Overview statistic tests PB

Jonas van Nijnatten

21 november 2018

Contents

contact: J.J.vanNijnatten@uva.nl

broncode: https://github.com/jonasvannijnatten/R_Data_Visualization

---

1 Software

1.1 Versies

software versions used for this tutorial:
- R version 3.5.1 (2018-07-02)
- car-package version: 3.0.0 (2018-03-23)

1.2 Installatie

Benodigde packages downloaden & installeren:

```
install.packages(pkgs="kableExtra",   repos="https://www.freestatistics.org/cran/")
install.packages(pkgs="car",   repos="https://www.freestatistics.org/cran/")
```

Benodigde packages activeren:

```
library(package=kableExtra)
library(package=car)
```

Table 1: data.long

ID	condition	score
1	A	34.29326
2	A	25.89247
3	A	22.94641
4	A	20.07555
5	A	24.81194
1	B	35.46555
2	B	35.38522
3	B	37.45495
4	B	42.83193
5	B	36.51421

2 T-test

2.1 Independent samples T-test

Show code for data generation

```
# generate data
N = 40
data.long = data.frame(
  ID = 1:N,
  condition = rep(x = c("A","B"), each = N),
  score = c(rnorm(n = N, mean = 25, sd = 6.5), rnorm(n = N, mean = 35, sd = 6.5))
)
```

Test for assumption of normality

```
by(data = data.long$score, INDICES = data.long$condition, FUN = shapiro.test)
```

```
## data.long$condition: A
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.97503, p-value = 0.511
##
## ------------------------------------------------------------
## data.long$condition: B
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.98031, p-value = 0.701
```

Test for equality of variances

```
leveneTest(y = data.long$score, group = data.long$condition)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  1  0.1801 0.6725
##       78
```

T-test

```
t.test(formula = score ~ condition, data = data.long, paired=FALSE, alternative="two.sided", var.equal=T
```

```
##
##  Two Sample t-test
##
```

Table 2: data.long

ID	condition	score
1	A	26.576829
2	A	20.551406
3	A	24.620292
4	A	24.503925
5	A	4.718622
1	B	39.235208
2	B	25.407539
3	B	23.709544
4	B	33.823236
5	B	44.859329

```
## data:  score by condition
## t = -7.2228, df = 78, p-value = 2.959e-10
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -12.962753  -7.360894
## sample estimates:
## mean in group A mean in group B
##        25.69379        35.85561
```

2.2 Paired samples T-test

Show code for data generation

```
# generate data
N = 30
data.long = data.frame(
  ID = rep(1:N,2),
  condition = rep(x = c("A","B"), each = N),
  score = c(rnorm(n = N, mean = 25, sd = 6.5), rnorm(n = N, mean = 35, sd = 6.5))
)
```

Test for assumption of normality

```
# calculate difference scores
diffScore = data.long$score[data.long$condition=="A"] - data.long$score[data.long$condition=="B"]
shapiro.test(diffScore)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  diffScore
## W = 0.96016, p-value = 0.3126
```

T-test

```
t.test(formula = score ~ condition, data = data.long, paired=TRUE, alternative="two.sided", var.equal=TR
```

```
##
##  Paired t-test
##
## data:  score by condition
## t = -5.4472, df = 29, p-value = 7.325e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -14.358335  -6.519436
## sample estimates:
## mean of the differences
##               -10.43889
```

Table 3: data.long

experience	salary
12.47743	9862.848
19.15308	10209.297
11.23352	10123.747
15.21043	10096.900
20.13432	10094.887

3 Correlation

Show code for data generation

```
# generate data
set.seed(05)
nrobs = 100
experience = rnorm(n = nrobs, mean = 15, sd = 3)
salary     = 10000 + ( 5 * experience ) + rnorm(n = nrobs, mean = 0, sd = 100)
data.long = data.frame(experience, salary)
# calculate correlation coefficient r
corr_coef = cor(x = data.long$experience, y = data.long$salary)
rm(list = c("nrobs", "experience", "salary"))
```

Test for assumption of normality

```
apply(X = data.long, MARGIN = 2, FUN = shapiro.test)
```

```
## $experience
##
##  Shapiro-Wilk normality test
##
## data:  newX[, i]
## W = 0.98711, p-value = 0.445
##
##
## $salary
##
##  Shapiro-Wilk normality test
##
## data:  newX[, i]
## W = 0.99257, p-value = 0.8607
```

Pearson correlation

```
cor.test(x=data.long$experience, y=data.long$salary, alternative = "two.sided", method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  data.long$experience and data.long$salary
## t = 2.5509, df = 98, p-value = 0.01229
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.05584335 0.42510763
## sample estimates:
##       cor
## 0.2495245
```

Spearman correlation

```
cor.test(x=data.long$experience, y=data.long$salary, alternative = "two.sided", method = "spearman")
```

```
##
```

```
##  Spearman's rank correlation rho
##
## data:  data.long$experience and data.long$salary
## S = 121190, p-value = 0.006181
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.2727873
```

Table 4: data.long

experience	salary
12.47743	9862.848
19.15308	10209.297
11.23352	10123.747
15.21043	10096.900
20.13432	10094.887

4 Regression

Show code for data generation

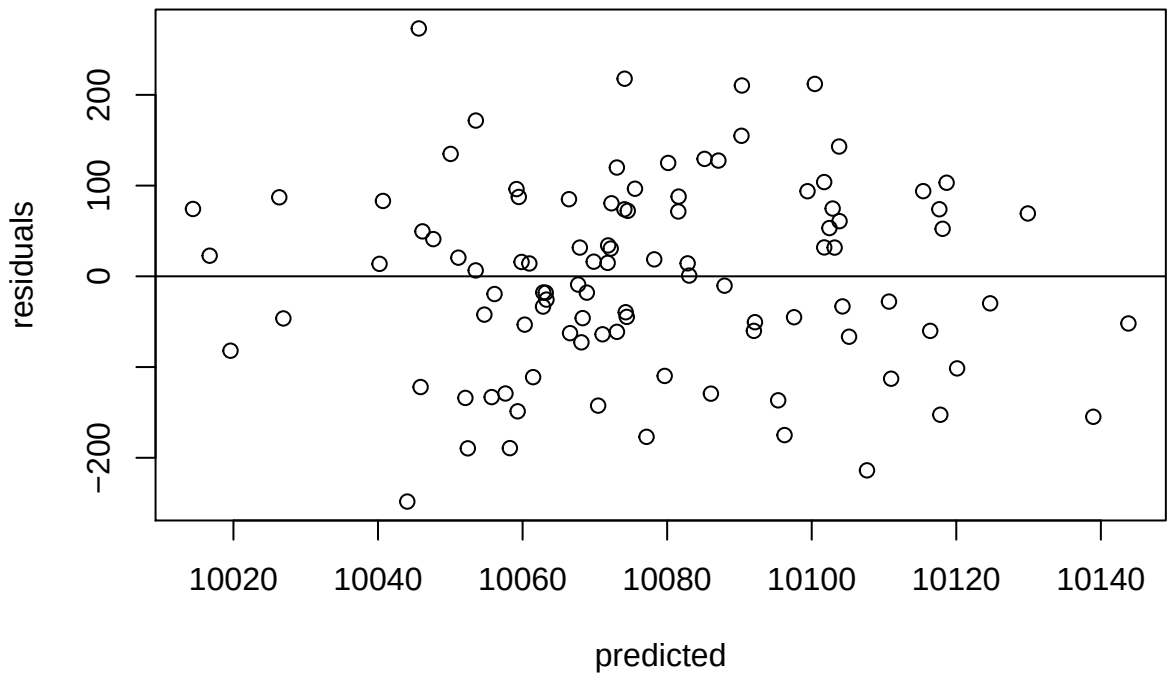
```
# generate data
set.seed(05)
nrobs = 100
experience = rnorm(n = nrobs, mean = 15, sd = 3)
salary     = 10000 + ( 5 * experience ) + rnorm(n = nrobs, mean = 0, sd = 100)
data.long = data.frame(experience, salary)
# calculate correlation coefficient r
corr_coef = cor(x = data.long$experience, y = data.long$salary)
rm(list = c("nrobs", "experience", "salary"))
```

Fit linear model

```
linearModel = lm(formula = salary ~ experience, data = data.long)
```

Test for assumption of normality & equal variances

```
# plot the residuals and look at the distribution around the 0-line and if the spread is equal over all
plot(x=linearModel$fitted.values, y=linearModel$residuals, xlab = "predicted", ylab = "residuals"); abli
```




```
shapiro.test(linearModel$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  linearModel$residuals
## W = 0.99208, p-value = 0.8268
```

Regression test

```
summary(linearModel)
```

```
##
## Call:
## lm(formula = salary ~ experience, data = data.long)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -248.24  -61.55   10.13   74.34  273.17
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 9934.674     56.812 174.870   <2e-16 ***
## experience     9.437      3.700   2.551   0.0123 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 104.4 on 98 degrees of freedom
## Multiple R-squared:  0.06226,    Adjusted R-squared:  0.05269
## F-statistic: 6.507 on 1 and 98 DF,  p-value: 0.01229
```

Table 5: data.long

subj	condition	score
1	A	17.537437
2	A	6.978033
21	B	21.403886
22	B	19.120283
41	C	15.895668
42	C	13.543431

5 One-way independent samples ANOVA

Toon code om voorbeeld data te genereren

```
set.seed(05)   # set seed
nrofconds = 3  # set number of conditions
nrofsubs  = 20 # set number of subjects
subj = as.factor(1:(nrofsubs*nrofconds))      # create array with subject IDs
condition = as.factor(rep(LETTERS[1:nrofconds],each=nrofsubs))   # create array with condition values
score = as.vector( replicate(
          nrofconds , rnorm(n = nrofsubs, mean = sample(8,1)+10 , sd = sample(5,1) )
        ) )                                          # create array with measurement values
data.long = data.frame(subj, condition, score);      # combine arrays into a data.frame
rm(list=c("subj", "condition","score","nrofconds","nrofsubs")) # delete unnecessary variables
```

Test for assumption of normality

```
by(data = data.long$score, INDICES = data.long$condition, FUN = shapiro.test)
```

```
## data.long$condition: A
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.9637, p-value = 0.6201
##
## ------------------------------------------------------------
## data.long$condition: B
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.92868, p-value = 0.1456
##
## ------------------------------------------------------------
## data.long$condition: C
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.94927, p-value = 0.3561
```

Test for equality of variances

```
leveneTest(y = data.long$score, group = data.long$condition)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value    Pr(>F)
## group  2  8.9065 0.0004306 ***
##       57
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA* with the aov method

```
myModel = aov(formula = score ~ condition, data = data.long)
summary(myModel)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## condition    2  475.4  237.69   24.88 1.71e-08 ***
## Residuals   57  544.5    9.55
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA* with the linear model method

```
myModel = lm(formula = score ~ condition, data = data.long)
summary(myModel)
```

```
##
## Call:
## lm(formula = score ~ condition, data = data.long)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9619 -1.5380 -0.1243  1.4997  7.6197
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.2260     0.6911  16.243  < 2e-16 ***
## conditionB    6.8584     0.9774   7.017 2.98e-09 ***
## conditionC    2.8167     0.9774   2.882  0.00557 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.091 on 57 degrees of freedom
## Multiple R-squared:  0.4661, Adjusted R-squared:  0.4474
## F-statistic: 24.88 on 2 and 57 DF,  p-value: 1.708e-08
```

Table 6: data.long

subj	score	treatment	control
1	12.347533	A	control
2	15.659598	A	experimental
61	11.536699	B	control
62	8.740125	B	experimental
121	18.740275	C	control
122	18.493456	C	experimental

6 Factorial independent samples ANOVA

Show code for data generation

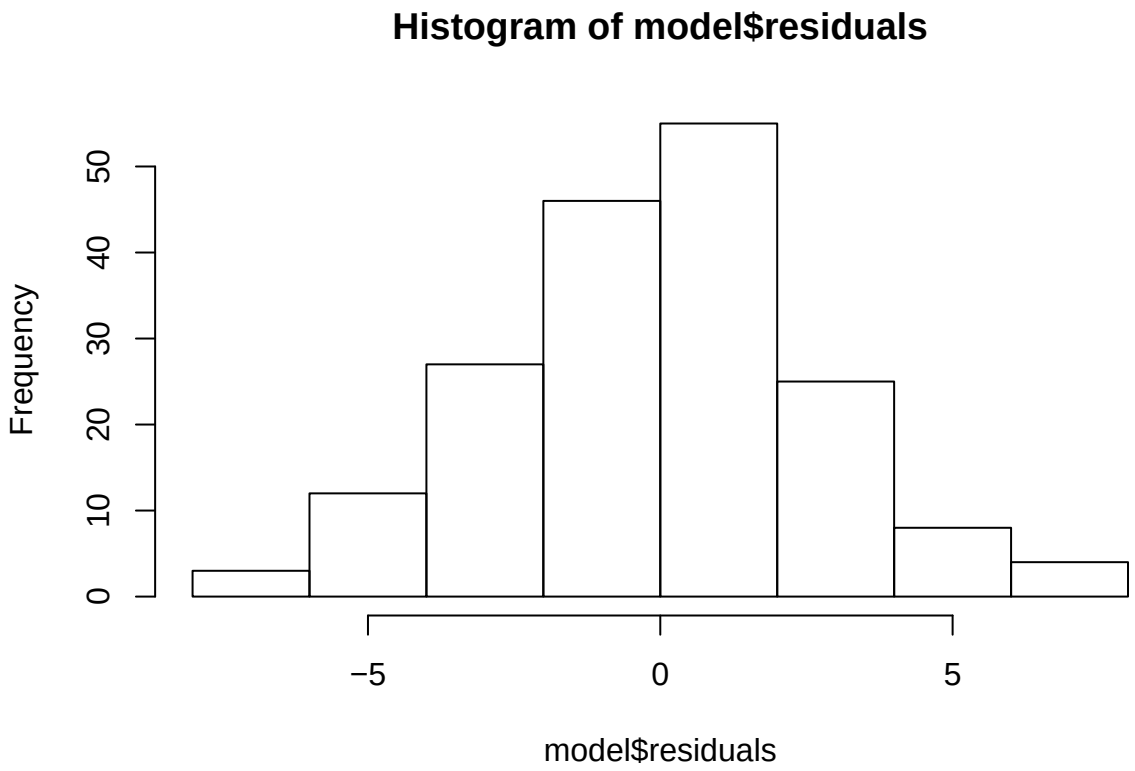
```
set.seed(01)   # set seed
nrofcondsf1 = 3  # set number of conditions for factor 1
nrofcondsf2 = 2  # set number of conditions for factor 2
nrofsubs    = nrofcondsf1*nrofcondsf2*30 # set number of subjects per condition
subj = as.factor(1:(nrofsubs))      # create array with subject IDs
# create array witht treatment conditions
treatment = as.factor(rep(LETTERS[1:nrofcondsf1],each=nrofsubs/nrofcondsf1))
# create array with control / experimental
control   = as.factor(rep(c("control","experimental"),times=nrofsubs/nrofcondsf2))
# create array with measurement values
score = as.vector( replicate(nrofcondsf1, replicate (
          nrofcondsf2 , rnorm(
            n = (nrofsubs/(nrofcondsf1*nrofcondsf2)),
            mean = 0 , sd = sample(5,1) ) + sample(8,1)+10
        ) ) )
# combine arrays into a data.frame
data.long = data.frame(subj, score, treatment, control);
# delete unnecessary arrays
rm(list=c("control","nrofcondsf1","nrofcondsf2","nrofsubs","score","subj","treatment"))
```

Fit model with multiple prediction factors

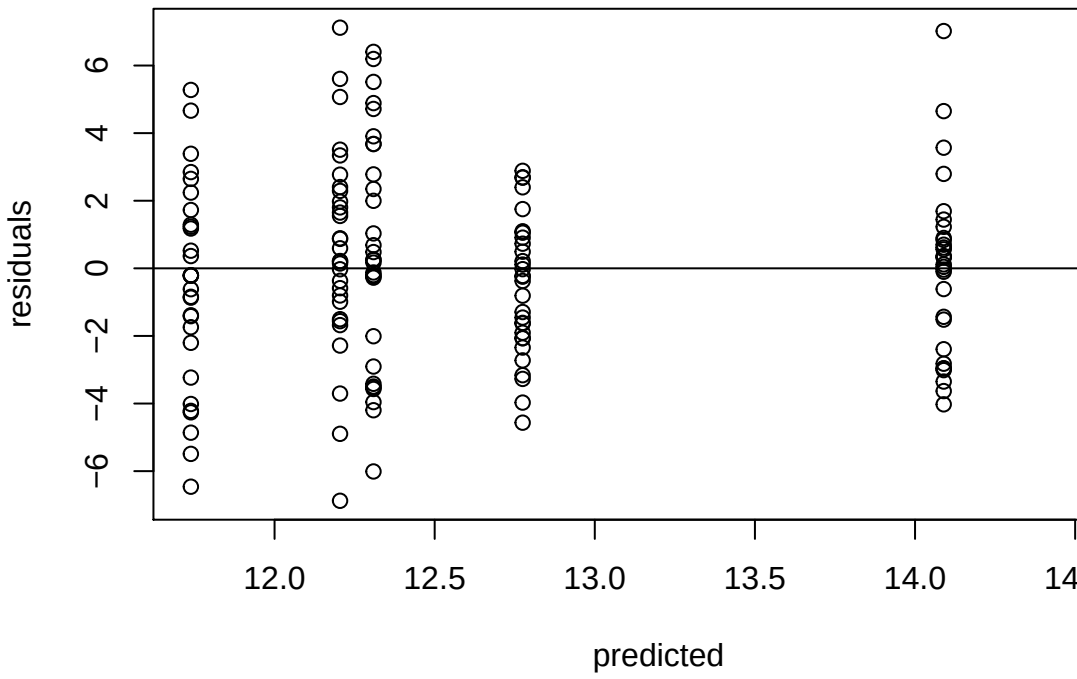
```
model = lm(formula = score ~ treatment+control, data = data.long)
```

Test for assumption of normality

```
hist(model$residuals)
plot(x = fitted(model), y = residuals(model), xlab="predicted", ylab="residuals"); abline(h=0)
by(data = data.long$score, INDICES = paste(data.long$treatment, data.long$control), FUN = shapiro.test)
```



Show residuals histogram



Show residuals vs predicted plot
Show output of Shapiro-Wilk test

```
## paste(data.long$treatment, data.long$control): A control
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
```

```
## W = 0.98372, p-value = 0.9135
##
## ------------------------------------------------------------
## paste(data.long$treatment, data.long$control): A experimental
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.97615, p-value = 0.7166
##
## ------------------------------------------------------------
## paste(data.long$treatment, data.long$control): B control
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.98224, p-value = 0.8815
##
## ------------------------------------------------------------
## paste(data.long$treatment, data.long$control): B experimental
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.95899, p-value = 0.2919
##
## ------------------------------------------------------------
## paste(data.long$treatment, data.long$control): C control
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.94015, p-value = 0.09179
##
## ------------------------------------------------------------
## paste(data.long$treatment, data.long$control): C experimental
##
##  Shapiro-Wilk normality test
##
## data:  dd[x, ]
## W = 0.93785, p-value = 0.07961
```

Test for equality of variances

```
leveneTest(y = data.long$score, group = as.factor(paste(data.long$treatment, data.long$control)))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   5  1.7187 0.1327
##       174
```

ANOVA with the linear model method

```
model = lm(formula = score ~ treatment+control, data = data.long)
library(car)
Anova(mod = model, type = 'II')
```

```
## Anova Table (Type II tests)
##
## Response: score
##            Sum Sq  Df F value    Pr(>F)
## treatment  185.93   2 12.2267 1.066e-05 ***
## control     14.63   1  1.9244    0.1671
## Residuals 1338.23 176
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model)
```

```
##
## Call:
## lm(formula = score ~ treatment + control, data = data.long)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.8762 -1.6438  0.1135  1.5856  7.1196
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          12.2045     0.4111  29.691  < 2e-16 ***
## treatmentB           -0.4658     0.5034  -0.925 0.356064
## treatmentC            1.8850     0.5034   3.744 0.000245 ***
## controlexperimental   0.5702     0.4111   1.387 0.167127
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.757 on 176 degrees of freedom
## Multiple R-squared:  0.1303, Adjusted R-squared:  0.1155
## F-statistic: 8.793 on 3 and 176 DF,  p-value: 1.829e-05
```

Show code for data generation

```
set.seed(01)   # set seed
nrofcondsf1 = 3  # set number of conditions for factor 1
nrofcondsf2 = 2  # set number of conditions for factor 2
nrofsubs    = nrofcondsf1*nrofcondsf2*20 # set number of subjects per condition
subj = as.factor(1:(nrofsubs))      # create array with subject IDs
# create array witht treatment conditions
treatment = as.factor(rep(LETTERS[1:nrofcondsf1],each=nrofsubs/nrofcondsf1))
# create array with control / experimental
control   = as.factor(rep(c("control","experimental"),times=nrofsubs/nrofcondsf2))
# create array with measurement values
score = as.vector( replicate(nrofcondsf1, replicate (
          nrofcondsf2 , rchisq(
            n = (nrofsubs/(nrofcondsf1*nrofcondsf2)),
            df = 3)
        )+  sample(14,1)+10  ) )
# combine arrays into a data.frame
data.long = data.frame(subj, score, treatment, control);
# delete unnecessary arrays
rm(list=c("control","nrofcondsf1","nrofcondsf2","nrofsubs","score","subj","treatment"))
```

Table 7: data.long

subj	cond	score
1	A	13.36729
2	A	11.32874
1	B	17.22369
2	B	11.03194
1	C	18.11333
2	C	15.62249

6.1 One-way repeated measures ANOVA

Show code for data generation

```
# Generate dataset
set.seed(01)   # set seed
nrofsubs  = 20 # set number of subjects
nrofconds = 3  # set number of conditions
subj = as.factor(rep(1:nrofsubs,nrofconds))      # create array with subject IDs
cond = as.factor(rep(LETTERS[1:nrofconds],each=nrofsubs))   # create array with condition values
score = as.vector( replicate(
          nrofconds , rnorm(n = nrofsubs, mean = sample(8,1)+10 , sd = sample(5,1) )
        ) )                                      # create array with measurement values
data.long = data.frame(subj, cond, score);       # combine arrays into a data.frame
rm(list=c("cond","nrofconds","nrofsubs","score","subj")) # delete arrays
```

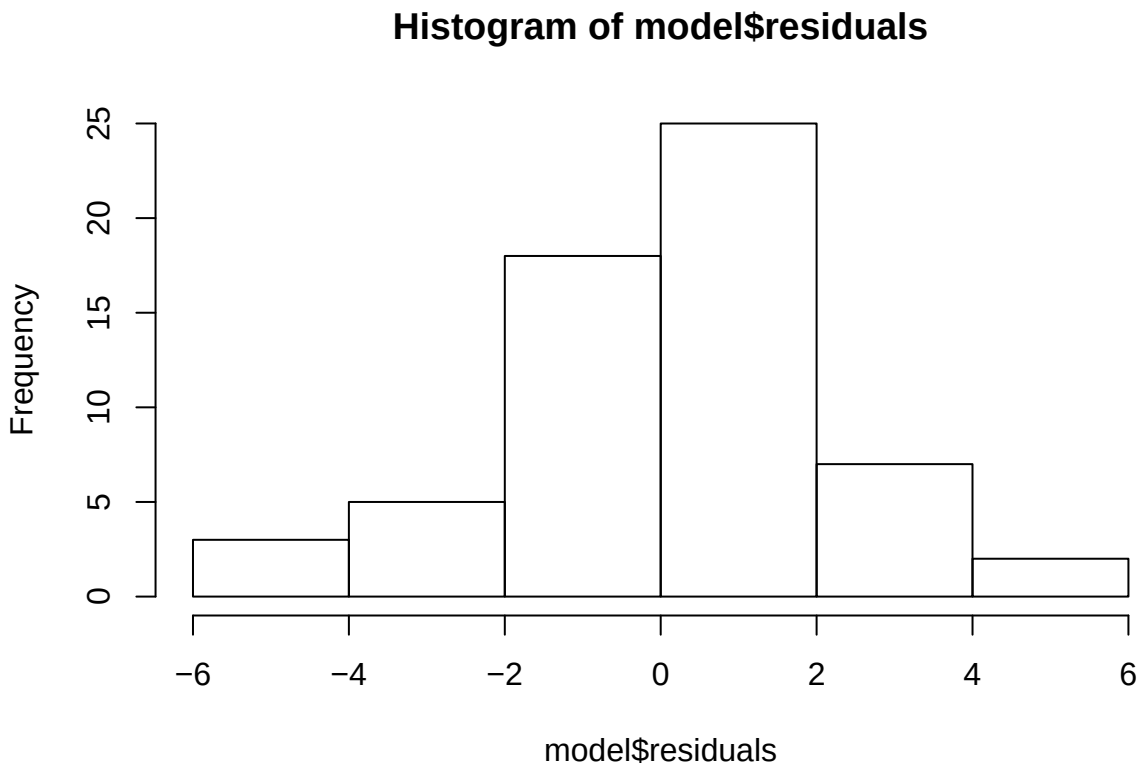
```
# Generate dataset
set.seed(01)   # set seed
nrofsubs  = 20 # set number of subjects
data.wide = data.frame(
  subj = as.factor(1:nrofsubs)   ,
  A =  rnorm(n = nrofsubs, mean = sample(8,1)+10 , sd = sample(5,1)) ,
  B =  rnorm(n = nrofsubs, mean = sample(8,1)+10 , sd = sample(5,1)) ,
  C =  rnorm(n = nrofsubs, mean = sample(8,1)+10 , sd = sample(5,1))
)
rm(list=c("nrofsubs")) # delete arrays
```

fit linear model

```
model = lm(formula = cbind(data.wide$A, data.wide$B, data.wide$C)~1)
```

Test for assumption of normality

```
hist(model$residuals)
```




```
shapiro.test(model$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model$residuals
## W = 0.97837, p-value = 0.3637
```

Test for assumption of sphericity

```
mauchly.test(model, X=~1)
```

```
##
##  Mauchly's test of sphericity
##  Contrasts orthogonal to
##  ~1
##
##
## data:  SSD matrix from lm(formula = cbind(data.wide$A, data.wide$B, data.wide$C) ~ 1)
## W = 0.88731, p-value = 0.341
```

ANOVA

```
anova(model, X = ~1, test="Spherical")
```

```
## Analysis of Variance Table
##
##
## Contrasts orthogonal to
## ~1
##
## Greenhouse-Geisser epsilon: 0.8987
## Huynh-Feldt epsilon:        0.9867
##
##             Df      F num Df den Df     Pr(>F)     G-G Pr     H-F Pr
## (Intercept)  1 21.953      2     38 4.6015e-07 1.4959e-06 5.3683e-07
```

```
## Residuals   19
```

Table 8: data.long

subj	score	treatment	control
1	13.73940	A	control
2	19.35138	A	control
1	21.60730	A	experimental
2	20.96033	A	experimental
1	31.81861	B	control
2	24.01975	B	control
1	13.66997	B	experimental
2	14.21302	B	experimental
1	24.98184	C	control
2	20.38903	C	control
2	14.74345	C	experimental

6.2 Factorial repeated measures ANOVA

Show code for data generation

```
set.seed(02)  # set seed
nrofcondsf1 = 3  # set number of conditions for factor 1
nrofcondsf2 = 2  # set number of conditions for factor 2
nrofsubs    = 10 # set number of subjects
# create array with subject IDs
subj = as.factor(rep(1:(nrofsubs),times=nrofcondsf1*nrofcondsf2))
# create array witht treatment conditions
treatment = as.factor(rep(LETTERS[1:nrofcondsf1],each=nrofsubs*nrofcondsf2))
# create array with control / experimental
control   = as.factor(
  rep(rep(c("control","experimental"),each=nrofsubs),times=nrofcondsf1))
# create array with measurement values
score = as.vector( replicate(nrofcondsf1,
                   replicate(nrofcondsf2,
                   rnorm(n = (nrofsubs), mean = sample(14,1)+10 , sd = sample(5,1)
                                       ) ) ) )
# combine arrays into a data.frame
data.long = data.frame(subj, score, treatment, control);
# delete arrays
rm(list=c("control","nrofcondsf1","nrofcondsf2","nrofsubs", "score", "subj", "treatment"))
```