Climate Water Loss Experiment - CEWL Data Wrangling

Savannah Weaver

2021

Contents

Packages

```
`%nin%` = Negate(`%in%`)
if (!require("tidyverse")) install.packages("tidyverse")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.6     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("tidyverse") # workflow and plots
if (!require("UsingR")) install.packages("UsingR")
```

```
## Loading required package: UsingR
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Loading required package: HistData
```

```
## Loading required package: Hmisc
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following object is masked _by_ '.GlobalEnv':
##
##     %nin%
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
##
## Attaching package: 'UsingR'
```

```
## The following object is masked from 'package:survival':
##
##     cancer
```

```
library("UsingR")
if (!require("lme4")) install.packages("lme4")
```

```
## Loading required package: lme4
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library("lme4") # for LMMs
if (!require("lmerTest")) install.packages("lmerTest")
```

```
## Loading required package: lmerTest
```

```
##
## Attaching package: 'lmerTest'
```

```
## The following object is masked from 'package:lme4':
##
##     lmer
```

```
## The following object is masked from 'package:stats':
##
##     step
```

```
library("lmerTest") # for p-values
```

Background and Goals

This CEWL (cutaneous evaporative water loss) data was collected June - August using a handheld evaporimeter (BioX AquFlux) on adult male *Sceloporus occidentalis*. Measurements were taken on the dorsum in 5 technical replicates before and after 8 days in different climate treatments. In this R script, I bring all the data files into one dataframe, check the distribution of replicates, omit outliers, and average remaining replicates. The final values will be more precise and accurate estimates of the true CEWL, and those values will be used in the capture_analysis and experiment_analysis R script files. Please refer to **doi:** for the published scientific journal article and full details.

Load Data

1. Compile a list of the filenames I need to read-in.

```
# make a list of file names of all data to load in
filenames <- list.files(path = "data/CEWL", pattern = "\\.csv$")
```

2. Make a function that will read in the data from each csv, name and organize the data correctly.

```
read_CEWL_file <- function(filename) {

  dat <- read.csv(file.path("data/CEWL", filename), # load file
                  header = TRUE # each csv has headers
                  ) %>%
    # select only the relevant values
    dplyr::select(date = Date,
                  time = Time,
                  status = Status,
                  ID_rep_no = Comments,
                  CEWL_g_m2h = 'TEWL..g..m2h..',
                  msmt_temp_C = 'AmbT..C.',
                  msmt_RH_percent = 'AmbRH....'
                  ) %>%
    # extract individual_ID and replicate number
```

```
    dplyr::mutate(ID_rep_no = as.character(ID_rep_no),
                  individual_ID = as.numeric(substr(ID_rep_no, 1, 3)),
                  replicate_no = as.numeric(substr(ID_rep_no, 5, 5))
                  )

  # return the dataframe for that single csv file
  dat
}
```

3. Apply the function I made to all of the filenames I compiled, then put all of those dataframes into one dataframe. This will print warnings saying that header and col.names are different lengths, because the data has extra notes cols that we read-in, but get rid of.

4. Filter out failed measurements and properly format data classes.

```
# apply function to get data from all csvs
all_CEWL_data <- lapply(filenames, read_CEWL_file) %>%
  # paste all data files together into one df by row
  reduce(rbind) %>%
  # only use completed measurements
  dplyr::filter(status == "Normal") %>%
  # properly format data classes
  mutate(date_time = as.POSIXct(paste(date, time),
                                format = "%m/%d/%y %I:%M:%S %p"),
         date = as.Date(date,
                        format = "%m/%d/%y"),
         time = as.POSIXct(time,
                           format = "%I:%M:%S %p"),
         status = as.factor(status),
         individual_ID = as.factor(individual_ID),
         #replicate_no = as.factor(replicate_no)
         # don't make replicate a factor
         # that way I can easily add new levels later
         )
summary(all_CEWL_data)
```

```
##       date                 time                          status
##  Min.   :2021-06-16   Min.   :2021-11-03 09:23:23   Normal:1373
##  1st Qu.:2021-06-26   1st Qu.:2021-11-03 10:45:58
##  Median :2021-07-20   Median :2021-11-03 12:26:23
##  Mean   :2021-07-20   Mean   :2021-11-03 12:36:22
##  3rd Qu.:2021-08-08   3rd Qu.:2021-11-03 14:05:51
##  Max.   :2021-08-30   Max.   :2021-11-03 18:08:37
##
##   ID_rep_no           CEWL_g_m2h     msmt_temp_C    msmt_RH_percent
##  Length:1373        Min.   : 5.09   Min.   :24.70   Min.   :25.50
##  Class :character   1st Qu.:19.29   1st Qu.:26.20   1st Qu.:46.00
##  Mode  :character   Median :24.11   Median :26.70   Median :47.80
##                     Mean   :24.92   Mean   :26.73   Mean   :46.69
##                     3rd Qu.:28.43   3rd Qu.:27.10   3rd Qu.:50.50
##                     Max.   :81.42   Max.   :29.20   Max.   :56.80
##
##  individual_ID   replicate_no     date_time
##  237    :  15   Min.   :1.000   Min.   :2021-06-16 09:50:20
##  302    :  15   1st Qu.:2.000   1st Qu.:2021-06-26 14:03:08
```

```
##  206    :  11   Median :3.000   Median :2021-07-20 14:55:57
##  215    :  11   Mean   :2.991   Mean   :2021-07-21 11:44:58
##  201    :  10   3rd Qu.:4.000   3rd Qu.:2021-08-08 15:22:33
##  202    :  10   Max.   :5.000   Max.   :2021-08-30 11:32:07
##  (Other):1301
```

5. Load in and format the cloacal temperature measured at the time of CEWL measurement.

```
cloacal_temp_C <- read.csv("./data/c_temps.csv", # filename
                           na.strings=c("","NA") # fix empty cells
                           ) %>%
  # select variables of interest
  dplyr::select(date, time_c_temp,
                day,
                individual_ID,
                cloacal_temp_C) %>%
  # properly format data classes
  mutate(date_time = as.POSIXct(paste(date, time_c_temp),
                                format = "%m/%d/%y %H:%M"),
         date = as.Date(date, format = "%m/%d/%y"),
         time_c_temp = (as.POSIXct(time_c_temp, format = "%H:%M")),
         day = as.factor(day),
         individual_ID = as.factor(individual_ID),
         cloacal_temp_C = as.numeric(cloacal_temp_C)
         ) %>%
  # get rid of rows with missing data
  dplyr::filter(complete.cases(.))
summary(cloacal_temp_C)
```

```
##       date             time_c_temp                      day
##  Min.   :2021-06-16   Min.   :2021-11-03 09:26:00   capture :140
##  1st Qu.:2021-06-26   1st Qu.:2021-11-03 10:48:00   post-exp:135
##  Median :2021-07-20   Median :2021-11-03 12:27:00
##  Mean   :2021-07-21   Mean   :2021-11-03 12:37:09
##  3rd Qu.:2021-08-08   3rd Qu.:2021-11-03 14:05:00
##  Max.   :2021-08-30   Max.   :2021-11-03 18:09:00
##
##  individual_ID cloacal_temp_C    date_time
##  201    :  2   Min.   :23.00   Min.   :2021-06-16 09:54:00
##  202    :  2   1st Qu.:25.00   1st Qu.:2021-06-26 14:06:30
##  203    :  2   Median :26.00   Median :2021-07-20 15:02:00
##  204    :  2   Mean   :25.93   Mean   :2021-07-21 13:55:42
##  205    :  2   3rd Qu.:27.00   3rd Qu.:2021-08-08 15:25:30
##  206    :  2   Max.   :30.00   Max.   :2021-08-30 11:32:00
##  (Other):263
```

6. Load in and format the tmt assignments so we know which lizards were removed from the experiment.

```
tmt <- read.csv("./data/tmt_assignments.csv") %>%
  # properly format data classes
  mutate(trial_number = as.factor(trial_number),
         temp_tmt = as.factor(temp_tmt),
         humidity_tmt = as.factor(humidity_tmt),
         individual_ID = as.factor(individual_ID),
         conclusion = as.factor(conclusion)
         )
```

```
summary(tmt)
```

```
##  trial_number temp_tmt  humidity_tmt individual_ID     SVL_mm
##  1:26         cool:70   dry  :70     201    :  1   Min.   :60.00
##  2:32         hot :71   humid:71     202    :  1   1st Qu.:66.00
##  3:35                                203    :  1   Median :67.00
##  4:28                                204    :  1   Mean   :67.67
##  5:20                                205    :  1   3rd Qu.:70.00
##                                      206    :  1   Max.   :77.00
##                                      (Other):135
##     conclusion     notes               shed           tail_broken
##  canceled:  7   Length:141         Length:141         Length:141
##  complete:134   Class :character   Class :character   Class :character
##                 Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##      died
##  Length:141
##  Class :character
##  Mode  :character
##
##
##
##
```

```
# specifically save a df of canceled ones
canceled <- tmt %>%
  dplyr::filter(conclusion == "canceled") %>%
  dplyr::select(individual_ID)
canceled
```

```
##   individual_ID
## 1           212
## 2           233
## 3           248
## 4           254
## 5           283
## 6           284
## 7           304
```

Check Data

Dates

We should only have measurements from day 0 (beginning of date ranges below) and day 8 (end of date ranges below) for each trial.

Trail 1: June 16-24 Trail 2: June 26 - July 4 Trial 3: July 20-28 Trial 4: August 8-16 Trial 5: August 22-30

```
all_CEWL_data %>%
  group_by(date) %>%
  summarise(count = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 10 x 2
##    date       count
##    <date>     <int>
##  1 2021-06-16   130
##  2 2021-06-24   125
##  3 2021-06-26   158
##  4 2021-07-04   144
##  5 2021-07-20   175
##  6 2021-07-28   163
##  7 2021-08-08   140
##  8 2021-08-16   138
##  9 2021-08-22   100
## 10 2021-08-30   100
```

All the correct dates, and only the correct dates, are in our dataset. In every trial except trial 5, the number of observations decreases post-experiment compared to pre-experiment, either due to lost lizards or the few that died during the experiment.

Number of Measurements

Each individual should have 10 total measurements (5 before the experiment, 5 after).

```
rep_check <- all_CEWL_data %>%
                group_by(individual_ID) %>%
                summarise(n = n()) %>%
                arrange(n)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
rep_check
```

```
## # A tibble: 141 x 2
##    individual_ID     n
##    <fct>         <int>
##  1 254               3
##  2 212               5
##  3 233               5
##  4 239               5
##  5 248               5
##  6 283               5
##  7 284               5
##  8 303               5
##  9 213               9
## 10 216               9
## # ... with 131 more rows
```

Oof... Many individuals have more or less than 10 CEWL measurements.

too many: 206 & 215 = 11; 237 & 302 = 15 too few: 254 = 3; 213, 216, 245, 278, 289, 294, 305 = 9

There are also a handful with 5 measurements... Check whether these are the ones that had their treatment canceled (thus would only have measurements from pre experiment, not post).

```
# get the individuals with only 5 measures
rep_check5_msmts <- rep_check %>%
  dplyr::filter(n == 5)
rep_check5_msmts
```

```
## # A tibble: 7 x 2
##   individual_ID     n
##   <fct>         <int>
## 1 212               5
## 2 233               5
## 3 239               5
## 4 248               5
## 5 283               5
## 6 284               5
## 7 303               5
```

```
# when individuals with 5 reps makes sense
rep_check5_msmts %>%
  dplyr::filter(individual_ID %in% canceled$individual_ID)
```

```
## # A tibble: 5 x 2
##   individual_ID     n
##   <fct>         <int>
## 1 212               5
## 2 233               5
## 3 248               5
## 4 283               5
## 5 284               5
```

Of the 7 individuals with only 5 CEWL values, 5 individual lizards (*212*, *233*, *248*, *283*, *284*) had their treatment canceled, so we have an explanation for their missing data.

```
# when individuals with 5 reps DOES NOT make sense
rep_check5_msmts %>%
  dplyr::filter(individual_ID %nin% canceled$individual_ID)
```

```
## # A tibble: 2 x 2
##   individual_ID     n
##   <fct>         <int>
## 1 239               5
## 2 303               5
```

239 and 303 having 5 values is still unexplained and may be due to an error.

```
# individuals with canceled tmt but msmt n != 5
canceled %>% dplyr::filter(individual_ID %nin% rep_check5_msmts$individual_ID)
```

```
##   individual_ID
## 1           254
## 2           304
```

```
# check their n's
rep_check %>% dplyr::filter(individual_ID %in% c(254, 304))
```

```
## # A tibble: 2 x 2
##   individual_ID     n
##   <fct>         <int>
## 1 254               3
## 2 304              10
```

```
# check why canceled
tmt %>% dplyr::filter(individual_ID %in% c(254, 304))
```

```
##   trial_number temp_tmt humidity_tmt individual_ID SVL_mm conclusion
```

```
## 1            2     cool        humid           254     60   canceled
## 2            4     cool          dry           304     68   canceled
##                        notes shed tail_broken died
## 1 escaped during capture day
## 2                  recapture
```

Individuals 254 and 304 had their treatments canceled, but their n!=5. 254 only had 3 measurements taken because they were lost during CEWL measurement pre-treatment. Individual 304 has the correct number of observations (10), but it was canceled because we realized after the experiment that his toe was already clipped, thus was a recapture from a previous trial and we did not want to include his data. There were no measurement errors for these individuals. Whereas 254's capture measurements can be used for the capture analysis, 304's measurements should be removed from the dataset completely.

Save the individuals with measurement n's that I need to investigate further.

```
indiv_too_few <- rep_check %>% dplyr::filter(n == 9)
indiv_too_many <- rep_check %>% dplyr::filter(n > 10)
other_weird <- rep_check %>% dplyr::filter(individual_ID %in%
                                             c(239, 303)) # only 5 msmts

# save together to investigate further
weird_n <- indiv_too_few %>%
  rbind(indiv_too_many, other_weird) %>%
  arrange(n)
weird_n
```

```
## # A tibble: 13 x 2
##    individual_ID     n
##    <fct>         <int>
##  1 239               5
##  2 303               5
##  3 213               9
##  4 216               9
##  5 245               9
##  6 278               9
##  7 289               9
##  8 294               9
##  9 305               9
## 10 206              11
## 11 215              11
## 12 237              15
## 13 302              15
```

Next, check how many measurements each individual has for each date.

```
rep_check_1a <- all_CEWL_data %>%
  dplyr::filter(individual_ID %nin% weird_n$individual_ID) %>%
                group_by(individual_ID, date) %>%
                summarise(n = n()) %>%
                arrange(n)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
rep_check_1a
```

```
## # A tibble: 250 x 3
## # Groups:   individual_ID [128]
##    individual_ID date           n
```

```
##    <fct>         <date>     <int>
##  1 254           2021-06-26     3
##  2 201           2021-06-16     5
##  3 201           2021-06-24     5
##  4 202           2021-06-16     5
##  5 202           2021-06-24     5
##  6 203           2021-06-16     5
##  7 203           2021-06-24     5
##  8 204           2021-06-16     5
##  9 204           2021-06-24     5
## 10 205           2021-06-16     5
## # ... with 240 more rows
```

```
unique(rep_check_1a$n)
```

```
## [1] 3 5
```

It seems I have extracted all of the weird measurements. Every n on a given date ==5 for the individuals not included in my dataframe "weird_n", with the exception of individual 254, which I've already accounted for.

Now I can focus on the observations for the individuals in weird_n.

```
# save ones with one day of 5 msmts so I can filter out others' complete days
two_5s <- all_CEWL_data %>%
  dplyr::filter(individual_ID %in% c(239, 303)) %>%
  group_by(individual_ID, date) %>%
  summarise(n = n())
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
# get the weird msmt days for others
rep_check_1b <- all_CEWL_data %>%
  dplyr::filter(individual_ID %in% weird_n$individual_ID) %>%
  group_by(individual_ID, date) %>%
  summarise(n = n()) %>%
  dplyr::filter(n!=5) %>%
  rbind(two_5s) %>%
  arrange(n)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
rep_check_1b
```

```
## # A tibble: 13 x 3
## # Groups:   individual_ID [13]
##    individual_ID date           n
##    <fct>         <date>     <int>
##  1 213           2021-06-24     4
##  2 216           2021-06-24     4
##  3 245           2021-07-04     4
##  4 278           2021-07-28     4
##  5 289           2021-07-28     4
##  6 294           2021-08-16     4
##  7 305           2021-08-16     4
##  8 239           2021-06-26     5
##  9 303           2021-08-16     5
## 10 206           2021-06-24     6
## 11 215           2021-06-24     6
```

```
## 12 237           2021-07-04    10
## 13 302           2021-08-08    10
```

I have yet to figure out why individuals 213 and 216 (June 24), 245 (July 4), 278 and 289 (July 28), 294 and 305 (August 16) only have 4 observations on that date. The most likely explanation is that we miscounted replicates and only did 4, rather than 5. They have the correct number of measurements on their other measurement days.

Individuals 206 and 215 both have one extra replicate on June 24. Individuals 237 and 302 both have **10** replicates! On July 4 and August 8, respectively. They have the correct number of measurements on their other measurement days.

239 and 303 only have one day of measurements.

I will need to do more digging to figure out why these individuals have the wrong number of measurements on these dates.

Extra/Missing Measurements

Get all the data for the ones that aren't right:

```
rep_check_2 <- all_CEWL_data %>%
  left_join(rep_check_1b, by = c("individual_ID", "date")) %>%
  dplyr::filter(complete.cases(n))
```

Look at the weird data one at a time, starting with sets with too many replicates.

```
rep_check_2 %>%
  dplyr::filter(individual_ID == 302)
```

```
##          date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1  2021-08-08 2021-11-03 13:01:16 Normal     302-1      17.68        27.0
## 2  2021-08-08 2021-11-03 13:02:37 Normal     302-2      13.61        26.9
## 3  2021-08-08 2021-11-03 13:03:39 Normal     302-3      16.91        27.0
## 4  2021-08-08 2021-11-03 13:04:37 Normal     302-4      19.00        26.8
## 5  2021-08-08 2021-11-03 13:05:43 Normal     302-5      19.29        26.8
## 6  2021-08-08 2021-11-03 13:09:00 Normal     302-1      20.07        26.9
## 7  2021-08-08 2021-11-03 13:09:48 Normal     302-2      23.49        26.9
## 8  2021-08-08 2021-11-03 13:10:54 Normal     302-3      16.11        27.1
## 9  2021-08-08 2021-11-03 13:11:54 Normal     302-4      19.93        27.1
## 10 2021-08-08 2021-11-03 13:12:48 Normal     302-5      19.18        27.1
##    msmt_RH_percent individual_ID replicate_no           date_time  n
## 1             48.7           302            1 2021-08-08 13:01:16 10
## 2             49.1           302            2 2021-08-08 13:02:37 10
## 3             48.5           302            3 2021-08-08 13:03:39 10
## 4             49.1           302            4 2021-08-08 13:04:37 10
## 5             49.1           302            5 2021-08-08 13:05:43 10
## 6             48.9           302            1 2021-08-08 13:09:00 10
## 7             48.8           302            2 2021-08-08 13:09:48 10
## 8             48.5           302            3 2021-08-08 13:10:54 10
## 9             48.4           302            4 2021-08-08 13:11:54 10
## 10            48.3           302            5 2021-08-08 13:12:48 10
```

```
tmt %>%
  dplyr::filter(individual_ID == 302)
```

```
##   trial_number temp_tmt humidity_tmt individual_ID SVL_mm conclusion notes shed
## 1            4      hot        humid           302     63   complete
```

```
##   tail_broken died
## 1
```

Individual 302 has two sets of replicates from his capture day. One set is probably from him and the other set belongs to the lizard measured before or after him. Thankfully, on capture day, lizards are measured in number order, so I know it's probably either Individual 301 or 303. Since 303 is missing measurements, we'll check that.

```
all_CEWL_data %>%
  dplyr::filter(individual_ID == 303)
```

```
##         date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1 2021-08-16 2021-11-03 12:45:54 Normal     303-1      37.53        27.2
## 2 2021-08-16 2021-11-03 12:46:44 Normal     303-2      38.48        27.0
## 3 2021-08-16 2021-11-03 12:47:20 Normal     303-3      39.38        27.1
## 4 2021-08-16 2021-11-03 12:47:58 Normal     303-4      41.51        27.1
## 5 2021-08-16 2021-11-03 12:48:44 Normal     303-5      42.80        27.1
##   msmt_RH_percent individual_ID replicate_no           date_time
## 1            49.8           303            1 2021-08-16 12:45:54
## 2            49.6           303            2 2021-08-16 12:46:44
## 3            49.8           303            3 2021-08-16 12:47:20
## 4            49.8           303            4 2021-08-16 12:47:58
## 5            49.7           303            5 2021-08-16 12:48:44
```

```
tmt %>%
  dplyr::filter(individual_ID == 303)
```

```
##   trial_number temp_tmt humidity_tmt individual_ID SVL_mm conclusion notes
## 1            4      hot        humid           303     62   complete
##      shed tail_broken died
## 1 8/12/21
```

As suspected, Individual 303 only has pre-experiment measurements. We can check the time cloacal temperature was measured for these lizards on capture day to see which set of CEWL measurements belongs to who.

```
cloacal_temp_C %>%
  dplyr::filter(individual_ID %in% c(302,303) &
                  date == as.Date("2021-08-08"))
```

```
##         date         time_c_temp     day individual_ID cloacal_temp_C
## 1 2021-08-08 2021-11-03 13:06:00 capture           302             27
## 2 2021-08-08 2021-11-03 13:13:00 capture           303             27
##             date_time
## 1 2021-08-08 13:06:00
## 2 2021-08-08 13:13:00
```

302's temperature was taken at 13:06 and 303's temperature was taken at 13:13, so **the 13:01-13:05 CEWL measurements are for 302 and the 13:09-13:12 CEWL measurements are for 303**.

Discrepancies in number of measurements for individuals 302 and 303 solved!

```
rep_check_3 <- rep_check_2 %>%
  dplyr::filter(individual_ID %nin% c(302, 303)) %>%
  arrange(individual_ID)
# remaining individuals with replicate n's to investigate
unique(rep_check_3$individual_ID)
```

```
##  [1] 206 213 215 216 237 239 245 278 289 294 305
```

```
## 141 Levels: 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 ... 341
```

Next:

```
rep_check_2 %>%
  dplyr::filter(individual_ID == 237)
```

```
##          date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1  2021-07-04 2021-11-03 10:26:36 Normal     237-1      73.23        25.8
## 2  2021-07-04 2021-11-03 10:28:19 Normal     237-2      77.56        26.0
## 3  2021-07-04 2021-11-03 10:29:49 Normal     237-3      81.42        25.9
## 4  2021-07-04 2021-11-03 10:31:07 Normal     237-4      80.39        26.0
## 5  2021-07-04 2021-11-03 10:32:44 Normal     237-5      77.70        25.9
## 6  2021-07-04 2021-11-03 12:21:01 Normal     237-1      37.01        26.4
## 7  2021-07-04 2021-11-03 12:21:46 Normal     237-2      33.68        26.4
## 8  2021-07-04 2021-11-03 12:22:26 Normal     237-3      30.93        26.4
## 9  2021-07-04 2021-11-03 12:23:04 Normal     237-4      30.31        26.4
## 10 2021-07-04 2021-11-03 12:24:07 Normal     237-5      25.85        26.3
##    msmt_RH_percent individual_ID replicate_no           date_time  n
## 1             47.6           237            1 2021-07-04 10:26:36 10
## 2             47.1           237            2 2021-07-04 10:28:19 10
## 3             47.4           237            3 2021-07-04 10:29:49 10
## 4             47.1           237            4 2021-07-04 10:31:07 10
## 5             47.4           237            5 2021-07-04 10:32:44 10
## 6             46.4           237            1 2021-07-04 12:21:01 10
## 7             46.3           237            2 2021-07-04 12:21:46 10
## 8             46.4           237            3 2021-07-04 12:22:26 10
## 9             46.2           237            4 2021-07-04 12:23:04 10
## 10            46.3           237            5 2021-07-04 12:24:07 10
```

```
tmt %>%
  dplyr::filter(individual_ID == 237)
```

```
##   trial_number temp_tmt humidity_tmt individual_ID SVL_mm conclusion notes
## 1            2      hot        humid           237     71   complete
##     shed tail_broken died
## 1 7/4/21
```

Individual 237 also has an extra set of replicate measurements on the post-experiment day. The two sets of measurements are taken at two very different time blocks: 10:26-10:32 vs 12:21-12:24.

Interestingly, a closeby number is missing some measurements:

```
rep_check_2 %>%
  dplyr::filter(individual_ID == 239)
```

```
##         date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1 2021-06-26 2021-11-03 13:24:04 Normal     239-1      24.55        26.6
## 2 2021-06-26 2021-11-03 13:25:20 Normal     239-2      21.52        26.6
## 3 2021-06-26 2021-11-03 13:26:39 Normal     239-3      19.46        26.6
## 4 2021-06-26 2021-11-03 13:27:34 Normal     239-4      20.78        26.6
## 5 2021-06-26 2021-11-03 13:28:26 Normal     239-5      19.75        26.6
##   msmt_RH_percent individual_ID replicate_no           date_time n
## 1            47.6           239            1 2021-06-26 13:24:04 5
## 2            47.6           239            2 2021-06-26 13:25:20 5
## 3            47.8           239            3 2021-06-26 13:26:39 5
## 4            47.8           239            4 2021-06-26 13:27:34 5
## 5            47.7           239            5 2021-06-26 13:28:26 5
```

```
tmt %>%
  dplyr::filter(individual_ID == 239)
```

```
##   trial_number temp_tmt humidity_tmt individual_ID SVL_mm conclusion notes shed
## 1            2     cool        humid           239     69   complete
##   tail_broken died
## 1
```

Individual 239 is missing his post-experiment measurements on July 4. So, see if I can use cloacal temperature measurement times again to fix:

```
cloacal_temp_C %>%
  dplyr::filter(individual_ID %in% c(237,239) &
                  date == as.Date("2021-07-04"))
```

```
##         date         time_c_temp      day individual_ID cloacal_temp_C
## 1 2021-07-04 2021-11-03 12:24:00 post-exp           237             23
## 2 2021-07-04 2021-11-03 10:33:00 post-exp           239             23
##             date_time
## 1 2021-07-04 12:24:00
## 2 2021-07-04 10:33:00
```

237's temperature was taken at 12:24 and 239's temperature was taken at 10:33, so **the 12:21-12:24 CEWL measurements are for 237 and the 10:26-10:32 CEWL measurements are for 239**.

Discrepancies in number of measurements for individuals 237 and 239 solved!

Update list of individuals to investigate:

```
rep_check_4 <- rep_check_3 %>%
  dplyr::filter(individual_ID %nin% c(237, 239)) %>%
  arrange(individual_ID)
# remaining individuals with replicate n's to investigate
unique(rep_check_4$individual_ID)
```

```
## [1] 206 213 215 216 245 278 289 294 305
## 141 Levels: 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 ... 341
```

Next:

```
rep_check_2 %>%
  dplyr::filter(individual_ID == 215)
```

```
##         date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1 2021-06-24 2021-11-03 11:12:45 Normal     215-1      26.01        26.8
## 2 2021-06-24 2021-11-03 11:13:32 Normal     215-2      26.33        26.9
## 3 2021-06-24 2021-11-03 11:14:28 Normal     215-3      25.47        26.9
## 4 2021-06-24 2021-11-03 11:15:24 Normal     215-4      25.42        27.0
## 5 2021-06-24 2021-11-03 11:16:14 Normal     215-5      26.70        27.0
## 6 2021-06-24 2021-11-03 11:53:32 Normal     215-1      19.25        27.1
##   msmt_RH_percent individual_ID replicate_no           date_time n
## 1            44.2           215            1 2021-06-24 11:12:45 6
## 2            44.2           215            2 2021-06-24 11:13:32 6
## 3            44.4           215            3 2021-06-24 11:14:28 6
## 4            44.1           215            4 2021-06-24 11:15:24 6
## 5            43.9           215            5 2021-06-24 11:16:14 6
## 6            43.9           215            1 2021-06-24 11:53:32 6
```

The measurement from June 24 at 11:53:32 has a completely different time and CEWL value than the other

measurements for Individual 215 on that day. I can check cloacal temperature times from that day to make sure it's not a measurement for 215 and check whether it might belong to someone else.

```
cloacal_temp_C %>%
  dplyr::filter(date == as.Date("2021-06-24")) %>%
  arrange(time_c_temp)
```

```
##          date         time_c_temp      day individual_ID cloacal_temp_C
## 1  2021-06-24 2021-11-03 09:31:00 post-exp           220             25
## 2  2021-06-24 2021-11-03 09:39:00 post-exp           219             23
## 3  2021-06-24 2021-11-03 09:45:00 post-exp           201             24
## 4  2021-06-24 2021-11-03 09:54:00 post-exp           218             27
## 5  2021-06-24 2021-11-03 10:00:00 post-exp           210             25
## 6  2021-06-24 2021-11-03 10:06:00 post-exp           207             26
## 7  2021-06-24 2021-11-03 10:12:00 post-exp           225             24
## 8  2021-06-24 2021-11-03 10:18:00 post-exp           211             24
## 9  2021-06-24 2021-11-03 10:24:00 post-exp           203             23
## 10 2021-06-24 2021-11-03 10:30:00 post-exp           209             25
## 11 2021-06-24 2021-11-03 10:37:00 post-exp           217             25
## 12 2021-06-24 2021-11-03 10:44:00 post-exp           205             25
## 13 2021-06-24 2021-11-03 10:51:00 post-exp           221             25
## 14 2021-06-24 2021-11-03 10:56:00 post-exp           224             25
## 15 2021-06-24 2021-11-03 11:02:00 post-exp           208             25
## 16 2021-06-24 2021-11-03 11:10:00 post-exp           214             25
## 17 2021-06-24 2021-11-03 11:16:00 post-exp           215             26
## 18 2021-06-24 2021-11-03 11:22:00 post-exp           202             25
## 19 2021-06-24 2021-11-03 11:34:00 post-exp           204             25
## 20 2021-06-24 2021-11-03 11:40:00 post-exp           206             23
## 21 2021-06-24 2021-11-03 11:48:00 post-exp           222             23
## 22 2021-06-24 2021-11-03 11:53:00 post-exp           213             25
## 23 2021-06-24 2021-11-03 11:58:00 post-exp           226             24
## 24 2021-06-24 2021-11-03 12:06:00 post-exp           216             25
## 25 2021-06-24 2021-11-03 12:10:00 post-exp           223             24
##              date_time
## 1  2021-06-24 09:31:00
## 2  2021-06-24 09:39:00
## 3  2021-06-24 09:45:00
## 4  2021-06-24 09:54:00
## 5  2021-06-24 10:00:00
## 6  2021-06-24 10:06:00
## 7  2021-06-24 10:12:00
## 8  2021-06-24 10:18:00
## 9  2021-06-24 10:24:00
## 10 2021-06-24 10:30:00
## 11 2021-06-24 10:37:00
## 12 2021-06-24 10:44:00
## 13 2021-06-24 10:51:00
## 14 2021-06-24 10:56:00
## 15 2021-06-24 11:02:00
## 16 2021-06-24 11:10:00
## 17 2021-06-24 11:16:00
## 18 2021-06-24 11:22:00
## 19 2021-06-24 11:34:00
## 20 2021-06-24 11:40:00
## 21 2021-06-24 11:48:00
```

```
## 22 2021-06-24 11:53:00
## 23 2021-06-24 11:58:00
## 24 2021-06-24 12:06:00
## 25 2021-06-24 12:10:00
```

215 had his cloacal temperature taken at 11:16, confirming that only the CEWL values from between 11:12-11:16 are his. Individual 213 had his cloacal temp taken at 11:53, and 226 had his taken at 11:58. Now I can check whether either of them are missing CEWL values and what time their CEWL measurements were taken.

```
rep_check_2 %>%
  dplyr::filter(individual_ID == 213)
```

```
##         date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1 2021-06-24 2021-11-03 11:49:30 Normal     213-1      23.19        27.2
## 2 2021-06-24 2021-11-03 11:50:49 Normal     213-2      20.78        27.2
## 3 2021-06-24 2021-11-03 11:51:45 Normal     213-3      20.78        27.1
## 4 2021-06-24 2021-11-03 11:52:32 Normal     213-4      20.45        27.2
##   msmt_RH_percent individual_ID replicate_no           date_time n
## 1            44.0           213            1 2021-06-24 11:49:30 4
## 2            43.7           213            2 2021-06-24 11:50:49 4
## 3            43.9           213            3 2021-06-24 11:51:45 4
## 4            43.7           213            4 2021-06-24 11:52:32 4
```

```
all_CEWL_data %>%
  dplyr::filter(individual_ID == 226)
```

```
##          date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1  2021-06-16 2021-11-03 16:29:15 Normal     226-1      21.09        29.1
## 2  2021-06-16 2021-11-03 16:30:18 Normal     226-2      18.53        29.1
## 3  2021-06-16 2021-11-03 16:31:04 Normal     226-3      20.51        29.2
## 4  2021-06-16 2021-11-03 16:31:42 Normal     226-4      21.02        29.2
## 5  2021-06-16 2021-11-03 16:32:21 Normal     226-5      18.82        29.1
## 6  2021-06-24 2021-11-03 11:55:19 Normal     226-1      43.27        27.2
## 7  2021-06-24 2021-11-03 11:56:02 Normal     226-2      37.17        27.1
## 8  2021-06-24 2021-11-03 11:56:43 Normal     226-3      33.46        27.3
## 9  2021-06-24 2021-11-03 11:57:29 Normal     226-4      30.50        27.2
## 10 2021-06-24 2021-11-03 11:58:13 Normal     226-5      29.32        27.2
##    msmt_RH_percent individual_ID replicate_no           date_time
## 1             28.2           226            1 2021-06-16 16:29:15
## 2             28.1           226            2 2021-06-16 16:30:18
## 3             27.8           226            3 2021-06-16 16:31:04
## 4             27.6           226            4 2021-06-16 16:31:42
## 5             27.6           226            5 2021-06-16 16:32:21
## 6             44.1           226            1 2021-06-24 11:55:19
## 7             43.8           226            2 2021-06-24 11:56:02
## 8             43.5           226            3 2021-06-24 11:56:43
## 9             43.6           226            4 2021-06-24 11:57:29
## 10            43.4           226            5 2021-06-24 11:58:13
```

Individual 226 isn't missing anything. BUT, individual 213 is missing his fifth replicate of CEWL measurements taken post-experiment. The 4 measurements currently attributed to him were taken between 11:49-11:52, so the extra value attributed to 215 at 11:53 fits perfectly into that sequence of replicates.

Discrepancies in number of measurements for individuals 215 and 213 solved!

Update list of individuals to investigate:

```
rep_check_5 <- rep_check_4 %>%
  dplyr::filter(individual_ID %nin% c(215, 213)) %>%
  arrange(individual_ID)
# remaining individuals with replicate n's to investigate
unique(rep_check_5$individual_ID)
```

```
## [1] 206 216 245 278 289 294 305
## 141 Levels: 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 ... 341
```

Next:

```
rep_check_2 %>%
  dplyr::filter(individual_ID == 206)
```

```
##         date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1 2021-06-24 2021-11-03 11:36:07 Normal     206-1      32.70        27.2
## 2 2021-06-24 2021-11-03 11:37:13 Normal     206-2      28.33        27.0
## 3 2021-06-24 2021-11-03 11:37:53 Normal     206-2      32.13        27.1
## 4 2021-06-24 2021-11-03 11:38:32 Normal     206-3      33.64        27.2
## 5 2021-06-24 2021-11-03 11:39:21 Normal     206-4      29.58        27.1
## 6 2021-06-24 2021-11-03 11:40:01 Normal     206-5      28.34        27.2
##   msmt_RH_percent individual_ID replicate_no           date_time n
## 1            43.8           206            1 2021-06-24 11:36:07 6
## 2            44.1           206            2 2021-06-24 11:37:13 6
## 3            44.2           206            2 2021-06-24 11:37:53 6
## 4            44.1           206            3 2021-06-24 11:38:32 6
## 5            44.0           206            4 2021-06-24 11:39:21 6
## 6            43.6           206            5 2021-06-24 11:40:01 6
```

Individual 206 has two #2 replicates taken at 11:37, just 40 seconds apart, which is the normal time in-between back-to-back measurements when there are no distractions. So, the extra measurement can be considered a sixth replicate and should be relabeled as such.

Mystery for Individual 206's weird number of replicates is solved.

Update list of individuals to investigate:

```
rep_check_6 <- rep_check_5 %>%
  dplyr::filter(individual_ID != 206) %>%
  arrange(individual_ID)
# remaining individuals with replicate n's to investigate
unique(rep_check_6$individual_ID)
```

```
## [1] 216 245 278 289 294 305
## 141 Levels: 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 ... 341
```

Next:

```
rep_check_2 %>%
  dplyr::filter(individual_ID == 216)
```

```
##         date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1 2021-06-24 2021-11-03 12:00:43 Normal     216-1      22.70        27.2
## 2 2021-06-24 2021-11-03 12:01:43 Normal     216-2      22.25        27.2
## 3 2021-06-24 2021-11-03 12:02:39 Normal     216-3      20.82        27.3
## 4 2021-06-24 2021-11-03 12:03:42 Normal     216-5      21.08        27.2
##   msmt_RH_percent individual_ID replicate_no           date_time n
## 1            43.6           216            1 2021-06-24 12:00:43 4
```

```
## 2            44.1           216            2 2021-06-24 12:01:43 4
## 3            43.4           216            3 2021-06-24 12:02:39 4
## 4            43.8           216            5 2021-06-24 12:03:42 4
```

Individual 216 is missing his 4th replicate. There is only one minute between replicates 3 and 5, so I believe the 4th replicate got accidentally skipped/forgotten.

216's mystery solved!

Update list of individuals to investigate:

```
rep_check_7 <- rep_check_6 %>%
  dplyr::filter(individual_ID != 216) %>%
  arrange(individual_ID) %>%
  group_by(individual_ID, date) %>%
  summarise(n = n())
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
# remaining individuals with replicate n's to investigate
rep_check_7
```

```
## # A tibble: 5 x 3
## # Groups:   individual_ID [5]
##   individual_ID date           n
##   <fct>         <date>     <int>
## 1 245           2021-07-04     4
## 2 278           2021-07-28     4
## 3 289           2021-07-28     4
## 4 294           2021-08-16     4
## 5 305           2021-08-16     4
```

The remaining individuals had only 4 replicates on one day, which is probably for the same reason as 216- one replicate was forgotten/we miscounted replicate numbers. No adjustment possible/necessary.

All unexpected n's are explained.

Make note of which individuals still won't have n = 5/10:

```
unconforming_but_fine <- data.frame(IDs = c(216, 245, 278, 289, 294, 305,
                                            206, 254),
                                    total_n = c(9, 9, 9, 9, 9, 9,
                                                11, 3),
                                    single_date_n = c(4, 4, 4, 4, 4, 4,
                                                      6, 3)
                                    )
```

Properly Re-Assign Measurements

1. 304's measurements should be removed from the dataset completely. This should remove 10 rows of data. Also give the dataset a specific order to follow to make indexing correct.

```
nrow(all_CEWL_data)
```

```
## [1] 1373
```

```
all_CEWL_data %>%
  dplyr::filter(individual_ID == 304)
```

```
##          date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1  2021-08-16 2021-11-03 10:51:21 Normal     304-1      17.23        26.3
```

```
## 2  2021-08-16 2021-11-03 10:52:07 Normal     304-2      21.89        26.2
## 3  2021-08-16 2021-11-03 10:52:58 Normal     304-3      20.67        26.2
## 4  2021-08-16 2021-11-03 10:53:44 Normal     304-4      22.84        26.2
## 5  2021-08-16 2021-11-03 10:54:24 Normal     304-5      22.75        26.1
## 6  2021-08-08 2021-11-03 13:15:55 Normal     304-1      19.24        26.9
## 7  2021-08-08 2021-11-03 13:17:02 Normal     304-2      18.62        27.2
## 8  2021-08-08 2021-11-03 13:17:52 Normal     304-3      18.99        27.1
## 9  2021-08-08 2021-11-03 13:18:47 Normal     304-4      19.11        26.8
## 10 2021-08-08 2021-11-03 13:19:48 Normal     304-5      19.20        26.9
##    msmt_RH_percent individual_ID replicate_no           date_time
## 1             51.9           304            1 2021-08-16 10:51:21
## 2             51.5           304            2 2021-08-16 10:52:07
## 3             51.6           304            3 2021-08-16 10:52:58
## 4             51.6           304            4 2021-08-16 10:53:44
## 5             51.5           304            5 2021-08-16 10:54:24
## 6             48.9           304            1 2021-08-08 13:15:55
## 7             48.1           304            2 2021-08-08 13:17:02
## 8             48.1           304            3 2021-08-08 13:17:52
## 9             48.7           304            4 2021-08-08 13:18:47
## 10            48.6           304            5 2021-08-08 13:19:48
```

```
all_CEWL_data_edited <- all_CEWL_data %>%
  dplyr::filter(individual_ID != 304) %>%
  arrange(date, individual_ID, time, replicate_no)
nrow(all_CEWL_data_edited)
```

```
## [1] 1363
```

```
all_CEWL_data_edited %>%
  dplyr::filter(individual_ID == 304)
```

```
##  [1] date            time            status          ID_rep_no
##  [5] CEWL_g_m2h      msmt_temp_C     msmt_RH_percent individual_ID
##  [9] replicate_no    date_time
## <0 rows> (or 0-length row.names)
```

2. Reassign the measurements attributed to individual 302 taken between 13:09-13:12 on August 8 as pre-experiment measurements for individual 303.

```
all_CEWL_data_edited[936:945, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 936 2021-08-08 2021-11-03 13:01:16 Normal     302-1      17.68        27.0
## 937 2021-08-08 2021-11-03 13:02:37 Normal     302-2      13.61        26.9
## 938 2021-08-08 2021-11-03 13:03:39 Normal     302-3      16.91        27.0
## 939 2021-08-08 2021-11-03 13:04:37 Normal     302-4      19.00        26.8
## 940 2021-08-08 2021-11-03 13:05:43 Normal     302-5      19.29        26.8
## 941 2021-08-08 2021-11-03 13:09:00 Normal     302-1      20.07        26.9
## 942 2021-08-08 2021-11-03 13:09:48 Normal     302-2      23.49        26.9
## 943 2021-08-08 2021-11-03 13:10:54 Normal     302-3      16.11        27.1
## 944 2021-08-08 2021-11-03 13:11:54 Normal     302-4      19.93        27.1
## 945 2021-08-08 2021-11-03 13:12:48 Normal     302-5      19.18        27.1
##     msmt_RH_percent individual_ID replicate_no           date_time
## 936            48.7           302            1 2021-08-08 13:01:16
## 937            49.1           302            2 2021-08-08 13:02:37
## 938            48.5           302            3 2021-08-08 13:03:39
## 939            49.1           302            4 2021-08-08 13:04:37
```

```
## 940            49.1           302            5 2021-08-08 13:05:43
## 941            48.9           302            1 2021-08-08 13:09:00
## 942            48.8           302            2 2021-08-08 13:09:48
## 943            48.5           302            3 2021-08-08 13:10:54
## 944            48.4           302            4 2021-08-08 13:11:54
## 945            48.3           302            5 2021-08-08 13:12:48
```

```
all_CEWL_data_edited[941:945, "individual_ID"] <- 303
all_CEWL_data_edited[936:945, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 936 2021-08-08 2021-11-03 13:01:16 Normal     302-1      17.68        27.0
## 937 2021-08-08 2021-11-03 13:02:37 Normal     302-2      13.61        26.9
## 938 2021-08-08 2021-11-03 13:03:39 Normal     302-3      16.91        27.0
## 939 2021-08-08 2021-11-03 13:04:37 Normal     302-4      19.00        26.8
## 940 2021-08-08 2021-11-03 13:05:43 Normal     302-5      19.29        26.8
## 941 2021-08-08 2021-11-03 13:09:00 Normal     302-1      20.07        26.9
## 942 2021-08-08 2021-11-03 13:09:48 Normal     302-2      23.49        26.9
## 943 2021-08-08 2021-11-03 13:10:54 Normal     302-3      16.11        27.1
## 944 2021-08-08 2021-11-03 13:11:54 Normal     302-4      19.93        27.1
## 945 2021-08-08 2021-11-03 13:12:48 Normal     302-5      19.18        27.1
##     msmt_RH_percent individual_ID replicate_no           date_time
## 936            48.7           302            1 2021-08-08 13:01:16
## 937            49.1           302            2 2021-08-08 13:02:37
## 938            48.5           302            3 2021-08-08 13:03:39
## 939            49.1           302            4 2021-08-08 13:04:37
## 940            49.1           302            5 2021-08-08 13:05:43
## 941            48.9           303            1 2021-08-08 13:09:00
## 942            48.8           303            2 2021-08-08 13:09:48
## 943            48.5           303            3 2021-08-08 13:10:54
## 944            48.4           303            4 2021-08-08 13:11:54
## 945            48.3           303            5 2021-08-08 13:12:48
```

3. Reassign the measurements attributed to individual 237 taken between 10:26-10:32 on July 4 as post-experiment measurements for individual 239.

```
all_CEWL_data_edited[459:468, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 459 2021-07-04 2021-11-03 10:26:36 Normal     237-1      73.23        25.8
## 460 2021-07-04 2021-11-03 10:28:19 Normal     237-2      77.56        26.0
## 461 2021-07-04 2021-11-03 10:29:49 Normal     237-3      81.42        25.9
## 462 2021-07-04 2021-11-03 10:31:07 Normal     237-4      80.39        26.0
## 463 2021-07-04 2021-11-03 10:32:44 Normal     237-5      77.70        25.9
## 464 2021-07-04 2021-11-03 12:21:01 Normal     237-1      37.01        26.4
## 465 2021-07-04 2021-11-03 12:21:46 Normal     237-2      33.68        26.4
## 466 2021-07-04 2021-11-03 12:22:26 Normal     237-3      30.93        26.4
## 467 2021-07-04 2021-11-03 12:23:04 Normal     237-4      30.31        26.4
## 468 2021-07-04 2021-11-03 12:24:07 Normal     237-5      25.85        26.3
##     msmt_RH_percent individual_ID replicate_no           date_time
## 459            47.6           237            1 2021-07-04 10:26:36
## 460            47.1           237            2 2021-07-04 10:28:19
## 461            47.4           237            3 2021-07-04 10:29:49
## 462            47.1           237            4 2021-07-04 10:31:07
## 463            47.4           237            5 2021-07-04 10:32:44
## 464            46.4           237            1 2021-07-04 12:21:01
```

```
## 465            46.3           237            2 2021-07-04 12:21:46
## 466            46.4           237            3 2021-07-04 12:22:26
## 467            46.2           237            4 2021-07-04 12:23:04
## 468            46.3           237            5 2021-07-04 12:24:07
```

```
all_CEWL_data_edited[459:463, "individual_ID"] <- 239
all_CEWL_data_edited[459:468, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 459 2021-07-04 2021-11-03 10:26:36 Normal     237-1      73.23        25.8
## 460 2021-07-04 2021-11-03 10:28:19 Normal     237-2      77.56        26.0
## 461 2021-07-04 2021-11-03 10:29:49 Normal     237-3      81.42        25.9
## 462 2021-07-04 2021-11-03 10:31:07 Normal     237-4      80.39        26.0
## 463 2021-07-04 2021-11-03 10:32:44 Normal     237-5      77.70        25.9
## 464 2021-07-04 2021-11-03 12:21:01 Normal     237-1      37.01        26.4
## 465 2021-07-04 2021-11-03 12:21:46 Normal     237-2      33.68        26.4
## 466 2021-07-04 2021-11-03 12:22:26 Normal     237-3      30.93        26.4
## 467 2021-07-04 2021-11-03 12:23:04 Normal     237-4      30.31        26.4
## 468 2021-07-04 2021-11-03 12:24:07 Normal     237-5      25.85        26.3
##     msmt_RH_percent individual_ID replicate_no           date_time
## 459            47.6           239            1 2021-07-04 10:26:36
## 460            47.1           239            2 2021-07-04 10:28:19
## 461            47.4           239            3 2021-07-04 10:29:49
## 462            47.1           239            4 2021-07-04 10:31:07
## 463            47.4           239            5 2021-07-04 10:32:44
## 464            46.4           237            1 2021-07-04 12:21:01
## 465            46.3           237            2 2021-07-04 12:21:46
## 466            46.4           237            3 2021-07-04 12:22:26
## 467            46.2           237            4 2021-07-04 12:23:04
## 468            46.3           237            5 2021-07-04 12:24:07
```

4. Reassign the measurement attributed to individual 215 at 11:53 on June 24 as the fifth replicate for individual 213 on that date.

```
all_CEWL_data_edited[187:201, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 187 2021-06-24 2021-11-03 11:49:30 Normal     213-1      23.19        27.2
## 188 2021-06-24 2021-11-03 11:50:49 Normal     213-2      20.78        27.2
## 189 2021-06-24 2021-11-03 11:51:45 Normal     213-3      20.78        27.1
## 190 2021-06-24 2021-11-03 11:52:32 Normal     213-4      20.45        27.2
## 191 2021-06-24 2021-11-03 11:07:24 Normal     214-1      41.48        27.0
## 192 2021-06-24 2021-11-03 11:08:05 Normal     214-2      37.31        26.9
## 193 2021-06-24 2021-11-03 11:08:43 Normal     214-3      35.28        26.9
## 194 2021-06-24 2021-11-03 11:09:29 Normal     214-4      32.45        27.0
## 195 2021-06-24 2021-11-03 11:10:07 Normal     214-5      32.04        27.0
## 196 2021-06-24 2021-11-03 11:12:45 Normal     215-1      26.01        26.8
## 197 2021-06-24 2021-11-03 11:13:32 Normal     215-2      26.33        26.9
## 198 2021-06-24 2021-11-03 11:14:28 Normal     215-3      25.47        26.9
## 199 2021-06-24 2021-11-03 11:15:24 Normal     215-4      25.42        27.0
## 200 2021-06-24 2021-11-03 11:16:14 Normal     215-5      26.70        27.0
## 201 2021-06-24 2021-11-03 11:53:32 Normal     215-1      19.25        27.1
##     msmt_RH_percent individual_ID replicate_no           date_time
## 187            44.0           213            1 2021-06-24 11:49:30
## 188            43.7           213            2 2021-06-24 11:50:49
## 189            43.9           213            3 2021-06-24 11:51:45
```

```
## 190            43.7           213            4 2021-06-24 11:52:32
## 191            43.8           214            1 2021-06-24 11:07:24
## 192            43.6           214            2 2021-06-24 11:08:05
## 193            43.7           214            3 2021-06-24 11:08:43
## 194            43.6           214            4 2021-06-24 11:09:29
## 195            43.7           214            5 2021-06-24 11:10:07
## 196            44.2           215            1 2021-06-24 11:12:45
## 197            44.2           215            2 2021-06-24 11:13:32
## 198            44.4           215            3 2021-06-24 11:14:28
## 199            44.1           215            4 2021-06-24 11:15:24
## 200            43.9           215            5 2021-06-24 11:16:14
## 201            43.9           215            1 2021-06-24 11:53:32
```

```
all_CEWL_data_edited[201, "replicate_no"] <- 5
all_CEWL_data_edited[201, "individual_ID"] <- 213
all_CEWL_data_edited[187:201, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 187 2021-06-24 2021-11-03 11:49:30 Normal     213-1      23.19        27.2
## 188 2021-06-24 2021-11-03 11:50:49 Normal     213-2      20.78        27.2
## 189 2021-06-24 2021-11-03 11:51:45 Normal     213-3      20.78        27.1
## 190 2021-06-24 2021-11-03 11:52:32 Normal     213-4      20.45        27.2
## 191 2021-06-24 2021-11-03 11:07:24 Normal     214-1      41.48        27.0
## 192 2021-06-24 2021-11-03 11:08:05 Normal     214-2      37.31        26.9
## 193 2021-06-24 2021-11-03 11:08:43 Normal     214-3      35.28        26.9
## 194 2021-06-24 2021-11-03 11:09:29 Normal     214-4      32.45        27.0
## 195 2021-06-24 2021-11-03 11:10:07 Normal     214-5      32.04        27.0
## 196 2021-06-24 2021-11-03 11:12:45 Normal     215-1      26.01        26.8
## 197 2021-06-24 2021-11-03 11:13:32 Normal     215-2      26.33        26.9
## 198 2021-06-24 2021-11-03 11:14:28 Normal     215-3      25.47        26.9
## 199 2021-06-24 2021-11-03 11:15:24 Normal     215-4      25.42        27.0
## 200 2021-06-24 2021-11-03 11:16:14 Normal     215-5      26.70        27.0
## 201 2021-06-24 2021-11-03 11:53:32 Normal     215-1      19.25        27.1
##     msmt_RH_percent individual_ID replicate_no           date_time
## 187            44.0           213            1 2021-06-24 11:49:30
## 188            43.7           213            2 2021-06-24 11:50:49
## 189            43.9           213            3 2021-06-24 11:51:45
## 190            43.7           213            4 2021-06-24 11:52:32
## 191            43.8           214            1 2021-06-24 11:07:24
## 192            43.6           214            2 2021-06-24 11:08:05
## 193            43.7           214            3 2021-06-24 11:08:43
## 194            43.6           214            4 2021-06-24 11:09:29
## 195            43.7           214            5 2021-06-24 11:10:07
## 196            44.2           215            1 2021-06-24 11:12:45
## 197            44.2           215            2 2021-06-24 11:13:32
## 198            44.4           215            3 2021-06-24 11:14:28
## 199            44.1           215            4 2021-06-24 11:15:24
## 200            43.9           215            5 2021-06-24 11:16:14
## 201            43.9           213            5 2021-06-24 11:53:32
```

5. Relabel one of 206's June 24 #2 replicates as 206's sixth replicate.

```
all_CEWL_data_edited[156:161, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 156 2021-06-24 2021-11-03 11:36:07 Normal     206-1      32.70        27.2
```

```
## 157 2021-06-24 2021-11-03 11:37:13 Normal     206-2      28.33        27.0
## 158 2021-06-24 2021-11-03 11:37:53 Normal     206-2      32.13        27.1
## 159 2021-06-24 2021-11-03 11:38:32 Normal     206-3      33.64        27.2
## 160 2021-06-24 2021-11-03 11:39:21 Normal     206-4      29.58        27.1
## 161 2021-06-24 2021-11-03 11:40:01 Normal     206-5      28.34        27.2
##     msmt_RH_percent individual_ID replicate_no           date_time
## 156            43.8           206            1 2021-06-24 11:36:07
## 157            44.1           206            2 2021-06-24 11:37:13
## 158            44.2           206            2 2021-06-24 11:37:53
## 159            44.1           206            3 2021-06-24 11:38:32
## 160            44.0           206            4 2021-06-24 11:39:21
## 161            43.6           206            5 2021-06-24 11:40:01
```

```
all_CEWL_data_edited[158, "replicate_no"] <- 6
all_CEWL_data_edited[156:161, ]
```

```
##           date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 156 2021-06-24 2021-11-03 11:36:07 Normal     206-1      32.70        27.2
## 157 2021-06-24 2021-11-03 11:37:13 Normal     206-2      28.33        27.0
## 158 2021-06-24 2021-11-03 11:37:53 Normal     206-2      32.13        27.1
## 159 2021-06-24 2021-11-03 11:38:32 Normal     206-3      33.64        27.2
## 160 2021-06-24 2021-11-03 11:39:21 Normal     206-4      29.58        27.1
## 161 2021-06-24 2021-11-03 11:40:01 Normal     206-5      28.34        27.2
##     msmt_RH_percent individual_ID replicate_no           date_time
## 156            43.8           206            1 2021-06-24 11:36:07
## 157            44.1           206            2 2021-06-24 11:37:13
## 158            44.2           206            6 2021-06-24 11:37:53
## 159            44.1           206            3 2021-06-24 11:38:32
## 160            44.0           206            4 2021-06-24 11:39:21
## 161            43.6           206            5 2021-06-24 11:40:01
```

Re-Check Data

Dates

```
all_CEWL_data_edited %>%
  group_by(date) %>%
  summarise(count = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 10 x 2
##    date       count
##    <date>     <int>
##  1 2021-06-16   130
##  2 2021-06-24   125
##  3 2021-06-26   158
##  4 2021-07-04   144
##  5 2021-07-20   175
##  6 2021-07-28   163
##  7 2021-08-08   135
##  8 2021-08-16   133
##  9 2021-08-22   100
## 10 2021-08-30   100
```

Still correct.

Number of Measurements

Each individual should have 10 total measurements (5 before the experiment, 5 after).

```
unconforming_but_fine
```

```
##   IDs total_n single_date_n
## 1 216       9             4
## 2 245       9             4
## 3 278       9             4
## 4 289       9             4
## 5 294       9             4
## 6 305       9             4
## 7 206      11             6
## 8 254       3             3
```

```
canceled
```

```
##   individual_ID
## 1           212
## 2           233
## 3           248
## 4           254
## 5           283
## 6           284
## 7           304
```

```
all_CEWL_data_edited %>%
  group_by(individual_ID) %>%
  summarise(n = n()) %>%
  arrange(n)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 140 x 2
##    individual_ID     n
##    <fct>         <int>
##  1 254               3
##  2 212               5
##  3 233               5
##  4 248               5
##  5 283               5
##  6 284               5
##  7 216               9
##  8 245               9
##  9 278               9
## 10 289               9
## # ... with 130 more rows
```

```
all_CEWL_data_edited %>%
  group_by(individual_ID, date) %>%
  summarise(n = n()) %>%
  arrange(n)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## # A tibble: 274 x 3
## # Groups:   individual_ID [140]
##    individual_ID date           n
##    <fct>         <date>     <int>
##  1 254           2021-06-26     3
##  2 216           2021-06-24     4
##  3 245           2021-07-04     4
##  4 278           2021-07-28     4
##  5 289           2021-07-28     4
##  6 294           2021-08-16     4
##  7 305           2021-08-16     4
##  8 201           2021-06-16     5
##  9 201           2021-06-24     5
## 10 202           2021-06-16     5
## # ... with 264 more rows
```

Every number of replicates is explained, whether it was the expected n (5/10) or not.

Measurement Times

Also check that all the measurement times for a given individual on a certain date are within ~10 minutes:

```
all_CEWL_data_edited %>%
  group_by(individual_ID, date) %>%
  summarise(min_time = min(date_time),
            max_time = max(date_time),
            msmt_time_range_minutes = ((max_time-min_time)/60)) %>%
  dplyr::select(individual_ID, date, msmt_time_range_minutes) %>%
  arrange(msmt_time_range_minutes)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## # A tibble: 274 x 3
## # Groups:   individual_ID [140]
##    individual_ID date       msmt_time_range_minutes
##    <fct>         <date>     <drtn>
##  1 254           2021-06-26 1.700000 secs
##  2 245           2021-07-04 1.950000 secs
##  3 278           2021-07-28 2.116667 secs
##  4 294           2021-08-16 2.266667 secs
##  5 316           2021-08-16 2.450000 secs
##  6 251           2021-07-04 2.516667 secs
##  7 279           2021-07-28 2.516667 secs
##  8 282           2021-07-28 2.516667 secs
##  9 243           2021-07-04 2.533333 secs
## 10 277           2021-07-28 2.550000 secs
## # ... with 264 more rows
```

I want to double check on individuals 305 on August 16 and 233 on June 26 because they have measurement time ranges of ~10.5 and ~91 minutes, respectively, which is much greater than the typical 1.7-7.8 minute range for all the other individuals.

```
# CEWL
all_CEWL_data_edited %>%
  dplyr::filter(individual_ID %in% c(305, 233))
```

```
##          date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
```

```
## 1  2021-06-26 2021-11-03 12:42:14 Normal     233-1      16.53        26.4
## 2  2021-06-26 2021-11-03 12:43:03 Normal     233-2      17.10        26.4
## 3  2021-06-26 2021-11-03 12:43:40 Normal     233-3      20.69        26.3
## 4  2021-06-26 2021-11-03 12:44:43 Normal     233-5      14.64        26.3
## 5  2021-06-26 2021-11-03 14:13:10 Normal     233-5      22.34        26.6
## 6  2021-08-08 2021-11-03 13:22:37 Normal     305-1      26.78        26.9
## 7  2021-08-08 2021-11-03 13:23:23 Normal     305-2      31.81        26.9
## 8  2021-08-08 2021-11-03 13:25:03 Normal     305-3      20.24        26.7
## 9  2021-08-08 2021-11-03 13:25:49 Normal     305-4      25.67        26.7
## 10 2021-08-08 2021-11-03 13:26:38 Normal     305-5      24.27        26.7
## 11 2021-08-16 2021-11-03 12:04:28 Normal     305-1      26.49        26.7
## 12 2021-08-16 2021-11-03 12:05:26 Normal     305-2      27.63        26.6
## 13 2021-08-16 2021-11-03 12:06:23 Normal     305-3      24.55        26.8
## 14 2021-08-16 2021-11-03 12:14:55 Normal     305-5      27.28        27.1
##    msmt_RH_percent individual_ID replicate_no           date_time
## 1             48.0           233            1 2021-06-26 12:42:14
## 2             47.8           233            2 2021-06-26 12:43:03
## 3             47.8           233            3 2021-06-26 12:43:40
## 4             47.7           233            5 2021-06-26 12:44:43
## 5             47.2           233            5 2021-06-26 14:13:10
## 6             48.7           305            1 2021-08-08 13:22:37
## 7             48.7           305            2 2021-08-08 13:23:23
## 8             49.0           305            3 2021-08-08 13:25:03
## 9             49.1           305            4 2021-08-08 13:25:49
## 10            49.2           305            5 2021-08-08 13:26:38
## 11            49.3           305            1 2021-08-16 12:04:28
## 12            49.6           305            2 2021-08-16 12:05:26
## 13            49.6           305            3 2021-08-16 12:06:23
## 14            49.5           305            5 2021-08-16 12:14:55
```

```
# cloacal temps
cloacal_temp_C %>%
  dplyr::filter(individual_ID %in% c(305, 233))
```

```
##         date         time_c_temp      day individual_ID cloacal_temp_C
## 1 2021-06-26 2021-11-03 12:45:00  capture           233             26
## 2 2021-08-08 2021-11-03 13:27:00  capture           305             26
## 3 2021-08-16 2021-11-03 12:15:00 post-exp           305             26
##             date_time
## 1 2021-06-26 12:45:00
## 2 2021-08-08 13:27:00
## 3 2021-08-16 12:15:00
```

The cloacal temperature for individual 305 was taken at 12:15 on August 16, which is right after the fifth replicate was recorded. Either the fourth replicate did not have a "Normal" (successful) measurement, or we got distracted and miscounted. The time range for 305 is fine.

The measurement for individual 233 at 14:13 must have been an incorrectly labeled measurement for another individual, since his cloacal temperature was taken at 12:45.

I can check whether any of the individuals with 4 replicates are missing one on that day:

```
rep_check_6 %>%
  group_by(individual_ID, date) %>%
  summarise(n = n()) #%>%
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## # A tibble: 6 x 3
## # Groups:   individual_ID [6]
##   individual_ID date           n
##   <fct>         <date>     <int>
## 1 216           2021-06-24     4
## 2 245           2021-07-04     4
## 3 278           2021-07-28     4
## 4 289           2021-07-28     4
## 5 294           2021-08-16     4
## 6 305           2021-08-16     4
```

```
  #dplyr::filter(date == as.Date("2021-06-26"))
```

Nothing matches. I think the measurement taken for individual 233 1.5 hours later than his other replicates should still be omitted since we cannot be confident that measurement was on him, and his cloacal temperature was taken prior to that CEWL measurement, which is contrary to our protocol of taking all CEWL measurements then .

Omit Temporal Outlier

This should remove one row of data.

```
nrow(all_CEWL_data_edited)
```

```
## [1] 1363
```

```
all_CEWL_data_edited2 <- all_CEWL_data_edited %>%
  dplyr::filter(individual_ID != 233 | # or
                  date_time != as.POSIXct("2021-06-26 14:13:10")) %>%
  arrange(date, individual_ID, time, replicate_no)
nrow(all_CEWL_data_edited2)
```

```
## [1] 1362
```

Check the values again:

```
all_CEWL_data_edited2 %>%
  dplyr::filter(individual_ID %in% c(233))
```

```
##         date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1 2021-06-26 2021-11-03 12:42:14 Normal     233-1      16.53        26.4
## 2 2021-06-26 2021-11-03 12:43:03 Normal     233-2      17.10        26.4
## 3 2021-06-26 2021-11-03 12:43:40 Normal     233-3      20.69        26.3
## 4 2021-06-26 2021-11-03 12:44:43 Normal     233-5      14.64        26.3
##   msmt_RH_percent individual_ID replicate_no           date_time
## 1            48.0           233            1 2021-06-26 12:42:14
## 2            47.8           233            2 2021-06-26 12:43:03
## 3            47.8           233            3 2021-06-26 12:43:40
## 4            47.7           233            5 2021-06-26 12:44:43
```

Re-Check Measurement Times

```
all_CEWL_data_edited2 %>%
  group_by(individual_ID, date) %>%
  summarise(min_time = min(date_time),
            max_time = max(date_time),
            msmt_time_range_minutes = (max_time-min_time)) %>%
```

```
  dplyr::select(individual_ID, date, msmt_time_range_minutes) %>%
  arrange(msmt_time_range_minutes)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## # A tibble: 274 x 3
## # Groups:   individual_ID [140]
##    individual_ID date       msmt_time_range_minutes
##    <fct>         <date>     <drtn>
##  1 254           2021-06-26 1.700000 mins
##  2 245           2021-07-04 1.950000 mins
##  3 278           2021-07-28 2.116667 mins
##  4 294           2021-08-16 2.266667 mins
##  5 316           2021-08-16 2.450000 mins
##  6 233           2021-06-26 2.483333 mins
##  7 251           2021-07-04 2.516667 mins
##  8 279           2021-07-28 2.516667 mins
##  9 282           2021-07-28 2.516667 mins
## 10 243           2021-07-04 2.533333 mins
## # ... with 264 more rows
```

Replicate Numbers

Replicates are numbered 1-5, so I can check whether the replicate numbers listed for each individual sum to the correct amount, with the exception of the individuals I know do not have 5 replicates on a given day.

```
# proper sum
rep_sum <- sum(1, 2, 3, 4, 5)
rep_sum # 15
```

```
## [1] 15
```

```
# calculate for each individual
all_CEWL_data_edited2 %>%
  group_by(individual_ID, date) %>%
  summarise(rep_sum = sum(as.numeric(replicate_no))) %>%
  dplyr::filter(rep_sum != 15) -> test_rep_nos
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
test_rep_nos
```

```
## # A tibble: 9 x 3
## # Groups:   individual_ID [9]
##   individual_ID date       rep_sum
##   <fct>         <date>       <dbl>
## 1 206           2021-06-24      21
## 2 216           2021-06-24      11
## 3 233           2021-06-26      11
## 4 245           2021-07-04      10
## 5 254           2021-06-26       6
## 6 278           2021-07-28      12
## 7 289           2021-07-28      11
## 8 294           2021-08-16      11
## 9 305           2021-08-16      11
```

```
# compare to my list of known incorrect values
test_rep_nos$individual_ID %in% weird_n$individual_ID
```

```
## [1]  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
```

Individuals 233 (sum 11) and 254 (sum 6) are missing from the weird_n list, but still have an incorrect replicate sum. I just previously discovered that 233 is missing his fourth replicate, and 254 only had three replicates measured before he escaped.

So, every individual on every date has the correct number of and properly labeled replicates. Now the replicates can be interrogated for outliers, then averaged into one observation for each individual on each date.

Replicates

Assess Variation

We want the Coefficient of Variation (CV) among our technical replicates to be small. We need to calculate it to identify whether there may be outliers.

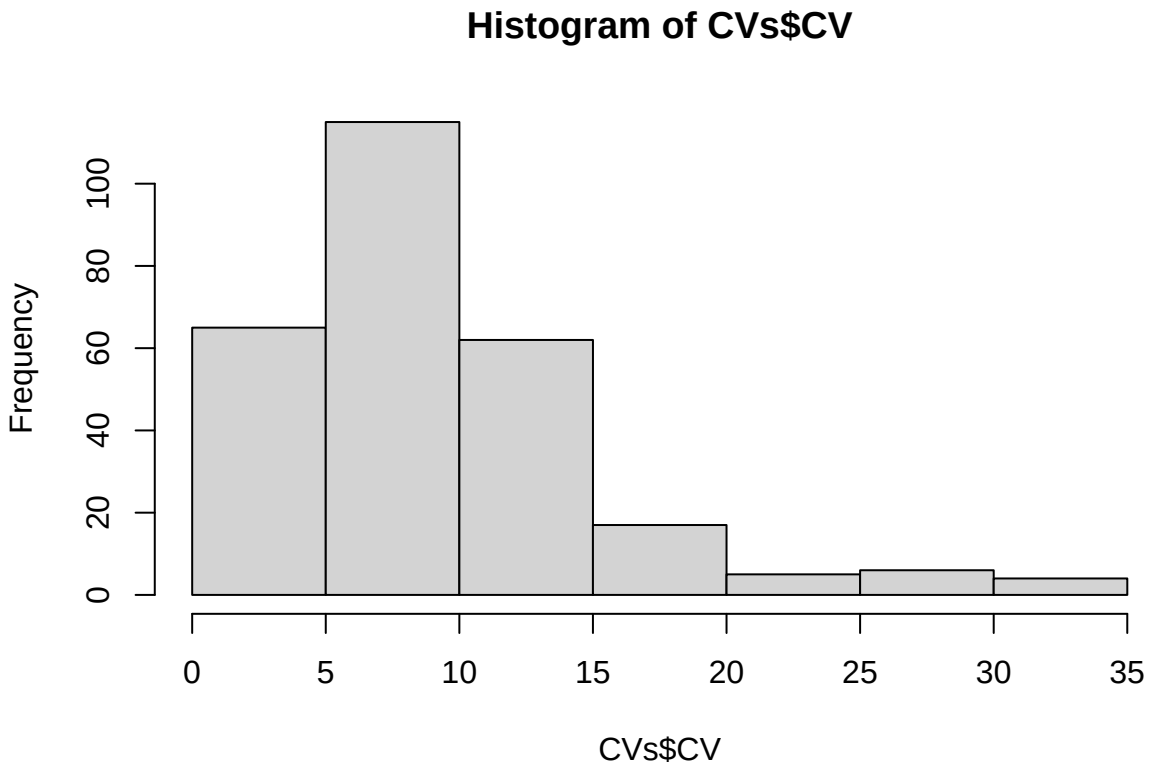
```
CVs <- all_CEWL_data_edited2 %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100,
            min = min(CEWL_g_m2h),
            max = max(CEWL_g_m2h),
            range = max - min
            )
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

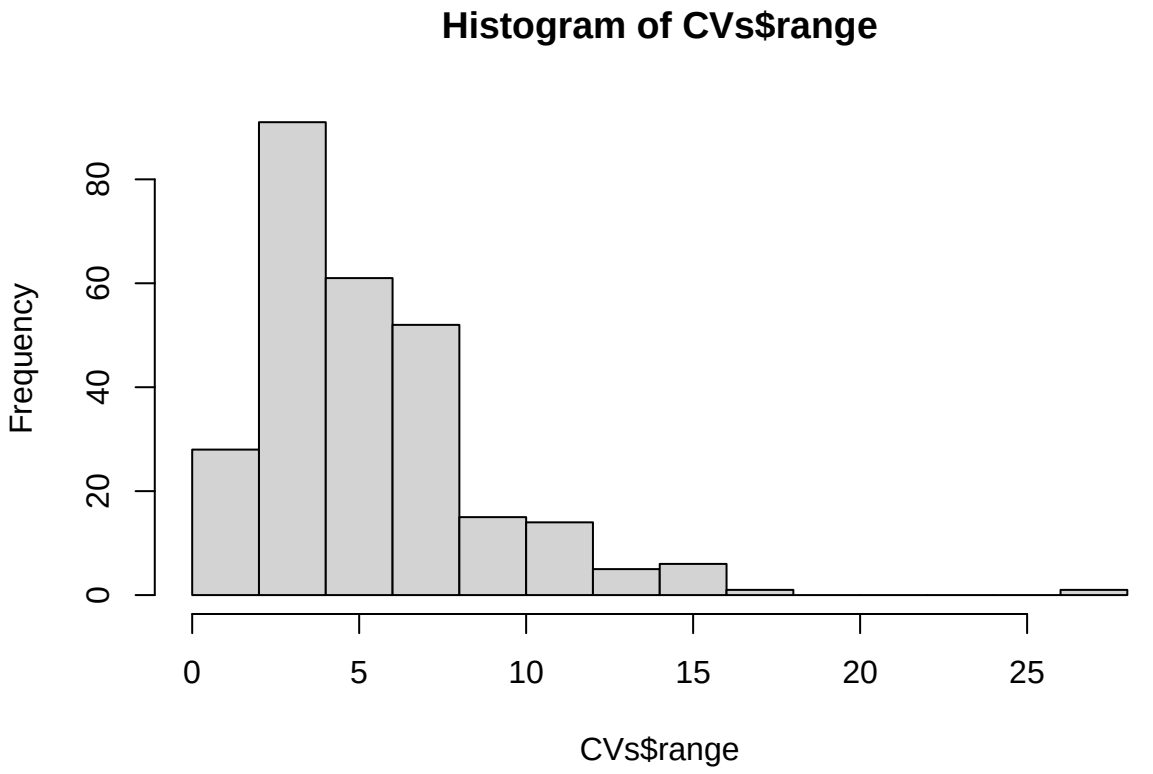
```
summary(CVs)
```

```
##  individual_ID      date                mean              SD
##  201    :  2   Min.   :2021-06-16   Min.   : 7.152   Min.   : 0.4277
##  202    :  2   1st Qu.:2021-06-26   1st Qu.:19.376   1st Qu.: 1.2193
##  203    :  2   Median :2021-07-20   Median :24.110   Median : 1.8015
##  204    :  2   Mean   :2021-07-20   Mean   :24.969   Mean   : 2.1705
##  205    :  2   3rd Qu.:2021-08-08   3rd Qu.:28.616   3rd Qu.: 2.6975
##  206    :  2   Max.   :2021-08-30   Max.   :78.060   Max.   :11.1086
##  (Other):262
##        CV              min              max            range
##  Min.   : 1.465   Min.   : 5.09   Min.   : 8.74   Min.   : 0.840
##  1st Qu.: 5.134   1st Qu.:17.82   1st Qu.:21.64   1st Qu.: 3.038
##  Median : 8.007   Median :21.68   Median :26.64   Median : 4.430
##  Mean   : 9.184   Mean   :22.41   Mean   :27.79   Mean   : 5.379
##  3rd Qu.:11.494   3rd Qu.:25.84   3rd Qu.:31.47   3rd Qu.: 6.960
##  Max.   :32.495   Max.   :73.23   Max.   :81.42   Max.   :26.340
##
```

```
hist(CVs$CV)
```




```
hist(CVs$range)
```



We expect CV for technical replicates to be $<$ 10-15%, so we must determine whether the CVs $>$ 15% are due to outlier replicates.

Find Outliers

First, create a function to look at the replicates for each individual on each day. This will be almost 300 iterations of the function. For each iteration, I will make a boxplot and extract any outliers, compiling a dataframe of outliers that I want to exclude from the final dataset. By printing the boxplots and compiling a dataframe of outliers, I can check the data against the plots to ensure confidence in the outliers quantified.

```
# write function to find outliers for each individual on each date
find_outliers <- function(df) {

  # initiate dataframe to compile info and list to compile plots
  outliers <- data.frame()
  #boxplots <- list()

  # initiate a for loop to go through every who in df
  for(indiv_ch in unique(df$individual_ID)) {

    # select data for only the individual of interest
    df_sub <- df %>%
      dplyr::filter(individual_ID == as.numeric(indiv_ch))

    # make a boxplot
    df_sub %>%
      ggplot(.) +
      geom_boxplot(aes(x = as.factor(date),
                       y = CEWL_g_m2h,
                       fill = as.factor(date))) +
      ggtitle(paste("Individual", indiv_ch)) +
      theme_classic() -> plot

    # print/save
    print(plot)
    #boxplots[[indiv_ch]] <- plot

    # extract outliers
    outs <- df_sub %>%
      group_by(individual_ID, date) %>%
      summarise(outs = boxplot.stats(CEWL_g_m2h)$out)

    # add to running dataframe of outliers
    outliers <- outliers %>%
      rbind(outs)
  }
  #return(boxplots)
  return(outliers)
}
```

Now apply the function to the data:

```
par(mfrow = c(71, 2))
outliers_found <- find_outliers(all_CEWL_data_edited2)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

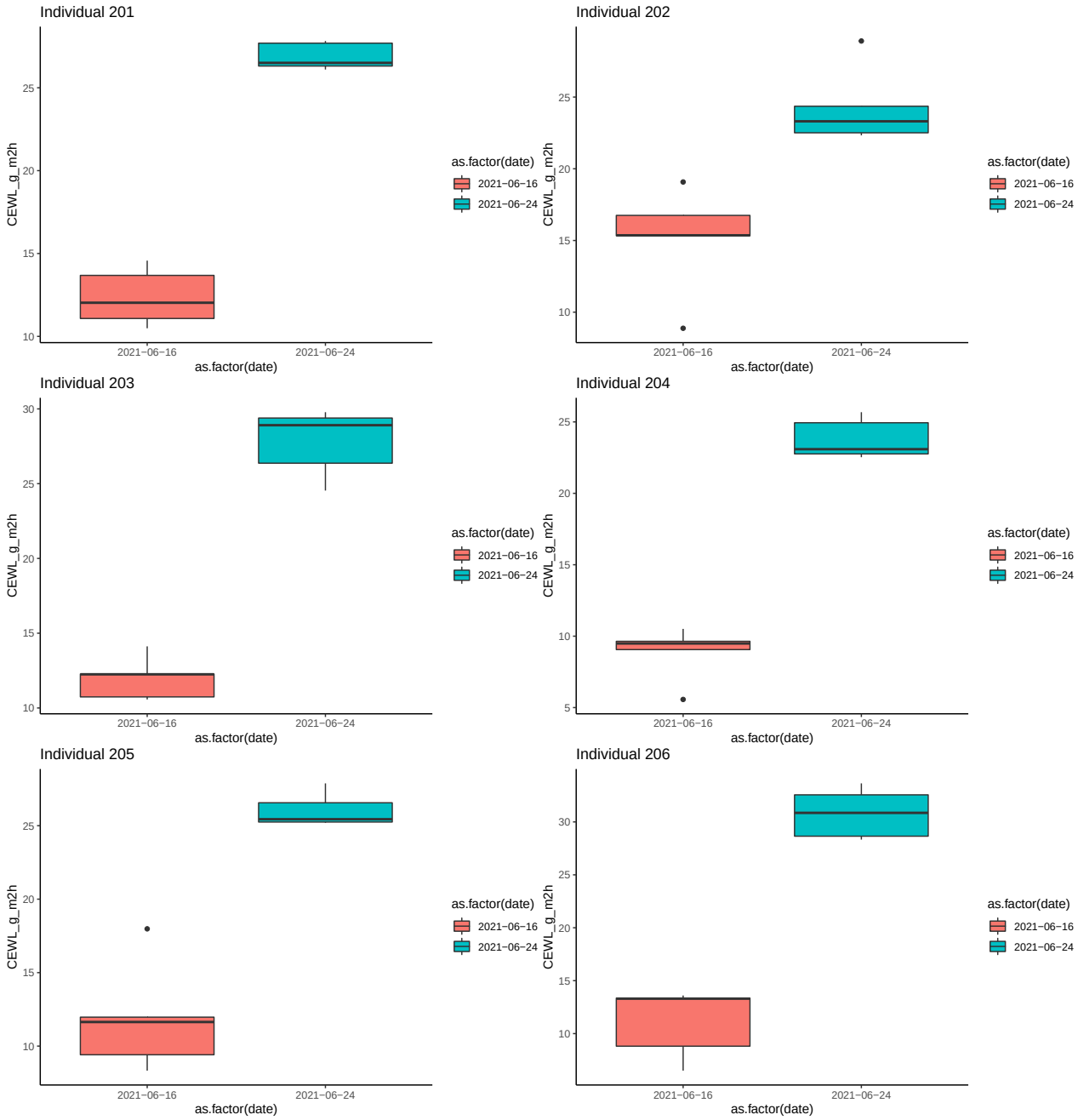
```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

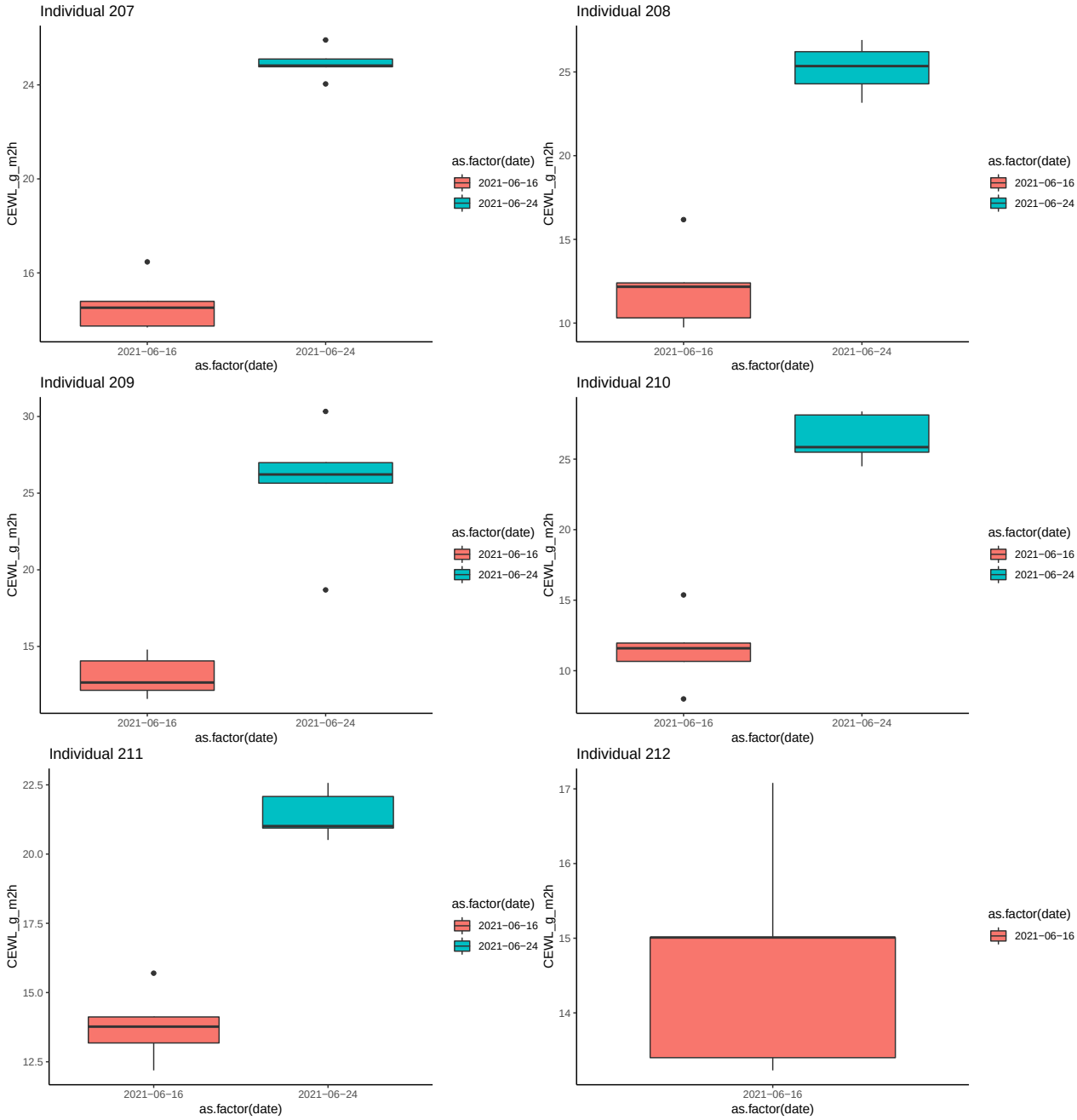
```
outliers_found
```

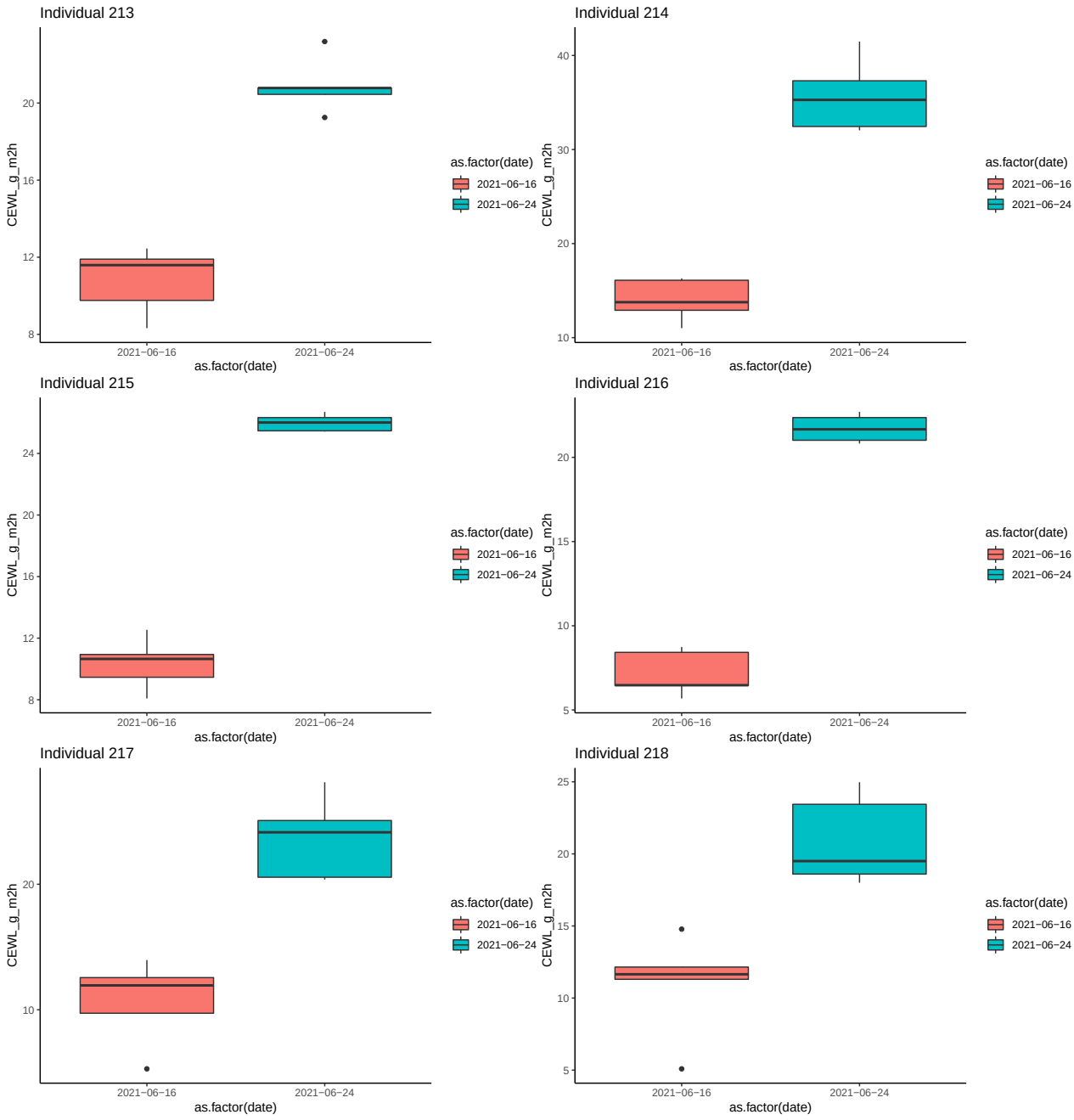
```
## # A tibble: 134 x 3
## # Groups:   individual_ID, date [101]
##    individual_ID date        outs
##    <fct>         <date>     <dbl>
##  1 202           2021-06-16  8.88
##  2 202           2021-06-16 19.1
##  3 202           2021-06-24 28.9
##  4 204           2021-06-16  5.57
##  5 205           2021-06-16 18.0
##  6 207           2021-06-16 16.5
##  7 207           2021-06-24 25.9
##  8 207           2021-06-24 24.0
##  9 208           2021-06-16 16.2
## 10 209           2021-06-24 30.3
```

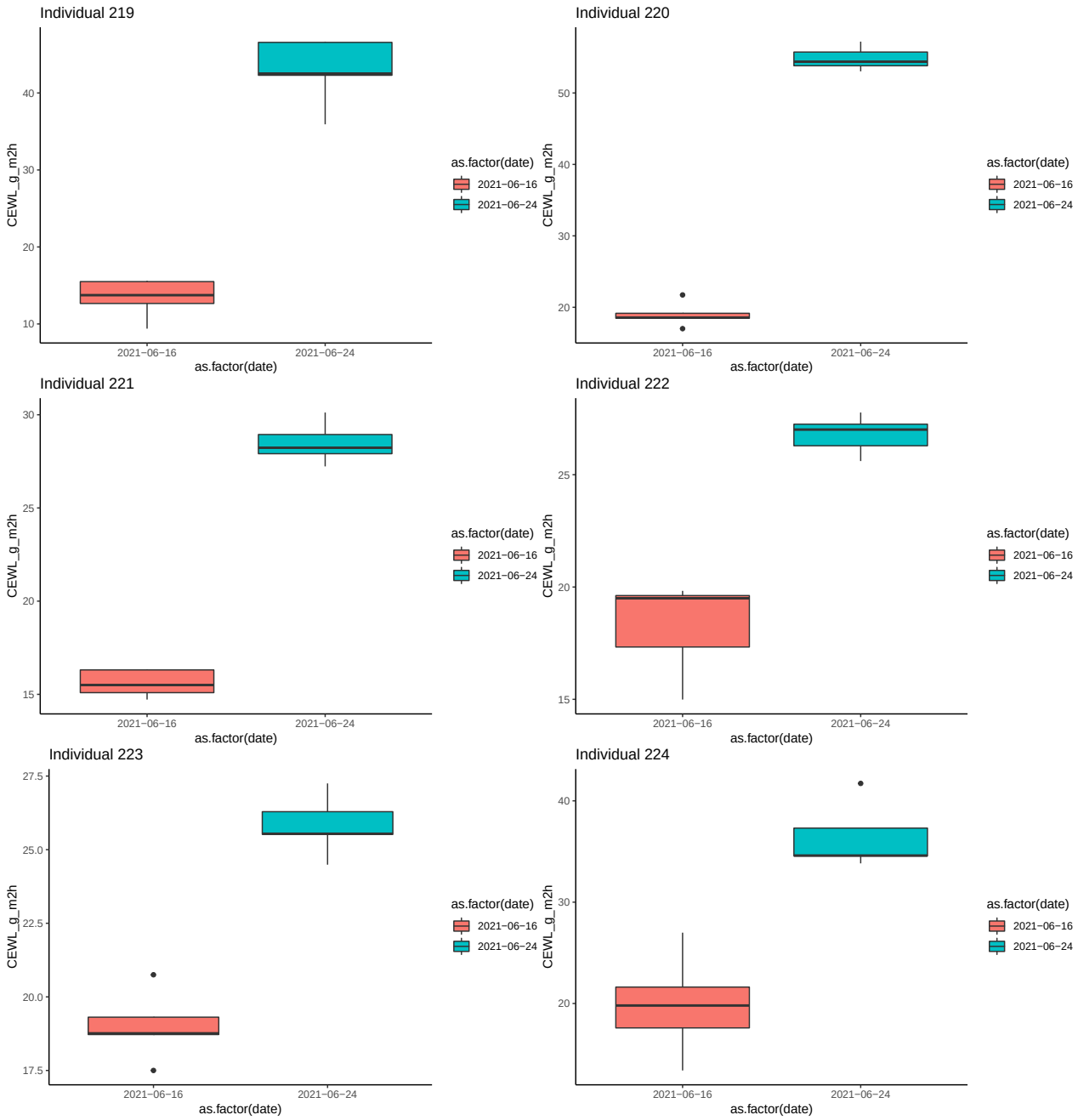
```
## # ... with 124 more rows
```

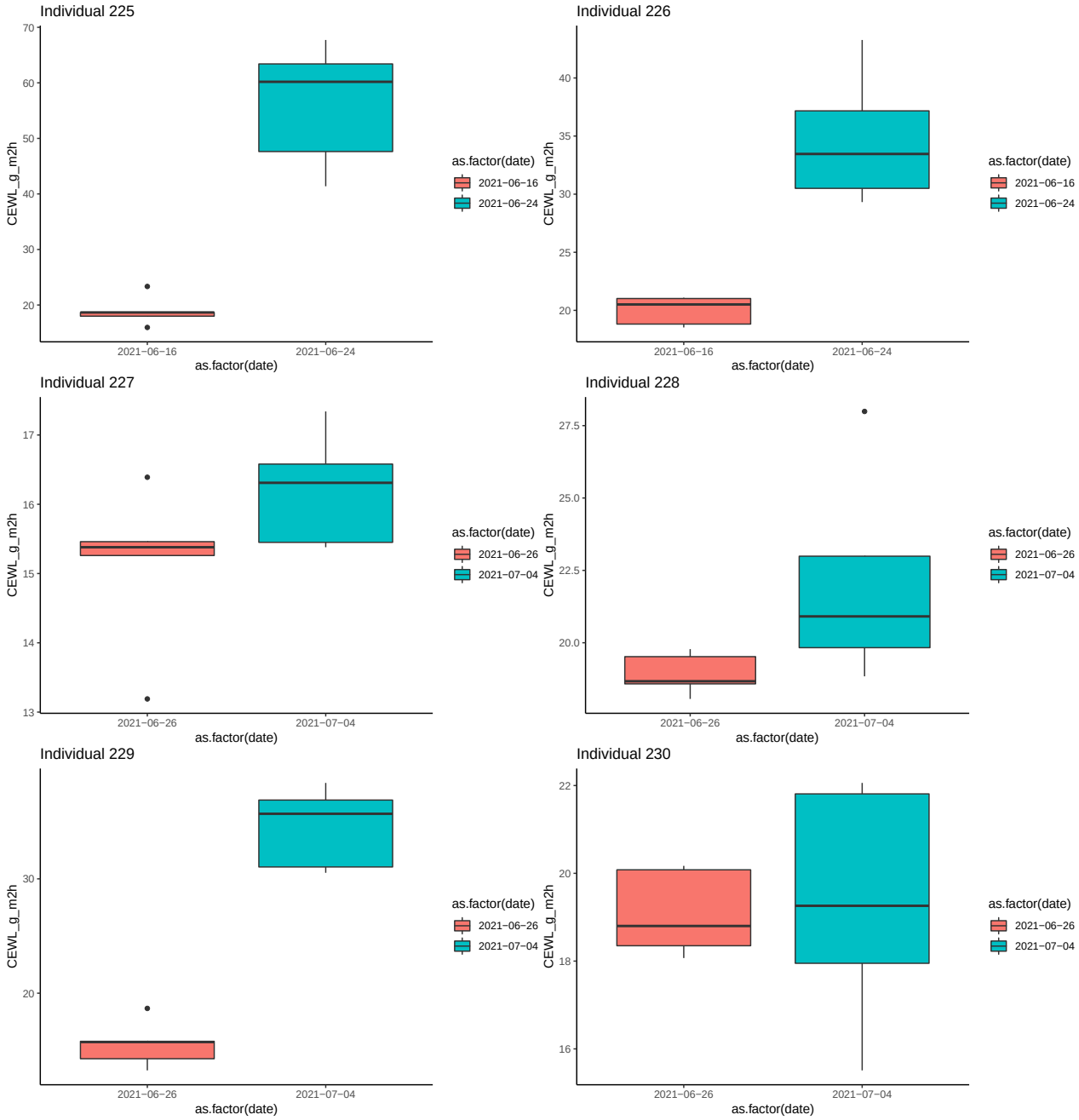
```
par(mfrow = c(1, 1))
```

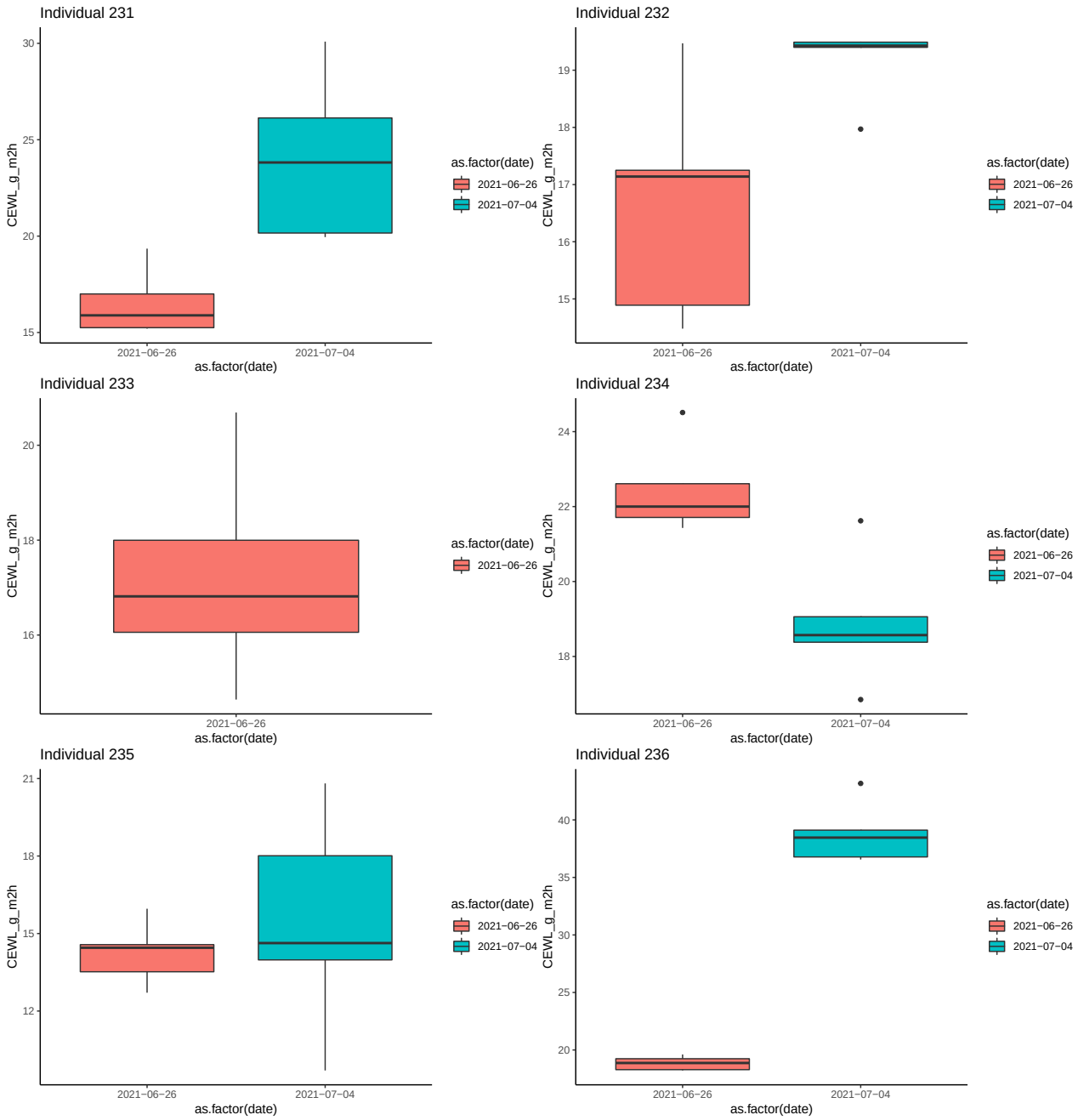


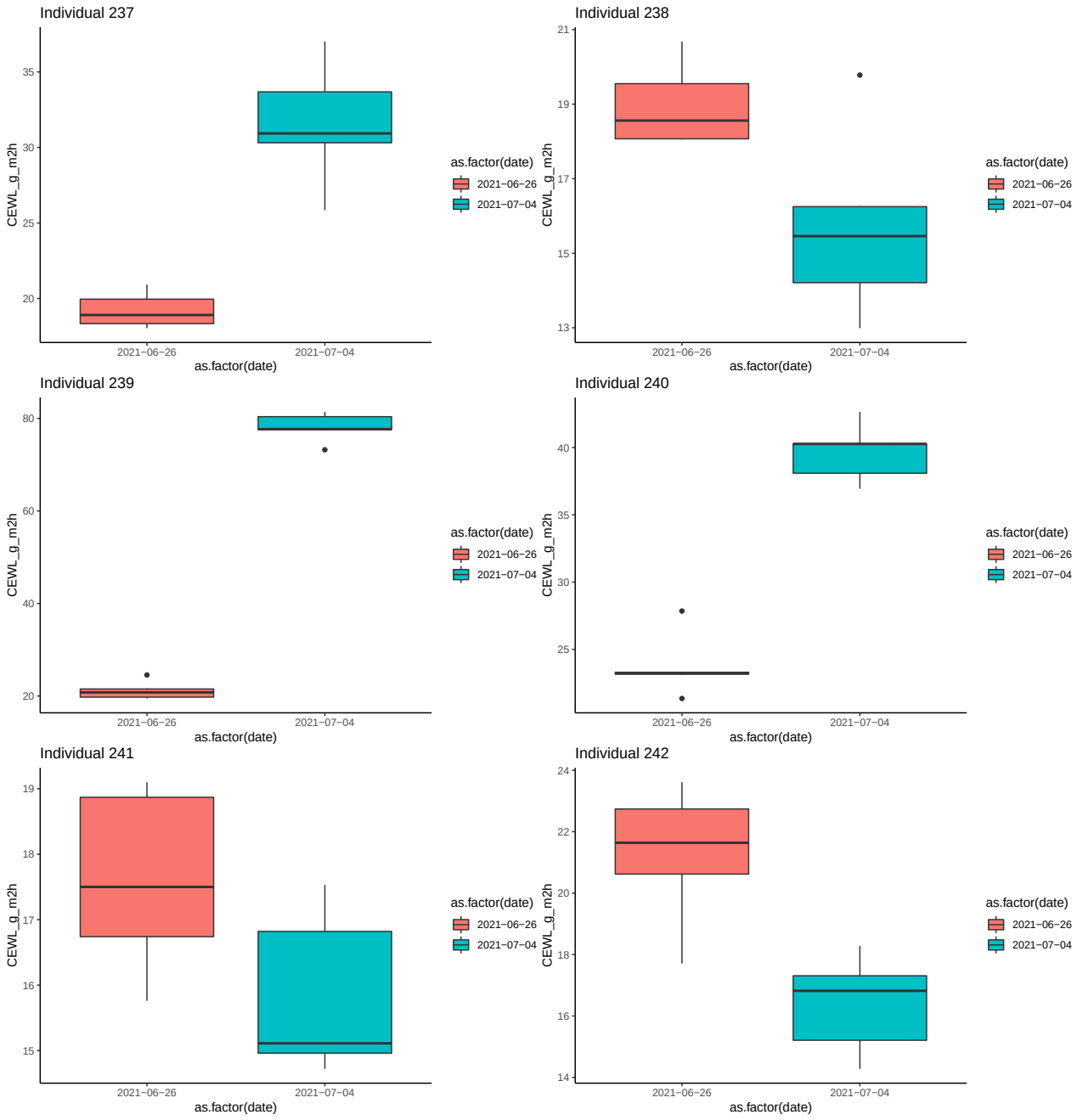

Individual 207
Individual 208
Individual 209
Individual 210
Individual 211
Individual 212
CEWL_g_m2h
as.factor(date)
2021−06−16
2021−06−24

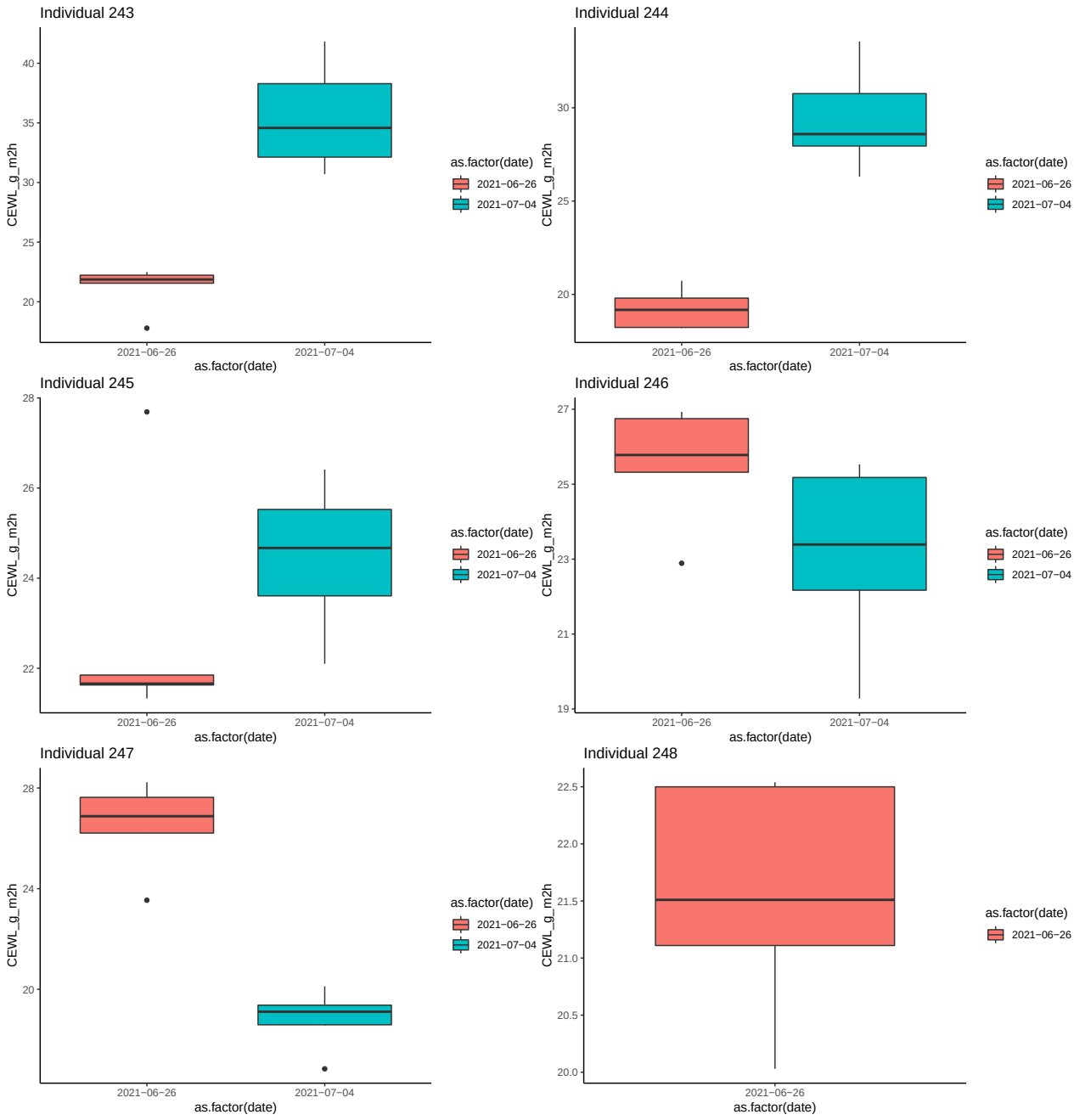

Individual 213
Individual 214
Individual 215
Individual 216
Individual 217
Individual 218
CEWL_g_m2h
as.factor(date)
2021−06−16
2021−06−24

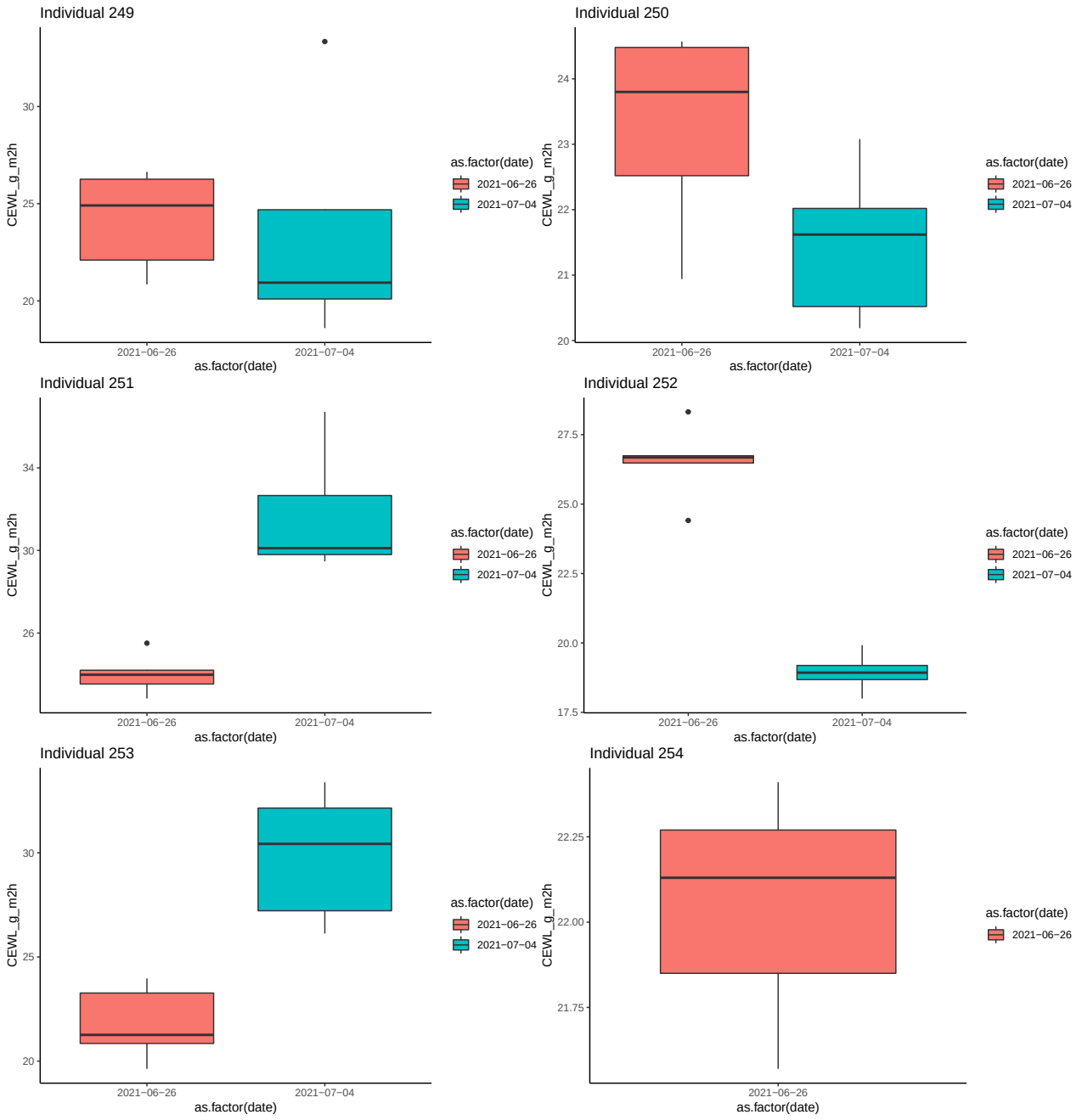

Individual 219
Individual 220
Individual 221
Individual 222
Individual 223
Individual 224
CEWL_g_m2h
as.factor(date)
2021−06−16
2021−06−24

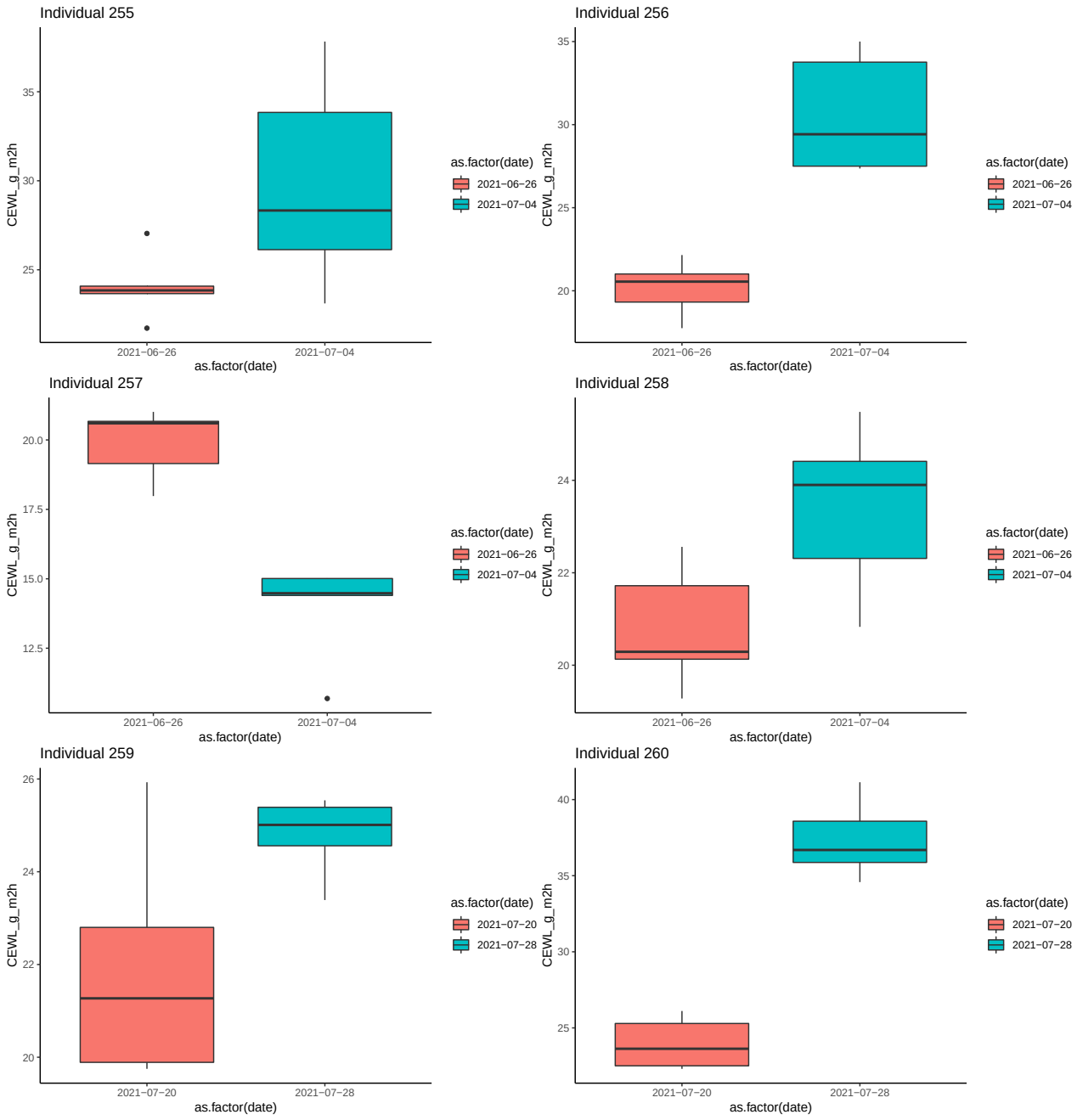

Individual 225
Individual 226
Individual 227
Individual 228
Individual 229
Individual 230
CEWL_g_m2h
as.factor(date)
2021−06−16
2021−06−24
2021−06−26
2021−07−04

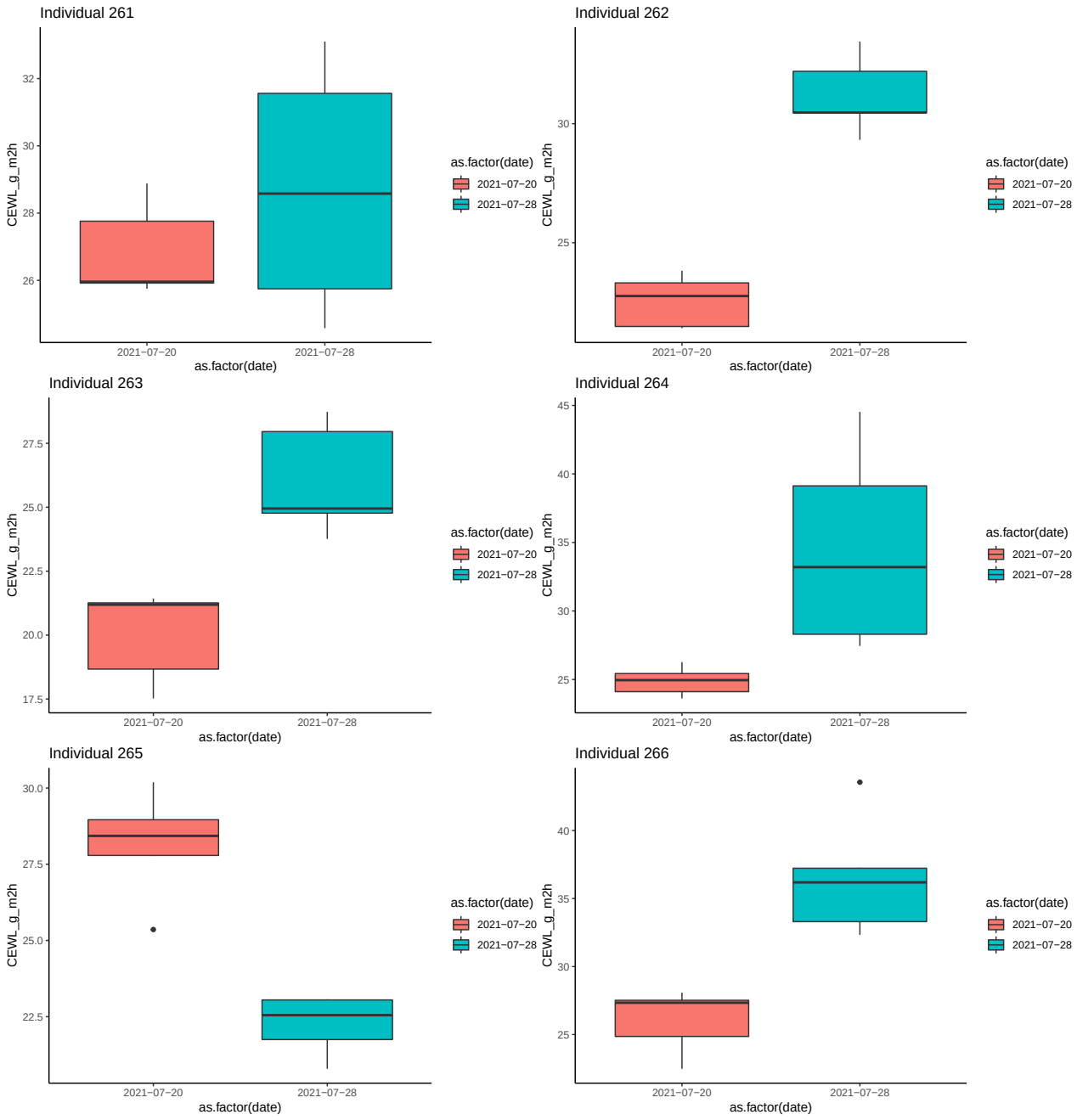

Individual 231
Individual 232
Individual 233
Individual 234
Individual 235
Individual 236
CEWL_g_m2h
as.factor(date)
2021−06−26
2021−07−04

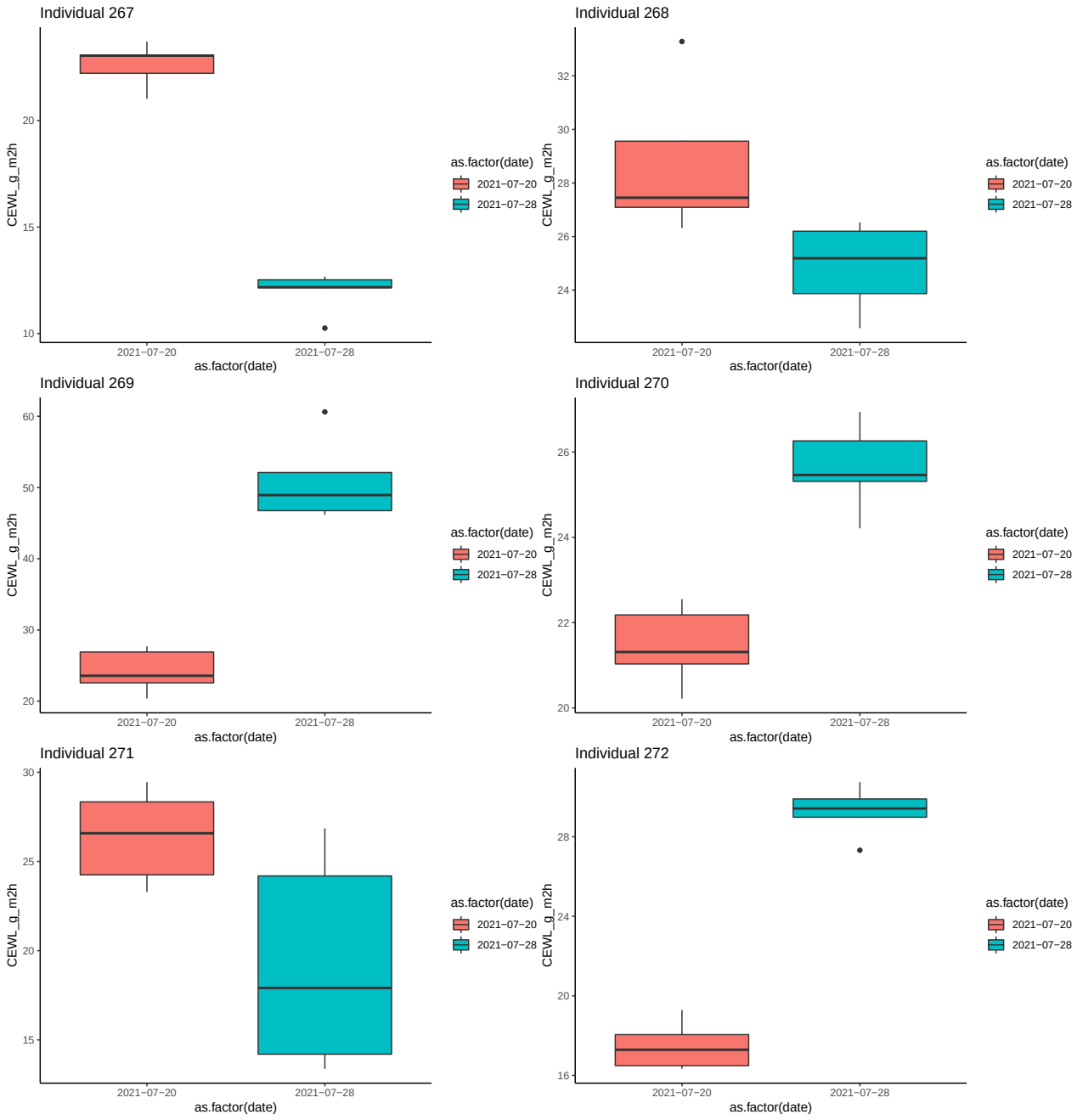

Individual 237
Individual 238
Individual 239
Individual 240
Individual 241
Individual 242
CEWL_g_m2h
as.factor(date)
2021−06−26
2021−07−04

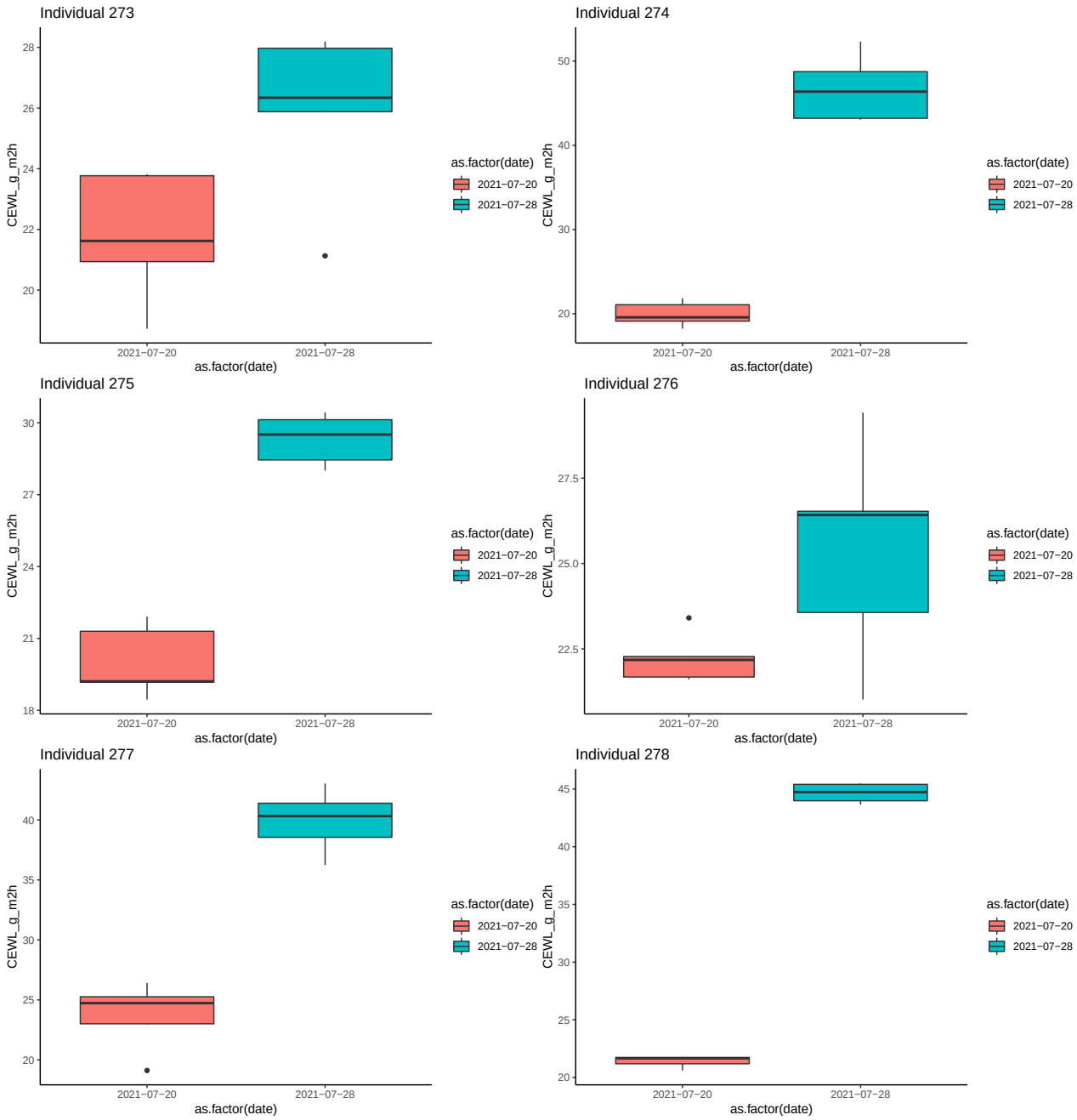

Individual 243
Individual 244
Individual 245
Individual 246
Individual 247
Individual 248
CEWL_g_m2h
as.factor(date)
2021−06−26
2021−07−04

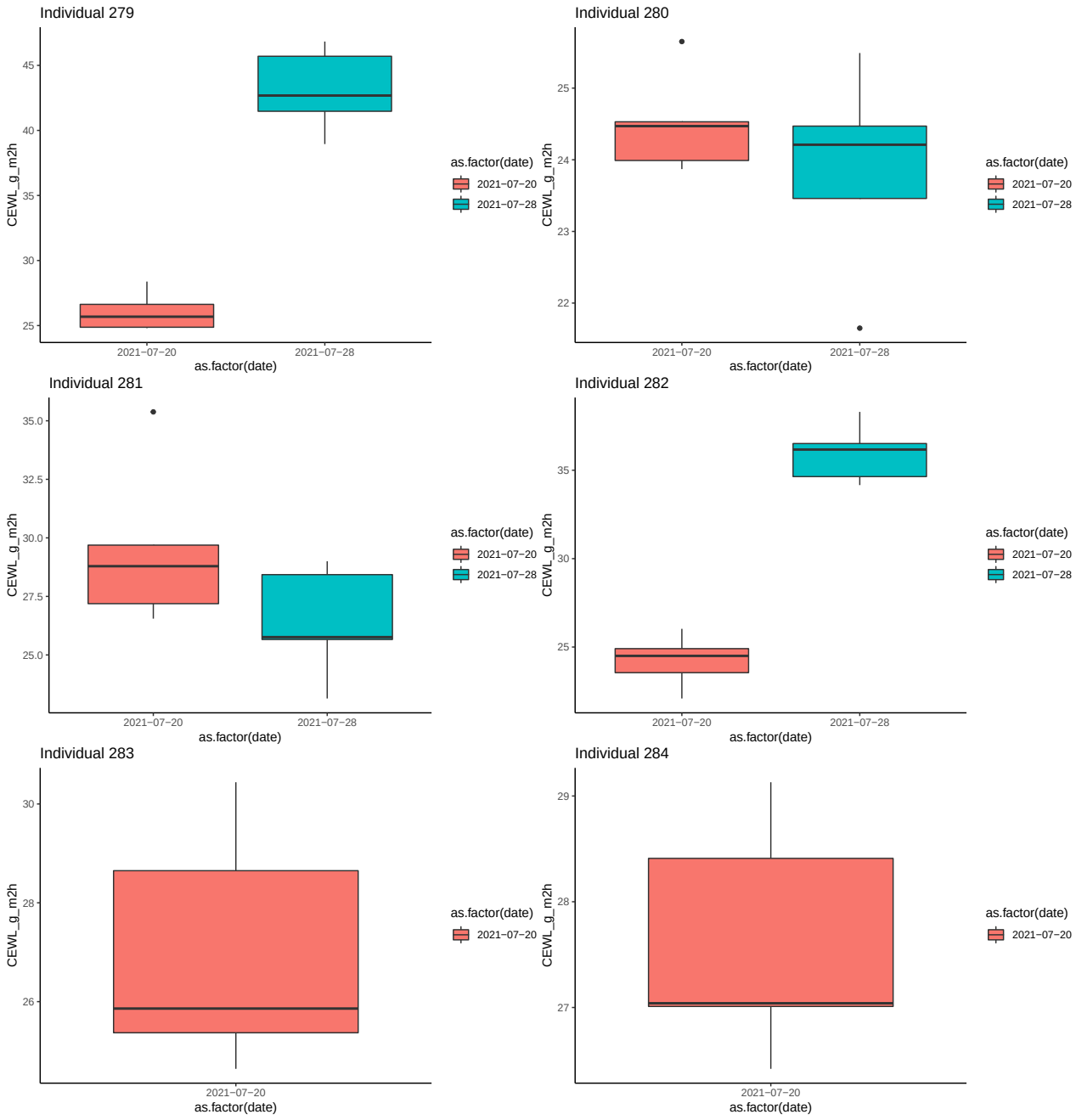

Individual 249
Individual 250
Individual 251
Individual 252
Individual 253
Individual 254
CEWL_g_m2h
as.factor(date)
2021−06−26
2021−07−04

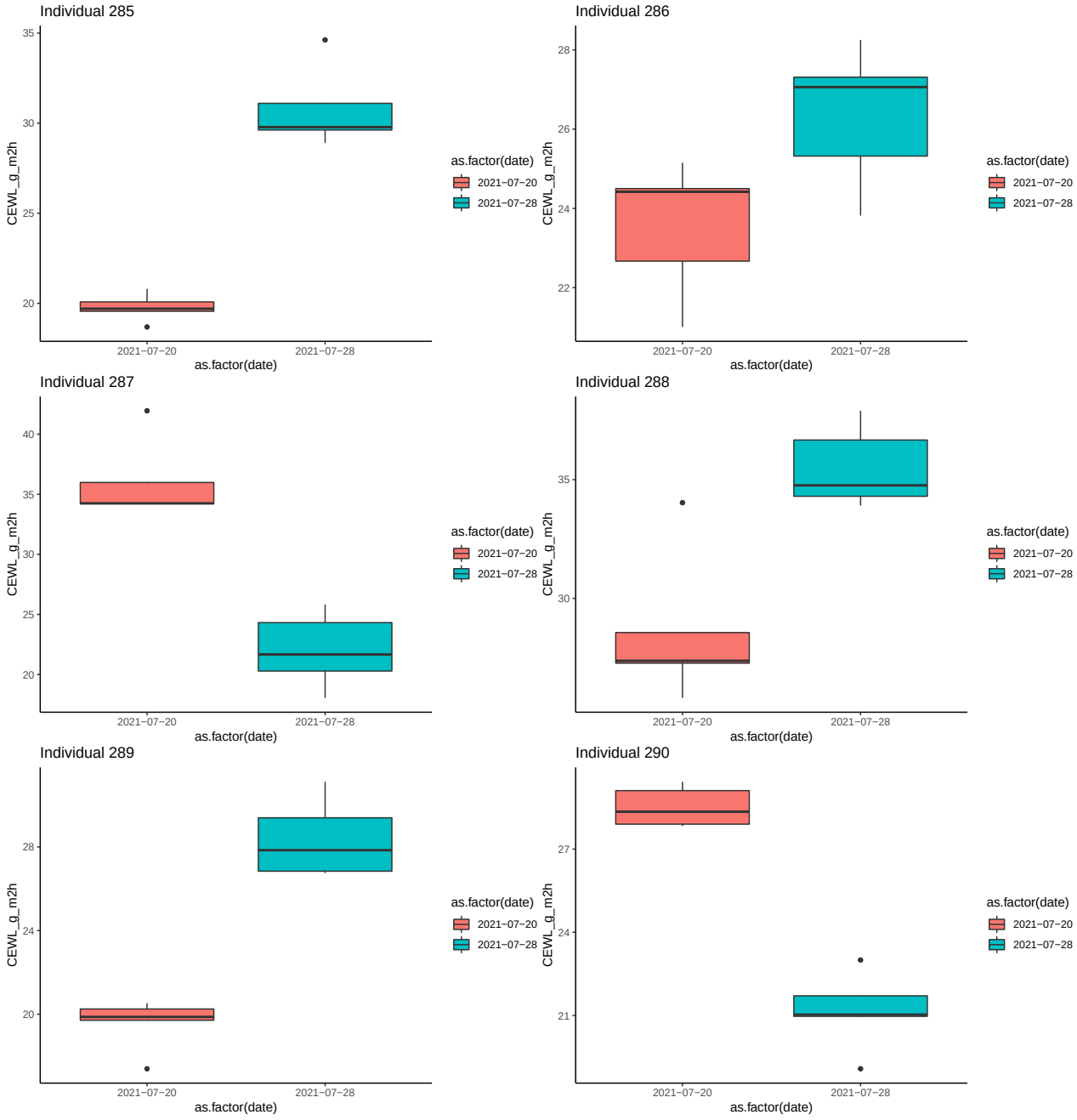

Individual 255
Individual 256
Individual 257
Individual 258
Individual 259
Individual 260
CEWL_g_m2h
as.factor(date)
2021−06−26
2021−07−04
2021−07−20
2021−07−28

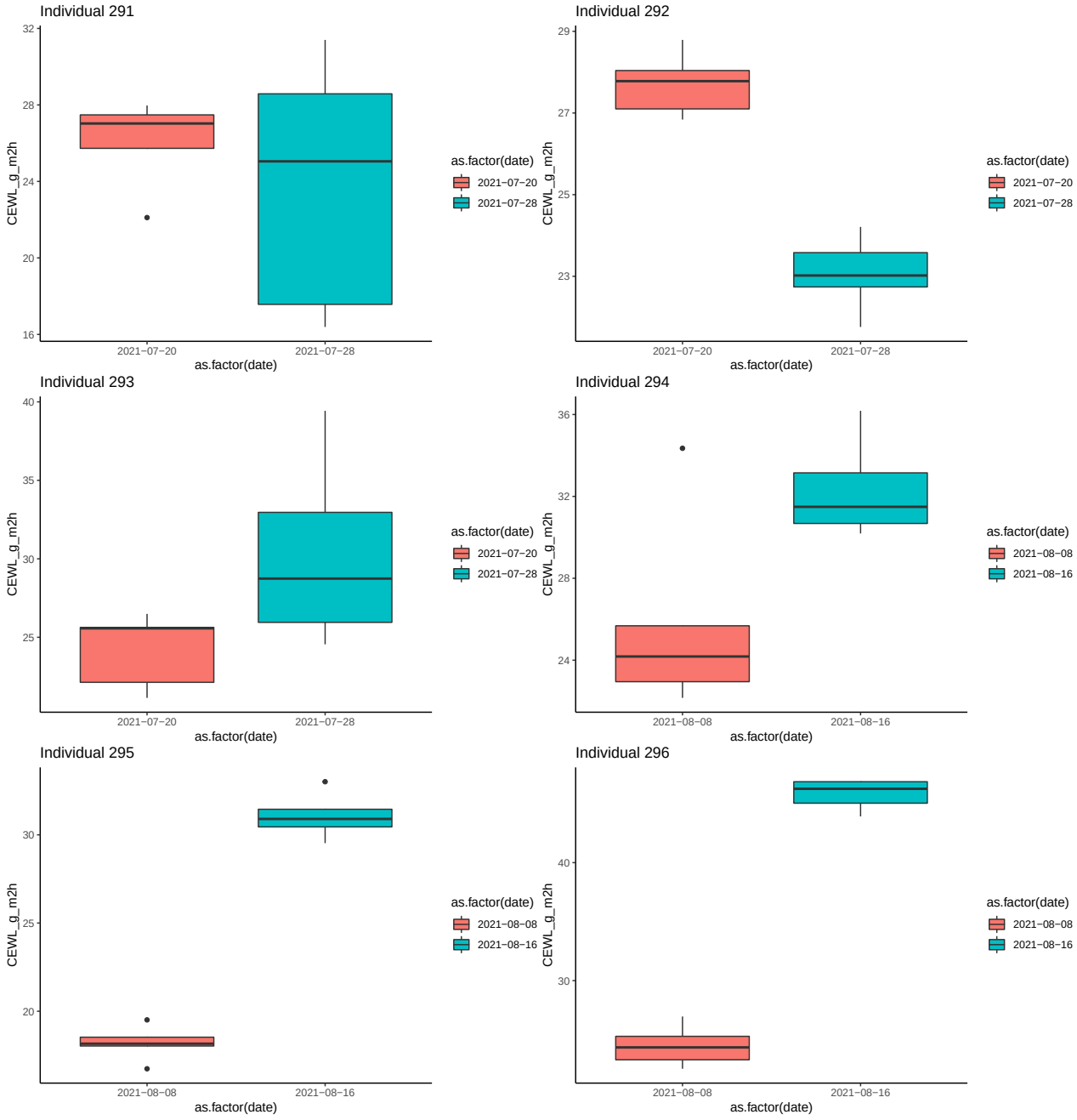

Individual 261
Individual 262
Individual 263
Individual 264
Individual 265
Individual 266
CEWL_g_m2h
as.factor(date)
2021−07−20
2021−07−28

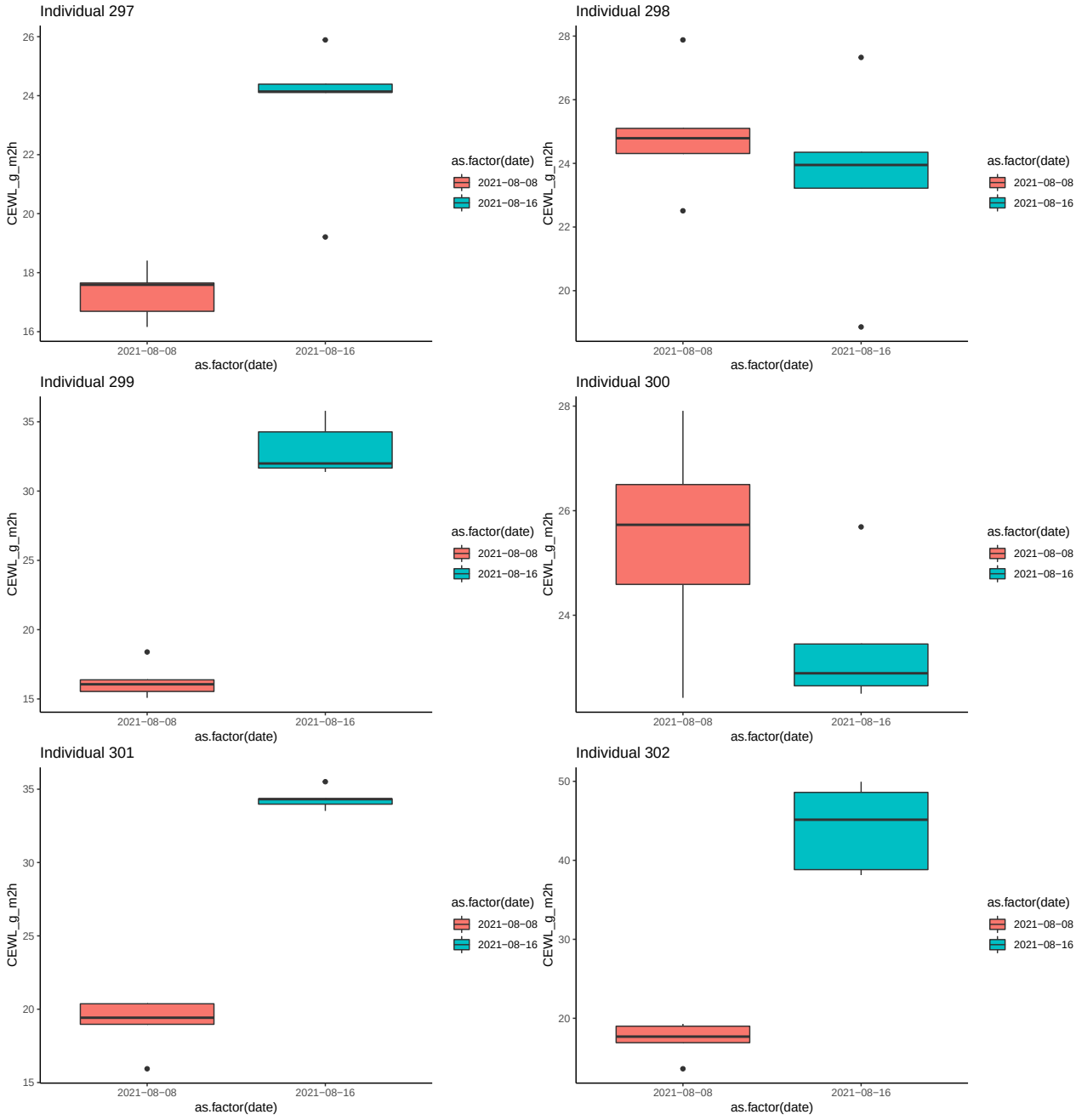

Individual 267
Individual 268
Individual 269
Individual 270
Individual 271
Individual 272
CEWL_g_m2h
as.factor(date)
2021−07−20
2021−07−28

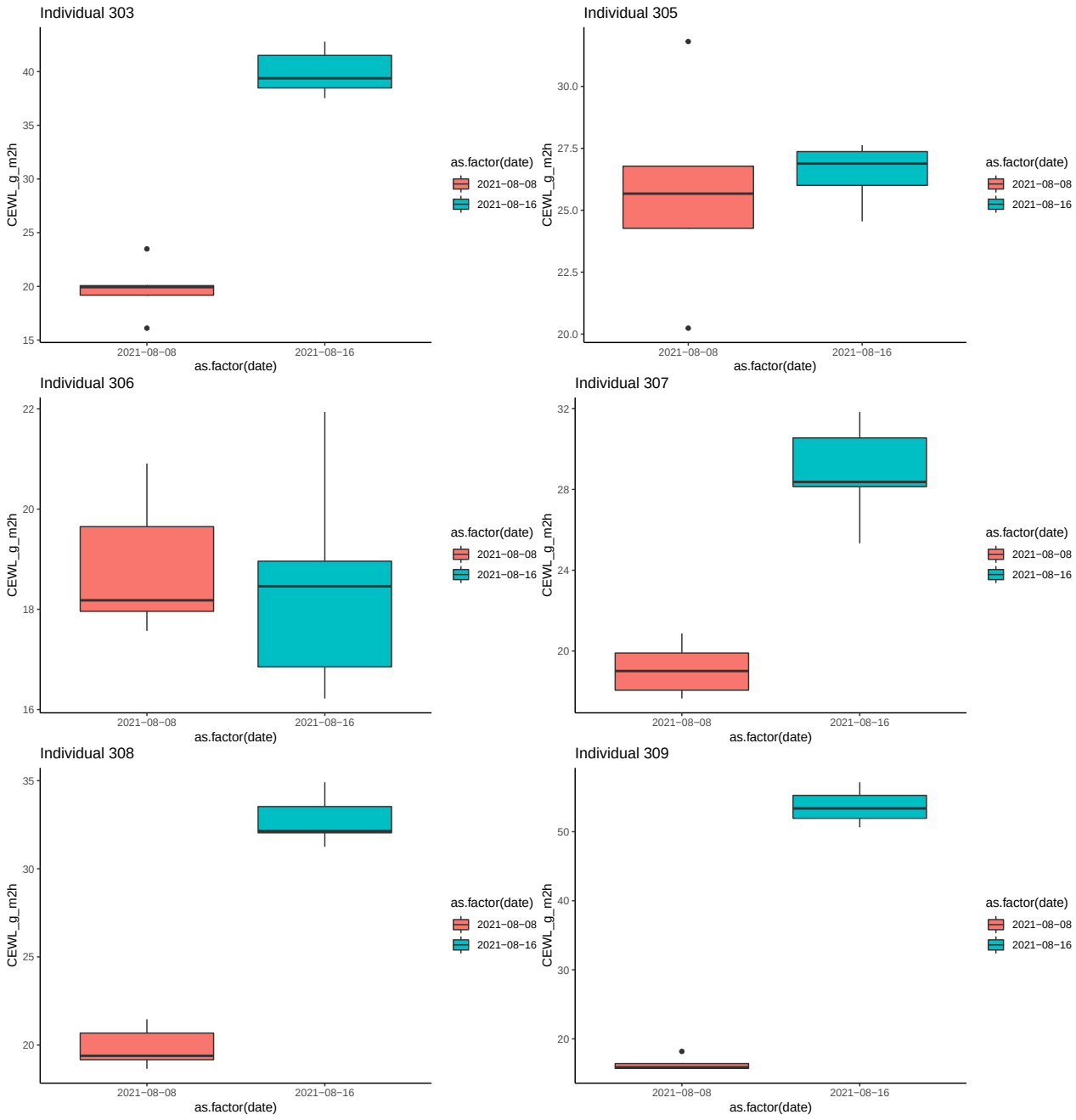

Individual 273
Individual 274
Individual 275
Individual 276
Individual 277
Individual 278
CEWL_g_m2h
as.factor(date)
2021−07−20
2021−07−28

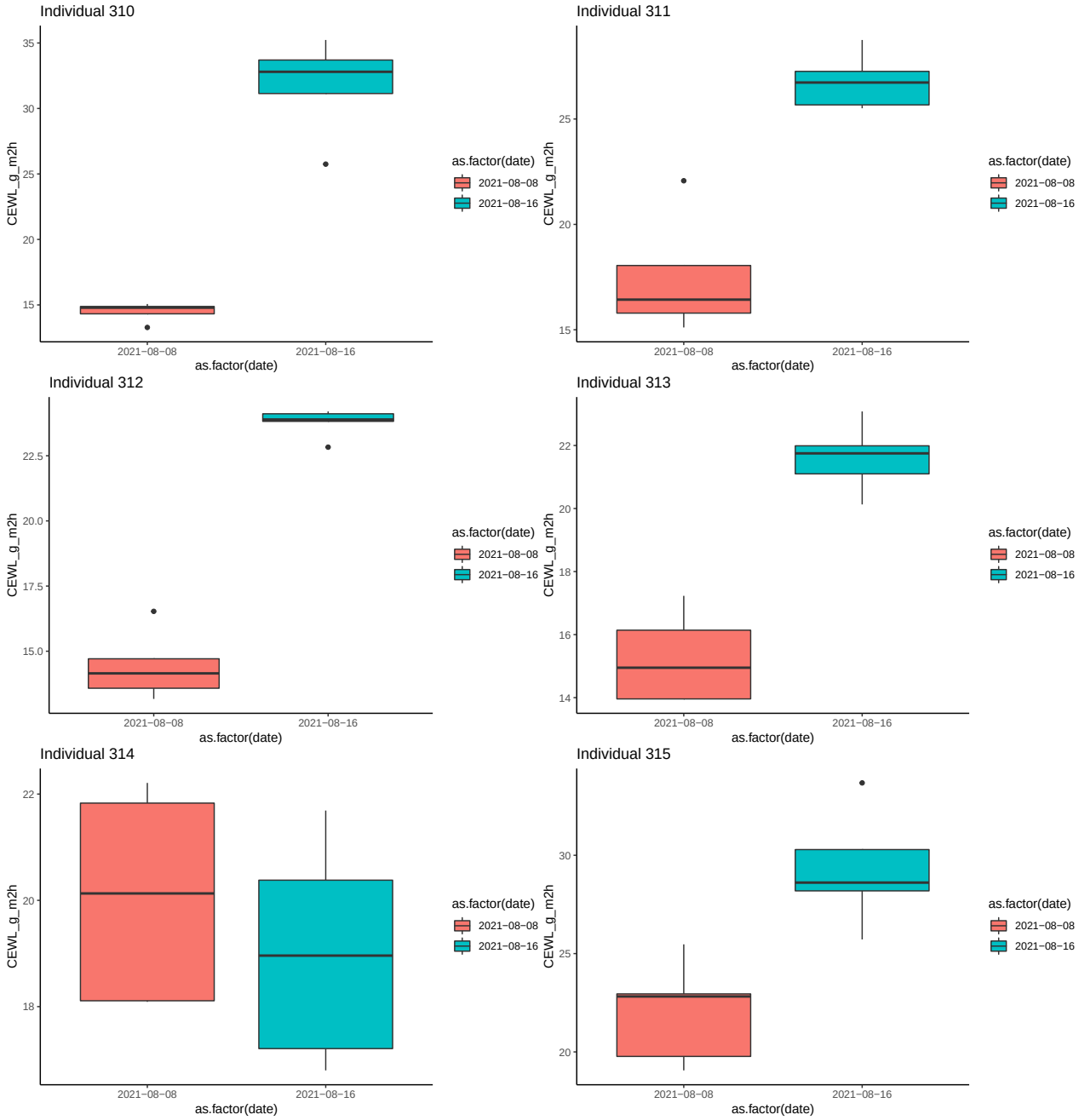

Individual 279
Individual 280
Individual 281
Individual 282
Individual 283
Individual 284
CEWL_g_m2h
as.factor(date)
2021−07−20
2021−07−28

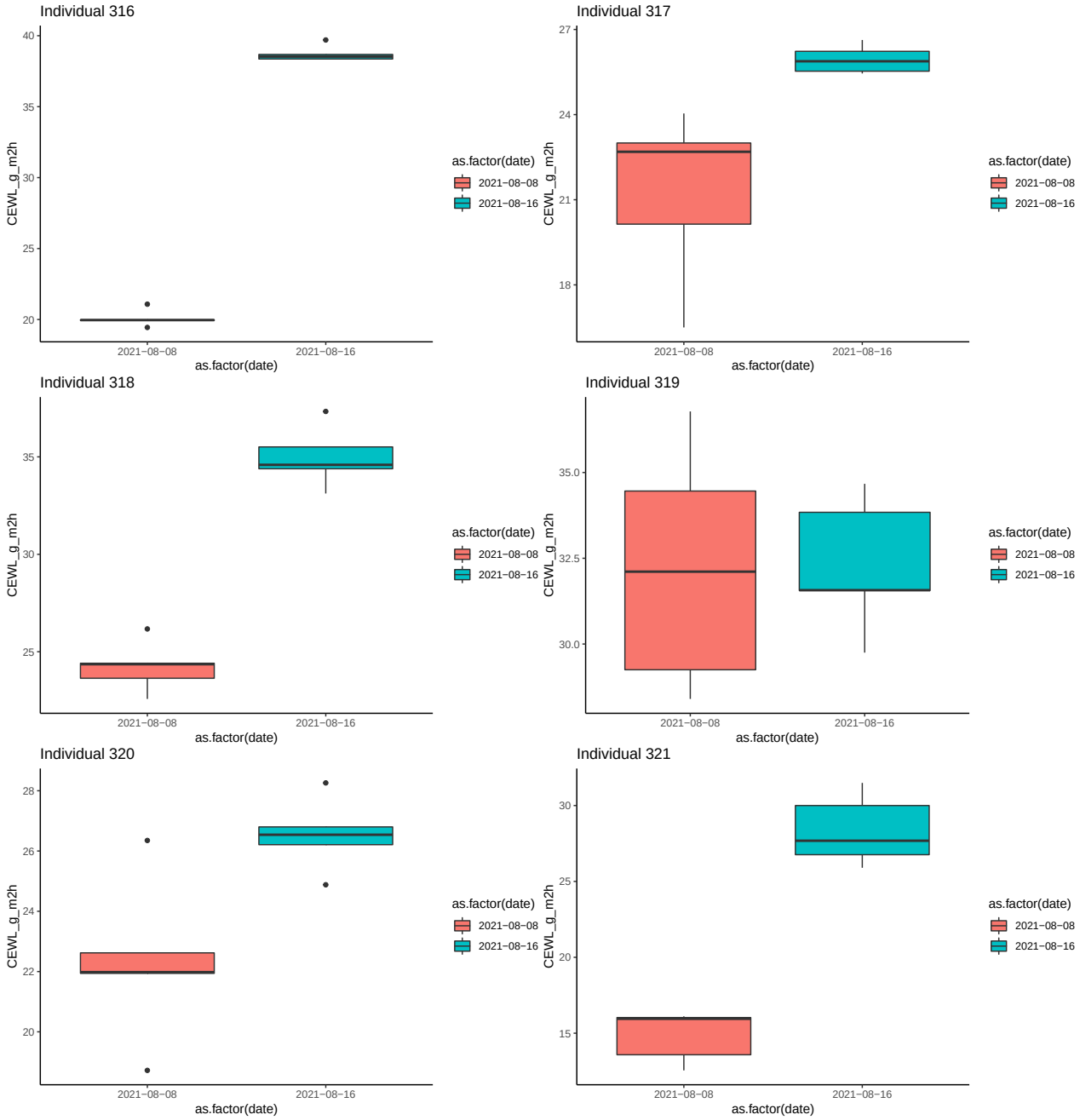

Individual 285
Individual 286
Individual 287
Individual 288
Individual 289
Individual 290
CEWL_g_m2h
as.factor(date)
2021−07−20
2021−07−28

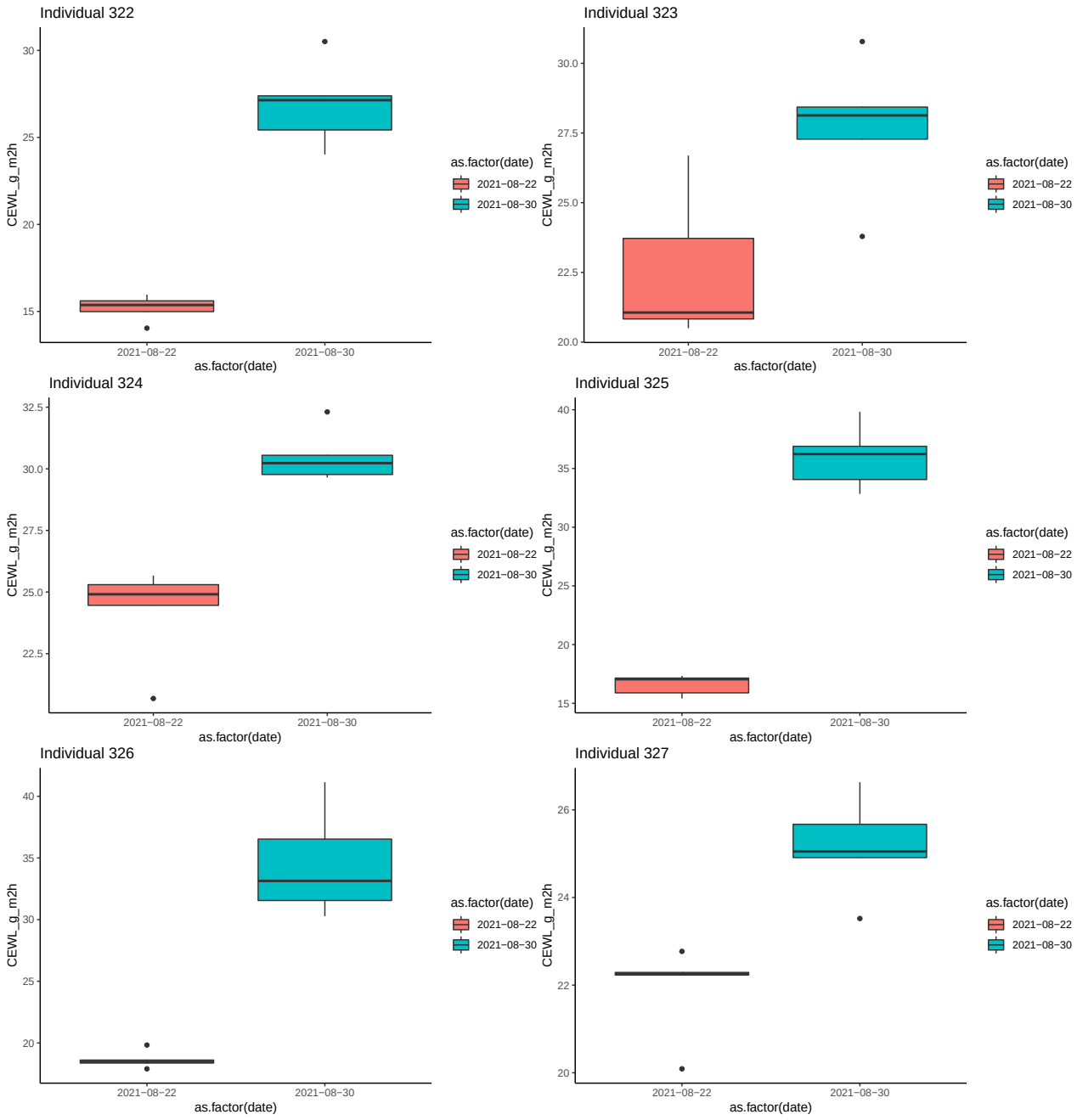

Individual 291
CEWL_g_m2h
as.factor(date)
2021−07−20
2021−07−28
Individual 292
Individual 293
Individual 294
2021−08−08
2021−08−16
Individual 295
Individual 296

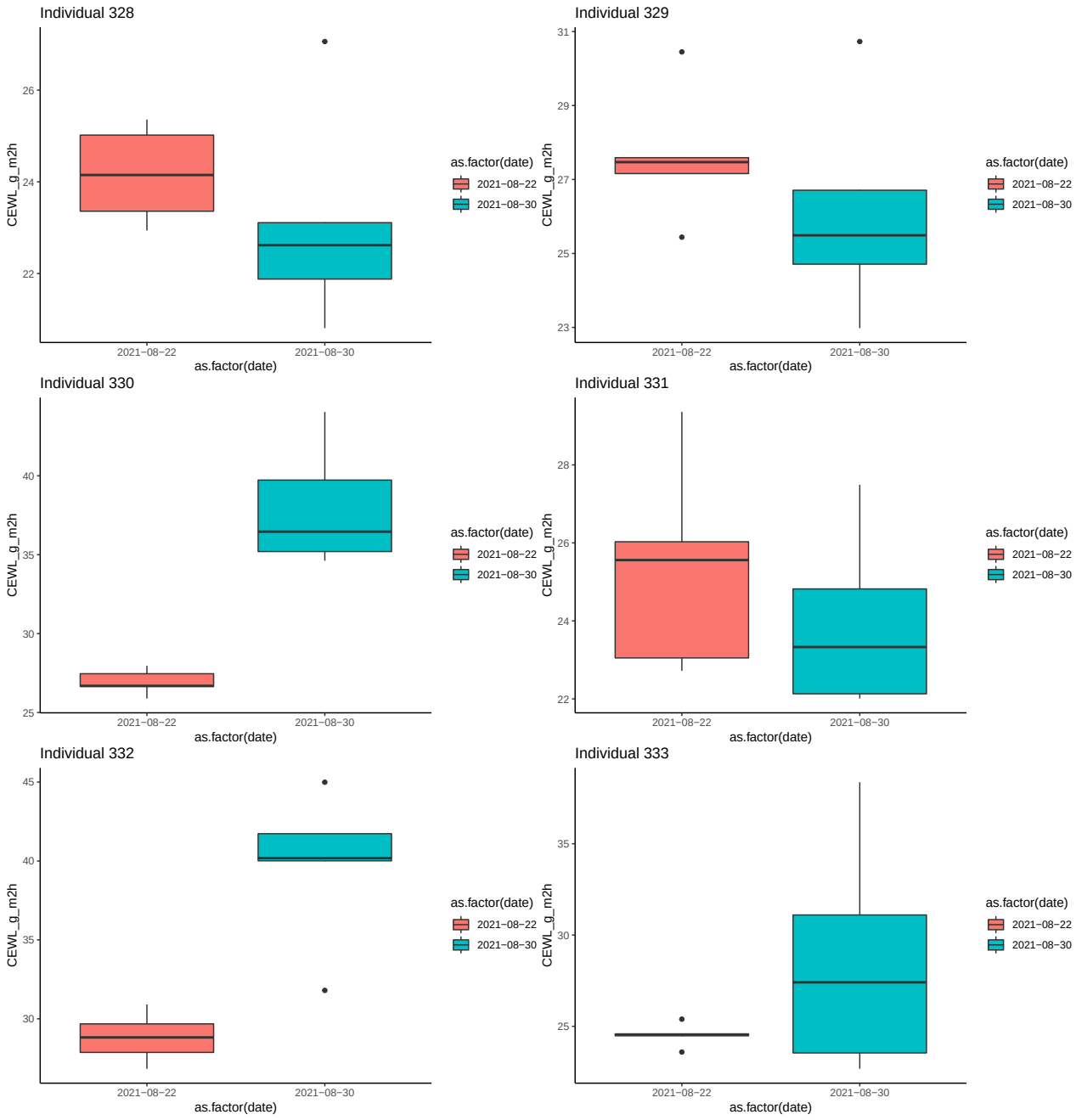

Individual 297
Individual 298
Individual 299
Individual 300
Individual 301
Individual 302
CEWL_g_m2h
as.factor(date)
2021−08−08
2021−08−16

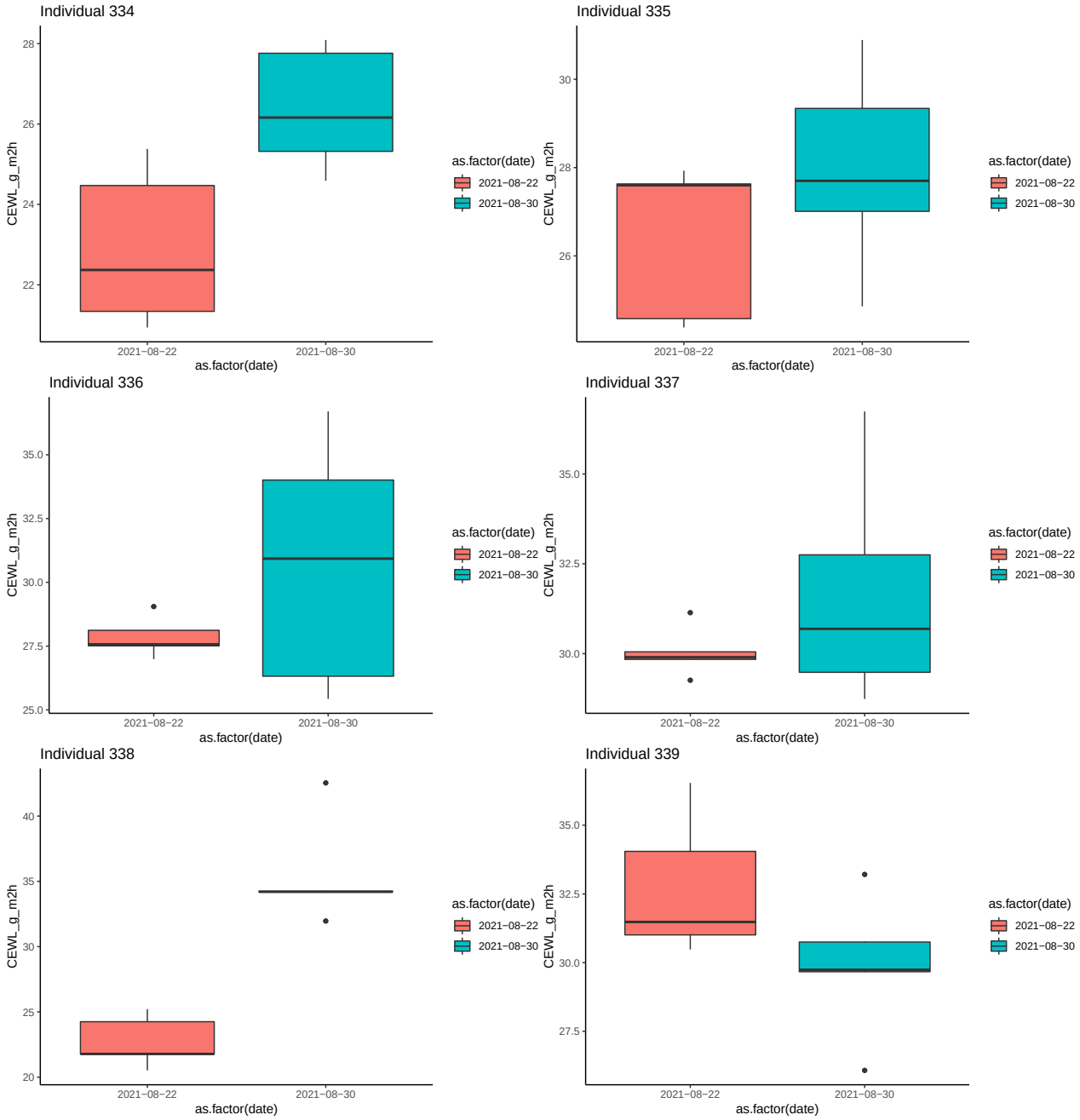

Individual 303
Individual 305
Individual 306
Individual 307
Individual 308
Individual 309
CEWL_g_m2h
as.factor(date)
2021−08−08
2021−08−16

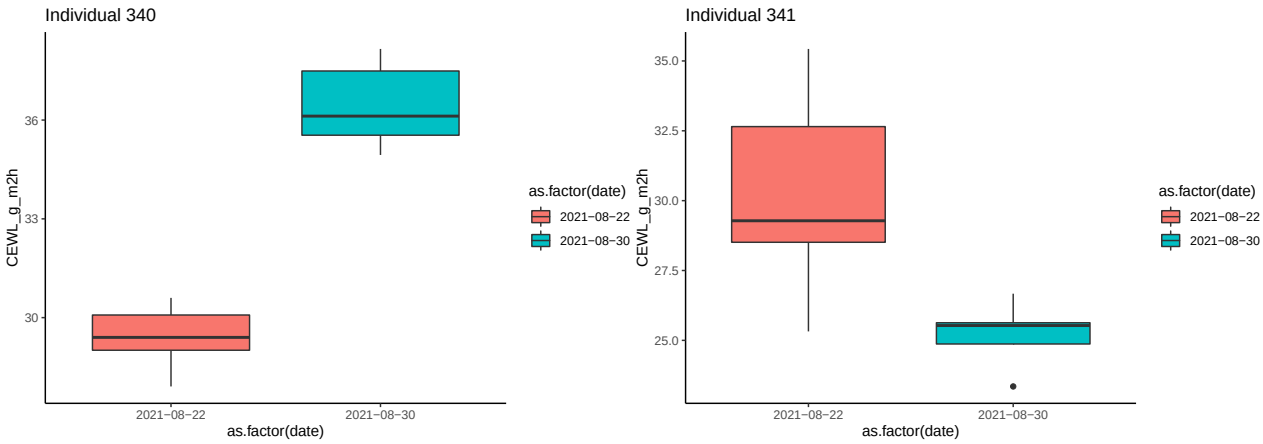

Individual 310
Individual 311
Individual 312
Individual 313
Individual 314
Individual 315
CEWL_g_m2h
as.factor(date)
2021−08−08
2021−08−16


Individual 316
Individual 317
Individual 318
Individual 319
Individual 320
Individual 321
CEWL_g_m2h
as.factor(date)
2021−08−08
2021−08−16


Individual 322
Individual 323
Individual 324
Individual 325
Individual 326
Individual 327
CEWL_g_m2h
as.factor(date)
2021−08−22
2021−08−30


Individual 328
Individual 329
Individual 330
Individual 331
Individual 332
Individual 333
CEWL_g_m2h
as.factor(date)
2021−08−22
2021−08−30


Individual 334
CEWL_g_m2h
as.factor(date)
2021−08−22
2021−08−30
Individual 335
Individual 336
Individual 337
Individual 338
Individual 339



Based on the plots, the dataframe of outliers I compiled is correct.

Remove Outliers

Now I will create a secondary version of the same function, but instead of compiling outliers, I will omit them from the dataset.

```
# write function to find and exclude outliers
omit_outliers <- function(df) {

  # initiate dataframe to compile info and list to compile plots
  cleaned <- data.frame()

  # initiate a for loop to go through every who in df
  for(indiv_ch in unique(df$individual_ID)) {

    # select data for only the individual of interest
    df_sub <- df %>%
      dplyr::filter(individual_ID == as.numeric(indiv_ch))

    # extract outliers
    outs <- df_sub %>%
      group_by(individual_ID, date) %>%
      summarise(outs = boxplot.stats(CEWL_g_m2h)$out)

    # filter outliers from data subset for this individual
    filtered <- df_sub %>%
      dplyr::filter(CEWL_g_m2h %nin% outs$outs)

    # add to running dataframe of cleaned data
    cleaned <- cleaned %>%
      rbind(filtered)
  }
  return(cleaned)
}
```

Apply function to data and check that the new data subsets still contain the right amount of data:

```
outliers_omitted <- omit_outliers(all_CEWL_data_edited2)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)

## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)

## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)

## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)

## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
```

```r
nrow(all_CEWL_data_edited2) == nrow(outliers_omitted) + nrow(outliers_found)
```

```
## [1] TRUE
```

Re-Assess Variation

```r
new_CVs <- outliers_omitted %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100,
            min = min(CEWL_g_m2h),
```

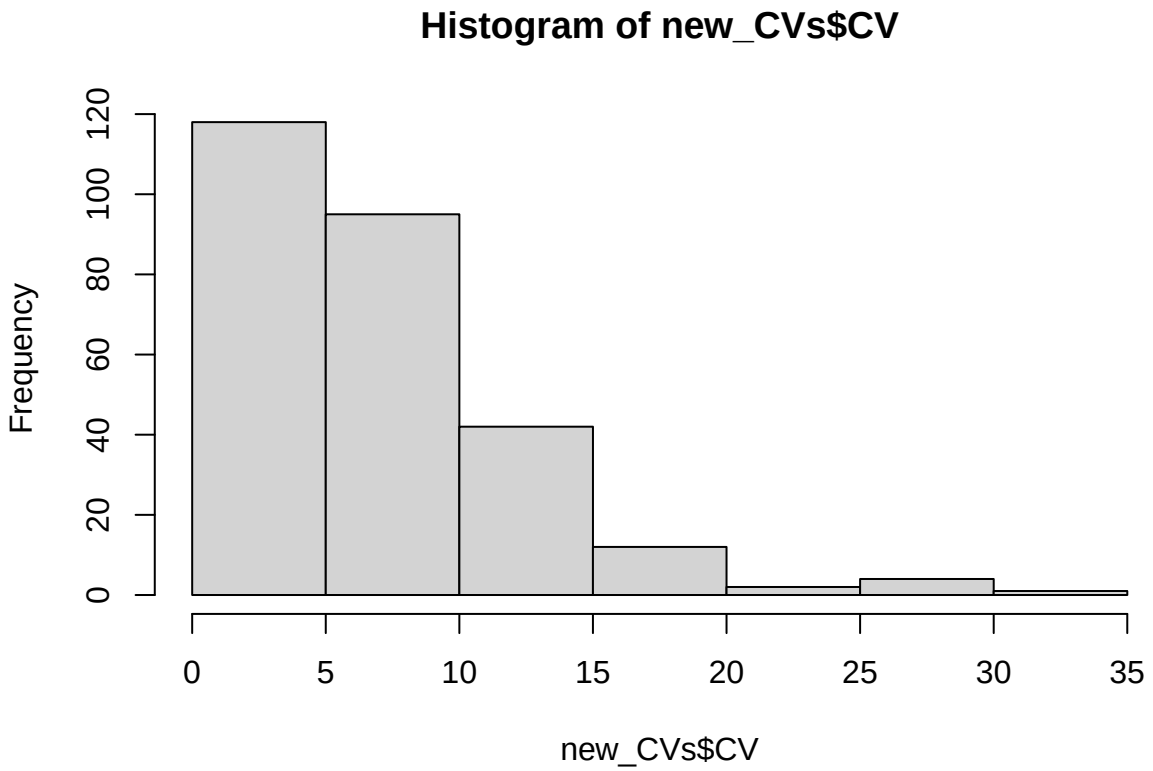
```
            max = max(CEWL_g_m2h),
            range = max - min)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

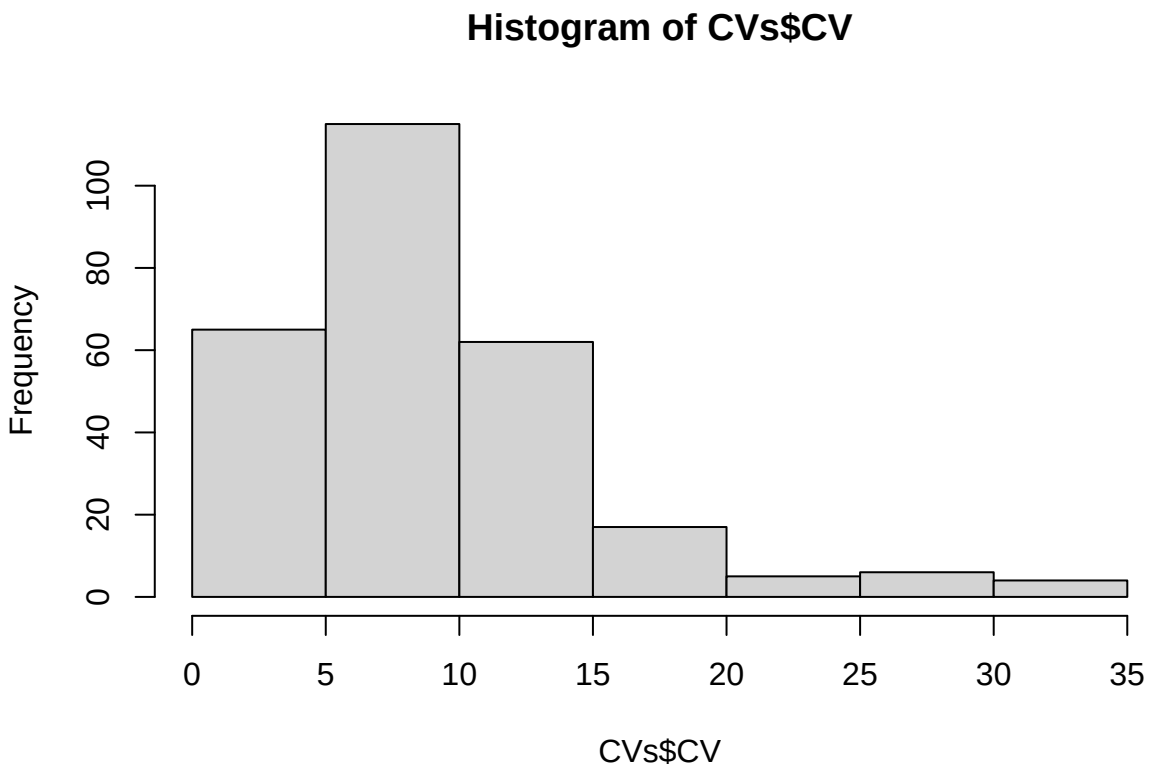
```
summary(new_CVs)
```

```
##  individual_ID      date                mean              SD
##  201    :  2   Min.   :2021-06-16   Min.   : 7.152   Min.   : 0.02517
##  202    :  2   1st Qu.:2021-06-26   1st Qu.:19.741   1st Qu.: 0.77272
##  203    :  2   Median :2021-07-20   Median :24.091   Median : 1.34563
##  204    :  2   Mean   :2021-07-20   Mean   :24.899   Mean   : 1.66977
##  205    :  2   3rd Qu.:2021-08-08   3rd Qu.:28.462   3rd Qu.: 2.22499
##  206    :  2   Max.   :2021-08-30   Max.   :79.267   Max.   :11.10858
##  (Other):262
##        CV                min              max             range
##  Min.   : 0.08437   Min.   : 5.68   Min.   : 8.74   Min.   : 0.050
##  1st Qu.: 3.06097   1st Qu.:18.07   1st Qu.:20.91   1st Qu.: 1.720
##  Median : 5.70605   Median :22.49   Median :25.88   Median : 3.220
##  Mean   : 6.91705   Mean   :23.00   Mean   :26.96   Mean   : 3.960
##  3rd Qu.: 9.51344   3rd Qu.:26.29   3rd Qu.:30.44   3rd Qu.: 5.285
##  Max.   :31.06794   Max.   :77.56   Max.   :81.42   Max.   :26.340
##
```

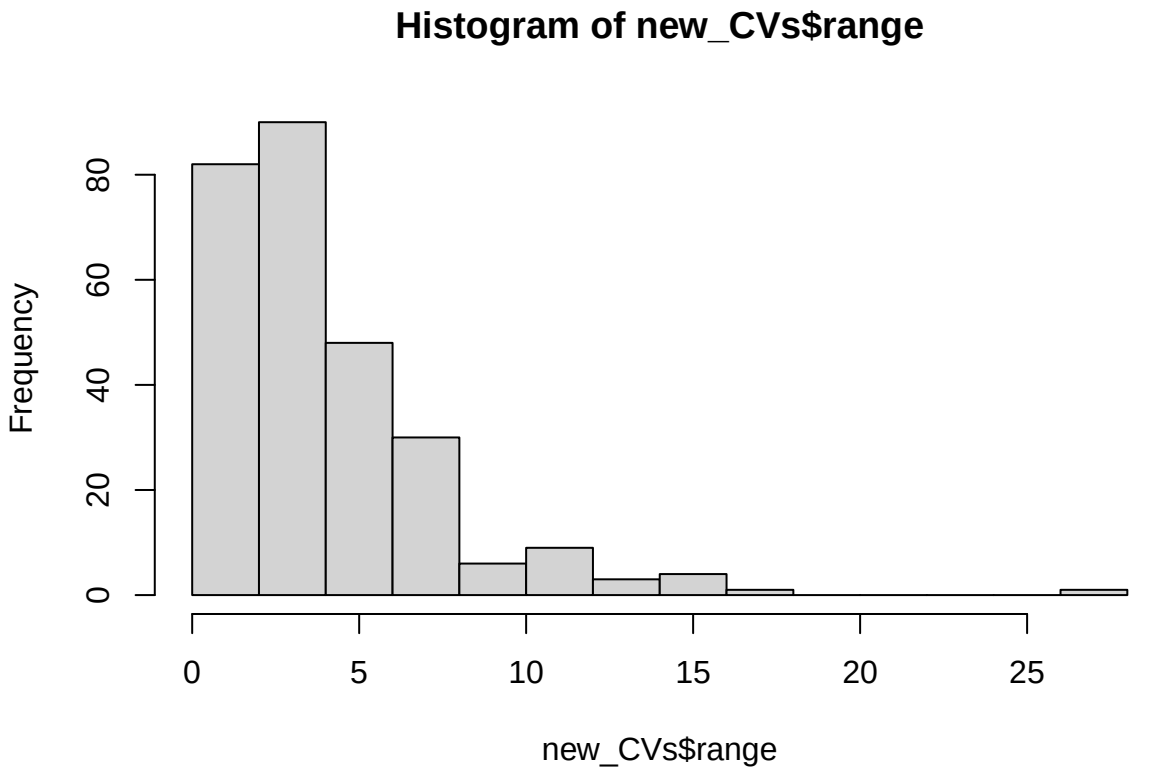
```
hist(new_CVs$CV)
```



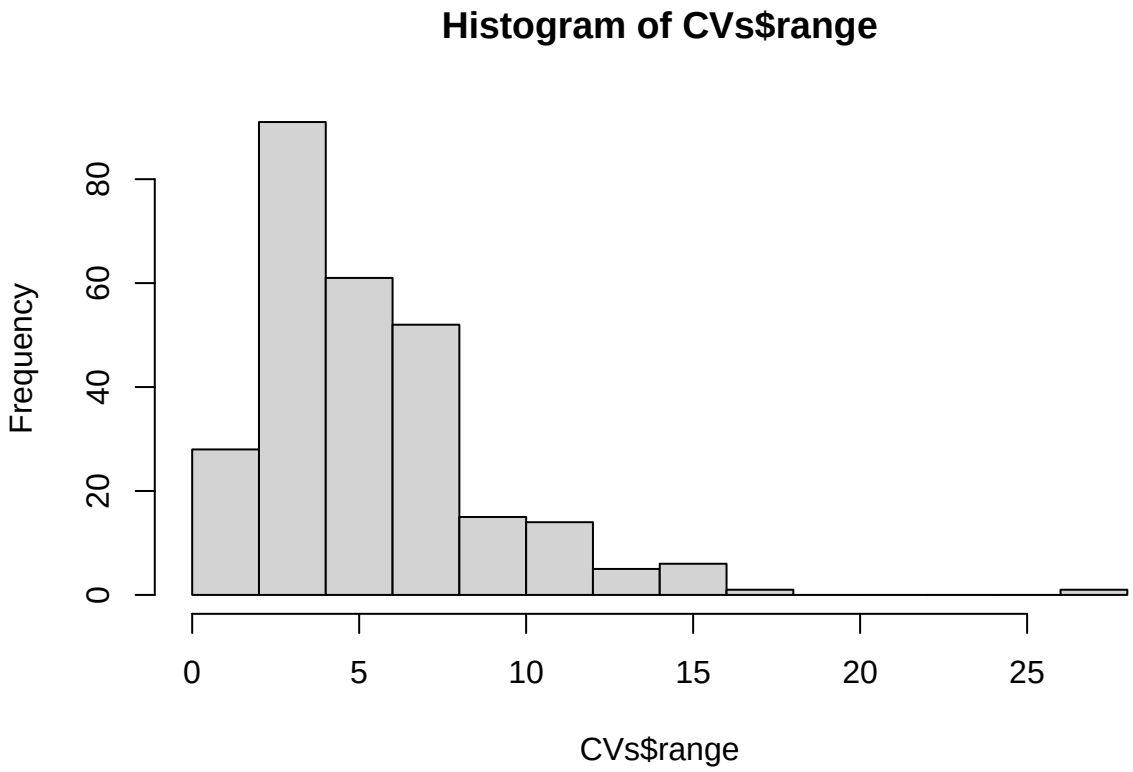

```
hist(CVs$CV)
```




```
hist(new_CVs$range)
```




```
hist(CVs$range)
```



Unfortunately, CVs are still skewed to the right, but overall, CVs are much lower and are mostly < 5-10%. We will continue with this dataset.

Average Replicates (outliers removed) & Join Cloacal Temp Data

```
CEWL_final <- outliers_omitted %>%
  group_by(date, individual_ID) %>%
  summarise(CEWL_g_m2h = mean(CEWL_g_m2h)) %>%
  left_join(cloacal_temp_C, by = c('date', 'individual_ID')) %>%
  dplyr::filter(complete.cases(CEWL_g_m2h, cloacal_temp_C))
```

```
## `summarise()` regrouping output by 'date' (override with `.groups` argument)
```

```
head(CEWL_final)
```

```
## # A tibble: 6 x 7
## # Groups:   date [1]
##   date       individual_ID CEWL_g_m2h time_c_temp         day     cloacal_temp_C
##   <date>     <fct>               <dbl> <dttm>              <fct>            <dbl>
## 1 2021-06-16 201                 12.4  2021-11-03 09:54:00 capture             26
## 2 2021-06-16 202                 15.8  2021-11-03 10:02:00 capture             29
## 3 2021-06-16 203                 12.0  2021-11-03 10:09:00 capture             28
## 4 2021-06-16 204                  9.68 2021-11-03 10:20:00 capture             29
## 5 2021-06-16 205                 10.3  2021-11-03 10:28:00 capture             27
## 6 2021-06-16 206                 11.1  2021-11-03 10:36:00 capture             27
## # ... with 1 more variable: date_time <dttm>
```

Final Synthesis

Re-Check Data

Check that we still have data for every individual, except for 254 and 304. 254 did not have his cloacal temperature taken before escaping, thus could not be included in any capture day models. 304 was omitted completely because he was accidentally recaptured and we only want his data from the first time he was included in the experiment.

I can check this by comparing a list of the individual IDs used (201-341) to the individual IDs in our final dataset, then selecting/printing the IDs used that are not in the final dataset.

```
c(seq(201, 341, 1))[c(seq(201, 341, 1)) %nin% unique(CEWL_final$individual_ID)]
```

```
## [1] 254 304
```

We expected individuals 254 and 304 not to be in the final dataset, so all is as expected.

Check how many observations were used to calculate mean CEWL for each individual on each date:

```
outliers_omitted %>%
  group_by(individual_ID, date) %>%
  summarise(n = n()) %>%
  arrange(n)
```

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
## # A tibble: 274 x 3
## # Groups:   individual_ID [140]
##    individual_ID date           n
##    <fct>         <date>     <int>
##  1 202           2021-06-16     3
##  2 207           2021-06-24     3
##  3 209           2021-06-24     3
##  4 210           2021-06-16     3
##  5 213           2021-06-24     3
##  6 218           2021-06-16     3
##  7 220           2021-06-16     3
##  8 223           2021-06-16     3
##  9 225           2021-06-16     3
## 10 227           2021-06-26     3
## # ... with 264 more rows
```

Between 3-6, awesome! That means we omitted 2 or less replicates for each individual on each measurement date.

Export

Save the cleaned data for models and figures.

```
write.csv(CEWL_final, "./data/CEWL_dat_all_clean.csv")
```

Reporting

We omitted a total of 136 measurements from our CEWL dataset. We realized post-experiment that individual 304 was a recapture, and had already undergone experimental conditions once before, so his data was completely excluded. One measurement attributed to individual 233 on June 26 was made 1.5 hours later than his other CEWL measurements and cloacal temperature on that day, which would represent a major deviation from our protocol. This measurement could not be confidently, truly attributed to individual

233, thus was omitted. We used the boxplot.stats function in R to extract outliers from each set of technical replicates, totaling 134 points qualifying as outliers which were thus removed.

After data cleaning, every individual still had at least 3 technical replicates for each of their measurement dates, with most individuals retaining all 5 original replicates. The distribution of coefficient of variation values was more-heavily distributed between 0-10% after data cleaning than before.