Please note that the document was prepared as a Rmarkdown file. For optimally reading this file, please copy the code in a R document and open as Rmarkdown in Rstudio.

---

title: "R Script for the Data Analysis from the Manuscript 'Behavioural stress propagation in benthic invertebrates caused by acute pH drop-induced metabolites'"

author: "Lauric Feugere"

output:

  pdf_document:

  word_document:

  html_document:

    fig_width: 8

    fig_height: 6

    toc_float: yes

    toc: yes

  html_notebook:

    toc: yes

    fig_caption: yes

    number_sections: yes

    fig_width: 8

    fig_height: 6

  chunk_output_type: inline

editor_options:

  chunk_output_type: console

```{r, setup, include=FALSE}

knitr::opts_chunk$set(message = FALSE, warning=F)

```

Preparing the R Environment

The present R code was created to analyse the data from the manuscript **Behavioural stress propagation in benthic invertebrates caused by acute pH drop-induced metabolites**

Lauric Feugere^1^, Lauren Angell^1^, James Fagents^1^, Rebecca Nightingale^1^, Kirsty Rowland^1^, Saffiyah Skinner^1^, Jorg Hardege^1^, Helga Bartels-Hardege^1^, Katharina C. Wollenberg Valero^1^*

^1^ Department of Biological and Marine Sciences, University of Hull, Cottingham Road, Kingston-upon-Hull HU67RX, United Kingdom.

*Corresponding author: k.wollenberg-valero@hull.ac.uk, ORCID: 0000-0001-8858-1804

R packages

First, all the necessary R packages are installed and loaded into R.

```{r Package set up, echo=TRUE, warning=FALSE}
if(!require(ggplot2)){install.packages("ggplot2")}; library(ggplot2)
if(!require(rcompanion)){install.packages("rcompanion")}; library(rcompanion)
if(!require(bestNormalize)){install.packages("bestNormalize")}; library(bestNormalize)
if(!require(effsize)) install.packages("effsize"); library(effsize)
if(!require(emmeans)){install.packages("emmeans")}; library(emmeans)
if(!require(car)){install.packages("car")}; library(car)
if(!require(gnm)){install.packages("gnm")}; library(gnm)
if(!require(viridis)){install.packages("viridis")}; library(viridis)
if(!require(devtools)){install.packages("devtools")}; library(devtools)
if(!require(lme4)){install.packages("lme4")}; library(lme4)
if(!require(arm)){install.packages("arm")}; library(arm)
if(!require(survival)){install.packages("survival")}; library(survival)
if(!require(survminer)){install.packages("survminer")}; library(survminer)
if(!require(tidyr)){install.packages("tidyr")}; library(tidyr)
if(!require(finalfit)){install.packages("finalfit")}; library(finalfit)
if(!require(broom)){install.packages("broom")}; library(broom)
if(!require(knitr)){install.packages("knitr")}; library(knitr)
if(!require(cowplot)){install.packages("cowplot")}; library(cowplot)
devtools::install_github("zabore/ezfun"); library(ezfun)
if(!require(grid)){install.packages("grid")}; library(grid)
if(!require(gridExtra)){install.packages("gridExtra")}; library(gridExtra)
if(!require(sjPlot)){install.packages("sjPlot")}; library(sjPlot)
if(!require('ggfortify'))install.packages("ggfortify", repos="http://cran.us.r-project.org"); library(ggfortify)
if(!require('kableExtra'))install.packages("kableExtra", repos="http://cran.us.r-project.org"); library(kableExtra)
knitr::opts_chunk$set(echo = TRUE)

own_theme <- function () {
  theme(axis.title.y = element_text(face='bold', colour='black', size=15 , margin = margin(t = 0, r = 5, b = 0, l = 0)),
        axis.text.x=element_text(face='bold', size=15, colour = "black"),
        axis.text.y=element_text(face='bold', size=15, colour = "black" ),
        axis.line = element_line(size = 0.5, linetype = "solid"),
        axis.ticks.x = element_blank(),
        axis.ticks = element_line(size=0.25, colour = "black"),
        axis.ticks.length.y = unit(0.25, "cm"))+
        theme(plot.background = element_rect(color = "black"))+
```

```
  theme(legend.position="top")+
  theme(legend.text = element_text(face='bold', colour='black', size=15 ))+
  theme(legend.title = element_text(face='bold', colour='black', size=15 ))+
  theme(axis.line.x = element_blank())+
  theme(legend.background = element_rect(fill="transparent")) +
    theme(plot.title = element_text(hjust = 0.5))
  }

if(!require('ggthemes'))install.packages("ggthemes", repos="http://cran.us.r-project.org"); library(ggthemes)

own_theme_hist <- function () {
  theme_classic()+
  theme(axis.title = element_text(face='bold', colour='black', size=15 , margin = margin(t = 0, r = 5, b = 0, l = 0)),
        axis.text = element_text(face='bold', size=15, colour = "black" ),
        axis.line = element_line(size = 1, linetype = "solid"),
         axis.ticks = element_line(size=0.25, colour = "black"),
        axis.ticks.length = unit(0.25, "cm"))+
        theme(plot.background = element_rect(color = "black"))+
  theme(legend.position="top")+
  theme(legend.text = element_text(face='bold', colour='black', size=15 ))+
  theme(legend.title = element_text(face='bold', colour='black', size=15 ))+
  theme(legend.background = element_rect(fill="transparent")) +
    theme(plot.title = element_text(hjust = 0.5))
  }

```

Importing the Data

To find the data, please follow the link to the repository given in the manuscript. There csv files need to be extracted from the different sheets of the main Excel file before running this R script.

```{r table_dataset_list, echo=TRUE, warning=FALSE}

description_datasets <- data.frame("Dataset"=c("Dpug.csv", "Cmae.csv", "Hdiv.csv", "Conc_Dpug.csv", "Conc_Hdiv.csv"), "Description"=c(
    "Behavioural data of Diogenes pugilator (Dpug) in a factorial design of pH drop x Metabolites x Donor",
    "Behavioural data of Carcinus maenas (Cmae) in a factorial design of pH drop x Metabolites x Donor",
    "Behavioural data of Hediste diversicolor (Hdiv) in a factorial design of pH drop x Metabolites x Donor",
    "Behavioural data of Diogenes pugilator (Dpug) in CM, SM, and CM100%",
```

```
   "Behavioural data of Hediste diversicolor (Hdiv) in CM, SM, and CM100%"
  ))



description_datasets %>%
  kbl(digits = 4,caption = "Description of the datasets") %>%
  kable_classic(full_width = F, html_font = "Cambria")



abbreviations_table <- data.frame("Abbreviation"=c("CM", "pH drop", "SM", "pH drop+SM",  "heter","Cons.","Saur", "Cmae or SC", "Dpug or HC", "Hdiv or RW"), "Description"=c("Control Metabolites at pH 8.1 (control or regular pH)", "Control Metabolites at pH 7.6 (pH drop)", "Stress Metabolites at pH 8.1", "Stress Metabolites at pH 7.6",  "heterospecific donor","conspecific donor","gilt-head Sea Bream Sparus aurata", "Green Shore Crab Carcinus maenas", "Small hermit crab Diogenes pugilator", "Harbour ragworm Hediste (Nereis) diversicolor" ))


abbreviations_table %>%
  kbl(digits = 4,caption = "List of Abbreviations used in the datasets") %>%
  kable_classic(full_width = F, html_font = "Cambria")

```
```

The following line of code imports the datasets. Please use the name of the dataset accordingly to the exported csv file names.

```{r Data_import, echo=TRUE, warning=FALSE}

Cmae <- read.csv("Cmae.csv")
Hdiv <- read.csv("Hdiv.csv")
Dpug  <- read.csv("Dpug.csv")
Conc_Hdiv <- read.csv("Conc_Hdiv.csv")
Conc_Dpug <- read.csv("Conc_Dpug.csv")

```

The predictors (pH, metabolites) and covariates (donor, recipient) were assigned as factors. In addition, the binary data was also assigned as factors to be used in generalised linear models for binomial data.

```{r factorising predictors, echo=TRUE, warning=FALSE}
Hdiv$Treatment <- as.factor(Hdiv$Treatment) # treatments (pH drop x metabolites)
Hdiv$Treatment2 <- as.factor(Hdiv$Treatment2) # treatments (pH drop x metabolites x Donor)
Hdiv$pH <- as.factor(Hdiv$pH) # predictor (pH values)
Hdiv$Metabolites <- as.factor(Hdiv$Metabolites) # predictor (control vs stress metabolites)
Hdiv$pH_drop <- as.factor(Hdiv$pH_drop) # binary predictor 1 - pH drop = 0 means 'control pH of 8.1'; pH drop = 1 means 'pH drop to pH = 7.6'
```
```

```
Hdiv$SM <- as.factor(Hdiv$SM) # binary predictor 2 - Stress metabolites. SM = 0 means control metabolites from unstressed animals, SM = 1 means stress metabolites from stressed animals

Hdiv$Donor <- as.factor(Hdiv$Donor) # binary Predictor 3 - metabolites come from either a conspecific or from the heterospecific (Sparus aurata)

Hdiv$Avoidance <- as.factor(Hdiv$Avoidance) # binary data

Hdiv$Success <- as.factor(Hdiv$Success) # binary data

Hdiv$time_s <- as.numeric(Hdiv$time_s) # ensure that the time is numeric

Hdiv$Donor_heter <- as.factor(Hdiv$Donor_heter) # origin of metabolites is from conspecific or heterospecifics (S. aurata)



Dpug$Treatment <- as.factor(Dpug$Treatment) # treatments (pH drop x metabolites)

Dpug$Treatment2 <- as.factor(Dpug$Treatment2) #  treatments (pH drop x metabolites x Donor)

Dpug$pH <- as.factor(Dpug$pH) # predictor (pH values)

Dpug$Metabolites <- as.factor(Dpug$Metabolites) # predictor (control vs stress metabolites)

Dpug$pH_drop <- as.factor(Dpug$pH_drop) # binary predictor 1 - pH drop = 0 means 'control pH of 8.1'; pH drop = 1 means 'pH drop to pH = 7.6'

Dpug$SM <- as.factor(Dpug$SM) # binary predictor 2 - Stress metabolites. SM = 0 means control metabolites from unstressed animals, SM = 1 means stress metabolites from stressed animals

Dpug$Donor <- as.factor(Dpug$Donor) # binary Predictor 3 - metabolites come from either a conspecific or from the heterospecific (Sparus aurata)

Dpug$Success <- as.factor(Dpug$Success) # binary data

Dpug$Freeze <- as.factor(Dpug$Freeze) # binary data

Dpug$Avoidance <- as.factor(Dpug$Avoidance) # binary data

Dpug$time_s <- as.numeric(Dpug$time_s) # ensure that the time is numeric

Dpug$Donor_heter <- as.factor(Dpug$Donor_heter) # origin of metabolites is from conspecific or heterospecifics (S. aurata)



Cmae$Treatment <- as.factor(Cmae$Treatment) # treatments (pH drop x metabolites)

Cmae$Treatment2 <- as.factor(Cmae$Treatment2) # treatments (pH drop x metabolites x Donor)

Cmae$pH <- as.factor(Cmae$pH) # predictor (pH values)

Cmae$Metabolites <- as.factor(Cmae$Metabolites) # predictor (control vs stress metabolites)

Cmae$pH_drop <- as.factor(Cmae$pH_drop) # binary predictor 1 - pH drop = 0 means 'control pH of 8.1'; pH drop = 1 means 'pH drop to pH = 7.6'

Cmae$SM <- as.factor(Cmae$SM) # binarypredictor 2 - Stress metabolites. SM = 0 means control metabolites from unstressed animals, SM = 1 means stress metabolites from stressed animals

Cmae$Donor <- as.factor(Cmae$Donor) # binary Predictor 3 - metabolites come from either a conspecific or from the heterospecific (Sparus aurata)

Cmae$Success <- as.factor(Cmae$Success) # binary data

Cmae$Freeze <- as.factor(Cmae$Freeze) # binary data

Cmae$Avoidance <- as.factor(Cmae$Avoidance) # binary data

Cmae$time_s <- as.numeric(Cmae$time_s) # ensure that the time is numeric

Cmae$Donor_heter <- as.factor(Cmae$Donor_heter) # origin of metabolites is from conspecific or heterospecifics (S. aurata)
```

```
Conc_Hdiv$Treatment <- as.factor(Conc_Hdiv$Treatment) # treatments

Conc_Hdiv$Treatment2 <- as.factor(Conc_Hdiv$Treatment2) # treatments

Conc_Hdiv$Donor <- as.factor(Conc_Hdiv$Donor)

Conc_Hdiv$pH <- as.factor(Conc_Hdiv$pH)

Conc_Hdiv$Metabolites <- as.factor(Conc_Hdiv$Metabolites)

Conc_Hdiv$pH_drop <- as.factor(Conc_Hdiv$pH_drop)

Conc_Hdiv$SM <- as.factor(Conc_Hdiv$SM)

Conc_Hdiv$Success <- as.factor(Conc_Hdiv$Success)  # binary data

Conc_Hdiv$Avoidance <- as.factor(Conc_Hdiv$Avoidance)  # binary data

Conc_Hdiv$time_s <- as.numeric(Conc_Hdiv$time_s) # ensure that the time is numeric


Conc_Dpug$Treatment <- as.factor(Conc_Dpug$Treatment)

Conc_Dpug$Treatment2 <- as.factor(Conc_Dpug$Treatment2)

Conc_Dpug$Donor <- as.factor(Conc_Dpug$Donor)

Conc_Dpug$pH <- as.factor(Conc_Dpug$pH)

Conc_Dpug$Metabolites <- as.factor(Conc_Dpug$Metabolites)

Conc_Dpug$pH_drop <- as.factor(Conc_Dpug$pH_drop)

Conc_Dpug$SM <- as.factor(Conc_Dpug$SM)

Conc_Dpug$Success <- as.factor(Conc_Dpug$Success)  # binary data

Conc_Dpug$Freeze <- as.factor(Conc_Dpug$Freeze)  # binary data

Conc_Dpug$Avoidance <- as.factor(Conc_Dpug$Avoidance)  # binary data

Conc_Dpug$time_s <- as.numeric(Conc_Dpug$time_s)   # ensure that the time is numeric



```

The analysis was conducted on the factorial design (pH x Metabolites x donor x recipient) (referred to as '2x2x2x3_Factorial_Design') in the 'Data' table. Additional statistical tests were conducted on the condition 'CM100%'.

Preparing the data

Behaviour data (factorial design)

The binary data was analysed for all specimens. The time-to-success data was analysed as a survival analysis (aka time-to-event) for specimens that reached the feeding cue (crabs) or that burrowed the head (harbour ragworm) while accounting for censoring of the animals that did not complete the experiment within 300 seconds. The data was analysed using a model of pH drop x Stress Metabolites x Donor for each recipient species. Posthoc tests were used to see the differences in each donor/recipient subgroups. Therefore, the dataset is split into subset for each analysis, that is, by type of behaviour (time-to-success, avoidance, success), by recipient species, and by metabolite donor. In addition, the following code chunk recodes the binary data in a new column so that barplots could be created using the ggplot2 package to show the 'yes' and 'no' responses while being able to change the colours.

```{r Subsets, echo=TRUE, warning=FALSE}

Cmae.Time <- subset(Cmae, is.na(Cmae$time_s)==F & is.na(Cmae$Success)==F) # Create Time dataset for Cmae. Remove NA

Dpug.Time <- subset(Dpug, is.na(Dpug$time_s)==F & is.na(Dpug$Success)==F) # Create Time dataset for Dpug. Remove NA

Hdiv.Time <- subset(Hdiv, is.na(Hdiv$time_s)==F & is.na(Hdiv$Success)==F) # Create Time dataset for Hdiv. Remove NA


# For each binary variable (avoidance, escaping, freezing), a new column was coded as "group_NO" and "group_YES", for the treatments, the recipients, and the donors.


# Dpug


# Avoidance data

Dpug.Avoid <- subset(Dpug, is.na(Dpug$Avoidance)==F) # Create avoidance dataset for Dpug. NA to remove to have success data


# Avoidance Per treatment

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==0 & Dpug.Avoid$Avoidance==0] <- "1CM_NO"

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==0 & Dpug.Avoid$Avoidance==1] <- "2CM_YES"

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==0 & Dpug.Avoid$Avoidance==0] <- "3pH_drop_NO"

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==0 & Dpug.Avoid$Avoidance==1] <- "4pH_drop_YES"

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==1 & Dpug.Avoid$Avoidance==0] <- "5SM_NO"

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==1 & Dpug.Avoid$Avoidance==1] <- "6SM_YES"

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==1 & Dpug.Avoid$Avoidance==0] <- "7pH_dropSM_NO"

Dpug.Avoid$avoid_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==1 & Dpug.Avoid$Avoidance==1] <- "8pH_dropSM_YES"


# Escaping Per treatment

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==0 & Dpug.Avoid$Escape==0] <- "1CM_NO"

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==0 & Dpug.Avoid$Escape==1] <- "2CM_YES"

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==0 & Dpug.Avoid$Escape==0] <- "3pH_drop_NO"

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==0 & Dpug.Avoid$Escape==1] <- "4pH_drop_YES"

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==1 & Dpug.Avoid$Escape==0] <- "5SM_NO"

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==1 & Dpug.Avoid$Escape==1] <- "6SM_YES"

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==1 & Dpug.Avoid$Escape==0] <- "7pH_dropSM_NO"

Dpug.Avoid$escape_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==1 & Dpug.Avoid$Escape==1] <- "8pH_dropSM_YES"


# Freezing Per treatment

Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==0 & Dpug.Avoid$Freeze==0] <- "1CM_NO"

Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==0 & Dpug.Avoid$Freeze==1] <- "2CM_YES"

Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==0 & Dpug.Avoid$Freeze==0] <- "3pH_drop_NO"

Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==0 & Dpug.Avoid$Freeze==1] <- "4pH_drop_YES"

Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==1 & Dpug.Avoid$Freeze==0] <- "5SM_NO"

Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==0 & Dpug.Avoid$SM==1 & Dpug.Avoid$Freeze==1] <- "6SM_YES"

Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==1 & Dpug.Avoid$Freeze==0] <- "7pH_dropSM_NO"
```

```
Dpug.Avoid$freeze_treat[Dpug.Avoid$pH_drop==1 & Dpug.Avoid$SM==1 & Dpug.Avoid$Freeze==1] <- "8pH_dropSM_YES"

# Cmae

# Avoidance data
Cmae.Avoid <- subset(Cmae, is.na(Cmae$Avoidance)==F) # NA to remove to have success data

# Avoidance Per treatment
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==0 & Cmae.Avoid$Avoidance==0] <- "1CM_NO"
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==0 & Cmae.Avoid$Avoidance==1] <- "2CM_YES"
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==0 & Cmae.Avoid$Avoidance==0] <- "3pH_drop_NO"
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==0 & Cmae.Avoid$Avoidance==1] <- "4pH_drop_YES"
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==1 & Cmae.Avoid$Avoidance==0] <- "5SM_NO"
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==1 & Cmae.Avoid$Avoidance==1] <- "6SM_YES"
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==1 & Cmae.Avoid$Avoidance==0] <- "7pH_dropSM_NO"
Cmae.Avoid$avoid_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==1 & Cmae.Avoid$Avoidance==1] <- "8pH_dropSM_YES"

# Escaping Per treatment
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==0 & Cmae.Avoid$Escape==0] <- "1CM_NO"
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==0 & Cmae.Avoid$Escape==1] <- "2CM_YES"
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==0 & Cmae.Avoid$Escape==0] <- "3pH_drop_NO"
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==0 & Cmae.Avoid$Escape==1] <- "4pH_drop_YES"
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==1 & Cmae.Avoid$Escape==0] <- "5SM_NO"
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==1 & Cmae.Avoid$Escape==1] <- "6SM_YES"
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==1 & Cmae.Avoid$Escape==0] <- "7pH_dropSM_NO"
Cmae.Avoid$escape_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==1 & Cmae.Avoid$Escape==1] <- "8pH_dropSM_YES"

# Freezing Per treatment
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==0 & Cmae.Avoid$Freeze==0] <- "1CM_NO"
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==0 & Cmae.Avoid$Freeze==1] <- "2CM_YES"
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==0 & Cmae.Avoid$Freeze==0] <- "3pH_drop_NO"
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==0 & Cmae.Avoid$Freeze==1] <- "4pH_drop_YES"
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==1 & Cmae.Avoid$Freeze==0] <- "5SM_NO"
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==0 & Cmae.Avoid$SM==1 & Cmae.Avoid$Freeze==1] <- "6SM_YES"
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==1 & Cmae.Avoid$Freeze==0] <- "7pH_dropSM_NO"
Cmae.Avoid$freeze_treat[Cmae.Avoid$pH_drop==1 & Cmae.Avoid$SM==1 & Cmae.Avoid$Freeze==1] <- "8pH_dropSM_YES"

# Hdiv
# Avoidance data
Hdiv.Avoid <- subset(Hdiv, is.na(Hdiv$Avoidance)==F) # NA to remove to have success data
```

```
# Avoidance Per treatment
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==0 & Hdiv.Avoid$SM==0 & Hdiv.Avoid$Avoidance==0] <- "1CM_NO"
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==0 & Hdiv.Avoid$SM==0 & Hdiv.Avoid$Avoidance==1] <- "2CM_YES"
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==1 & Hdiv.Avoid$SM==0 & Hdiv.Avoid$Avoidance==0] <- "3pH_drop_NO"
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==1 & Hdiv.Avoid$SM==0 & Hdiv.Avoid$Avoidance==1] <- "4pH_drop_YES"
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==0 & Hdiv.Avoid$SM==1 & Hdiv.Avoid$Avoidance==0] <- "5SM_NO"
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==0 & Hdiv.Avoid$SM==1 & Hdiv.Avoid$Avoidance==1] <- "6SM_YES"
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==1 & Hdiv.Avoid$SM==1 & Hdiv.Avoid$Avoidance==0] <- "7pH_dropSM_NO"
Hdiv.Avoid$avoid_treat[Hdiv.Avoid$pH_drop==1 & Hdiv.Avoid$SM==1 & Hdiv.Avoid$Avoidance==1] <- "8pH_dropSM_YES"



# Subset for each Donor subgroup
Cmae.Cmae.Time <- subset(Cmae.Time, Cmae.Time$Donor=="Cmae")
Saur.Cmae.Time <- subset(Cmae.Time, Cmae.Time$Donor=="Saur")
Cmae.Cmae.Avoid <- subset(Cmae.Avoid, Cmae.Avoid$Donor=="Cmae")
Saur.Cmae.Avoid <- subset(Cmae.Avoid, Cmae.Avoid$Donor=="Saur")

Dpug.Dpug.Time <- subset(Dpug.Time, Dpug.Time$Donor=="Dpug")
Saur.Dpug.Time <- subset(Dpug.Time, Dpug.Time$Donor=="Saur")
Dpug.Dpug.Avoid <- subset(Dpug.Avoid, Dpug.Avoid$Donor=="Dpug")
Saur.Dpug.Avoid <- subset(Dpug.Avoid, Dpug.Avoid$Donor=="Saur")

Hdiv.Hdiv.Time <- subset(Hdiv.Time, Hdiv.Time$Donor=="Hdiv")
Saur.Hdiv.Time <- subset(Hdiv.Time, Hdiv.Time$Donor=="Saur")
Hdiv.Hdiv.Avoid <- subset(Hdiv.Avoid, Hdiv.Avoid$Donor=="Hdiv")
Saur.Hdiv.Avoid <- subset(Hdiv.Avoid, Hdiv.Avoid$Donor=="Saur")

```

Statistical Analysis of Factorial Design of pH drop x Stress Metabolites x Donor

The data was analysed in two levels. First, all recipient species were analysed by including metabolite donors as a cofactor to understand the overall effects of pH and metabolites. Second, the data was split by recipient species and by metabolite donor.

Dpug

Avoidance

The binomial generalised linear model is used to model the effects of predictors on the avoidance response of Dpug.

```{r Dpug avoidance glm, echo=TRUE, warning=FALSE}

glz.Dpug.avoid.null <- glm(Avoidance ~ 1, data=Dpug.Avoid,family=binomial(link = "logit")) # null model

glz.Dpug.avoid <- glm(Avoidance ~ pH_drop*SM*Donor, data=Dpug.Avoid,family=binomial(link = "logit")) # basic model with three factors

glz.Dpug.avoid.mod2 <- glm(Avoidance ~ pH_drop*SM*Donor+Water_use, data=Dpug.Avoid,family=binomial(link = "logit")) #  include covariate 'water use'

glz.Dpug.avoid.mod3 <- glm(Avoidance ~ pH_drop*SM*Donor+Water_use+Crab_size, data=Dpug.Avoid,family=binomial(link = "logit")) # include covariate 'crab size'

anova(glz.Dpug.avoid.null, glz.Dpug.avoid, glz.Dpug.avoid.mod2,glz.Dpug.avoid.mod3, test="Chisq")

glz.Dpug.avoid.anova <- as.data.frame(anova(glz.Dpug.avoid.null, glz.Dpug.avoid, glz.Dpug.avoid.mod2,glz.Dpug.avoid.mod3, test="Chisq")) # The water use and the crab size can be disregarded. Avoidance responses were observed only in Autumn 2019 (not in Autumn 2018). The conserved model is glz.Dpug.avoid


glz.Dpug.avoid.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of avoidance in Dpug"  ) %>%

  kable_classic(full_width = F, html_font = "Cambria")



paste("AIC of glz.Dpug.avoid model = ", AIC(glz.Dpug.avoid)) # AIC

as.data.frame(summary(glz.Dpug.avoid)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of binomial generalised linear model for the avoidance in Dpug" ) %>%

  kable_classic(full_width = F, html_font = "Cambria")


# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

# pH*SM*Donor term - posthoc tests - Run the full posthoc matrix regardless of whether the interaction term is significant in the model


as.data.frame(emmeans( glz.Dpug.avoid, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the avoidance in Dpug - model term (pH drop:Stress metabolites)|donor"  ) %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Dpug.avoid, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the avoidance in Dpug - model term (pH drop:Stress metabolites)|donor"  ) %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(summary(emmeans( glz.Dpug.avoid, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the avoidance in Dpug - model term donor|(pH drop:Stress metabolites)"  ) %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Dpug.avoid, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%
```

```
  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the avoidance in Dpug - model term donor|(pH drop:Stress metabolites)"  ) %>%
  kable_classic(full_width = F, html_font = "Cambria")

plot_model(glz.Dpug.avoid, type = "pred", terms = c("pH_drop", "SM", "Donor"))
glz.Dpug.avoid_model_data <- as.data.frame(get_model_data(glz.Dpug.avoid, type = "pred", terms = c("pH_drop", "SM", "Donor")))
colnames(glz.Dpug.avoid_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')
glz.Dpug.avoid_model_data %>%
  kbl(digits = 4,caption = "Predicted model values for the avoidance in Dpug according to the generalised linear model"  ) %>%
  kable_classic(full_width = F, html_font = "Cambria")
```

The percentages of avoidance/no avoidance behaviours are retrieved for each group.

```
```{r Dpug/Dpug avoidance percent, echo=TRUE, warning=FALSE}

# Dpug/Dpug

Dpug.Dpug.no.Avoid <- c(
  round(length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="CM" & Dpug.Dpug.Avoid$Avoid==0])/length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="CM"])*100, 0),
  round(length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="pH_drop" & Dpug.Dpug.Avoid$Avoid==0])/length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="SM" & Dpug.Dpug.Avoid$Avoid==0])/length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="SM"])*100, 0),
  round(length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="pH_drop+SM" & Dpug.Dpug.Avoid$Avoid==0])/length(Dpug.Dpug.Avoid$Avoid[Dpug.Dpug.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Dpug.Dpug.yes.Avoid <- 100-Dpug.Dpug.no.Avoid

Dpug.Dpug.Avoid.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Avoidance"= c(Dpug.Dpug.yes.Avoid),
          "No Avoidance"=c(Dpug.Dpug.no.Avoid))

Dpug.Dpug.Avoid.percent %>%
  kbl(digits = 4,caption = "Effect of treatment on percentage values of Avoidance/No avoidance for Dpug/Dpug"  ) %>%
  kable_classic(full_width = F, html_font = "Cambria")

# Saur-Dpug

Saur.Dpug.no.Avoid <- c(
  round(length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="CM" & Saur.Dpug.Avoid$Avoid==0])/length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="CM"])*100, 0),
  round(length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="pH_drop" & Saur.Dpug.Avoid$Avoid==0])/length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="pH_drop"])*100, 0),
```
```

```
  round(length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="SM" &
Saur.Dpug.Avoid$Avoid==0])/length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="SM"])*100, 0),

  round(length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="pH_drop+SM" &
Saur.Dpug.Avoid$Avoid==0])/length(Saur.Dpug.Avoid$Avoid[Saur.Dpug.Avoid$Treatment=="pH_drop+SM"])*100, 0))


Saur.Dpug.yes.Avoid <- 100-Saur.Dpug.no.Avoid


Saur.Dpug.Avoid.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Avoidance"= c(Saur.Dpug.yes.Avoid),

          "No Avoidance"=c(Saur.Dpug.no.Avoid))


Saur.Dpug.Avoid.percent %>%

  kbl(digits = 4,caption = "Effect of treatment on percentage values of Avoidance/No avoidance for Saur/Dpug"  ) %>%

  kable_classic(full_width = F, html_font = "Cambria")



```

The avoidance response of Dpug is plotted in function of predictors.

```{r Dpug/Dpug avoidance plot, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)
# Dpug - avoid per treatment across all donors
p_glz.Dpug.avoid.Treat <- ggplot(data = Dpug.Avoid, aes(x=Sort_treatment, fill=factor(avoid_treat)))+

  geom_bar(position="fill", colour = "black") +

  scale_fill_manual("Avoidance", values = alpha(c("#1F968BFF",  "#1F968BFF", "#73D055FF",    "#73D055FF", "#FDE725FF",  "#FDE725FF",
"#D8DE29FF",  "#D8DE29FF"), c( 0.6,1,0.6,1,0.6,1,0.6, 1)),labels = c("No", "Yes", "No", "Yes", "No", "Yes", "No", "Yes")) +

  labs(x=NULL, y = "Avoidance %")+

  scale_x_discrete(labels=c("CM", "pH drop", "SM", "pH drop\n+SM")) +

  theme_classic()+

  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+

  own_theme()


# Facet by Donor
p_glz.Dpug.avoid.Treat.facet <- p_glz.Dpug.avoid.Treat + theme_bw() + theme(legend.position="top",

                                  panel.border = element_rect(colour = "black", fill=NA, size=0.5),

                                  panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor) + own_theme() +
theme(legend.position="none")


```


Freezing

The binomial generalised linear model is used to model the effects of predictors on the freezing response of Dpug.

```{r Dpug freezing glm, echo=TRUE, warning=FALSE}

glz.Dpug.freeze.null <- glm(Freeze ~ 1, data=Dpug.Avoid,family=binomial(link = "logit"))

glz.Dpug.freeze <- glm(Freeze ~ pH_drop*SM*Donor, data=Dpug.Avoid,family=binomial(link = "logit"))

glz.Dpug.freeze.mod2 <- glm(Freeze ~ pH_drop*SM*Donor+Water_use, data=Dpug.Avoid,family=binomial(link = "logit"))

glz.Dpug.freeze.mod3 <- glm(Freeze ~ pH_drop*SM*Donor+Water_use+Crab_size, data=Dpug.Avoid,family=binomial(link = "logit"))

anova(glz.Dpug.freeze.null, glz.Dpug.freeze, glz.Dpug.freeze.mod2,glz.Dpug.freeze.mod3, test="Chisq")

glz.Dpug.freeze.anova <- anova(glz.Dpug.freeze.null, glz.Dpug.freeze, glz.Dpug.freeze.mod2,glz.Dpug.freeze.mod3, test="Chisq") # The water use and the crab size can be disregarded. Year is not included since the freezing behaviour was only recorded in Autumn 2019 (not in Autumn 2018)


glz.Dpug.freeze.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of freezing in Dpug" ) %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of glz.Dpug.freeze model = ", AIC(glz.Dpug.freeze)) # AIC

as.data.frame(summary(glz.Dpug.freeze)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of binomial generalised linear model for the freezing in Dpug" ) %>%

  kable_classic(full_width = F, html_font = "Cambria")



# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

# pH*SM*Donor term - posthoc tests - Run the full posthoc matrix regardless of whether the interaction term is significant in the model


as.data.frame(emmeans( glz.Dpug.freeze, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the freezing in Dpug - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Dpug.freeze, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the freezing in Dpug - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(summary(emmeans( glz.Dpug.freeze, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the freezing in Dpug - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")
```

```
as.data.frame(emmeans( glz.Dpug.freeze, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%
  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the freezing in Dpug - model term donor|(pH drop:Stress metabolites)") %>%
  kable_classic(full_width = F, html_font = "Cambria")

plot_model(glz.Dpug.freeze, type = "pred", terms = c("pH_drop", "SM", "Donor"))

glz.Dpug.freeze_model_data <- as.data.frame(get_model_data(glz.Dpug.freeze, type = "pred", terms = c("pH_drop", "SM", "Donor")))
colnames(glz.Dpug.freeze_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')
glz.Dpug.freeze_model_data %>%
  kbl(digits = 4,caption = "Predicted model values for the freezing in Dpug according to the generalised linear model") %>%
  kable_classic(full_width = F, html_font = "Cambria")

```

The percentages of freezing/no freezing are retrieved for each group in Dpug.

```{r Dpug freezing percent, echo=TRUE, warning=FALSE}

# Dpug/Dpug

Dpug.Dpug.no.Freeze <- c(
  round(length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="CM" & Dpug.Dpug.Avoid$Freeze==0])/length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="CM"])*100, 0),
  round(length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="pH_drop" & Dpug.Dpug.Avoid$Freeze==0])/length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="SM" & Dpug.Dpug.Avoid$Freeze==0])/length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="SM"])*100, 0),
  round(length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="pH_drop+SM" & Dpug.Dpug.Avoid$Freeze==0])/length(Dpug.Dpug.Avoid$Freeze[Dpug.Dpug.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Dpug.Dpug.yes.Freeze <- 100-Dpug.Dpug.no.Freeze

Dpug.Dpug.Freeze.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Freeze"= c(Dpug.Dpug.yes.Freeze),
          "No Freeze"=c(Dpug.Dpug.no.Freeze))


Dpug.Dpug.Freeze.percent %>%
  kbl(digits = 4,caption = "Effect of treatment on percentage values of freezing/No freezing for Dpug/Dpug") %>%
  kable_classic(full_width = F, html_font = "Cambria")

# Saur-Dpug
```

```
Saur.Dpug.no.Freeze <- c(
  round(length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="CM" &
Saur.Dpug.Avoid$Freeze==0])/length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="CM"])*100, 0),
  round(length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="pH_drop" &
Saur.Dpug.Avoid$Freeze==0])/length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="SM" &
Saur.Dpug.Avoid$Freeze==0])/length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="SM"])*100, 0),
  round(length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="pH_drop+SM" &
Saur.Dpug.Avoid$Freeze==0])/length(Saur.Dpug.Avoid$Freeze[Saur.Dpug.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Saur.Dpug.yes.Freeze <- 100-Saur.Dpug.no.Freeze

Saur.Dpug.Freeze.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Freeze %"= c(Saur.Dpug.yes.Freeze),
          "No Freeze %"=c(Saur.Dpug.no.Freeze))

Saur.Dpug.Freeze.percent %>%
  kbl(digits = 4,caption = "Effect of treatment on percentage values of freezing/No freezing for Saur/Dpug") %>%
  kable_classic(full_width = F, html_font = "Cambria")
```

```

The freezing responses of Dpug are plotted in function of predictors.

```{r Dpug freezing plot, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)

# Dpug - freezing per treatment across all donors
p_glz.Dpug.freezing.Treat <- ggplot(data = Dpug.Avoid, aes(x=Sort_treatment, fill=factor(freeze_treat)))+
  geom_bar(position="fill", colour = "black") +
  scale_fill_manual("Freezing", values = alpha(c("#1F968BFF",  "#1F968BFF", "#73D055FF",    "#73D055FF", "#FDE725FF",  "#FDE725FF",
"#D8DE29FF",  "#D8DE29FF"), c( 0.6,1,0.6,1,0.6,1,0.6, 1)),labels = c("No", "Yes", "No", "Yes", "No", "Yes", "No", "Yes")) +
  labs(x=NULL, y = "Freezing %")+
  scale_x_discrete(labels=c("CM", "pH drop", "SM", "pH drop\n+SM")) +
  theme_classic()+
  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
  own_theme()

# Facet by Donor
p_glz.Saur.Dpug.freezing.facet <- p_glz.Dpug.freezing.Treat + theme_bw() + theme(legend.position="top",
```
```

panel.border = element_rect(colour = "black", fill=NA, size=0.5),

panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor) + own_theme() + theme(legend.position="none")

p_glz.Saur.Dpug.freezing.facet

```

### Escaping

The binomial generalised linear model is used to model the effects of predictors on the escaping response of Dpug.

```{r Dpug escaping glm, echo=TRUE, warning=FALSE}

glz.Dpug.escape.null <- glm(Escape ~ 1, data=Dpug.Avoid,family=binomial(link = "logit"))

glz.Dpug.escape <- glm(Escape ~ pH_drop*SM*Donor, data=Dpug.Avoid,family=binomial(link = "logit"))

glz.Dpug.escape.mod2 <- glm(Escape ~ pH_drop*SM*Donor+Water_use, data=Dpug.Avoid,family=binomial(link = "logit"))

glz.Dpug.escape.mod3 <- glm(Escape ~ pH_drop*SM*Donor+Water_use+Crab_size, data=Dpug.Avoid,family=binomial(link = "logit"))

anova(glz.Dpug.escape.null, glz.Dpug.escape, glz.Dpug.escape.mod2, glz.Dpug.escape.mod3,test="Chisq")

glz.Dpug.escape.anova <- anova(glz.Dpug.escape.null, glz.Dpug.escape, glz.Dpug.escape.mod2, glz.Dpug.escape.mod3,test="Chisq") # The water use can be disregarded. The crab size is kept in the model. Year is not included since the escaping behaviour was recorded only in Autumn 2019 (not in Autumn 2018).

glz.Dpug.escape.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of escape in Dpug") %>%

  kable_classic(full_width = F, html_font = "Cambria")

# The model with 'crab size' as a covariate is constructed.

glz.Dpug.escape.mod4 <- glm(Escape ~ pH_drop*SM*Donor+Crab_size, data=Dpug.Avoid,family=binomial(link = "logit"))

glz.Dpug.escape.anova2 <- anova(glz.Dpug.escape, glz.Dpug.escape.mod4,test="Chisq") # Verify that the model has a better fit with the crab size

glz.Dpug.escape.anova2 %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of escape in Dpug") %>%

  kable_classic(full_width = F, html_font = "Cambria")

paste("AIC of glz.Dpug.escape.mod4 model = ", AIC(glz.Dpug.escape.mod4)) # AIC

as.data.frame(summary(glz.Dpug.escape.mod4)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of binomial generalised linear model for the escape in Dpug") %>%

  kable_classic(full_width = F, html_font = "Cambria")

# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

# pH*SM*Donor term - posthoc tests - Run the full posthoc matrix regardless of whether the interaction term is significant in the model

```
as.data.frame(emmeans(glz.Dpug.escape.mod4, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the escaping in Dpug - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(emmeans(glz.Dpug.escape.mod4, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the escaping in Dpug - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(summary(emmeans(glz.Dpug.escape.mod4, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the escaping in Dpug - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(emmeans(glz.Dpug.escape.mod4, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the escaping in Dpug - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")

plot_model(glz.Dpug.escape.mod4, type = "pred", terms = c("pH_drop", "SM", "Donor"))

glz.Dpug.escape_model_data <- as.data.frame(get_model_data(glz.Dpug.escape.mod4, type = "pred", terms = c("pH_drop", "SM", "Donor")))

colnames(glz.Dpug.escape_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')

glz.Dpug.escape_model_data %>%

  kbl(digits = 4,caption = "Predicted model values for the escape in Dpug according to the generalised linear model") %>%

  kable_classic(full_width = F, html_font = "Cambria")
```

The percentages of escaping/no escaping are retrieved for each group in Dpug.

```{r Dpug escaping percent, echo=TRUE, warning=FALSE}

# Dpug/Dpug

Dpug.Dpug.no.Escape <- c(

  round(length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="CM" & Dpug.Dpug.Avoid$Escape==0])/length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="CM"])*100, 0),

  round(length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="pH_drop" & Dpug.Dpug.Avoid$Escape==0])/length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="pH_drop"])*100, 0),

  round(length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="SM" & Dpug.Dpug.Avoid$Escape==0])/length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="SM"])*100, 0),
```

```
  round(length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="pH_drop+SM" &
Dpug.Dpug.Avoid$Escape==0])/length(Dpug.Dpug.Avoid$Escape[Dpug.Dpug.Avoid$Treatment=="pH_drop+SM"])*100, 0))


Dpug.Dpug.yes.Escape <- 100-Dpug.Dpug.no.Escape


Dpug.Dpug.Escape.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Escape"= c(Dpug.Dpug.yes.Escape),
          "No Escape"=c(Dpug.Dpug.no.Escape))


Dpug.Dpug.Escape.percent %>%
  kbl(digits = 4,caption = "Percentage of escape/no escape for Dpug/Dpug") %>%
  kable_classic(full_width = F, html_font = "Cambria")


# Saur-Dpug
Saur.Dpug.no.Escape <- c(
  round(length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="CM" &
Saur.Dpug.Avoid$Escape==0])/length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="CM"])*100, 0),
  round(length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="pH_drop" &
Saur.Dpug.Avoid$Escape==0])/length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="SM" &
Saur.Dpug.Avoid$Escape==0])/length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="SM"])*100, 0),
  round(length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="pH_drop+SM" &
Saur.Dpug.Avoid$Escape==0])/length(Saur.Dpug.Avoid$Escape[Saur.Dpug.Avoid$Treatment=="pH_drop+SM"])*100, 0))


Saur.Dpug.yes.Escape <- 100-Saur.Dpug.no.Escape


Saur.Dpug.Escape.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Escape"= c(Saur.Dpug.yes.Escape),
          "No Escape"=c(Saur.Dpug.no.Escape))



Saur.Dpug.Escape.percent %>%
  kbl(digits = 4,caption = "Percentage of escape/no escape for Saur/Dpug") %>%
  kable_classic(full_width = F, html_font = "Cambria")


```

The escaping response of Dpug is plotted in function of predictors.

```{r Dpug escaping plot, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)
# Dpug - escaping per treatment across all donors
```

```
p_glz.Dpug.escaping.Treat <- ggplot(data = Dpug.Avoid, aes(x=Sort_treatment, fill=factor(escape_treat)))+
 geom_bar(position="fill", colour = "black") +
 scale_fill_manual("Escaping", values = alpha(c("#1F968BFF",  "#1F968BFF", "#73D055FF",    "#73D055FF", "#FDE725FF",  "#FDE725FF", "#D8DE29FF",  "#D8DE29FF"), c( 0.6,1,0.6,1,0.6,1,0.6, 1)),labels = c("No", "Yes", "No", "Yes", "No", "Yes", "No", "Yes")) +
 labs(x=NULL, y = "Escaping %")+
 scale_x_discrete(labels=c("CM", "pH drop", "SM", "pH drop\n+SM")) +
 theme_classic()+
 scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
 own_theme()

# Facet by Donor
p_glz.Saur.Dpug.escaping.facet <- p_glz.Dpug.escaping.Treat + theme_bw() + theme(legend.position="top",
                                  panel.border = element_rect(colour = "black", fill=NA, size=0.5),
                                  panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor) + own_theme() + theme(legend.position="none")
p_glz.Saur.Dpug.escaping.facet

# Covariate - crab size
p_glz.Dpug.escaping.size1 <- ggplot(data = Dpug.Avoid, aes(x=Crab_size, fill=factor(Escape)))+
 geom_bar(position="fill", colour = "black") +
  scale_fill_grey(start=0.85, end=0.7) +
 labs(x="crab size (cm)", y = "Escaping %")+
 theme_classic()+
 scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
 own_theme()
p_glz.Dpug.escaping.size1

# Size as integer
glz.Dpug.escape.mod5 <- glm(Escape ~ pH_drop*SM*Donor+as.integer(Crab_size), data=Dpug.Avoid,family=binomial(link = "logit")) # Numbers as integer to better see the difference
as.data.frame(summary(glz.Dpug.escape.mod5)$coefficients) %>%
 kbl(digits = 4,caption = "Summary of binomial generalised linear model for the escaping in Dpug in function of size as integer") %>%
 kable_classic(full_width = F, html_font = "Cambria")

p_glz.Dpug.escaping.size2 <- ggplot(data = Dpug.Avoid, aes(x=as.integer(Crab_size), fill=factor(Escape)))+
 geom_bar(position="fill", colour = "black") +
   scale_fill_grey(start=0.85, end=0.7) +
 labs(x="crab size (cm)", y = "Escaping %")+
 theme_classic()+
 scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
 own_theme()
```

See the probabilities

ggplot(Dpug.Avoid, aes(x=Crab_size, y=as.numeric(as.character(Escape)))) + geom_point(position = position_jitter(width = 0.01, height = 0.01)) + stat_smooth(method="glm", method.args=list(family="binomial"), se=FALSE)

p_glz.Dpug.escaping.size2

```

### Time-to-success

The Survival analysis (aka time-to-event) is used to model the effects of predictors on the time response.

```{r interpret the censoring 1, echo=TRUE, warning=FALSE}

# Imagine that reaching the cue means the death of the animal:

# Event == finding the cue ('death') coded as 1

# No event == Not finding the cue ('survival') coded as 0

survival_coding <- data.frame("Event in survival study"=c("Dead", "Survived"), "Event in our study"=c("Found the cue",

   "Did not find the cue"), "Event occurence"=c("Yes", "No"), "Success"=c("TRUE", "FALSE"), "Success2"=c("1", "0"), "Censoring status"=c("2", "1"),

  "Probability in survival study"=c("Death probability", "Survival probability"), "Probability in our study"=c("Success Probability", "Failure probability"))

 survival_coding %>%

  kbl(digits = 4,caption = "Summary of binomial generalised linear model for the escaping in Dpug in function of size as integer") %>%

  kable_classic(full_width = F, html_font = "Cambria")

```

The Cox proportional hazard model is used to model the effects of predictors on the time response of Dpug.

```{r Dpug Cox time, echo=TRUE, warning=FALSE}

# Dpug across all donors

Dpug.Time.Coxph.null <- coxph(Surv(time_s, Censoring_status) ~ 1, data = Dpug.Time)

Dpug.Time.Coxph <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor, data = Dpug.Time)

Dpug.Time.Coxph.mod2 <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor+Water_use, data = Dpug.Time)

Dpug.Time.Coxph.mod3 <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor+Water_use+Crab_size, data = Dpug.Time)

Dpug.Time.anova <- anova(Dpug.Time.Coxph.null, Dpug.Time.Coxph, Dpug.Time.Coxph.mod2, Dpug.Time.Coxph.mod3, test="chisq") # Water use and crab size can be disregarded

```
anova(Dpug.Time.Coxph.null, Dpug.Time.Coxph, Dpug.Time.Coxph.mod2, Dpug.Time.Coxph.mod3, test="chisq") # Water use and crab size can be disregarded

round(Dpug.Time.anova, 4) %>%
  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of time in Dpug") %>%
  kable_classic(full_width = F, html_font = "Cambria")

paste("AIC of Dpug.Time.Coxph model = ", AIC(Dpug.Time.Coxph)) # AIC
as.data.frame(summary(Dpug.Time.Coxph)$coefficients) %>%
  kbl(digits = 4,caption = "Summary of Cox proportional model for the time in Dpug") %>%
  kable_classic(full_width = F, html_font = "Cambria")

To have a hazard ratio table
coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor, data = Dpug.Time) %>%
  gtsummary::tbl_regression(exp = TRUE)

# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans
plot_model(Dpug.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor"))

pH*SM*Donor term - posthoc tests - Run the full posthoc matrix regardless of whether the interaction term is significant in the model
as.data.frame(emmeans(Dpug.Time.Coxph, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%
  kbl(digits = 4,caption = "Summary of pairwise tests for the Cox proportional model for the time in Dpug - model term (pH drop:Stress metabolites)|donor") %>%
  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(emmeans(Dpug.Time.Coxph, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%
  kbl(digits = 4,caption = "Statistic values for the Cox proportional model for the time in Dpug - model term (pH drop:Stress metabolites)|donor") %>%
  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(summary(emmeans(Dpug.Time.Coxph, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%
  kbl(digits = 4,caption = "Summary of pairwise tests for the Cox proportional model for the time in Dpug - model term donor|(pH drop:Stress metabolites)") %>%
  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(emmeans(Dpug.Time.Coxph, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%
  kbl(digits = 4,caption = "Statistic values for the Cox proportional for the time in Dpug - model term donor|(pH drop:Stress metabolites)") %>%
  kable_classic(full_width = F, html_font = "Cambria")

glz.Dpug.time_model_data <- as.data.frame(get_model_data(Dpug.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor")))
colnames(glz.Dpug.time_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')
```

```
glz.Dpug.time_model_data %>%
  kbl(digits = 4,caption = "Predicted model values for the time in Dpug according to the Cox proportional model") %>%
  kable_classic(full_width = F, html_font = "Cambria")

```

The time response of Dpug is plotted using hazard ratio plots and Kaplan-Meier survival curves.

```{r Dpug time plot, echo=TRUE, warning=FALSE}

# Forest plot
ggforest(Dpug.Time.Coxph, data=Dpug.Time)

# Raw Kaplan-Meier curve of all the subpopulations (facetting by Donor type)
Dpug.Time.Survfit.Treat <- survfit( Surv(time_s, Censoring_status) ~ pH_drop+SM+Donor, data = Dpug.Time )
Dpug.Time.Survfit.Treat.plot <- ggsurvplot(Dpug.Time.Survfit.Treat, conf.int = T, fun = "event",
                                           ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("npg"),
 legend = "bottom",
 legend.title = "Treatment",
xlim = c(0, 300),
             break.x.by = 60,
              risk.table.height=.3,
 tables.theme = theme_cleantable())

Dpug.Time.Survfit.Treat.plot2 <- Dpug.Time.Survfit.Treat.plot$plot + theme_bw() + theme(legend.position="top",
                                  panel.border = element_rect(colour = "black", fill=NA, size=0.5),
                                  panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor)+
theme(plot.background = element_rect(color = "black"))
Dpug.Time.Survfit.Treat.plot2

```

Cmae

Avoidance

The binomial generalised linear model is used to model the effects of predictors on the avoidance response of Cmae.

```{r Cmae avoidance glm, echo=TRUE, warning=FALSE}

glz.Cmae.avoid.null <- glm(Avoidance ~ 1, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.avoid <- glm(Avoidance ~ pH_drop*SM*Donor, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.avoid.mod2 <- glm(Avoidance ~ pH_drop*SM*Donor+Water_use, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.avoid.mod3 <- glm(Avoidance ~ pH_drop*SM*Donor+Water_use+Crab_size, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.avoid.anova <- anova(glz.Cmae.avoid.null, glz.Cmae.avoid, glz.Cmae.avoid.mod2, glz.Cmae.avoid.mod3, test="Chisq") # The water use is kept in the model but the crab size is disregarded. Year is not included in the model since the avoidance behaviour was recorded only in Autumn 2019 (not in Autumn 2018)

anova(glz.Cmae.avoid.null, glz.Cmae.avoid, glz.Cmae.avoid.mod2, glz.Cmae.avoid.mod3, test="Chisq")


glz.Cmae.avoid.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of avoidance in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


glz.Cmae.avoid.anova2 <- anova(glz.Cmae.avoid.null, glz.Cmae.avoid.mod2, test="Chisq")

glz.Cmae.avoid.anova2 %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of avoidance in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of glz.Cmae.avoid.mod2 model = ", AIC(glz.Cmae.avoid.mod2)) # AIC

as.data.frame(summary(glz.Cmae.avoid.mod2)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of generalised linear model for the avoidance in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

# pH*SM*Donor term - posthoc tests - run because the term in the model is significant

as.data.frame(emmeans( glz.Cmae.avoid.mod2, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the avoidance in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Cmae.avoid.mod2, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the generalised linear model for the avoidance in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(summary(emmeans( glz.Cmae.avoid.mod2, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the avoidance in Cmae - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")
```

```
as.data.frame(emmeans( glz.Cmae.avoid.mod2, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%
  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the avoidance in Cmae - model term donor|(pH drop:Stress metabolites)") %>%
  kable_classic(full_width = F, html_font = "Cambria")

plot_model(glz.Cmae.avoid.mod2, type = "pred", terms = c("pH_drop", "SM", "Donor"))

glz.Cmae.avoid_model_data <- as.data.frame(get_model_data(glz.Cmae.avoid.mod2, type = "pred", terms = c("pH_drop", "SM", "Donor")))
colnames(glz.Cmae.avoid_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')
glz.Cmae.avoid_model_data %>%
  kbl(digits = 4,caption = "Predicted model values for the avoidance in Cmae according to the generalised linear model") %>%
  kable_classic(full_width = F, html_font = "Cambria")

```

THe percentages of avoidance/no avoidance is retrieved for each group of Cmae.

```{r Cmae/Cmae avoidance percent, echo=TRUE, warning=FALSE}
# Cmae/Cmae

Cmae.Cmae.no.Avoid <- c(
  round(length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="CM" & Cmae.Cmae.Avoid$Avoid==0])/length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="CM"])*100, 0),
  round(length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="pH_drop" & Cmae.Cmae.Avoid$Avoid==0])/length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="SM" & Cmae.Cmae.Avoid$Avoid==0])/length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="SM"])*100, 0),
  round(length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="pH_drop+SM" & Cmae.Cmae.Avoid$Avoid==0])/length(Cmae.Cmae.Avoid$Avoid[Cmae.Cmae.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Cmae.Cmae.yes.Avoid <- 100-Cmae.Cmae.no.Avoid

Cmae.Cmae.Avoid.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Avoidance"= c(Cmae.Cmae.yes.Avoid),
          "No Avoidance"=c(Cmae.Cmae.no.Avoid))

Cmae.Cmae.Avoid.percent %>%
  kbl(digits = 4,caption = "Percentages of avoidance/no avoidance in Cmae/Cmae") %>%
  kable_classic(full_width = F, html_font = "Cambria")


# Saur-Cmae
Saur.Cmae.no.Avoid <- c(
```

```
  round(length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="CM" &
Saur.Cmae.Avoid$Avoid==0])/length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="CM"])*100, 0),

  round(length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="pH_drop" &
Saur.Cmae.Avoid$Avoid==0])/length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="pH_drop"])*100, 0),

  round(length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="SM" &
Saur.Cmae.Avoid$Avoid==0])/length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="SM"])*100, 0),

  round(length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="pH_drop+SM" &
Saur.Cmae.Avoid$Avoid==0])/length(Saur.Cmae.Avoid$Avoid[Saur.Cmae.Avoid$Treatment=="pH_drop+SM"])*100, 0))


Saur.Cmae.yes.Avoid <- 100-Saur.Cmae.no.Avoid



Saur.Cmae.Avoid.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Avoidance"= c(Saur.Cmae.yes.Avoid),

          "No Avoidance"=c(Saur.Cmae.no.Avoid))


Saur.Cmae.Avoid.percent %>%

  kbl(digits = 4,caption = "Percentages of avoidance/no avoidance in Saur/Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


```


The avoidance response of Cmae is plotted in function of predictors.


```{r Cmae/Cmae avoidance plot, echo=TRUE, warning=FALSE}

detach("package:cowplot", unload=TRUE)

library(cowplot)

# Cmae - avoid per treatment across all donors

p_glz.Cmae.avoid.Treat <- ggplot(data = Cmae.Avoid, aes(x=Sort_treatment, fill=factor(avoid_treat)))+

  geom_bar(position="fill", colour = "black") +

  scale_fill_manual("Avoidance", values = alpha(c("#1F968BFF",  "#1F968BFF", "#73D055FF",    "#73D055FF", "#FDE725FF",  "#FDE725FF",
"#D8DE29FF",  "#D8DE29FF"), c( 0.6,1,0.6,1,0.6,1,0.6, 1)),labels = c("No", "Yes", "No", "Yes", "No", "Yes", "No", "Yes")) +

  labs(x=NULL, y = "Avoidance %")+

  scale_x_discrete(labels=c("CM", "pH drop", "SM", "pH drop\n+SM")) +

  theme_classic()+

  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+

  own_theme()


# Facet by Donor

p_glz.Saur.Cmae.avoid.facet <- p_glz.Cmae.avoid.Treat + theme_bw() + theme(legend.position="top",

                                panel.border = element_rect(colour = "black", fill=NA, size=0.5),

                                panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor) + own_theme() +
theme(legend.position="none")

p_glz.Saur.Cmae.avoid.facet
```

```
# Covariate - water use

glz.Cmae.avoid.wateruse <- glm(Avoidance ~ Water_use, data=Cmae.Avoid,family=binomial(link = "logit"))

as.data.frame(summary(glz.Cmae.avoid.wateruse)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of generalised linear model for the avoidance in Cmae - in function of water use") %>%

  kable_classic(full_width = F, html_font = "Cambria")


p_glz.Cmae.avoid.wateruse <- ggplot(data = Cmae.Avoid, aes(x=Water_use, fill=factor(Avoidance)))+

  geom_bar(position="fill", colour = "black") +

     scale_fill_grey(start=0.85, end=0.7) +

  labs(x="Water uses", y = "Avoidance %")+

  theme_classic()+

  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+

  own_theme()

# See the probabilities

ggplot(Cmae.Avoid, aes(x=Water_use, y=as.numeric(as.character(Avoidance)))) + geom_point(position = position_jitter(width = 0.01, height = 0.01)) +  stat_smooth(method="glm", method.args=list(family="binomial"), se=FALSE)

p_glz.Cmae.avoid.wateruse

```


Freezing

The binomial generalised linear model is used to model the effects of predictors on the freezing response of Cmae.


```{r Cmae freezing glm, echo=TRUE, warning=FALSE}

glz.Cmae.freeze.null <- glm(Freeze ~ 1, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.freeze <- glm(Freeze ~ pH_drop*SM*Donor, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.freeze.mod2 <- glm(Freeze ~ pH_drop*SM*Donor+Water_use, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.freeze.mod3 <- glm(Freeze ~ pH_drop*SM*Donor+Water_use+Crab_size, data=Cmae.Avoid,family=binomial(link = "logit"))

glz.Cmae.freeze.anova <- anova(glz.Cmae.freeze.null, glz.Cmae.freeze, glz.Cmae.freeze.mod2, glz.Cmae.freeze.mod3, test="Chisq") # The water use and crab size can be disregarded.Year is not included in the model since the freezing behaviour was observed only in Autumn 2019 (not in Autumn 2018).

anova(glz.Cmae.freeze.null, glz.Cmae.freeze, glz.Cmae.freeze.mod2, glz.Cmae.freeze.mod3, test="Chisq")


glz.Cmae.freeze.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of freezing in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of glz.Cmae.freeze model = ", AIC(glz.Cmae.freeze)) # AIC

as.data.frame(summary(glz.Cmae.freeze)$coefficients) %>%
```

```
  kbl(digits = 4,caption = "Summary of generalised linear model for the freezing in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

# pH*SM*Donor term - posthoc tests - run because the term in the model is  significant


as.data.frame(emmeans( glz.Cmae.freeze, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the freezing in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Cmae.freeze, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the generalised linear model for the freezing in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(summary(emmeans( glz.Cmae.freeze, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the freezing in Cmae - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Cmae.freeze, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the freezing in Cmae - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


plot_model(glz.Cmae.freeze, type = "pred", terms = c("pH_drop", "SM", "Donor"))


glz.Cmae.freeze_model_data <- as.data.frame(get_model_data(glz.Cmae.freeze, type = "pred", terms = c("pH_drop", "SM", "Donor")))

colnames(glz.Cmae.freeze_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')

glz.Cmae.freeze_model_data %>%

  kbl(digits = 4,caption = "Predicted model values for the freezing in Cmae according to the generalised linear model") %>%

  kable_classic(full_width = F, html_font = "Cambria")


```
```

The percentages of freezing/no freezing are retrieved for Cmae.

```{r Cmae freezing percent, echo=TRUE, warning=FALSE}

# Cmae/Cmae
```
```

```
Cmae.Cmae.no.Freeze <- c(
  round(length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="CM" &
Cmae.Cmae.Avoid$Freeze==0])/length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="CM"])*100, 0),
  round(length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="pH_drop" &
Cmae.Cmae.Avoid$Freeze==0])/length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="SM" &
Cmae.Cmae.Avoid$Freeze==0])/length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="SM"])*100, 0),
  round(length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="pH_drop+SM" &
Cmae.Cmae.Avoid$Freeze==0])/length(Cmae.Cmae.Avoid$Freeze[Cmae.Cmae.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Cmae.Cmae.yes.Freeze <- 100-Cmae.Cmae.no.Freeze

Cmae.Cmae.Freeze.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Freeze"= c(Cmae.Cmae.yes.Freeze),
          "No Freeze"=c(Cmae.Cmae.no.Freeze))

Cmae.Cmae.Freeze.percent %>%
  kbl(digits = 4,caption = "Percentages of freezing/no freezing in Cmae/Cmae") %>%
  kable_classic(full_width = F, html_font = "Cambria")

# Saur-Cmae
Saur.Cmae.no.Freeze <- c(
  round(length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="CM" &
Saur.Cmae.Avoid$Freeze==0])/length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="CM"])*100, 0),
  round(length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="pH_drop" &
Saur.Cmae.Avoid$Freeze==0])/length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="SM" &
Saur.Cmae.Avoid$Freeze==0])/length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="SM"])*100, 0),
  round(length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="pH_drop+SM" &
Saur.Cmae.Avoid$Freeze==0])/length(Saur.Cmae.Avoid$Freeze[Saur.Cmae.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Saur.Cmae.yes.Freeze <- 100-Saur.Cmae.no.Freeze

Saur.Cmae.Freeze.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Freeze"= c(Saur.Cmae.yes.Freeze),
          "No Freeze"=c(Saur.Cmae.no.Freeze))

Saur.Cmae.Freeze.percent %>%
  kbl(digits = 4,caption = "Percentages of freezing/no freezing in Saur/Cmae") %>%
  kable_classic(full_width = F, html_font = "Cambria")

```
```
```

The freezing response of Cmae is plotted in function of predictors.

```{r Cmae freezing plot, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)
# Cmae - freezing per treatment across all donors
p_glz.Cmae.freezing.Treat <- ggplot(data = Cmae.Avoid, aes(x=Sort_treatment, fill=factor(freeze_treat)))+
 geom_bar(position="fill", colour = "black") +
 scale_fill_manual("Freezing", values = alpha(c("#1F968BFF",  "#1F968BFF", "#73D055FF",    "#73D055FF", "#FDE725FF",  "#FDE725FF", "#D8DE29FF",  "#D8DE29FF"), c( 0.6,1,0.6,1,0.6,1,0.6, 1)),labels = c("No", "Yes", "No", "Yes", "No", "Yes", "No", "Yes")) +
 labs(x=NULL, y = "Freezing %")+
 scale_x_discrete(labels=c("CM", "pH drop", "SM", "pH drop\n+SM")) +
 theme_classic()+
 scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
 own_theme()

# Facet by Donor
p_glz.Saur.Cmae.freezing.facet <- p_glz.Cmae.freezing.Treat + theme_bw() + theme(legend.position="top",
                                panel.border = element_rect(colour = "black", fill=NA, size=0.5),
                                panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor) + own_theme() + theme(legend.position="none")
p_glz.Saur.Cmae.freezing.facet

```

Escaping

The binomial generalised linear model is used to model the effects of predictors on the escaping response of Cmae.

```{r Cmae escaping glm, echo=TRUE, warning=FALSE}
glz.Cmae.escape.null <- glm(Escape ~ 1, data=Cmae.Avoid,family=binomial(link = "logit"))
glz.Cmae.escape <- glm(Escape ~ pH_drop*SM*Donor, data=Cmae.Avoid,family=binomial(link = "logit"))
glz.Cmae.escape.mod2 <- glm(Escape ~ pH_drop*SM*Donor+Water_use, data=Cmae.Avoid,family=binomial(link = "logit"))
glz.Cmae.escape.mod3 <- glm(Escape ~ pH_drop*SM*Donor+Water_use+Crab_size, data=Cmae.Avoid,family=binomial(link = "logit"))
glz.Cmae.escape.anova <- anova(glz.Cmae.escape.null, glz.Cmae.escape, glz.Cmae.escape.mod2, glz.Cmae.escape.mod3, test="Chisq") # The water use and crab size can be disregarded. Year is not included in the model since the escaping behaviour was recorded only in Autumn 2019 (not in Autumn 2018).
anova(glz.Cmae.escape.null, glz.Cmae.escape, glz.Cmae.escape.mod2, glz.Cmae.escape.mod3, test="Chisq")

glz.Cmae.escape.anova %>%
 kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of escaping in Cmae") %>%
```

```
  kable_classic(full_width = F, html_font = "Cambria")



paste("AIC of glz.Cmae.escape model = ", AIC(glz.Cmae.escape)) # AIC

as.data.frame(summary(glz.Cmae.escape)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of generalised linear model for the escaping in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

# pH*SM*Donor term - posthoc tests - Run the full posthoc matrix regardless of whether the interaction term is significant in the model


as.data.frame(emmeans( glz.Cmae.escape, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the escaping in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Cmae.escape, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the generalised linear model for the escaping in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(summary(emmeans( glz.Cmae.escape, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the escaping in Cmae - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( glz.Cmae.escape, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the escaping in Cmae - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


plot_model(glz.Cmae.escape, type = "pred", terms = c("pH_drop", "SM", "Donor"))


glz.Cmae.escape_model_data <- as.data.frame(get_model_data(glz.Cmae.escape, type = "pred", terms = c("pH_drop", "SM", "Donor")))

colnames(glz.Cmae.escape_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')

glz.Cmae.escape_model_data %>%

  kbl(digits = 4,caption = "Predicted model values for the escaping in Cmae according to the generalised linear model") %>%

  kable_classic(full_width = F, html_font = "Cambria")


```
```
```

THe percentages of escaping/no escaping are retrieved for Cmae.

```{r Cmae escaping percent, echo=TRUE, warning=FALSE}

# Cmae/Cmae

Cmae.Cmae.no.Escape <- c(

  round(length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="CM" &
Cmae.Cmae.Avoid$Escape==0])/length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="CM"])*100, 0),

  round(length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="pH_drop" &
Cmae.Cmae.Avoid$Escape==0])/length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="pH_drop"])*100, 0),

  round(length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="SM" &
Cmae.Cmae.Avoid$Escape==0])/length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="SM"])*100, 0),

  round(length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="pH_drop+SM" &
Cmae.Cmae.Avoid$Escape==0])/length(Cmae.Cmae.Avoid$Escape[Cmae.Cmae.Avoid$Treatment=="pH_drop+SM"])*100, 0))


Cmae.Cmae.yes.Escape <- 100-Cmae.Cmae.no.Escape


Cmae.Cmae.Escape.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Escape"= c(Cmae.Cmae.yes.Escape),

          "No Escape"=c(Cmae.Cmae.no.Escape))

Cmae.Cmae.Escape.percent %>%

  kbl(digits = 4,caption = "Percentage of escape/no escape in Cmae/Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")



# Saur-Cmae

Saur.Cmae.no.Escape <- c(

  round(length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="CM" &
Saur.Cmae.Avoid$Escape==0])/length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="CM"])*100, 0),

  round(length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="pH_drop" &
Saur.Cmae.Avoid$Escape==0])/length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="pH_drop"])*100, 0),

  round(length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="SM" &
Saur.Cmae.Avoid$Escape==0])/length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="SM"])*100, 0),

  round(length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="pH_drop+SM" &
Saur.Cmae.Avoid$Escape==0])/length(Saur.Cmae.Avoid$Escape[Saur.Cmae.Avoid$Treatment=="pH_drop+SM"])*100, 0))


Saur.Cmae.yes.Escape <- 100-Saur.Cmae.no.Escape



Saur.Cmae.Escape.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Escape"= c(Saur.Cmae.yes.Escape),

          "No Escape"=c(Saur.Cmae.no.Escape))


Saur.Cmae.Escape.percent %>%

  kbl(digits = 4,caption = "Percentage of escape/no escape in Saur/Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")
```

```

The escaping response of Cmae is plotted in function of predictors.

```{r Cmae escaping plot, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)
# Cmae - escaping per treatment across all donors
p_glz.Cmae.escaping.Treat <- ggplot(data = Cmae.Avoid, aes(x=Sort_treatment, fill=factor(escape_treat)))+
  geom_bar(position="fill", colour = "black") +
  scale_fill_manual("Escaping", values = alpha(c("#1F968BFF",  "#1F968BFF", "#73D055FF",    "#73D055FF", "#FDE725FF",  "#FDE725FF", "#D8DE29FF",  "#D8DE29FF"), c( 0.6,1,0.6,1,0.6,1,0.6, 1)),labels = c("No", "Yes", "No", "Yes", "No", "Yes", "No", "Yes")) +
  labs(x=NULL, y = "Escaping %")+
  scale_x_discrete(labels=c("CM", "pH drop", "SM", "pH drop\n+SM")) +
  theme_classic()+
  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
  own_theme()

# Facet by Donor
p_glz.Saur.Cmae.escaping.facet <- p_glz.Cmae.escaping.Treat + theme_bw() + theme(legend.position="top",
                                  panel.border = element_rect(colour = "black", fill=NA, size=0.5),
                                  panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor) + own_theme() + theme(legend.position="none")
p_glz.Saur.Cmae.escaping.facet

```

Time-to-success

The Cox proportional hazard model is used to model the effects of predictors on the time response of Cmae.

```{r Cmae Cox time, echo=TRUE, warning=FALSE}
# Cmae across all donors
Cmae.Time.Coxph.null <- coxph(Surv(time_s, Censoring_status) ~ 1, data = Cmae.Time)
Cmae.Time.Coxph <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor, data = Cmae.Time)
Cmae.Time.Coxph.mod2 <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor+Water_use, data = Cmae.Time)
Cmae.Time.Coxph.mod3 <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor+Water_use+as.factor(Year), data = Cmae.Time)
anova(Cmae.Time.Coxph.null, Cmae.Time.Coxph, Cmae.Time.Coxph.mod2, Cmae.Time.Coxph.mod3, test="chisq")
```
```

```
Cmae.Time.anova <- anova(Cmae.Time.Coxph.null, Cmae.Time.Coxph, Cmae.Time.Coxph.mod2, Cmae.Time.Coxph.mod3, test="chisq")# Water use and year can be disregarded

Cmae.Time.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of time in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


# Crab size was not recorded in 2018. Checking whether it had an impact for the 2019 subset

Cmae.Time.Coxph.2019.null <- coxph(Surv(time_s, Censoring_status) ~ 1, data = Cmae.Time[Cmae.Time$Year=="Autumn_2019",])

Cmae.Time.Coxph.2019 <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor, data = Cmae.Time[Cmae.Time$Year=="Autumn_2019",])

Cmae.Time.Coxph.2019.size <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor+Crab_size, data = Cmae.Time[Cmae.Time$Year=="Autumn_2019",])

Cmae.Time.anova2 <- anova(Cmae.Time.Coxph.2019, Cmae.Time.Coxph.2019.size, test="chisq") # Crab size can be disregarded

anova(Cmae.Time.Coxph.2019, Cmae.Time.Coxph.2019.size, test="chisq") # Crab size can be disregarded


Cmae.Time.anova2 %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of time in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of Cmae.Time.Coxph model = ", AIC(Cmae.Time.Coxph)) # AIC

as.data.frame(summary(Cmae.Time.Coxph)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of Cox proportional model for the time in Cmae") %>%

  kable_classic(full_width = F, html_font = "Cambria")


# To have a hazard ratio table

coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor, data = Cmae.Time) %>%

  gtsummary::tbl_regression(exp = TRUE)


# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

plot_model(Cmae.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor"))


as.data.frame(emmeans( Cmae.Time.Coxph, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the Cox proportional model for the time in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( Cmae.Time.Coxph, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the Cox proportional model for the time in Cmae - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(summary(emmeans( Cmae.Time.Coxph, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts)%>%
```

```
  kbl(digits = 4,caption = "Summary of pairwise tests for the Cox proportional model for the time in Cmae - model term donor|(pH drop:Stress metabolites)") %>%
  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(emmeans( Cmae.Time.Coxph, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%
  kbl(digits = 4,caption = "Statistic values for the Cox proportional for the time in Cmae - model term donor|(pH drop:Stress metabolites)") %>%
  kable_classic(full_width = F, html_font = "Cambria")

glz.Cmae.time_model_data <- as.data.frame(get_model_data(Cmae.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor")))
colnames(glz.Cmae.time_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')
glz.Cmae.time_model_data %>%
  kbl(digits = 4,caption = "Predicted model values for the time in Cmae according to the Cox proportional model") %>%
  kable_classic(full_width = F, html_font = "Cambria")
```

The time response of Cmae is plotted using hazard ratio plots and Kaplan-Meier survival curves.

```{r Cmae time plot, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)
# Forest
ggforest(Cmae.Time.Coxph, data=Cmae.Time)

# Raw Kaplan-Meier curve of all the subpopulations (facetting by Donor type)
Cmae.Time.Survfit.Treat <- survfit( Surv(time_s, Censoring_status) ~ pH_drop+SM+Donor, data = Cmae.Time )
Cmae.Time.Survfit.Treat.plot <- ggsurvplot(Cmae.Time.Survfit.Treat, conf.int = T, fun = "event",
                                           ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("npg"),
 legend = "bottom",
 legend.title = "Treatment",
xlim = c(0, 300),
             break.x.by = 60,
              risk.table.height=.3,
 tables.theme = theme_cleantable())

Cmae.Time.Survfit.Treat.plot2 <- Cmae.Time.Survfit.Treat.plot$plot + theme_bw() + theme(legend.position="top",
                                  panel.border = element_rect(colour = "black", fill=NA, size=0.5),
```

panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor)+
theme(plot.background = element_rect(color = "black"))

Cmae.Time.Survfit.Treat.plot2

```

## Hdiv

### Avoidance

The binomial generalised linear model is used to model the effects of predictors on the avoidance response of Hdiv.

```{r Hdiv avoidance glm, echo=TRUE, warning=FALSE}

# No behavioural avoidance was recorded for the CM condition in Hdiv/Hdiv subset. This leads to the analysis of the data using a model for each subset:

# For Hdiv group: pairwise comparisons of OA, SM, OA+SM. For Saur: pH x SM model can be used.

# Overall effect of donor:

glz.Hdiv.avoid.donor.null <- glm(Avoidance ~ 1, data=Hdiv.Avoid,family=binomial(link = "logit"))

glz.Hdiv.avoid.donor <- glm(Avoidance ~ Donor, data=Hdiv.Avoid,family=binomial(link = "logit"))

glz.Hdiv.avoid.donor.mod2 <- glm(Avoidance ~ Donor+Water_use, data=Hdiv.Avoid,family=binomial(link = "logit"))

glz.Hdiv.avoid.anova <- anova(glz.Hdiv.avoid.donor.null, glz.Hdiv.avoid.donor, glz.Hdiv.avoid.donor.mod2, test="Chisq") # water use is kept in the model

anova(glz.Hdiv.avoid.donor.null, glz.Hdiv.avoid.donor, glz.Hdiv.avoid.donor.mod2, test="Chisq")

glz.Hdiv.avoid.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of avoidance in Hdiv") %>%

  kable_classic(full_width = F, html_font = "Cambria")

paste("AIC of glz.Hdiv.avoid.donor.mod2 model = ", AIC(glz.Hdiv.avoid.donor.mod2)) # AIC

as.data.frame(summary(glz.Hdiv.avoid.donor.mod2)$coefficients) %>%

  kbl(digits = 4,caption = "Summary of binomial generalised linear model for the avoidance in Hdiv") %>%

  kable_classic(full_width = F, html_font = "Cambria")

# Hdiv/Hdiv subset

glz.Hdiv.Hdiv.avoid.null <- glm(Avoidance ~ 1, data=Hdiv.Hdiv.Avoid,family=binomial(link = "logit"))

glz.Hdiv.Hdiv.avoid <- glm(Avoidance ~ Treatment, data=Hdiv.Hdiv.Avoid,family=binomial(link = "logit")) # do in function of 'treatments' because we don't have the CM group for the 2-way design here

glz.Hdiv.Hdiv.avoid.mod2 <- glm(Avoidance ~ Treatment+Water_use, data=Hdiv.Hdiv.Avoid,family=binomial(link = "logit"))

glz.Hdiv.Hdiv.avoid.anova <- anova(glz.Hdiv.Hdiv.avoid.null, glz.Hdiv.Hdiv.avoid, glz.Hdiv.Hdiv.avoid.mod2, test="Chisq") # The water use can be disregarded

glz.Hdiv.Hdiv.avoid.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of avoidance in Hdiv/Hdiv") %>%

```
  kable_classic(full_width = F, html_font = "Cambria")

Pairwise comparisons
as.data.frame(emmeans(glz.Hdiv.Hdiv.avoid, pairwise ~ Treatment, adjust="fdr")$emmeans) %>%
  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the avoidance in Hdiv") %>%
  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(emmeans(glz.Hdiv.Hdiv.avoid, pairwise ~ Treatment, adjust="fdr")$contrasts) %>%
  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the avoidance in Hdiv") %>%
  kable_classic(full_width = F, html_font = "Cambria")

# Saur/Hdiv subset
glz.Saur.Hdiv.avoid.null <- glm(Avoidance ~ 1, data=Saur.Hdiv.Avoid,family=binomial(link = "logit"))
glz.Saur.Hdiv.avoid <- glm(Avoidance ~ pH_drop*SM, data=Saur.Hdiv.Avoid,family=binomial(link = "logit"))
glz.Saur.Hdiv.avoid.mod2 <- glm(Avoidance ~ pH_drop*SM+Water_use, data=Saur.Hdiv.Avoid,family=binomial(link = "logit"))
glz.Saur.Hdiv.avoid.anova <- anova(glz.Saur.Hdiv.avoid.null, glz.Saur.Hdiv.avoid, glz.Saur.Hdiv.avoid.mod2, test="Chisq") # The water use can be disregarded
glz.Saur.Hdiv.avoid.anova %>%
  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of avoidance in Saur/Hdiv") %>%
  kable_classic(full_width = F, html_font = "Cambria")

paste("AIC of glz.Saur.Hdiv.avoid model = ", AIC(glz.Saur.Hdiv.avoid)) # AIC
as.data.frame(summary(glz.Saur.Hdiv.avoid)$coefficients) %>%
  kbl(digits = 4,caption = "Summary of binomial generalised linear model for the avoidance in Saur/Hdiv") %>%
  kable_classic(full_width = F, html_font = "Cambria")

Post hoc tests
pH*SM term
as.data.frame(emmeans(glz.Saur.Hdiv.avoid, pairwise ~ pH_drop*SM, adjust="fdr")$emmeans) %>%
  kbl(digits = 4,caption = "Statistic values for the binomial generalised linear model for the avoidance in Saur/Hdiv") %>%
  kable_classic(full_width = F, html_font = "Cambria")

as.data.frame(emmeans(glz.Saur.Hdiv.avoid, pairwise ~ pH_drop*SM, adjust="fdr")$contrasts) %>%
  kbl(digits = 4,caption = "Summary of pairwise tests for the binomial generalised linear model for the avoidance in Saur/Hdiv") %>%
  kable_classic(full_width = F, html_font = "Cambria")

plot_model(glz.Saur.Hdiv.avoid, type = "pred", terms = c("pH_drop", "SM"))
glz.Saur.Hdiv.avoid_model_data <- as.data.frame(get_model_data(glz.Saur.Hdiv.avoid, type = "pred", terms = c("pH_drop", "SM")))
colnames(glz.Saur.Hdiv.avoid_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'SM')
glz.Saur.Hdiv.avoid_model_data %>%
```

```
  kbl(digits = 4,caption = "Predicted model values for the avoidance in Hdiv according to the  binomial generalised linear model") %>%
  kable_classic(full_width = F, html_font = "Cambria")

```

The percentages of avoidance/no avoidance are retrieved for each group in Hdiv.

```{r Hdiv/Hdiv avoidance percent, echo=TRUE, warning=FALSE}

# Hdiv - per donor
Hdiv.no.Avoid.Donor <- c(
  round(length(Hdiv.Avoid$Avoid[Hdiv.Avoid$Donor=="Saur" &
Hdiv.Avoid$Avoid==0])/length(Hdiv.Avoid$Avoid[Hdiv.Avoid$Donor=="Saur"])*100, 0),
  round(length(Hdiv.Avoid$Avoid[Hdiv.Avoid$Donor=="Hdiv" &
Hdiv.Avoid$Avoid==0])/length(Hdiv.Avoid$Avoid[Hdiv.Avoid$Donor=="Hdiv"])*100, 0))

Hdiv.yes.Avoid.Donor <- 100-Hdiv.no.Avoid.Donor


Hdiv.yes.Avoid.percent <- data.frame("Group"=c("Saur", "Hdiv"), "Avoidance"= c(Hdiv.yes.Avoid.Donor),
          "No avoidance"=c(Hdiv.no.Avoid.Donor))
Hdiv.yes.Avoid.percent %>%
  kbl(digits = 4,caption = "Percentage of avoidance/no avoidance for Hdiv in function of donor") %>%
  kable_classic(full_width = F, html_font = "Cambria")


# Hdiv/Hdiv

Hdiv.Hdiv.no.Avoid <- c(
  round(length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="CM" &
Hdiv.Hdiv.Avoid$Avoid==0])/length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="CM"])*100, 0),
  round(length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="pH_drop" &
Hdiv.Hdiv.Avoid$Avoid==0])/length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="SM" &
Hdiv.Hdiv.Avoid$Avoid==0])/length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="SM"])*100, 0),
  round(length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="pH_drop+SM" &
Hdiv.Hdiv.Avoid$Avoid==0])/length(Hdiv.Hdiv.Avoid$Avoid[Hdiv.Hdiv.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Hdiv.Hdiv.yes.Avoid <- 100-Hdiv.Hdiv.no.Avoid


 Hdiv.Hdiv.avoid.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Avoidance"= c(Hdiv.Hdiv.yes.Avoid),
          "No Avoidance"=c(Hdiv.Hdiv.no.Avoid))
```

```
Hdiv.Hdiv.avoid.percent %>%
  kbl(digits = 4,caption = "Percentage of avoidance/no avoidance for Hdiv/Hdiv in function of treatments") %>%
  kable_classic(full_width = F, html_font = "Cambria")



# Saur-Hdiv
Saur.Hdiv.no.Avoid <- c(
  round(length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="CM" &
Saur.Hdiv.Avoid$Avoid==0])/length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="CM"])*100, 0),
  round(length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="pH_drop" &
Saur.Hdiv.Avoid$Avoid==0])/length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="pH_drop"])*100, 0),
  round(length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="SM" &
Saur.Hdiv.Avoid$Avoid==0])/length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="SM"])*100, 0),
  round(length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="pH_drop+SM" &
Saur.Hdiv.Avoid$Avoid==0])/length(Saur.Hdiv.Avoid$Avoid[Saur.Hdiv.Avoid$Treatment=="pH_drop+SM"])*100, 0))

Saur.Hdiv.yes.Avoid <- 100-Saur.Hdiv.no.Avoid

Saur.Hdiv.avoid.percent <- data.frame("Group"=c("CM", "pH_drop", "SM", "pH_drop+SM"), "Avoidance %"= c(Saur.Hdiv.yes.Avoid),
          "No Avoidance %"=c(Saur.Hdiv.no.Avoid))
Saur.Hdiv.avoid.percent %>%
  kbl(digits = 4,caption = "Percentage of avoidance/no avoidance for Saur/Hdiv in function of treatments") %>%
  kable_classic(full_width = F, html_font = "Cambria")



```

The avoidance response of Hdiv is plotted in function of predictors. Note that due to a missing treatment (CM in Saur/Hdiv), two models are used to estimate (1) the overall effect of donor, and (2) the effects of treatments within each donor group.

```{r Hdiv/Hdiv avoidance plot, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)
# Hdiv - avoid per treatment across all donors
p_glz.Hdiv.avoid.Treat <- ggplot(data = Hdiv.Avoid, aes(x=Sort_treatment, fill=factor(avoid_treat)))+
  geom_bar(position="fill", colour = "black") +
  scale_fill_manual("Avoidance", values = alpha(c("#1F968BFF",  "#1F968BFF", "#73D055FF",    "#73D055FF", "#FDE725FF",  "#FDE725FF",
"#D8DE29FF",  "#D8DE29FF"), c( 0.6,1,0.6,1,0.6,1,0.6, 1)),labels = c("No", "Yes", "No", "Yes", "No", "Yes", "No", "Yes")) +
  labs(x=NULL, y = "Avoidance %")+
  scale_x_discrete(labels=c("CM", "pH drop", "SM", "pH drop\n+SM")) +
  theme_classic()+
  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
```

```
  own_theme()

# Facet by Donor
p_glz.Saur.Hdiv.avoid.facet <- p_glz.Hdiv.avoid.Treat + theme_bw() + theme(legend.position="top",
                                panel.border = element_rect(colour = "black", fill=NA, size=0.5),
                                panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor) + own_theme() + theme(legend.position="none")
p_glz.Saur.Hdiv.avoid.facet

# Covariate - water use
p_glz.Hdiv.avoid.wateruse <- ggplot(data = Hdiv.Avoid, aes(x=Water_use, fill=factor(Avoidance)))+
  geom_bar(position="fill", colour = "black") +
     scale_fill_grey(start=0.85, end=0.7) +
  labs(x="Water use", y = "Avoidance %")+
  theme_classic()+
  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.5), labels = c(0,50,100))+
  own_theme()
 p_glz.Hdiv.avoid.wateruse
```

Time-to-success

The Cox proportional hazard model is used to model the effects of predictors on the time response of Hdiv.

```{r Hdiv Cox time, echo=TRUE, warning=FALSE}
# Hdiv across all donors
Hdiv.Time.Coxph.null <- coxph(Surv(time_s, Censoring_status) ~ 1, data = Hdiv.Time)
Hdiv.Time.Coxph <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor, data = Hdiv.Time)
Hdiv.Time.Coxph.mod2 <- coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor+Water_use, data = Hdiv.Time)
Hdiv.Time.anova <- anova(Hdiv.Time.Coxph.null, Hdiv.Time.Coxph, Hdiv.Time.Coxph.mod2, test="chisq") # Water use can be disregarded
anova(Hdiv.Time.Coxph.null, Hdiv.Time.Coxph, Hdiv.Time.Coxph.mod2, test="chisq")

Hdiv.Time.anova %>%
  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of time in Hdiv") %>%
  kable_classic(full_width = F, html_font = "Cambria")

paste("AIC of Hdiv.Time.Coxph model = ", AIC(Hdiv.Time.Coxph)) # AIC
as.data.frame(summary(Hdiv.Time.Coxph)$coefficients) %>%
  kbl(digits = 4,caption = "Summary of Cox proportional model for the time in Hdiv") %>%
```

```
  kable_classic(full_width = F, html_font = "Cambria")


# To have a hazard ratio table

coxph(Surv(time_s, Censoring_status) ~ pH_drop*SM*Donor, data = Hdiv.Time) %>%

  gtsummary::tbl_regression(exp = TRUE)


# Post hoc tests. Not all tests are of interest. Only tests of interest are retrieved using 'within group segregation' '|' with emmeans

plot_model(Hdiv.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor"))

as.data.frame(emmeans( Hdiv.Time.Coxph, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the Cox proportional model for the time in Hdiv - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( Hdiv.Time.Coxph, pairwise ~ (pH_drop*SM)|Donor, adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the Cox proportional model for the time in Hdiv - model term (pH drop:Stress metabolites)|donor") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(summary(emmeans( Hdiv.Time.Coxph, pairwise ~ Donor|(pH_drop*SM), adjust="fdr"), by=NULL, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of pairwise tests for the Cox proportional model for the time in Hdiv - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


as.data.frame(emmeans( Hdiv.Time.Coxph, pairwise ~ Donor|(pH_drop*SM), adjust="fdr")$emmeans) %>%

  kbl(digits = 4,caption = "Statistic values for the Cox proportional for the time in Hdiv - model term donor|(pH drop:Stress metabolites)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


glz.Hdiv.time_model_data <- as.data.frame(get_model_data(Hdiv.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor")))

colnames(glz.Hdiv.time_model_data) <- c('pH drop', 'Predicted frequency', 'SE', 'lower CI', 'upper CI', 'Stress metabolites', 'Donor', 'SM')

glz.Cmae.time_model_data %>%

  kbl(digits = 4,caption = "Predicted model values for the time in Hdiv according to the Cox proportional model") %>%

  kable_classic(full_width = F, html_font = "Cambria")


```


```{r Hdiv time plot, echo=TRUE, warning=FALSE}

detach("package:cowplot", unload=TRUE)

library(cowplot)
```

```
# Forest plot
ggforest(Hdiv.Time.Coxph, data=Hdiv.Time)

# Raw Kaplan-Meier curve of all the subpopulations (facetting by Donor type)
Hdiv.Time.Survfit.Treat <- survfit( Surv(time_s, Censoring_status) ~ pH_drop+SM+Donor, data = Hdiv.Time )
Hdiv.Time.Survfit.Treat.plot <- ggsurvplot(Hdiv.Time.Survfit.Treat, conf.int = T, fun = "event",
                                           ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("npg"),
 legend = "bottom",
 legend.title = "Treatment",
xlim = c(0, 300),
ylim=c(0,1),
            break.x.by = 60,
             risk.table.height=.3,
 tables.theme = theme_cleantable())

Hdiv.Time.Survfit.Treat.plot2 <- Hdiv.Time.Survfit.Treat.plot$plot + theme_bw() + theme(legend.position="top",
                                panel.border = element_rect(colour = "black", fill=NA, size=0.5),
                                panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ facet_grid(~Donor)+
theme(plot.background = element_rect(color = "black"))
Hdiv.Time.Survfit.Treat.plot2

```

Panel plot for of 'interaction plots'

All the interaction plots are placed together in one panel.

```{r plot model panel, echo=TRUE, warning=FALSE}
detach("package:cowplot", unload=TRUE)
library(cowplot)

Plot.model.panel.grid <- plot_grid(
  # Survival
  plot_model(Dpug.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
  plot_model(Cmae.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
  plot_model(Hdiv.Time.Coxph, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
  # Avoidance
  plot_model(glz.Dpug.avoid, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
```

```
 plot_model(glz.Cmae.avoid.mod2, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
 plot_model(glz.Saur.Hdiv.avoid, type = "pred", terms = c("pH_drop", "SM")) +sjPlot::theme_sjplot2(),
 # Escaping
 plot_model(glz.Dpug.escape.mod4, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
 plot_model(glz.Cmae.escape, type = "pred", terms = c("pH_drop", "SM", "Donor"))+sjPlot::theme_sjplot2() ,
# Freezing
 plot_model(glz.Dpug.freeze, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
 plot_model(glz.Cmae.freeze, type = "pred", terms = c("pH_drop", "SM", "Donor")) +sjPlot::theme_sjplot2(),
       align = "hv", nrow = 4, ncol=3, labels="AUTO") + theme(plot.background = element_rect(color = "black"))
Plot.model.panel.grid



Plot.covariate.grid <- plot_grid(
 p_glz.Dpug.escaping.size2,
 p_glz.Cmae.avoid.wateruse,
 p_glz.Hdiv.avoid.wateruse,
 align = "hv", nrow = 1, ncol=3, labels="AUTO") + theme(plot.background = element_rect(color = "black"))
Plot.covariate.grid



```

Next, we tested the hypothesis that the allocation of energy towards the avoidance response caused delays in the time response.

```{r correlation and effect of measured variables, echo=TRUE, warning=FALSE}

Dpug.Time.Avoidance <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Avoidance), data = Dpug.Time) # P=1.31e-09
Cmae.Time.Avoidance <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Avoidance), data = Cmae.Time) # P=4.25e-06
Hdiv.Time.Avoidance <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Avoidance), data = Hdiv.Time) # P=0.71

summary(coxph(Surv(time_s, Censoring_status) ~ as.factor(Avoidance), data = Dpug.Time))$coefficients %>%   kbl(digits = 4,caption = "Cox proportional model of relationship between avoidance and time in Dpug") %>%   kable_classic(full_width = F, html_font = "Cambria")

summary(coxph(Surv(time_s, Censoring_status) ~ as.factor(Avoidance), data = Cmae.Time))$coefficients %>%   kbl(digits = 4,caption = "Cox proportional model of relationship between avoidance and time in Cmae") %>%   kable_classic(full_width = F, html_font = "Cambria")

summary(coxph(Surv(time_s, Censoring_status) ~ as.factor(Avoidance), data = Hdiv.Time))$coefficients %>%   kbl(digits = 4,caption = "Cox proportional model of relationship between avoidance and time in Hdiv") %>%   kable_classic(full_width = F, html_font = "Cambria")

Dpug.Time.Escape <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Escape), data = Dpug.Time) # P=0.206
Dpug.Time.Freeze <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Freeze), data = Dpug.Time) # P=1.06e-11
```

```
summary(coxph(Surv(time_s, Censoring_status) ~ as.factor(Escape), data = Dpug.Time))$coefficients %>%  kbl(digits = 4,caption = "Cox proportional model of relationship between escape and time in Dpug") %>%  kable_classic(full_width = F, html_font = "Cambria")

summary(coxph(Surv(time_s, Censoring_status) ~ as.factor(Freeze), data = Dpug.Time))$coefficients %>%  kbl(digits = 4,caption = "Cox proportional model of relationship between escape and time in Cmae") %>%  kable_classic(full_width = F, html_font = "Cambria")

Cmae.Time.Escape <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Escape), data = Cmae.Time) # P=0.00471

Cmae.Time.Freeze <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Freeze), data = Cmae.Time) # P=0.000293

summary(coxph(Surv(time_s, Censoring_status) ~ as.factor(Escape), data = Cmae.Time))$coefficients

summary(coxph(Surv(time_s, Censoring_status) ~ as.factor(Freeze), data = Cmae.Time))$coefficients

Dpug.Time.Avoid.plot <- ggsurvplot(Dpug.Time.Avoidance, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
ylim=c(0,1),
 xlim = c(0, 300),
            break.x.by = 60,
             risk.table.height=.3,
 tables.theme = theme_cleantable())


Cmae.Time.Avoid.plot <- ggsurvplot(Cmae.Time.Avoidance, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
ylim=c(0,1),
 xlim = c(0, 300),
            break.x.by = 60,
             risk.table.height=.3,
 tables.theme = theme_cleantable())


Hdiv.Time.Avoid.plot <- ggsurvplot(Hdiv.Time.Avoidance, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
```

```
ylim=c(0,1),
 xlim = c(0, 300),
              break.x.by = 60,
               risk.table.height=.3,
 tables.theme = theme_cleantable())

Dpug.Time.Escape.plot <- ggsurvplot(Dpug.Time.Escape, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
ylim=c(0,1),
 xlim = c(0, 300),
              break.x.by = 60,
               risk.table.height=.3,
 tables.theme = theme_cleantable())



Dpug.Time.Freeze.plot <- ggsurvplot(Dpug.Time.Freeze, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
ylim=c(0,1),
 xlim = c(0, 300),
              break.x.by = 60,
               risk.table.height=.3,
 tables.theme = theme_cleantable())



Cmae.Time.Escape.plot <- ggsurvplot(Cmae.Time.Escape, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
ylim=c(0,1),
 xlim = c(0, 300),
              break.x.by = 60,
               risk.table.height=.3,
```

```
 tables.theme = theme_cleantable())


Cmae.Time.Freeze.plot <- ggsurvplot(Cmae.Time.Freeze, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
ylim=c(0,1),
 xlim = c(0, 300),
              break.x.by = 60,
               risk.table.height=.3,
 tables.theme = theme_cleantable())



Dpug.Time.Freeze.plot
Cmae.Time.Avoid.plot
Cmae.Time.Escape.plot
Cmae.Time.Freeze.plot


```


Supplementary analyses to invalidate the alternative hypothesis that stress metabolites are 'just' more concentrated control metabolites

First, the assumption of the preliminary data from 2018 (in green shore crabs only) was that the number of water uses is not impacting the 'time-to-success' analysis.


```{r water use 2018, echo=TRUE, warning=FALSE}
summary(coxph(Surv(time_s, Censoring_status) ~ Water_use, data = Cmae.Time[Cmae.Time$Year=="Autumn_2018",]))$coefficient %>% kbl(digits = 4,caption = "Cox proportional model of verifying that the water use could be used up to five times according to preliminary data") %>%   kable_classic(full_width = F, html_font = "Cambria")


p.reusedwater.2018Cmae.boxplot <- ggplot(Cmae.Time[Cmae.Time$Year=="Autumn_2018",], aes(x=as.factor(Water_use), y=time_s, fill=as.factor(Water_use))) +
  stat_boxplot(size=1.25, geom ='errorbar', width = 0.1)+
  geom_jitter(  position = "jitter",fill = "grey", cex=6, pch=21)+
  geom_violin(alpha=0.5)+
  geom_boxplot(alpha=0.2, size=1.25, fill=c("grey40", "grey50", "grey60", "grey70", "grey80"), outlier.shape = NA)+
  labs(x="Number of Water uses", y = "Time-to-success (s)")+
  theme_classic()+
  scale_fill_manual("", values =c("grey40", "grey50", "grey60", "grey70", "grey80"), labels = c("1", "2", "3", "4", "5"),
                    guide=FALSE)+own_theme_hist()+
```

```
  stat_summary(fun.data="mean_se", fun.args = list(mult=1),
               geom="pointrange", size=2, fill="black")


coxph.reusedwater.2018Cmae <- survfit(Surv(time_s, Censoring_status) ~ as.factor(Water_use), data =
Cmae.Time[Cmae.Time$Year=="Autumn_2018",]) # P=0.3727

p.reusedwater.2018Cmae.survplot <- ggsurvplot(coxph.reusedwater.2018Cmae, conf.int = T, fun = "event",  ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
ylim=c(0,1),
 xlim = c(0, 300),
              break.x.by = 60,
               risk.table.height=.3,
 tables.theme = theme_cleantable())



# Show residuals of Dpug, Cmae, Hdiv models
Dpug.Time.res.t <- bestNormalize::bestNormalize(Dpug.Time.Coxph$residuals)$x.t
Cmae.Time.res.t <- bestNormalize::bestNormalize(Cmae.Time.Coxph$residuals)$x.t
Hdiv.Time.res.t <- bestNormalize::bestNormalize(Hdiv.Time.Coxph$residuals)$x.t

hist(Dpug.Time.res.t)
hist(Cmae.Time.res.t)
hist(Hdiv.Time.res.t)

Dpug.Time.res.dat <- data.frame("Time.res.t"=Dpug.Time.res.t, "Water_use"=Dpug.Time$Water_use)
Cmae.Time.res.dat <- data.frame("Time.res.t"=Cmae.Time.res.t, "Water_use"=Cmae.Time$Water_use)
Hdiv.Time.res.dat <- data.frame("Time.res.t"=Hdiv.Time.res.t, "Water_use"=Hdiv.Time$Water_use)

summary(aov(lm(data=Dpug.Time.res.dat, Time.res.t~Water_use)))
summary(aov(lm(data=Cmae.Time.res.dat, Time.res.t~Water_use)))
summary(aov(lm(data=Hdiv.Time.res.dat, Time.res.t~Water_use)))

ggplot(Cmae.Time[Cmae.Time$Year=="Autumn_2018",], aes(x=as.factor(Water_use), y=time_s, fill=as.factor(Water_use))) +
  stat_boxplot(size=1.25, geom ='errorbar', width = 0.1)+
  geom_jitter(  position = "jitter",fill = "grey", cex=6, pch=21)+
  geom_violin(alpha=0.5)+
  geom_boxplot(alpha=0.2, size=1.25, fill=c("grey40", "grey50", "grey60", "grey70", "grey80"), outlier.shape = NA)+
  labs(x="Number of Water uses", y = "Time-to-success (s)")+
  theme_classic()+
```

```
  scale_fill_manual("", values =c("grey40", "grey50", "grey60", "grey70", "grey80"), labels = c("1", "2", "3", "4", "5"),
                    guide=FALSE)+own_theme_hist()+
  stat_summary(fun.data="mean_se", fun.args = list(mult=1),
               geom="pointrange", size=2, fill="black")


ggplot(Cmae.Time[Cmae.Time$Year=="Autumn_2018",], aes(x=as.factor(Water_use), y=time_s, fill=as.factor(Water_use))) +
  stat_boxplot(size=1.25, geom ='errorbar', width = 0.1)+
  geom_jitter(  position = "jitter",fill = "grey", cex=6, pch=21)+
  geom_violin(alpha=0.5)+
  geom_boxplot(alpha=0.2, size=1.25, fill=c("grey40", "grey50", "grey60", "grey70", "grey80"), outlier.shape = NA)+
  labs(x="Number of Water uses", y = "Time-to-success (s)")+
  theme_classic()+
  scale_fill_manual("", values =c("grey40", "grey50", "grey60", "grey70", "grey80"), labels = c("1", "2", "3", "4", "5"),
                    guide=FALSE)+own_theme_hist()+
  stat_summary(fun.data="mean_se", fun.args = list(mult=1),
               geom="pointrange", size=2, fill="black")


ggplot(Cmae.Time[Cmae.Time$Year=="Autumn_2018",], aes(x=as.factor(Water_use), y=time_s, fill=as.factor(Water_use))) +
  stat_boxplot(size=1.25, geom ='errorbar', width = 0.1)+
  geom_jitter(  position = "jitter",fill = "grey", cex=6, pch=21)+
  geom_violin(alpha=0.5)+
  geom_boxplot(alpha=0.2, size=1.25, fill=c("grey40", "grey50", "grey60", "grey70", "grey80"), outlier.shape = NA)+
  labs(x="Number of Water uses", y = "Time-to-success (s)")+
  theme_classic()+
  scale_fill_manual("", values =c("grey40", "grey50", "grey60", "grey70", "grey80"), labels = c("1", "2", "3", "4", "5"),
                    guide=FALSE)+own_theme_hist()+
  stat_summary(fun.data="mean_se", fun.args = list(mult=1),
               geom="pointrange", size=2, fill="black")

```
```

After seeing the effects of stress metabolites on the behaviour, we wondered whether these effects are due to putative stress metabolites, or to higher concentration of control metabolites induced by a stress-related high metabolism. In order to discriminate stress metabolites to more regularly excreted metabolites, we compared CM and SM to CM100%. This was performed in Saur/Hdiv and Saur/Dpug groups only.

```{r SuppData preparation, echo=TRUE, warning=FALSE}

Conc_Dpug.Time <- subset(Conc_Dpug, is.na(Conc_Dpug$time_s)==F & is.na(Conc_Dpug$Success)==F) # Remove NA

Conc_Hdiv.Time <- subset(Conc_Hdiv, is.na(Conc_Hdiv$time_s)==F & is.na(Conc_Hdiv$Success)==F) # Remove NA
```
```

```
Conc_Dpug.Avoid <- subset(Conc_Dpug, is.na(Conc_Dpug$Avoid)==F) # Remove NA

Conc_Hdiv.Avoid <- subset(Conc_Hdiv, is.na(Conc_Hdiv$Avoid)==F) # Remove NA


# Supp. data - Saur/Dpug - Success per Treatment

Conc_Dpug.Avoid$escape_supp[Conc_Dpug.Avoid$Treatment=="CM" & Conc_Dpug.Avoid$Escape==0] <- "1CM_NO"

Conc_Dpug.Avoid$escape_supp[Conc_Dpug.Avoid$Treatment=="CM" & Conc_Dpug.Avoid$Escape==1] <- "2CM_NO"

Conc_Dpug.Avoid$escape_supp[Conc_Dpug.Avoid$Treatment=="CM100%" & Conc_Dpug.Avoid$Escape==0] <- "3CM100%_NO"

Conc_Dpug.Avoid$escape_supp[Conc_Dpug.Avoid$Treatment=="CM100%" & Conc_Dpug.Avoid$Escape==1] <- "4CM100%_YES"

Conc_Dpug.Avoid$escape_supp[Conc_Dpug.Avoid$Treatment=="SM" & Conc_Dpug.Avoid$Escape==0] <- "5SM_NO"

Conc_Dpug.Avoid$escape_supp[Conc_Dpug.Avoid$Treatment=="SM" & Conc_Dpug.Avoid$Escape==1] <- "6SM_YES"


# Supp. data - Saur/Hdiv - Success per Treatment

Conc_Hdiv.Avoid$avoid_supp[Conc_Hdiv.Avoid$Treatment=="CM" & Conc_Hdiv.Avoid$Avoidance==0] <- "1CM_NO"

Conc_Hdiv.Avoid$avoid_supp[Conc_Hdiv.Avoid$Treatment=="CM" & Conc_Hdiv.Avoid$Avoidance==1] <- "2CM_NO"

Conc_Hdiv.Avoid$avoid_supp[Conc_Hdiv.Avoid$Treatment=="CM100%" & Conc_Hdiv.Avoid$Avoidance==0] <- "3CM100%_NO"

Conc_Hdiv.Avoid$avoid_supp[Conc_Hdiv.Avoid$Treatment=="CM100%" & Conc_Hdiv.Avoid$Avoidance==1] <- "4CM100%_YES"

Conc_Hdiv.Avoid$avoid_supp[Conc_Hdiv.Avoid$Treatment=="SM" & Conc_Hdiv.Avoid$Avoidance==0] <- "5SM_NO"

Conc_Hdiv.Avoid$avoid_supp[Conc_Hdiv.Avoid$Treatment=="SM" & Conc_Hdiv.Avoid$Avoidance==1] <- "6SM_YES"


```

Saur/Dpug

Escaping

```{r SuppData CM100 Saur/Dpug escape, echo=TRUE, warning=FALSE}

glz.Conc.Dpug.Escape.null  <- glm(Escape ~ 1, data=Conc_Dpug.Avoid,family=binomial(link = "logit"))

glz.Conc.Dpug.Escape  <- glm(Escape ~ Treatment, data=Conc_Dpug.Avoid,family=binomial(link = "logit"))

glz.Conc.Dpug.Escape.mod2 <- glm(Escape ~ Treatment+Water_use, data=Conc_Dpug.Avoid,family=binomial(link = "logit"))

glz.Conc.Dpug.Escape.mod3 <- glm(Escape ~ Treatment+Water_use+Crab_size, data=Conc_Dpug.Avoid,family=binomial(link = "logit"))

glz.Conc.Dpug.Escape.anova <- anova(glz.Conc.Dpug.Escape.null, glz.Conc.Dpug.Escape, glz.Conc.Dpug.Escape.mod2,
glz.Conc.Dpug.Escape.mod3, test="Chisq") # water use and crab size can be disregarded

anova(glz.Conc.Dpug.Escape.null, glz.Conc.Dpug.Escape, glz.Conc.Dpug.Escape.mod2, glz.Conc.Dpug.Escape.mod3, test="Chisq")


glz.Conc.Dpug.Escape.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of escaping in Dpug (supplementary data including
CM100%)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of glz.Conc.Dpug.Escape model = ", AIC(glz.Conc.Dpug.Escape)) # AIC

as.data.frame(emmeans(glz.Conc.Dpug.Escape, pairwise~Treatment, adjust="fdr")$contrasts) %>%
```

```
  kbl(digits = 4,caption = "Summary of binomial generalised linear model for the escaping of Dpug (supplementary data including CM100%)") %>%
  kable_classic(full_width = F, html_font = "Cambria")
```

```{r CM100 Saur/Dpug escape plot, echo=TRUE, warning=FALSE}

p.glz.Conc.Dpug.escape <- ggplot(data = Conc_Dpug.Avoid, aes(x=Treatment, fill=factor(escape_supp)))+
  geom_bar(position="fill", colour = "black") +
  scale_fill_manual("Escape", values = alpha(c("#1F968BFF",  "#1F968BFF",  "steelblue4", "steelblue4",   "#FDE725FF", "#FDE725FF"), c( 0.6, 1,0.6,1,0.6, 1)),labels = c("No", "Yes"  , "No", "Yes", "No", "Yes")) +
  labs(x=NULL, y = "Escape %")+
  scale_x_discrete(labels=c( "CM", "CM-100%", "SM")) +  theme_classic()+
  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.25), labels = c(0,25,50,75,100))+own_theme()
 p.glz.Conc.Dpug.escape

escape.conc.Dpug.no.escape <- c(
  round(length(Conc_Dpug.Avoid$Escape[Conc_Dpug.Avoid$Treatment=="CM" & Conc_Dpug.Avoid$Escape==0])/length(Conc_Dpug.Avoid$Escape[Conc_Dpug.Avoid$Treatment=="CM"])*100, 0),
  round(length(Conc_Dpug.Avoid$escape[Conc_Dpug.Avoid$Treatment=="CM100%" & Conc_Dpug.Avoid$Escape==0])/length(Conc_Dpug.Avoid$Escape[Conc_Dpug.Avoid$Treatment=="CM100%"])*100, 0),
  round(length(Conc_Dpug.Avoid$escape[Conc_Dpug.Avoid$Treatment=="SM" & Conc_Dpug.Avoid$Escape==0])/length(Conc_Dpug.Avoid$Escape[Conc_Dpug.Avoid$Treatment=="SM"])*100, 0))

escape.conc.Dpug.yes.escape <- 100-escape.conc.Dpug.no.escape
escape.conc.Dpug.percent <- data.frame("Group"=c("CM", "CM100%", "SM"), "Escape"= c(escape.conc.Dpug.yes.escape),
          "No escape"=c(escape.conc.Dpug.no.escape))
 escape.conc.Dpug.percent %>%
  kbl(digits = 4,caption = "Percentages of escaping of Dpug (supplementary data including CM100%)") %>%
  kable_classic(full_width = F, html_font = "Cambria")
```

Time-to-success

```{r CM100 Dpug, echo=TRUE, warning=FALSE}

glz.Conc.Dpug.Time.null  <- coxph(Surv(time_s, Censoring_status) ~ 1, data = Conc_Dpug.Time)
```

```
glz.Conc.Dpug.Time  <- coxph(Surv(time_s, Censoring_status) ~ Treatment, data = Conc_Dpug.Time)

glz.Conc.Dpug.Time.mod2 <- coxph(Surv(time_s, Censoring_status) ~ Treatment+Crab_size, data = Conc_Dpug.Time)

glz.Conc.Dpug.Time.mod3 <- coxph(Surv(time_s, Censoring_status) ~ Treatment+Crab_size+Water_use, data = Conc_Dpug.Time)

glz.Conc.Dpug.Time.anova <- anova(glz.Conc.Dpug.Time.null, glz.Conc.Dpug.Time, glz.Conc.Dpug.Time.mod2, glz.Conc.Dpug.Time.mod3, test="Chisq") # Crab size and water use can be disregarded

anova(glz.Conc.Dpug.Time.null, glz.Conc.Dpug.Time, glz.Conc.Dpug.Time.mod2, glz.Conc.Dpug.Time.mod3, test="Chisq")


glz.Conc.Dpug.Time.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of time in Dpug (supplementary data including CM100%)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of glz.Conc.Dpug.Time model = ", AIC(glz.Conc.Dpug.Time)) # AIC

as.data.frame(emmeans(glz.Conc.Dpug.Time, pairwise~Treatment, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of Cox proportional model for the time of Dpug (supplementary data including CM100%)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


# alternatively

Dpug.Conc.Time.pairwise_survdiff <- pairwise_survdiff(Surv(time_s, Censoring_status) ~  Treatment,

    data = Conc_Dpug.Time, p.adjust.method = 'fdr')



```



```{r DPug CM100 time plot, echo=TRUE, warning=FALSE}

# ggforest

ggforest(coxph(Surv(time_s, Censoring_status) ~ Treatment, data = Conc_Dpug.Time), data=Conc_Dpug.Time)


# Raw Kaplan-Meier curve

Conc.Dpug.Time.Survfit <- survfit( Surv(time_s, Censoring_status) ~ Treatment, data = Conc_Dpug.Time )

Conc.Dpug.Time.Survfit.plot <- ggsurvplot(Conc.Dpug.Time.Survfit, conf.int = T, fun = "event",

                                          ylab="Success probability",

 size = 1,

 linetype = "strata",

 palette = c("viridis"),

 legend = "bottom",

 legend.title = "Treatment",

 legend.labs = c("CM (Saur/Dpug)","CM100% (Saur/Dpug)", "SM (Saur/Dpug)"),

xlim = c(0, 300),

            break.x.by = 60,
```

```
            risk.table.height=.3,

 tables.theme = theme_cleantable())


Conc.Dpug.Time.Survfit.plot2 <- Conc.Dpug.Time.Survfit.plot$plot + theme_bw() + theme(legend.position="top",

                                panel.border = element_rect(colour = "black", fill=NA, size=0.5),

                                panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ theme(plot.background =
element_rect(color = "black"))

Conc.Dpug.Time.Survfit.plot2
```

Saur/Hdiv

Avoidance

```{r SuppData CM100 Saur/Hdiv avoidance, echo=TRUE, warning=FALSE}

glz.Conc.Hdiv.Avoid.null  <- glm(Avoidance ~ 1, data=Conc_Hdiv.Avoid,family=binomial(link = "logit"))

glz.Conc.Hdiv.Avoid  <- glm(Avoidance ~ Treatment, data=Conc_Hdiv.Avoid,family=binomial(link = "logit"))

glz.Conc.Hdiv.Avoid.mod2 <- glm(Avoidance ~ Treatment+Water_use, data=Conc_Hdiv.Avoid,family=binomial(link = "logit"))

glz.Conc.Hdiv.Avoid.anova <- anova(glz.Conc.Hdiv.Avoid.null, glz.Conc.Hdiv.Avoid, glz.Conc.Hdiv.Avoid.mod2, test="Chisq") # water use can
be disregarded

anova(glz.Conc.Hdiv.Avoid.null, glz.Conc.Hdiv.Avoid, glz.Conc.Hdiv.Avoid.mod2, test="Chisq")

glz.Conc.Hdiv.Avoid.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of avoidance in Hdiv (supplementary data including
CM100%)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of glz.Conc.Hdiv.Avoid model = ", AIC(glz.Conc.Hdiv.Avoid)) # AIC

as.data.frame(emmeans(glz.Conc.Hdiv.Avoid, pairwise~Treatment, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of binomial generalised linear model of avoidance of Hdiv (supplementary data including CM100%)")
%>%

  kable_classic(full_width = F, html_font = "Cambria")



```


```{r plot SuppData CM100 Saur/Hdiv avoidance, echo=TRUE, warning=FALSE}


p.glz.Conc.Hdiv.avoid <- ggplot(data = Conc_Hdiv.Avoid, aes(x=Treatment, fill=factor(avoid_supp)))+

  geom_bar(position="fill", colour = "black") +

  scale_fill_manual("Avoidance", values = alpha(c("#1F968BFF",  "#1F968BFF",  "steelblue4", "steelblue4",   "#FDE725FF", "#FDE725FF"), c(
0.6, 1,0.4,1,0.3, 1)),labels = c("No", "Yes"  , "No", "Yes", "No", "Yes")) +
```

```
  labs(x=NULL, y = "Avoidance %")+

  scale_x_discrete(labels=c( "CM", "CM-100%", "SM")) +  theme_classic()+

  scale_y_continuous(limits = c(0,1), breaks = seq(0, 1, by = 0.25), labels = c(0,25,50,75,100))+own_theme()

 p.glz.Conc.Hdiv.avoid


avoid.conc.Hdiv.no.avoid <- c(

  round(length(Conc_Hdiv.Avoid$Avoidance[Conc_Hdiv.Avoid$Treatment=="CM" &
Conc_Hdiv.Avoid$Avoidance==0])/length(Conc_Hdiv.Avoid$Avoidance[Conc_Hdiv.Avoid$Treatment=="CM"])*100, 0),

  round(length(Conc_Hdiv.Avoid$Avoidance[Conc_Hdiv.Avoid$Treatment=="CM100%" &
Conc_Hdiv.Avoid$Avoidance==0])/length(Conc_Hdiv.Avoid$Avoidance[Conc_Hdiv.Avoid$Treatment=="CM100%"])*100, 0),

  round(length(Conc_Hdiv.Avoid$Avoidance[Conc_Hdiv.Avoid$Treatment=="SM" &
Conc_Hdiv.Avoid$Avoidance==0])/length(Conc_Hdiv.Avoid$Avoidance[Conc_Hdiv.Avoid$Treatment=="SM"])*100, 0))


avoid.conc.Hdiv.yes.avoid <- 100-avoid.conc.Hdiv.no.avoid


avoid.conc.Hdiv.percent <- data.frame("Group"=c("CM", "CM100%", "SM"), "Avoidance"= c(avoid.conc.Hdiv.yes.avoid),

          "No Avoidance"=c(avoid.conc.Hdiv.no.avoid))

avoid.conc.Hdiv.percent %>%

  kbl(digits = 4,caption = "Percentages of avoidance of Hdiv (supplementary data including CM100%)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


```


Time-to-success


```{r SuppData CM100 Saur/Hdiv time, echo=TRUE, warning=FALSE}


glz.Conc.Hdiv.Time.null  <- coxph(Surv(time_s, Censoring_status) ~ 1, data = Conc_Hdiv.Time)

glz.Conc.Hdiv.Time  <- coxph(Surv(time_s, Censoring_status) ~ Treatment, data = Conc_Hdiv.Time)

glz.Conc.Hdiv.Time.mod2 <- coxph(Surv(time_s, Censoring_status) ~ Treatment+Water_use, data = Conc_Hdiv.Time)

glz.Conc.Hdiv.Time.anova <- anova(glz.Conc.Hdiv.Time.null, glz.Conc.Hdiv.Time, glz.Conc.Hdiv.Time.mod2, test="Chisq") # Water use can be disregarded

anova(glz.Conc.Hdiv.Time.null, glz.Conc.Hdiv.Time, glz.Conc.Hdiv.Time.mod2, test="Chisq")


glz.Conc.Hdiv.Time.anova %>%

  kbl(digits = 4,caption = "Summary of Chis-quared test for model fit comparison of time in Hdiv (supplementary data including CM100%)") %>%

  kable_classic(full_width = F, html_font = "Cambria")


paste("AIC of glz.Conc.Hdiv.Time model = ", AIC(glz.Conc.Hdiv.Time)) # AIC

as.data.frame(emmeans(glz.Conc.Hdiv.Time, pairwise~Treatment, adjust="fdr")$contrasts) %>%

  kbl(digits = 4,caption = "Summary of Cox proportional model for the time of Hdiv (supplementary data including CM100%)") %>%
```

```
 kable_classic(full_width = F, html_font = "Cambria")



# alternatively
Hdiv.Conc.Time.pairwise_survdiff <- pairwise_survdiff(Surv(time_s, Censoring_status) ~  Treatment,
    data = Conc_Hdiv.Time, p.adjust.method = 'fdr')

```

```{r Hdiv CM100 plot, echo=TRUE, warning=FALSE}

# ggforest
ggforest(coxph(Surv(time_s, Censoring_status) ~ Treatment, data = Conc_Hdiv.Time), data=Conc_Hdiv.Time)

# Raw Kaplan-Meier curve
Conc.Hdiv.Time.Survfit <- survfit( Surv(time_s, Censoring_status) ~ Treatment, data = Conc_Hdiv.Time )
Conc.Hdiv.Time.Survfit.plot <- ggsurvplot(Conc.Hdiv.Time.Survfit, conf.int = T, fun = "event",
                                          ylab="Success probability",
 size = 1,
 linetype = "strata",
 palette = c("viridis"),
 legend = "bottom",
 legend.title = "Treatment",
 legend.labs = c("CM (Saur/Hdiv)","CM100% (Saur/Hdiv)", "SM (Saur/Hdiv)"),
xlim = c(0, 300),
             break.x.by = 60,
              risk.table.height=.3,
 tables.theme = theme_cleantable())

Conc.Hdiv.Time.Survfit.plot2 <- Conc.Hdiv.Time.Survfit.plot$plot + theme_bw() + theme(legend.position="top",
                                panel.border = element_rect(colour = "black", fill=NA, size=0.5),
                                panel.grid.major = element_blank(), panel.grid.minor = element_blank())+ theme(plot.background =
element_rect(color = "black"))
Conc.Hdiv.Time.Survfit.plot2
```


Panel plot from Supplementary data

A panel of graphical representations of the effects of CM100% is prepared for all the supplementary data.

```{r Supplementary data plot, echo=TRUE, warning=FALSE}

detach("package:cowplot", unload=TRUE)

library(cowplot)

SuppData_plot_grid <- plot_grid(

    p.glz.Conc.Dpug.escape+ theme(legend.position = "none"),

    p.glz.Conc.Hdiv.avoid+ theme(legend.position = "none"),

    Conc.Dpug.Time.Survfit.plot2,

    Conc.Hdiv.Time.Survfit.plot2,

    align = "h", nrow = 2, ncol=2, labels="AUTO", rel_heights = c(1/4, 1/2)) + theme(plot.background = element_rect(color = "black"))

SuppData_plot_grid


```


Model boostrapping


```{r bootstraps coxph, echo=TRUE}


# Boostrapping - show the peak of density of boostrap matching the pvalue or estimate


# Dpug subsets and models

set.seed(1) # So you can reproduce these results

Dpug.randomdf.list <- setNames(lapply(1:100,

                function(x) {

                  x <- Dpug.Time[sample(nrow(Dpug.Time), nrow(Dpug.Time)/2, replace = F),]; x }),

            paste0("attempt.", 1:100))


Dpug.randomdf.list.model <- lapply(X=Dpug.randomdf.list, FUN=coxph, formula = Surv(time_s, Censoring_status)~pH_drop*SM*Donor)


Dpug.randomdf.list.model.est <- as.data.frame(t(sapply(Dpug.randomdf.list.model, function(f) summary(f)$coefficients[,1]))) # get the estimates of each term for all models


# Cmae subsets and models

set.seed(2) # So you can reproduce these results

Cmae.randomdf.list <- setNames(lapply(1:100,

                function(x) {

                  x <- Cmae.Time[sample(nrow(Cmae.Time), nrow(Cmae.Time)/2, replace = F),]; x }),

            paste0("attempt.", 1:100))


Cmae.randomdf.list.model <- lapply(X=Cmae.randomdf.list, FUN=coxph, formula = Surv(time_s, Censoring_status)~pH_drop*SM*Donor)
```

```
Cmae.randomdf.list.model.est <- as.data.frame(t(sapply(Cmae.randomdf.list.model, function(f) summary(f)$coefficients[,1]))) # get the estimates of each term for all models


# Hdiv subsets and models


set.seed(3) # So you can reproduce these results

Hdiv.randomdf.list <- setNames(lapply(1:100,

                  function(x) {

                    x <- Hdiv.Time[sample(nrow(Hdiv.Time), nrow(Hdiv.Time)/2, replace = F),]; x }),

            paste0("attempt.", 1:100))

Hdiv.randomdf.list.model <- lapply(X=Hdiv.randomdf.list, FUN=coxph, formula = Surv(time_s, Censoring_status)~pH_drop*SM*Donor)


Hdiv.randomdf.list.model.est <- as.data.frame(t(sapply(Hdiv.randomdf.list.model, function(f) summary(f)$coefficients[,1]))) # get the estimates of each term for all models



```


```{r boostrap coxph ztest, echo=TRUE}


# one-sample z test

if(!require(distributions3)){install.packages("distributions3")}; library(distributions3)

z.testonesample = function(x, mu0, sigma){

   n = length(x)

   xbar = mean(x)


   z_stat <- sqrt(n)*(xbar - mu0) / (sigma)

   p_value=2*pnorm(-abs(z_stat))

   return(c("Z"=z_stat, "P"=p_value))}



# calculate the z-statistic


Dpug.randomdf.list.model.est.ztest.pH <- z.testonesample(x=Dpug.randomdf.list.model.est$pH_drop1,
mu0=as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,1],sigma= as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,3])

Dpug.randomdf.list.model.est.ztest.SM <- z.testonesample(x=Dpug.randomdf.list.model.est$SM1,
mu0=as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,1],sigma= as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,3])

Dpug.randomdf.list.model.est.ztest.Donor <- z.testonesample(x=Dpug.randomdf.list.model.est$DonorSaur,
mu0=as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,1],sigma= as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,3])
```

```
Cmae.randomdf.list.model.est.ztest.pH <- z.testonesample(x=Cmae.randomdf.list.model.est$pH_drop1,
mu0=as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,1],sigma= as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,3])

Cmae.randomdf.list.model.est.ztest.SM <-z.testonesample(x=Cmae.randomdf.list.model.est$SM1,
mu0=as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,1],sigma= as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,3])

Cmae.randomdf.list.model.est.ztest.Donor <-z.testonesample(x=Cmae.randomdf.list.model.est$DonorSaur,
mu0=as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,1],sigma= as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,3])


Hdiv.randomdf.list.model.est.ztest.pH <- z.testonesample(x=Hdiv.randomdf.list.model.est$pH_drop1,
mu0=as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,1],sigma= as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,3])

Hdiv.randomdf.list.model.est.ztest.SM <-z.testonesample(x=Hdiv.randomdf.list.model.est$SM1,
mu0=as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,1],sigma= as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,3])

Hdiv.randomdf.list.model.est.ztest.Donor <-z.testonesample(x=Hdiv.randomdf.list.model.est$DonorSaur,
mu0=as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,1],sigma= as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,3])


data.frame(

  "Species"=c(rep("Dpug", 3), rep("Cmae", 3), rep("Hdiv", 3)),

  "Factor"= rep(c("pH drop", "SM", "Donor"), 3),

  "Z"=c(Dpug.randomdf.list.model.est.ztest.pH[1], Dpug.randomdf.list.model.est.ztest.SM[1], Dpug.randomdf.list.model.est.ztest.Donor[1],

Cmae.randomdf.list.model.est.ztest.pH[1], Cmae.randomdf.list.model.est.ztest.SM[1], Cmae.randomdf.list.model.est.ztest.Donor[1],
Hdiv.randomdf.list.model.est.ztest.pH[1], Hdiv.randomdf.list.model.est.ztest.SM[1], Hdiv.randomdf.list.model.est.ztest.Donor[1]),

  "P"=c(Dpug.randomdf.list.model.est.ztest.pH[2], Dpug.randomdf.list.model.est.ztest.SM[2], Dpug.randomdf.list.model.est.ztest.Donor[2],

Cmae.randomdf.list.model.est.ztest.pH[2], Cmae.randomdf.list.model.est.ztest.SM[2], Cmae.randomdf.list.model.est.ztest.Donor[2],
Hdiv.randomdf.list.model.est.ztest.pH[2], Hdiv.randomdf.list.model.est.ztest.SM[2], Hdiv.randomdf.list.model.est.ztest.Donor[2])


) %>%

  kbl(digits = 4,caption = "Summary of z-test for bootstrap estimate versus real estimate in Coxph models") %>%

  kable_classic(full_width = F, html_font = "Cambria")


```

```{r plots bootstrap coxph, echo=TRUE}

# Plots

# Dpug

# pH drop


Dpug.randomdf.list.model.est.dens.pH <- data.frame("estimate"=density(Dpug.randomdf.list.model.est$pH_drop1)$x,
"prob"=density(Dpug.randomdf.list.model.est$pH_drop1)$y)

density(Dpug.randomdf.list.model.est$pH_drop1)$x[which.max(density(Dpug.randomdf.list.model.est$pH_drop1)$y)] # get the bootstrap estimate
```

```
Dpug.Time.Coxph.pH.SE1 <-
as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,1]-as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,3]

Dpug.Time.Coxph.pH.SE2 <-
as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,1]+as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,3]

Dpug.randomdf.list.model.est.pH.plot <- ggplot(data= Dpug.randomdf.list.model.est, aes(x=pH_drop1))  + own_theme_hist() +
 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+
 geom_density(size=1)+  geom_area(data = subset(Dpug.randomdf.list.model.est.dens.pH, estimate > Dpug.Time.Coxph.pH.SE1 & estimate < Dpug.Time.Coxph.pH.SE2 ), aes(y=prob, x=estimate),
                  fill = "skyblue1",
                 color = "black", alpha=0.6) +
 geom_vline(aes(xintercept=as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,1]),
        color="blue", linetype=1, size=1.5)+
  geom_vline(aes(xintercept=density(pH_drop1)$x[which.max(density(pH_drop1)$y)]),
        color="black", linetype=4, size=1.5) +
  xlab("Boostrapped pH drop Estimate") + ylab("Density") + annotate("text",  x=min(Dpug.randomdf.list.model.est.dens.pH$estimate), y = Inf, label =
                                                paste0("Mod. = ", round(as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,1],2), " ± ", round(as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[1,3], 2),
                                                       "\nBoot. = ", round(mean(Dpug.randomdf.list.model.est$pH_drop1),2), " ± ", round(sd(Dpug.randomdf.list.model.est$pH_drop1),2), "\nP = 0.8844"), hjust=0, vjust=1, size=5)

# SM
Dpug.randomdf.list.model.est.dens.SM <- data.frame("estimate"=density(Dpug.randomdf.list.model.est$SM1)$x, "prob"=density(Dpug.randomdf.list.model.est$SM1)$y)
density(Dpug.randomdf.list.model.est$SM1)$x[which.max(density(Dpug.randomdf.list.model.est$SM1)$y)] # get the bootstrap estimate

Dpug.Time.Coxph.SM.SE1 <-
as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,1]-as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,3]

Dpug.Time.Coxph.SM.SE2 <-
as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,1]+as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,3]

Dpug.randomdf.list.model.est.SM.plot <- ggplot(data= Dpug.randomdf.list.model.est, aes(x=SM1))  + own_theme_hist() +
 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+
 geom_density(size=1)+  geom_area(data = subset(Dpug.randomdf.list.model.est.dens.SM, estimate > Dpug.Time.Coxph.SM.SE1 & estimate  < Dpug.Time.Coxph.SM.SE2 ), aes(y=prob, x=estimate),
                  fill = "skyblue1",
                 color = "black", alpha=0.6) +
 geom_vline(aes(xintercept=as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,1]),
        color="blue", linetype=1, size=1.5)+
  geom_vline(aes(xintercept=density(SM1)$x[which.max(density(SM1)$y)]),
```

```
        color="black", linetype=4, size=1.5)  +

  xlab("Boostrapped SM Estimate") + ylab("Density")+ annotate("text",  x=min(Dpug.randomdf.list.model.est.dens.SM$estimate), y = Inf, label =

                                                paste0("Mod. = ", round(as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,1],2), " ± ", round(as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[2,3], 2),

                                                       "\nBoot. = ", round(mean(Dpug.randomdf.list.model.est$SM1),2), " ± ", round(sd(Dpug.randomdf.list.model.est$SM1),2), "\nP = 0.1623"), hjust=0, vjust=1, size=5)



# Donor

Dpug.randomdf.list.model.est.dens.Donor <- data.frame("estimate"=density(Dpug.randomdf.list.model.est$DonorSaur)$x, "prob"=density(Dpug.randomdf.list.model.est$DonorSaur)$y)

density(Dpug.randomdf.list.model.est$DonorSaur)$x[which.max(density(Dpug.randomdf.list.model.est$DonorSaur)$y)] # get the bootstrap estimate



Dpug.Time.Coxph.Donor.SE1 <-

as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,1]-as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,3]



Dpug.Time.Coxph.Donor.SE2 <-
as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,1]+as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,3]



Dpug.randomdf.list.model.est.Donor.plot <- ggplot(data= Dpug.randomdf.list.model.est, aes(x=DonorSaur))  + own_theme_hist() +

 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+

 geom_density(size=1)+  geom_area(data = subset(Dpug.randomdf.list.model.est.dens.Donor, estimate > Dpug.Time.Coxph.Donor.SE1 & estimate  < Dpug.Time.Coxph.Donor.SE2 ), aes(y=prob, x=estimate),

                  fill = "skyblue1",

                 color = "black", alpha=0.6) +

 geom_vline(aes(xintercept=as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,1]),

        color="blue", linetype=1, size=1.5)+

  geom_vline(aes(xintercept=density(DonorSaur)$x[which.max(density(DonorSaur)$y)]),

        color="black", linetype=4, size=1.5)  +

  xlab("Boostrapped Donor Estimate") + ylab("Density")+ annotate("text",  x=min(Dpug.randomdf.list.model.est.dens.Donor$estimate), y = Inf, label =

                                                paste0("Mod. = ", round(as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,1],2), " ± ", round(as.data.frame(summary(Dpug.Time.Coxph)$coefficients)[3,3], 2),

                                                       "\nBoot. = ", round(mean(Dpug.randomdf.list.model.est$DonorSaur),2), " ± ", round(sd(Dpug.randomdf.list.model.est$DonorSaur),2), "\nP = 0.0762"), hjust=0, vjust=1, size=5)



# Cmae

# pH drop



Cmae.randomdf.list.model.est.dens.pH <- data.frame("estimate"=density(Cmae.randomdf.list.model.est$pH_drop1)$x, "prob"=density(Cmae.randomdf.list.model.est$pH_drop1)$y)

density(Cmae.randomdf.list.model.est$pH_drop1)$x[which.max(density(Cmae.randomdf.list.model.est$pH_drop1)$y)] # get the bootstrap estimate
```

```
Cmae.Time.Coxph.pH.SE1 <-

as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,1]-as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,3]


Cmae.Time.Coxph.pH.SE2 <-
as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,1]+as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,3]


Cmae.randomdf.list.model.est.pH.plot <- ggplot(data= Cmae.randomdf.list.model.est, aes(x=pH_drop1))  + own_theme_hist() +

 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+

 geom_density(size=1)+  geom_area(data = subset(Cmae.randomdf.list.model.est.dens.pH, estimate > Cmae.Time.Coxph.pH.SE1 &
estimate  < Cmae.Time.Coxph.pH.SE2 ), aes(y=prob, x=estimate),

                fill = "skyblue1",

               color = "black", alpha=0.6) +

 geom_vline(aes(xintercept=as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,1]),

        color="blue", linetype=1, size=1.5)+

  geom_vline(aes(xintercept=density(pH_drop1)$x[which.max(density(pH_drop1)$y)]),

        color="black", linetype=4, size=1.5)  +

  xlab("Boostrapped pH drop Estimate") + ylab("Density")+ annotate("text",  x=min(Cmae.randomdf.list.model.est.dens.pH$estimate), y =
Inf, label =

                                        paste0("Mod. = ", round(as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,1],2), " ± ",
round(as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[1,3], 2),

                                            "\nBoot. = ", round(mean(Cmae.randomdf.list.model.est$pH_drop1),2), " ± ",
round(sd(Cmae.randomdf.list.model.est$pH_drop1),2), "\nP = 0.3594"), hjust=0, vjust=1, size=5)


# SM

Cmae.randomdf.list.model.est.dens.SM <- data.frame("estimate"=density(Cmae.randomdf.list.model.est$SM1)$x,
"prob"=density(Cmae.randomdf.list.model.est$SM1)$y)

density(Cmae.randomdf.list.model.est$SM1)$x[which.max(density(Cmae.randomdf.list.model.est$SM1)$y)] # get the bootstrap estimate


Cmae.Time.Coxph.SM.SE1 <-

as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,1]-as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,3]


Cmae.Time.Coxph.SM.SE2 <-
as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,1]+as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,3]


Cmae.randomdf.list.model.est.SM.plot <- ggplot(data= Cmae.randomdf.list.model.est, aes(x=SM1))  + own_theme_hist() +

 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+

 geom_density(size=1)+  geom_area(data = subset(Cmae.randomdf.list.model.est.dens.SM, estimate > Cmae.Time.Coxph.SM.SE1 &
estimate  < Cmae.Time.Coxph.SM.SE2 ), aes(y=prob, x=estimate),

                fill = "skyblue1",

               color = "black", alpha=0.6) +

 geom_vline(aes(xintercept=as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,1]),

        color="blue", linetype=1, size=1.5)+

  geom_vline(aes(xintercept=density(SM1)$x[which.max(density(SM1)$y)]),
```

```
        color="black", linetype=4, size=1.5)+

  xlab("Boostrapped SM Estimate") + ylab("Density")+ annotate("text",  x=min(Cmae.randomdf.list.model.est.dens.SM$estimate), y = Inf, label =

                                                paste0("Mod. = ", round(as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,1],2), " ± ",
round(as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[2,3], 2),

                                                    "\nBoot. = ", round(mean(Cmae.randomdf.list.model.est$SM1),2), " ± ",
round(sd(Cmae.randomdf.list.model.est$SM1),2), "\nP = 0.7966"), hjust=0, vjust=1, size=5)



# Donor

Cmae.randomdf.list.model.est.dens.Donor <- data.frame("estimate"=density(Cmae.randomdf.list.model.est$DonorSaur)$x,
"prob"=density(Cmae.randomdf.list.model.est$DonorSaur)$y)

density(Cmae.randomdf.list.model.est$DonorSaur)$x[which.max(density(Cmae.randomdf.list.model.est$DonorSaur)$y)] # get the
bootstrap estimate



Cmae.Time.Coxph.Donor.SE1 <-

as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,1]-as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,3]



Cmae.Time.Coxph.Donor.SE2 <-
as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,1]+as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,3]



Cmae.randomdf.list.model.est.Donor.plot <- ggplot(data= Cmae.randomdf.list.model.est, aes(x=DonorSaur))  + own_theme_hist() +

 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+

 geom_density(size=1)+  geom_area(data = subset(Cmae.randomdf.list.model.est.dens.Donor, estimate > Cmae.Time.Coxph.Donor.SE1 &
estimate  < Cmae.Time.Coxph.Donor.SE2 ), aes(y=prob, x=estimate),

                  fill = "skyblue1",

                 color = "black", alpha=0.6) +

 geom_vline(aes(xintercept=as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,1]),

        color="blue", linetype=1, size=1.5)+

  geom_vline(aes(xintercept=density(DonorSaur)$x[which.max(density(DonorSaur)$y)]),

        color="black", linetype=4, size=1.5)  +

  xlab("Boostrapped Donor Estimate") + ylab("Density")+ annotate("text",  x=min(Cmae.randomdf.list.model.est.dens.Donor$estimate), y =
Inf, label =

                                                paste0("Mod. = ", round(as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,1],2), " ± ",
round(as.data.frame(summary(Cmae.Time.Coxph)$coefficients)[3,3], 2),

                                                    "\nBoot. = ", round(mean(Cmae.randomdf.list.model.est$DonorSaur),2), " ± ",
round(sd(Cmae.randomdf.list.model.est$DonorSaur),2), "\nP = 0.9848"), hjust=0, vjust=1, size=5)



# Hdiv

# pH drop



Hdiv.randomdf.list.model.est.dens.pH <- data.frame("estimate"=density(Hdiv.randomdf.list.model.est$pH_drop1)$x,
"prob"=density(Hdiv.randomdf.list.model.est$pH_drop1)$y)

density(Hdiv.randomdf.list.model.est$pH_drop1)$x[which.max(density(Hdiv.randomdf.list.model.est$pH_drop1)$y)] # get the bootstrap
estimate
```

```
Hdiv.Time.Coxph.pH.SE1 <-
as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,1]-as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,3]

Hdiv.Time.Coxph.pH.SE2 <-
as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,1]+as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,3]

Hdiv.randomdf.list.model.est.pH.plot <- ggplot(data= Hdiv.randomdf.list.model.est, aes(x=pH_drop1))  + own_theme_hist() +
 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+
 geom_density(size=1)+  geom_area(data = subset(Hdiv.randomdf.list.model.est.dens.pH, estimate > Hdiv.Time.Coxph.pH.SE1 & estimate < Hdiv.Time.Coxph.pH.SE2 ), aes(y=prob, x=estimate),
                  fill = "skyblue1",
                 color = "black", alpha=0.6) +
 geom_vline(aes(xintercept=as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,1]),
        color="blue", linetype=1, size=1.5)+
  geom_vline(aes(xintercept=density(pH_drop1)$x[which.max(density(pH_drop1)$y)]),
        color="black", linetype=4, size=1.5)  +
  xlab("Boostrapped pH drop Estimate") + ylab("Density")+ annotate("text",  x=min(Hdiv.randomdf.list.model.est.dens.pH$estimate), y = Inf, label =
                                                paste0("Mod. = ", round(as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,1],2), " ± ", round(as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[1,3], 2),
                                                       "\nBoot. = ", round(mean(Hdiv.randomdf.list.model.est$pH_drop1),2), " ± ", round(sd(Hdiv.randomdf.list.model.est$pH_drop1),2), "\nP = 0.8700"), hjust=0, vjust=1, size=5)

# SM
Hdiv.randomdf.list.model.est.dens.SM <- data.frame("estimate"=density(Hdiv.randomdf.list.model.est$SM1)$x, "prob"=density(Hdiv.randomdf.list.model.est$SM1)$y)
density(Hdiv.randomdf.list.model.est$SM1)$x[which.max(density(Hdiv.randomdf.list.model.est$SM1)$y)] # get the bootstrap estimate

Hdiv.Time.Coxph.SM.SE1 <-
as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,1]-as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,3]

Hdiv.Time.Coxph.SM.SE2 <-
as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,1]+as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,3]

Hdiv.randomdf.list.model.est.SM.plot <- ggplot(data= Hdiv.randomdf.list.model.est, aes(x=SM1))  + own_theme_hist() +
 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+
 geom_density(size=1)+  geom_area(data = subset(Hdiv.randomdf.list.model.est.dens.SM, estimate > Hdiv.Time.Coxph.SM.SE1 & estimate < Hdiv.Time.Coxph.SM.SE2 ), aes(y=prob, x=estimate),
                  fill = "skyblue1",
                 color = "black", alpha=0.6) +
 geom_vline(aes(xintercept=as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,1]),
        color="blue", linetype=1, size=1.5)+
  geom_vline(aes(xintercept=density(SM1)$x[which.max(density(SM1)$y)]),
```

```
        color="black", linetype=4, size=1.5) +

  xlab("Boostrapped SM Estimate") + ylab("Density")+ annotate("text",  x=min(Hdiv.randomdf.list.model.est.dens.SM$estimate), y = Inf, label =

                                                paste0("Mod. = ", round(as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,1],2), " ± ", round(as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[2,3], 2),

                                                        "\nBoot. = ", round(mean(Hdiv.randomdf.list.model.est$SM1),2), " ± ", round(sd(Hdiv.randomdf.list.model.est$SM1),2), "\nP = 0.7150"), hjust=0, vjust=1, size=5)



# Donor

Hdiv.randomdf.list.model.est.dens.Donor <- data.frame("estimate"=density(Hdiv.randomdf.list.model.est$DonorSaur)$x, "prob"=density(Hdiv.randomdf.list.model.est$DonorSaur)$y)

density(Hdiv.randomdf.list.model.est$DonorSaur)$x[which.max(density(Hdiv.randomdf.list.model.est$DonorSaur)$y)] # get the bootstrap estimate



Hdiv.Time.Coxph.Donor.SE1 <-

as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,1]-as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,3]



Hdiv.Time.Coxph.Donor.SE2 <-
as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,1]+as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,3]



Hdiv.randomdf.list.model.est.Donor.plot <- ggplot(data= Hdiv.randomdf.list.model.est, aes(x=DonorSaur))  + own_theme_hist() +

 geom_histogram(aes(y=..density..), colour="black", fill="grey80", alpha=0.5 )+

 geom_density(size=1)+  geom_area(data = subset(Hdiv.randomdf.list.model.est.dens.Donor, estimate > Hdiv.Time.Coxph.Donor.SE1 & estimate  < Hdiv.Time.Coxph.Donor.SE2 ), aes(y=prob, x=estimate),

                  fill = "skyblue1",

                 color = "black", alpha=0.6) +

 geom_vline(aes(xintercept=as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,1]),

        color="blue", linetype=1, size=1.5)+

  geom_vline(aes(xintercept=density(DonorSaur)$x[which.max(density(DonorSaur)$y)]),

        color="black", linetype=4, size=1.5)  +

  xlab("Boostrapped Donor Estimate") + ylab("Density")+ annotate("text",  x=min(Hdiv.randomdf.list.model.est.dens.Donor$estimate), y = Inf, label =

                                                paste0("Mod. = ", round(as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,1],2), " ± ", round(as.data.frame(summary(Hdiv.Time.Coxph)$coefficients)[3,3], 2),

                                                        "\nBoot. = ", round(mean(Hdiv.randomdf.list.model.est$DonorSaur),2), " ± ", round(sd(Hdiv.randomdf.list.model.est$SM1),2), "\nP = 0.9816"), hjust=0, vjust=1, size=5)



library(cowplot)

plot_grid(Dpug.randomdf.list.model.est.pH.plot,

        Dpug.randomdf.list.model.est.SM.plot,

        Dpug.randomdf.list.model.est.Donor.plot,


        Cmae.randomdf.list.model.est.pH.plot,
```

```
Cmae.randomdf.list.model.est.SM.plot,

Cmae.randomdf.list.model.est.Donor.plot,


Hdiv.randomdf.list.model.est.pH.plot,

Hdiv.randomdf.list.model.est.SM.plot,

Hdiv.randomdf.list.model.est.Donor.plot,

nrow=3, ncol=3)
```

```

```{r check shade plot area, eval=FALSE, include=FALSE}

# Note that std error area is precise at 0.001 at least (since the area is drawn under the density curve with x values that are the closest to the limits of the SE of the real estimate)

# Dpug

min(Dpug.randomdf.list.model.est.dens.pH$estimate[Dpug.randomdf.list.model.est.dens.pH$estimate > Dpug.Time.Coxph.pH.SE1]); Dpug.Time.Coxph.pH.SE1

max(Dpug.randomdf.list.model.est.dens.pH$estimate[Dpug.randomdf.list.model.est.dens.pH$estimate < Dpug.Time.Coxph.pH.SE2]); Dpug.Time.Coxph.pH.SE2


min(Dpug.randomdf.list.model.est.dens.SM$estimate[Dpug.randomdf.list.model.est.dens.SM$estimate > Dpug.Time.Coxph.SM.SE1]); Dpug.Time.Coxph.SM.SE1

max(Dpug.randomdf.list.model.est.dens.SM$estimate[Dpug.randomdf.list.model.est.dens.SM$estimate < Dpug.Time.Coxph.SM.SE2]); Dpug.Time.Coxph.SM.SE2


min(Dpug.randomdf.list.model.est.dens.Donor$estimate[Dpug.randomdf.list.model.est.dens.Donor$estimate > Dpug.Time.Coxph.Donor.SE1]); Dpug.Time.Coxph.Donor.SE1

max(Dpug.randomdf.list.model.est.dens.Donor$estimate[Dpug.randomdf.list.model.est.dens.Donor$estimate < Dpug.Time.Coxph.Donor.SE2]); Dpug.Time.Coxph.Donor.SE2


# Cmae

min(Cmae.randomdf.list.model.est.dens.pH$estimate[Cmae.randomdf.list.model.est.dens.pH$estimate > Cmae.Time.Coxph.pH.SE1]); Cmae.Time.Coxph.pH.SE1

max(Cmae.randomdf.list.model.est.dens.pH$estimate[Cmae.randomdf.list.model.est.dens.pH$estimate < Cmae.Time.Coxph.pH.SE2]); Cmae.Time.Coxph.pH.SE2


min(Cmae.randomdf.list.model.est.dens.SM$estimate[Cmae.randomdf.list.model.est.dens.SM$estimate > Cmae.Time.Coxph.SM.SE1]); Cmae.Time.Coxph.SM.SE1

max(Cmae.randomdf.list.model.est.dens.SM$estimate[Cmae.randomdf.list.model.est.dens.SM$estimate < Cmae.Time.Coxph.SM.SE2]); Cmae.Time.Coxph.SM.SE2


min(Cmae.randomdf.list.model.est.dens.Donor$estimate[Cmae.randomdf.list.model.est.dens.Donor$estimate > Cmae.Time.Coxph.Donor.SE1]); Cmae.Time.Coxph.Donor.SE1

max(Cmae.randomdf.list.model.est.dens.Donor$estimate[Cmae.randomdf.list.model.est.dens.Donor$estimate < Cmae.Time.Coxph.Donor.SE2]); Cmae.Time.Coxph.Donor.SE2
```
```

Hdiv

min(Hdiv.randomdf.list.model.est.dens.pH$estimate[Hdiv.randomdf.list.model.est.dens.pH$estimate > Hdiv.Time.Coxph.pH.SE1]); Hdiv.Time.Coxph.pH.SE1

max(Hdiv.randomdf.list.model.est.dens.pH$estimate[Hdiv.randomdf.list.model.est.dens.pH$estimate < Hdiv.Time.Coxph.pH.SE2]); Hdiv.Time.Coxph.pH.SE2

min(Hdiv.randomdf.list.model.est.dens.SM$estimate[Hdiv.randomdf.list.model.est.dens.SM$estimate > Hdiv.Time.Coxph.SM.SE1]); Hdiv.Time.Coxph.SM.SE1

max(Hdiv.randomdf.list.model.est.dens.SM$estimate[Hdiv.randomdf.list.model.est.dens.SM$estimate < Hdiv.Time.Coxph.SM.SE2]); Hdiv.Time.Coxph.SM.SE2

min(Hdiv.randomdf.list.model.est.dens.Donor$estimate[Hdiv.randomdf.list.model.est.dens.Donor$estimate > Hdiv.Time.Coxph.Donor.SE1]); Hdiv.Time.Coxph.Donor.SE1

max(Hdiv.randomdf.list.model.est.dens.Donor$estimate[Hdiv.randomdf.list.model.est.dens.Donor$estimate < Hdiv.Time.Coxph.Donor.SE2]); Hdiv.Time.Coxph.Donor.SE2

```

# AIC analysis.

```{r}

# Dpug time - sequential addition

Cand.mod.Dpug.Time <- list()

##global model

Cand.mod.Dpug.Time[[1]] <- Dpug.Time.Coxph

Cand.mod.Dpug.Time[[2]] <- Dpug.Time.Coxph.mod2

Cand.mod.Dpug.Time[[3]] <- Dpug.Time.Coxph.mod3

Modnames.Dpug <- c("SM*pH*Donor", "SM*pH*Donor+water",

"SM*pH*Donor+water+size")

AICcmodavg::aictab(cand.set = Cand.mod.Dpug.Time, modnames = Modnames.Dpug)

AICcmodavg::bictab(cand.set = Cand.mod.Dpug.Time, modnames = Modnames.Dpug)

anova(Dpug.Time.Coxph.null, Dpug.Time.Coxph, Dpug.Time.Coxph.mod2, Dpug.Time.Coxph.mod3, test="chisq") # Water use and crab size can be disregarded

# Overall this shows that there is not strong enough evidence to include the covariates according the chi-square test and the delta AIC (<2) and delta BIC

anova(Dpug.Time.Coxph, Dpug.Time.Coxph.mod2, Dpug.Time.Coxph.mod3, test="chisq") # Water use and crab size can be disregarded

# Cmae

Cand.mod.Cmae.Time <- list()

```
##global model

Cand.mod.Cmae.Time[[1]] <- Cmae.Time.Coxph

Cand.mod.Cmae.Time[[2]] <- Cmae.Time.Coxph.mod2

Cand.mod.Cmae.Time[[3]] <- Cmae.Time.Coxph.mod3

Modnames.Cmae <- c("SM*pH*Donor", "SM*pH*Donor+water",

        "SM*pH*Donor+Year")

AICcmodavg::aictab(cand.set = Cand.mod.Cmae.Time, modnames = Modnames.Cmae)

AICcmodavg::bictab(cand.set = Cand.mod.Cmae.Time, modnames = Modnames.Cmae)

anova(Cmae.Time.Coxph.null, Cmae.Time.Coxph, Cmae.Time.Coxph.mod2, Cmae.Time.Coxph.mod3, test="chisq")

Overall this shows that there is not strong enough evidence to include the covariates according the chi-square test and the delta AIC and delta BIC

anova(Cmae.Time.Coxph, Cmae.Time.Coxph.mod2, Cmae.Time.Coxph.mod3, test="chisq")

Cand.mod.Cmae.Time2 <- list()

##global model

Cand.mod.Cmae.Time2[[1]] <- Cmae.Time.Coxph.2019

Cand.mod.Cmae.Time2[[2]] <- Cmae.Time.Coxph.2019.size

Modnames.Cmae2 <- c("SM*pH*Donor", "SM*pH*Donor+size")

AICcmodavg::aictab(cand.set = Cand.mod.Cmae.Time2, modnames = Modnames.Cmae2)

AICcmodavg::bictab(cand.set = Cand.mod.Cmae.Time2, modnames = Modnames.Cmae2)

anova(Cmae.Time.Coxph.2019.null, Cmae.Time.Coxph.2019, Cmae.Time.Coxph.2019.size, test="chisq") # Crab size can be disregarded

Overall this shows that there is not strong enough evidence to include the covariates according the chi-square test and the delta AIC and delta BIC

Hdiv time

Cand.mod.Hdiv.Time <- list()

##global model

Cand.mod.Hdiv.Time[[1]] <- Hdiv.Time.Coxph

Cand.mod.Hdiv.Time[[2]] <- Hdiv.Time.Coxph.mod2

Modnames.Hdiv <- c("SM*pH*Donor", "SM*pH*Donor+water")

AICcmodavg::aictab(cand.set = Cand.mod.Hdiv.Time, modnames = Modnames.Hdiv)

AICcmodavg::bictab(cand.set = Cand.mod.Hdiv.Time, modnames = Modnames.Hdiv)

anova(Hdiv.Time.Coxph.null, Hdiv.Time.Coxph, Hdiv.Time.Coxph.mod2, test="chisq")

Overall this shows that there is not strong enough evidence to include the covariates according the chi-square test and the delta AIC and delta BIC

anova(Hdiv.Time.Coxph, Hdiv.Time.Coxph.mod2, test="chisq")
```

```
# Dpug Avoidance
Cand.mod.Dpug.Avoid <- list()
##global model
Cand.mod.Dpug.Avoid[[1]] <- glz.Dpug.avoid
Cand.mod.Dpug.Avoid[[2]] <- glz.Dpug.avoid.mod2
Cand.mod.Dpug.Avoid[[3]] <- glz.Dpug.avoid.mod3

Modnames.Dpug.Avoid <- c("SM*pH*Donor", "SM*pH*Donor+water",
        "SM*pH*Donor+size")

AICcmodavg::aictab(cand.set = Cand.mod.Dpug.Avoid, modnames = Modnames.Dpug.Avoid)
AICcmodavg::bictab(cand.set = Cand.mod.Dpug.Avoid, modnames = Modnames.Dpug.Avoid)
anova(glz.Dpug.avoid.null, glz.Dpug.avoid, glz.Dpug.avoid.mod2,glz.Dpug.avoid.mod3, test="Chisq")
# Overall this shows that there is not strong enough evidence to include the covariates according the chi-square test and the delta AIC and delta BIC

# Cmae Avoidance
Cand.mod.Cmae.Avoid <- list()
##global model
Cand.mod.Cmae.Avoid[[1]] <- glz.Cmae.avoid
Cand.mod.Cmae.Avoid[[2]] <- glz.Cmae.avoid.mod2
Cand.mod.Cmae.Avoid[[3]] <- glz.Cmae.avoid.mod3

Modnames.Cmae.Avoid <- c("SM*pH*Donor", "SM*pH*Donor+water",
        "SM*pH*Donor+size")


AICcmodavg::aictab(cand.set = Cand.mod.Cmae.Avoid, modnames = Modnames.Cmae.Avoid)
AICcmodavg::bictab(cand.set = Cand.mod.Cmae.Avoid, modnames = Modnames.Cmae.Avoid)
anova(glz.Cmae.avoid.null, glz.Cmae.avoid, glz.Cmae.avoid.mod2, glz.Cmae.avoid.mod3, test="Chisq")
# Overall this shows that there is evidence to include water use as a covariate according the chi-square test and the delta AIC and delta BIC

# Hdiv Avoidance
Cand.mod.Hdiv.Avoid <- list()
##global model
Cand.mod.Hdiv.Avoid[[1]] <- glz.Hdiv.avoid.donor
Cand.mod.Hdiv.Avoid[[2]] <- glz.Hdiv.avoid.donor.mod2

Modnames.Hdiv.Avoid <- c("Donor", "Donor+water" )
```

```
AICcmodavg::aictab(cand.set = Cand.mod.Hdiv.Avoid, modnames = Modnames.Hdiv.Avoid)

AICcmodavg::bictab(cand.set = Cand.mod.Hdiv.Avoid, modnames = Modnames.Hdiv.Avoid)

anova(glz.Hdiv.avoid.donor.null, glz.Hdiv.avoid.donor, glz.Hdiv.avoid.donor.mod2, test="Chisq")


# Hdiv/Hdiv Avoidance

Cand.mod.Hdiv.Hdiv.Avoid <- list()

##global model

Cand.mod.Hdiv.Hdiv.Avoid[[1]] <- glz.Hdiv.Hdiv.avoid

Cand.mod.Hdiv.Hdiv.Avoid[[2]] <- glz.Hdiv.Hdiv.avoid.mod2


Modnames.Hdiv.Hdiv.Avoid <- c("Treatment", "Treatment+water" )


AICcmodavg::aictab(cand.set = Cand.mod.Hdiv.Hdiv.Avoid, modnames = Modnames.Hdiv.Hdiv.Avoid)

AICcmodavg::bictab(cand.set = Cand.mod.Hdiv.Hdiv.Avoid, modnames = Modnames.Hdiv.Hdiv.Avoid)

anova(glz.Hdiv.Hdiv.avoid.null, glz.Hdiv.Hdiv.avoid, glz.Hdiv.Hdiv.avoid.mod2, test="Chisq") # The water use can be disregarded
```