Climate Water Loss Experiment - CEWL Data Wrangling

Savannah Weaver

Contents

Packages

Background and Goals

This CEWL (cutaneous evaporative water loss) data was collected ...

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")
```

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),
                  na.strings=c("","NA"),
                # each csv has headers
                header = TRUE
```

```
                ) %>%
    # 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),
                  ID_len = as.factor(nchar(ID_rep_no)),

                  individual_ID = as.factor(case_when(
                    ID_len == 7 ~ as.character(paste(substr(ID_rep_no, 1, 1),
                                                     substr(ID_rep_no, 3, 5),
                                                     sep = "")),
                    ID_len == 6 & substr(ID_rep_no, 1, 1) == "W"
                        ~ as.character(substr(ID_rep_no, 1, 4)),
                    ID_len == 6 & substr(ID_rep_no, 1, 1) %in% c("M", "F")
                        ~ as.character(paste(substr(ID_rep_no, 1, 1),
                                                     substr(ID_rep_no, 3, 4),
                                                     sep = "")),
                    ID_len == 5 ~ as.character(substr(ID_rep_no, 1, 3))
                    )),

                  # works
                  replicate_no = as.factor(case_when(
                    ID_len == 7 ~ as.character(substr(ID_rep_no, 7, 7)),
                    ID_len == 6 ~ as.character(substr(ID_rep_no, 6, 6)),
                    ID_len == 5 ~ as.character(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. Additionally, 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) %>%
  # filter out failed measurements
  dplyr::filter(status == "Normal") %>%
  # correctly format data classes
  mutate(date = as.Date(date, format = "%m/%d/%y"),
         time = as.POSIXct(time, format = "%H:%M"),
         status = as.factor(status)
         )
```

```
summary(all_CEWL_data)
```

```
##       date                 time                          status
##  Min.   :2021-04-23   Min.   :2022-06-25 01:00:00   Normal:456
##  1st Qu.:2021-04-24   1st Qu.:2022-06-25 02:24:45
##  Median :2021-05-07   Median :2022-06-25 03:46:00
##  Mean   :2021-05-12   Mean   :2022-06-25 04:22:43
##  3rd Qu.:2021-05-08   3rd Qu.:2022-06-25 05:02:15
##  Max.   :2021-07-14   Max.   :2022-06-25 12:59:00
##
##   ID_rep_no           CEWL_g_m2h     msmt_temp_C    msmt_RH_percent ID_len
##  Length:456         Min.   :-1.32   Min.   :18.90   Min.   :11.50   5:122
##  Class :character   1st Qu.: 7.74   1st Qu.:28.50   1st Qu.:14.50   6:244
##  Mode  :character   Median :10.21   Median :30.30   Median :16.95   7: 90
##                     Mean   :10.62   Mean   :29.55   Mean   :21.36
##                     3rd Qu.:12.89   3rd Qu.:31.50   3rd Qu.:24.30
##                     Max.   :65.31   Max.   :33.70   Max.   :41.60
##
##  individual_ID replicate_no
##  F12    : 13   1:117
##  M10    : 13   2:118
##  M11    : 13   3:118
##  M19    : 13   4: 52
##  M20    : 13   5: 51
##  M09    : 12
##  (Other):379
```

Check Data

Each lizard measured on each date should have 3-5 technical replicates, and those measurements should have been taken around the same time.

```
all_CEWL_data %>%
                group_by(individual_ID, date) %>%
                summarise(n = n(),
                          time_range = max(time) - min(time)) %>%
                arrange(n)
```

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

```
## # A tibble: 118 x 4
## # Groups:   individual_ID [80]
##    individual_ID date           n time_range
##    <fct>         <date>     <int> <drtn>
##  1 F01           2021-04-23     3 120 secs
##  2 F02           2021-04-23     3 120 secs
##  3 F03           2021-04-23     3 120 secs
##  4 F04           2021-04-23     3  60 secs
##  5 F05           2021-04-24     3 120 secs
##  6 F06           2021-04-24     3 120 secs
##  7 F07           2021-04-24     3  60 secs
##  8 F08           2021-04-24     3  60 secs
##  9 F09           2021-04-24     3 120 secs
## 10 F10           2021-04-24     3 120 secs
```

```
## # ... with 108 more rows
```

The number of measurements taken is good! Almost always 3 or 5, with two lizards who only got 4 measurements, which is fine. But, M01 on April 23 and M03a on July 14 have abnormal time ranges of 43140 seconds (almost 12h), so we need to check that data.

```
all_CEWL_data %>% dplyr::filter(individual_ID %in% c("M01", "M03A"))
```

```
##          date                time status ID_rep_no CEWL_g_m2h msmt_temp_C
## 1  2021-04-23 2022-06-25 12:57:00 Normal     M01_1       0.69        31.0
## 2  2021-04-23 2022-06-25 12:59:00 Normal     M01_2       0.14        30.7
## 3  2021-04-23 2022-06-25 01:00:00 Normal     M01_3       1.12        30.5
## 4  2021-07-14 2022-06-25 12:58:00 Normal   M-03A-1       9.98        27.4
## 5  2021-07-14 2022-06-25 12:59:00 Normal   M-03A-2       9.16        27.8
## 6  2021-07-14 2022-06-25 01:00:00 Normal   M-03A-3      11.05        28.0
## 7  2021-07-14 2022-06-25 01:01:00 Normal   M-03A-4      13.29        28.1
## 8  2021-07-14 2022-06-25 01:02:00 Normal   M-03A-5       8.69        28.4
## 9  2021-07-14 2022-06-25 05:00:00 Normal    M-01-1      13.70        27.4
## 10 2021-07-14 2022-06-25 05:01:00 Normal    M-01-2      10.94        27.2
## 11 2021-07-14 2022-06-25 05:02:00 Normal    M-01-3      11.35        27.0
## 12 2021-07-14 2022-06-25 05:03:00 Normal    M-01-4       9.39        26.8
## 13 2021-07-14 2022-06-25 05:04:00 Normal    M-01-5       8.90        26.6
##    msmt_RH_percent ID_len individual_ID replicate_no
## 1             15.9      5           M01            1
## 2             16.3      5           M01            2
## 3             16.7      5           M01            3
## 4             37.1      7          M03A            1
## 5             36.8      7          M03A            2
## 6             37.1      7          M03A            3
## 7             35.9      7          M03A            4
## 8             35.2      7          M03A            5
## 9             39.7      6           M01            1
## 10            39.6      6           M01            2
## 11            39.5      6           M01            3
## 12            39.6      6           M01            4
## 13            39.6      6           M01            5
```

Aha, it seems the problem is that the time isn't perfectly formatted, so 1 pm is coded as 1 am –> the measurements in question went across hours of 12 noon to 1 pm, so when reformatted, it seems like 1 am to 12 pm. It's fine as-is, and nothing is amiss with the data.

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 %>%
  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
```

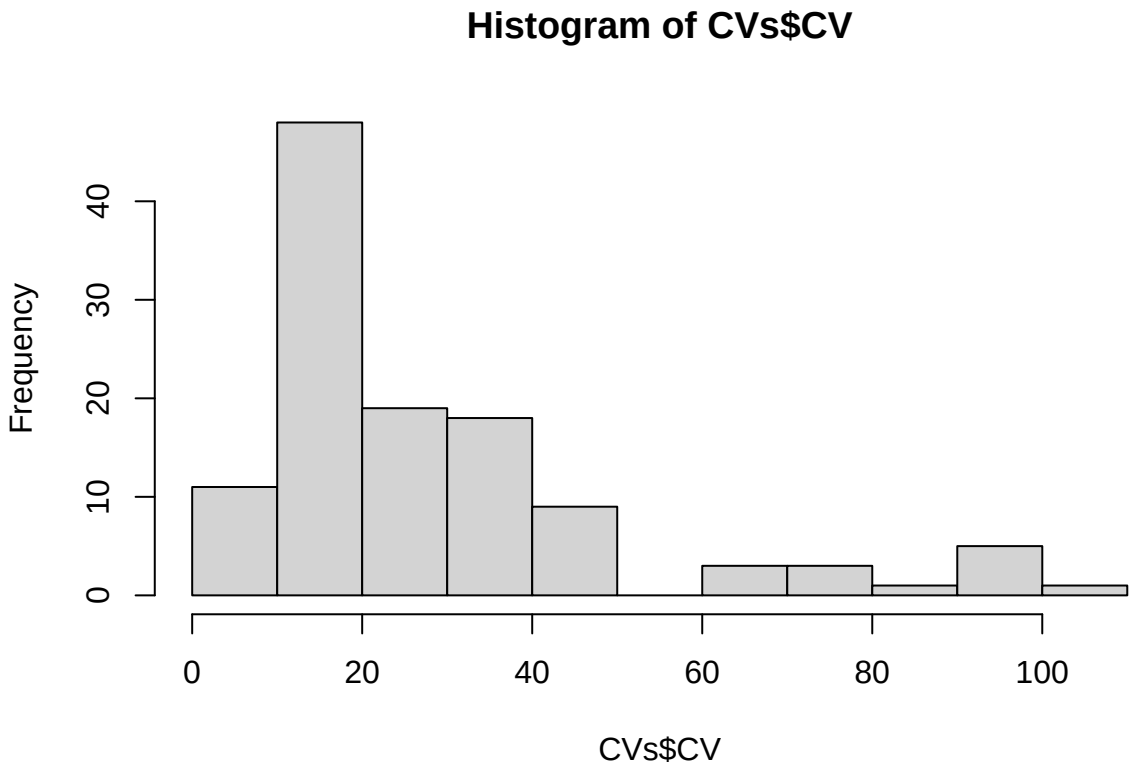
```
            )
```

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

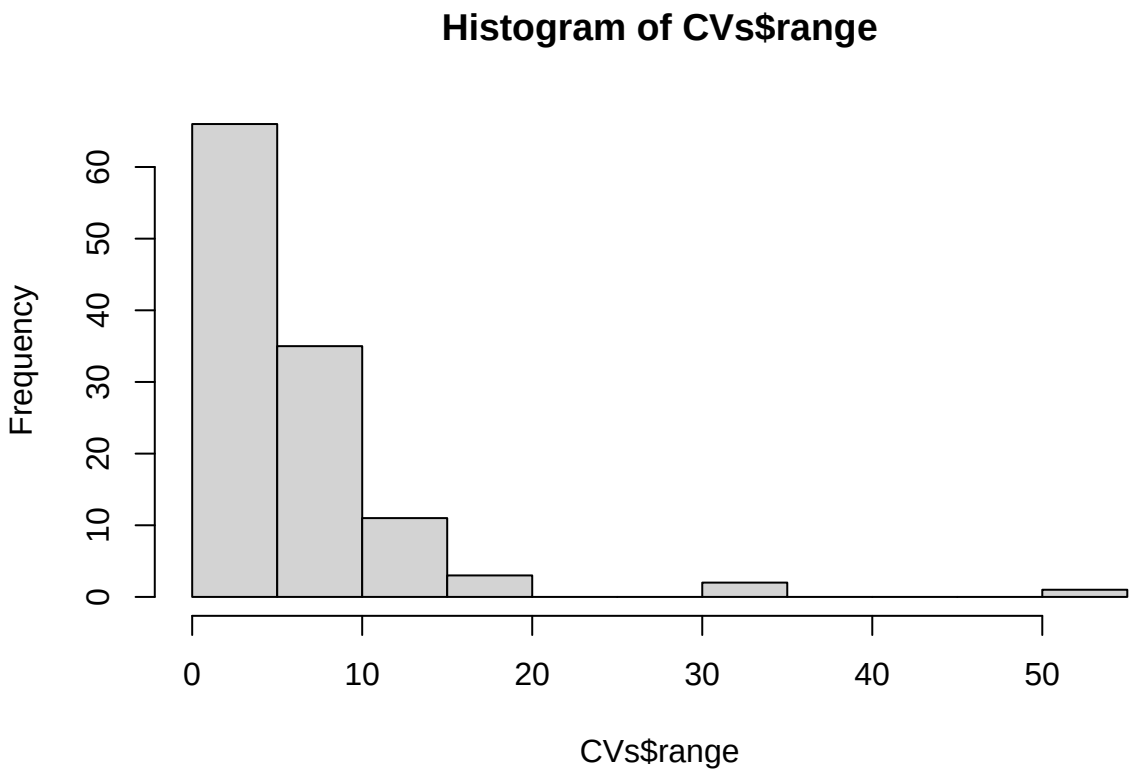
```
summary(CVs)
```

```
##  individual_ID      date                mean              SD
##  F12    :  3   Min.   :2021-04-23   Min.   : 0.650   Min.   : 0.1124
##  M09    :  3   1st Qu.:2021-04-24   1st Qu.: 8.486   1st Qu.: 1.4849
##  M10    :  3   Median :2021-04-24   Median :10.443   Median : 2.0290
##  M11    :  3   Mean   :2021-05-08   Mean   :10.823   Mean   : 2.9641
##  M19    :  3   3rd Qu.:2021-05-08   3rd Qu.:13.391   3rd Qu.: 3.1195
##  M20    :  3   Max.   :2021-07-14   Max.   :31.550   Max.   :29.3242
##  (Other):100
##        CV               min              max            range
##  Min.   :  1.956   Min.   :-1.320   Min.   : 1.12   Min.   : 0.220
##  1st Qu.: 15.021   1st Qu.: 6.723   1st Qu.:10.21   1st Qu.: 3.130
##  Median : 20.135   Median : 8.245   Median :13.32   Median : 4.600
##  Mean   : 28.713   Mean   : 8.159   Mean   :14.36   Mean   : 6.196
##  3rd Qu.: 35.639   3rd Qu.:10.500   3rd Qu.:16.37   3rd Qu.: 6.772
##  Max.   :105.713   Max.   :19.640   Max.   :65.31   Max.   :52.900
##
```

```
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. 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 == (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)
```

```
## `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', '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' (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' (override with `.groups` argument)
```

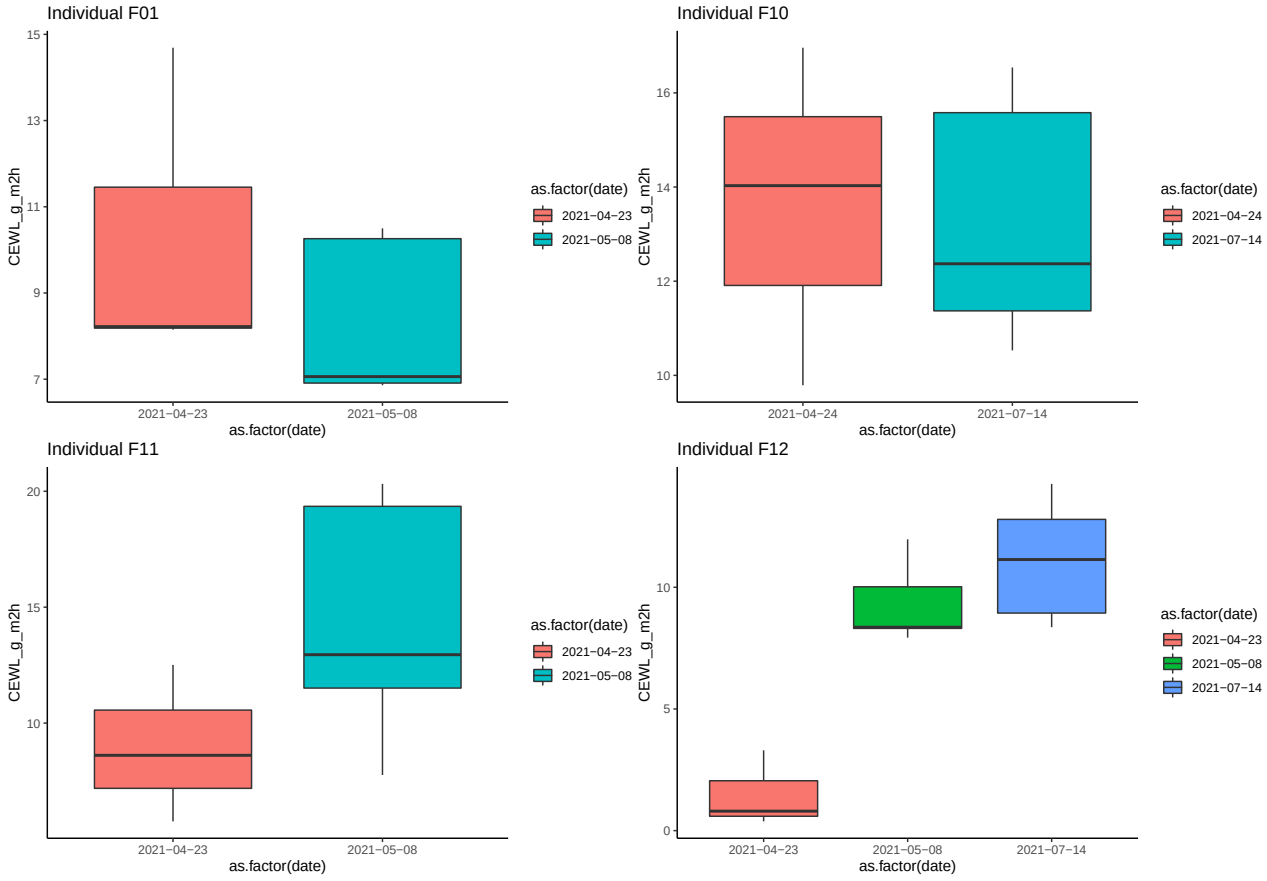
```
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()` regrouping output by 'individual_ID', 'date' (override with `.groups` argument)
## `summarise()`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

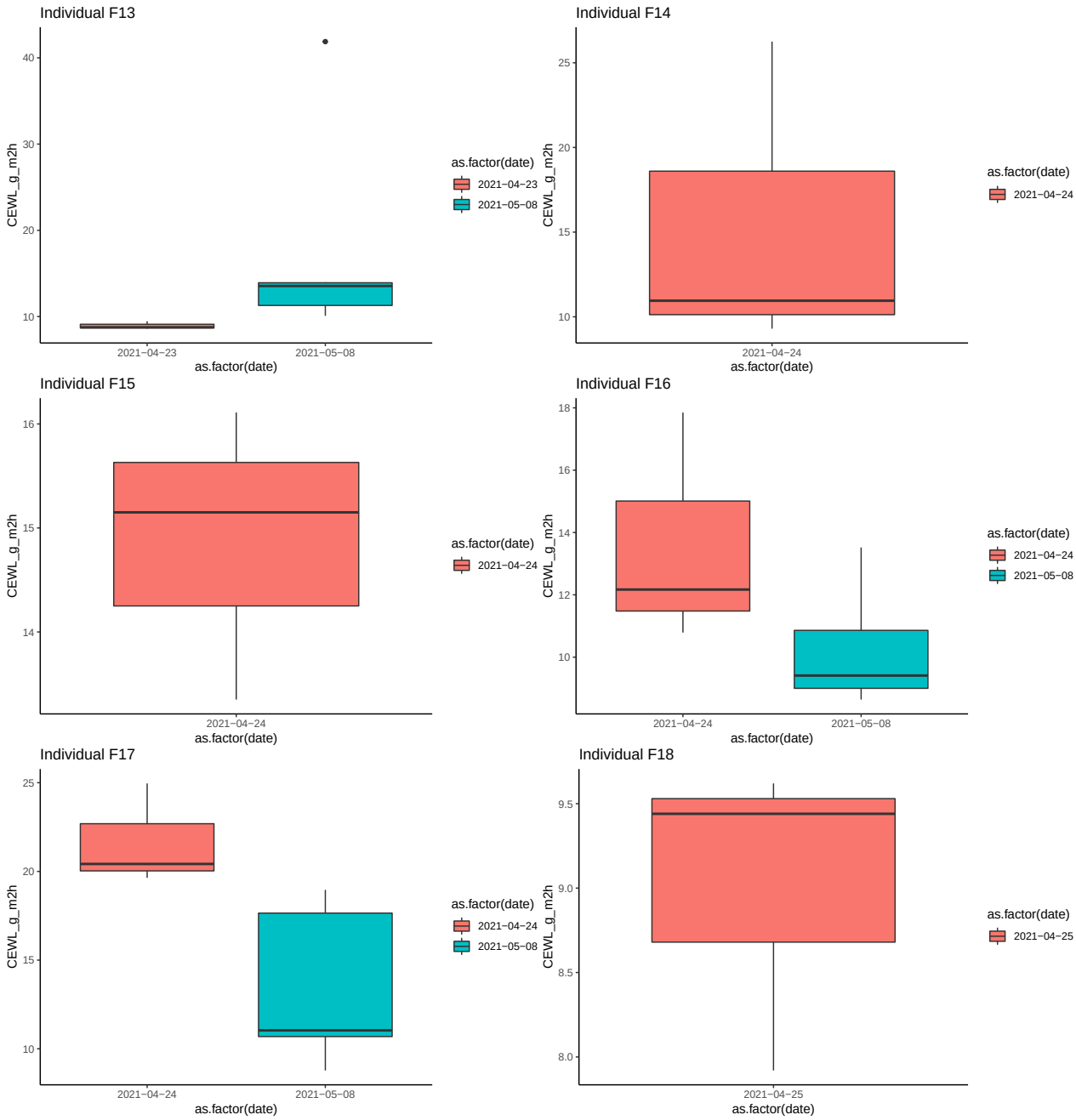
```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

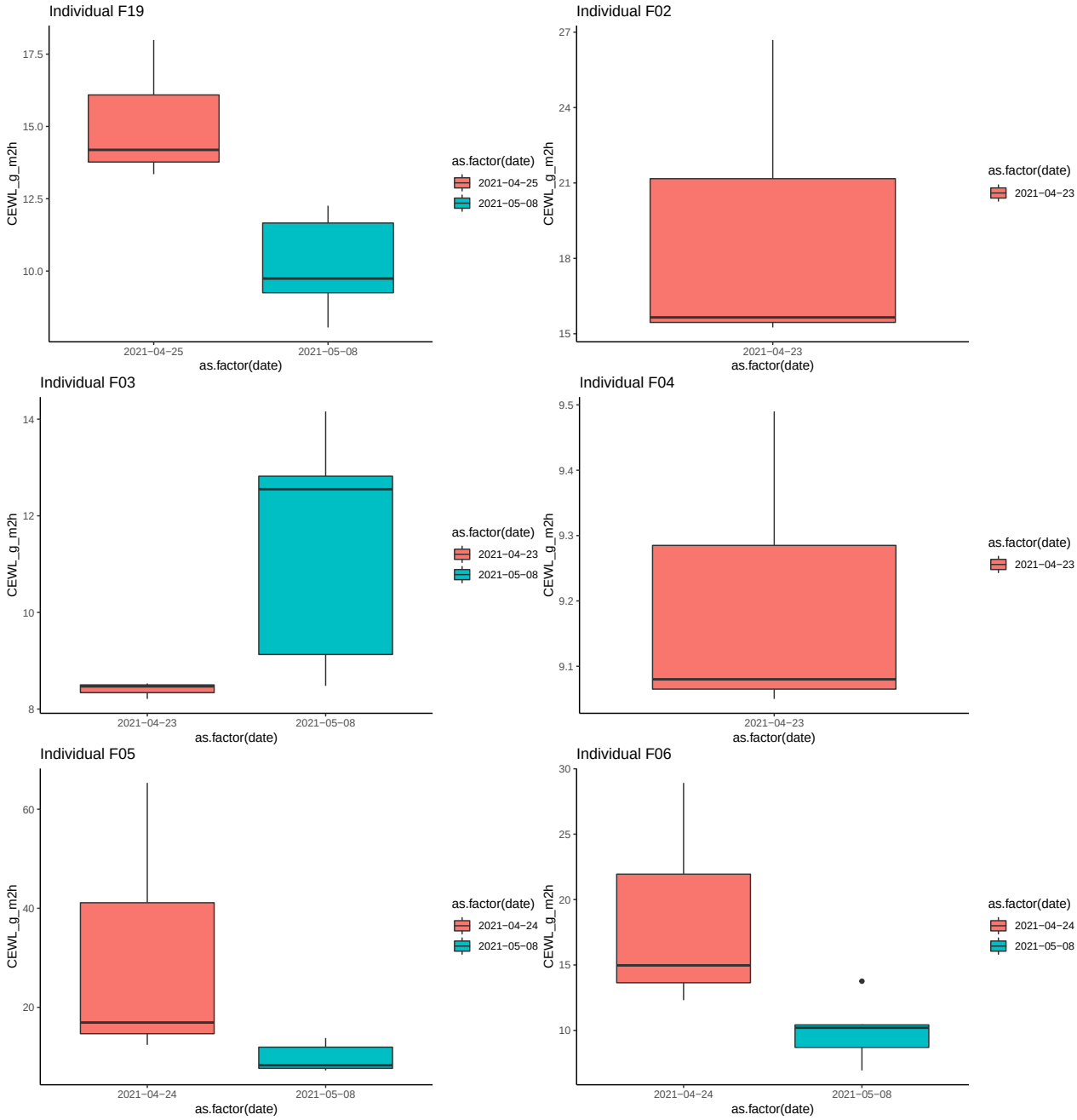
```
outliers_found
```

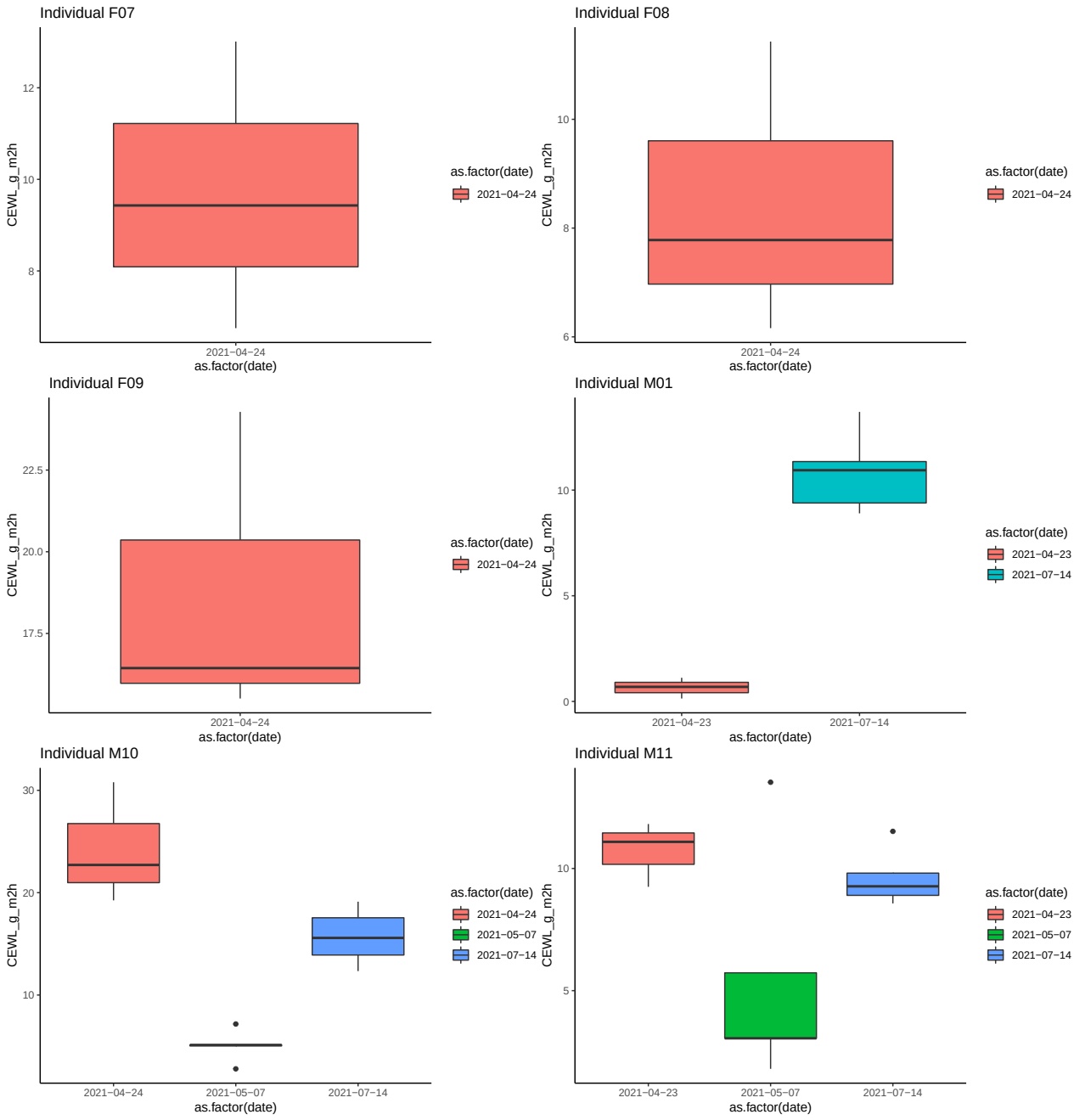
```
## # A tibble: 24 x 3
## # Groups:   individual_ID, date [18]
##    individual_ID date        outs
##    <fct>         <date>     <dbl>
##  1 F13           2021-05-08 41.9
##  2 F06           2021-05-08 13.8
##  3 M10           2021-05-07  7.17
##  4 M10           2021-05-07  2.79
##  5 M11           2021-05-07 13.5
##  6 M11           2021-07-14 11.5
##  7 M13           2021-07-14 17.1
##  8 M13           2021-07-14 11.3
##  9 M19           2021-07-14 17.9
## 10 M20           2021-07-14 11.3
## # ... with 14 more rows
```

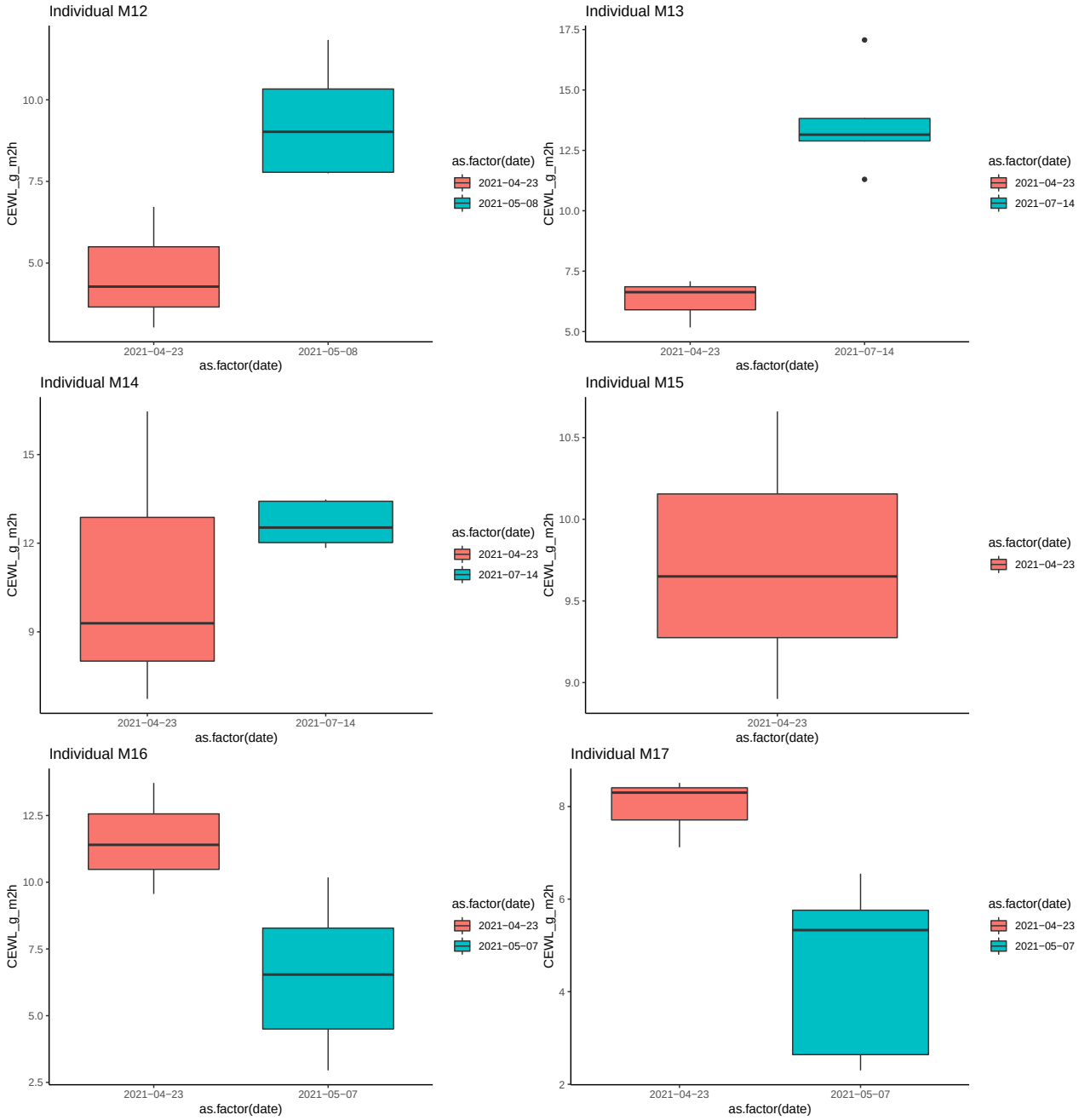
```
par(mfrow = c(1, 1))
```

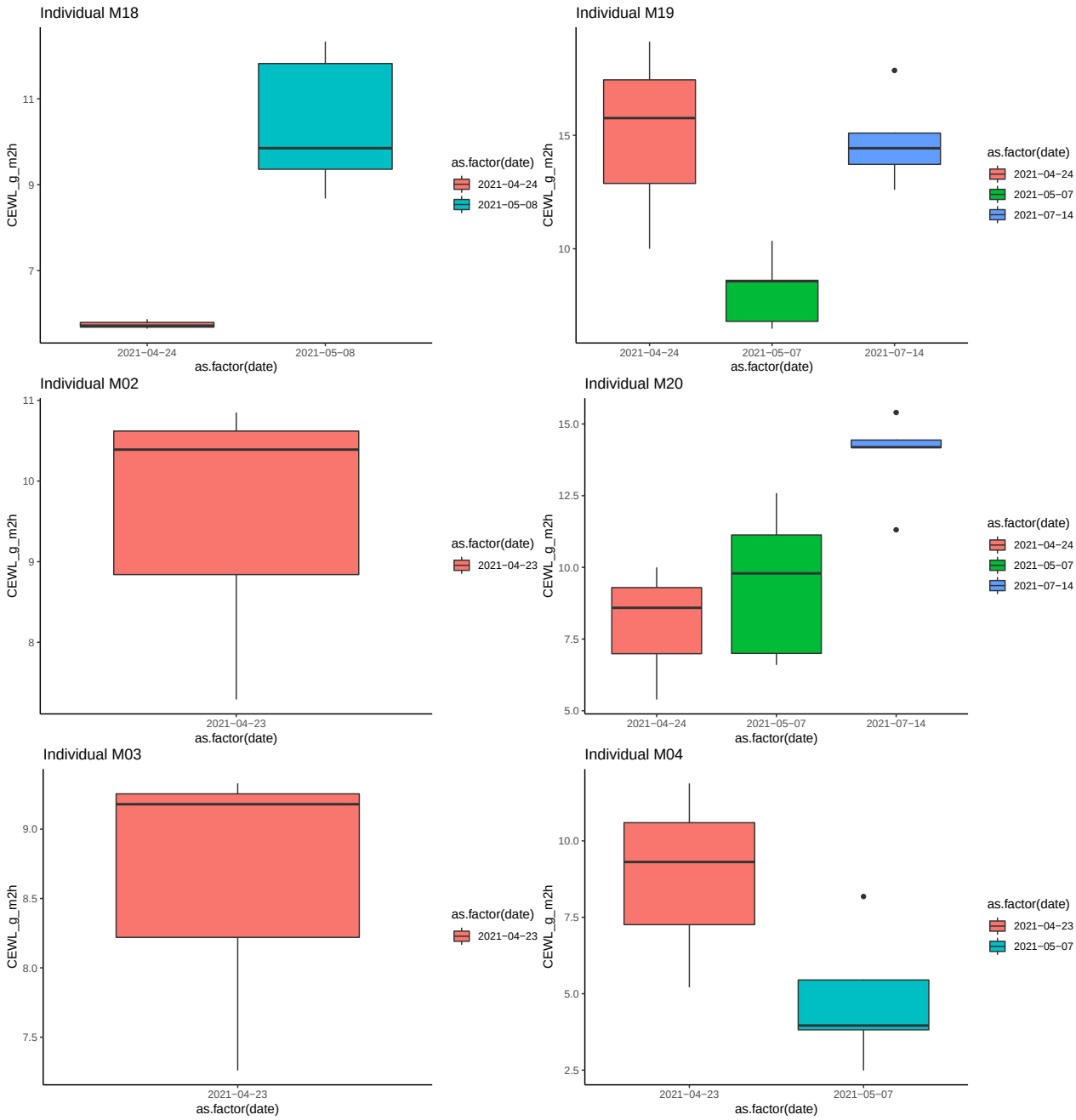


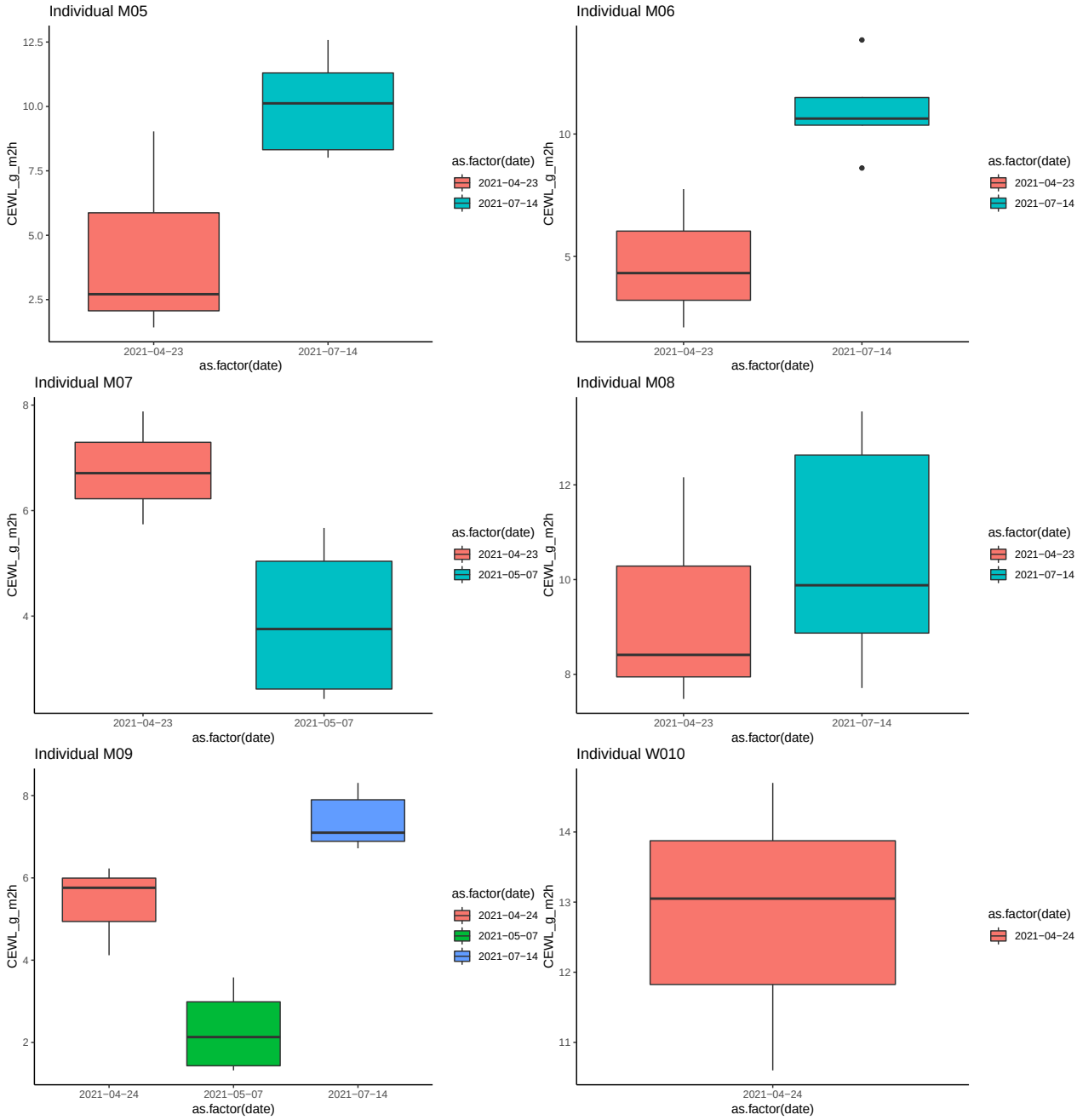

Individual F13
CEWL_g_m2h
40
30
20
10
2021−04−23
2021−05−08
as.factor(date)
as.factor(date)
2021−04−23
2021−05−08
Individual F14
CEWL_g_m2h
25
20
15
10
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual F15
CEWL_g_m2h
16
15
14
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual F16
CEWL_g_m2h
18
16
14
12
10
2021−04−24
2021−05−08
as.factor(date)
as.factor(date)
2021−04−24
2021−05−08
Individual F17
CEWL_g_m2h
25
20
15
10
2021−04−24
2021−05−08
as.factor(date)
as.factor(date)
2021−04−24
2021−05−08
Individual F18
CEWL_g_m2h
9.5
9.0
8.5
8.0
2021−04−25
as.factor(date)
as.factor(date)
2021−04−25

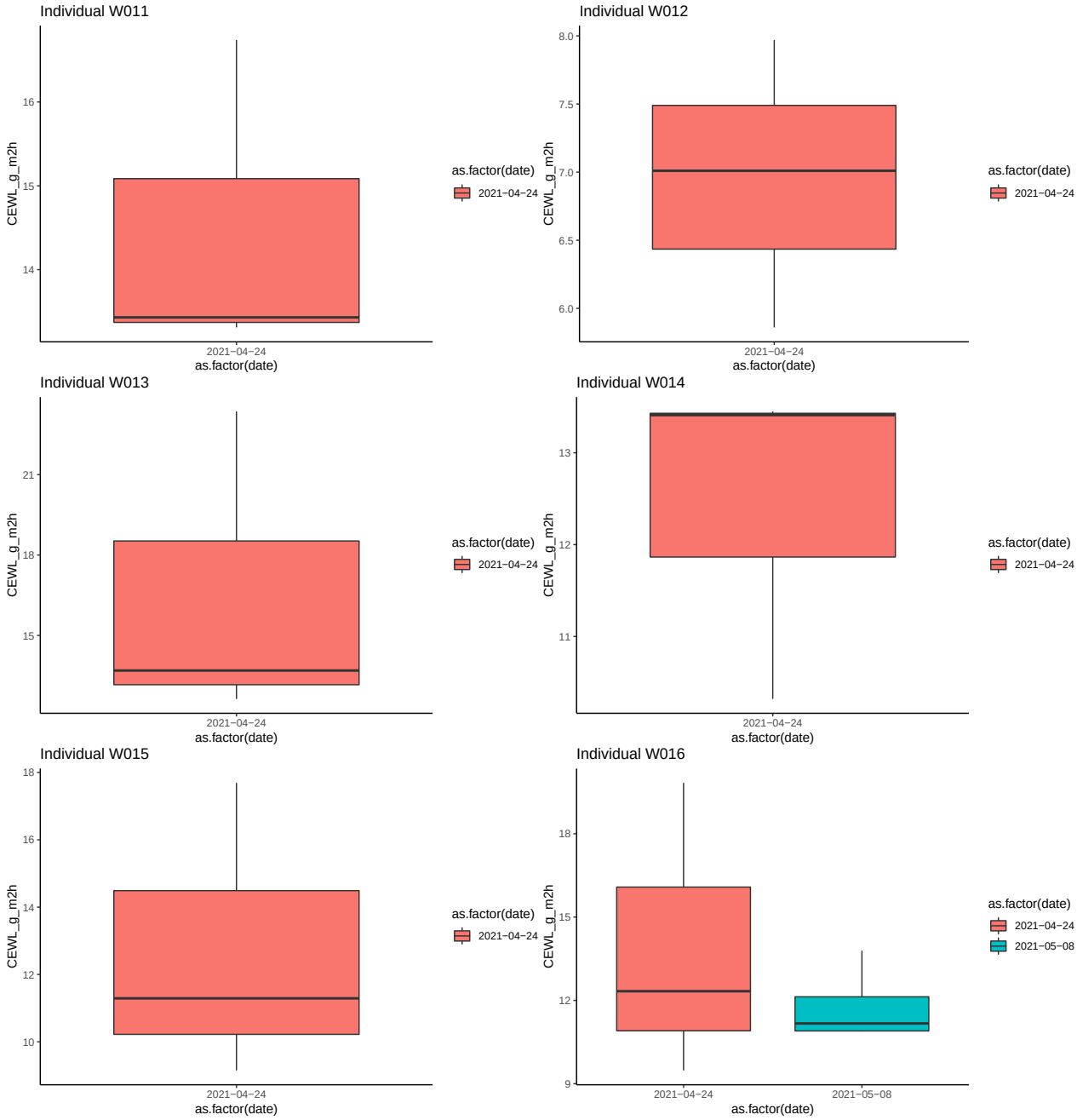

Individual F19
CEWL_g_m2h
17.5
15.0
12.5
10.0
2021−04−25
2021−05−08
as.factor(date)
as.factor(date)
2021−04−25
2021−05−08
Individual F02
CEWL_g_m2h
27
24
21
18
15
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual F03
CEWL_g_m2h
14
12
10
8
2021−04−23
2021−05−08
as.factor(date)
as.factor(date)
2021−04−23
2021−05−08
Individual F04
CEWL_g_m2h
9.5
9.4
9.3
9.2
9.1
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual F05
CEWL_g_m2h
60
40
20
2021−04−24
2021−05−08
as.factor(date)
as.factor(date)
2021−04−24
2021−05−08
Individual F06
CEWL_g_m2h
30
25
20
15
10
2021−04−24
2021−05−08
as.factor(date)
as.factor(date)
2021−04−24
2021−05−08

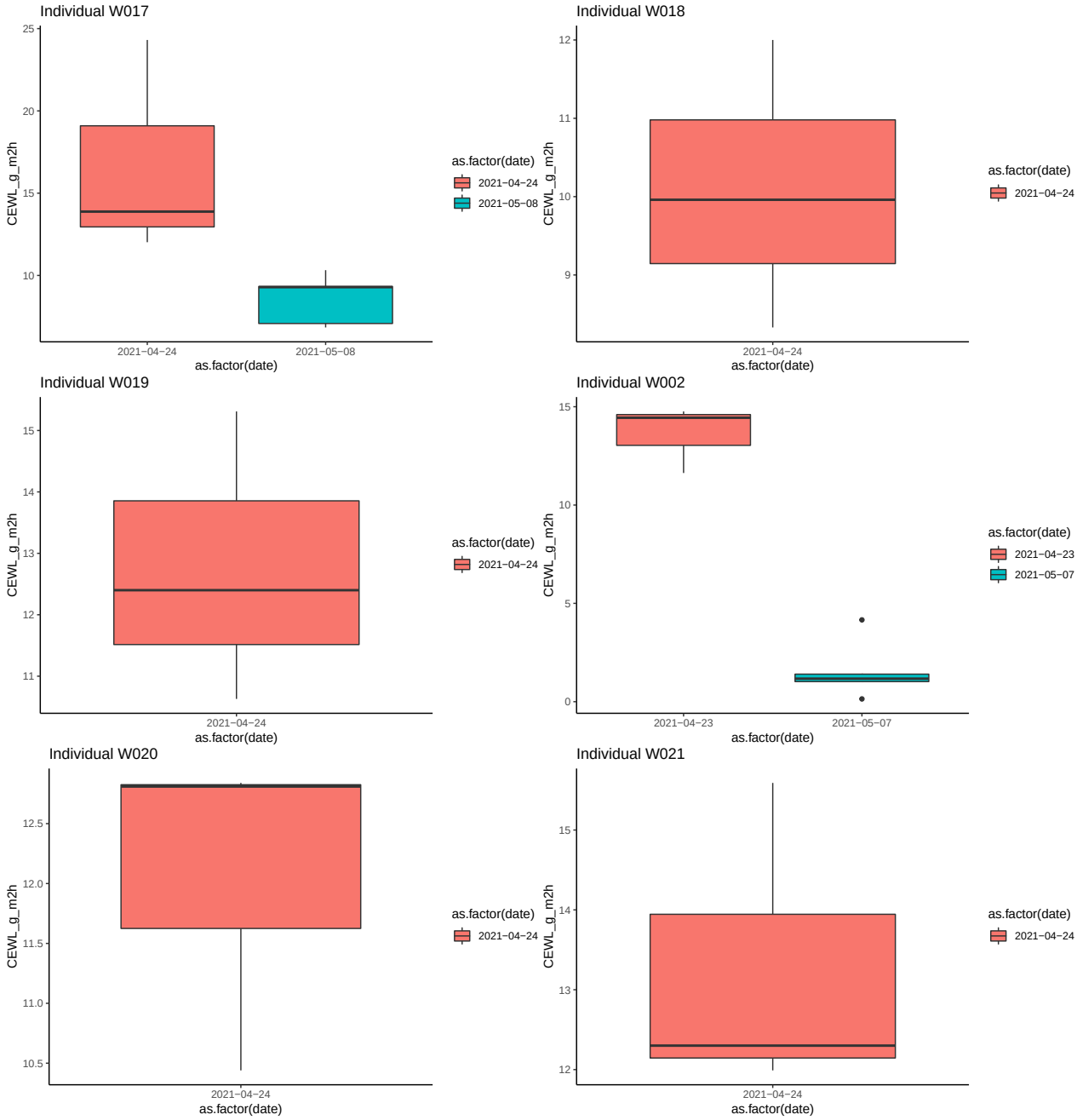

Individual F07
CEWL_g_m2h
12
10
8
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual F08
CEWL_g_m2h
10
8
6
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual F09
CEWL_g_m2h
22.5
20.0
17.5
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual M01
CEWL_g_m2h
10
5
0
2021−04−23
2021−07−14
as.factor(date)
as.factor(date)
2021−04−23
2021−07−14
Individual M10
CEWL_g_m2h
30
20
10
2021−04−24
2021−05−07
2021−07−14
as.factor(date)
as.factor(date)
2021−04−24
2021−05−07
2021−07−14
Individual M11
CEWL_g_m2h
10
5
2021−04−23
2021−05−07
2021−07−14
as.factor(date)
as.factor(date)
2021−04−23
2021−05−07
2021−07−14

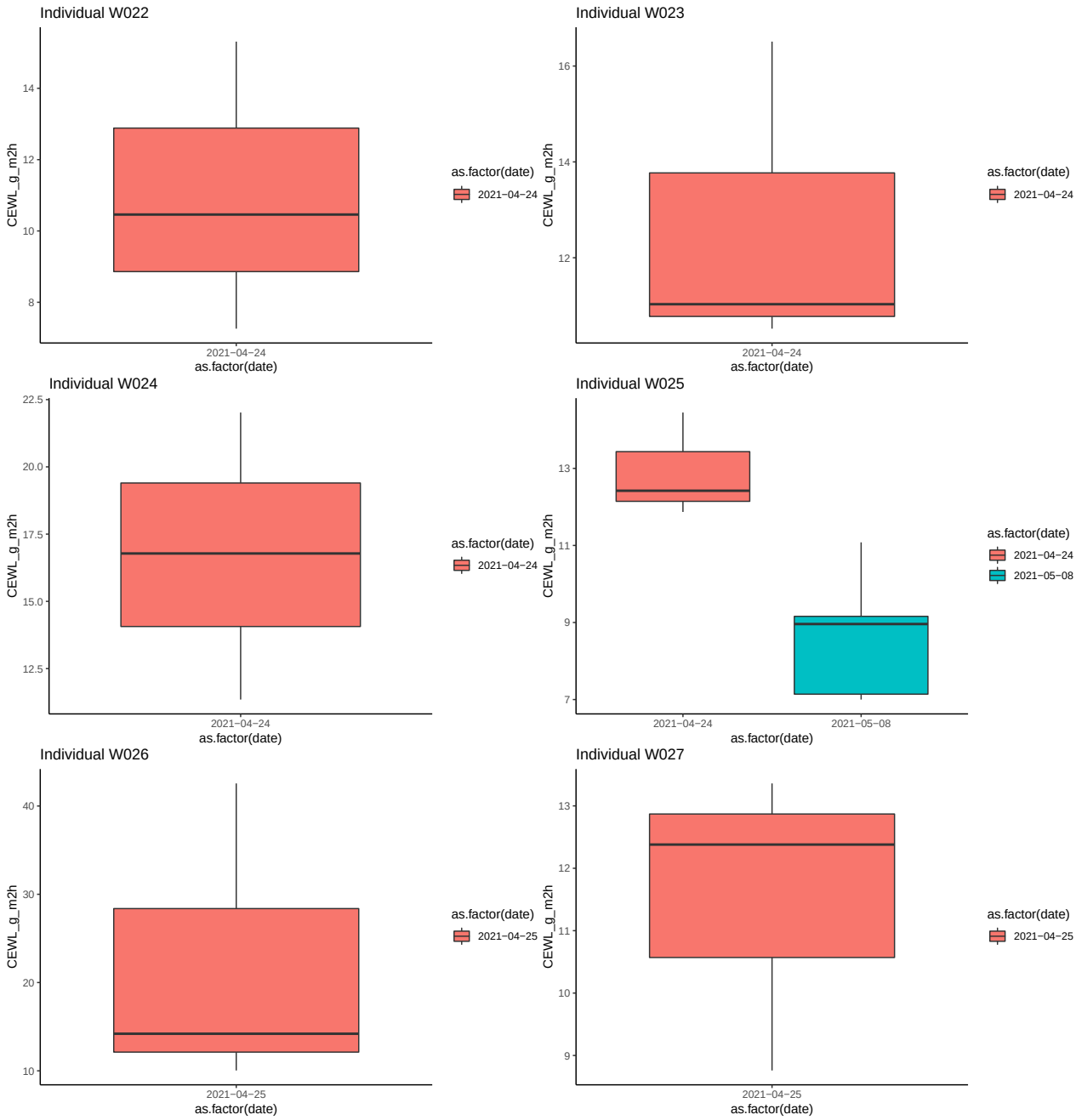

Individual M12
CEWL_g_m2h
10.0
7.5
5.0
2021−04−23
2021−05−08
as.factor(date)
as.factor(date)
2021−04−23
2021−05−08
Individual M13
CEWL_g_m2h
17.5
15.0
12.5
10.0
7.5
5.0
2021−04−23
2021−07−14
as.factor(date)
as.factor(date)
2021−04−23
2021−07−14
Individual M14
CEWL_g_m2h
15
12
9
2021−04−23
2021−07−14
as.factor(date)
as.factor(date)
2021−04−23
2021−07−14
Individual M15
CEWL_g_m2h
10.5
10.0
9.5
9.0
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual M16
CEWL_g_m2h
12.5
10.0
7.5
5.0
2.5
2021−04−23
2021−05−07
as.factor(date)
as.factor(date)
2021−04−23
2021−05−07
Individual M17
CEWL_g_m2h
8
6
4
2
2021−04−23
2021−05−07
as.factor(date)
as.factor(date)
2021−04−23
2021−05−07

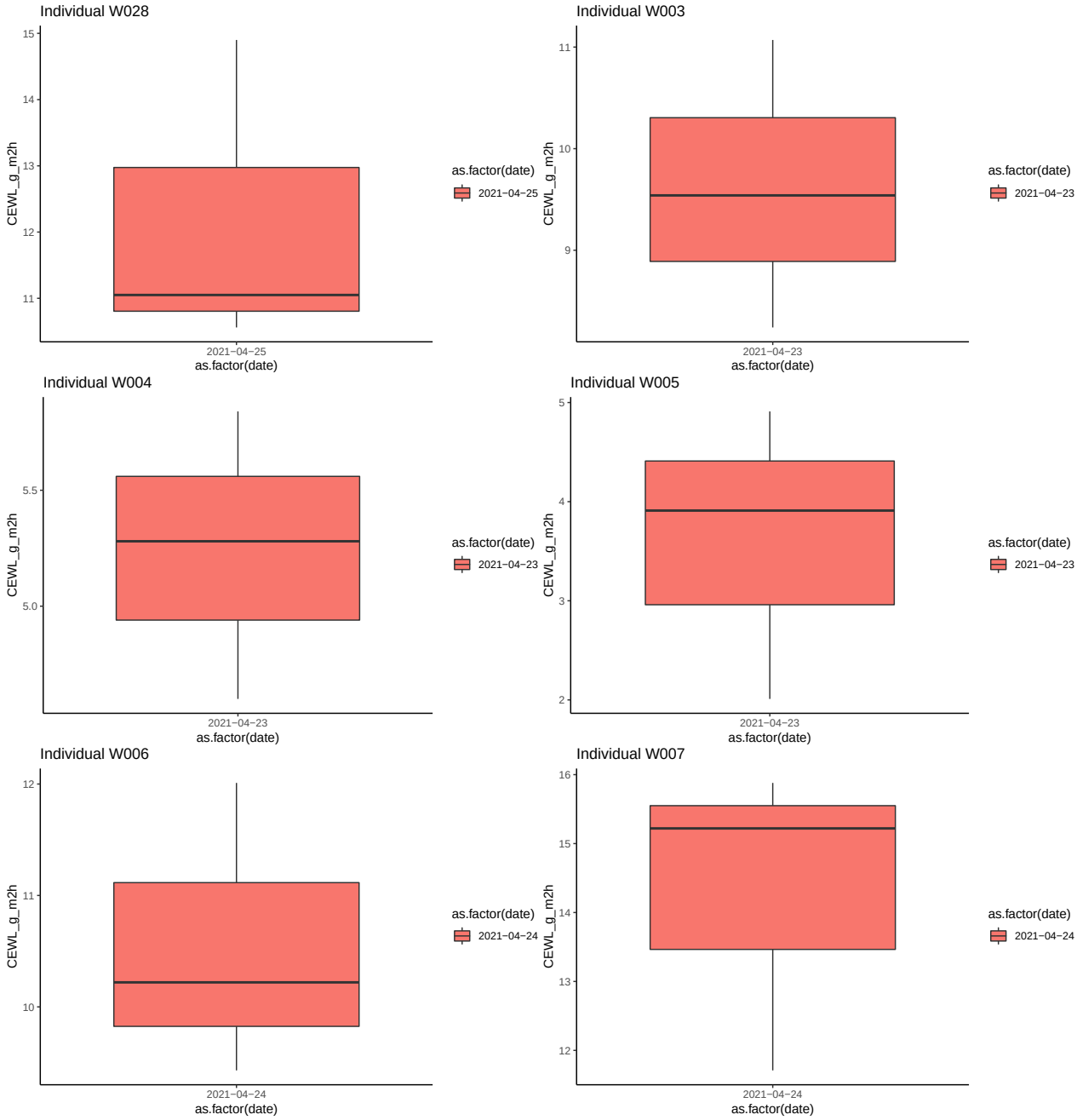

Individual M18
CEWL_g_m2h
11
9
7
2021−04−24
2021−05−08
as.factor(date)
as.factor(date)
2021−04−24
2021−05−08
Individual M19
CEWL_g_m2h
15
10
2021−04−24
2021−05−07
2021−07−14
as.factor(date)
as.factor(date)
2021−04−24
2021−05−07
2021−07−14
Individual M02
CEWL_g_m2h
11
10
9
8
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual M20
CEWL_g_m2h
15.0
12.5
10.0
7.5
5.0
2021−04−24
2021−05−07
2021−07−14
as.factor(date)
as.factor(date)
2021−04−24
2021−05−07
2021−07−14
Individual M03
CEWL_g_m2h
9.0
8.5
8.0
7.5
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual M04
CEWL_g_m2h
10.0
7.5
5.0
2.5
2021−04−23
2021−05−07
as.factor(date)
as.factor(date)
2021−04−23
2021−05−07

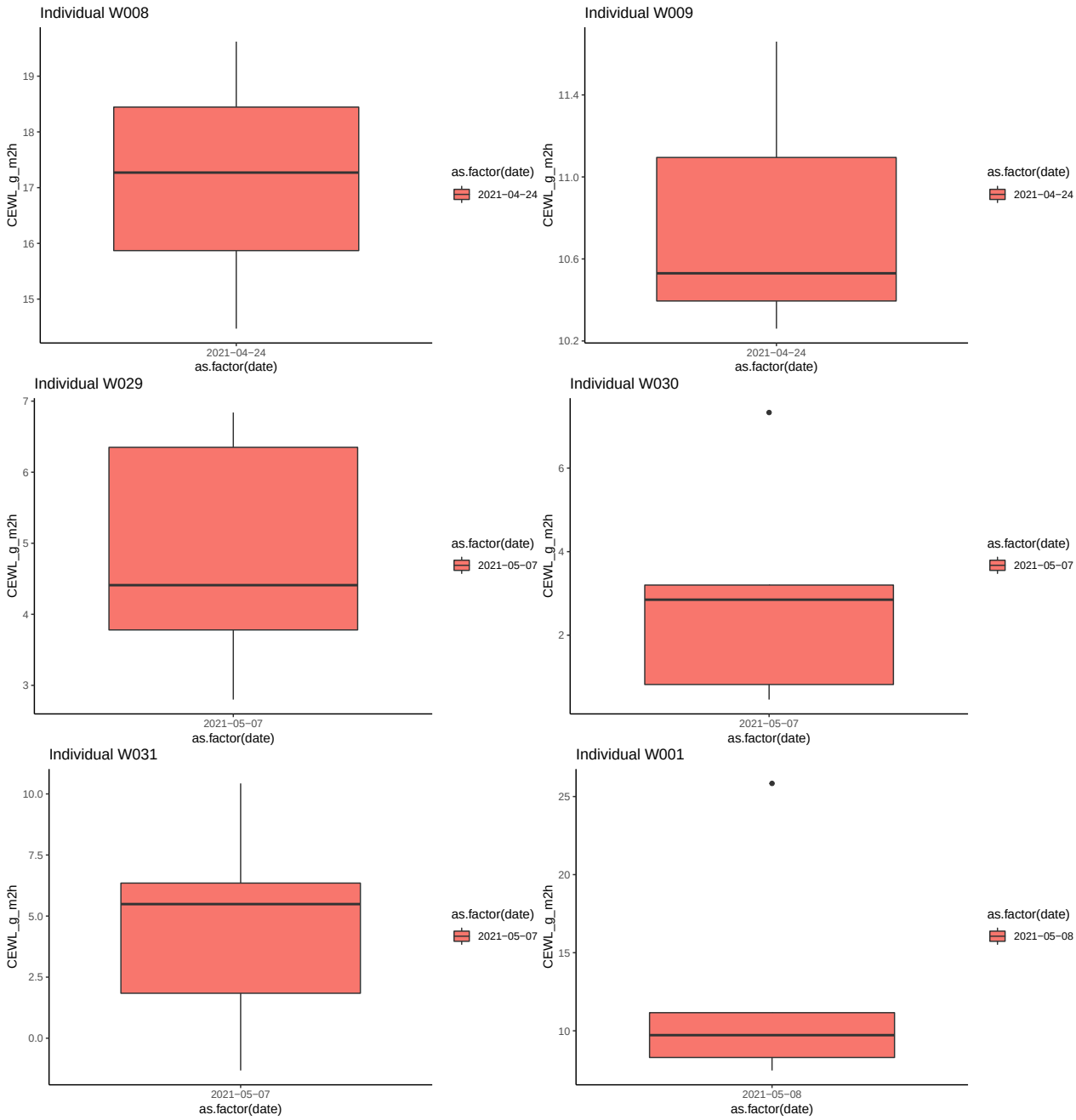

Individual M05
CEWL_g_m2h
12.5
10.0
7.5
5.0
2.5
2021−04−23
2021−07−14
as.factor(date)
as.factor(date)
2021−04−23
2021−07−14
Individual M06
CEWL_g_m2h
10
5
2021−04−23
2021−07−14
as.factor(date)
as.factor(date)
2021−04−23
2021−07−14
Individual M07
CEWL_g_m2h
8
6
4
2021−04−23
2021−05−07
as.factor(date)
as.factor(date)
2021−04−23
2021−05−07
Individual M08
CEWL_g_m2h
12
10
8
2021−04−23
2021−07−14
as.factor(date)
as.factor(date)
2021−04−23
2021−07−14
Individual M09
CEWL_g_m2h
8
6
4
2
2021−04−24
2021−05−07
2021−07−14
as.factor(date)
as.factor(date)
2021−04−24
2021−05−07
2021−07−14
Individual W010
CEWL_g_m2h
14
13
12
11
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24

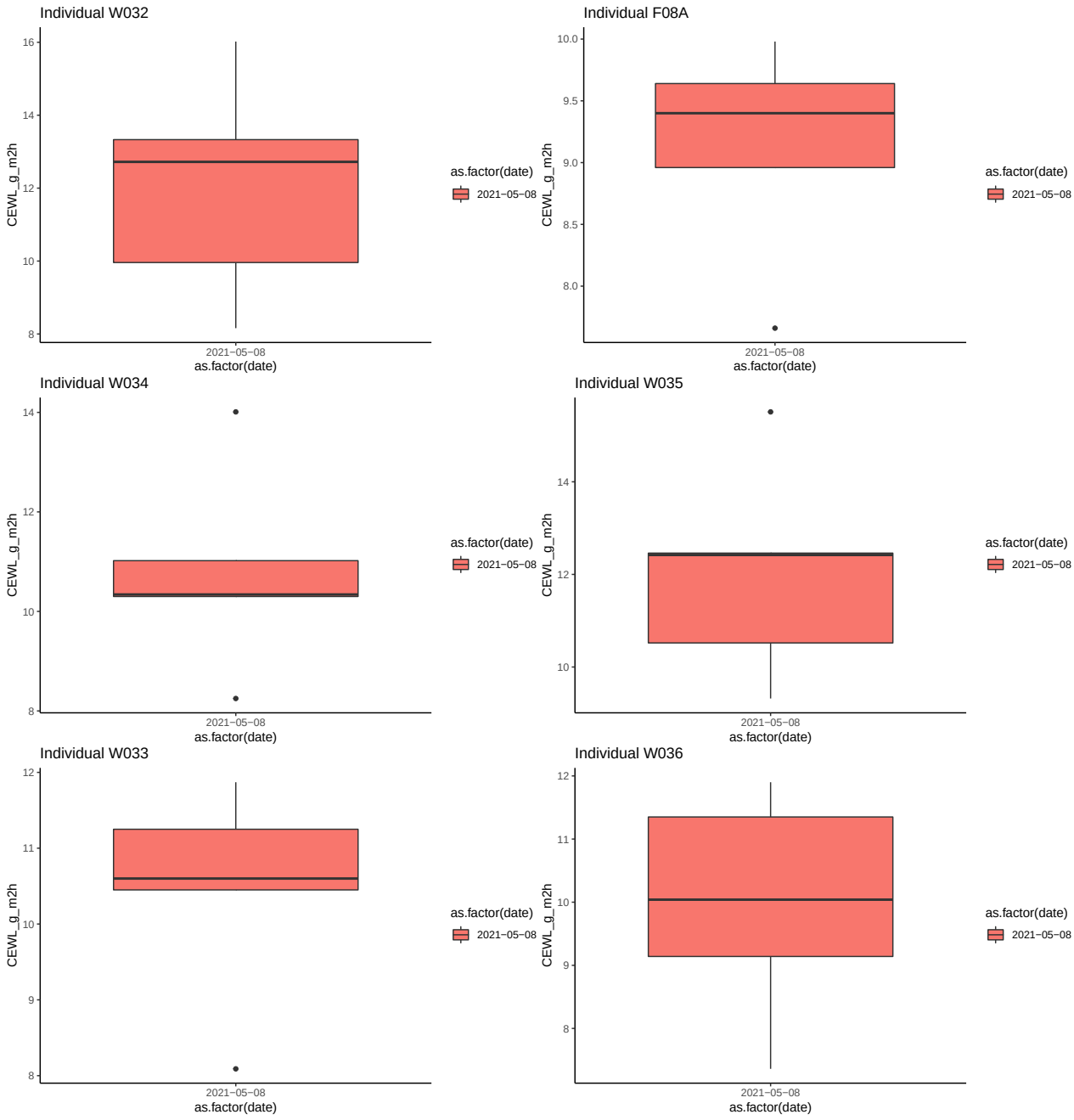

Individual W011
Individual W012
Individual W013
Individual W014
Individual W015
Individual W016
CEWL_g_m2h
as.factor(date)
2021−04−24
2021−05−08

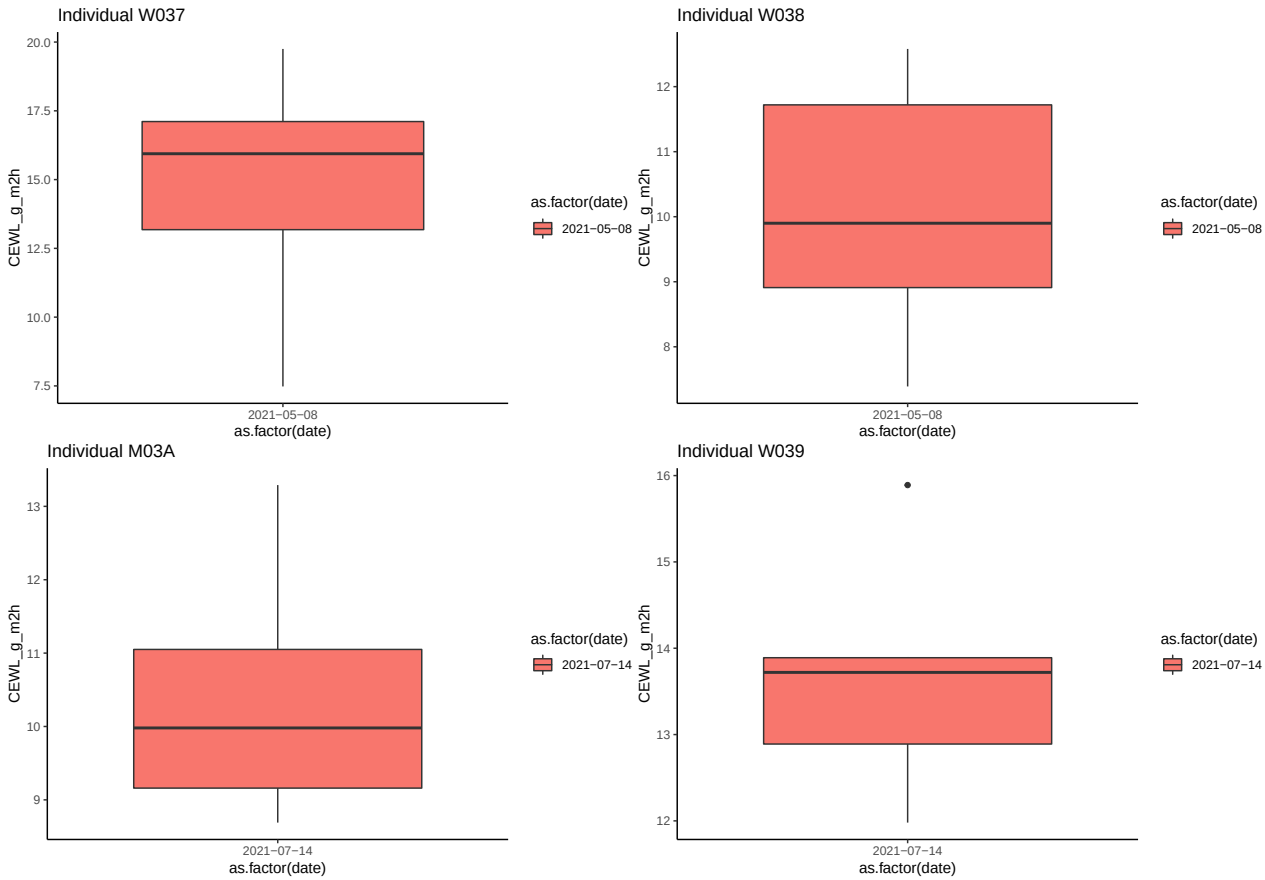

Individual W017
CEWL_g_m2h
as.factor(date)
2021−04−24
2021−05−08
Individual W018
Individual W019
Individual W002
2021−04−23
2021−05−07
Individual W020
Individual W021


Individual W022
CEWL_g_m2h
14
12
10
8
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual W023
CEWL_g_m2h
16
14
12
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual W024
CEWL_g_m2h
22.5
20.0
17.5
15.0
12.5
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual W025
CEWL_g_m2h
13
11
9
7
2021−04−24
2021−05−08
as.factor(date)
as.factor(date)
2021−04−24
2021−05−08
Individual W026
CEWL_g_m2h
40
30
20
10
2021−04−25
as.factor(date)
as.factor(date)
2021−04−25
Individual W027
CEWL_g_m2h
13
12
11
10
9
2021−04−25
as.factor(date)
as.factor(date)
2021−04−25


Individual W028
CEWL_g_m2h
15
14
13
12
11
2021−04−25
as.factor(date)
as.factor(date)
2021−04−25
Individual W003
CEWL_g_m2h
11
10
9
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual W004
CEWL_g_m2h
5.5
5.0
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual W005
CEWL_g_m2h
5
4
3
2
2021−04−23
as.factor(date)
as.factor(date)
2021−04−23
Individual W006
CEWL_g_m2h
12
11
10
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual W007
CEWL_g_m2h
16
15
14
13
12
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24


Individual W008
CEWL_g_m2h
19
18
17
16
15
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual W009
CEWL_g_m2h
11.4
11.0
10.6
10.2
2021−04−24
as.factor(date)
as.factor(date)
2021−04−24
Individual W029
CEWL_g_m2h
7
6
5
4
3
2021−05−07
as.factor(date)
as.factor(date)
2021−05−07
Individual W030
CEWL_g_m2h
6
4
2
2021−05−07
as.factor(date)
as.factor(date)
2021−05−07
Individual W031
CEWL_g_m2h
10.0
7.5
5.0
2.5
0.0
2021−05−07
as.factor(date)
as.factor(date)
2021−05−07
Individual W001
CEWL_g_m2h
25
20
15
10
2021−05−08
as.factor(date)
as.factor(date)
2021−05−08


Individual W032
Individual F08A
Individual W034
Individual W035
Individual W033
Individual W036
CEWL_g_m2h
as.factor(date)
2021−05−08



Based on the plots, the dataframe of outliers I compiled is correct. (yay!)

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 == (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)
nrow(all_CEWL_data) == nrow(outliers_omitted) + nrow(outliers_found)
```

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

Re-Assess Variation

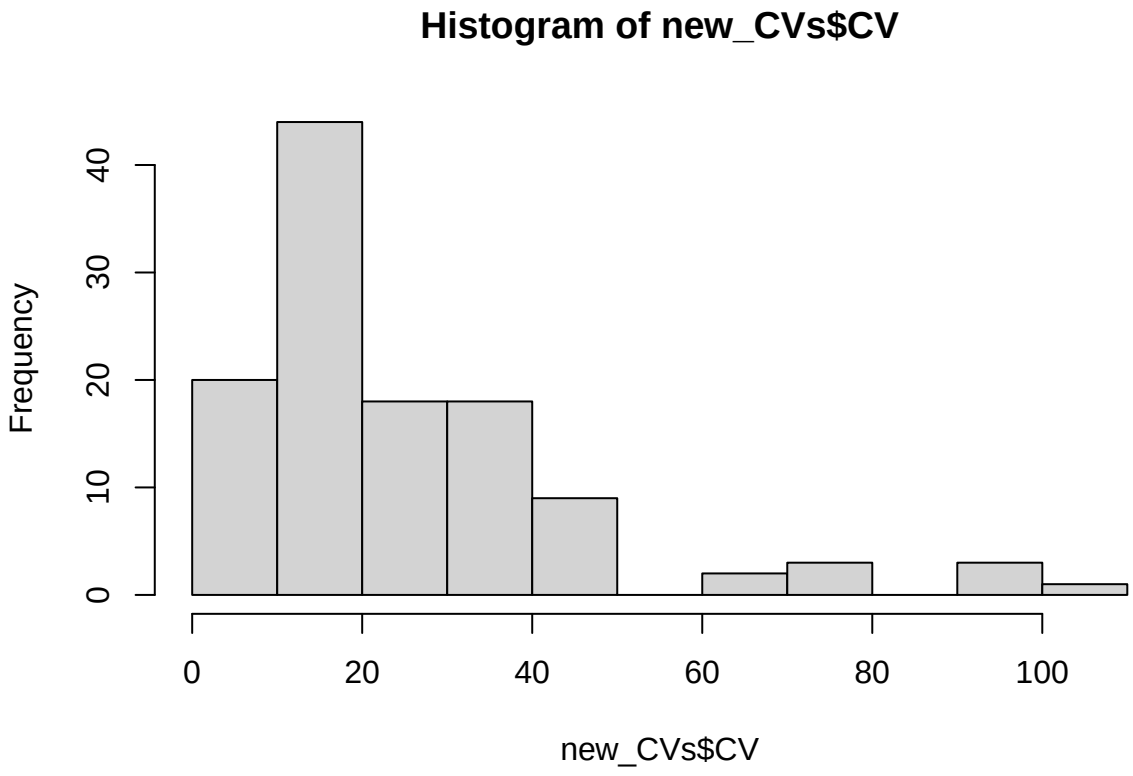
```
new_CVs <- outliers_omitted %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100,
            min = min(CEWL_g_m2h),
            max = max(CEWL_g_m2h),
            range = max - min)
```

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

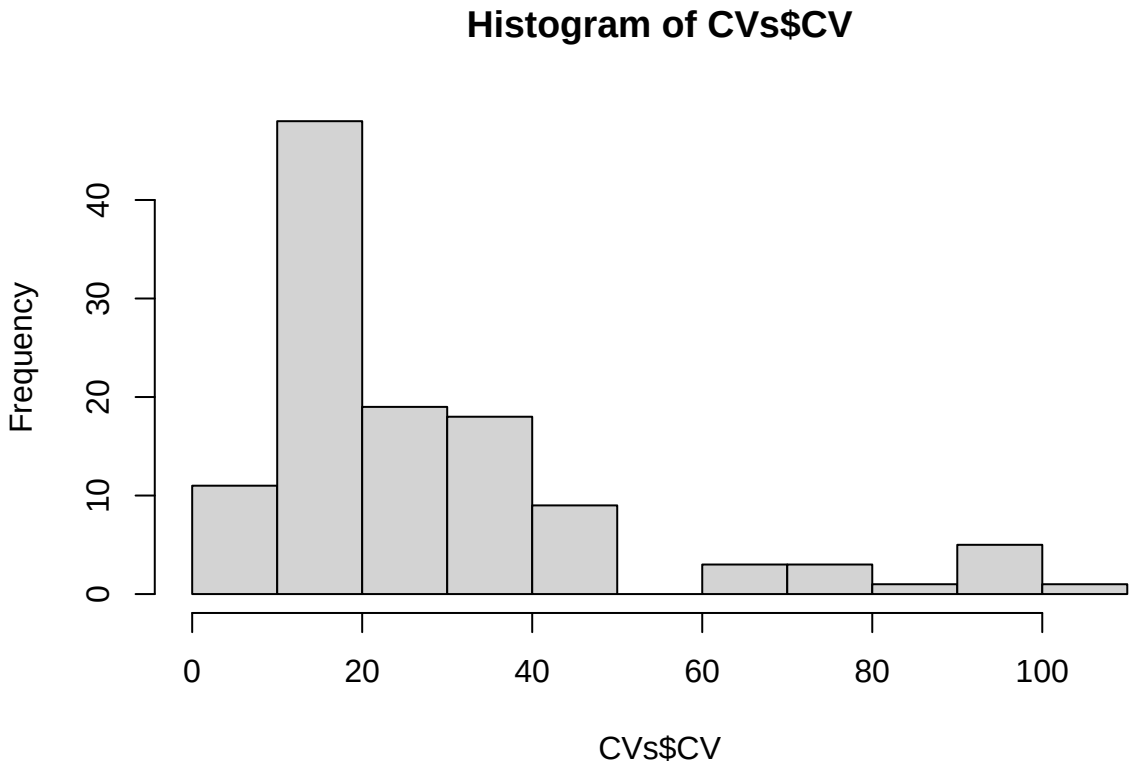
```
summary(new_CVs)
```

```
##  individual_ID      date                mean             SD
##  F12    :  3   Min.   :2021-04-23   Min.   : 0.650   Min.   : 0.05508
##  M09    :  3   1st Qu.:2021-04-24   1st Qu.: 8.486   1st Qu.: 1.21719
##  M10    :  3   Median :2021-04-24   Median :10.421   Median : 1.85776
##  M11    :  3   Mean   :2021-05-08   Mean   :10.682   Mean   : 2.65196
##  M19    :  3   3rd Qu.:2021-05-08   3rd Qu.:13.239   3rd Qu.: 2.88268
##  M20    :  3   Max.   :2021-07-14   Max.   :31.550   Max.   :29.32424
##  (Other):100
##        CV               min              max             range
##  Min.   :  1.032   Min.   :-1.320   Min.   : 1.120   Min.   : 0.100
##  1st Qu.: 13.433   1st Qu.: 6.723   1st Qu.: 9.985   1st Qu.: 2.518
##  Median : 19.265   Median : 8.420   Median :12.545   Median : 4.060
##  Mean   : 25.543   Mean   : 8.287   Mean   :13.639   Mean   : 5.352
##  3rd Qu.: 33.436   3rd Qu.:10.502   3rd Qu.:15.520   3rd Qu.: 6.197
##  Max.   :105.713   Max.   :19.640   Max.   :65.310   Max.   :52.900
##
```

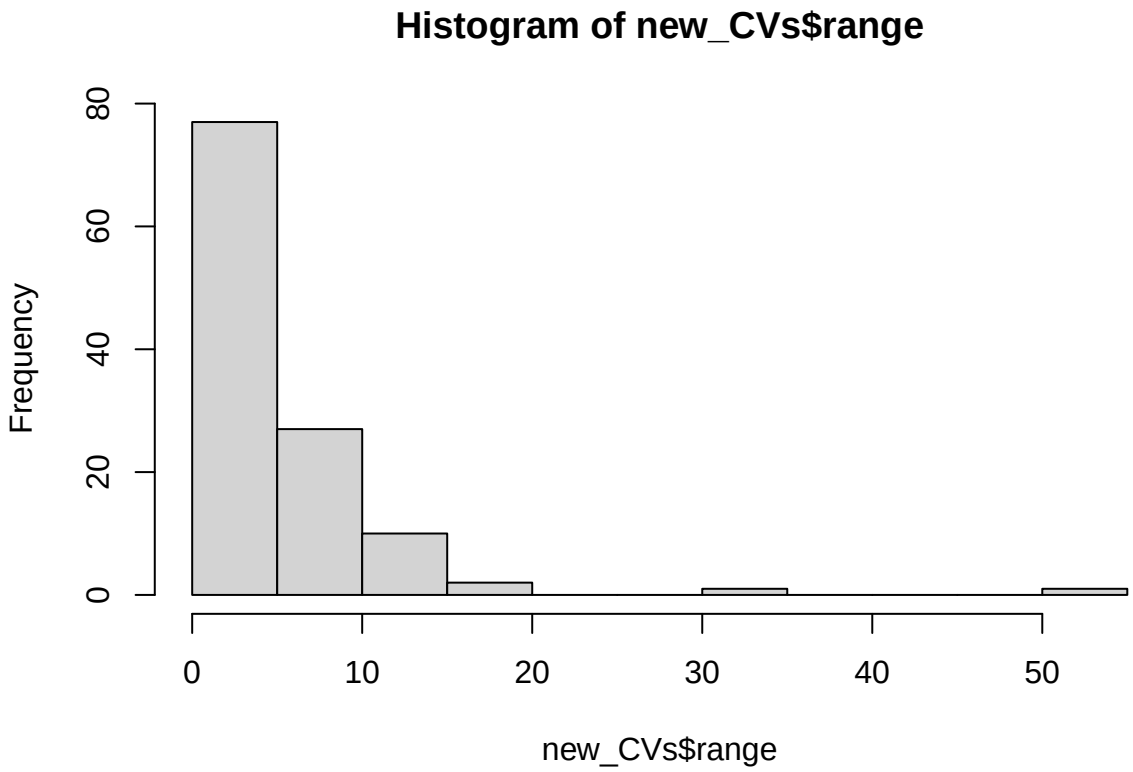
```
hist(new_CVs$CV)
```



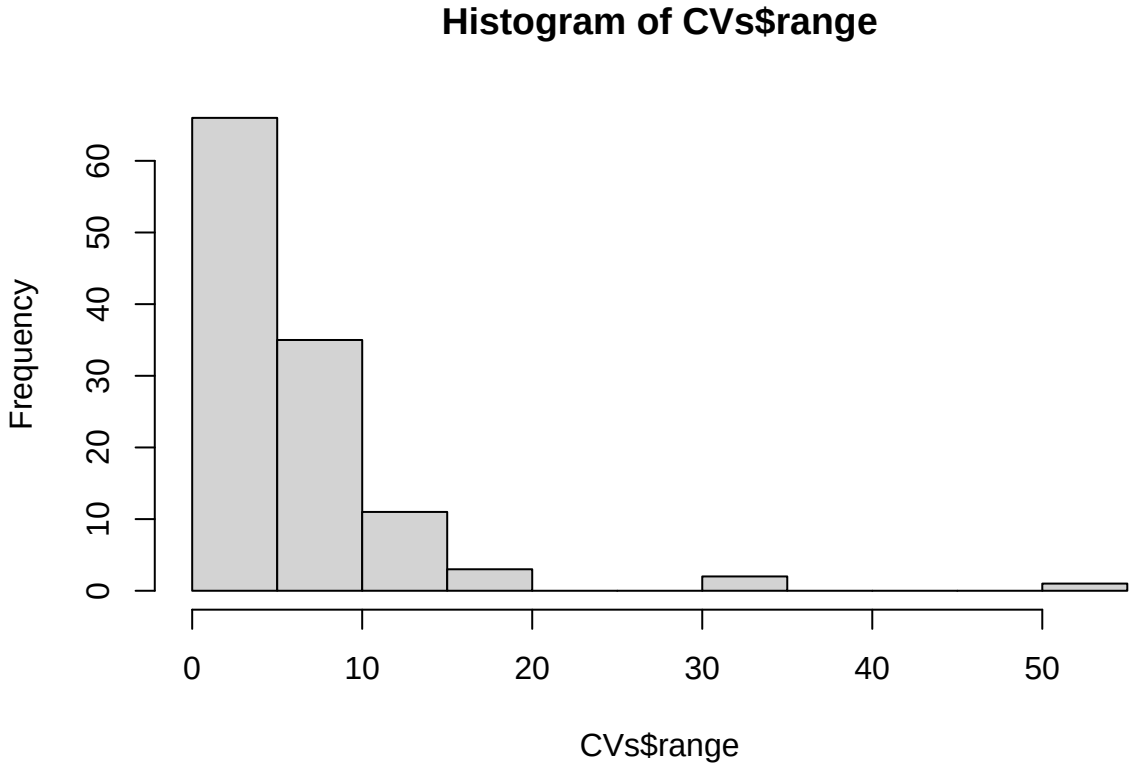

```
hist(CVs$CV)
```




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




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



Unfortunately, CVs are still skewed to the right, but many CVs were improved. We will continue with this dataset.

Average Replicates (outliers removed)

```
CEWL_final <- outliers_omitted %>%
  group_by(date, individual_ID) %>%
  summarise(CEWL_g_m2h = mean(CEWL_g_m2h),
            msmt_temp_C = mean(msmt_temp_C),
            msmt_RH_percent = mean(msmt_RH_percent))
```

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

```
head(CEWL_final)
```

```
## # A tibble: 6 x 5
## # Groups:   date [1]
##   date       individual_ID CEWL_g_m2h msmt_temp_C msmt_RH_percent
##   <date>     <fct>               <dbl>       <dbl>           <dbl>
## 1 2021-04-23 F01                 10.4         31.7            12.2
## 2 2021-04-23 F02                 19.2         33.4            16.8
## 3 2021-04-23 F03                  8.40        32.0            14.2
## 4 2021-04-23 F04                  9.21        25.0            26.0
## 5 2021-04-23 F11                  8.96        31.9            14.0
## 6 2021-04-23 F12                  1.49        32.2            13.5
```

Final Synthesis

Re-Check Data

Check that we still have data for every individual.

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.

```
unique(CEWL_final$individual_ID) %in% unique(all_CEWL_data$individual_ID)
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [31] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [46] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [61] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [76] TRUE TRUE TRUE TRUE TRUE
```

```
unique(all_CEWL_data$individual_ID) %in% unique(CEWL_final$individual_ID)
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [31] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [46] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [61] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [76] TRUE TRUE TRUE TRUE TRUE
```

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: 118 x 3
## # Groups:   individual_ID [80]
##    individual_ID date           n
##    <fct>         <date>     <int>
##  1 F01           2021-04-23     3
##  2 F02           2021-04-23     3
##  3 F03           2021-04-23     3
##  4 F04           2021-04-23     3
##  5 F05           2021-04-24     3
##  6 F06           2021-04-24     3
##  7 F07           2021-04-24     3
##  8 F08           2021-04-24     3
##  9 F09           2021-04-24     3
## 10 F10           2021-04-24     3
## # ... with 108 more rows
```

Between 3-5, 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 24 measurements from our CEWL dataset. We used the boxplot.stats function in R to extract outliers from each set of technical replicates, totaling 24 points qualifying as outliers which were thus removed. After data cleaning, every individual still had 3-5 technical replicates for each of their measurement dates. The distribution of coefficient of variation values was slightly better after data cleaning than before. For the individuals who only had 3 replicates taken, they probably didn't have enough reps for outliers to be found and omitted, so that's probably what made CVs stay high.