Humpback Whales and Ship Noise

Fabian Blasch

06/07/2022

1 Data and Desciptive Statistics

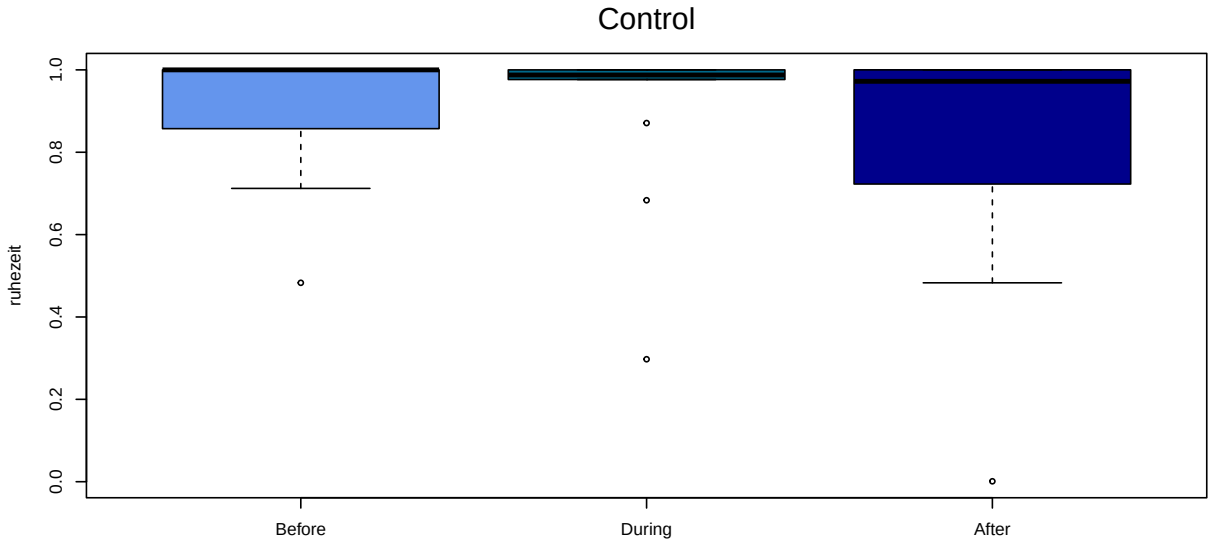
```r
# import data (this dataset is unfortunatelly not public)
openxlsx::read.xlsx("./../Data/Humpback_Whales_Data.xlsx") -> dat_whale

# fist a quick look at the missing values in the data
sapply(dat_whale,  \(x) sum(is.na(x))) |> knitr::kable(col.names = "NAs")
```

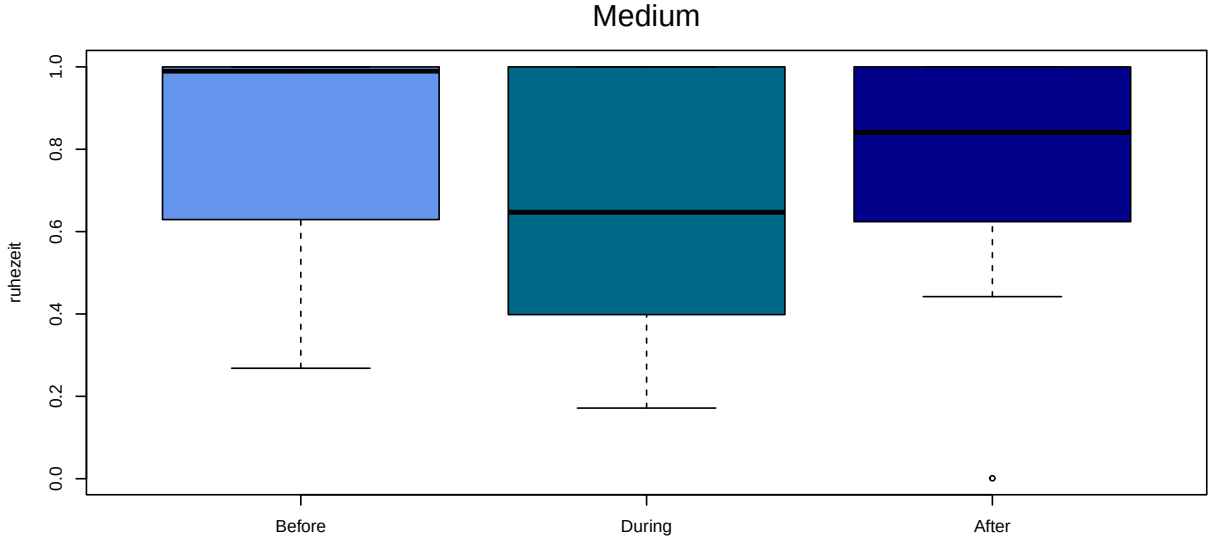
	NAs
Individuum	0
Treatment	0
Szenario	0
ruhezeit	0
speed	5
Atem	0

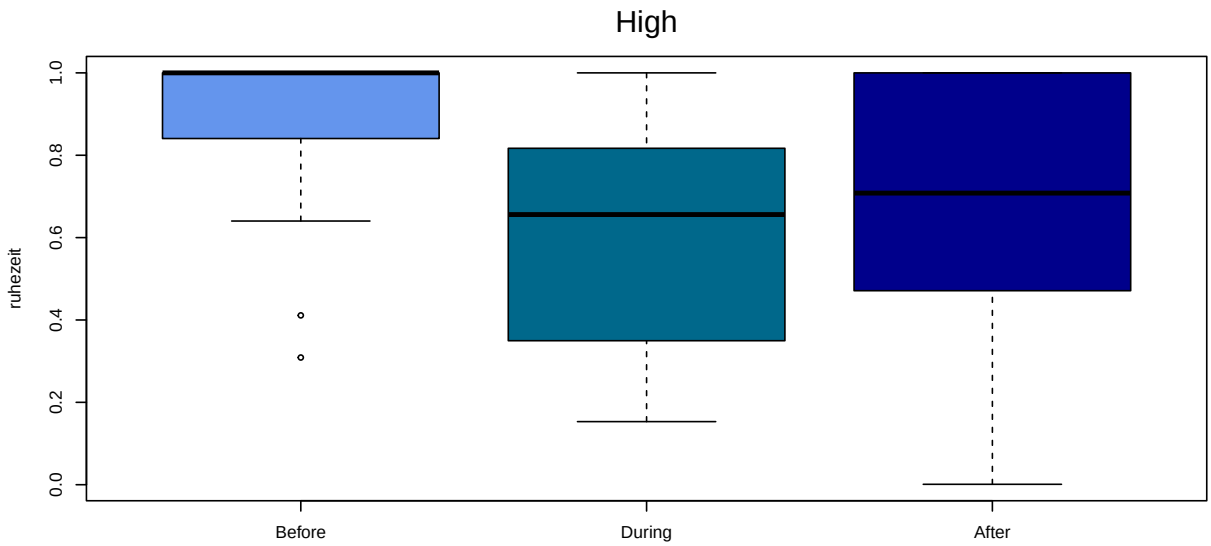
```r
# harmonize names
colnames(dat_whale) <- tolower(colnames(dat_whale))

# to numeric
lapply(dat_whale[, c("ruhezeit", "speed", "atem")], as.numeric) -> dat_whale[, c("ruhezeit", "speed", "a

# to factor
lapply(dat_whale[ ,!(colnames(dat_whale) %in%  c("ruhezeit", "speed", "atem"))],
       as.factor) -> dat_whale[ ,!(colnames(dat_whale) %in%  c("ruhezeit", "speed", "atem"))]

# relevel
factor(dat_whale[, "szenario"],
       levels = c("Before", "During", "After")) -> dat_whale[, "szenario"]
factor(dat_whale[, "treatment"], c("Control", "Medium", "High")) -> dat_whale[, "treatment"]

# add log
within(dat_whale,{
   logspeed <- log(speed)
   atem[atem == 0] <- 0.001
   ruhezeit[ruhezeit == 0] <- 0.001
   logatem <- log(atem)
   logruhezeit <- log(ruhezeit)
   sqrtatem <- sqrt(atem)
   sqrtspeed <- sqrt(speed)
   sqrtruhezeit <- sqrt(ruhezeit * 100)
}) -> dat_whale


# frist split into different intensities
dat_whale_intens <- split(dat_whale, dat_whale[, "treatment"])
```

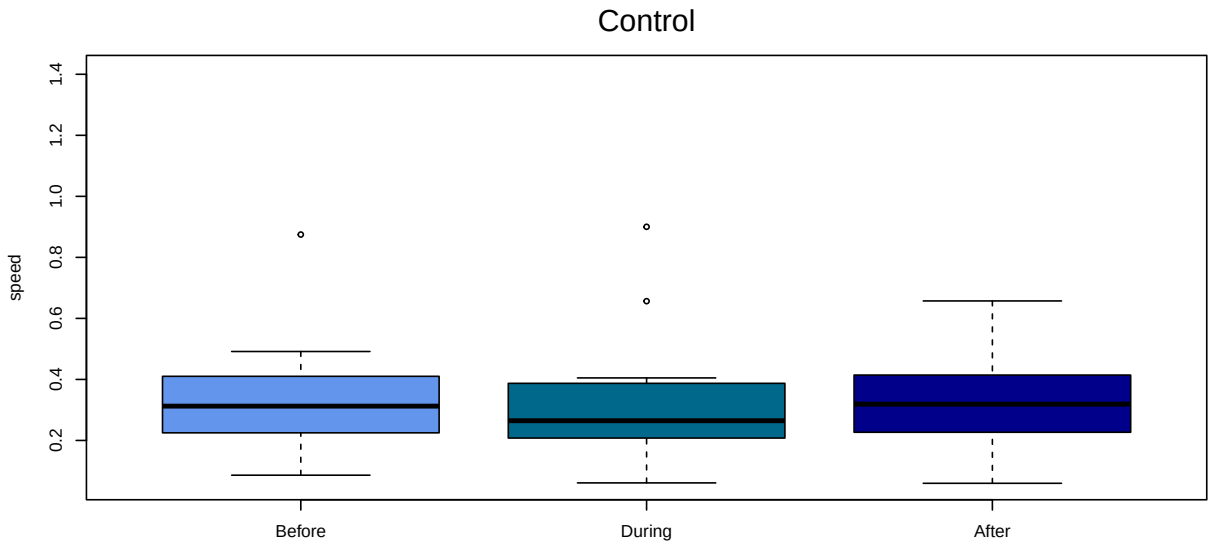
```r
# build formulas
formulae <- paste(c("ruhezeit", "speed", "atem"), "~", "szenario")

# max and min for plot y-axis
sapply(c(min, max), \(x){
```

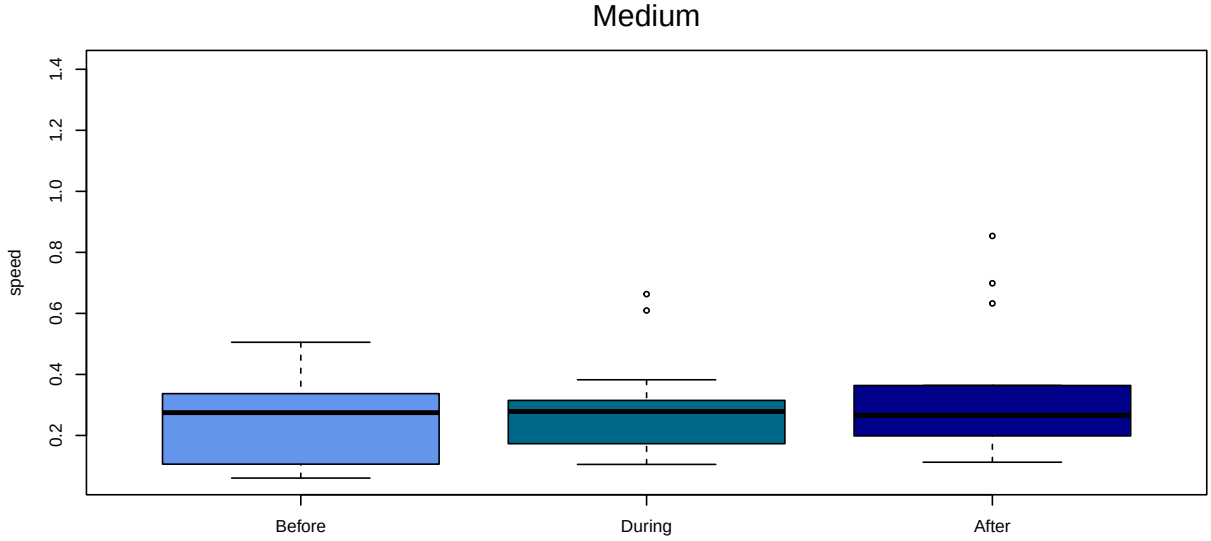
```
   sapply(dat_whale[, c("ruhezeit", "speed", "atem")], \(y) x(y, na.rm = TRUE))


}) -> ylims

# over szenarios
invis.Map(\(y, nom, lims){

   # safe for presentation
   # pdf(paste0("./../Presentation_1/", nom, ".pdf"))

   # align
   par(mfrow = c(3, 1), mar = c(2, 4, 4, 2) + 0.1)

   # over treatment
   invis.Map(\(x, nom){

      # boxplots
      boxplot(as.formula(y), data = x,
              col = c("cornflowerblue", "deepskyblue4", "darkblue"),
              ylim = c(lims[1], lims[2]))

      # add label
      mtext(nom, side = 3, line = 1, cex = 1.2)

   }, dat_whale_intens, names(dat_whale_intens))

   # close graph. device
   # dev.off()

}, formulae, c("resting", "speed", "respatory"), ylims |> t() |> as.data.frame())
```

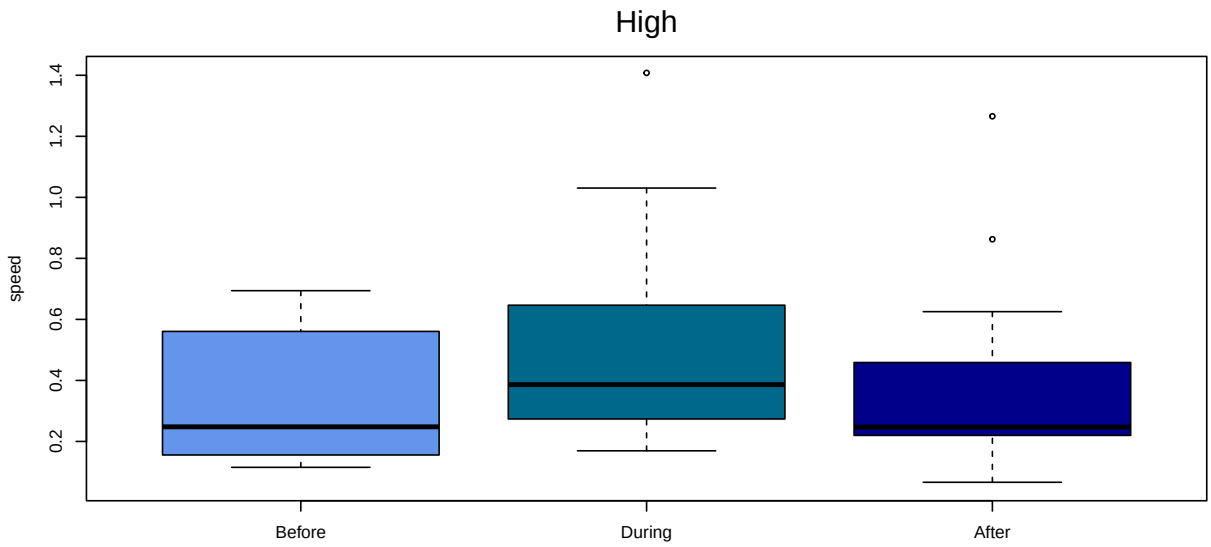

Control
ruhezeit
0.0
0.2
0.4
0.6
0.8
1.0
Before
During
After

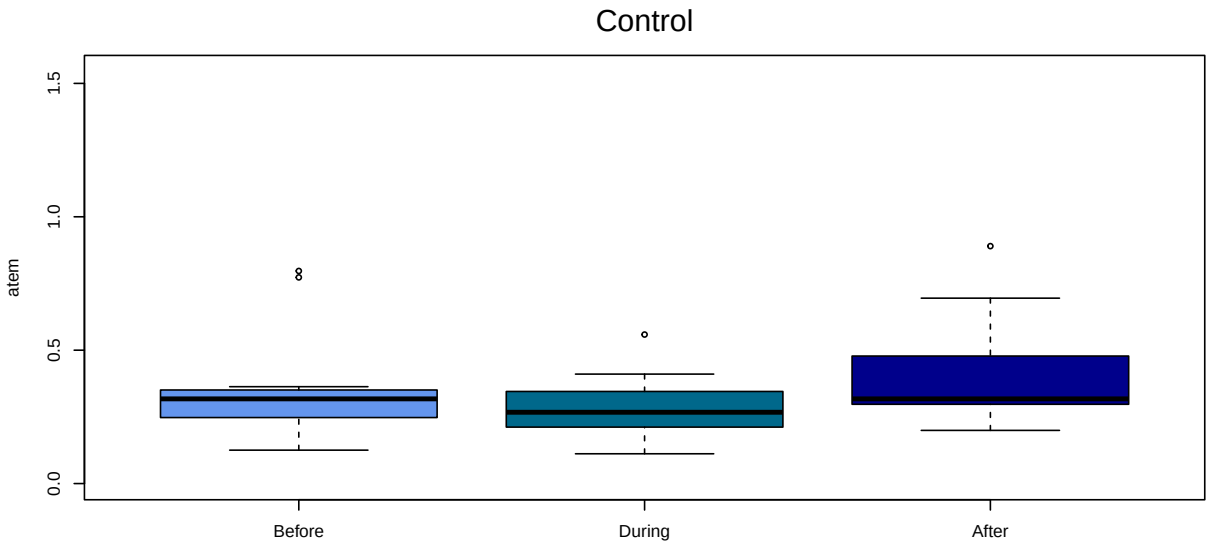

Medium
ruhezeit
0.0
0.2
0.4
0.6
0.8
1.0
Before
During
After

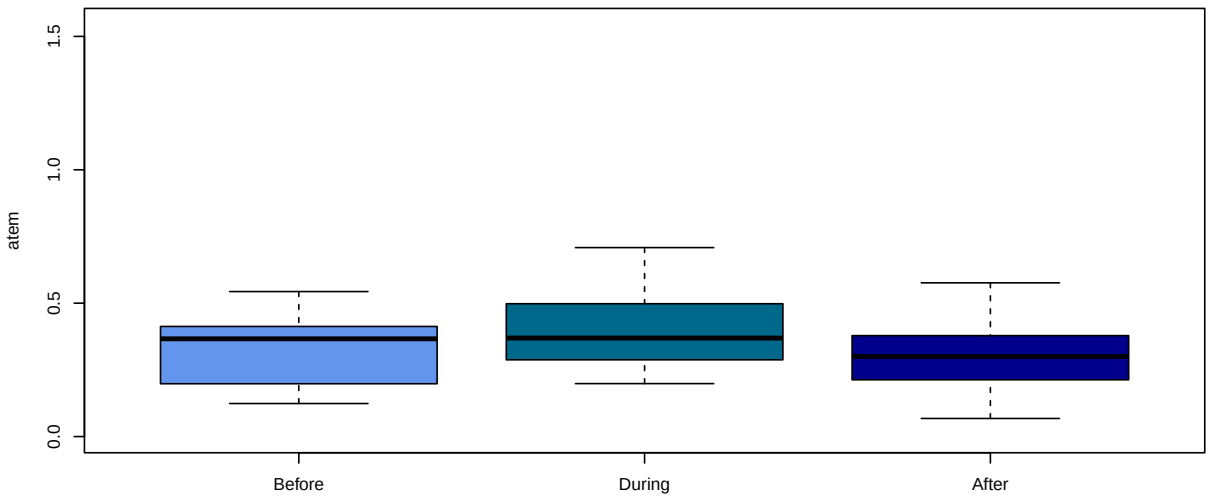

High
ruhezeit
0.0
0.2
0.4
0.6
0.8
1.0
Before
During
After

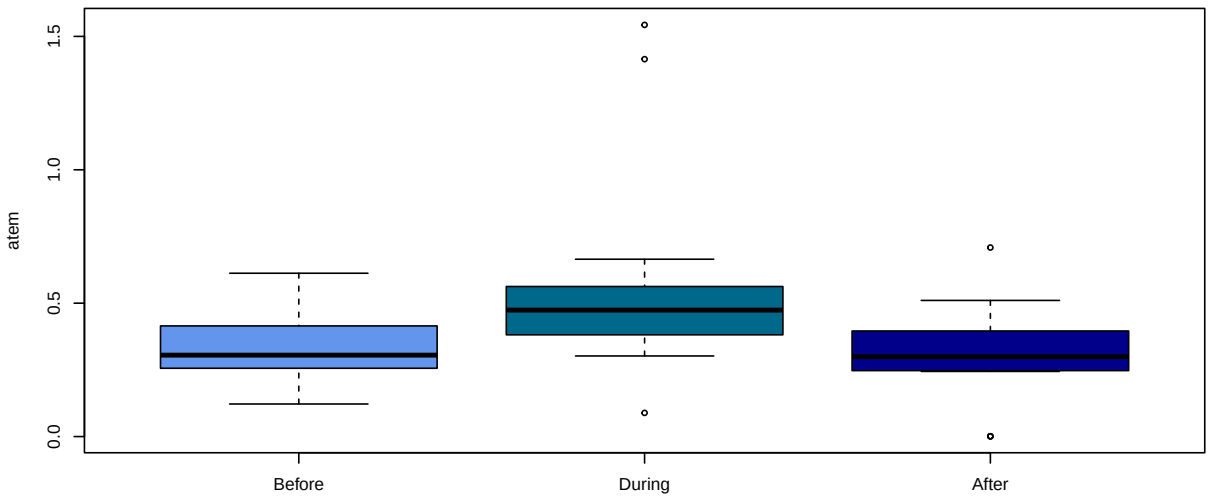

Control
speed
0.2
0.4
0.6
0.8
1.0
1.2
1.4
Before
During
After


Medium
speed
0.2
0.4
0.6
0.8
1.0
1.2
1.4
Before
During
After


High
speed
0.2
0.4
0.6
0.8
1.0
1.2
1.4
Before
During
After

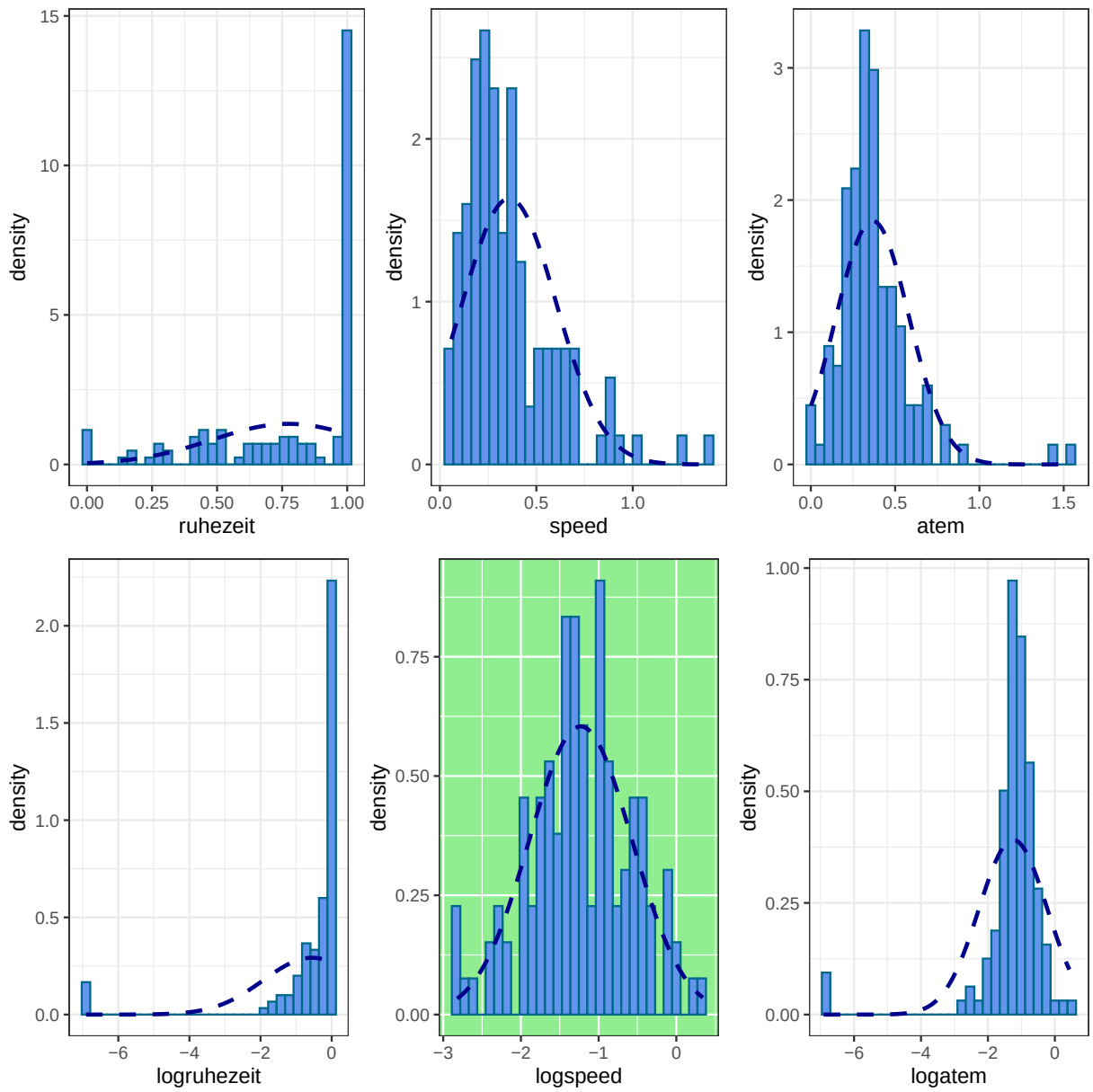

Control
atem
0.0
0.5
1.0
1.5
Before
During
After


Medium
atem
0.0
0.5
1.0
1.5
Before
During
After


High
atem
0.0
0.5
1.0
1.5
Before
During
After

2 Distribution of Covariates

```
# vars
nom <- c("ruhezeit", "speed", "atem")
nom <- c(nom, paste0("log", nom))

# loop to generate plots
# Map(\(x, bool) Dens_norm_plot(y = x, bg_alt = bool),
#     nom, c(F, F, F, F, T, F, F ,F ,T)) -> plots

# display (remove useless bin width messages)
# print((plots[[1]] + plots[[2]] + plots[[3]]) /
#       (plots[[4]] + plots[[5]] + plots[[6]]) /
#       (plots[[7]] + plots[[8]] + plots[[9]])) |>
#       suppressWarnings() |>
#       suppressMessages()

# loop to generate plots
Map(\(x, bool) Dens_norm_plot(y = x, bg_alt = bool),
    nom, c(F, F, F, F, T, F)) -> plots


# safe for presentation
# pdf("./../Presentation_2/Variables.pdf")

# display (remove useless bin width messages)
print((plots[[1]] + plots[[2]] + plots[[3]]) /
      (plots[[4]] + plots[[5]] + plots[[6]])) |>
      suppressWarnings() |>
      suppressMessages()
```




```
# close graph. device
# dev.off()
```

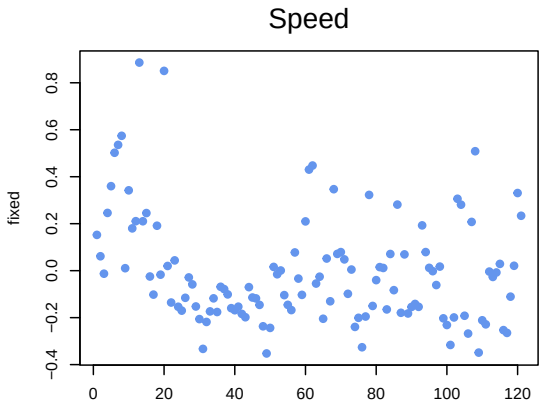
3 Model

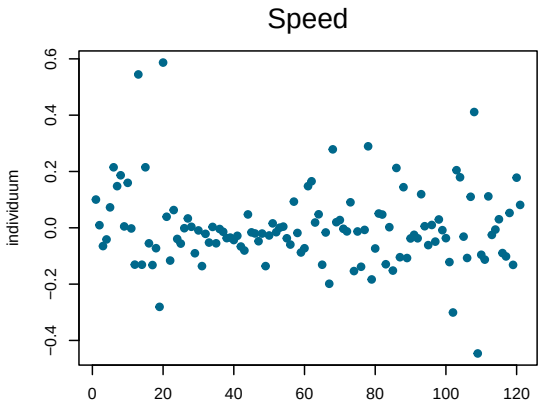
```
# formulas
formulae_lmm <- c("speed ~ I(treatment) * I(szenario)",
                  "atem ~ I(treatment) * I(szenario)",
                  "logspeed ~ I(treatment) * I(szenario)")

formulae_glmm <- "ruhezeit ~ I(treatment) * I(szenario)"

# comb
```

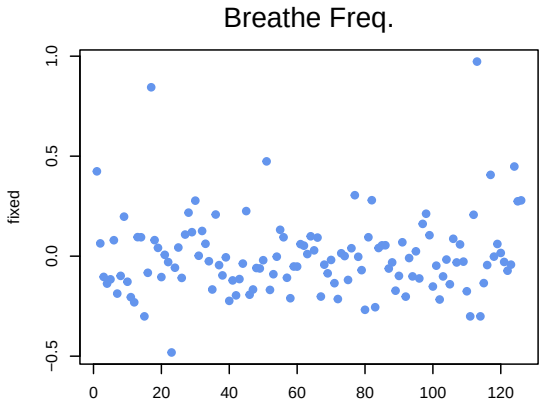
```r
formulae_cmb <- list(formulae_lmm, formulae_glmm)

# fit models
Map(\(type, bool1){

   Map(\(x, corr){

      if(bool1){ # LMMs

         # fit LMM
         nlme::lme(as.formula(x), random = ~ 1 | individuum,
                   data = dat_whale, na.action = na.omit,
                   method = "REML", correlation = corr) -> fit

         # summary
         list(fit,
              summary(fit))

      } else { # GLMMs

         # fit glmm PQL
         MASS::glmmPQL(as.formula(x), random = ~ 1 | individuum,
                       family = binomial(link = "logit"),
                       data = dat_whale) -> fit

         # summary
         list(fit,
              summary(fit))
      }

   }, type, list(corAR1(), NULL, corAR1(), NULL)) |> setNames(type)

}, formulae_cmb, c(TRUE, FALSE)) |> setNames(c("LMM", "GLMM")) |>
   suppressMessages() |> suppressWarnings() -> models

# remove empty
lapply(models, \(x){

  # subset models from loop list
  x[!(names(x) |> is.na())]

}) -> models

# fits and summaries
LMM_fits <- lapply(models[[1]], "[[", 1)
LMM_summaries <- lapply(models[[1]], "[[", 2)
glmmPQL_fits <- lapply(models[[2]], "[[", 1)
glmmPQL_summaries <- lapply(models[[2]], "[[", 2)

# rebind fits for plotting
fits <- c(LMM_fits, glmmPQL_fits)
summaries <- c(LMM_summaries, glmmPQL_summaries)
```

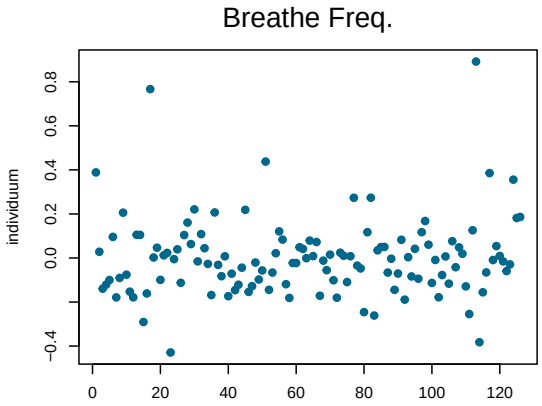
4 Residual Diagnostics

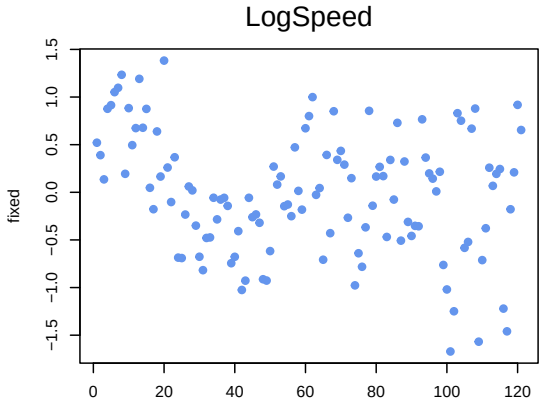
```
# adjust names
nome <- c("Speed", "Breathe Freq.", "LogSpeed", "% Resting")

# align
par(mfrow = c(4, 2), mar = c(2, 4, 4, 2) + 0.1)

# residual plots
invis.Map(\(x, nom){

   invis.Map(\(y, col){

      # plots
      plot(x[["residuals"]][, y], type = "p", ylab = y, xlab = "", pch = 19,
           col = col)

      # label
      mtext(nom, side = 3, line = 1, cex = 1.2)

   }, c("fixed", "individuum"), c("cornflowerblue", "deepskyblue4"))

}, fits, nome)
```

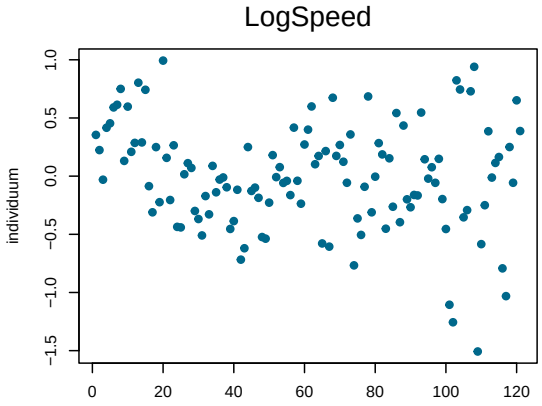

Speed
fixed
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
0
20
40
60
80
100
120

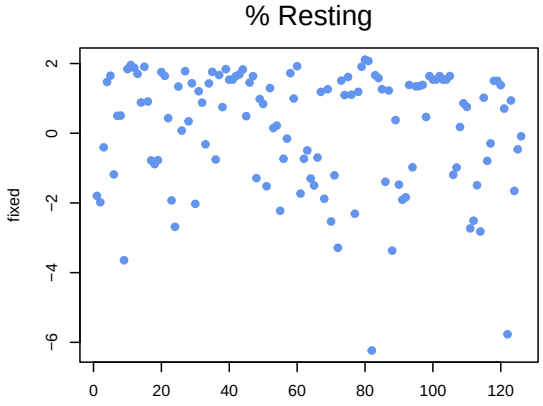

Speed
individuum
−0.4
−0.2
0.0
0.2
0.4
0.6
0
20
40
60
80
100
120

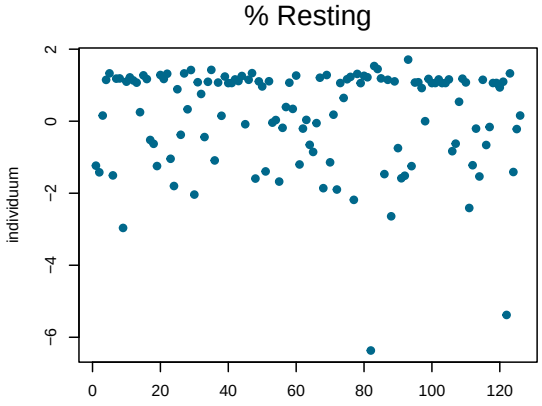

Breathe Freq.
fixed
−0.5
0.0
0.5
1.0
0
20
40
60
80
100
120


Breathe Freq.
individuum
−0.4
0.0
0.2
0.4
0.6
0.8
0
20
40
60
80
100
120

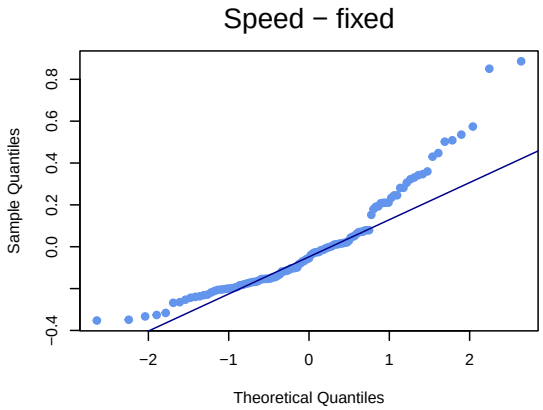

LogSpeed
fixed
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
0
20
40
60
80
100
120

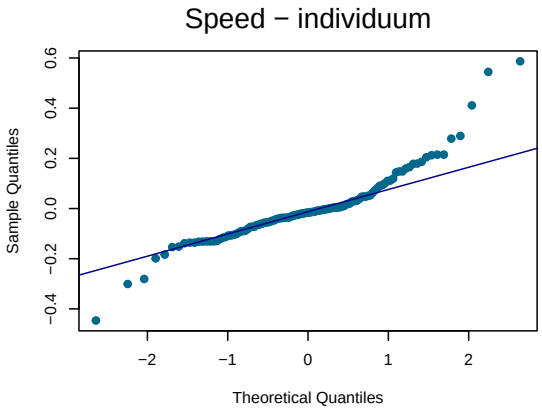

LogSpeed
individuum
−1.5
−1.0
−0.5
0.0
0.5
1.0
0
20
40
60
80
100
120

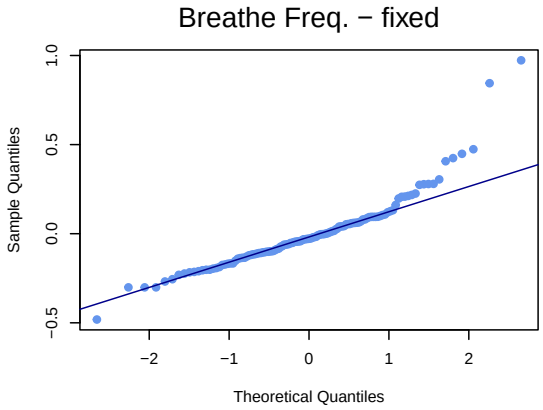

% Resting
fixed
−6
−4
−2
0
2
0
20
40
60
80
100
120

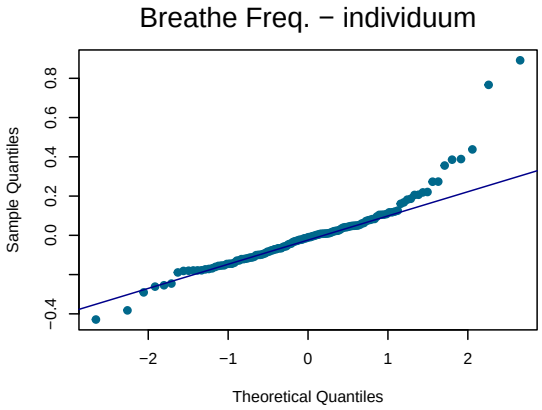

% Resting
individuum
−6
−4
−2
0
2
0
20
40
60
80
100
120

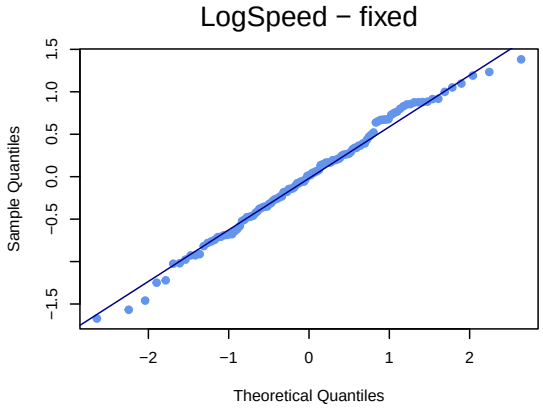
```
# qqplots
par(mfrow = c(4, 2), mar = c(3.8, 4, 4, 2) + 0.1)

# residual plots
invis.Map(\(x, nom){

   invis.Map(\(y, col){

      # plots
      qqnorm(x[["residuals"]][, y], pch = 19, col = col, main = "")

      # line
      qqline(x[["residuals"]][, y], col = "darkblue")

      # label
      mtext(paste(nom, "-", y), side = 3, line = 1, cex = 1.2)

   }, c("fixed", "individuum"), c("cornflowerblue", "deepskyblue4"))

}, fits, nome)
```

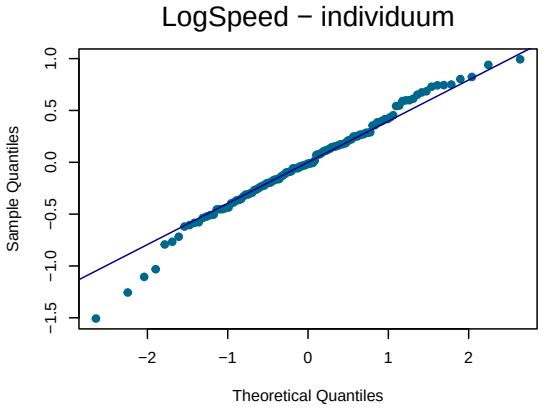

Speed − fixed
Sample Quantiles
Theoretical Quantiles

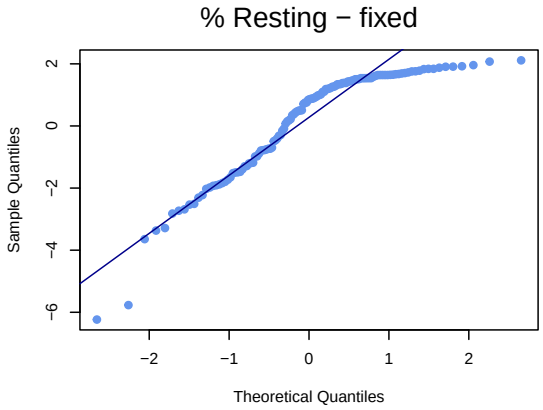

Speed − individuum
Sample Quantiles
Theoretical Quantiles

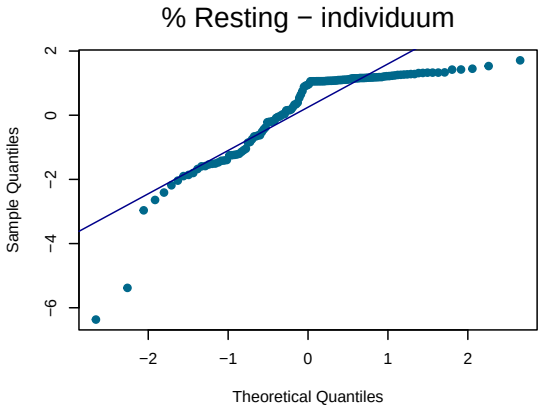

Breathe Freq. − fixed
Sample Quantiles
Theoretical Quantiles


Breathe Freq. − individuum
Sample Quantiles
Theoretical Quantiles


LogSpeed − fixed
Sample Quantiles
Theoretical Quantiles


LogSpeed − individuum
Sample Quantiles
Theoretical Quantiles


% Resting − fixed
Sample Quantiles
Theoretical Quantiles


% Resting − individuum
Sample Quantiles
Theoretical Quantiles

5 Coefficients and t-Tests

```
invis.lapply(names(summaries), \(x){

   # mods
   mod <- summaries[[x]][["tTable"]]

   # print table
   knitr::kable(mod)

}) |> setNames(names(summaries))
```

```
## $`speed ~ I(treatment) * I(szenario)`
##
##
## |                                    |      Value| Std.Error| DF|    t-value|   p-value|
## |:-----------------------------------|----------:|---------:|--:|----------:|---------:|
## |(Intercept)                         |  0.3392015| 0.0675835| 73|  5.0189974| 0.0000035|
## |I(treatment)Medium                  | -0.0796592| 0.0950938| 39| -0.8376908| 0.4073089|
## |I(treatment)High                    |  0.0128286| 0.0923367| 39|  0.1389326| 0.8902184|
## |I(szenario)During                   | -0.0132775| 0.0661239| 73| -0.2007976| 0.8414152|
## |I(szenario)After                    |  0.0367240| 0.0672287| 73|  0.5462546| 0.5865566|
## |I(treatment)Medium:I(szenario)During |  0.0571256| 0.0935131| 73|  0.6108835| 0.5431742|
## |I(treatment)High:I(szenario)During   |  0.1831087| 0.0903425| 73|  2.0268285| 0.0463329|
## |I(treatment)Medium:I(szenario)After  |  0.0554731| 0.0934130| 73|  0.5938474| 0.5544499|
## |I(treatment)High:I(szenario)After    |  0.0262892| 0.0928123| 73|  0.2832513| 0.7777859|
##
## $`atem ~ I(treatment) * I(szenario)`
##
##
## |                                    |      Value| Std.Error| DF|    t-value|   p-value|
## |:-----------------------------------|----------:|---------:|--:|----------:|---------:|
## |(Intercept)                         |  0.3487213| 0.0568680| 78|  6.1321165| 0.0000000|
## |I(treatment)Medium                  | -0.0228728| 0.0789743| 39| -0.2896229| 0.7736389|
## |I(treatment)High                    | -0.0161951| 0.0776965| 39| -0.2084406| 0.8359698|
## |I(szenario)During                   | -0.0644547| 0.0757239| 78| -0.8511814| 0.3972742|
## |I(szenario)After                    |  0.0673726| 0.0757239| 78|  0.8897145| 0.3763551|
## |I(treatment)Medium:I(szenario)During |  0.1421530| 0.1051600| 78|  1.3517785| 0.1803532|
## |I(treatment)High:I(szenario)During   |  0.3022035| 0.1034585| 78|  2.9210116| 0.0045609|
## |I(treatment)Medium:I(szenario)After  | -0.0945670| 0.1051600| 78| -0.8992675| 0.3712780|
## |I(treatment)High:I(szenario)After    | -0.0978333| 0.1034585| 78| -0.9456288| 0.3472582|
##
## $`logspeed ~ I(treatment) * I(szenario)`
##
##
## |                                    |      Value| Std.Error| DF|    t-value|   p-value|
## |:-----------------------------------|----------:|---------:|--:|----------:|---------:|
## |(Intercept)                         | -1.2308608| 0.1830245| 73| -6.7251132| 0.0000000|
## |I(treatment)Medium                  | -0.3283010| 0.2580591| 39| -1.2721934| 0.2108378|
## |I(treatment)High                    | -0.0179156| 0.2500591| 39| -0.0716453| 0.9432503|
## |I(szenario)During                   | -0.1078457| 0.1708506| 73| -0.6312281| 0.5298630|
## |I(szenario)After                    |  0.0821429| 0.2035207| 73|  0.4036098| 0.6876796|
## |I(treatment)Medium:I(szenario)During |  0.3416313| 0.2415008| 73|  1.4146176| 0.1614320|
## |I(treatment)High:I(szenario)During   |  0.5076296| 0.2334264| 73|  2.1746880| 0.0328943|
```

```
## |I(treatment)Medium:I(szenario)After  |  0.2675867| 0.2830882| 73|  0.9452415| 0.3476538|
## |I(treatment)High:I(szenario)After    |  0.0196943| 0.2805153| 73|  0.0702076| 0.9442204|
##
## $`ruhezeit ~ I(treatment) * I(szenario)`
##
##
## |                                    |      Value| Std.Error| DF|    t-value|   p-value|
## |:-----------------------------------|----------:|---------:|--:|----------:|---------:|
## |(Intercept)                         |  2.3808540| 0.6231463| 78|  3.8206985| 0.0002658|
## |I(treatment)Medium                  | -0.7865013| 0.7862894| 39| -1.0002696| 0.3233462|
## |I(treatment)High                    | -0.3502397| 0.8093376| 39| -0.4327486| 0.6675812|
## |I(szenario)During                   | -0.0267061| 0.8095529| 78| -0.0329887| 0.9737679|
## |I(szenario)After                    | -1.0226216| 0.7058604| 78| -1.4487589| 0.1514133|
## |I(treatment)Medium:I(szenario)During | -0.7751521| 0.9789670| 78| -0.7918062| 0.4308751|
## |I(treatment)High:I(szenario)During   | -1.5056474| 0.9912816| 78| -1.5188896| 0.1328334|
## |I(treatment)Medium:I(szenario)After  |  0.7143004| 0.9101265| 78|  0.7848364| 0.4349268|
## |I(treatment)High:I(szenario)After    | -0.3409966| 0.9105983| 78| -0.3744753| 0.7090669|
```