

Rapport de stage partiel

**Mise en place d'une application web
surveillant la croissance des coraux
en mésocosmes.**

Jordan Benrezkallah

Maître de stage : Philippe Grosjean

Encadrant de stage : Guyliann Engels

Promoteur : Aline Leonet et David Coornaert

HEH - Campus technique - Bloc 3 du cursus Bachelier en Biotechnique

Année académique 2018-2019

Table des matières

1 Présentation de l'entreprise	3
2 Présentation de l'équipe	5
2.1 Philippe Grosjean	5
2.2 Guyliann Engels	5
2.3 Antoine Batigny	6
2.4 Rémy Dugauquier	6
2.5 Madeleine Gille	7
3 But	8
3.1 Outils monitorings	8
3.1.1 Masse immergée et masse squelettique	8
3.1.2 Tableur	9
3.1.3 Application Shiny	9
3.2 Outils utilisés	12
3.3 Objectifs réalisés	12
3.4 Planning de travail	12
4 Communications interpersonnelles	13
4.1 Difficultés rencontrées	13
4.1.1 R vs python	13
4.1.2 Shiny communication entre ui.R et server.R	13
4.1.3 Apport au sein de l'entreprise	13
5 Annexe	14

Chapitre 1

Présentation de l'entreprise

Mon stage de fin d'études, se déroule dans à l'université de l'UMons dans le service d'Écologie Numérique des Milieux Aquatiques (abrégué en EcoNum) du département de Biologie.

L'Université de Mons (UMONS), est une université francophone implantée en Belgique, dans la province du Hainaut. Elle est constituée de 2 Ecoles et de 7 Facultés, dont la faculté des Sciences.

Le Département de biologie de la faculté des Sciences est impliqué dans la formation des étudiants et dans la recherche.

Le service d'Écologie Numérique des Milieux Aquatiques travaille dans l'enseignement des biostatistiques et de l'écologie aquatique à l'Université de Mons.

Le laboratoire EcoNum étudie les systèmes biologiques aquatiques complexes, tels les communautés planctoniques et les récifs coralliens, face aux changements de leur environnement.

Le laboratoire développe aussi des outils de traitement statistique, y compris dans le domaine du data mining, des big data, et de la recherche reproductible. Il participe à des études sur les logiciels Open Source et développe des chémostats.

Le laboratoire d'EcoNum est situé sur le campus de la plaine de Nimy (A) (voir figure 1.1) dans le pentagone (1) (voir figure 1.2).



FIGURE 1.1 – Carte de la ville de Mons

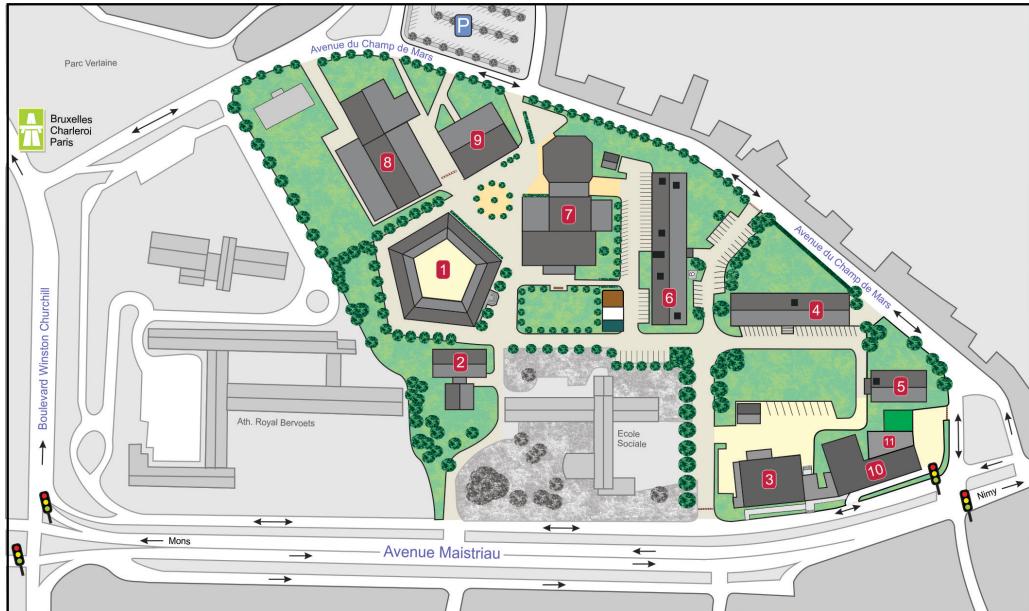


FIGURE 1.2 – Carte du campus de la plaine de Nimy

Chapitre 2

Présentation de l'équipe

2.1 Philippe Grosjean



FIGURE 2.1 – Monsieur Philippe Grosjean, chef du service d'EcoNum.

Mon maître de stage est Monsieur Philippe Grosjean, il enseigne les biostatistiques, l'écologie aquatique, l'écophysiologie et l'océanographie générale aux étudiants biologistes.

Il travaille également dans le domaine de la recherche sur plusieurs projets, dont l'identification automatique du plancton par photographie, des algorithmes de *machine learning*, développement de logiciel pour l'écologie.

Il développe des outils Open Source comme la *SciViews Box*, qui est une machine virtuelle contenant une suite de logiciel pré-configuré pour l'utilisation de ses étudiants.

Il encadre 1 doctorant et 2 étudiants en masters.

2.2 Guyliann Engels

Guyliann Engels est chercheur et professeur-assistant, il effectue sa thèse sur l'écophysiologie du corail, où il utilise un mésocosme pour étudier les stress des coraux engendrés par la modification de leurs nutriments essentiels. Il utilise fréquemment les outils de statistiques R et RStudio (avec R Markdown, R Notebook).

Il encadre mon travail.



FIGURE 2.2 – Guyliann Engels, doctorant encadrant le stage.

2.3 Antoine Batigny



FIGURE 2.3 – Technicien du service d’EcoNum.

Antoine Batigny est technicien, c'est lui qui s'occupe de gérer les mésocosmes et de l'auto-analyseur.

2.4 Rémy Dugauquier



FIGURE 2.4 – Etudiant en master

Rémy Dugauquier est en dernière année de master et fait son TFE sur le plancton. à développer

2.5 Madeleine Gille



FIGURE 2.5 – Etudiante en master

Madeleine Gille est en dernière année de master et fait son TFE sur le corail. **à déve- lopper**

Chapitre 3

But

Le but du stage est de créer une application Shiny, qui suit l'évolution des coraux situés dans les mésocosmes. Les coraux seront utilisés dans des expériences par le laboratoire, il est donc nécessaire de visualiser leur croissance. L'application doit pouvoir être utilisée facilement par d'autres personnes à *posteriori*, il faut donc l'automatiser et anticiper les problèmes à venir.

Le stage se déroule en 2 parties, la première est une phase d'apprentissage, la deuxième est la création de l'application et l'implémentation d'outils pour le monitoring de la croissance des coraux.

La phase d'apprentissage comprend :

- Apprentissage du langage de programmation R, de ses paquets et de l'environnement RStudio.

La phase de création d'outils comprend :

- Un relevé régulier de la croissance des coraux.
- La réalisation d'une application web Shiny, surveillant la croissance (monitoring) des coraux de l'espèce *S. hystrix*.

3.1 Outils monitorings

3.1.1 Masse immergée et masse squelettique

Pour évaluer la croissance des boutures de coraux, on utilise la masse squelettique. Pour l'obtenir sans détruire le corail, on mesure la masse immergée du corail dans l'eau de mer avec une balance munie d'un crochet. Après avoir mesuré la température et la salinité on peut convertir la masse immergée en masse squelettique à l'aide de la formule ci-dessous :

$$m_{\text{squelettique}} = \frac{m_{\text{immerge}}}{\frac{1 - \rho_{\text{eau}}}{\rho_{\text{squelettique}}}} \quad (3.1)$$

ρ_{eau} est déterminé via l'équation d'état de l'eau de mer grâce à la mesure de la salinité et de la température. Le $\rho_{\text{squelettique}}$ est la densité de l'aragonite(CaCO₃) du squelette du corail.

3.1.2 Tableur

Pour fonctionner, l'application doit recevoir un tableau de donnée. Pour l'instant, j'utilise ma licence d'Excel d'office 365 fourni par la HEH. Par la suite, j'aimerai utiliser un tableau en ligne afin que n'importe qui, qui a besoin de remplir un tableau de donnée puisse le faire depuis n'importe quelle machine connectée à internet.

Afin d'éviter au maximum des erreurs d'encodages, j'ai utilisé des règles pour mettre en évidence les cases non remplies, formater le type des cellules et mettre un dégradé de couleur suivant l'avancement des données.

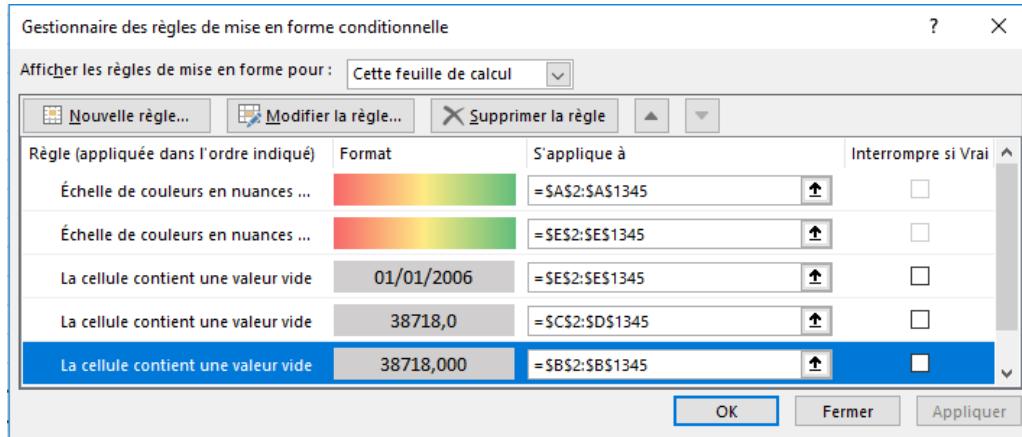


FIGURE 3.1 – Mise en forme conditionnelle d'Excel

Le tableau de donnée contient 7 colonnes :

- ID : corresponds à l'identifiant de la bouture.
- weight : corresponds à la masse immergée de la bouture.
- temp : corresponds à la température de l'eau de mer.
- salinity : corresponds à la salinité de l'eau de mer
- date : corresponds à la date et heure du relevé.
- commentaire : donne quelques annotations.

3.1.3 Application Shiny

L'application est divisée en deux éléments, une partie “ui” (User Interface), c'est la partie qui affiche les éléments graphiques de l'interface Shiny à l'utilisateur, et une partie “server”, qui contient toutes les commandes R qui s'opère côté serveur.

Il est possible mettre l'intégralité du code dans un seul fichier app.R, mais pour plus de clarté j'ai divisé mon script en deux fichiers ui.R et server.R (voir page annexe).

Mon application présente 2 onglets, le premier créer un graphique interactif.

Par défaut, le graphique montre l'évolution de la masse squelettique en fonction du temps.

On peut sélectionner le taux de croissance en fonction du temps.

Il est possible de sélectionner les ID dans un menu déroulant ou de directement cliquer à droite du graphique sur les ID triés par couleur.

ID	weight	temp	salinity	date	Commentaire
1	0,415	25,1	35,1	11/2/19 14:20	
2	0,286	25,1	35,1	11/2/19 14:20	
3		25,1	35,1	11/2/19 14:20	14h20
4	1,059	25,1	35,1	11/2/19 14:20	
5	0,677	25,1	35,1	11/2/19 14:20	
6	0,394	25,1	35,1	11/2/19 14:20	
7	0,795	25,1	35,1	11/2/19 14:20	
8	0,228	25,1	35,1	11/2/19 14:20	
9	0,508	25,1	35,1	11/2/19 14:20	
10	0,929	25,1	35,1	11/2/19 14:20	
11	0,519	25,1	35,1	11/2/19 14:20	
12	1,088	25,1	35,1	11/2/19 14:20	
13	0,603	25,1	35,1	11/2/19 14:20	
14	0,224	25,1	35,1	11/2/19 14:20	
15		25,1	35,1	11/2/19 14:20	
16		25,1	35,1	11/2/19 14:20	numéro 16 à rejeter, mal mesuré
17	0,465	25,1	35,1	11/2/19 14:20	

FIGURE 3.2 – Tableau de donnée

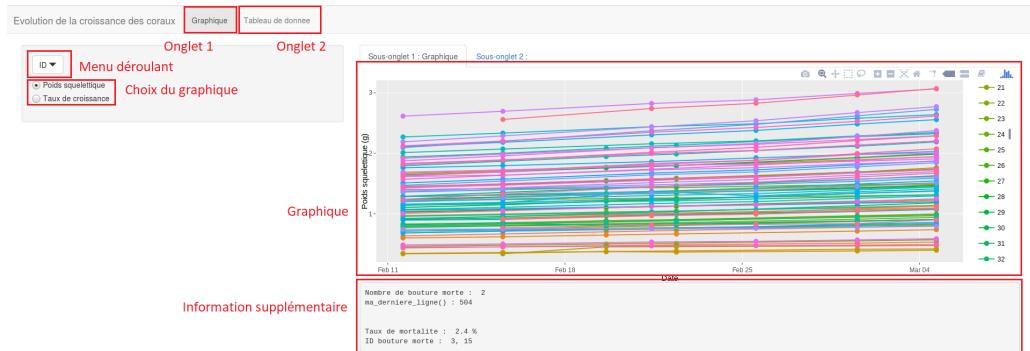


FIGURE 3.3 – Application Shiny : légende

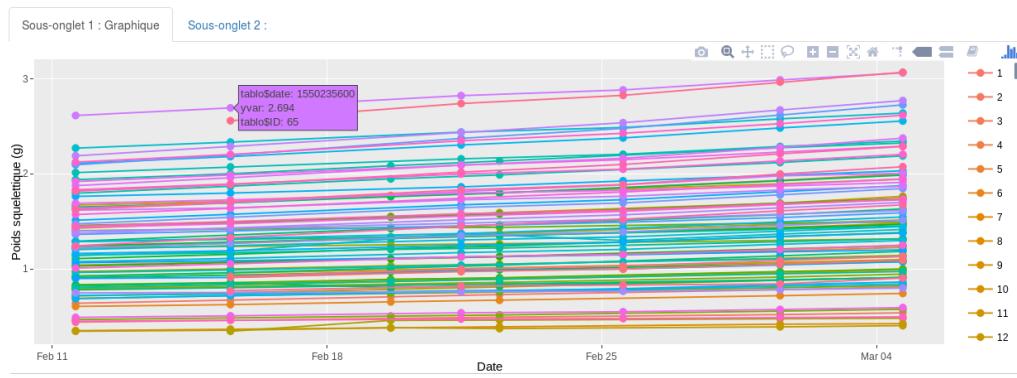


FIGURE 3.4 – Application Shiny : masse squelettique

Le menu déroulant permet de tout sélectionner ou de tout désélectionner.

En passant le curseur sur les points du graphique, on peut obtenir quelques informations. Sous le graphique, des informations supplémentaires : le nombre de boutures mortes, leur

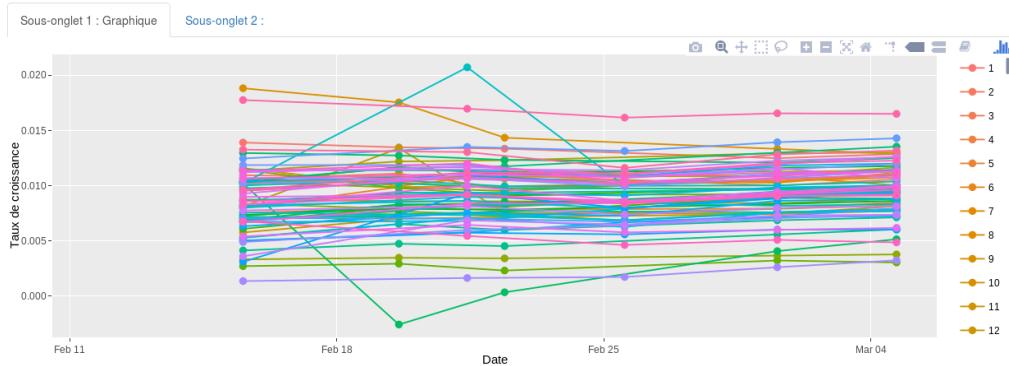


FIGURE 3.5 – Application Shiny : taux de croissance

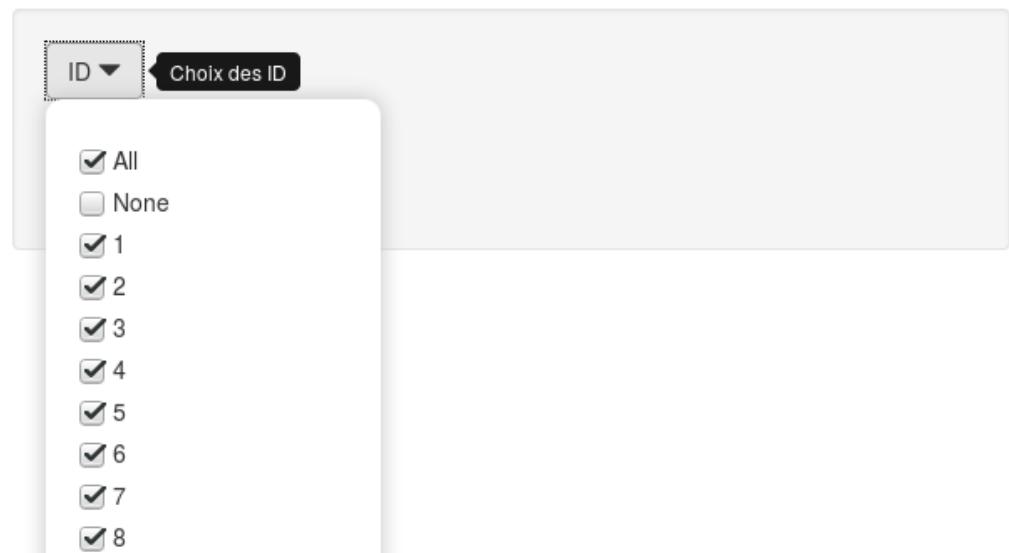


FIGURE 3.6 – Application Shiny : menu déroulant

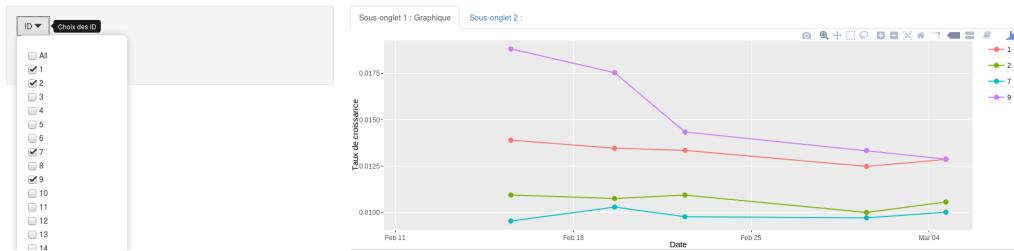


FIGURE 3.7 – Application Shiny : affichage interactif

ID et le taux de mortalité sont calculés.

Le deuxième onglet contient le tableau de donnée où de nouvelles colonnes ont été calculées, il y a l'ajout de la masse squelettique et du “ratio” qui correspond au taux de croissance.

```

Nombre de bouture morte : 2
ma_derniere_ligne() : 504

Taux de mortalite : 2.4 %
ID bouture morte : 3, 15

```

FIGURE 3.8 – Application Shiny : informations supplémentaires

Show 10 entries								Search:
	ID	weight	temp	salinity	date	skw	ratio	
1	1	0.415	25.1	35.1	2019-02-11T14:20:00Z	0.638		
2	2	0.286	25.1	35.1	2019-02-11T14:20:00Z	0.44		
3	3		25.1	35.1	2019-02-11T14:20:00Z			
4	4	1.059	25.1	35.1	2019-02-11T14:20:00Z	1.627		
5	5	0.677	25.1	35.1	2019-02-11T14:20:00Z	1.04		
6	6	0.394	25.1	35.1	2019-02-11T14:20:00Z	0.605		
7	7	0.795	25.1	35.1	2019-02-11T14:20:00Z	1.222		
8	8	0.228	25.1	35.1	2019-02-11T14:20:00Z	0.35		
9	9	0.508	25.1	35.1	2019-02-11T14:20:00Z	0.781		
10	10	0.929	25.1	35.1	2019-02-11T14:20:00Z	1.428		

Showing 1 to 10 of 504 entries

Previous 1 2 3 4 5 ... 51 Next

FIGURE 3.9 – Application Shiny : tableau de donnée

3.2 Outils utilisés

Les outils utilisés sont :

- La machine virtuelle *SciViews Box*, contenant un linux (Xubuntu), R, RStudio et les paquets nécessaires pré-installés.
- Les langages de programmation : R et R Markdown (LatTex).
- Les paquets : Shiny, tidyverse, ggplot2, dyplr, plotly, googlesheets, ect.

3.3 Objectifs réalisés

Les objectifs réalisés sont :

- Bouturer les coraux et relever leurs masses immergées.
- Créer un tableau Excel contenant les données nécessaires.
- Créer un prototype d’application web à l’aide du paquet Shiny.

3.4 Planning de travail

Les horaires de stages sont flexibles, on peut arriver entre 7 et 9 heure et il faut prester au moins 8 heures par jour.

Chapitre 4

Communications interpersonnelles

4.1 Difficultés rencontrées

4.1.1 R vs python

La principale difficulté rencontrée au début est de passer de l'apprentissage du langage de programmation *python* à *R*. Ce sont tous les deux des langages de programmation interprétés qui ont leur façon de faire. *python* a été créé pour faire de la programmation informatique généraliste, il est utilisé dans de larges domaines par des informaticiens. Tandis que, *R* est dédié aux analyses statistiques, plutôt utilisées par des spécialistes ou des scientifiques.

Dans le domaine du *data scientist*, *R* et *python* se valent, chacun ayant ses avantages et inconvénients. C'est dans leur syntaxe et leur philosophie qu'ils divergent fortement.

4.1.2 Shiny communication entre ui.R et server.R

Le schéma de communication basique entre les deux scripts commence par la déclaration d'une variable *inputId = ma_variable* dans ui.R. Celui-ci est ensuite appelé dans server.R sous la forme *input\$ma_variable*, cette variable sera ensuite traitée dans un bloc de code délimiter par des crochets.

Le souci vient du fait que Shiny utilise du Javascript pour dynamiser l'interface de l'utilisateur sous une couche de code masqué, cette couche simplifie le travail avec R. Si on sort du cadre de l'utilisation prévu par Shiny, on se heurte à de grands soucis de codage. Shiny restreint donc, la communication entre les différents blocs de code. Dans certaines situations cela peut lourdement compliquer le travail.

4.1.3 Apport au sein de l'entreprise

...

Chapitre 5

Annexe

```
library(shiny)
library(shinyWidgets)
library(DT)
library(plotly)
library(shinythemes)

shinyUI(
  navbarPage(
    # theme = shinytheme("slate"),
    title = "Evolution de la croissance des coraux", # Titre onglet 1
    tabPanel("Graphique", # Onglet principal 1
      # Sidebar : volet de gauche - Input
      sidebarPanel(
        uiOutput(outputId = "ID"),# Selection des ID a afficher
        uiOutput(outputId = "Ratio")
      ),
      # mainPanel : Volet de droite - Output
      mainPanel(
        tabsetPanel( # Sous-onglet
          tabPanel("Sous-onglet 1 : Graphique",
            plotlyOutput(outputId = "monplot"),
            #sortie console
            verbatimTextOutput(outputId = "boutures_mortes")),
          tabPanel("Sous-onglet 2 : ")
        )
      )
    ),
    tabPanel("Tableau de donnee", # Onglet principal 2
      # sidebar : volet de gauche - Input
      sidebarPanel(
      ),
      # mainPanel : Volet de droite - Output
      mainPanel(
        tabsetPanel(
```

```

        tabPanel("Beau tableau", DT::dataTableOutput("tableau"))
    )
)
)
)

# / #
# / #
# , /.
# , \ / .
# , ' . V. ` .
# / . . \ \
# /` . ' _ \
# , ' : ;, ` .
# | @ | . . | @ |
# , - . - ' ; . : - , ' - , - .
# ' -- - - \ / , - - - - . \ / - ' - - - -
# (---- _ / / / _ / / _ / ----)
# ` . _ , - ' \ ` - . - ' / ` - . _ , '
# ` - . - - - , - ' #

#####
# Application Shiny
# creer un graphique et un tableau a partir d'un fichier .csv
# les valeurs manquantes "NA" sont detecter comme etant des boutures mortes.
#
# Pour utiliser correctement l'application,
# il est important de respecter la syntaxe des noms des colonnes qui sont :
# | ID | weight | temp | salinity | date |
#
# Le format de la date doit etre de type :
# dd/mm/yyyy
#
# Il est egalement necessaire de commenter les lignes :
#cp_tableo[81:84, 2] <- "oublie"
#cp_tableo[16, 2] <- "a rejeter"
#botableo[81:84, 2] <- "oublie"
#botableo[16, 2] <- "a rejeter"
#
# Ces lignes sont specifiques a mon jeux de donnees
#####
#
# Titre : Croissance des coraux
# Auteur : Jordan Benrezkallah
# Date debut : 04/03/2019
# Date fin : 06/05/2019
#
#####

# Importation des librairies :

```

```

library(shiny)
library(ggplot2)
library(lubridate)
library(tidyverse)
library(dplyr)
library(plotly)
library(googlesheets)
SciViews::R
# library(scales)

# #Fonction de Raphael :
# source(file = "../R/fonctions.R")
# #Mes fonctions
# source(file = "../R/fonction.R")

# Importation de mes donnees (format csv)
#correction a faire : chemin relatif
#tablo <- gdata::read.xls("~/shared/Github/coral_growth001/data/raw/monBordel/tablo.xlsx")

tablo <- read.table("~/shared/Github/coral_growth001/data/my_data/tablogs.csv", header = TRUE, s

# GOOGLE SHEETS#
# tablo <- gs_title("tablo")
# tablo <- gs_read(tablo)
# tablo

# Determination du nombre de ligne de tableau a utiliser
# !\ Baser sur la premiere valeur NA rencontre dans la colonne "temp" !\
# Fonction a ameliorer de facon a ne garder seulement les lignes completes (ID, weight, temp, s

ma_derniere_ligne <- function(){
  a <- 0
  for (i in tablo$temp) {
    if (!is.na(i)) {
      a <- a + 1
    }
  }
  return(a)
}

# Extraction des 5 colonnes (id, weight, temp, salinity et date) jusqu'a la derniere
# ligne de la colonne "temp" du fichier .csv
tablo <- tablo[1:ma_derniere_ligne(), 1:5]

### Calcul du poids squelettique :
#a corriger : rho_aragonite
#P = Pression hydrostatique, elle vaut 0 a la surface
skeleton_weight <- function(S = tablo$salinity, T = tablo$temp, P = 0,
                           buoyant_weight = tablo$weight, rho_aragonite = 2930){


```

```

rho_water <- seacarb::rho(S = S, T = T , P = P)
skl_wgt <- buoyant_weight / (1 - (rho_water / rho_aragonite))
skl_wgt <- round(skl_wgt, digits = 3)
return(skl_wgt)
}

#Ajout de la colonne du poids squelettique
tablo <- mutate(tablo, skw = skeleton_weight())

# changer le type de l'ID de "int" a "factor"
tablo$ID <- factor(tablo$ID)

#changer le type (mode) de la date
tablo$date <- dmy_hm(tablo$date)

#parse_date_time(tablo$date, locale = locale("fr"), orders = "dmy HMS")
tablo$date <- as_datetime(tablo$date)

# arrondir la datetime a l'heure pres
# tablo$date <- round_date(tablo$date, "hour")

# Nombre de ID different
nbr_ID <- unique(tablo$ID)

#Je fais une copie pour pouvoir travailler dessus sans creer de probleme d'affichage
cp_tablo <- tablo
botablo <- tablo

# affiche dans le tablo a presenter
botablo[81:84, 2] <- "oublie"

#la valeur de la bouture 16 est a rejeter
botablo[16, 2] <- "a rejeter"

#Remplace les valeurs manquantes par "Bouture morte"
botablo[is.na(botablo)] <- "Bouture morte"

#Tableau a afficher sur l'app Shiny :
botablo <- transmute(botablo,
                       ID = botablo$ID,
                       "Masse immerge (g)" = botablo$weight,
                       "Masse squelettique (g)" = skeleton_weight(),
                       "Temperature (c)" = botablo$temp,
                       "Salinité (g/L)" = botablo$salinity,
                       Date = botablo$date)

# Taux de croissance
tablo %>%
  group_by(., ID) %>%
  arrange(., date) %>%
  mutate(., delta_date = difftime(date, date[1], units = "days" ),

```

```

ratio = (skw - skw[1]) / skw[1] / as.double(delta_date)) -> tablo1
# a cause du group_by je ne peux pas modifier directement "tablo"
tablo <- mutate(tablo, ratio = tablo1$ratio)

#tablo$ratio[is.nan(tablo$ratio)] <- "HOHOH"

#####
##### ----- Partie logique du serveur ----- #####
shinyServer(function(input, output, session) {

# -----Selection des ID-----

# Recuperation de l'ID du fichier ui.R
output$ID <- renderUI({

#Menu deroulant
dropdown(
checkboxGroupInput(inputId = "choix_id", label = NULL,
choices = c("All", "None", nbr_ID), selected = c("All")),
width = "200px", size = "default", label = "ID",
tooltip = tooltipOptions(placement = "right", title = "Choix des ID")
)
})

#-----Choix taux de croissance-----
output$Ratio <- renderUI({
radioButtons(inputId = "choix_ratio", label = NULL,
choices = c("Masse squelettique", "Taux de croissance"),
selected = "Taux de croissance")
})

# -----Output de mon graphique-----
output$monplot <- renderPlotly{

#Filtrer les lignes par rapport a ce qui a ete selectionne
if ("All" %in% input$choix_id) {
updateCheckboxGroupInput(session, inputId = "choix_id", label = "select All",
choices = c("All", "None", nbr_ID), selected = c("All", nbr_ID)
)
}

if ("None" %in% input$choix_id) {
updateCheckboxGroupInput(session, inputId = "choix_id", label = "select All",
choices = c("All", nbr_ID), selected = NULL
)
}

else {
tablo <- filter(tablo, tablo$ID %in% input$choix_id)
vvar = tablo$skw
}
}
}

```

```

y_nom_axe <- "Masse squelettique (g)"
}

# Choix du taux de croissance
if ("Taux de croissance" %in% input$choix_ratio) {
  #tutu <- filter(tutu, tutu$ID %in% input$choix_id)
  yvar = tablo$ratio
  y_nom_axe <- "Taux de croissance"
}

# Tableau
p <- ggplot(tablo, aes(x = tablo$date, y = yvar, colour = tablo$ID)) +
  geom_point(size = 2, show.legend = FALSE) + geom_line(show.legend = F) +
  xlab("Date") + ylab(y_nom_axe)
## theme( axis.line = element_line(color = "darkgray", size = 2, linetype = "solid"))

#p + scale_x_date(labels = date_format("%d-%m-%y"))

#Pour remettre plotly, il faut changer : renderPlotly (server.R), plotlyOutput (ui.R) et de
p <- ggplotly(p)

### Legende qui ne fonctionne pas, probleme d'attribution...

# Legende lorsque l'on passe son curseur :
# ma_legende <- paste("ID :", factor_ID, "\n", "Poids :", tablo$weight, "\n", "Date :", mad
# pp <- ggplotly(p)
# pp <- style(pp, text = ma_legende, hoverinfo = "text")
})
#-----Sortie console-----#
output$boutures_mortes <- renderPrint({
  ### Cette partie sert a compter les boutures mortes

  #remplacer les weight de valeur NA des id 81 a 84 par "oublie"
  #cela va servir pour ne pas les compter dans les boutures mortes
  cp_tablo[81:84, 2] <- "oublie"

  #la valeur de la bouture 16 est a rejeter
  cp_tablo[16, 2] <- "a rejeter"

  #les 2 lignes ci-dessous empêche la visualisation du graphique si je ne met pas cp_tablo
  ID_NA <- subset(cp_tablo, is.na(weight) == TRUE, ID)
  ID_NA <- unique(ID_NA)
  ID_NA <- ID_NA$ID

  #nombre de boutures mortes :
  nbr_bouture_morte <- length(ID_NA)

  #Taux de mortalite :
  Taux_mort <- round((nbr_bouture_morte / length(as.numeric(unique(cp_tablo$ID)))) * 100, dig

  cat("Nombre de bouture morte : ", nbr_bouture_morte, "\nma_derniere_ligne() :",

```

```
    ma_derniere_ligne(), "\n", "\n", "\nTaux de mortalite : ",  
    Taux_mort, "%", "\nID bouture morte : ", paste(ID_NA, collapse = ", "))  
}  
  
# -----Tableau-----  
output$tableau <- DT::renderDataTable({DT::datatable(tablo)}  
}  
})
```