Chemo.05.ResistanceIndex

Angel Rain-Franco

12/1/2021

Contents

#Load packages

```
rm(list = ls())
library(readxl)  # read excel files
library(dplyr)
library(ggplot2)  #Plots
library(tidyverse)
library(rstatix)  # HOV test
library(cowplot)  #Arrange plots
library(olsrr)  #Normality test (Kolmogorov-Smirnov)
library(kableExtra)  #Export regular tables
library(nlme)  #Mixed linear models
library(lmerTest)  #P values mixed linear models
# Color palette
cbbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442", "#0072B2",
    "#D55E00", "#CC79A7")
```

1 Load complementary data

1.1 Abundance and salinity

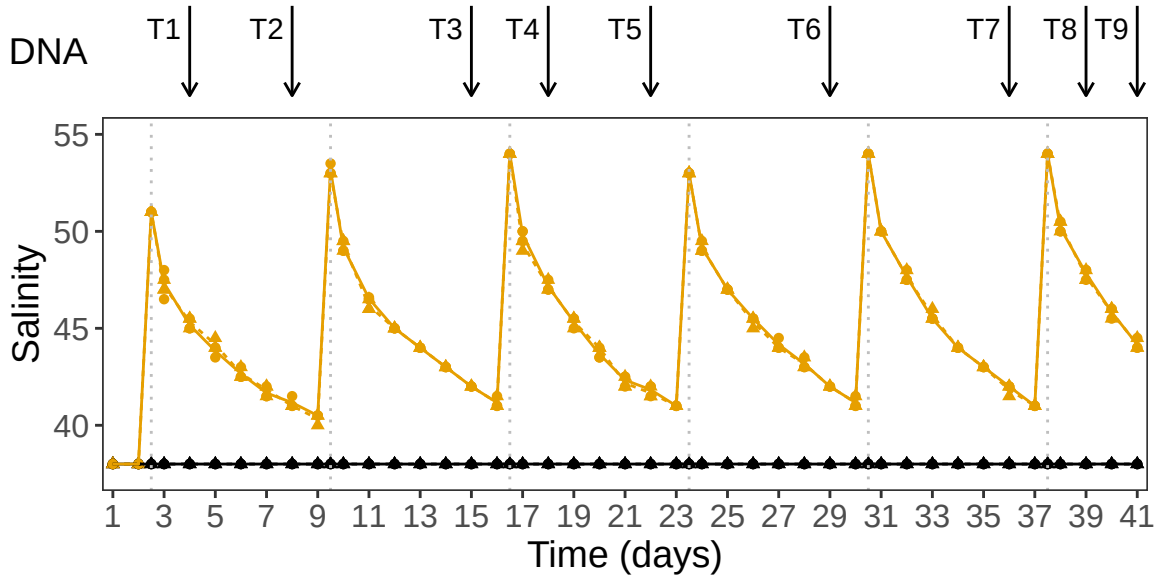
```
Dat = data.frame(read_xlsx("../data/comm.rates/Supplementary_Tables.xlsx", sheet = "Table S8"))
Dat = Dat[Dat$Time < 42, ]  #Only keep the samples until one day 41 for figure
DatS = Dat  #Dataframe only for Salinity figure
Dat = Dat[complete.cases(Dat), ]  #Removing unused columns
Dat$DOM = as.factor(Dat$DOM)
levels(Dat$DOM) = c("eDOM", "oDOM")
Dat$id = paste0(Dat$Time, ".", Dat$DOM, ".", Dat$Treatment, ".", Dat$Replicate)
Dat$id = as.factor(Dat$id)

# Creating the salinity changes and DNA frequency sampling for Figure1 ms
# chemostats DNA sampling frequency plot
pDNA = ggplot() + geom_segment(aes(x = c(4, 8, 15, 18, 22, 29, 36, 39, 41),
    y = rep(2, 9), xend = c(4, 8, 15, 18, 22, 29, 36, 39, 41), yend = rep(0,
        9)), arrow = arrow(length = unit(0.2, "cm")), size = 0.5) + labs(y = "DNA",
    x = "") + annotate("text", x = c(4, 8, 15, 18, 22, 29, 36, 39, 41) - 1,
    y = rep(1.5, 9), label = c("T1", "T2", "T3", "T4", "T5", "T6", "T7", "T8",
        "T9"), size = 4) + scale_x_continuous(breaks = c(1:41), expand = c(0.01,
    0.01), limits = c(1, 41)) + theme_bw() + theme(panel.grid.minor = element_blank(),
    panel.grid.major = element_blank(), axis.text.x = element_blank(), axis.text.y = element_blank()) +
    theme(text = element_text(size = 14, family = "ArialMT")) + labs(tag = "") +
    theme(axis.ticks.y = element_blank(), axis.ticks.x = element_blank(), panel.border = element_blank()
    theme(plot.margin = margin(t = -15, r = 5, b = -10, l = 0, unit = "pt")) +
    theme(axis.title.y = element_text(angle = 0, vjust = 0.5))
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
```

```
# Salinity time series figure (control vs disturbance)

Salinity_plot = DatS %>%
    ggplot() + geom_point(aes(Time, Salinity, colour = Treatment, shape = DOM)) +
    stat_summary(aes(Time, Salinity, colour = Treatment, linetype = DOM), fun = mean,
        geom = "line") + geom_vline(xintercept = c(2.5, 9.5, 16.5, 23.5, 30.5,
    37.5), linetype = "dotted", colour = "grey") + scale_color_manual(values = cbbPalette,
    name = "") + xlab("Time (days)") + ylab("Salinity") + ylim(37.5, 55) + labs(tag = "") +
    scale_x_continuous(breaks = seq(1, 41, 2), expand = c(0.01, 0.01)) + theme_bw() +
    theme(panel.grid.minor = element_blank(), panel.grid.major = element_blank(),
        legend.position = "none", axis.text.x = element_text(size = 12), axis.text.y = element_text(siz
        axis.title.y = element_text(size = 14), axis.title.x = element_text(size = 14)) +
    theme(text = element_text(family = "ArialMT")) + theme(plot.margin = margin(t = -15,
    r = 5, b = 0, l = 0, unit = "pt")) + theme(plot.tag = element_text(size = 14,
    face = "bold", vjust = -4))

pdf("../figures/Figure_Salinity.pdf", width = 7, height = 2)
plot_grid(pDNA, Salinity_plot, ncol = 1, axis = "l", rel_heights = c(0.15, 0.5),
    hjust = -4, align = "v")
dev.off()
```

```
## pdf
##   2
```

```
plot_grid(pDNA, Salinity_plot, ncol = 1, axis = "l", rel_heights = c(0.15, 0.55),
    hjust = -4, align = "v")
```



1.2 Bacterial abundance time series

```
# Bacterial abundance time series
Abundance_plot = Dat %>%
    ggplot(aes(x = Time, y = Bacteria/1e+06, group = interaction(Treatment,
        DOM))) + geom_point(aes(colour = Treatment, shape = DOM), size = 1) +
    scale_colour_manual(values = cbbPalette, name = "Disturbance") + scale_shape_manual(values = c(21,
    4), name = "DOM level", labels = c("eDOM", "oDOM")) + labs(tag = "") + scale_x_continuous(breaks = s
    41, 2), expand = c(0.01, 0.01)) + ylim(0.1, 4) + xlab("Time (days)") + ylab(expression("BA (x10"^6 =
    "cell mL"^-1 * ")")) + geom_smooth(aes(colour = Treatment), method = "loess",
    se = F, span = 0.3) + geom_vline(xintercept = c(2.5, 9.5, 16.5, 23.5, 30.5,
    37.5), linetype = "dotted", colour = "grey") + theme_bw() + theme(panel.grid.minor = element_blank()
    panel.grid.major = element_blank(), legend.position = c(0.5, 0.88), legend.box = "horizontal",
    legend.box.margin = margin(), legend.key.size = unit(0.1, "cm"), legend.box.spacing = unit(0,
        "cm"), legend.spacing.x = unit(0.3, "cm"), legend.spacing.y = unit(0,
        "cm"), axis.title.x = element_text(size = 14), axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 10)) + theme(text = element_text(family = "ArialMT")) +
    theme(plot.margin = margin(t = -6, r = 5, b = 2, l = 0, unit = "pt"), legend.text = element_text(si
```

2 Loading datasets Community functioning

Dataset includes microbial community respiration and bacterial production

Week	DOM	Treatment	Rep	Comment	Value	Units	Variable
1	eDOM	C	1	after	0.0892454	Percentage	BGE
2	eDOM	C	1	after	0.0081200	Percentage	BGE
3	eDOM	C	1	after	0.1400752	Percentage	BGE
4	eDOM	C	1	after	0.0495788	Percentage	BGE
5	eDOM	C	1	after	0.0423676	Percentage	BGE
6	eDOM	C	1	after	0.0205098	Percentage	BGE

```
df.comm.funct <- data.frame(read_xlsx("../data/comm.rates/Supplementary_Tables.xlsx",
    sheet = "Table S9"))
tibble(df.comm.funct)
```

```
## # A tibble: 288 x 8
##     Week DOM   Treatment   Rep Comment  Value Variable              Units
##    <dbl> <chr> <chr>     <dbl> <chr>    <dbl> <chr>                 <chr>
##  1     1 eDOM  C             1 after   0.0195 Community.Respiration umol O2 L-1~
##  2     2 eDOM  C             1 after   0.178  Community.Respiration umol O2 L-1~
##  3     3 eDOM  C             1 after   0.0344 Community.Respiration umol O2 L-1~
##  4     4 eDOM  C             1 after   0.131  Community.Respiration umol O2 L-1~
##  5     5 eDOM  C             1 after   0.213  Community.Respiration umol O2 L-1~
##  6     6 eDOM  C             1 after   0.285  Community.Respiration umol O2 L-1~
##  7     1 oDOM  C             1 after   2.48   Community.Respiration umol O2 L-1~
##  8     2 oDOM  C             1 after   0.360  Community.Respiration umol O2 L-1~
##  9     3 oDOM  C             1 after   0.105  Community.Respiration umol O2 L-1~
## 10     4 oDOM  C             1 after   0.372  Community.Respiration umol O2 L-1~
## # ... with 278 more rows
```

2.0.1 Bacterial growth efficiency (BGE) calculation

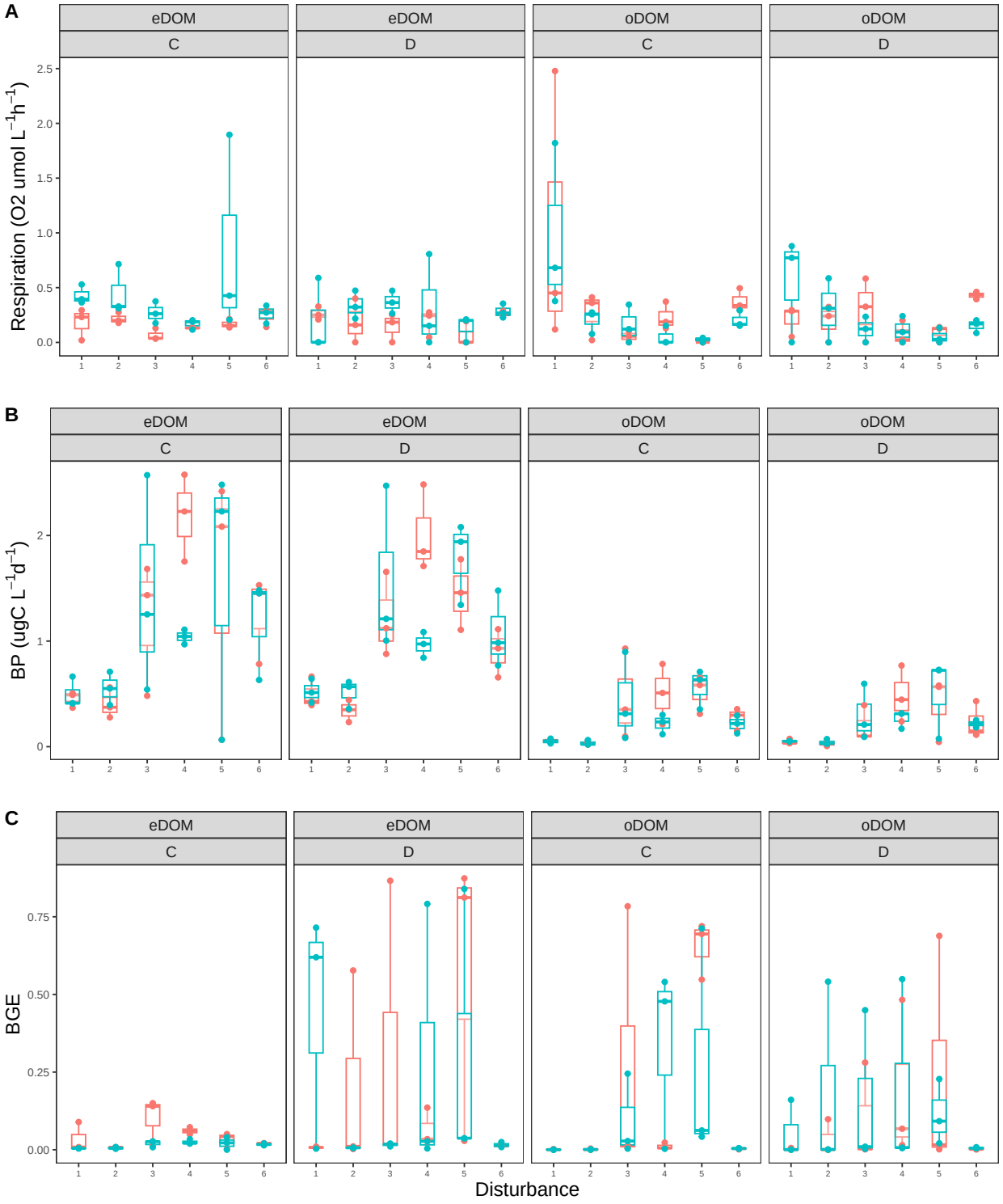
```
# RES2= Bulk respiration (uM_h_corr or umol L-1 h-1) need to be scaled to
# daily rate RES3= Bulk production (ugC L-1 d-1) need to be transformed to
# molar by the molecular weigth of Carbon

Ratio = 0.89  #Coefficient to transform O2 consumption to CO2 (Williams and del Giorgio, 2005)
BGE = df.comm.funct[1:144, 1:5]  #Get metadata to store BGE
BGE$Value = (df.comm.funct$Value[df.comm.funct$Variable == "Bacterial.production"]/12)/((df.comm.funct$
    "Bacterial.production"]/12) + (df.comm.funct$Value[df.comm.funct$Variable ==
    "Community.Respiration"] * Ratio * 24))

BGE$Units = "Percentage"
BGE$Variable = "BGE"
kbl(head(BGE)) %>%
    kable_styling()
```

```
# Combining datasets
df.comm.funct = rbind(df.comm.funct, BGE)
```

2.1 Functional rates

Figures

```
# setting up graphic setting for the figures
my_theme = theme_bw() + theme(text = element_text(size = 14, family = "ArialMT")) +
    theme(legend.position = "none", panel.grid.minor = element_blank(), panel.grid.major = element_blank
        axis.text.x = element_text(size = 6), axis.text.y = element_text(size = 8))
```



3 Resistance index calculation

3.1 Resistance index definition

Here we defined the resistance index as the difference between the natural log of the fold change of the disturbance treatment against the control

```
# Option 1
resistance1 <- function(control, disturbance) {
    abs(log(control) - log(disturbance))
}
```

3.2 Estimation of the resistance index

Here we defined calculation step by step for the resistance index for all the functional parameters.

```
# Pooling data into a list to perform all the calculation at one by
# applying a loop for each element (functional measurement) of the list.

# Set up levels as factors
df.comm.funct$Variable = factor(df.comm.funct$Variable)
df.comm.funct$Variable = factor(df.comm.funct$Variable, c("Bacterial.production",
    "Community.Respiration", "BGE"))
# Empty list for storing results
List.ratio = list()
for (i in 1:3) {
    # Splitting the data between before and after the salt pulse
    # disturbance. Here the function aggregate allow us to retrieve the
    # data sorted always in the same way.
    index.level = levels(df.comm.funct$Variable)[i]

    # Before disturbance
    T_bef = aggregate(Value ~ Week + Treatment + DOM + Rep + Comment, data = df.comm.funct[df.comm.funct
        index.level & df.comm.funct$Comment == "before", ], FUN = "mean")
    # After disturbance
    T_aft = aggregate(Value ~ Week + Treatment + DOM + Rep + Comment, data = df.comm.funct[df.comm.funct
        index.level & df.comm.funct$Comment == "after", ], FUN = "mean")

    # Pooling data for calculations
    T_aft$Before = T_bef$Value

    # Calculating the response ratio (RR) as F_after/F_before
    T_aft$RR = (T_aft$Value/T_aft$Before)

    # Extracting RR for Control and Disturbed treatments
    dist = T_aft[T_aft$Treatment == "D", ]  #Disturbance treatment
    cont = T_aft[T_aft$Treatment == "C", ]  #Control

    # Calculate the mean of the control RR. To represent the overall
    # variability of the control we used the mean value of the triplicated
    # measurements.
```

```
    T0 <- cont %>%
        group_by(interaction(DOM, Treatment, Week)) %>%
        mutate(mControl_RR = mean(RR))  # Calculate the mean of the controls
    T0 <- data.frame(T0)

    # State column as factors
    T0$Week = as.factor(T0$Week)
    T0$DOM = as.factor(T0$DOM)
    T0$DOM = factor(T0$DOM, c("oDOM", "eDOM"))
    # Calculate the absolute difference between the LRRs
    # (meanControl-Disturbed_replicates)
    T0$res.index = resistance1(T0$mControl_RR, dist$RR) * -1  #option
    T0$Variable = index.level
    List.ratio[[i]] = T0
}
```

3.3 Resistance index plots

3.4 Statistical analysis

```
# Setup elements for loop
List.aov = list()
M.stats = matrix(NA, 3, 8)
MLM_oDOM = list()
MLM_eDOM = list()
# Define variable names
rownames(M.stats) = c("BP", "Respiration", "BGE")
# Define stats names
colnames(M.stats) = c("F", "P-value", "F", "P-value", "Slope", "P-value", "Slope",
    "P-value")

# Loop for functional data
for (i in 1:3) {
    # Normality
    List.ratio[[i]] %>%
        group_by(DOM, Week) %>%
        shapiro_test(res.index)

    # Homogeneity of variances
    List.ratio[[i]] %>%
        group_by(Week) %>%
        levene_test(res.index ~ DOM)

    # ANOVA Repeated measurement ANOVA
    # (https://stats.idre.ucla.edu/r/seminars/repeated-measures-analysis-with-r/)
    # https://m-clark.github.io/docs/mixedModels/anovamixed.html#introduction

    summary(aov(res.index ~ DOM * Week + Error(Rep), data = List.ratio[[i]]))
    tmp = aov(res.index ~ DOM * Week + Error(Rep), data = List.ratio[[i]])

    # Retrieving stats from results
```

Function	RM-ANOVA				MLM			
Variable	DOM		Week		oDOM		eDOM	
	F	P-value	F	P-value	Slope	P-value	Slope	P-value
BP	10.828	0.003	0.564	0.726	0.035	0.515	-0.026	0.061
Respiration	0.413	0.527	2.130	0.098	0.170	0.623	0.629	0.029
BGE	0.924	0.347	1.719	0.170	0.241	0.404	0.655	0.009

```
    M.stats[i, 1] = as.numeric(unlist(summary(tmp))["Error: Within.F value1"])
    M.stats[i, 2] = as.numeric(unlist(summary(tmp))["Error: Within.Pr(>F)1"])
    M.stats[i, 3] = as.numeric(unlist(summary(tmp))["Error: Within.F value2"])
    M.stats[i, 4] = as.numeric(unlist(summary(tmp))["Error: Within.Pr(>F)2"])

    # Mixed linear model for oDOM
    MLM_oDOM[[i]] = lme(res.index ~ as.numeric(Week), random = ~1 | Rep, data = List.ratio[[i]][List.ra
        "oDOM", ])
    # Retrieving stats from results
    M.stats[i, 5] = as.numeric(unlist(summary(MLM_oDOM[[i]]))$`coefficients.fixed.as.numeric(Week)`)  #
    M.stats[i, 6] = as.numeric(unlist(summary(MLM_oDOM[[i]]))$tTable10)  #get P-value

    # Mixed linear model for eDOM
    MLM_eDOM[[i]] = lme(res.index ~ as.numeric(Week), random = ~1 | Rep, data = List.ratio[[i]][List.ra
        "eDOM", ])
    # Retrieving stats from results
    M.stats[i, 7] = as.numeric(unlist(summary(MLM_eDOM[[i]]))$`coefficients.fixed.as.numeric(Week)`)  #
    M.stats[i, 8] = as.numeric(unlist(summary(MLM_eDOM[[i]]))$tTable10)  #get P-value
}
```

Table summary statistical analyses Statistical results for the repeated measurement ANOVA applied to the resistance index. Results from mixed model to screen for time trend are also included in the table.

```
kable(M.stats, digits = 3, booktabs = TRUE, format = "latex") %>%
    kable_classic() %>%
    add_header_above(c(Variable = 1, DOM = 2, Week = 2, oDOM = 2, eDOM = 2)) %>%
    add_header_above(c(Function = 1, `RM-ANOVA` = 4, MLM = 4)) %>%
    kable_styling(latex_options = c("striped", "condensed", "scale_down"), position = "center",
        full_width = FALSE)
```

3.5 Exporting Figure 4 ms chemostats

3.6 Export figure

```
pdf("../figures/Figure4_ResistanceIndex.pdf", width = 7, height = 9)
plot_grid(pBP, pR, pBGE, Abundance_plot, ncol = 1, axis = "l", rel_heights = c(0.4,
    0.4, 0.4, 0.65), hjust = 0, align = "v", labels = c("A", "B", "C", "D"),
    label_size = 18)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
dev.off()
```

```
## pdf
##   2
```

```
plot_grid(pBP, pR, pBGE, Abundance_plot, ncol = 1, axis = "l", rel_heights = c(0.4,
    0.4, 0.4, 0.65), hjust = 0, align = "v", labels = c("A", "B", "C", "D"),
    label_size = 18)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

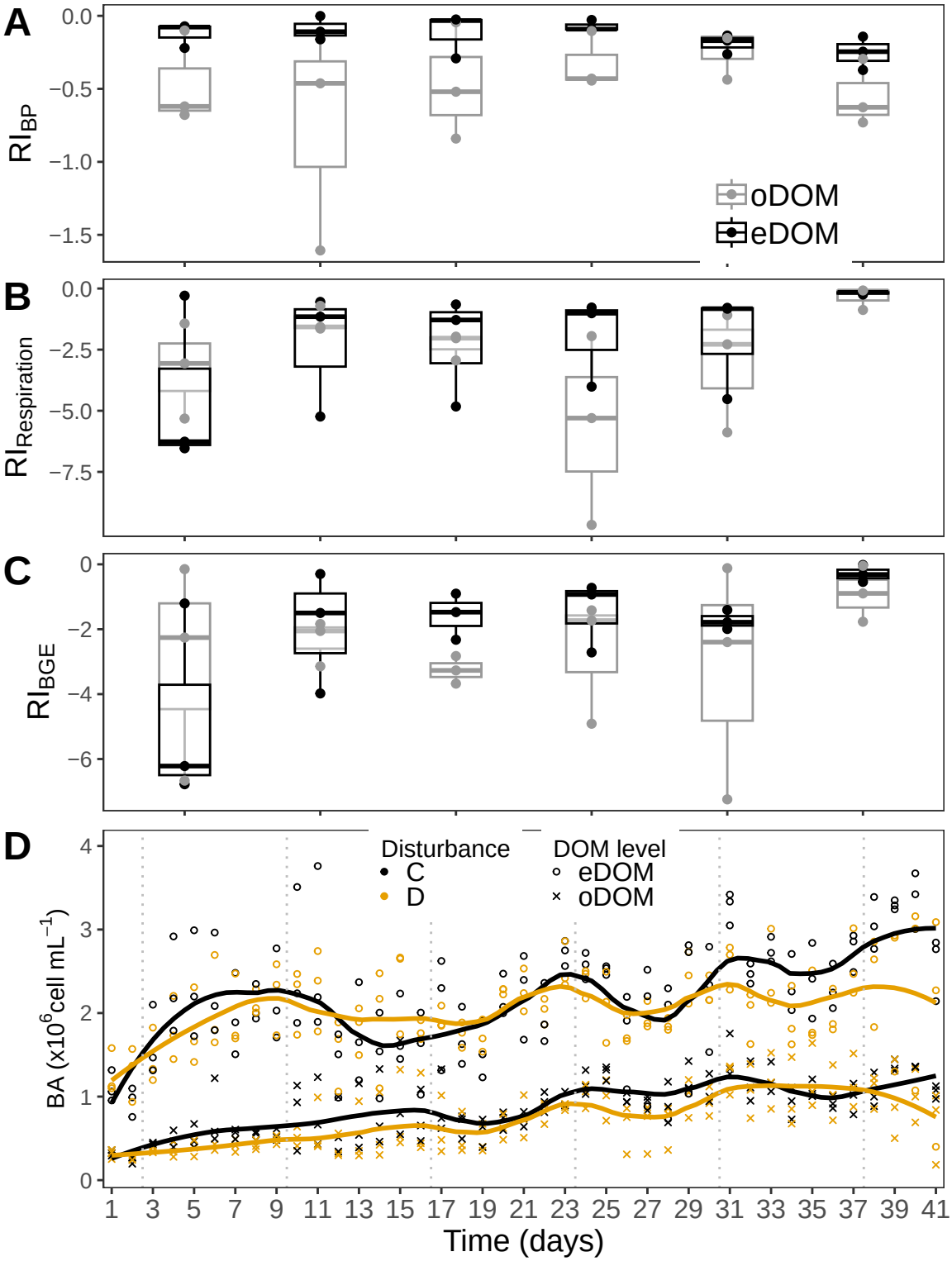


Figure 1: Community functional parameters. Resistance indices for A) bacterial production (BP), B) respiration and C)bacterial growth efficiency (BGE). D) Bacterial abundance (BA), lines indicate locally fitted values (loess smoothing) under each disturbance and DOM regime.