

# DataInBrief - Processing of photosynthesis data and calculation of ACi curves

Bernd Berauer

2025-03-18

**Note:** To execute this code it is necessary to adjust the file paths in here to match your own file organization.

## 1. Data preparation for ACi - Fit

Load necessary libraries for analysis.

```
library(tidyverse)
library(lubridate)
library(readLicorData)
library(plantecophys)
library(readxl)
library(ggpubr)
```

**1.1 Load & merge raw data** Create a list of files to read based on the LiCor raw data.

```
### create list of files to read create path to read the LiCor files from
photo.files <- grep(list.files(path = "./Manuscript/DataInBrief/Data/photosynthesis_raw/",
  full.names = T, recursive = T, include.dirs = F), pattern = ".xlsx", value = T,
  invert = T)
# create unique names for the LiCor files
photo.names <- gsub(".*(/[^\"]*)$", "\\1", photo.files) # regex expression for cut everything before the
```

Read in all LiCor files, bind to one Data frame and store on hard drive

```
### read in LiCor Data
df.licor.output <- NULL

for (i in 1:length(photo.files)) {
  df.tmp <- licorData(photo.files[i])

  if (nrow(df.tmp) > 0) {
    df.tmp <- df.tmp %>%
      select(., !starts_with(c("hhmmss", "time"))) %>%
      mutate(., ID = photo.names[i]) %>%
      separate(., ID, into = c("Date_Time", "climate_chamber", "meas", "pot_nr"),
        sep = "_", remove = F) %>%
      mutate(., across(.cols = everything(), .fns = ~as.character(.x))) %>%
```

```

mutate(., climate_chamber = str_replace_all(string = climate_chamber,
      pattern = "ch", replacement = "CC_")) %>%
select(., c(ID, Date_Time, climate_chamber, pot_nr, obs, elapsed, date,
      A, E, Ci, Ca, gtc, gsw, VPDleaf, RHcham, Qamb_out, CO2_r, CO2_s,
      Tleaf, Pci, everything()))

  # assign(paste('df', photo.names[i], sep = '_'), df.tmp)
df.licor.output <- bind_rows(df.licor.output, df.tmp)
} else {
  # There is no data in the file, we move on to the next iterator
}

}

# check if all 285 files were read in correct
length(unique(df.licor.output$ID)) # yes, all there

df.licor.output <- df.licor.output %>%
  arrange(., pot_nr, Date_Time)
# store the data on the hard drive
write.csv(df.licor.output, "./Manuscript/DataInBrief/Data/aci_AllMerged_18032025.csv",
  row.names = F)

```

**1.2 Inspect & clean raw data** Create graphs showing the CO2 ramp and the ACi curve of each measurement. This is used to visually assess quality of conducted measurements.

```

df.licor <- read.csv("./Manuscript/DataInBrief/Data/aci_AllMerged_18032025.csv")

tmp.ID <- df.licor %>%
  arrange(., pot_nr, Date_Time) %>%
  select(., ID) %>%
  distinct()

for (i in 1:nrow(tmp.ID)) {
  df.tmp <- df.licor %>%
    filter(., ID == tmp.ID$ID[i])

  plot.ramp <- ggplot(df.tmp, aes(x = obs, y = CO2_r)) + geom_point() + labs(x = "Observation",
    y = "CO2 reference [ppm]") + theme_bw()

  plot.aci <- ggplot(df.tmp, aes(x = Ci, y = Adyn)) + geom_point() + labs(x = "Ci [ppm]",
    y = "Assimilation") + theme_bw()

  plot.arrange <- ggarrange(plot.ramp, plot.aci)
  annotate_figure(plot.arrange, fig.lab = paste("# Pot", unique(df.tmp$pot_nr),
    sep = " "), fig.lab.face = "bold", top = text_grob(df.tmp$ID, color = "grey20",
    size = 14))

  tmp.filename <- paste("./Manuscript/DataInBrief/Data/output_graphs/aci_raw/",
    unique(df.tmp$ID), ".jpg", sep = "")

  ggsave(filename = tmp.filename, device = "jpg", units = "cm", height = 10, width = 15,

```

```

    dpi = 150)
}

```

Check if measurements show weird behaviour: - to many observations (72 are the result of the CO2-Ramp conducted) - to many measurements (2 are planned, one well watered and one drought condition measurement per pot)

```

# dataframe containing minimum information on measurements
df.unique.measurements <- df.licor %>%
  select(., ID, Date_Time, chamber, pot_nr) %>%
  distinct()

# check if each pot number occurs twice (so was measured well watered or under
# drought) / some pots occur only once -> Those were not measured.
df.nr.pots.measured <- df.licor %>%
  select(ID, pot_nr) %>%
  distinct() %>%
  count(pot_nr) %>%
  arrange(., desc(n))

# check number of observations per unique identifier / the vast majority has
# 72, some have more and only one has less
df.nr.obs <- df.licor %>%
  group_by(ID) %>%
  summarise(., n = n()) %>%
  arrange(., desc(n))

df.nr.obs %>%
  filter(n != 72) # 6 pots which does not have the planned number of observations! Those need manual

df.tomuchobs <- df.licor %>%
  filter(., ID %in% c(df.nr.obs$ID[df.nr.obs$n != 72]))

# Inspect data issues within each of the identified pots and check if solving
# them improves the curve
pot_35 <- df.tomuchobs %>%
  filter(., pot_nr == 35) # >!  
< Looks Good! No change/improvement by removing data # irregularities i
pot_47 <- df.tomuchobs %>%
  filter(., pot_nr == 47) #>% distinct(., date, .keep_all = T) #>% filter(., obs != c(10:17, 131,
pot_79 <- df.tomuchobs %>%
  filter(., pot_nr == 79) #>% distinct(., CO2_r, .keep_all = T) # >!  
< Looks CRAP! Ramp is Okay but
pot_83 <- df.tomuchobs %>%
  filter(., pot_nr == 83) #>% distinct(., date, .keep_all = T) %>% filter(., obs != 110) # >!  
< Look
pot_94 <- df.tomuchobs %>%
  filter(., pot_nr == 94) # >!  
< Looks Good! No change/improvement by removing data # Co2_r ramp looks
pot_144 <- df.tomuchobs %>%
  filter(., pot_nr == 144) # >!  
< Looks Good! No change/improvement by removing data # last observation

```

Based on the above procedure, visual assessment, and inspection of too many observations we recommend to remove ID: “2024-08-21-1013\_ch2\_pot\_79” from further analysis.

```
df.licor.clean <- df.licor %>%
  filter(., ID != "2024-08-21-1013_ch2_pot_79") # keep only the rows which do NOT have a note on rem

# Store on hard drive
write.csv(df.licor.clean, "./Manuscript/DataInBrief/Data/aci_cleaned_18032025.csv",
  row.names = F)
```

## 2. ACi Fit

Load cleaned data set (in case you start here)

```
df.licor <- read.csv("./Manuscript/DataInBrief/Data/aci_cleaned_18032025.csv")
```

**2.1 Fit ACi curves using the *plantecophys* package (created by Duursma et al)** We use the *default* fitting method without fitting *TPU* limitation. For more details please see package handbook.

```
# Bilinear without TPU
dat.aci.default <- fitacis(data = df.licor, # >!  
  group = "ID",  
  varnames = list(ALEAF = "Adyn", Tleaf = "Tleaf", Ci = "Ci", PPFD = "Qabs"),  
  Tcorrect = T,  
  fitmethod='default', # fits the data points way better then the "default" me  
  fitTPU = F, # this takes a lot of time for calculation  
  id = c("pot_nr", "Date_Time")  
)

# Extract coefficients
v.id <- names(dat.aci.default)

dat.coefficients.default <- NULL

for (i in 1:length(v.id)) {
  tmp.par <- dat.aci.default[[v.id[i]]]$pars

  t.tmp.par <- data.frame(t(tmp.par)) %>%  
    rownames_to_column(.) %>%  
    pivot_wider(.,  
      names_from = "rowname",  
      values_from = c(Vcmax, Jmax, Rd)) %>%  
    mutate(.,  
      ID = v.id[i])

  dat.coefficients.default <- bind_rows(dat.coefficients.default, t.tmp.par)
}

df.aci.default.clean <- dat.coefficients.default %>%  
  separate(., ID, into = c("Date_Time", "climate_chamber", "meas", "pot_nr"), sep = "_", remove = F) %>%  
  mutate(.,  
    climate_chamber = str_replace_all(string = climate_chamber, pattern = "ch", replacement = "CC_")  
    Date_Time = ymd_hm(Date_Time)) %>%  
  separate(.,
```

```

        Date_Time, into = c("date", "time"), sep = " ",
        remove = F) %>%
mutate(.,
        water = if_else(date <= "2024-08-19", "WellWatered", "Drought"),
        water = if_else(is.na(water), "WellWatered", water)) %>%
arrange(., pot_nr, date) %>%
select(., ID, pot_nr, climate_chamber, water, Date_Time, date, time, everything()) %>%
select(., !meas)

write.csv(df.aci.default.clean, file = "./Manuscript/DataInBrief/Data/aci_parameters_19032025.csv", row

```

**Note:** The files containing the merged raw photosynthesis data [*aci\_AllMerged\_18032025*] and the calculated parameters of the ACi curves [*aci\_parameters\_19032025*] created with this Code are contained within the Excel-Worksheet [*DataInBrief\_ExperimentalData\_18032025*] and named *photosynthesis\_merged* and *aci\_parameters*, respectively.