Biométricas

Ana Carolina Murad Lima

2023-06-22

```
# Bibliotecas
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.3.1
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
# Leitura e tratamento dos dados
dados <- read_excel("Biométricas ok.xlsx")

# Ordenar o dataframe por quatro colunas diferentes
dados <- with(dados, dados[order(CICLO, BLOCO, EFLUENTE, INOCULO), ])

# Converter as colunas para tipo numérico e arredondar valores em duas casas
for (i in 5:9) {
  dados[, i] <- as.numeric(unlist(dados[, i]))
}
dados[5:9] =  round(dados[5:9], digits = 2)
str(dados)
```

```
## tibble [70 x 9] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO                          : num [1:70] 1 1 1 1 1 1 1 1 1 1 ...
##  $ EFLUENTE                       : num [1:70] 0 0 25 25 50 50 75 75 100 100 ...
##  $ INOCULO                        : chr [1:70] "COM" "SEM" "COM" "SEM" ...
##  $ CICLO                          : num [1:70] 1 1 1 1 1 1 1 1 1 1 ...
##  $ NUMERO DE PLANTAS POR M2       : num [1:70] 43 43 32 32 43 27 16 48 43 27 ...
```

```
##  $ MASSA FRESCA PARTE AEREA KG POR M2: num [1:70] 4.77 4.66 5.25 4.13 8.01 3.51 3.32 6.49 6.55 3.11
##  $ MASSA SECA PARTE AEREA KG POR M2  : num [1:70] 0.93 0.86 1.09 0.9 1.62 0.72 0.66 1.35 1.37 0.65 .
##  $ MASSA FRESCA RAIZ KG POR M2       : num [1:70] 0.38 0.47 0.42 0.44 0.59 0.41 0.34 0.61 0.59 0.37
##  $ AREA FOLIAR M2 POR M2             : num [1:70] 4.73 5.71 5.88 2.32 5.13 3.61 3.18 8.22 6.37 2.13
```

```r
# Transformar as colunas em variáveis categóricas
dados$BLOCO <- factor(dados$BLOCO)
dados$CICLO <- factor(dados$CICLO)
dados$INOCULO <- factor(dados$INOCULO)
dados$EFLUENTE <- factor(dados$EFLUENTE)
```

```r
# Níveis para cada fator de tratamentos
library(dae)
n.F <- 2
n.D <- 5
n.Bloco <- 4
tr <- data.frame(cbind(INOCULO = paste("I", rep(1:n.F, each = n.D, times =
                                                  n.Bloco),
                                       sep = ""),
                       EFLUENTE = paste("E", rep(1:n.D, times = n.F*n.Bloco),
                                        sep = "")))
units <- list(Bloco = n.Bloco,
              Parcela = (n.F*n.D))
nest <- list(Parcela = "Bloco")
(lay <- designRandomize(allocated = tr,
                        recipient = units,
                        nested.recipients = nest,
                        seed = 9719532))
```

```
##    Bloco Parcela INOCULO EFLUENTE
## 1      1       1      I2       E3
## 2      1       2      I1       E2
## 3      1       3      I1       E1
## 4      1       4      I1       E4
## 5      1       5      I2       E4
## 6      1       6      I1       E5
## 7      1       7      I2       E1
## 8      1       8      I1       E3
## 9      1       9      I2       E5
## 10     1      10      I2       E2
## 11     2       1      I1       E1
## 12     2       2      I2       E5
## 13     2       3      I2       E4
## 14     2       4      I1       E3
## 15     2       5      I2       E3
## 16     2       6      I2       E1
## 17     2       7      I1       E4
## 18     2       8      I2       E2
## 19     2       9      I1       E2
## 20     2      10      I1       E5
## 21     3       1      I1       E2
## 22     3       2      I2       E4
## 23     3       3      I2       E5
## 24     3       4      I2       E2
```

```
## 25     3       5      I1       E5
## 26     3       6      I2       E3
## 27     3       7      I1       E3
## 28     3       8      I1       E4
## 29     3       9      I1       E1
## 30     3      10      I2       E1
## 31     4       1      I2       E5
## 32     4       2      I2       E4
## 33     4       3      I1       E3
## 34     4       4      I1       E2
## 35     4       5      I1       E4
## 36     4       6      I2       E1
## 37     4       7      I2       E3
## 38     4       8      I2       E2
## 39     4       9      I1       E1
## 40     4      10      I1       E5
```

```
table(lay$I)
```

```
##
## I1 I2
## 20 20
```

```
table(lay$E)
```

```
##
## E1 E2 E3 E4 E5
##  8  8  8  8  8
```

```
lay$Tratamento <- factor(paste(lay$I, lay$E, sep = ":"))
print(lay$Tratamento)
```

```
##  [1] I2:E3 I1:E2 I1:E1 I1:E4 I2:E4 I1:E5 I2:E1 I1:E3 I2:E5 I2:E2 I1:E1 I2:E5
## [13] I2:E4 I1:E3 I2:E3 I2:E1 I1:E4 I2:E2 I1:E2 I1:E5 I1:E2 I2:E4 I2:E5 I2:E2
## [25] I1:E5 I2:E3 I1:E3 I1:E4 I1:E1 I2:E1 I2:E5 I2:E4 I1:E3 I1:E2 I1:E4 I2:E1
## [37] I2:E3 I2:E2 I1:E1 I1:E5
## Levels: I1:E1 I1:E2 I1:E3 I1:E4 I1:E5 I2:E1 I2:E2 I2:E3 I2:E4 I2:E5
```

```
# Separar os dados de acordo com o período de tempo da coleta
dados_1 = dados[c(1:5)]
dados_1$Num_plantas_M2 = dados_1$`NUMERO DE PLANTAS POR M2`
dados_1 = dados_1[-5]
dados_2 = dados[c(1:4,6)]
dados_2$Massa_fre_aerea_M2 = dados_2$`MASSA FRESCA PARTE AEREA KG POR M2`
dados_2 = dados_2[-5]
dados_3 = dados[c(1:4,7)]
dados_3$Massa_sec_aerea_M2 = dados_3$`MASSA SECA PARTE AEREA KG POR M2`
dados_3 = dados_3[-5]
dados_4 = dados[c(1:4,8)]
dados_4$Massa_fre_raiz_M2 = dados_4$`MASSA FRESCA RAIZ KG POR M2`
dados_4 = dados_4[-5]
```

```r
dados_5 = dados[c(1:4,9)]
dados_5$Area_foliar_M2 = dados_5$`AREA FOLIAR M2 POR M2`
dados_5 = dados_5[-5]

# Estrutura dos dados após separados
"dados_1"
```

```
## [1] "dados_1"
```

```r
str(dados_1)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO         : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
##  $ EFLUENTE      : Factor w/ 5 levels "0","25","50",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ INOCULO       : Factor w/ 2 levels "COM","SEM": 1 2 1 2 1 2 1 2 1 2 ...
##  $ CICLO         : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Num_plantas_M2: num [1:70] 43 43 32 32 43 27 16 48 43 27 ...
```

```r
"dados_2"
```

```
## [1] "dados_2"
```

```r
str(dados_2)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO             : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
##  $ EFLUENTE          : Factor w/ 5 levels "0","25","50",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ INOCULO           : Factor w/ 2 levels "COM","SEM": 1 2 1 2 1 2 1 2 1 2 ...
##  $ CICLO             : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Massa_fre_aerea_M2: num [1:70] 4.77 4.66 5.25 4.13 8.01 3.51 3.32 6.49 6.55 3.11 ...
```

```r
"dados_3"
```

```
## [1] "dados_3"
```

```r
str(dados_3)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO             : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
##  $ EFLUENTE          : Factor w/ 5 levels "0","25","50",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ INOCULO           : Factor w/ 2 levels "COM","SEM": 1 2 1 2 1 2 1 2 1 2 ...
##  $ CICLO             : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Massa_sec_aerea_M2: num [1:70] 0.93 0.86 1.09 0.9 1.62 0.72 0.66 1.35 1.37 0.65 ...
```

```r
"dados_4"
```

```
## [1] "dados_4"
```

```
str(dados_4)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO            : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
##  $ EFLUENTE         : Factor w/ 5 levels "0","25","50",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ INOCULO          : Factor w/ 2 levels "COM","SEM": 1 2 1 2 1 2 1 2 1 2 ...
##  $ CICLO            : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Massa_fre_raiz_M2: num [1:70] 0.38 0.47 0.42 0.44 0.59 0.41 0.34 0.61 0.59 0.37 ...
```

```
"dados_5"
```

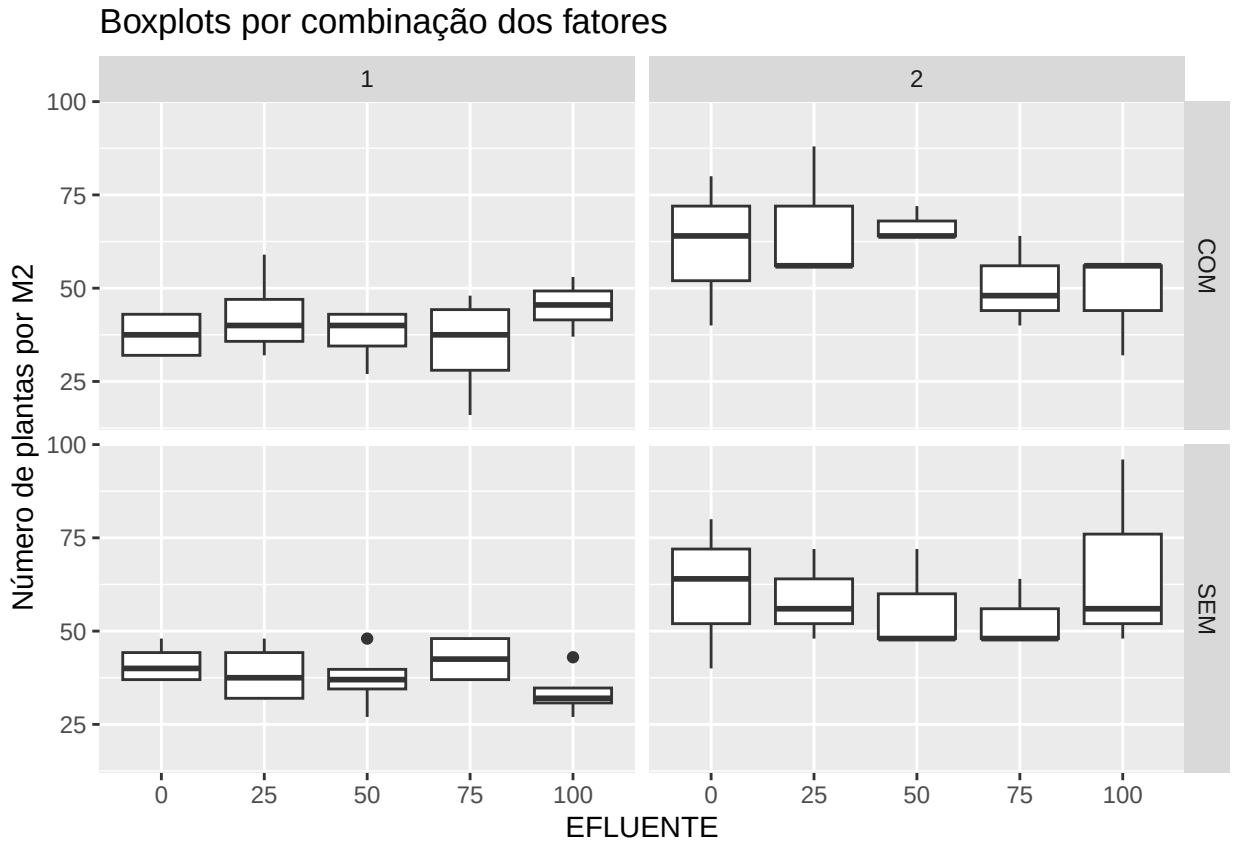
```
## [1] "dados_5"
```

```
str(dados_5)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO         : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
##  $ EFLUENTE      : Factor w/ 5 levels "0","25","50",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ INOCULO       : Factor w/ 2 levels "COM","SEM": 1 2 1 2 1 2 1 2 1 2 ...
##  $ CICLO         : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Area_foliar_M2: num [1:70] 4.73 5.71 5.88 2.32 5.13 3.61 3.18 8.22 6.37 2.13 ...
```

Análise para número de plantas por M2

```
# Gráficos com os boxplots para cada combinação de todos os fatores
ggplot(dados_1, aes(x = factor(EFLUENTE), y = Num_plantas_M2)) +
  geom_boxplot() +
  facet_grid(INOCULO ~ CICLO) +
  labs(x = "EFLUENTE", y = "Número de plantas por M2") +
  ggtitle("Boxplots por combinação dos fatores")
```




```r
# Calcular os limites inferiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Num_plantas_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_1, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
  limite_inferior <- q[1] - 1.5 * iqr
})

# Armazenar vetor com os limites inferiores
lim_inf = limites_outliers$Num_plantas_M2

# Calcular os limites superiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Num_plantas_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_1, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
  limite_superior <- q[2] + 1.5 * iqr
})

# Armazenar vetor com os limites superiores
lim_sup = limites_outliers$Num_plantas_M2

# Montar Dataframe com os limites inferior e superior
limites_outliers = limites_outliers[c(1:3)]
limites_outliers$LIM_INF = lim_inf
limites_outliers$LIM_SUP = lim_sup
```

```r
# Definir o número de replicação de acordo com o valor de CICLO
num_rep <- ifelse(limites_outliers$CICLO == 1, 4, 3)

# Replicar as linhas do dataframe
limites_outliers <- limites_outliers[rep(row.names(limites_outliers),
                                         num_rep), ]

# Redefinir os índices das linhas
rownames(limites_outliers) <- NULL

# Reordenar os dados do tempo 1 seguindo a combinação de fatores por cada bloco
blocos_dados_1 <- with(dados_1,
                           dados_1[order(CICLO, INOCULO, EFLUENTE), ])

# Definir como NA (valor ausente) os outliers
blocos_dados_1$Num_plantas_M2[which(blocos_dados_1$Num_plantas_M2 <
                                    limites_outliers$LIM_INF)] = NA
blocos_dados_1$Num_plantas_M2[which(blocos_dados_1$Num_plantas_M2 >
                                    limites_outliers$LIM_SUP)] = NA

# Calcular a média para cada grupo de 4 linhas
media_blocos_1 = aggregate(Num_plantas_M2 ~ EFLUENTE + INOCULO + CICLO,
                               data = blocos_dados_1, FUN = mean)

# Replicar cada linha por 4 vezes
media_blocos_1 = media_blocos_1[rep(row.names(media_blocos_1),
                                            each = 4), ]

# Redefinir os índices das linhas
rownames(media_blocos_1) <- NULL

# Preencher os NA's com as médias dos 4 blocos para a
# combinação de fatores específica
blocos_dados_1$Num_plantas_M2[which(is.na(blocos_dados_1$Num_plantas_M2))] =
  media_blocos_1$Num_plantas_M2[which(is.na(blocos_dados_1$Num_plantas_M2))]
```

```r
# Análises Descritivas
str(blocos_dados_1)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO         : Factor w/ 4 levels "1","2","3","4": 1 2 3 4 1 2 3 4 1 2 ...
##  $ EFLUENTE      : Factor w/ 5 levels "0","25","50",..: 1 1 1 1 2 2 2 2 3 3 ...
##  $ INOCULO       : Factor w/ 2 levels "COM","SEM": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CICLO         : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Num_plantas_M2: num [1:70] 43 32 32 43 32 37 43 59 43 27 ...
```

```r
summary(blocos_dados_1)
```

```
##  BLOCO  EFLUENTE INOCULO  CICLO  Num_plantas_M2
##  1:20   0  :14   COM:35   1:40   Min.   :16.00
##  2:20   25 :14   SEM:35   2:30   1st Qu.:37.00
##  3:20   50 :14                   Median :43.00
```

```
##  4:10   75 :14                   Mean   :47.21
##         100:14                   3rd Qu.:56.00
##                                  Max.   :96.00
```

```
# Número de observações
with(blocos_dados_1, tapply(Num_plantas_M2, list(EFLUENTE, INOCULO), length))
```

```
##     COM SEM
## 0     7   7
## 25    7   7
## 50    7   7
## 75    7   7
## 100   7   7
```

```
with(blocos_dados_1, tapply(Num_plantas_M2, list(EFLUENTE, INOCULO), sum))
```

```
##     COM      SEM
## 0   334 349.0000
## 25  371 331.0000
## 50  350 302.6667
## 75  291 330.0000
## 100 325 321.3333
```

```
with(blocos_dados_1, tapply(Num_plantas_M2, list(EFLUENTE, INOCULO), mean))
```

```
##          COM      SEM
## 0   47.71429 49.85714
## 25  53.00000 47.28571
## 50  50.00000 43.23810
## 75  41.57143 47.14286
## 100 46.42857 45.90476
```

```
with(blocos_dados_1, tapply(Num_plantas_M2, list(EFLUENTE, INOCULO), var))
```

```
##           COM       SEM
## 0   317.57143 264.47619
## 25  346.00000 195.57143
## 50  278.66667 217.61905
## 75  222.61905  82.14286
## 100  89.61905 600.39683
```

```
with(blocos_dados_1, tapply(Num_plantas_M2, list(EFLUENTE, INOCULO), sd))
```

```
##           COM      SEM
## 0   17.820534 16.26272
## 25  18.601075 13.98469
## 50  16.693312 14.75192
## 75  14.920424  9.06327
## 100  9.466734 24.50300
```

```r
# Tamanho das Médias
T.medias <- with(blocos_dados_1,
                 model.tables(aov(Num_plantas_M2 ~ CICLO + BLOCO + EFLUENTE + INOCULO +
                                    CICLO:EFLUENTE:INOCULO + CICLO:EFLUENTE +
                                    CICLO:INOCULO + EFLUENTE:INOCULO),
                              "means"))
T.medias
```

```
## Tables of means
## Grand mean
##
## 47.21429
##
##  CICLO
##         1     2
##     38.42 58.93
## rep 40.00 30.00
##
##  BLOCO
##         1    2     3     4
##     45.84 42.8 50.67 51.89
## rep 20.00 20.0 20.00 10.00
##
##  EFLUENTE
##         0    25    50    75   100
##     48.79 50.14 46.62 44.36 46.17
## rep 14.00 14.00 14.00 14.00 14.00
##
##  INOCULO
##       COM   SEM
##     47.74 46.69
## rep 35.00 35.00
##
##  CICLO:EFLUENTE
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   39.37 40.75 35.58 38.62 37.79
##   rep  8.00  8.00  8.00  8.00  8.00
##   2   61.33 62.67 61.33 52.00 57.33
##   rep  6.00  6.00  6.00  6.00  6.00
##
##  CICLO:INOCULO
##      INOCULO
## CICLO COM   SEM
##   1   39.55 37.30
##   rep 20.00 20.00
##   2   58.67 59.20
##   rep 15.00 15.00
##
##  EFLUENTE:INOCULO
##         INOCULO
## EFLUENTE COM   SEM
##      0   47.71 49.86
##      rep  7.00 46.43
```

```
##      25  53.00 47.29
##      rep 47.71 49.86
##      50  50.00 43.24
##      rep 53.00 47.29
##      75  41.57 47.14
##      rep 50.00 43.24
##      100 46.43 45.90
##      rep 41.57 47.14
##
##  CICLO:EFLUENTE:INOCULO
## , , INOCULO = COM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   37.50 42.75 37.50 34.75 45.25
##   rep  4.00  4.00  4.00  4.00  4.00
##   2   61.33 66.67 66.67 50.67 48.00
##   rep  3.00  3.00  3.00  3.00  3.00
##
## , , INOCULO = SEM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   41.25 38.75 33.67 42.50 30.33
##   rep  4.00  4.00  4.00  4.00  4.00
##   2   61.33 58.67 56.00 53.33 66.67
##   rep  3.00  3.00  3.00  3.00  3.00
```

```r
# Media dos blocos para fazer os testes de Normalidade e Homocedasticidade
media_blocos_1 = aggregate(Num_plantas_M2 ~ EFLUENTE + INOCULO + CICLO,
                                data = blocos_dados_1, FUN = mean)

# Realizar o teste Shapiro-Wilk no conjunto de dados completo
resultado_shapiro <- shapiro.test(media_blocos_1$Num_plantas_M2)$p.value

# Exibir o resultado do teste Shapiro-Wilk
print(resultado_shapiro)
```

```
## [1] 0.1578677
```

```r
# Interpretação do p-valor
if (resultado_shapiro > 0.05) {
  print("Os dados seguem uma distribuição normal.")
} else {
  print("Os dados não seguem uma distribuição normal.")
}
```

```
## [1] "Os dados seguem uma distribuição normal."
```

```r
# Realizar o teste de Bartlett no conjunto de dados completo
resultado_bartlett <- bartlett.test(media_blocos_1$Num_plantas_M2,
                                    media_blocos_1$EFLUENTE,
```

```
                                    media_blocos_1$INOCULO,
                                    media_blocos_1$CICLO)$p.value

# Exibir o resultado do teste de Bartlett
print(resultado_bartlett)
```

```
## [1] 0.8997773
```

```
# Interpretação do p-valor
if (resultado_bartlett > 0.05) {
  print("A variância é homogênea entre os grupos.")
} else {
  print("A variância não é homogênea entre os grupos.")
}
```

```
## [1] "A variância é homogênea entre os grupos."
```

```
# Remover o bloco 4, pois a análise é muito influenciada pela presença deste
# no ciclo 1 e ausência no ciclo 2
blocos_dados_1 = subset(blocos_dados_1, BLOCO != 4)
```

```
# Ajustar o modelo linear para parcelas subdivididas
modelo_PARCELASUB = aov(Num_plantas_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE +
                          Error(BLOCO*INOCULO), data = blocos_dados_1)

# Visualizar os resultados do modelo
summary(modelo_PARCELASUB)
```

```
##
## Error: BLOCO
##       Df Sum Sq Mean Sq
## BLOCO  2  629.6   314.8
##
## Error: INOCULO
##          Df Sum Sq Mean Sq
## INOCULO  1    2.4     2.4
##
## Error: BLOCO:INOCULO
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  2  729.4   364.7
##
## Error: Within
##                         Df Sum Sq Mean Sq F value   Pr(>F)
## CICLO                    1   7304    7304  53.944 1.17e-08 ***
## EFLUENTE                 4    242      61   0.447    0.774
## CICLO:INOCULO            1      0       0   0.002    0.965
## CICLO:EFLUENTE           4    269      67   0.497    0.738
## INOCULO:EFLUENTE         4    525     131   0.968    0.437
## CICLO:INOCULO:EFLUENTE   4    933     233   1.722    0.166
## Residuals               36   4874     135
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
modelo = modelo_PARCELASUB
```

```r
# Refazer o modelo sem o erro, apenas para facilitar as funções subsequentes
modelo = aov(Num_plantas_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE,
             data = blocos_dados_1)

# Realizar a análise de variância (ANOVA)
anova_result <- anova(modelo)

# Filtrar apenas os fatores significativos ao nível 0.05
fatores_significativos <- anova_result[anova_result$"Pr(>F)" < 0.05, ]

# Exibir as interações significativas
print(fatores_significativos)
```

```
## Analysis of Variance Table
##
## Response: Num_plantas_M2
##       Df Sum Sq Mean Sq F value    Pr(>F)
## CICLO  1 7304.1  7304.1  49.529 2.187e-08 ***
## NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Vetor com os fatores significativos
nomes_linhas = row.names(fatores_significativos)

# Filtrar apenas as interações
interacoes_significativas <- nomes_linhas[nchar(nomes_linhas) > 8]

# Exibir o resultado
print(interacoes_significativas)
```

```
## character(0)
```

```r
# Realizar o teste de Tukey para todas as interações significativas
tukey_result <- TukeyHSD(modelo)
```

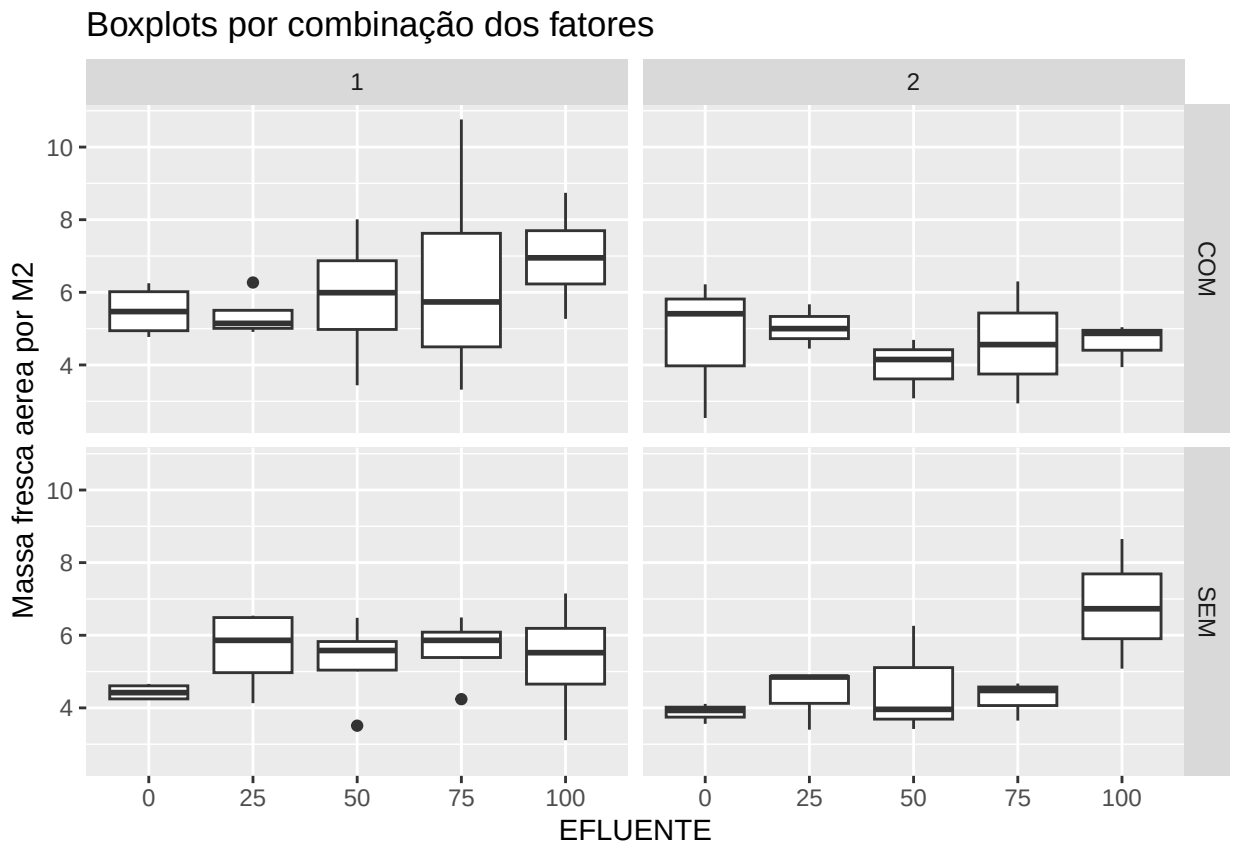
```r
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
```

```
    fator3 <- partes[[1]][3]
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == 0){
    media_interacao <- aggregate(Num_plantas_M2 ~ CICLO + INOCULO,
                                 data = blocos_dados_1, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Num_plantas_M2 ~ CICLO + EFLUENTE,
                                 data = blocos_dados_1, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "INOCULO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Num_plantas_M2 ~ INOCULO + EFLUENTE,
                                 data = blocos_dados_1, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == "EFLUENTE"){
    media_interacao <- aggregate(Num_plantas_M2 ~ CICLO + INOCULO + EFLUENTE,
                                 data = blocos_dados_1, FUN = mean)
    print(media_interacao)
  }
}
```

```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- partes[[1]][3]
  }

  inter_tukey = tukey_result[interacao]
  inter_tukey = data.frame(inter_tukey)
  colnames(inter_tukey) = c('diif','lwr','upr','p_adj')
  inter_tukey = subset(inter_tukey, p_adj < 0.05)
  print(inter_tukey)
}
```

Análise para Massa fresca aerea por M2

```
# Gráficos com os boxplots para cada combinação de todos os fatores
ggplot(dados_2, aes(x = factor(EFLUENTE), y = Massa_fre_aerea_M2)) +
  geom_boxplot() +
  facet_grid(INOCULO ~ CICLO) +
  labs(x = "EFLUENTE", y = "Massa fresca aerea por M2") +
  ggtitle("Boxplots por combinação dos fatores")
```




```
# Calcular os limites inferiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Massa_fre_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_2, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
  limite_inferior <- q[1] - 1.5 * iqr
})

# Armazenar vetor com os limites inferiores
lim_inf = limites_outliers$Massa_fre_aerea_M2

# Calcular os limites superiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Massa_fre_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_2, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
```

```
  limite_superior <- q[2] + 1.5 * iqr
})

# Armazenar vetor com os limites superiores
lim_sup = limites_outliers$Massa_fre_aerea_M2

# Montar Dataframe com os limites inferior e superior
limites_outliers = limites_outliers[c(1:3)]
limites_outliers$LIM_INF = lim_inf
limites_outliers$LIM_SUP = lim_sup

# Definir o número de replicação de acordo com o valor de CICLO
num_rep <- ifelse(limites_outliers$CICLO == 1, 4, 3)

# Replicar as linhas do dataframe
limites_outliers <- limites_outliers[rep(row.names(limites_outliers),
                                         num_rep), ]

# Redefinir os índices das linhas
rownames(limites_outliers) <- NULL

# Reordenar os dados do tempo 1 seguindo a combinação de fatores por cada bloco
blocos_dados_2 <- with(dados_2,
                       dados_2[order(CICLO, INOCULO, EFLUENTE), ])

# Definir como NA (valor ausente) os outliers
blocos_dados_2$Massa_fre_aerea_M2[which(blocos_dados_2$Massa_fre_aerea_M2 <
                                  limites_outliers$LIM_INF)] = NA
blocos_dados_2$Massa_fre_aerea_M2[which(blocos_dados_2$Massa_fre_aerea_M2 >
                                  limites_outliers$LIM_SUP)] = NA

# Calcular a média para cada grupo de 4 linhas
media_blocos_2 = aggregate(Massa_fre_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                           data = blocos_dados_2, FUN = mean)

# Replicar cada linha por 4 vezes
media_blocos_2 = media_blocos_2[rep(row.names(media_blocos_2),
                                    each = 4), ]

# Redefinir os índices das linhas
rownames(media_blocos_2) <- NULL

# Preencher os NA's com as médias dos 4 blocos para a
# combinação de fatores específica
blocos_dados_2$Massa_fre_aerea_M2[which(is.na(blocos_dados_2$Massa_fre_aerea_M2))] =
  media_blocos_2$Massa_fre_aerea_M2[which(is.na(blocos_dados_2$Massa_fre_aerea_M2))]
```

```
# Análises Descritivas
str(blocos_dados_2)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO             : Factor w/ 4 levels "1","2","3","4": 1 2 3 4 1 2 3 4 1 2 ...
##  $ EFLUENTE          : Factor w/ 5 levels "0","25","50",..: 1 1 1 1 2 2 2 2 3 3 ...
```

```
##  $ INOCULO           : Factor w/ 2 levels "COM","SEM": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CICLO             : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Massa_fre_aerea_M2: num [1:70] 4.77 5 6.25 5.94 5.25 ...
```

```
summary(blocos_dados_2)
```

```
##  BLOCO  EFLUENTE INOCULO  CICLO  Massa_fre_aerea_M2
##  1:20   0  :14   COM:35   1:40   Min.   : 2.540
##  2:20   25 :14   SEM:35   2:30   1st Qu.: 4.300
##  3:20   50 :14                   Median : 5.053
##  4:10   75 :14                   Mean   : 5.269
##         100:14                   3rd Qu.: 6.183
##                                  Max.   :10.760
```

```
# Número de observações
with(blocos_dados_2, tapply(Massa_fre_aerea_M2, list(EFLUENTE, INOCULO), length))
```

```
##     COM SEM
## 0     7   7
## 25    7   7
## 50    7   7
## 75    7   7
## 100   7   7
```

```
with(blocos_dados_2, tapply(Massa_fre_aerea_M2, list(EFLUENTE, INOCULO), sum))
```

```
##          COM   SEM
## 0   36.13000 29.34
## 25  35.38667 35.52
## 50  35.35000 37.16
## 75  39.35000 37.08
## 100 41.76000 41.76
```

```
with(blocos_dados_2, tapply(Massa_fre_aerea_M2, list(EFLUENTE, INOCULO), mean))
```

```
##          COM      SEM
## 0   5.161429 4.191429
## 25  5.055238 5.074286
## 50  5.050000 5.308571
## 75  5.621429 5.297143
## 100 5.965714 5.965714
```

```
with(blocos_dados_2, tapply(Massa_fre_aerea_M2, list(EFLUENTE, INOCULO), var))
```

```
##           COM      SEM
## 0   1.6707810 0.142981
## 25  0.1344476 1.318362
## 50  3.0735667 1.356414
## 75  6.9887476 1.074024
## 100 2.7665619 3.128995
```

```r
with(blocos_dados_2, tapply(Massa_fre_aerea_M2, list(EFLUENTE, INOCULO), sd))
```

```
##          COM       SEM
## 0   1.292587 0.3781282
## 25  0.366671 1.1481994
## 50  1.753159 1.1646520
## 75  2.643624 1.0363512
## 100 1.663299 1.7688966
```

```r
# Tamanho das Médias
T.medias <- with(blocos_dados_2,
                 model.tables(aov(Massa_fre_aerea_M2 ~ CICLO + BLOCO + EFLUENTE + INOCULO +
                                    CICLO:EFLUENTE:INOCULO + CICLO:EFLUENTE +
                                    CICLO:INOCULO + EFLUENTE:INOCULO),
                              "means"))
T.medias
```

```
## Tables of means
## Grand mean
##
## 5.269095
##
##  CICLO
##          1      2
##      5.709  4.683
## rep 40.000 30.000
##
##  BLOCO
##          1    2      3      4
##      5.213  4.9  5.479  5.699
## rep 20.000 20.0 20.000 10.000
##
##  EFLUENTE
##          0     25     50     75    100
##      4.676  5.065  5.179  5.459  5.966
## rep 14.000 14.000 14.000 14.000 14.000
##
##  INOCULO
##        COM    SEM
##      5.371  5.167
## rep 35.000 35.000
##
##  CICLO:EFLUENTE
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   4.962 5.332 5.869 6.229 6.151
##   rep 8.000 8.000 8.000 8.000 8.000
##   2   4.295 4.708 4.260 4.433 5.718
##   rep 6.000 6.000 6.000 6.000 6.000
##
##  CICLO:INOCULO
##      INOCULO
## CICLO COM    SEM
```

```
##   1    5.956  5.461
##   rep 20.000 20.000
##   2    4.591  4.775
##   rep 15.000 15.000
##
##  EFLUENTE:INOCULO
##         INOCULO
## EFLUENTE COM   SEM
##      0   5.161 4.191
##      rep 7.000 5.966
##      25  5.055 5.074
##      rep 5.161 4.191
##      50  5.050 5.309
##      rep 5.055 5.074
##      75  5.621 5.297
##      rep 5.050 5.309
##      100 5.966 5.966
##      rep 5.621 5.297
##
##  CICLO:EFLUENTE:INOCULO
## , , INOCULO = COM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   5.490 5.067 5.857 6.387 6.977
##   rep 4.000 4.000 4.000 4.000 4.000
##   2   4.723 5.040 3.973 4.600 4.617
##   rep 3.000 3.000 3.000 3.000 3.000
##
## , , INOCULO = SEM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   4.435 5.597 5.880 6.070 5.325
##   rep 4.000 4.000 4.000 4.000 4.000
##   2   3.867 4.377 4.547 4.267 6.820
##   rep 3.000 3.000 3.000 3.000 3.000
```

```
# Media dos blocos para fazer os testes de Normalidade e Homocedasticidade
media_blocos_2 = aggregate(Massa_fre_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                                  data = blocos_dados_2, FUN = mean)

# Realizar o teste Shapiro-Wilk no conjunto de dados completo
resultado_shapiro <- shapiro.test(media_blocos_2$Massa_fre_aerea_M2)$p.value

# Exibir o resultado do teste Shapiro-Wilk
print(resultado_shapiro)
```

```
## [1] 0.3888269
```

```
# Interpretação do p-valor
if (resultado_shapiro > 0.05) {
  print("Os dados seguem uma distribuição normal.")
```

```
} else {
  print("Os dados não seguem uma distribuição normal.")
}
```

```
## [1] "Os dados seguem uma distribuição normal."
```

```
# Realizar o teste de Bartlett no conjunto de dados completo
resultado_bartlett <- bartlett.test(media_blocos_2$Massa_fre_aerea_M2,
                                    media_blocos_2$EFLUENTE,
                                    media_blocos_2$INOCULO,
                                    media_blocos_2$CICLO)$p.value

# Exibir o resultado do teste de Bartlett
print(resultado_bartlett)
```

```
## [1] 0.6999333
```

```
# Interpretação do p-valor
if (resultado_bartlett > 0.05) {
  print("A variância é homogênea entre os grupos.")
} else {
  print("A variância não é homogênea entre os grupos.")
}
```

```
## [1] "A variância é homogênea entre os grupos."
```

```
# Remover o bloco 4, pois a análise é muito influenciada pela presença deste
# no ciclo 1 e ausência no ciclo 2
blocos_dados_2 = subset(blocos_dados_2, BLOCO != 4)
```

```
# Ajustar o modelo linear para parcelas subdivididas
modelo_PARCELASUB = aov(Massa_fre_aerea_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE +
                          Error(BLOCO*INOCULO), data = blocos_dados_2)

# Visualizar os resultados do modelo
summary(modelo_PARCELASUB)
```

```
##
## Error: BLOCO
##       Df Sum Sq Mean Sq
## BLOCO  2  3.364   1.682
##
## Error: INOCULO
##          Df Sum Sq Mean Sq
## INOCULO   1 0.1693  0.1693
##
## Error: BLOCO:INOCULO
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  2  13.54   6.768
##
## Error: Within
```

```
##                          Df Sum Sq Mean Sq F value Pr(>F)
## CICLO                     1  11.67  11.675   6.130 0.0181 *
## EFLUENTE                  4  10.22   2.554   1.341 0.2737
## CICLO:INOCULO             1   1.27   1.269   0.666 0.4197
## CICLO:EFLUENTE            4   5.55   1.387   0.728 0.5786
## INOCULO:EFLUENTE          4   4.05   1.011   0.531 0.7137
## CICLO:INOCULO:EFLUENTE    4   7.24   1.810   0.951 0.4463
## Residuals                36  68.56   1.905
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
modelo = modelo_PARCELASUB
```

```
# Refazer o modelo sem o erro, apenas para facilitar as funções subsequentes
modelo = aov(Massa_fre_aerea_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE,
             data = blocos_dados_2)

# Realizar a análise de variância (ANOVA)
anova_result <- anova(modelo)

# Filtrar apenas os fatores significativos ao nível 0.05
fatores_significativos <- anova_result[anova_result$"Pr(>F)" < 0.05, ]

# Exibir as interações significativas
print(fatores_significativos)
```

```
## Analysis of Variance Table
##
## Response: Massa_fre_aerea_M2
##       Df Sum Sq Mean Sq F value  Pr(>F)
## CICLO  1 11.675  11.675  5.4036 0.02554 *
## NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Vetor com os fatores significativos
nomes_linhas = row.names(fatores_significativos)

# Filtrar apenas as interações
interacoes_significativas <- nomes_linhas[nchar(nomes_linhas) > 8]

# Exibir o resultado
print(interacoes_significativas)
```

```
## character(0)
```

```
# Realizar o teste de Tukey para todas as interações significativas
tukey_result <- TukeyHSD(modelo)
```

```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
```

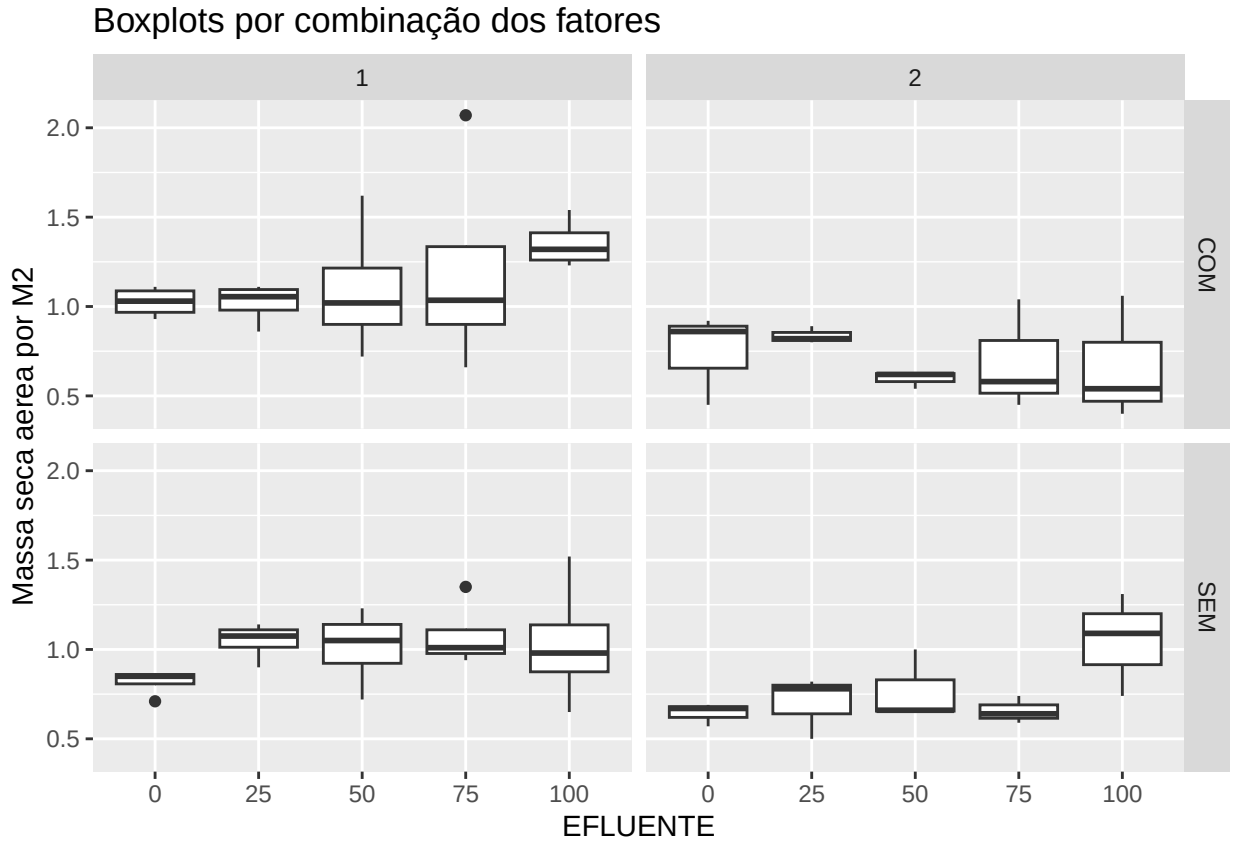
```
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- partes[[1]][3]
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == 0){
    media_interacao <- aggregate(Massa_fre_aerea_M2 ~ CICLO + INOCULO,
                                 data = blocos_dados_2, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Massa_fre_aerea_M2 ~ CICLO + EFLUENTE,
                                 data = blocos_dados_2, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "INOCULO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Massa_fre_aerea_M2 ~ INOCULO + EFLUENTE,
                                 data = blocos_dados_2, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == "EFLUENTE"){
    media_interacao <- aggregate(Massa_fre_aerea_M2 ~ CICLO + INOCULO + EFLUENTE,
                                 data = blocos_dados_2, FUN = mean)
    print(media_interacao)
  }
}
```

```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
```

```
    fator3 <- partes[[1]][3]
  }

  inter_tukey = tukey_result[interacao]
  inter_tukey = data.frame(inter_tukey)
  colnames(inter_tukey) = c('diif','lwr','upr','p_adj')
  inter_tukey = subset(inter_tukey, p_adj < 0.05)
  print(inter_tukey)
}
```

Análise para Massa seca aerea por M2

```
# Gráficos com os boxplots para cada combinação de todos os fatores
ggplot(dados_3, aes(x = factor(EFLUENTE), y = Massa_sec_aerea_M2)) +
  geom_boxplot() +
  facet_grid(INOCULO ~ CICLO) +
  labs(x = "EFLUENTE", y = "Massa seca aerea por M2") +
  ggtitle("Boxplots por combinação dos fatores")
```




```
# Calcular os limites inferiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Massa_sec_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_3, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
```

```
  iqr <- q[2] - q[1]
  limite_inferior <- q[1] - 1.5 * iqr
})

# Armazenar vetor com os limites inferiores
lim_inf = limites_outliers$Massa_sec_aerea_M2

# Calcular os limites superiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Massa_sec_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_3, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
  limite_superior <- q[2] + 1.5 * iqr
})

# Armazenar vetor com os limites superiores
lim_sup = limites_outliers$Massa_sec_aerea_M2

# Montar Dataframe com os limites inferior e superior
limites_outliers = limites_outliers[c(1:3)]
limites_outliers$LIM_INF = lim_inf
limites_outliers$LIM_SUP = lim_sup

# Definir o número de replicação de acordo com o valor de CICLO
num_rep <- ifelse(limites_outliers$CICLO == 1, 4, 3)

# Replicar as linhas do dataframe
limites_outliers <- limites_outliers[rep(row.names(limites_outliers),
                                         num_rep), ]

# Redefinir os índices das linhas
rownames(limites_outliers) <- NULL

# Reordenar os dados do tempo 1 seguindo a combinação de fatores por cada bloco
blocos_dados_3 <- with(dados_3,
                           dados_3[order(CICLO, INOCULO, EFLUENTE), ])

# Definir como NA (valor ausente) os outliers
blocos_dados_3$Massa_sec_aerea_M2[which(blocos_dados_3$Massa_sec_aerea_M2 <
                                    limites_outliers$LIM_INF)] = NA
blocos_dados_3$Massa_sec_aerea_M2[which(blocos_dados_3$Massa_sec_aerea_M2 >
                                    limites_outliers$LIM_SUP)] = NA

# Calcular a média para cada grupo de 4 linhas
media_blocos_3 = aggregate(Massa_sec_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                                data = blocos_dados_3, FUN = mean)

# Replicar cada linha por 4 vezes
media_blocos_3 = media_blocos_3[rep(row.names(media_blocos_3),
                                                each = 4), ]

# Redefinir os índices das linhas
rownames(media_blocos_3) <- NULL
```

```r
# Preencher os NA's com as médias dos 4 blocos para a
# combinação de fatores específica
blocos_dados_3$Massa_sec_aerea_M2[which(is.na(blocos_dados_3$Massa_sec_aerea_M2))] =
  media_blocos_3$Massa_sec_aerea_M2[which(is.na(blocos_dados_3$Massa_sec_aerea_M2))]
```

```r
# Análises Descritivas
str(blocos_dados_3)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO            : Factor w/ 4 levels "1","2","3","4": 1 2 3 4 1 2 3 4 1 2 ...
##  $ EFLUENTE         : Factor w/ 5 levels "0","25","50",..: 1 1 1 1 2 2 2 2 3 3 ...
##  $ INOCULO          : Factor w/ 2 levels "COM","SEM": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CICLO            : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Massa_sec_aerea_M2: num [1:70] 0.93 1.08 0.98 1.11 1.09 1.11 1.02 0.86 1.62 0.72 ...
```

```r
summary(blocos_dados_3)
```

```
##  BLOCO  EFLUENTE INOCULO  CICLO  Massa_sec_aerea_M2
##  1:20   0  :14   COM:35   1:40   Min.   :0.4000
##  2:20   25 :14   SEM:35   2:30   1st Qu.:0.6975
##  3:20   50 :14                   Median :0.9150
##  4:10   75 :14                   Mean   :0.9056
##         100:14                   3rd Qu.:1.0750
##                                  Max.   :1.6200
```

```r
# Número de observações
with(blocos_dados_3, tapply(Massa_sec_aerea_M2, list(EFLUENTE, INOCULO), length))
```

```
##     COM SEM
## 0     7   7
## 25    7   7
## 50    7   7
## 75    7   7
## 100   7   7
```

```r
with(blocos_dados_3, tapply(Massa_sec_aerea_M2, list(EFLUENTE, INOCULO), sum))
```

```
##      COM      SEM
## 0   6.33 5.343333
## 25  6.59 6.290000
## 50  6.17 6.360000
## 75  5.71 5.916667
## 100 7.41 7.270000
```

```r
with(blocos_dados_3, tapply(Massa_sec_aerea_M2, list(EFLUENTE, INOCULO), mean))
```

```
##           COM       SEM
## 0   0.9042857 0.7633333
## 25  0.9414286 0.8985714
## 50  0.8814286 0.9085714
## 75  0.8157143 0.8452381
## 100 1.0585714 1.0385714
```

```r
with(blocos_dados_3, tapply(Massa_sec_aerea_M2, list(EFLUENTE, INOCULO), var))
```

```
##            COM        SEM
## 0   0.04802857 0.01402222
## 25  0.01678095 0.05014762
## 50  0.14421429 0.05384762
## 75  0.06249524 0.03373651
## 100 0.18424762 0.09281429
```

```r
with(blocos_dados_3, tapply(Massa_sec_aerea_M2, list(EFLUENTE, INOCULO), sd))
```

```
##           COM       SEM
## 0   0.2191542 0.1184155
## 25  0.1295413 0.2239366
## 50  0.3797556 0.2320509
## 75  0.2499905 0.1836750
## 100 0.4292407 0.3046544
```

```r
# Tamanho das Médias
T.medias <- with(blocos_dados_3,
                 model.tables(aov(Massa_sec_aerea_M2 ~ CICLO + BLOCO + EFLUENTE + INOCULO +
                                    CICLO:EFLUENTE:INOCULO + CICLO:EFLUENTE +
                                    CICLO:INOCULO + EFLUENTE:INOCULO),
                              "means"))
T.medias
```

```
## Tables of means
## Grand mean
##
## 0.9055714
##
##  CICLO
##          1      2
##      1.034  0.735
## rep 40.000 30.000
##
##  BLOCO
##           1       2      3       4
##      0.8702  0.9068  0.915  0.9551
## rep 20.0000 20.0000 20.000 10.0000
##
##  EFLUENTE
##           0    25     50      75    100
##      0.8338  0.92  0.895  0.8305  1.049
## rep 14.0000 14.00 14.000 14.0000 14.000
##
##  INOCULO
##         COM     SEM
##      0.9203  0.8909
## rep 35.0000 35.0000
##
##  CICLO:EFLUENTE
```

```
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   0.939 1.034 1.054 0.948 1.193
##   rep 8.000 8.000 8.000 8.000 8.000
##   2   0.693 0.768 0.683 0.673 0.857
##   rep 6.000 6.000 6.000 6.000 6.000
##
##  CICLO:INOCULO
##      INOCULO
## CICLO COM    SEM
##   1    1.081  0.987
##   rep 20.000 20.000
##   2    0.707  0.763
##   rep 15.000 15.000
##
##  EFLUENTE:INOCULO
##         INOCULO
## EFLUENTE COM   SEM
##      0   0.904 0.763
##      rep 7.000 1.059
##      25  0.941 0.899
##      rep 0.904 0.763
##      50  0.881 0.909
##      rep 0.941 0.899
##      75  0.816 0.845
##      rep 0.881 0.909
##      100 1.059 1.039
##      rep 0.816 0.845
##
##  CICLO:EFLUENTE:INOCULO
## , , INOCULO = COM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   1.025 1.020 1.095 0.910 1.352
##   rep 4.000 4.000 4.000 4.000 4.000
##   2   0.743 0.837 0.597 0.690 0.667
##   rep 3.000 3.000 3.000 3.000 3.000
##
## , , INOCULO = SEM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   0.853 1.048 1.013 0.987 1.033
##   rep 4.000 4.000 4.000 4.000 4.000
##   2   0.643 0.700 0.770 0.657 1.047
##   rep 3.000 3.000 3.000 3.000 3.000
```

```r
# Media dos blocos para fazer os testes de Normalidade e Homocedasticidade
media_blocos_3 = aggregate(Massa_sec_aerea_M2 ~ EFLUENTE + INOCULO + CICLO,
                                data = blocos_dados_3, FUN = mean)

# Realizar o teste Shapiro-Wilk no conjunto de dados completo
resultado_shapiro <- shapiro.test(media_blocos_3$Massa_sec_aerea_M2)$p.value
```

```r
# Exibir o resultado do teste Shapiro-Wilk
print(resultado_shapiro)
```

```
## [1] 0.14652
```

```r
# Interpretação do p-valor
if (resultado_shapiro > 0.05) {
  print("Os dados seguem uma distribuição normal.")
} else {
  print("Os dados não seguem uma distribuição normal.")
}
```

```
## [1] "Os dados seguem uma distribuição normal."
```

```r
# Realizar o teste de Bartlett no conjunto de dados completo
resultado_bartlett <- bartlett.test(media_blocos_3$Massa_sec_aerea_M2,
                                    media_blocos_3$EFLUENTE,
                                    media_blocos_3$INOCULO,
                                    media_blocos_3$CICLO)$p.value

# Exibir o resultado do teste de Bartlett
print(resultado_bartlett)
```

```
## [1] 0.838975
```

```r
# Interpretação do p-valor
if (resultado_bartlett > 0.05) {
  print("A variância é homogênea entre os grupos.")
} else {
  print("A variância não é homogênea entre os grupos.")
}
```

```
## [1] "A variância é homogênea entre os grupos."
```

```r
# Remover o bloco 4, pois a análise é muito influenciada pela presença deste
# no ciclo 1 e ausência no ciclo 2
blocos_dados_3 = subset(blocos_dados_3, BLOCO != 4)
```

```r
# Ajustar o modelo linear para parcelas subdivididas
modelo_PARCELASUB = aov(Massa_sec_aerea_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE +
                          Error(BLOCO*INOCULO), data = blocos_dados_3)

# Visualizar os resultados do modelo
summary(modelo_PARCELASUB)
```

```
##
## Error: BLOCO
##       Df  Sum Sq Mean Sq
## BLOCO  2 0.02281  0.0114
##
```

```
## Error: INOCULO
##          Df   Sum Sq  Mean Sq
## INOCULO   1 0.004167 0.004167
##
## Error: BLOCO:INOCULO
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  2 0.2761  0.1381
##
## Error: Within
##                         Df Sum Sq Mean Sq F value  Pr(>F)
## CICLO                    1 1.1929  1.1929  26.739 8.9e-06 ***
## EFLUENTE                 4 0.3825  0.0956   2.144  0.0954 .
## CICLO:INOCULO            1 0.0807  0.0807   1.808  0.1871
## CICLO:EFLUENTE           4 0.0242  0.0060   0.136  0.9682
## INOCULO:EFLUENTE         4 0.0915  0.0229   0.513  0.7267
## CICLO:INOCULO:EFLUENTE   4 0.3091  0.0773   1.732  0.1643
## Residuals               36 1.6060  0.0446
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
modelo = modelo_PARCELASUB
```

```
# Refazer o modelo sem o erro, apenas para facilitar as funções subsequentes
modelo = aov(Massa_sec_aerea_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE,
             data = blocos_dados_3)

# Realizar a análise de variância (ANOVA)
anova_result <- anova(modelo)

# Filtrar apenas os fatores significativos ao nível 0.05
fatores_significativos <- anova_result[anova_result$"Pr(>F)" < 0.05, ]

# Exibir as interações significativas
print(fatores_significativos)
```

```
## Analysis of Variance Table
##
## Response: Massa_sec_aerea_M2
##       Df Sum Sq Mean Sq F value    Pr(>F)
## CICLO  1 1.1929  1.1929  24.084 1.772e-05 ***
## NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Vetor com os fatores significativos
nomes_linhas = row.names(fatores_significativos)

# Filtrar apenas as interações
interacoes_significativas <- nomes_linhas[nchar(nomes_linhas) > 8]

# Exibir o resultado
print(interacoes_significativas)
```

```
## character(0)
```

```
# Realizar o teste de Tukey para todas as interações significativas
tukey_result <- TukeyHSD(modelo)
```

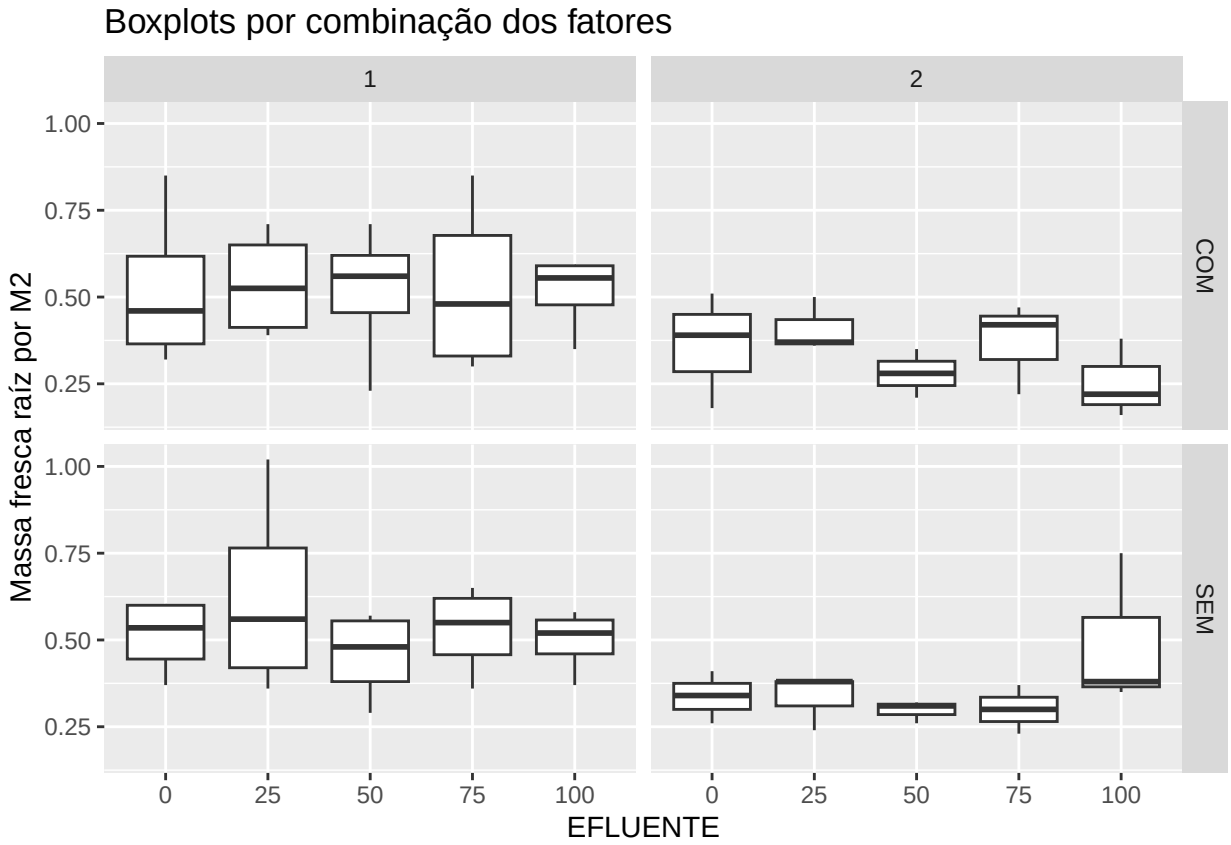
```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- partes[[1]][3]
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == 0){
    media_interacao <- aggregate(Massa_sec_aerea_M2 ~ CICLO + INOCULO,
                                 data = blocos_dados_3, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Massa_sec_aerea_M2 ~ CICLO + EFLUENTE,
                                 data = blocos_dados_3, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "INOCULO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Massa_sec_aerea_M2 ~ INOCULO + EFLUENTE,
                                 data = blocos_dados_3, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == "EFLUENTE"){
    media_interacao <- aggregate(Massa_sec_aerea_M2 ~ CICLO + INOCULO + EFLUENTE,
                                 data = blocos_dados_3, FUN = mean)
    print(media_interacao)
  }
}
```

```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
```

```
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- partes[[1]][3]
  }

  inter_tukey = tukey_result[interacao]
  inter_tukey = data.frame(inter_tukey)
  colnames(inter_tukey) = c('diif','lwr','upr','p_adj')
  inter_tukey = subset(inter_tukey, p_adj < 0.05)
  print(inter_tukey)
}
```

Análise para Massa fresca raíz por M2

```
# Gráficos com os boxplots para cada combinação de todos os fatores
ggplot(dados_4, aes(x = factor(EFLUENTE), y = Massa_fre_raiz_M2)) +
  geom_boxplot() +
  facet_grid(INOCULO ~ CICLO) +
  labs(x = "EFLUENTE", y = "Massa fresca raíz por M2") +
  ggtitle("Boxplots por combinação dos fatores")
```




```
# Calcular os limites inferiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Massa_fre_raiz_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_4, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
  limite_inferior <- q[1] - 1.5 * iqr
})

# Armazenar vetor com os limites inferiores
lim_inf = limites_outliers$Massa_fre_raiz_M2

# Calcular os limites superiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Massa_fre_raiz_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_4, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
  limite_superior <- q[2] + 1.5 * iqr
})

# Armazenar vetor com os limites superiores
lim_sup = limites_outliers$Massa_fre_raiz_M2

# Montar Dataframe com os limites inferior e superior
limites_outliers = limites_outliers[c(1:3)]
limites_outliers$LIM_INF = lim_inf
limites_outliers$LIM_SUP = lim_sup
```

```r
# Definir o número de replicação de acordo com o valor de CICLO
num_rep <- ifelse(limites_outliers$CICLO == 1, 4, 3)

# Replicar as linhas do dataframe
limites_outliers <- limites_outliers[rep(row.names(limites_outliers),
                                         num_rep), ]

# Redefinir os índices das linhas
rownames(limites_outliers) <- NULL

# Reordenar os dados do tempo 1 seguindo a combinação de fatores por cada bloco
blocos_dados_4 <- with(dados_4,
                           dados_4[order(CICLO, INOCULO, EFLUENTE), ])

# Definir como NA (valor ausente) os outliers
blocos_dados_4$Massa_fre_raiz_M2[which(blocos_dados_4$Massa_fre_raiz_M2 <
                                   limites_outliers$LIM_INF)] = NA
blocos_dados_4$Massa_fre_raiz_M2[which(blocos_dados_4$Massa_fre_raiz_M2 >
                                   limites_outliers$LIM_SUP)] = NA

# Calcular a média para cada grupo de 4 linhas
media_blocos_4 = aggregate(Massa_fre_raiz_M2 ~ EFLUENTE + INOCULO + CICLO,
                               data = blocos_dados_4, FUN = mean)

# Replicar cada linha por 4 vezes
media_blocos_4 = media_blocos_4[rep(row.names(media_blocos_4),
                                            each = 4), ]

# Redefinir os índices das linhas
rownames(media_blocos_4) <- NULL

# Preencher os NA's com as médias dos 4 blocos para a
# combinação de fatores específica
blocos_dados_4$Massa_fre_raiz_M2[which(is.na(blocos_dados_4$Massa_fre_raiz_M2))] =
  media_blocos_4$Massa_fre_raiz_M2[which(is.na(blocos_dados_4$Massa_fre_raiz_M2))]
```

```r
# Análises Descritivas
str(blocos_dados_4)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO           : Factor w/ 4 levels "1","2","3","4": 1 2 3 4 1 2 3 4 1 2 ...
##  $ EFLUENTE        : Factor w/ 5 levels "0","25","50",..: 1 1 1 1 2 2 2 2 3 3 ...
##  $ INOCULO         : Factor w/ 2 levels "COM","SEM": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CICLO           : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Massa_fre_raiz_M2: num [1:70] 0.38 0.32 0.85 0.54 0.42 0.39 0.71 0.63 0.59 0.23 ...
```

```r
summary(blocos_dados_4)
```

```
##  BLOCO  EFLUENTE INOCULO  CICLO  Massa_fre_raiz_M2
##  1:20   0  :14   COM:35   1:40   Min.   :0.160
##  2:20   25 :14   SEM:35   2:30   1st Qu.:0.340
##  3:20   50 :14                   Median :0.390
```

```
##  4:10   75 :14                  Mean   :0.446
##         100:14                  3rd Qu.:0.565
##                                 Max.   :1.020
```

```
# Número de observações
with(blocos_dados_4, tapply(Massa_fre_raiz_M2, list(EFLUENTE, INOCULO), length))
```

```
##     COM SEM
## 0     7   7
## 25    7   7
## 50    7   7
## 75    7   7
## 100   7   7
```

```
with(blocos_dados_4, tapply(Massa_fre_raiz_M2, list(EFLUENTE, INOCULO), sum))
```

```
##      COM  SEM
## 0   3.17 3.05
## 25  3.38 3.50
## 50  2.90 2.71
## 75  3.22 3.01
## 100 2.81 3.47
```

```
with(blocos_dados_4, tapply(Massa_fre_raiz_M2, list(EFLUENTE, INOCULO), mean))
```

```
##           COM       SEM
## 0   0.4528571 0.4357143
## 25  0.4828571 0.5000000
## 50  0.4142857 0.3871429
## 75  0.4600000 0.4300000
## 100 0.4014286 0.4957143
```

```
with(blocos_dados_4, tapply(Massa_fre_raiz_M2, list(EFLUENTE, INOCULO), var))
```

```
##            COM        SEM
## 0   0.04499048 0.01669524
## 25  0.01899048 0.07040000
## 50  0.03826190 0.01609048
## 75  0.04616667 0.02496667
## 100 0.02991429 0.02086190
```

```
with(blocos_dados_4, tapply(Massa_fre_raiz_M2, list(EFLUENTE, INOCULO), sd))
```

```
##           COM       SEM
## 0   0.2121096 0.1292101
## 25  0.1378059 0.2653300
## 50  0.1956065 0.1268482
## 75  0.2148643 0.1580084
## 100 0.1729575 0.1444365
```

```r
# Tamanho das Médias
T.medias <- with(blocos_dados_4,
                 model.tables(aov(Massa_fre_raiz_M2 ~ CICLO + BLOCO + EFLUENTE + INOCULO +
                                    CICLO:EFLUENTE:INOCULO + CICLO:EFLUENTE +
                                    CICLO:INOCULO + EFLUENTE:INOCULO),
                              "means"))
T.medias
```

```
## Tables of means
## Grand mean
##
## 0.446
##
##  CICLO
##          1       2
##      0.523  0.3433
## rep 40.000 30.0000
##
##  BLOCO
##           1       2       3     4
##      0.4303  0.3568  0.5088  0.53
## rep 20.0000 20.0000 20.0000 10.00
##
##  EFLUENTE
##           0      25      50     75     100
##      0.4443  0.4914  0.4007  0.445  0.4486
## rep 14.0000 14.0000 14.0000 14.000 14.0000
##
##  INOCULO
##         COM     SEM
##      0.4423  0.4497
## rep 35.0000 35.0000
##
##  CICLO:EFLUENTE
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   0.516 0.581 0.485 0.527 0.505
##   rep 8.000 8.000 8.000 8.000 8.000
##   2   0.348 0.372 0.288 0.335 0.373
##   rep 6.000 6.000 6.000 6.000 6.000
##
##  CICLO:INOCULO
##      INOCULO
## CICLO COM    SEM
##   1    0.523  0.523
##   rep 20.000 20.000
##   2    0.335  0.352
##   rep 15.000 15.000
##
##  EFLUENTE:INOCULO
##         INOCULO
## EFLUENTE COM   SEM
##      0   0.453 0.436
##      rep 7.000 0.401
```

```
##      25  0.483 0.500
##      rep 0.453 0.436
##      50  0.414 0.387
##      rep 0.483 0.500
##      75  0.460 0.430
##      rep 0.414 0.387
##      100 0.401 0.496
##      rep 0.460 0.430
##
##  CICLO:EFLUENTE:INOCULO
## , , INOCULO = COM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   0.523 0.538 0.515 0.527 0.512
##   rep 4.000 4.000 4.000 4.000 4.000
##   2   0.360 0.410 0.280 0.370 0.253
##   rep 3.000 3.000 3.000 3.000 3.000
##
## , , INOCULO = SEM
##
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   0.510 0.625 0.455 0.527 0.498
##   rep 4.000 4.000 4.000 4.000 4.000
##   2   0.337 0.333 0.297 0.300 0.493
##   rep 3.000 3.000 3.000 3.000 3.000
```

```r
# Media dos blocos para fazer os testes de Normalidade e Homocedasticidade
media_blocos_4 = aggregate(Massa_fre_raiz_M2 ~ EFLUENTE + INOCULO + CICLO,
                                data = blocos_dados_4, FUN = mean)

# Realizar o teste Shapiro-Wilk no conjunto de dados completo
resultado_shapiro <- shapiro.test(media_blocos_4$Massa_fre_raiz_M2)$p.value

# Exibir o resultado do teste Shapiro-Wilk
print(resultado_shapiro)
```

```
## [1] 0.07931075
```

```r
# Interpretação do p-valor
if (resultado_shapiro > 0.05) {
  print("Os dados seguem uma distribuição normal.")
} else {
  print("Os dados não seguem uma distribuição normal.")
}
```

```
## [1] "Os dados seguem uma distribuição normal."
```

```r
# Realizar o teste de Bartlett no conjunto de dados completo
resultado_bartlett <- bartlett.test(media_blocos_4$Massa_fre_raiz_M2,
                                    media_blocos_4$EFLUENTE,
```

```
                                    media_blocos_4$INOCULO,
                                    media_blocos_4$CICLO)$p.value

# Exibir o resultado do teste de Bartlett
print(resultado_bartlett)
```

```
## [1] 0.9933394
```

```
# Interpretação do p-valor
if (resultado_bartlett > 0.05) {
  print("A variância é homogênea entre os grupos.")
} else {
  print("A variância não é homogênea entre os grupos.")
}
```

```
## [1] "A variância é homogênea entre os grupos."
```

```
# Remover o bloco 4, pois a análise é muito influenciada pela presença deste
# no ciclo 1 e ausência no ciclo 2
blocos_dados_4 = subset(blocos_dados_4, BLOCO != 4)
```

```
# Ajustar o modelo linear para parcelas subdivididas
modelo_PARCELASUB = aov(Massa_fre_raiz_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE +
                          Error(BLOCO*INOCULO), data = blocos_dados_4)

# Visualizar os resultados do modelo
summary(modelo_PARCELASUB)
```

```
##
## Error: BLOCO
##       Df Sum Sq Mean Sq
## BLOCO  2 0.2311  0.1156
##
## Error: INOCULO
##          Df    Sum Sq   Mean Sq
## INOCULO  1 0.0002817 0.0002817
##
## Error: BLOCO:INOCULO
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  2 0.1117 0.05583
##
## Error: Within
##                         Df Sum Sq Mean Sq F value   Pr(>F)
## CICLO                    1 0.3450  0.3450  17.718 0.000163 ***
## EFLUENTE                 4 0.0294  0.0074   0.378 0.822775
## CICLO:INOCULO            1 0.0070  0.0070   0.362 0.551395
## CICLO:EFLUENTE           4 0.0090  0.0022   0.115 0.976271
## INOCULO:EFLUENTE         4 0.0465  0.0116   0.597 0.667078
## CICLO:INOCULO:EFLUENTE   4 0.0695  0.0174   0.892 0.478719
## Residuals               36 0.7011  0.0195
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
modelo = modelo_PARCELASUB
```

```r
# Refazer o modelo sem o erro, apenas para facilitar as funções subsequentes
modelo = aov(Massa_fre_raiz_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE,
             data = blocos_dados_4)

# Realizar a análise de variância (ANOVA)
anova_result <- anova(modelo)

# Filtrar apenas os fatores significativos ao nível 0.05
fatores_significativos <- anova_result[anova_result$"Pr(>F)" < 0.05, ]

# Exibir as interações significativas
print(fatores_significativos)
```

```
## Analysis of Variance Table
##
## Response: Massa_fre_raiz_M2
##       Df  Sum Sq Mean Sq F value    Pr(>F)
## BLOCO  2 0.23112 0.11556  5.4031 0.0086071 **
## CICLO  1 0.34504 0.34504 16.1325 0.0002689 ***
## NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Vetor com os fatores significativos
nomes_linhas = row.names(fatores_significativos)

# Filtrar apenas as interações
interacoes_significativas <- nomes_linhas[nchar(nomes_linhas) > 8]

# Exibir o resultado
print(interacoes_significativas)
```

```
## character(0)
```

```r
# Realizar o teste de Tukey para todas as interações significativas
tukey_result <- TukeyHSD(modelo)
```

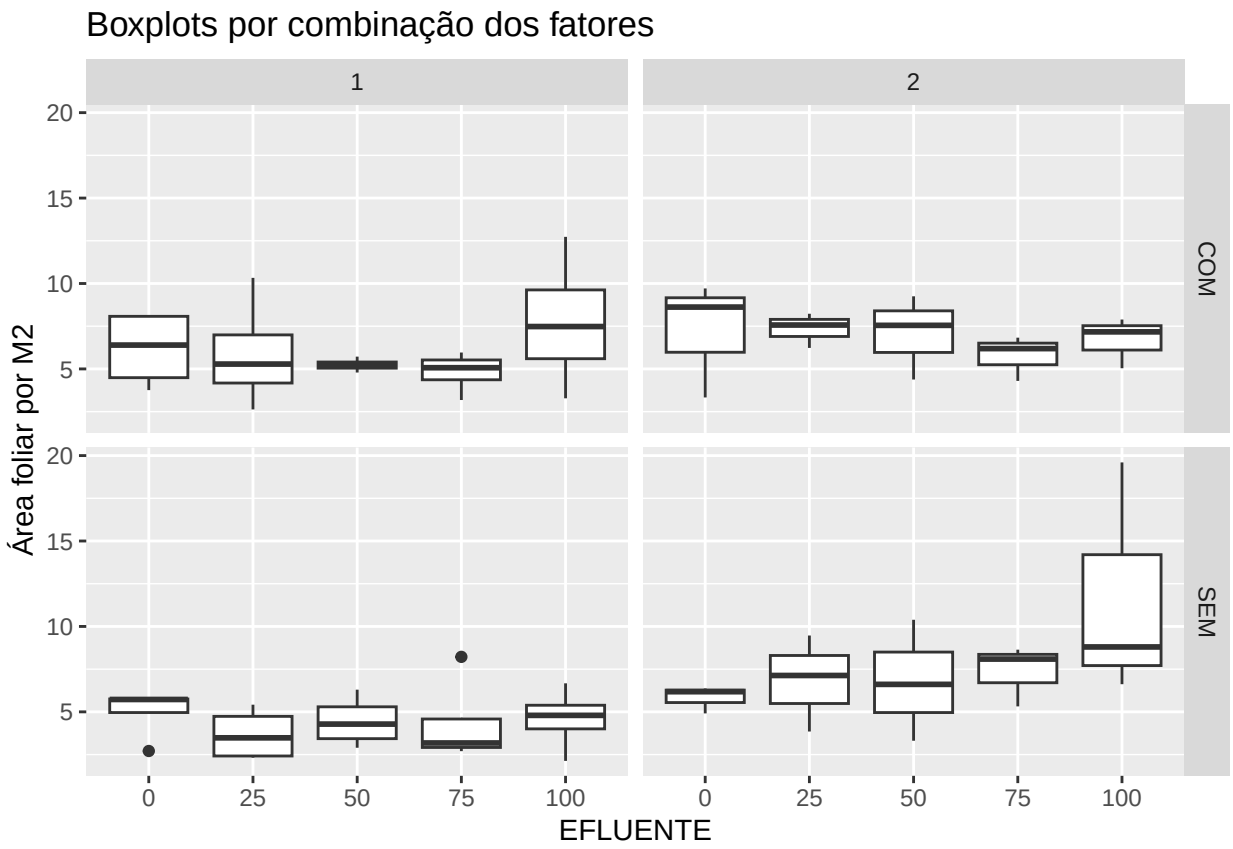
```r
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
```

```
    fator2 <- partes[[1]][2]
    fator3 <- partes[[1]][3]
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == 0){
    media_interacao <- aggregate(Massa_fre_raiz_M2 ~ CICLO + INOCULO,
                                 data = blocos_dados_4, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Massa_fre_raiz_M2 ~ CICLO + EFLUENTE,
                                 data = blocos_dados_4, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "INOCULO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Massa_fre_raiz_M2 ~ INOCULO + EFLUENTE,
                                 data = blocos_dados_4, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == "EFLUENTE"){
    media_interacao <- aggregate(Massa_fre_raiz_M2 ~ CICLO + INOCULO + EFLUENTE,
                                 data = blocos_dados_4, FUN = mean)
    print(media_interacao)
  }
}
```

```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- partes[[1]][3]
  }

  inter_tukey = tukey_result[interacao]
  inter_tukey = data.frame(inter_tukey)
  colnames(inter_tukey) = c('diif','lwr','upr','p_adj')
  inter_tukey = subset(inter_tukey, p_adj < 0.05)
  print(inter_tukey)
}
```

Análise para Massa fresca raíz por M2

```
# Gráficos com os boxplots para cada combinação de todos os fatores
ggplot(dados_5, aes(x = factor(EFLUENTE), y = Area_foliar_M2)) +
  geom_boxplot() +
  facet_grid(INOCULO ~ CICLO) +
  labs(x = "EFLUENTE", y = "Área foliar por M2") +
  ggtitle("Boxplots por combinação dos fatores")
```




```
# Calcular os limites inferiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Area_foliar_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_5, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
  limite_inferior <- q[1] - 1.5 * iqr
})

# Armazenar vetor com os limites inferiores
lim_inf = limites_outliers$Area_foliar_M2

# Calcular os limites superiores de outliers para cada combinação de fatores
limites_outliers <- aggregate(Area_foliar_M2 ~ EFLUENTE + INOCULO + CICLO,
                              data = dados_5, FUN = function(x) {
  q <- quantile(x, c(0.25, 0.75), na.rm = TRUE)
  iqr <- q[2] - q[1]
```

```
  limite_superior <- q[2] + 1.5 * iqr
})

# Armazenar vetor com os limites superiores
lim_sup = limites_outliers$Area_foliar_M2

# Montar Dataframe com os limites inferior e superior
limites_outliers = limites_outliers[c(1:3)]
limites_outliers$LIM_INF = lim_inf
limites_outliers$LIM_SUP = lim_sup

# Definir o número de replicação de acordo com o valor de CICLO
num_rep <- ifelse(limites_outliers$CICLO == 1, 4, 3)

# Replicar as linhas do dataframe
limites_outliers <- limites_outliers[rep(row.names(limites_outliers),
                                         num_rep), ]

# Redefinir os índices das linhas
rownames(limites_outliers) <- NULL

# Reordenar os dados do tempo 1 seguindo a combinação de fatores por cada bloco
blocos_dados_5 <- with(dados_5,
                           dados_5[order(CICLO, INOCULO, EFLUENTE), ])

# Definir como NA (valor ausente) os outliers
blocos_dados_5$Area_foliar_M2[which(blocos_dados_5$Area_foliar_M2 <
                                      limites_outliers$LIM_INF)] = NA
blocos_dados_5$Area_foliar_M2[which(blocos_dados_5$Area_foliar_M2 >
                                      limites_outliers$LIM_SUP)] = NA

# Calcular a média para cada grupo de 4 linhas
media_blocos_5 = aggregate(Area_foliar_M2 ~ EFLUENTE + INOCULO + CICLO,
                                 data = blocos_dados_5, FUN = mean)

# Replicar cada linha por 4 vezes
media_blocos_5 = media_blocos_5[rep(row.names(media_blocos_5),
                                              each = 4), ]

# Redefinir os índices das linhas
rownames(media_blocos_5) <- NULL

# Preencher os NA's com as médias dos 4 blocos para a
# combinação de fatores específica
blocos_dados_5$Area_foliar_M2[which(is.na(blocos_dados_5$Area_foliar_M2))] =
  media_blocos_5$Area_foliar_M2[which(is.na(blocos_dados_5$Area_foliar_M2))]
```

```
# Análises Descritivas
str(blocos_dados_5)
```

```
## tibble [70 x 5] (S3: tbl_df/tbl/data.frame)
##  $ BLOCO         : Factor w/ 4 levels "1","2","3","4": 1 2 3 4 1 2 3 4 1 2 ...
##  $ EFLUENTE      : Factor w/ 5 levels "0","25","50",..: 1 1 1 1 2 2 2 2 3 3 ...
```

```
##  $ INOCULO       : Factor w/ 2 levels "COM","SEM": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CICLO         : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Area_foliar_M2: num [1:70] 4.73 3.76 8.13 8.06 5.88 ...
```

```
summary(blocos_dados_5)
```

```
##  BLOCO  EFLUENTE INOCULO  CICLO  Area_foliar_M2
##  1:20   0  :14   COM:35   1:40   Min.   : 2.130
##  2:20   25 :14   SEM:35   2:30   1st Qu.: 4.412
##  3:20   50 :14                   Median : 5.730
##  4:10   75 :14                   Mean   : 6.042
##         100:14                   3rd Qu.: 7.455
##                                  Max.   :19.600
```

```
# Número de observações
with(blocos_dados_5, tapply(Area_foliar_M2, list(EFLUENTE, INOCULO), length))
```

```
##     COM SEM
## 0     7   7
## 25    7   7
## 50    7   7
## 75    7   7
## 100   7   7
```

```
with(blocos_dados_5, tapply(Area_foliar_M2, list(EFLUENTE, INOCULO), sum))
```

```
##       COM   SEM
## 0   46.34 40.46
## 25  45.56 35.15
## 50  42.12 38.08
## 75  36.60 34.13
## 100 51.07 53.41
```

```
with(blocos_dados_5, tapply(Area_foliar_M2, list(EFLUENTE, INOCULO), mean))
```

```
##          COM      SEM
## 0   6.620000 5.780000
## 25  6.508571 5.021429
## 50  6.017143 5.440000
## 75  5.228571 4.875714
## 100 7.295714 7.630000
```

```
with(blocos_dados_5, tapply(Area_foliar_M2, list(EFLUENTE, INOCULO), var))
```

```
##          COM        SEM
## 0   6.746267  0.2121333
## 25  6.252748  6.6564143
## 50  3.062790  6.8631333
## 75  1.554214  6.4498952
## 100 8.941129 32.1617333
```

```r
with(blocos_dados_5, tapply(Area_foliar_M2, list(EFLUENTE, INOCULO), sd))
```

```
##          COM       SEM
## 0   2.597358 0.4605793
## 25  2.500549 2.5800028
## 50  1.750083 2.6197583
## 75  1.246681 2.5396644
## 100 2.990172 5.6711316
```

```r
# Tamanho das Médias
T.medias <- with(blocos_dados_5,
                 model.tables(aov(Area_foliar_M2 ~ CICLO + BLOCO + EFLUENTE + INOCULO +
                                    CICLO:EFLUENTE:INOCULO + CICLO:EFLUENTE +
                                    CICLO:INOCULO + EFLUENTE:INOCULO),
                              "means"))
T.medias
```

```
## Tables of means
## Grand mean
##
## 6.041714
##
##  CICLO
##          1      2
##      5.133  7.253
## rep 40.000 30.000
##
##  BLOCO
##          1      2      3      4
##      6.059  5.888  5.565  7.269
## rep 20.000 20.000 20.000 10.000
##
##  EFLUENTE
##        0     25     50     75    100
##      6.2  5.765  5.729  5.052  7.463
## rep 14.0 14.000 14.000 14.000 14.000
##
##  INOCULO
##        COM    SEM
##      6.334  5.749
## rep 35.000 35.000
##
##  CICLO:EFLUENTE
##      EFLUENTE
## CICLO 0     25    50    75    100
##   1   5.960 4.779 4.839 3.920 6.170
##   rep 8.000 8.000 8.000 8.000 8.000
##   2   6.520 7.080 6.915 6.562 9.187
##   rep 6.000 6.000 6.000 6.000 6.000
##
##  CICLO:INOCULO
##      INOCULO
## CICLO COM    SEM
```

```
##   1    5.970  4.297
##   rep 20.000 20.000
##   2    6.819  7.686
##   rep 15.000 15.000
##
##  EFLUENTE:INOCULO
##         INOCULO
## EFLUENTE COM   SEM
##      0   6.620 5.780
##      rep 7.000 7.296
##      25  6.509 5.021
##      rep 6.620 5.780
##      50  6.017 5.440
##      rep 6.509 5.021
##      75  5.229 4.876
##      rep 6.017 5.440
##      100 7.296 7.630
##      rep 5.229 4.876
##
##  CICLO:EFLUENTE:INOCULO
## , , INOCULO = COM
##
##      EFLUENTE
## CICLO 0      25     50     75     100
##   1    6.170  5.883  5.235  4.820  7.742
##   rep  4.000  4.000  4.000  4.000  4.000
##   2    7.220  7.343  7.060  5.773  6.700
##   rep  3.000  3.000  3.000  3.000  3.000
##
## , , INOCULO = SEM
##
##      EFLUENTE
## CICLO 0      25     50     75     100
##   1    5.750  3.675  4.442  3.020  4.598
##   rep  4.000  4.000  4.000  4.000  4.000
##   2    5.820  6.817  6.770  7.350 11.673
##   rep  3.000  3.000  3.000  3.000  3.000
```

```r
# Media dos blocos para fazer os testes de Normalidade e Homocedasticidade
media_blocos_5 = aggregate(Area_foliar_M2 ~ EFLUENTE + INOCULO + CICLO,
                                 data = blocos_dados_5, FUN = mean)

# Realizar o teste Shapiro-Wilk no conjunto de dados completo
resultado_shapiro <- shapiro.test(media_blocos_5$Area_foliar_M2)$p.value

# Exibir o resultado do teste Shapiro-Wilk
print(resultado_shapiro)
```

```
## [1] 0.06302661
```

```r
# Interpretação do p-valor
if (resultado_shapiro > 0.05) {
  print("Os dados seguem uma distribuição normal.")
```

```r
} else {
  print("Os dados não seguem uma distribuição normal.")
}
```

```
## [1] "Os dados seguem uma distribuição normal."
```

```r
# Realizar o teste de Bartlett no conjunto de dados completo
resultado_bartlett <- bartlett.test(media_blocos_5$Area_foliar_M2,
                                    media_blocos_5$EFLUENTE,
                                    media_blocos_5$INOCULO,
                                    media_blocos_5$CICLO)$p.value

# Exibir o resultado do teste de Bartlett
print(resultado_bartlett)
```

```
## [1] 0.2479949
```

```r
# Interpretação do p-valor
if (resultado_bartlett > 0.05) {
  print("A variância é homogênea entre os grupos.")
} else {
  print("A variância não é homogênea entre os grupos.")
}
```

```
## [1] "A variância é homogênea entre os grupos."
```

```r
# Remover o bloco 4, pois a análise é muito influenciada pela presença deste
# no ciclo 1 e ausência no ciclo 2
blocos_dados_5 = subset(blocos_dados_5, BLOCO != 4)
```

```r
# Ajustar o modelo linear para parcelas subdivididas
modelo_PARCELASUB = aov(Area_foliar_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE +
                          Error(BLOCO*INOCULO), data = blocos_dados_5)

# Visualizar os resultados do modelo
summary(modelo_PARCELASUB)
```

```
##
## Error: BLOCO
##       Df Sum Sq Mean Sq
## BLOCO  2  2.512   1.256
##
## Error: INOCULO
##          Df   Sum Sq  Mean Sq
## INOCULO   1 0.002282 0.002282
##
## Error: BLOCO:INOCULO
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  2  35.52   17.76
##
## Error: Within
```

```
##                          Df Sum Sq Mean Sq F value   Pr(>F)
## CICLO                     1  95.89   95.89  16.913 0.000217 ***
## EFLUENTE                  4  30.56    7.64   1.347 0.271408
## CICLO:INOCULO             1  10.95   10.95   1.931 0.173168
## CICLO:EFLUENTE            4  16.68    4.17   0.736 0.573748
## INOCULO:EFLUENTE          4  12.79    3.20   0.564 0.690439
## CICLO:INOCULO:EFLUENTE    4  29.98    7.49   1.322 0.280486
## Residuals                36 204.10    5.67
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
modelo = modelo_PARCELASUB
```

```
# Refazer o modelo sem o erro, apenas para facilitar as funções subsequentes
modelo = aov(Area_foliar_M2 ~ BLOCO + CICLO * INOCULO * EFLUENTE,
             data = blocos_dados_5)

# Realizar a análise de variância (ANOVA)
anova_result <- anova(modelo)

# Filtrar apenas os fatores significativos ao nível 0.05
fatores_significativos <- anova_result[anova_result$"Pr(>F)" < 0.05, ]

# Exibir as interações significativas
print(fatores_significativos)
```

```
## Analysis of Variance Table
##
## Response: Area_foliar_M2
##       Df Sum Sq Mean Sq F value  Pr(>F)
## CICLO  1 95.887  95.887  15.206 0.00038 ***
## NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Vetor com os fatores significativos
nomes_linhas = row.names(fatores_significativos)

# Filtrar apenas as interações
interacoes_significativas <- nomes_linhas[nchar(nomes_linhas) > 8]

# Exibir o resultado
print(interacoes_significativas)
```

```
## character(0)
```

```
# Realizar o teste de Tukey para todas as interações significativas
tukey_result <- TukeyHSD(modelo)
```

```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
```

```
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- partes[[1]][3]
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == 0){
    media_interacao <- aggregate(Area_foliar_M2 ~ CICLO + INOCULO,
                                 data = blocos_dados_5, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Area_foliar_M2 ~ CICLO + EFLUENTE,
                                 data = blocos_dados_5, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "INOCULO" && fator2 == "EFLUENTE" && fator3 == 0){
    media_interacao <- aggregate(Area_foliar_M2 ~ INOCULO + EFLUENTE,
                                 data = blocos_dados_5, FUN = mean)
    print(media_interacao)
  }

  if(fator1 == "CICLO" && fator2 == "INOCULO" && fator3 == "EFLUENTE"){
    media_interacao <- aggregate(Area_foliar_M2 ~ CICLO + INOCULO + EFLUENTE,
                                 data = blocos_dados_5, FUN = mean)
    print(media_interacao)
  }
}
```

```
for (i in 1:length(interacoes_significativas)) {
  if (length(interacoes_significativas) == 0){
    break
  }
  interacao = interacoes_significativas[i]
  partes <- strsplit(interacao, split = ":")
  if (nchar(interacao) < 20){
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
    fator3 <- 0
  }
  else{
    fator1 <- partes[[1]][1]
    fator2 <- partes[[1]][2]
```

```
    fator3 <- partes[[1]][3]
  }

  inter_tukey = tukey_result[interacao]
  inter_tukey = data.frame(inter_tukey)
  colnames(inter_tukey) = c('diif','lwr','upr','p_adj')
  inter_tukey = subset(inter_tukey, p_adj < 0.05)
  print(inter_tukey)
}
```

```
# Juntar dados em um mesmo dataframe
dados_final = cbind(blocos_dados_1, blocos_dados_2["Massa_fre_aerea_M2"],
                    blocos_dados_3["Massa_sec_aerea_M2"],
                    blocos_dados_4["Massa_fre_raiz_M2"],
                    blocos_dados_5["Area_foliar_M2"])


# Criar planilha com todos os dados atualizados
library("xlsx")
```

```
## Warning: package 'xlsx' was built under R version 4.3.1
```

```
write.xlsx(dados_final, file = "Biometricas atualizado.xlsx",
      sheetName = "R - Biometricas", append = FALSE)
```