Humpback Whales and Ship Noise
A Replication Study of Sprogis et al. 2020

Fabian Blasch

06/17/2022

1 Introduction

The whale watching industry in Australia has been growing at a rapid rate since the cessation of whaling in 1979. Typically, whale watching companies operate large boats to transport tourist to coastal areas to observe groups of whales. Marine Biologists report accumulating evidence that boat-based whale-watching negatively affects whale's behavior. In particular, one publication by Sprogis, Videsen, and Madsen (2020) uses a controlled exposure experiment (CEE) to test whether vessel noises affect whales' behavior. To collect the necessary data the marine biologists approach whales with a boat and expose them to different noise levels (low, medium and high). Before, during and after the approach, many variables are measured. Three of those measurements were made available to me, the mean swim speed (m/s), the breathe frequency per minute and the proportion of time resting. The aim of this short report is thus to replicate the study for the provided variables. The figure below provides a concise summary of the CEE.(A) shows the experiment design and (B) displays the boat used for approaching the whales including the transponder used to generate the vessel noise imitations.

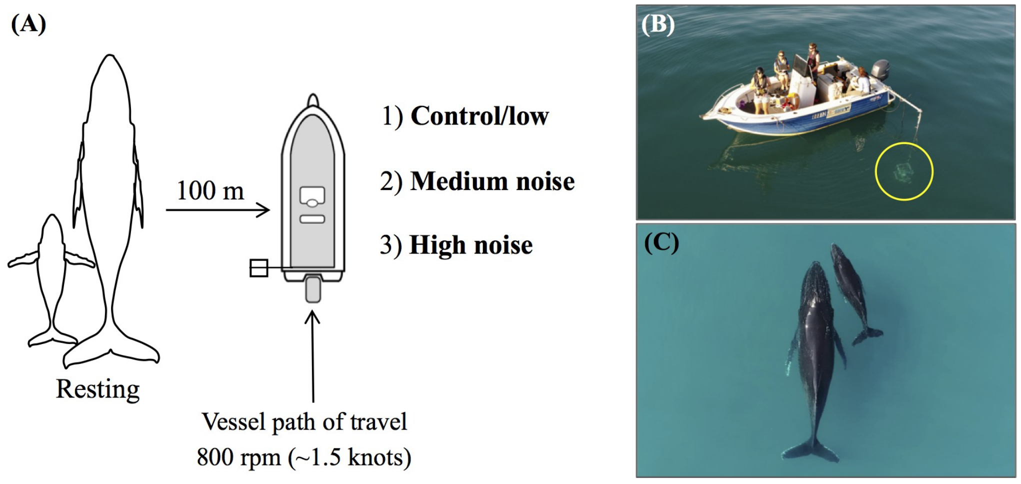


Figure 1: CEE Sprogis et al.

2 Data and Desciptive Statistics

To get a first impression of the data, we may first adjust the classes in the data set and compute log transformations of the variables. Subsequently, box plots across different noise intensities are displayed for the three levels of the scenario factor, i.e., before, during and after.

```r
# import data
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

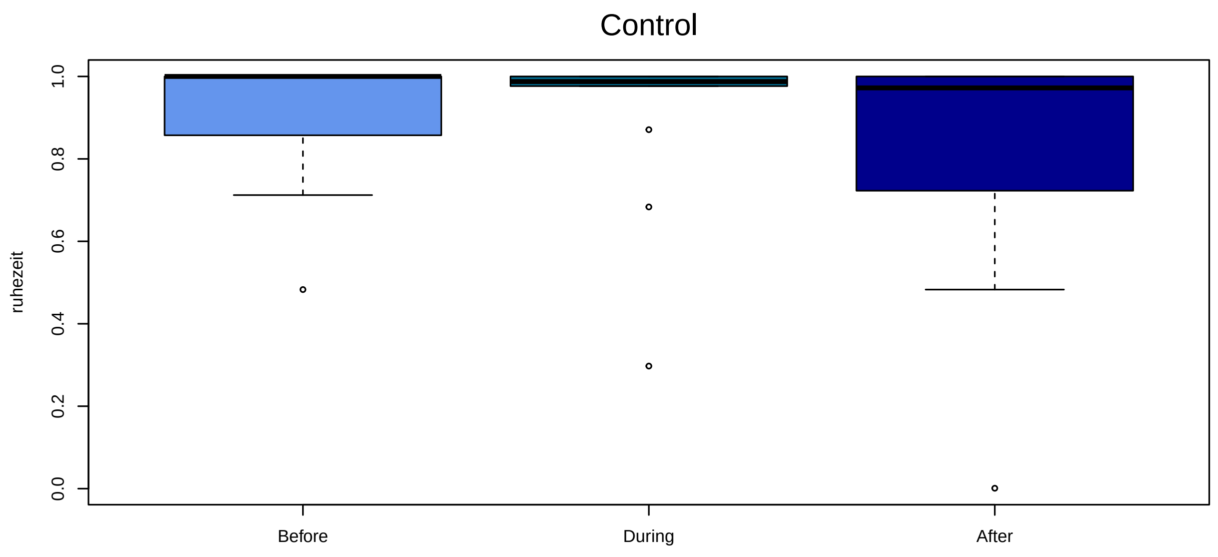
	NAs
ruhezeit	0
speed	5
Atem	0

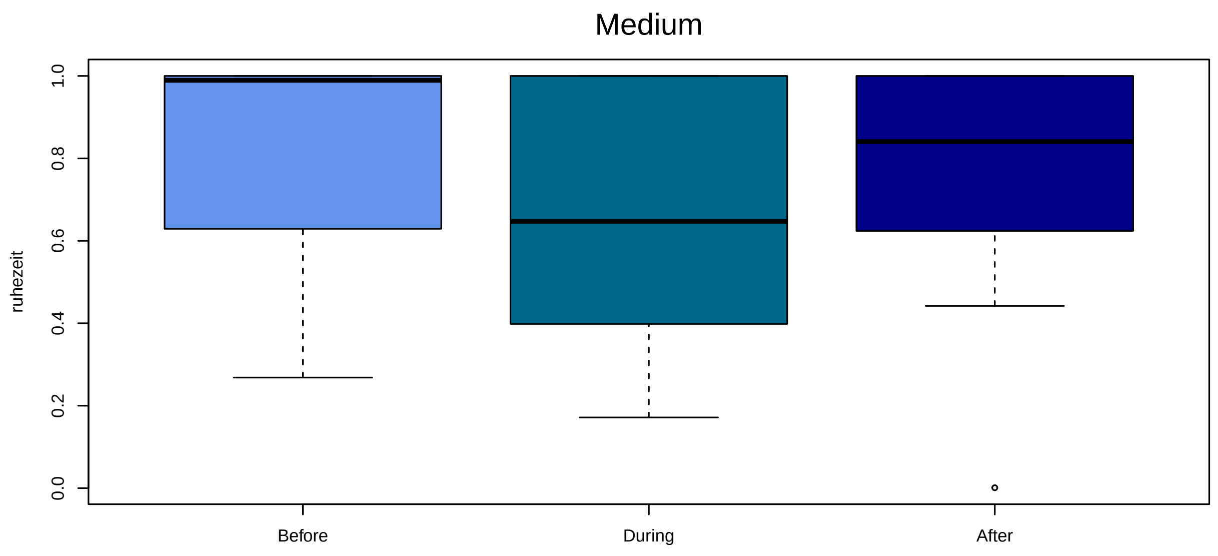
```r
# harmonize names
colnames(dat_whale) <- tolower(colnames(dat_whale))

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

# 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)
}) -> dat_whale

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

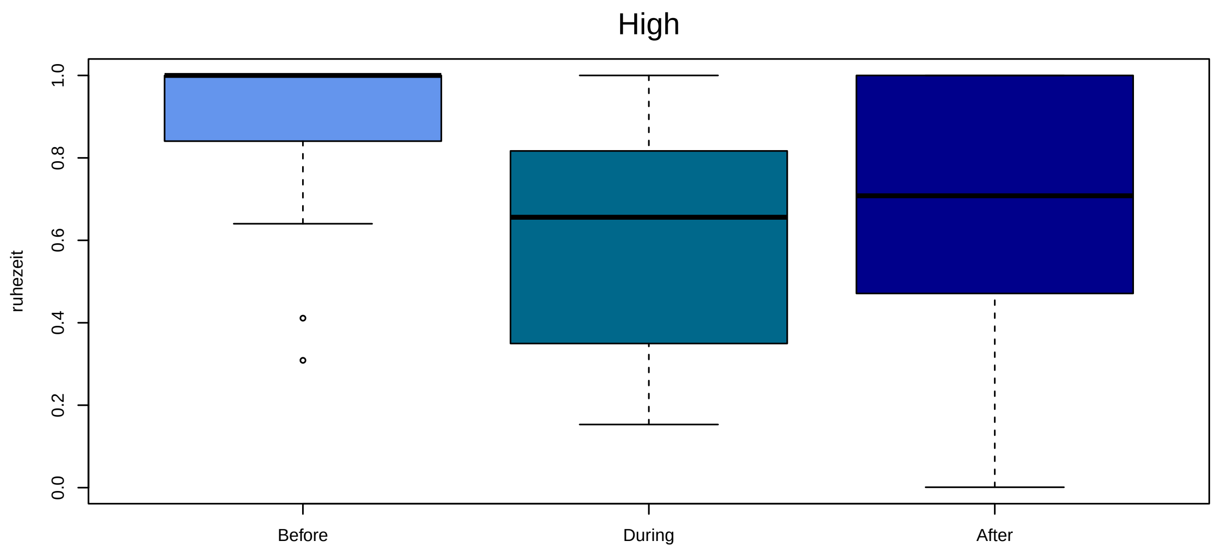
In regards to missing values in the data, we omit 5 observations because the mean swim speed is not available in those cases.

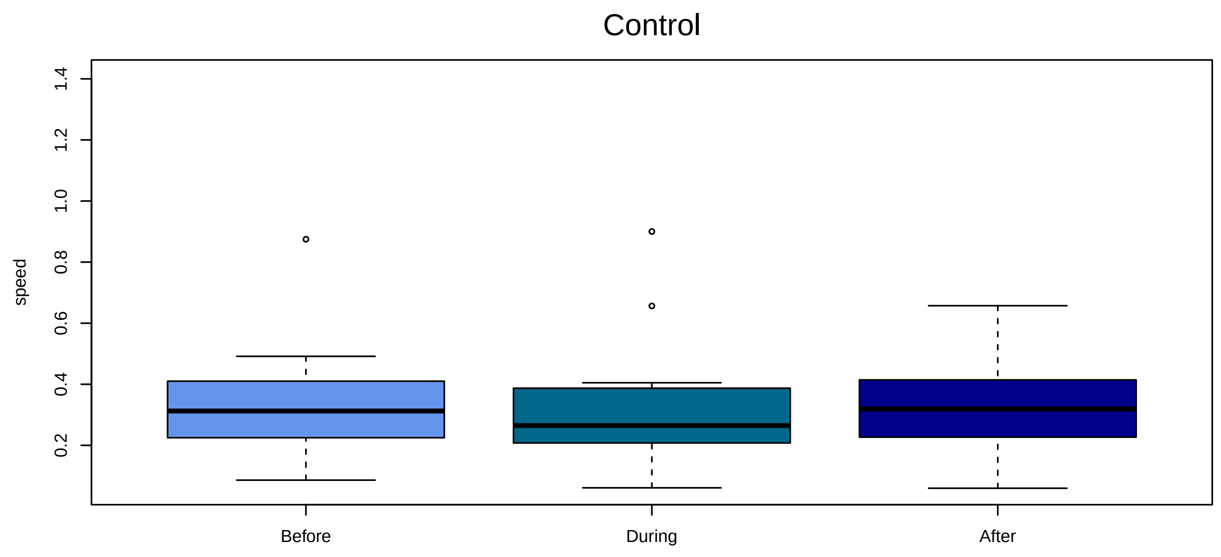
```r
# build formulae
formulae <- paste(c("ruhezeit", "speed", "atem"), "~", "szenario")

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

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


}) -> ylims

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

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

```

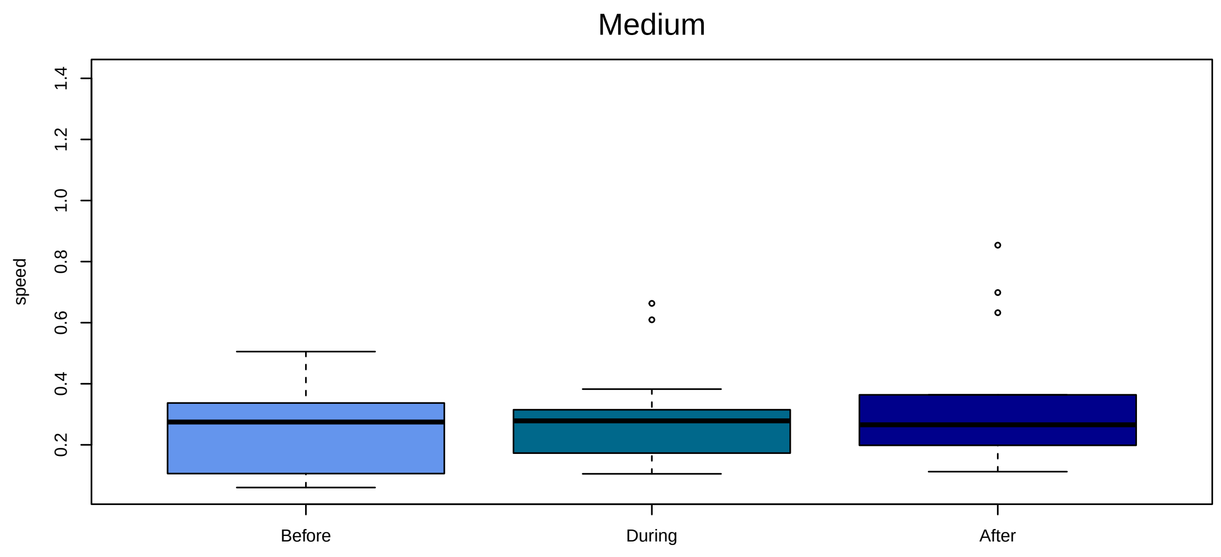
```
    # align
    par(mfrow = c(3, 1), mar = c(2, 4, 4, 2) + 0.1)

    # 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())
```

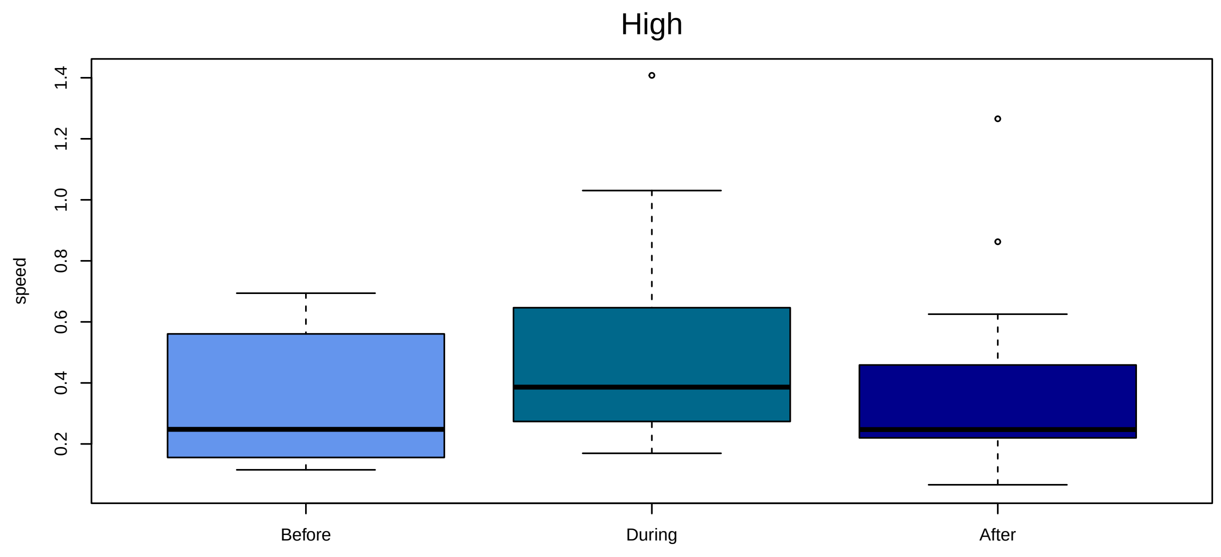

Control
ruhezeit
0.0
0.2
0.4
0.6
0.8
1.0
Before
During
After

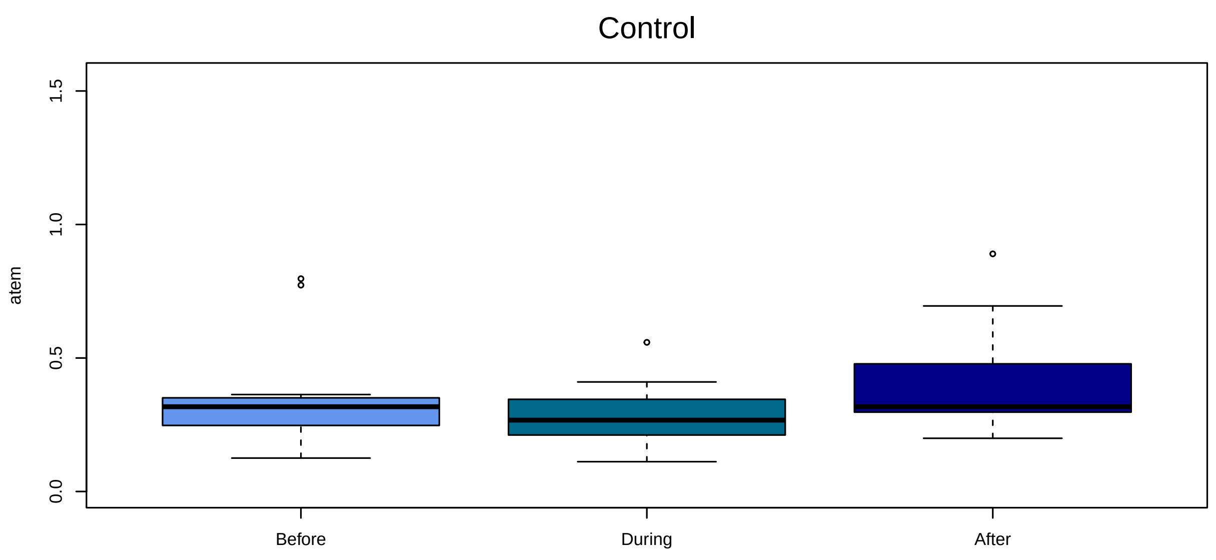

Medium
ruhezeit
0.0
0.2
0.4
0.6
0.8
1.0
Before
During
After

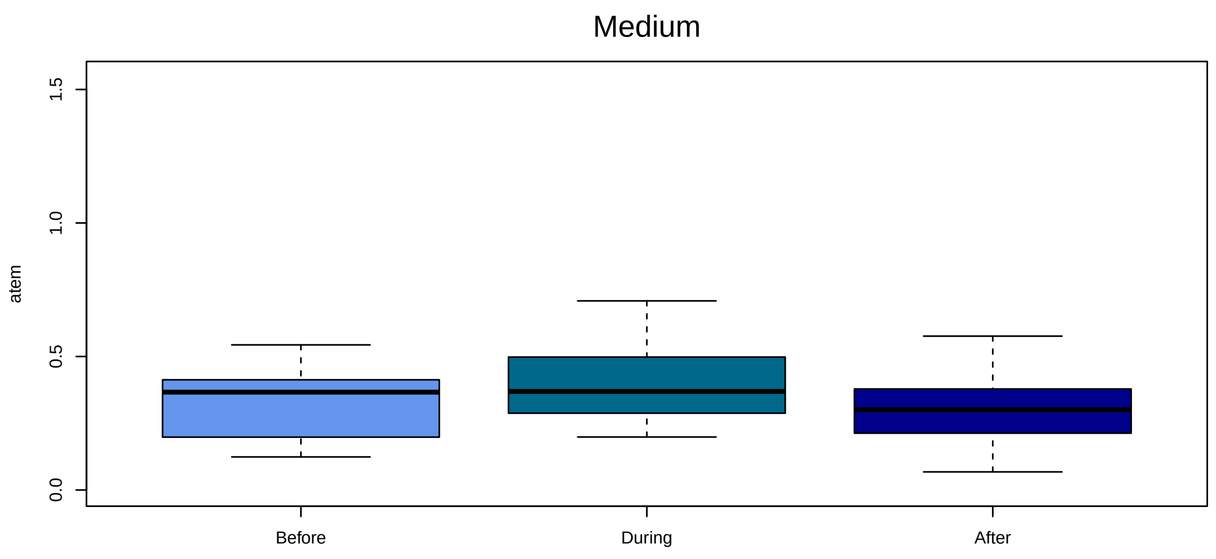

High
ruhezeit
0.0
0.2
0.4
0.6
0.8
1.0
Before
During
After

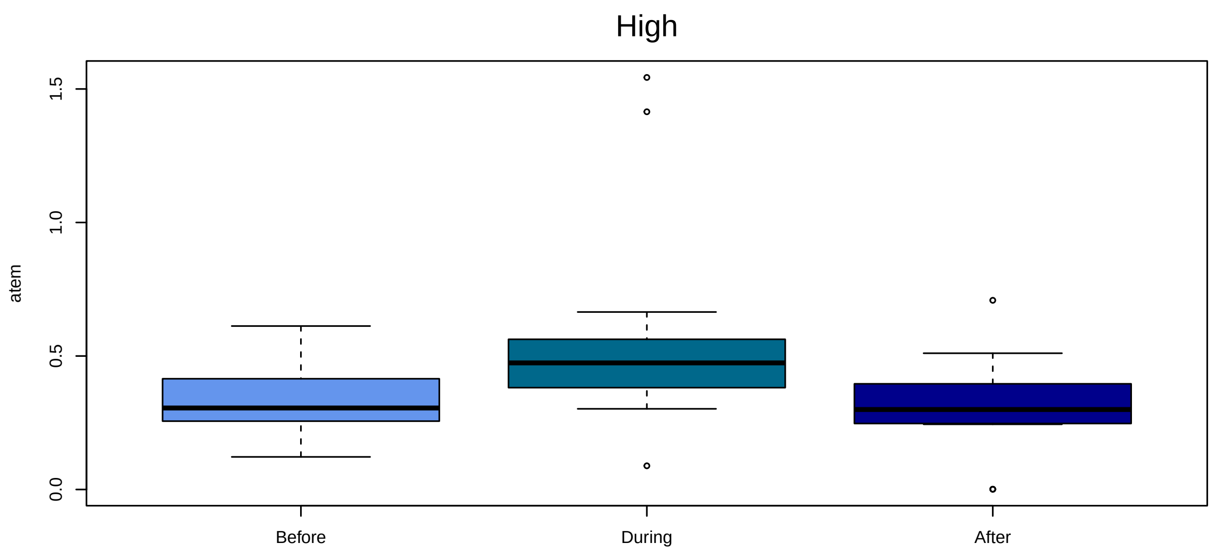

Control
speed
0.2
0.4
0.6
0.8
1.0
1.2
1.4
Before
During
After


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

In the last group of box plots which displays the different noise levels and phases for the mean swim speed (m/s) we observe that for high noise approaches the the mean swim speed seems to increase during the exposure. A similar observation can be made for the respiration rate.

3 Distribution of Covariates

Next we examine the distribution of our dependent variables, even though normally distributed dependent variables do not guarantee well behaved residuals, we may still look at the distribution of our variables and their log transformation. This is especially important because Sprogis, Videsen, and Madsen (2020) use t-tests to evaluate the significance of the estimates of their mixed effects models.

```r
# 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)) -> plots

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

# display
print((plots[[1]] + plots[[2]] + plots[[3]]) /
      (plots[[4]] + plots[[5]] + plots[[6]]))
```

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

We observe that that the log transformation does not yield very promising results in the case of the respiration rate and the proportion of time resting, however, for the mean swim speed the transformed variable is significantly closer to being normally distributed than in its untransformed state.

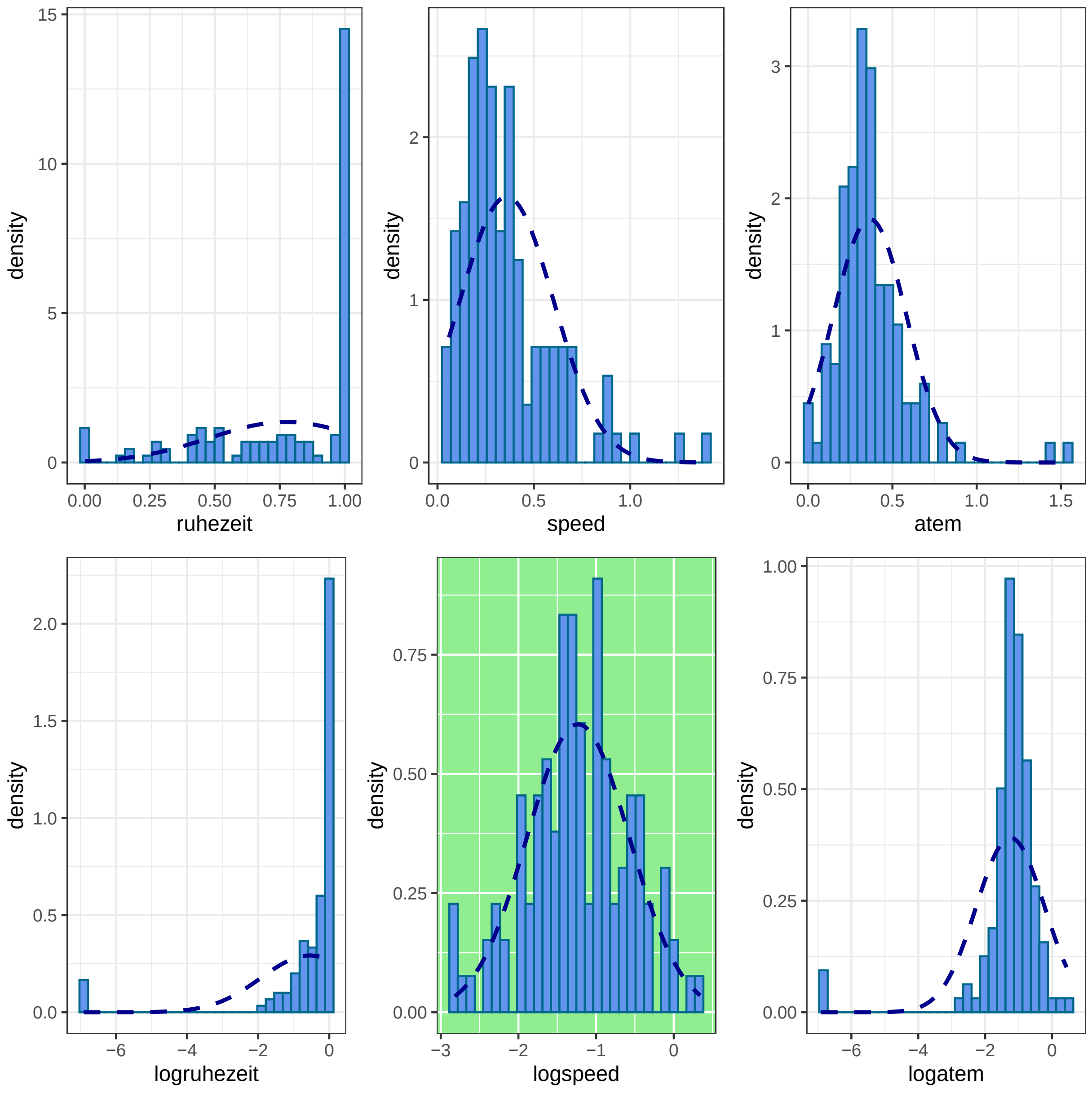
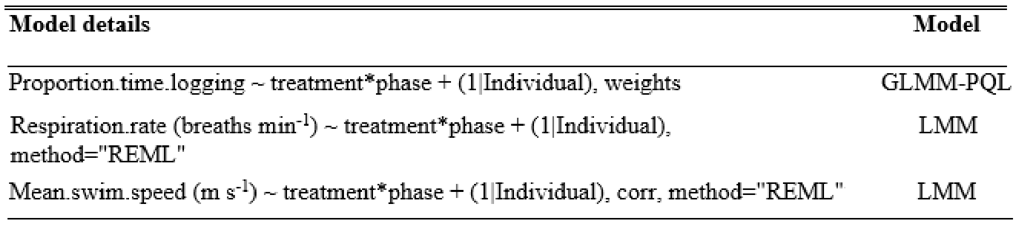


Figure 2: Variables and Transformations

4 Models

Fortunately, Sprogis, Videsen, and Madsen (2020) provide a supplementary file that details the models used. Accordingly, we are able to replicate the specific mixed effects models as described in the figure below. There is one exception, however, the data set that I was provided with does not include the duration of the observations that were used as weights in the GLMM-PQL model.



Model details	Model
Proportion.time.logging ~ treatment*phase + (1|Individual), weights	GLMM-PQL
Respiration.rate (breaths min^{-1}) ~ treatment*phase + (1|Individual), method="REML"	LMM
Mean.swim.speed (m s^{-1}) ~ treatment*phase + (1|Individual), corr, method="REML"	LMM

Figure 3: Models Sprogis et al.

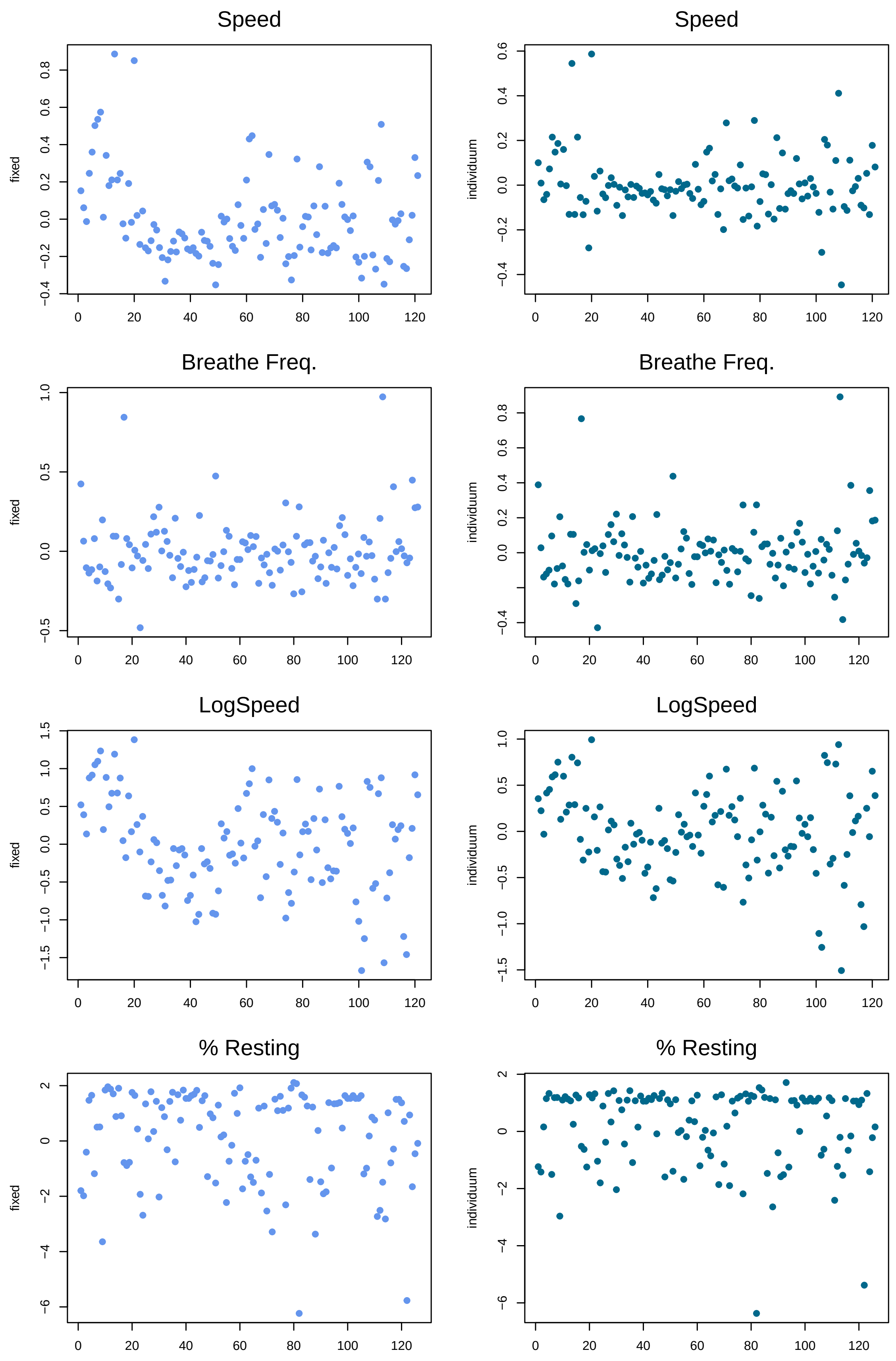
```r
# formulas
formulae_lmm <- c("speed ~ I(treatment) * I(szenario)",
                  "atem ~ I(treatment) * I(szenario)",
                  "logspeed ~ I(treatment) * I(szenario)")

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

# comb
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 { # GLMM

         # 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"))  -> 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)

# write into presentation folder
# saveRDS(fits, "./../Presentation_2/fits.RDS")
# saveRDS(summaries, "./../Presentation_2/summaries.RDS")
```

5 Residual Diagnostics

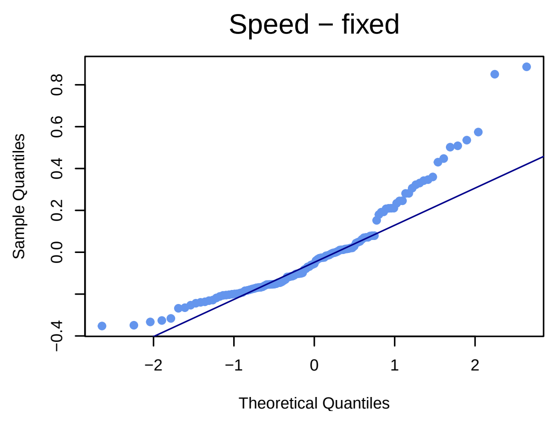
Before taking a look at the results, we first have to check whether the residuals of our models allow us to perform a t-test.

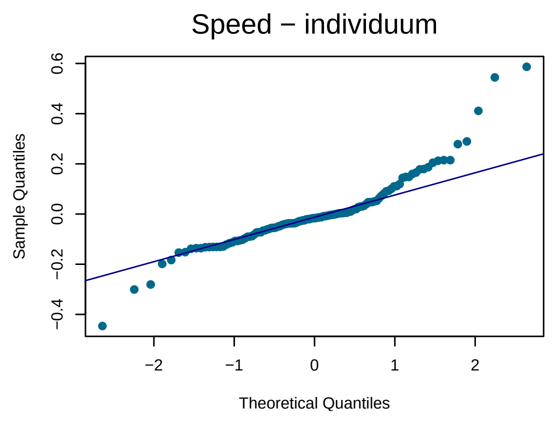
```
# adjust names
nome <- c("Speed", "Breathe Freq.", "LogSpeed", "% Resting")

# 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)
```

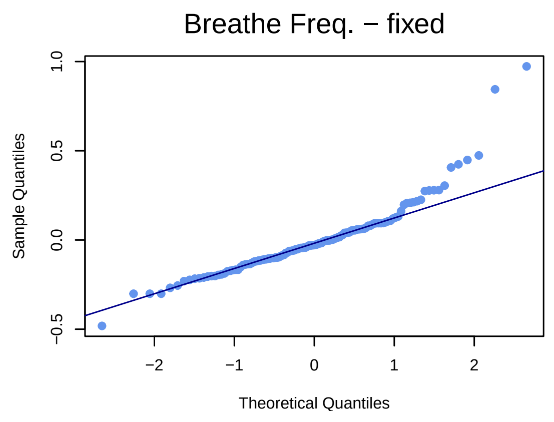
```
    }, c("fixed", "individuum"), c("cornflowerblue", "deepskyblue4"))

}, fits, nome)
```

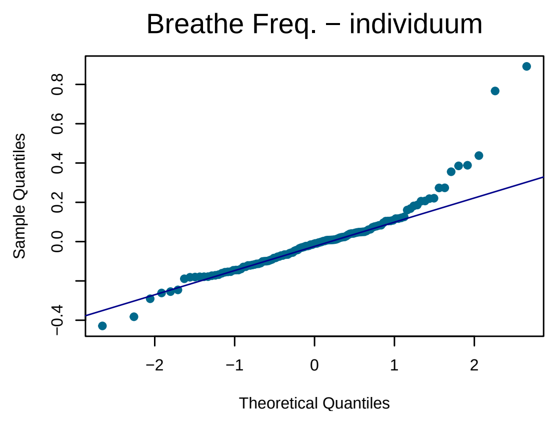

Speed
fixed
Speed
individuum
Breathe Freq.
fixed
Breathe Freq.
individuum
LogSpeed
fixed
LogSpeed
individuum
% Resting
fixed
% Resting
individuum

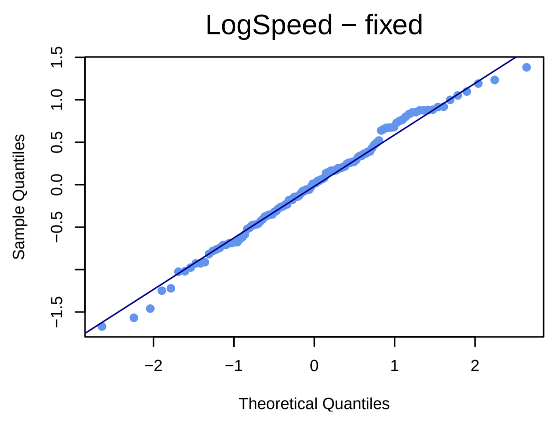
```r
# qqplots
par(mfrow = c(4, 2), mar = c(3.8, 4, 4, 2) + 0.1)

# 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)
```

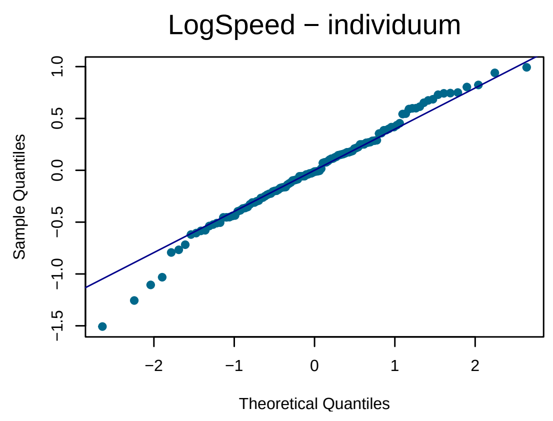

Speed − fixed
Sample Quantiles
Theoretical Quantiles
−0.4
0.0
0.2
0.4
0.6
0.8
−2
−1
0
1
2

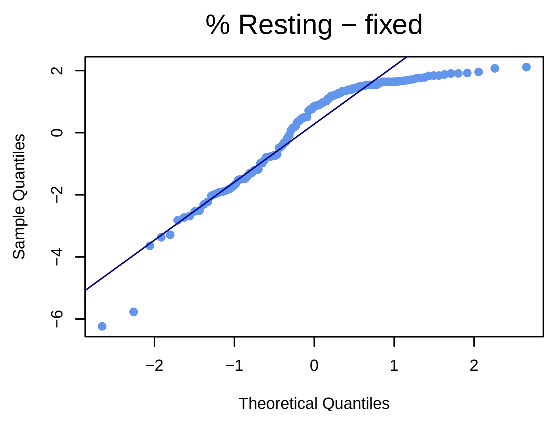

Speed − individuum
Sample Quantiles
Theoretical Quantiles
−0.4
−0.2
0.0
0.2
0.4
0.6
−2
−1
0
1
2

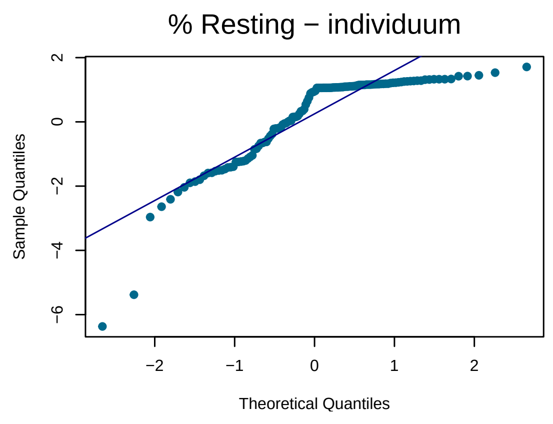

Breathe Freq. − fixed
Sample Quantiles
Theoretical Quantiles
−0.5
0.0
0.5
1.0
−2
−1
0
1
2


Breathe Freq. − individuum
Sample Quantiles
Theoretical Quantiles
−0.4
0.0
0.2
0.4
0.6
0.8
−2
−1
0
1
2


LogSpeed − fixed
Sample Quantiles
Theoretical Quantiles
−1.5
−0.5
0.0
0.5
1.0
1.5
−2
−1
0
1
2


LogSpeed − individuum
Sample Quantiles
Theoretical Quantiles
−1.5
−1.0
−0.5
0.0
0.5
1.0
−2
−1
0
1
2


% Resting − fixed
Sample Quantiles
Theoretical Quantiles
−6
−4
−2
0
2
−2
−1
0
1
2


% Resting − individuum
Sample Quantiles
Theoretical Quantiles
−6
−4
−2
0
2
−2
−1
0
1
2

In the first set of plots we observe a scatter plot of the residuals, both for the fixed and random effects. Those plots allow us to check whether there is a structure in the residuals that would imply heteroskedasticity.Fortunately, the residuals do not seem to have a such a structure.

The second set of plots provides us with information about the distribution of residuals, again the plots are split into the fixed effects residuals on the left and the random effects residuals on the right. The deviation of the dots from the darkblue line for all cases except for the logarithm of mean swim speed (m/s), indicate that the residuals are not sufficiently close to being normally distributed. Unfortunately, this makes the inference presented in the publication by Sprogis, Videsen, and Madsen (2020) invalid, as they make their conclusions based on the p-value obtained from the t-test of their estimates. The model that uses the log-transformed mean of swim speed, is not part of the publication and was estimated in hopes of well behaved residuals. Fortunately, we may therefore interpret the potential significance of the model parameters using a t-test. In the remaining cases this is unfortunately not possible.

6 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|
```

7 Conclusion

The ouput of the fitted models contain two factors, the treatment and the scenario. The treatment factor represents the different noise levels, the base level for this factor is *low*. The scenario factor represents the three different phases of observation, the base level is *before* in this case. As previously mentioned, only the model of log-transformed mean of swim speed fulfills the required assumptions for an interpretation of the estimate's significance based on a t-test. When fixing the type I error to 5%, i.e. setting the significance level to $\alpha = 0.05$, we find that the interaction between *high noise* and *during* is statistically significant $p = 0.0328$. We may thus conclude, that the log-transformed speed is significantly higher during the boat approaches with a high noise level when compared to the base level *low:before*.

For the remaining models one could attempt to bootstrap test statistics or confidence intervals for the model's estimates. Unfortunately, the function `confint()` which works for the package *lme4* does not (yet) provide the bootstrapped confidence intervals for the packages used in the publication of sprogis et al., i.e. *nlme* and *MASS*. The implementation of inference via bootstrapping is beyond the scope of this report, however, upon execution of `confint()` with an *nlme* object as an argument, the resulting Error contains a message that states that the functionality will soon be available for *nlme* objects.

Literature

Sprogis, Kate R, Simone Videsen, and Peter T Madsen. 2020. “Vessel Noise Levels Drive Behavioural Responses of Humpback Whales with Implications for Whale-Watching.” Edited by Christian Rutz, Rosalyn Gloag, and Patrick Miller. *eLife* 9 (June): e56760. https://doi.org/10.7554/eLife.56760.