Respiration workflow example

Samuel Gurr

2023-02-28

```
# LOAD PACKAGES :::::::::::::::::::::::::::::::::::::::::::::::::::::

library(devtools) # devtools::install_github # use devtools to instlal github link
library(LoLinR) # install_github('colin-olito/LoLinR') # install LoLinR from github
library(dplyr)
library(lubridate)
library(rMR)
library(ggplot2)
library(stringr)

# SET WORKING DIRECTORY AND OUTPUT PATH ::::::::::::::::::::::::::::::::::::::::::::::
knitr::opts_knit$set(root.dir = 'C:/Users/samjg/Documents/Github_repositories/Airradians_multigen_OA/RA
path.p    <- "Data/Physiology/Respiration" #the location of all your respirometry files
```

STEP 1: OUTPUT A RAW PLOT FOR ALL DATA

A. call the directroy names to loop through files (though we will NOT loop in this excersize

```
folder.names           <- basename(list.files(path = path.p, pattern = "202", recursive = FALSE)) #list
folder.names.table     <- data.frame(folder.names) # you see we have many data directories! each with r
```

- lets focus on ONLY row 15: 20230223 - contains F2 adults meausred with LoLigo (.txt files)

```
folder.names.table    <- folder.names.table[15,] # 20230223 - we will call this directroy for this work
```

B. call the file names in the directory 2023023

```
file.names.table      <- data.frame(txt.files = # call a data frame with a list of all text files wihti
                                (basename(
                                  list.files(
                                    path = paste(path.p,'/',folder.names.table ,sep=''), # "Data/
                                    pattern = "txt$", recursive = TRUE))))  %>%  # list all text
                           dplyr::filter(grepl('raw', txt.files))  # only call the txt files with 'raw' i

file.names.table
```

```
##       txt.files
## 1 run_1_raw.txt
## 2 run_2_raw.txt
## 3 run_3_raw.txt
```

```
# take a look at the file  - we have three .txt files in 20230223 directroy
# txt.files
# 1 run_1_raw.txt
# 2 run_2_raw.txt
# 3 run_3_raw.txt
```

C. Loop through each file and plot!

- before getting started let's first lets open a single file to understand how the loop works!

```
folder.names.table # our existing directroy 20230223 where the three runs in txt format are stored
```

```
## [1] "20230223"
```

```
file.names.table[1,1] # this calls the first row of the first columns in file.names.table as "run_1_raw
```

```
## [1] "run_1_raw.txt"
```

```
# lets paste these parameters into our file name to call 'run_1_raw.txt'
Resp.Data  <- read.delim2(file =
                            paste(path.p,'/',folder.names.table, '/', file.names.table[1,1], sep=''), #
                          header = TRUE, # yes there is aheader here that we want to retain (column nam
                          skip = 37,  # the raw loligo data has undesired metadata in the first 37 rows
                          fileEncoding= "windows-1252") #reads in the data files correctly in their raw
# head(Resp.Data)
```

- now lets do this in a looped fashion (commented throughout!)
  - note: if you want to run this line by line do the following:

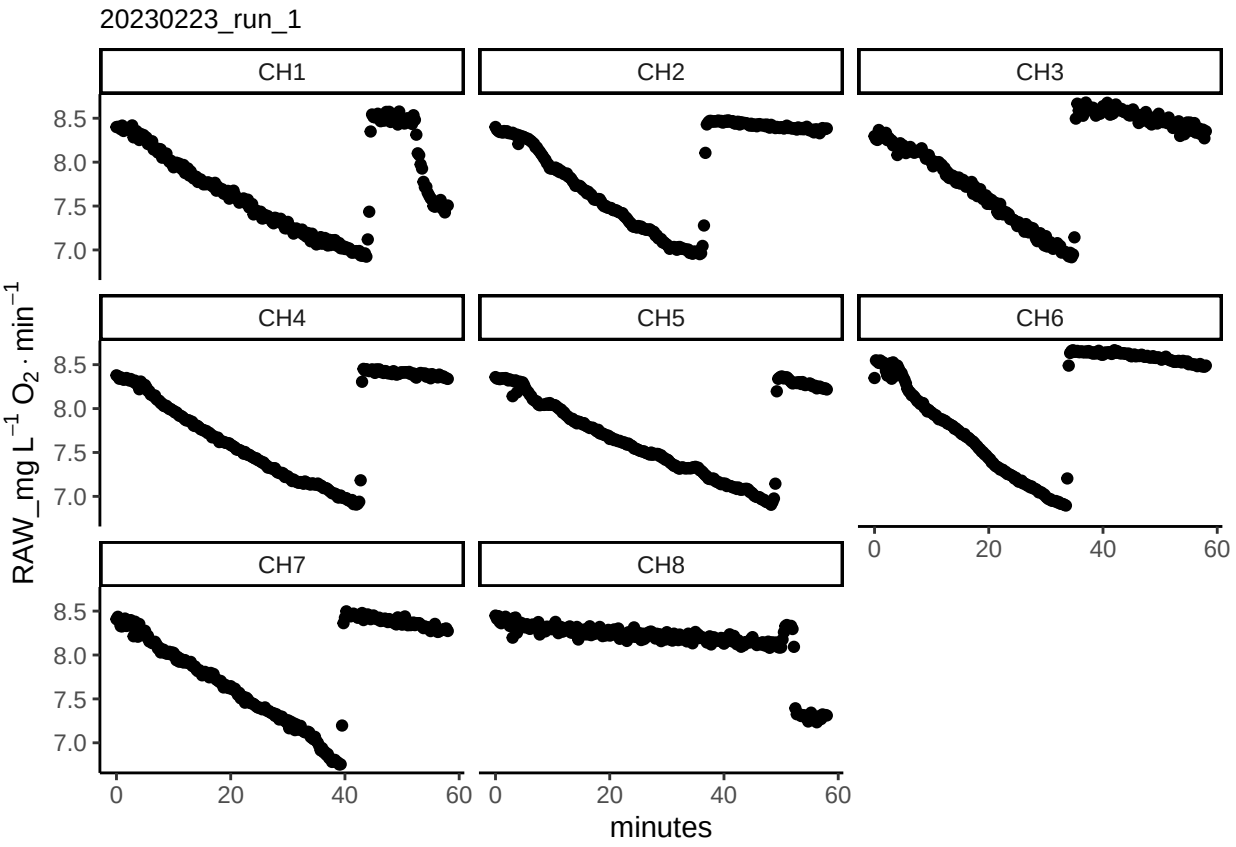
(1) do not run the 'for' line at the start - use Cntrl+Enter ro tun line by line

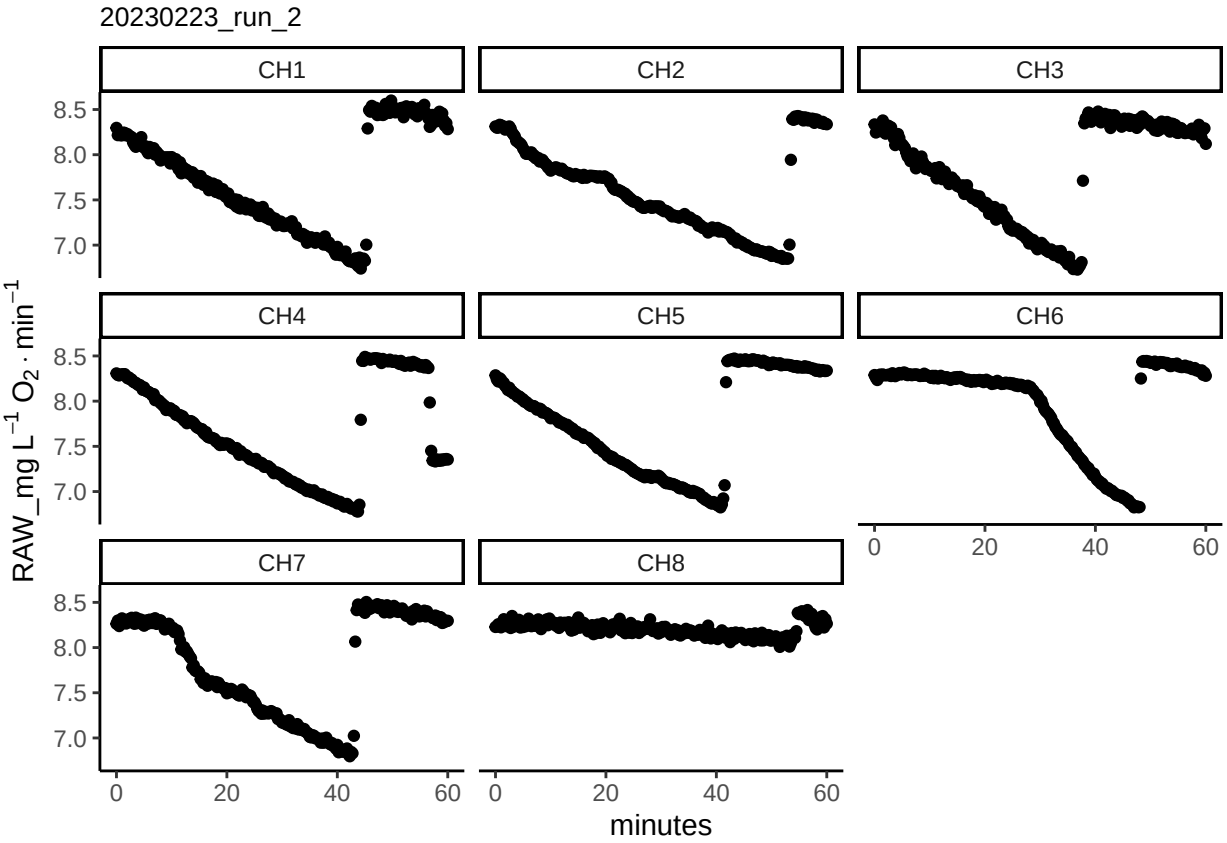
(2) change file.name to call [1,1], [2,1] OR [3,1] for the individual three files in 20230223 folder

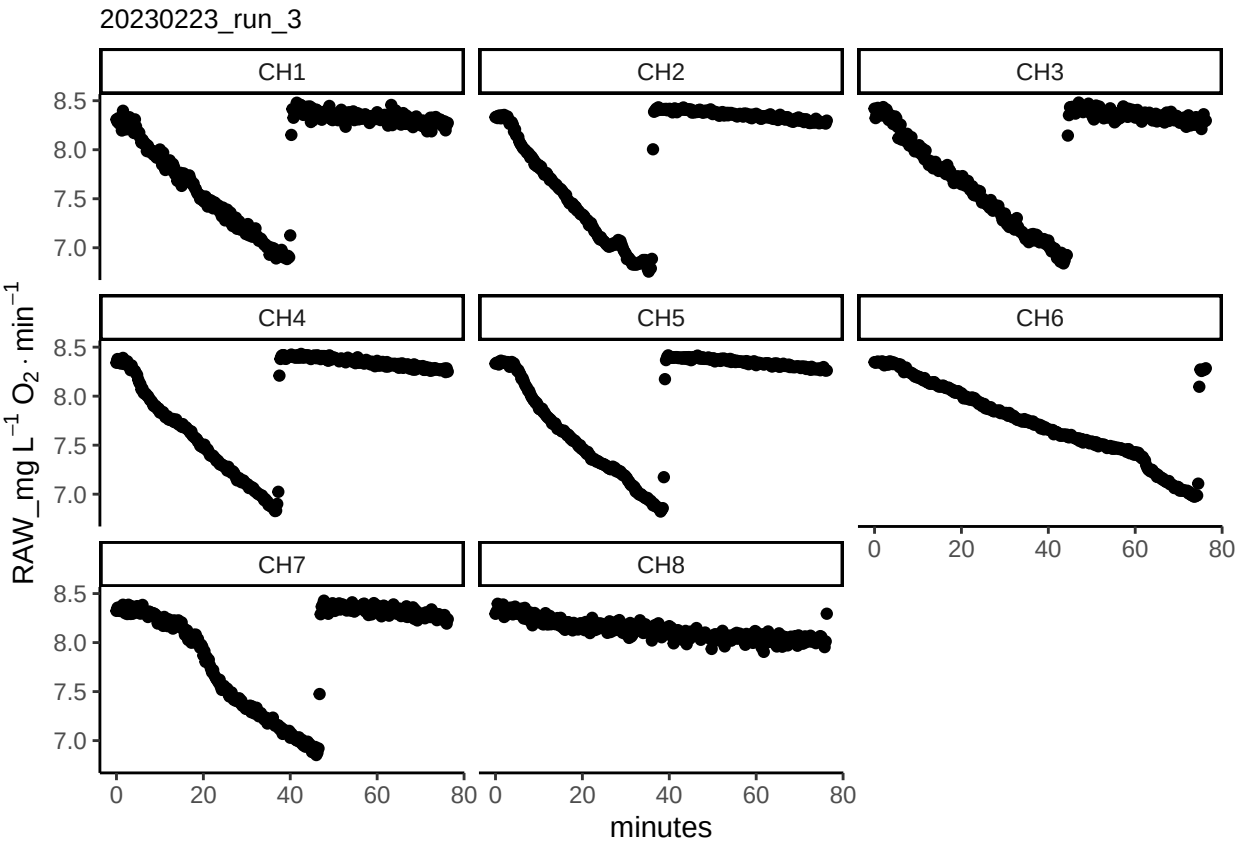
```
for(m in 1:nrow(file.names.table)) { # the following is ONLY structured for LiLigo data in txt format!

      file.name <- file.names.table[m,1] # CHANGE HERE TO file.names.table[1,1] IF YOU WNAT TO RUN LINE

      # read the file name one by one - runs the WHOLE loop with the first then the second and so on
      Resp.Data  <- read.delim2(file =
                                  paste(path.p,'/',folder.names.table, '/', file.name, sep=''),
                                header = TRUE,
                                skip = 37,
                                fileEncoding= "windows-1252") #reads in the data files one  by one firs
```

```r
# Data data
# reformat the raw date and call the seconds and minutes timestamp!
# raw format is "2/23/2023/11:52:07 AM" containing a lot of info but we need to parse and concvert

 Resp.Data$date      <- paste((sub("2023.*", "", Resp.Data$Date..Time..DD.MM.YYYY.HH.MM.SS.)), '20

 Resp.Data$time_Sec  <- period_to_seconds(hms(substr((strptime(sub(".*2023/", "", Resp.Data$Date..




# assign the remaining parameters
 Resp.Data$seconds   <- (Resp.Data$time_Sec - Resp.Data$time_Sec[1])    # seconds as a time series

 Resp.Data$minutes   <- (Resp.Data$time_Sec - Resp.Data$time_Sec[1])/60 # convert to minutes

 temperature_C       <- as.numeric(Resp.Data$CH1.temp...C.[1]) # call the temperature data - the d

 barromP_kPa         <- as.numeric(Resp.Data$Barometric.pressure..hPa.[1]) / 10 # call the baromet

 salinity.pp.thou    <- as.numeric(Resp.Data$Salinity....[1]) # call the salinity - again this was

 Resp.Data           <- Resp.Data %>% # use 'dplyr'
   #dplyr::filter(!Phase %in% 'Flush') %>% # remove the initial rows labeled flush
   dplyr::select(c(date, seconds, minutes, contains(".O2...air.sat")))  # all target oxygen conupt

 colnames(Resp.Data)[c(4:(ncol(Resp.Data)))] <- substr( ( colnames(Resp.Data)[c(4:(ncol(Resp.Data)]




 # We have data every second - this is comutationally intensize and redundant
 # truncate the data to every 15 sceonds
 Resp.Data_15sec = Resp.Data[seq(1, nrow(Resp.Data), 15), ]


 # lets plot!


 date.plot  <- folder.names.table # the directroy timestamp as 20230223

 run.plot   <- gsub("_raw.*","", file.name) # the looped run!

 plot_title <- paste(date.plot, run.plot, sep = '_') # the title merging these two parameters as d




 PLOT <- Resp.Data_15sec %>% # plote pipeline


   # before we plot we need to convert the data from air saturation to mg/L O2!
   dplyr::select(-c('date', 'seconds')) %>%  # select out the data we do not need- the date and se
```

```
    reshape2::melt(id.vars = "minutes",variable.name = "channel", value.name = "air.sat") %>% # mel
    dplyr::filter(!air.sat  %in% 'NaN') %>% # omit NAs in this data
    dplyr::mutate(mg.L.min =   (DO.unit.convert(as.numeric(air.sat),  # use teh presens package DO.
                                                DO.units.in = "pct", DO.units.out ="mg/L",
                                                bar.units.in = "kPa", bar.press = barromP_kPa, bar.
                                                temp.C = temperature_C,
                                                salinity.units = "pp.thou", salinity = salinity.pp.



    ggplot(aes(x = minutes , y = mg.L.min)) + # plot simple regression
    geom_smooth(method = "loess", se=FALSE, color="black", formula = mg.L.min ~ minutes) + # call a
    theme_classic() +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), legend.position =
    labs(y = expression(RAW_mg~L^{-1}~O[2]%.%min^{-1})) + # name y axis
    xlab("minutes") + # name the x axis
    geom_point() +
    ggtitle(plot_title) + # insert the title we called earlier based on the loop file name and date
    facet_wrap(~channel) # wrap by channel ID column - creates separte plots for each channel

  print(PLOT) # view!


  # D. output the plot in RAnalysis\Output\Respiration\workflow_example\plots_raw
  pdf(paste0("C:/Users/samjg/Documents/Github_repositories/Airradians_multigen_OA/RAnalysis/Output/
  print(PLOT)
  dev.off()

}
```


20230223_run_1
CH1
CH2
CH3
CH4
CH5
CH6
CH7
CH8
RAW_mg L−1 O2 · min−1
8.5
8.0
7.5
7.0
0
20
40
60
minutes


20230223_run_2
CH1
CH2
CH3
CH4
CH5
CH6
CH7
CH8
RAW_mg L−1 O2 · min−1
8.5
8.0
7.5
7.0
0
20
40
60
minutes


20230223_run_3
CH1
CH2
CH3
CH4
CH5
CH6
CH7
CH8
RAW_mg L−1 O2 · min−1
8.5
8.0
7.5
7.0
0
20
40
60
80
minutes