Experimental Design R Cheatsheet

Contents

Format

This cheatsheet includes R recipes that are useful for experimental design. The general format is,

`function_name`

package (if necessary)

Brief description

```
# code_example
```

I welcome you to leave comments or propose changes to the document on github, using either an issue or pull request.

Chapter 2

mean

Computes the sample mean $\frac{1}{n}\sum_{i=1}^{n} x_i$ of a vector of n numeric values.

```
mean(1:10)
```

```
## [1] 5.5
```

sd

Computes the sample standard deviation $\frac{1}{n-1}\sum_{i=1}^{n} (x_i - \bar{x})^2$ of a vector of n numeric values.

```
sd(1:10)
```

```
## [1] 3.02765
```

rnorm, rt, rchisq, rf

Simulates n samples from a normal, t, chi-square, or F-distribution, respectively. The parameters of the normal are the mean and standard deviation. The other distributions require degrees-of-freedom parameters.

```
n <- 5
rnorm(n, 0, 1)
```

```
## [1] -0.04286579  0.29608978 -0.98753393  1.17600803  1.07471170
```

```
rt(n, 1)
```

```
## [1] -0.44642988 -2.83558016 -2.61868713  0.43993619  0.02159252
```

```
rchisq(n, 1)
```

```
## [1] 0.9103231 0.7715337 0.5075819 1.9511023 1.7228036
```

```
rf(n, 1, 2)
```

```
## [1] 0.236679069 0.614783275 1.648223635 0.008637559 2.523934181
```

pnorm, pt, pchisq, pf

Computes the total probability of landing below x in a normal, t, chi-square, or F-distribution, respectively. Accepts the same parameters as the `r`-prefixed functions above.

```
x <- -1.96
pnorm(x, 0, 1)
```

```
## [1] 0.0249979
```

```
pt(x, 1)
```

```
## [1] 0.1501714
```

```
x <- 10
pchisq(x, 1)
```

```
## [1] 0.9984346
```

```r
pt(x, 1)
```

```
## [1] 0.9682745
```

```r
pf(x, 1, 2)
```

```
## [1] 0.9128709
```

qnorm, qt, qchisq, qf

Computes the x-coordinate such that a probability q of the distribution lies below x. Especially useful for computing cutoff values in hypothesis tests.

```r
q <- 0.05 / 2
qnorm(q)
```

```
## [1] -1.959964
```

```r
qt(q, 1)
```

```
## [1] -12.7062
```

```r
qchisq(q, 1)
```

```
## [1] 0.0009820691
```

```r
qf(q, 1, 2)
```

```
## [1] 0.001250782
```

dnorm, dt, dchisq, df

Computes the probability density of a distribution at a specific value x.

```r
x <- 2
dnorm(x)
```

```
## [1] 0.05399097
```

```r
dt(x, 1)
```

```
## [1] 0.06366198
```

```r
dchisq(x, 1)
```

```
## [1] 0.1037769
```

```r
df(x, 1, 2)
```

```
## [1] 0.08838835
```

z.test

library(BSDA)

Performs a two-sample z-test, testing for the difference in two means when the variance of both groups is known. Returns the estimate of the group means, the test statistic, a p-value, and a confidence interval (whose level is specified by the `conf.level` parameter).

```r
library(BSDA)
test_result <- z.test(rnorm(5), rnorm(5, 1), sigma.x = 1, sigma.y = 1)
test_result
```

```
##
##  Two-sample z-Test
##
## data:  rnorm(5) and rnorm(5, 1)
## z = -2.0567, p-value = 0.03971
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.54038827 -0.06120814
## sample estimates:
##  mean of x  mean of y
## -0.3363956  0.9644026
```

```
test_result$p.value # p.value
```

```
## [1] 0.039711
```

```
test_result$conf.int # confidence interval for the difference
```

```
## [1] -2.54038827 -0.06120814
## attr(,"conf.level")
## [1] 0.95
```

t.test

Performs a two-sample `t`-test, testing for the difference in two means when the variance of both groups are *unknown*. We typically assume equal variance (`var.equal = TRUE`) unless an initial / diagnostic analysis explicitly suggests to do otherwise. Returns the estimate of the group means, the test statistic, a p-value, and a confidence interval (whose level is specified by the `conf.level` parameter).

```
test_result <- t.test(rnorm(10), rnorm(10, 1), var.equal = TRUE)
test_result
```

```
##
##  Two Sample t-test
##
## data:  rnorm(10) and rnorm(10, 1)
## t = -2.0074, df = 18, p-value = 0.05997
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.01887489  0.04597652
## sample estimates:
## mean of x mean of y
## 0.1489373 1.1353864
```

```
test_result$p.value
```

```
## [1] 0.05996589
```

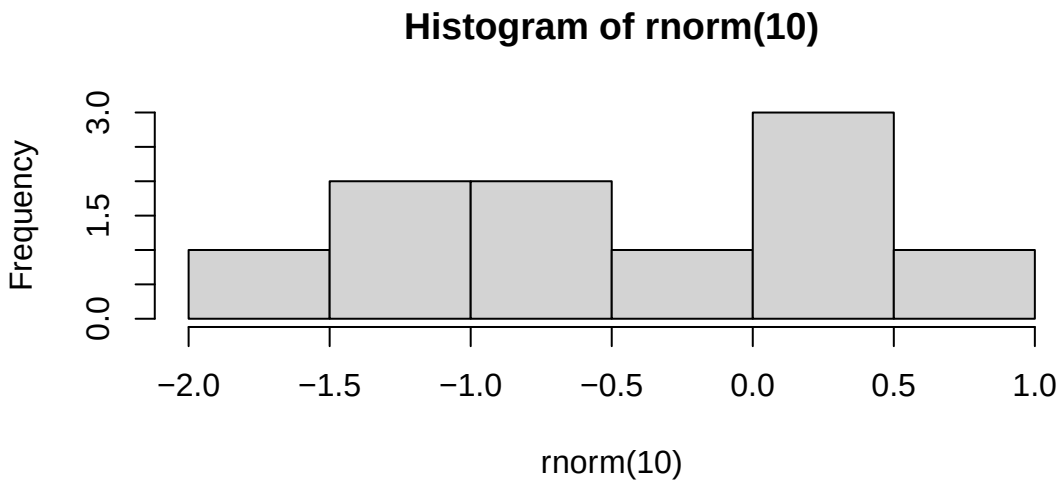
```
test_result$conf.int
```

```
## [1] -2.01887489  0.04597652
## attr(,"conf.level")
## [1] 0.95
```

hist

Built-in function to make histograms in R.

```
hist(rnorm(10))
```



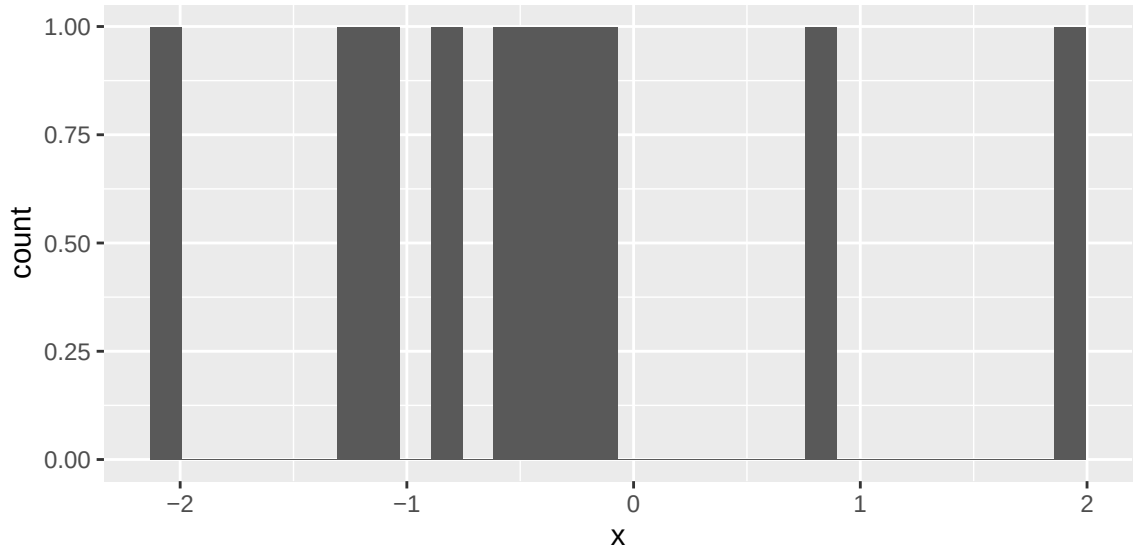
geom_histogram

library(ggplot2)

ggplot2 alternative to making histograms. The data must be input as a data.frame.

```
library(ggplot2)
df <- data.frame(x = rnorm(10))
ggplot(df) +
  geom_histogram(aes(x))
```

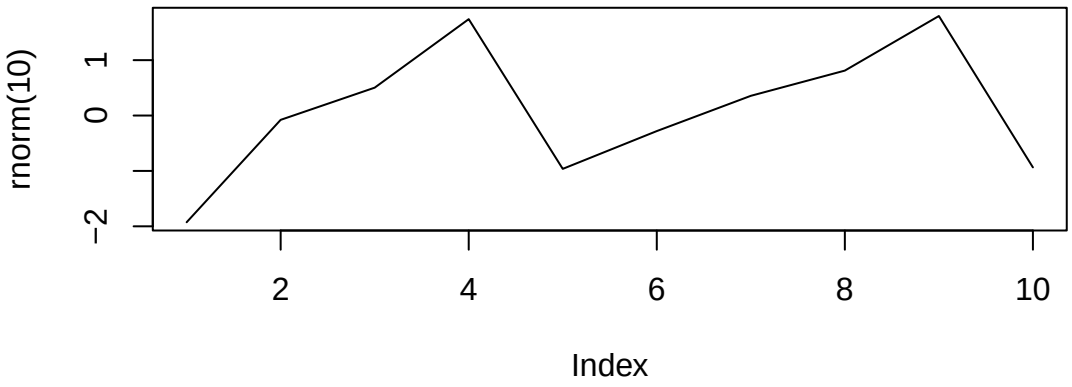
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



plot(x, type = "l")

Base R approach to generating a line plot.

```
plot(rnorm(10), type = "l")
```

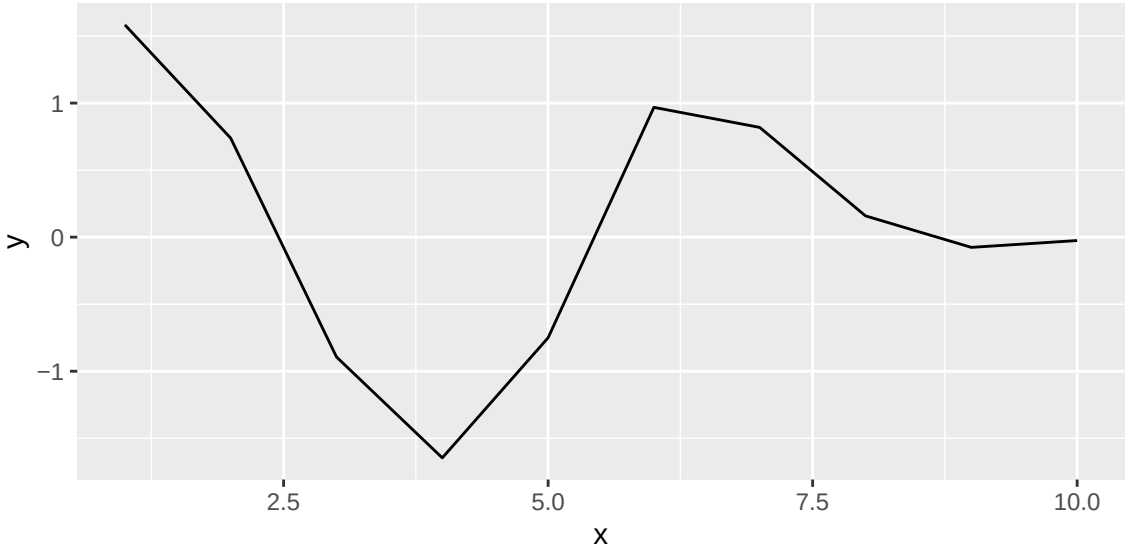


geom_line

library(ggplot2)

ggplot2 approach to generating a line plot. Expects a `data.frame` with at least two columns as input, used to specify the x and y coordinates.

```
library(ggplot2)
df <- data.frame(x = 1:10, y = rnorm(10))
ggplot(df) +
  geom_line(aes(x, y))
```



Chapter 3

lm and aov

Compute an ANOVA table describing how multiple levels of a factor are related to a response. Expects as input a `data.frame` whose columns include the response values and factor levels. Beware that if the levels are written as numeric values, then R will not treat each value as a separate group (use `as.factor` to convert into discrete groups). Note that the `anova()` function returns comparable output, but is not well-suited to contrast calculations.

```
experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
```

```
  y = rnorm(15)
)

fit <- lm(y ~ levels, data = experiment)
aov_table <- aov(fit)
summary(aov_table) # print the test results
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## levels       2  2.687   1.344   1.272  0.315
## Residuals   12 12.674   1.056
```

tidy

library(broom)

Convert the results from a base R test into a data.frame. Useful for extracting test results.

```
library(broom)
aov_df <- tidy(aov_table) # uses result from previous example
aov_df
```

```
## # A tibble: 2 x 6
##   term         df sumsq meansq statistic p.value
##   <chr>     <dbl> <dbl>  <dbl>     <dbl>   <dbl>
## 1 levels        2  2.69   1.34      1.27   0.315
## 2 Residuals    12 12.7    1.06     NA     NA
```

```
aov_df$p.value
```

```
## [1] 0.3154285        NA
```

resid

Extracts the residuals $e_{ij} = y_{ij} - \hat{y}_{ij}$ associated with a model. This is useful for checking model assumptions.

```
experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
  y = rnorm(15)
)

fit <- lm(y ~ levels, data = experiment)
resid(fit)
```

```
##           1           2           3           4           5           6           7           8
##  0.37989908 -0.78205704 -0.05579625 -0.98501890  1.44297311  1.05391099  0.04142371 -0.44936708 -1.58
```

predict

Extracts the fitted values $\hat{y}_{ij} = \hat{\mu} + \hat{\tau}_i$ associated with a model. Often useful when performing model checks.

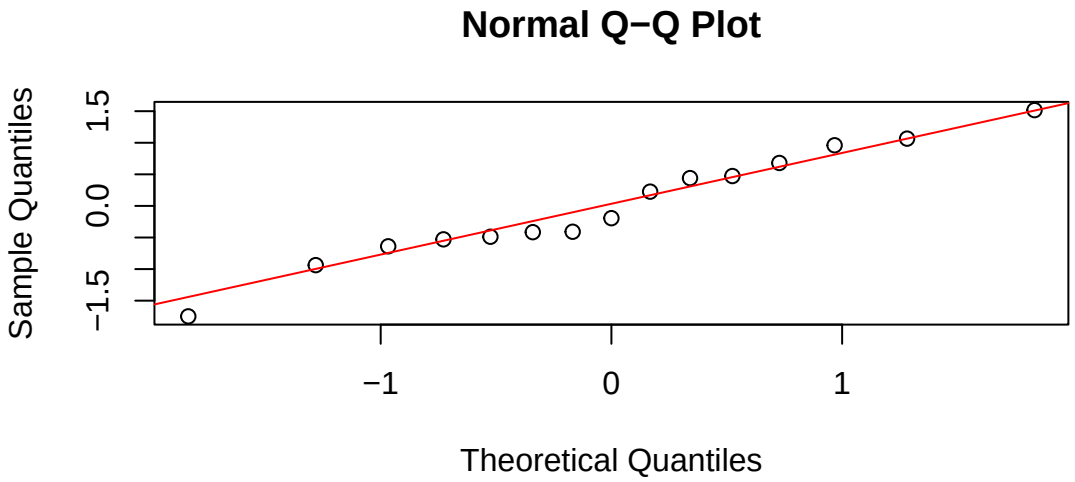
```
experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
  y = rnorm(15)
)

fit <- lm(y ~ levels, data = experiment)
predict(fit)
```

```
##         1         2         3         4         5         6         7         8         9
##  0.4718496  0.4718496  0.4718496  0.4718496  0.4718496 -0.7160472 -0.7160472 -0.7160472 -0.7160472 -0
```

qqnorm and qqline

Used to make QQ plots against a theoretical normal distribution. Usuall applied to the residuals of a fitted model.

```
qqnorm(resid(fit))
qqline(resid(fit), col = "red")
```



pivot_longer

library(tidyr)

Used to transform a "wide" dataset into a "tall" one. This is often useful for moving the replicates for an experimental factor into a single column, so that it can be used by the `lm` function.

```
library(tidyr)

experiment <- data.frame(
  levels = c("A", "B", "C"),
  rep1 = rnorm(3),
  rep2 = rnorm(3)
)
experiment_tall <- pivot_longer(experiment, -levels, names_to = "replicate")

experiment
```

```
##   levels       rep1       rep2
## 1      A -0.5236757 -0.1959504
## 2      B -0.1132412  1.1660793
## 3      C -0.4475183  1.1015888
```

```
experiment_tall
```

```
## # A tibble: 6 x 3
##   levels replicate  value
##   <chr>  <chr>      <dbl>
## 1 A      rep1      -0.524
## 2 A      rep2      -0.196
```

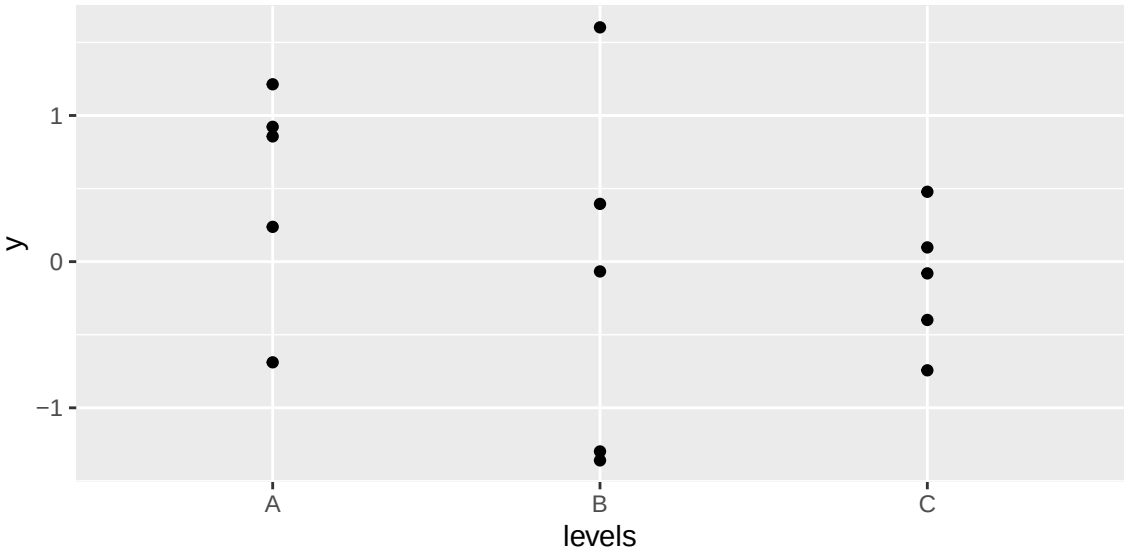
```
## 3 B      rep1      -0.113
## 4 B      rep2       1.17
## 5 C      rep1      -0.448
## 6 C      rep2       1.10
```

geom_point

library(ggplot2)

ggplot2 approach to drawing scatterplots. Often useful for plotting response values against a factor of interest.

```
experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
  y = rnorm(15)
)

ggplot(experiment) +
  geom_point(aes(levels, y))
```



fit.contrast

library(gmodels)

Conduct a hypothesis test for a specific contrast ($H_0 : \Gamma(c) = 0$ in an ANOVA model. Can be applied to multiple contrasts simultaneously by passing the vectors $c_1, \ldots, c_m$ as rows of a matrix.

```
library(gmodels)

experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
  y = rnorm(15)
)

# fit an anova model -- see entry on `aov` and `lm`
fit <- lm(y ~ levels, data = experiment)
aov_table <- aov(fit)
```

```r
# contrast between levels A and B
contrast <- c(1, -1, 0)
fit.contrast(aov_table, "levels", contrast)
```

```
##                     Estimate Std. Error  t value  Pr(>|t|)
## levels c=( 1 -1 0 ) 1.122082  0.4462606 2.514411 0.0271868
## attr(,"class")
## [1] "fit_contrast"
```

```r
# several contrasts
contrasts <- matrix(c(1, -1, 0, 1, 1, -1),  nrow = 2, byrow = TRUE)
fit.contrast(aov_table, "levels", contrasts)
```

```
##                       Estimate Std. Error    t value  Pr(>|t|)
## levels c=( 1 -1 0 )  1.1220825  0.4462606  2.5144106 0.0271868
## levels c=( 1 1 -1 ) -0.2799429  0.5797096 -0.4829019 0.6378554
## attr(,"class")
## [1] "fit_contrast"
```

`fit.contrast(..., conf.int = T)`

`library(gmodels)`

Compute the confidence interval for a contrast. Used in the exact same way as `fit.contrast`, but setting the `conf.int` parameter to the desired confidence level.

```r
# continues from previous example
fit.contrast(aov_table, "levels", contrast, conf.int = 0.95)
```

```
##                     Estimate Std. Error  t value  Pr(>|t|)  lower CI upper CI
## levels c=( 1 -1 0 ) 1.122082  0.4462606 2.514411 0.0271868 0.1497641 2.094401
## attr(,"class")
## [1] "fit_contrast"
```

`PostHocTest(..., method = "scheffe")`

`library(DescTools)`

Adjust the widths of confidence intervals for multiple contrasts using Scheffe's method.

```r
library(DescTools)

experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
  y = rnorm(15)
)

# fit an anova model -- see entry on `aov` and `lm`
fit <- lm(y ~ levels, data = experiment)
aov_table <- aov(fit)

# two contrasts
contrasts <- matrix(c(1, -1, 0, 1, 1, -1),  nrow = 2, byrow = TRUE)
PostHocTest(aov_table, method = "scheffe", contrast = t(contrasts))
```

```
##
##   Posthoc multiple comparisons of means: Scheffe Test
##     95% family-wise confidence level
```

```
##
## $levels
##             diff    lwr.ci   upr.ci   pval
## A-B   -0.2985533 -2.058236 1.461129 0.8951
## A,B-C  0.4686882 -1.686474 2.623850 0.8344
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

TukeyHSD

library(DescTools)

Use Tukey's Honest Significant Difference method to build experimentwise-valid confidence intervals for the contrast between all pairs of factor levels.

```
library(DescTools)
experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
  y = rnorm(15)
)

# fit an anova model -- see entry on `aov` and `lm`
aov_table <- aov(lm(y ~ levels, data = experiment))
TukeyHSD(aov_table)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = lm(y ~ levels, data = experiment))
##
## $levels
##           diff       lwr      upr     p adj
## B-A 0.71056919 -1.338319 2.759457 0.6354122
## C-A 0.77604192 -1.272846 2.824930 0.5845457
## C-B 0.06547273 -1.983415 2.114361 0.9960024
```

PostHocTest(..., method = "lsd")

library(DescTools)

Use Fisher's Least Significance Difference method to perform tests for the differences between all pairs of factor levels. Note that Fisher's method does not control the experimentwise error rate, like Tukey's or Scheffe's methods do.

```
library(DescTools)

experiment <- data.frame(
  levels = rep(c("A", "B", "C"), each = 5),
  y = rnorm(15)
)

# fit an anova model -- see entry on `aov` and `lm`
fit <- lm(y ~ levels, data = experiment)
aov_table <- aov(fit)
```

```r
# two contrasts
PostHocTest(aov_table, method = "lsd")
```

```
##
##   Posthoc multiple comparisons of means : Fisher LSD
##     95% family-wise confidence level
##
## $levels
##           diff     lwr.ci    upr.ci   pval
## B-A  0.9749289 -0.4132617 2.3631195 0.1519
## C-A  0.4579537 -0.9302369 1.8461443 0.4860
## C-B -0.5169751 -1.9051657 0.8712154 0.4329
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Chapter 4