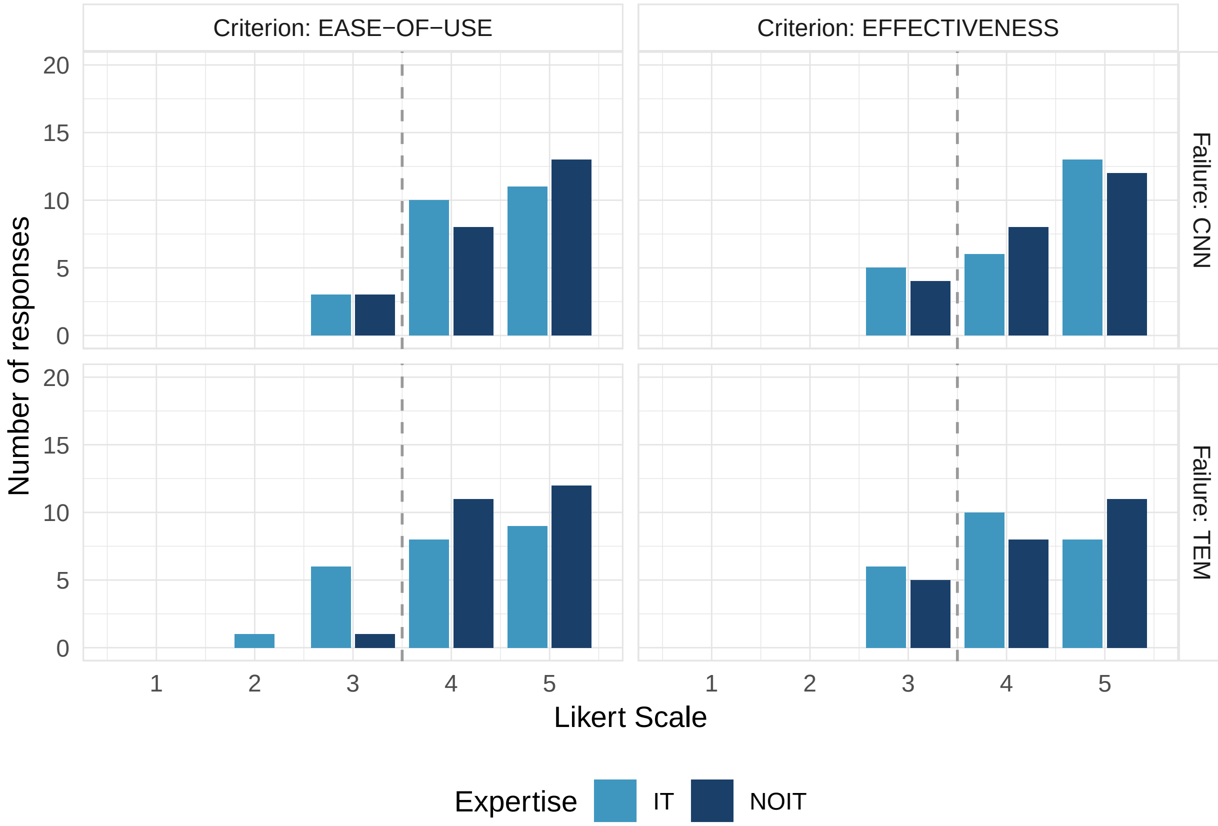
Plot usability statistics by criterion, failure and expertise.

```
# Plot box and whiskers using abovementioned rationale and prepared theme
surveysPlotStats <-
  ggplot(surveys, aes(x = Criterion, y = Response,
                      colour = Solution, label = Tester)) +
  geom_hline(yintercept = 3, linetype = "dashed", colour = "gray60") +
  geom_boxplot(fill = "gray95", lwd = 0.5) +
  facet_grid(Expertise ~ Failure, labeller = label_both) +
  scale_colour_manual(values = c("#1A406A","#9EBF43")) +
  scale_y_continuous(breaks = c(1,2,3,4,5)) +
  labs(y = "Likert Scale") +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, size = 10)) +
  plotTheme
# Visualise plot
surveysPlotStats
```



Plot responses count by criterion, failure and expertise.

```
# Plot bars using abovementioned rationale and prepared theme
surveysPlotCount <- ggplot(surveys, aes(x = Response, fill = Expertise)) +
    geom_bar(position = position_dodge2(preserve = "single"), width = 0.9) +
    geom_vline(xintercept = 3.5, linetype = "dashed", color = "gray60") +
    facet_grid(Failure ~ Criterion, labeller = label_both) +
    scale_fill_manual(values=c06Palette) +
    scale_y_continuous("Number of responses", limits = c(0,20),
                       breaks = c(0,5,10,15,20)) +
    scale_x_continuous("Likert Scale", limits = c(0.5,5.5),
                       breaks = c(1,2,3,4,5)) +
    theme(legend.position = "bottom",
          panel.background = element_rect(colour = "gray90", fill = "white"),
          panel.grid.major = element_line(colour = "gray90", size = 0.25),
          panel.grid.minor = element_line(colour = "gray90", size = 0.125),
          axis.ticks = element_blank(),
          strip.background = element_rect(colour = "gray90", fill = "white"),
          text = element_text(size = 11))
# Visualise plot
surveysPlotCount
```


Criterion: EASE−OF−USE
Criterion: EFFECTIVENESS
Failure: CNN
Failure: TEM
Number of responses
0
5
10
15
20
1
2
3
4
5
Likert Scale
Expertise
IT
NOIT