

BNLL CEWL Data Wrangling

Savannah Weaver

Contents

Packages	1
Background and Goals	1
Load Data	1
Check Data	3
Replicates	4
Assess Variation	4
Find Outliers	6
Remove Outliers	22
Re-Assess Variation	23
Find “Outliers”	26
Remove Outliers	28
Average Replicates (outliers removed)	32
Final Synthesis	32
Re-Check Data	32
Export	33
Reporting	33

Packages

Background and Goals

This CEWL (cutaneous evaporative water loss) data was measured in 3-5 technical replicates on the mid-dorsum of Blunt-nosed Leopard Lizards (*Gambelia sila*) between April - July 2021. In this R script, I 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 for each lizard, and those values will be used in the analyses R script file. Please refer to **doi:** for the published scientific paper 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(paste(substr(ID_rep_no, 1, 1),
        substr(ID_rep_no, 2, 4),
        sep = "-")),
      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(paste(substr(ID_rep_no, 1, 1),
        substr(ID_rep_no, 2, 3),
        sep = "-"))
    ),

    # 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.   :2022-07-03 01:00:00   Normal:456
## 1st Qu.:2021-04-24   1st Qu.:2022-07-03 02:24:45
## Median :2021-05-07   Median :2022-07-03 03:46:00
## Mean   :2021-05-12   Mean   :2022-07-03 04:22:43
## 3rd Qu.:2021-05-08   3rd Qu.:2022-07-03 05:02:15
## Max.   :2021-07-14   Max.   :2022-07-03 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
## F-12   : 13   1:117
## M-10   : 13   2:118
## M-11   : 13   3:118
## M-19   : 13   4: 52
## M-20   : 13   5: 51
## M-09   : 12
## (Other):379
```

```
unique(all_CEWL_data$individual_ID)
```

```
## [1] F-01 F-10 F-11 F-12 F-13 F-14 F-15 F-16 F-17 F-18 F-19 F-02
## [13] F-03 F-04 F-05 F-06 F-07 F-08 F-09 M-01 M-10 M-11 M-12 M-13
## [25] M-14 M-15 M-16 M-17 M-18 M-19 M-02 M-20 M-03 M-04 M-05 M-06
## [37] M-07 M-08 M-09 W-010 W-011 W-012 W-013 W-014 W-015 W-016 W-017 W-018
## [49] W-019 W-002 W-020 W-021 W-022 W-023 W-024 W-025 W-026 W-027 W-028 W-003
## [61] W-004 W-005 W-006 W-007 W-008 W-009 W-029 W-030 W-031 W-001 W-032 F-08A
## [73] W-034 W-035 W-033 W-036 W-037 W-038 M-03A W-039
## 80 Levels: F-01 F-02 F-03 F-04 F-05 F-06 F-07 F-08 F-09 F-10 F-11 F-12 ... W-039
```

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 F-01         2021-04-23      3 120 secs
## 2 F-02         2021-04-23      3 120 secs
## 3 F-03         2021-04-23      3 120 secs
## 4 F-04         2021-04-23      3  60 secs
## 5 F-05         2021-04-24      3 120 secs
## 6 F-06         2021-04-24      3 120 secs
## 7 F-07         2021-04-24      3  60 secs
## 8 F-08         2021-04-24      3  60 secs
## 9 F-09         2021-04-24      3 120 secs
## 10 F-10        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"))

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

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
  )
```

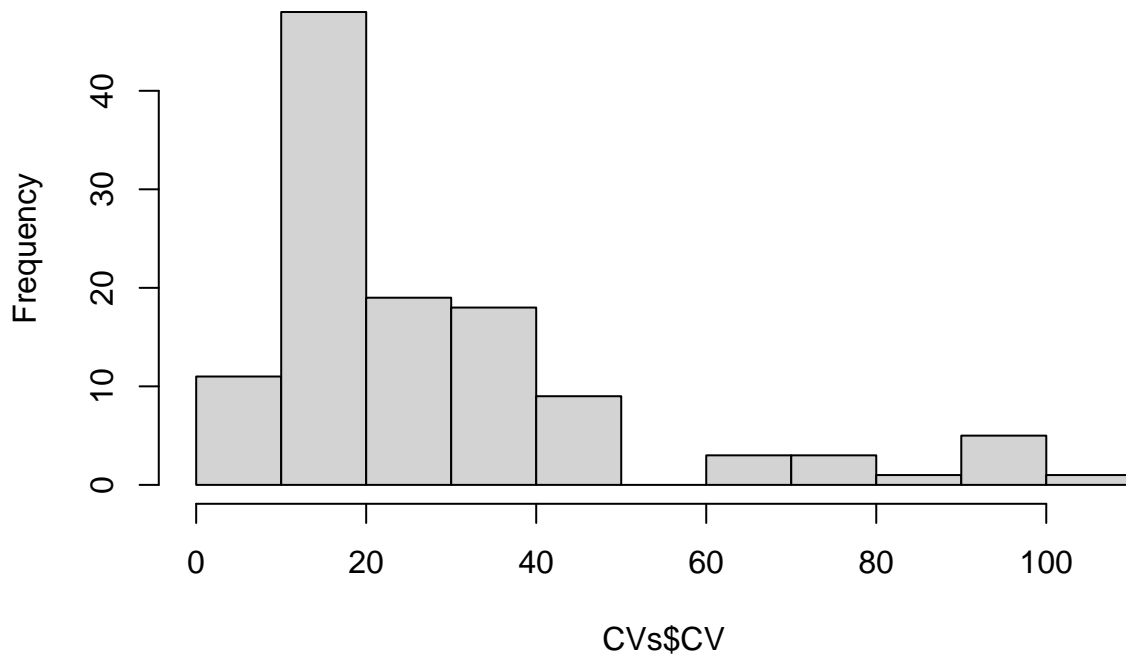
```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
```

```
summary(CVs)
```

```
## individual_ID      date      mean      SD
## F-12      : 3  Min.    :2021-04-23  Min.    : 0.650  Min.    : 0.1124
## M-09      : 3  1st Qu.:2021-04-24  1st Qu.: 8.486  1st Qu.: 1.4849
## M-10      : 3  Median :2021-04-24  Median :10.443  Median : 2.0290
## M-11      : 3  Mean    :2021-05-08  Mean    :10.823  Mean    : 2.9641
## M-19      : 3  3rd Qu.:2021-05-08  3rd Qu.:13.391  3rd Qu.: 3.1195
## M-20      : 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
##
```

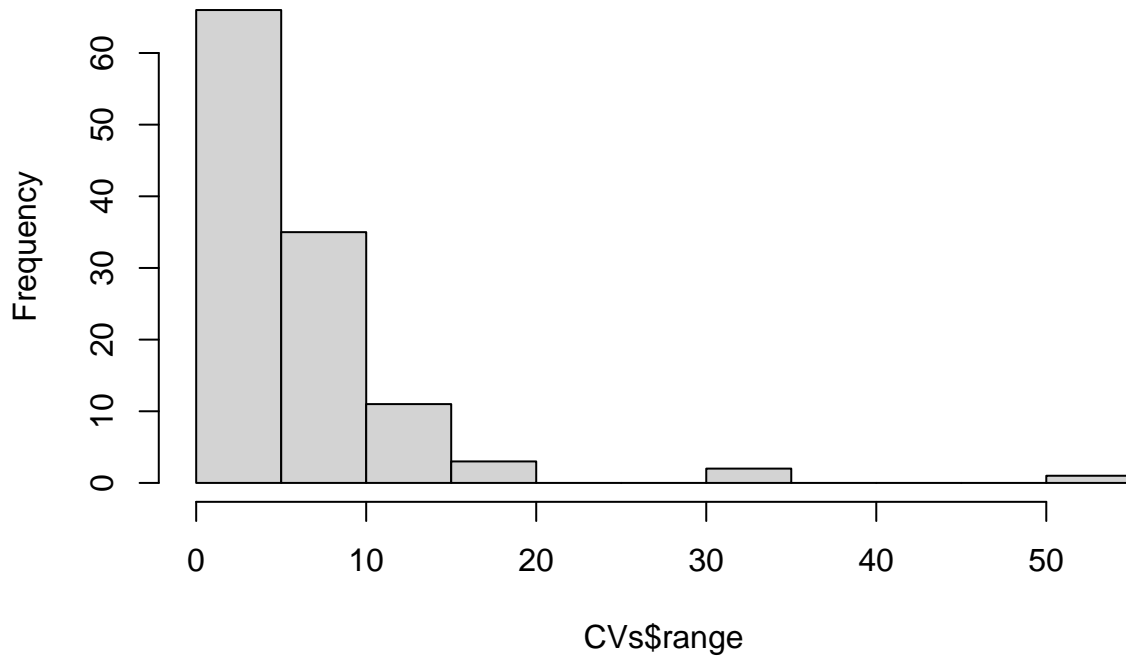
```
hist(CVs$CV)
```

Histogram of CVs\$CV



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

Histogram of CVs\$range



We expect CV for technical replicates to be $< 10\text{-}15\%$, so we must determine whether the CVs $> 15\%$ are due to outlier replicates. The range should also generally be within 5 units for these measurements. :(

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:

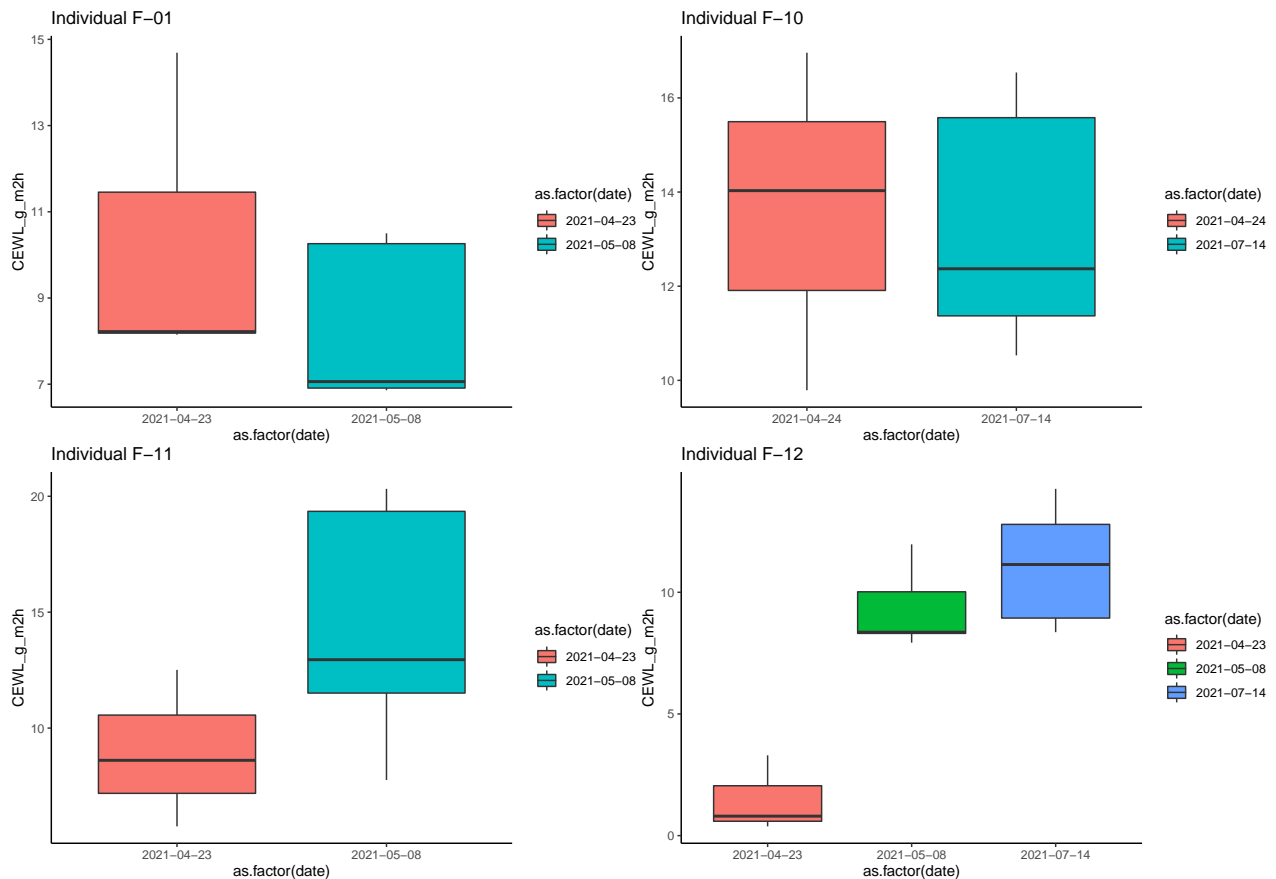
```
par(mfrow = c(71, 2))
outliers_found <- find_outliers(all_CEWL_data)
```

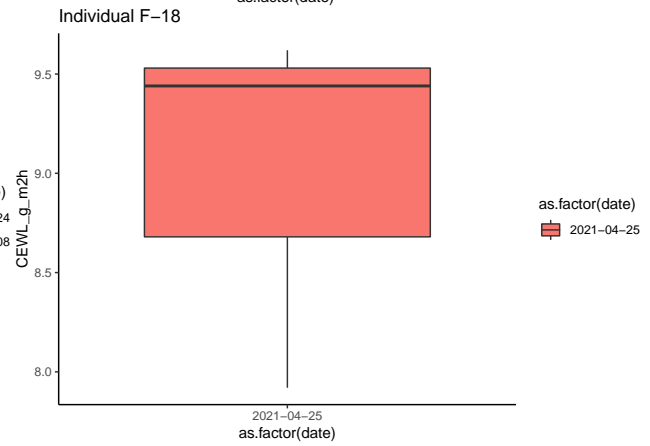
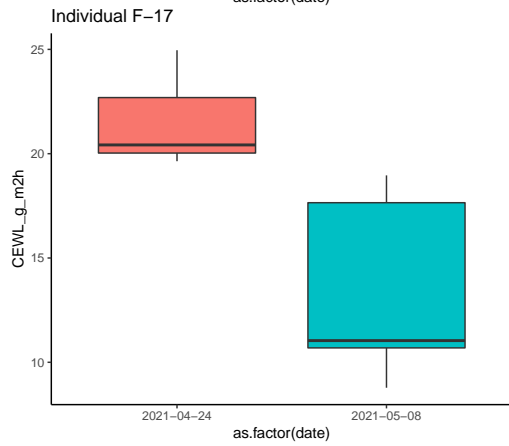
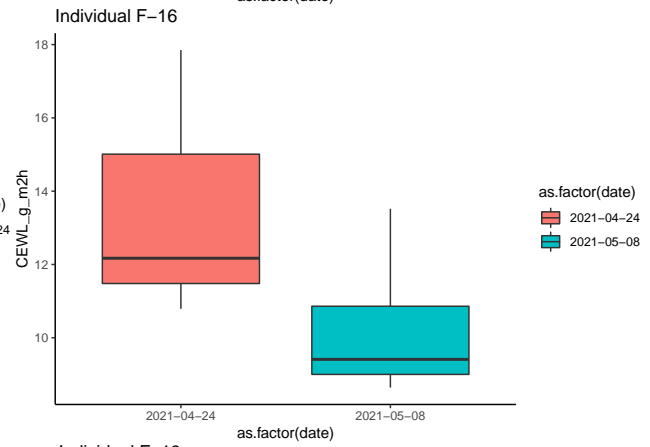
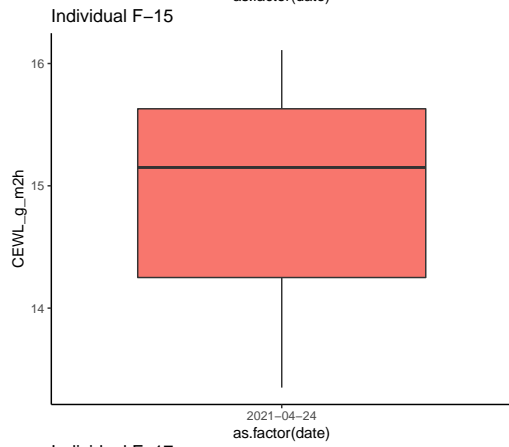
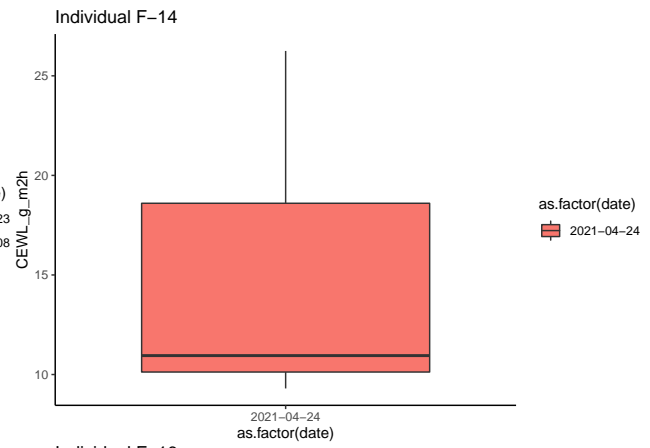
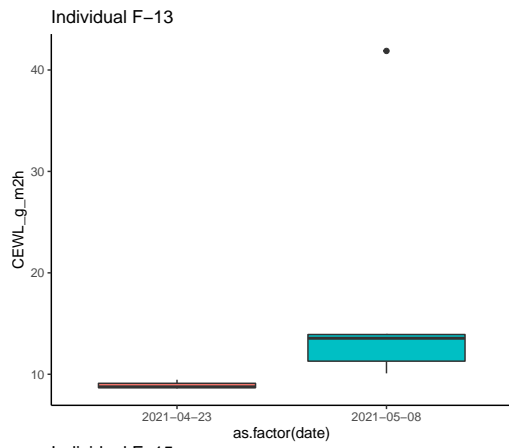
[illegible]

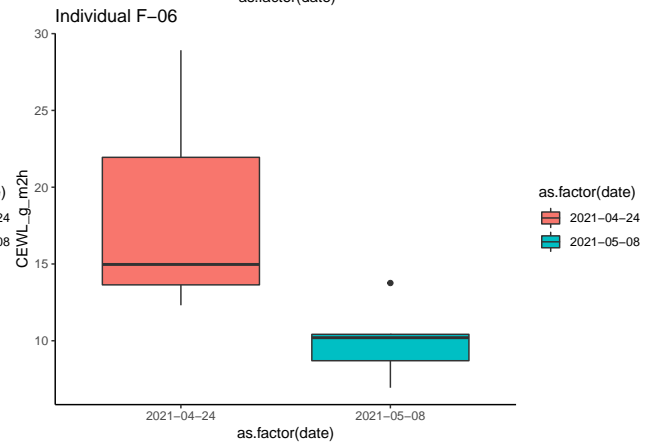
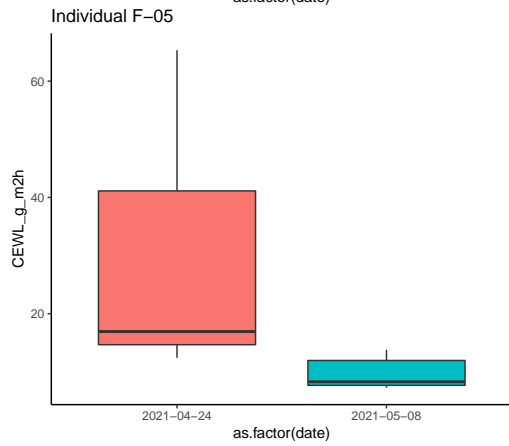
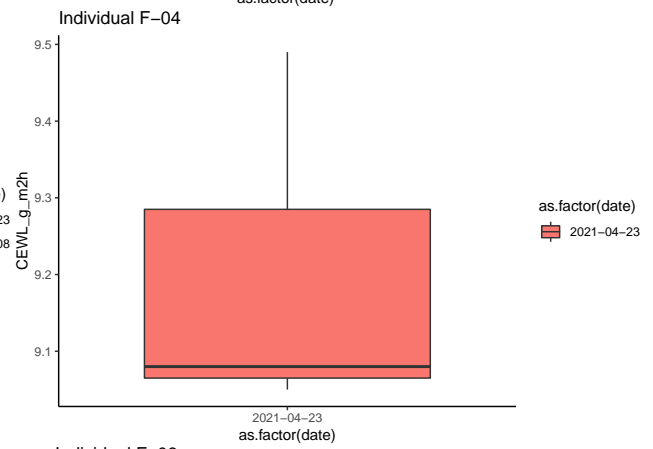
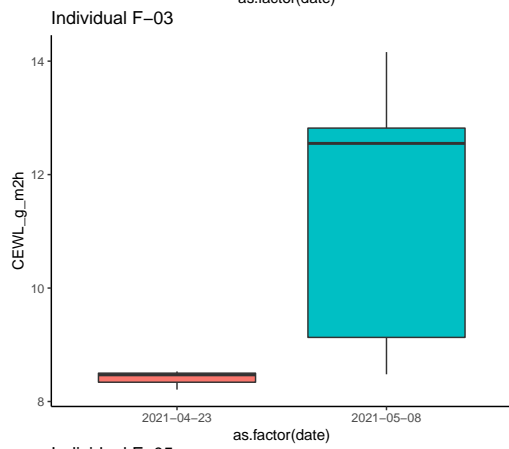
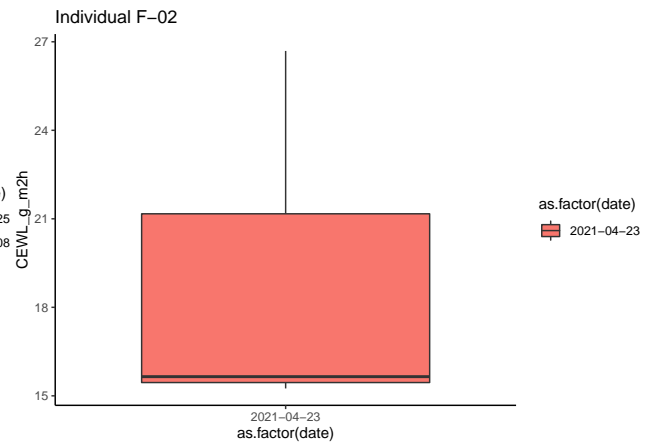
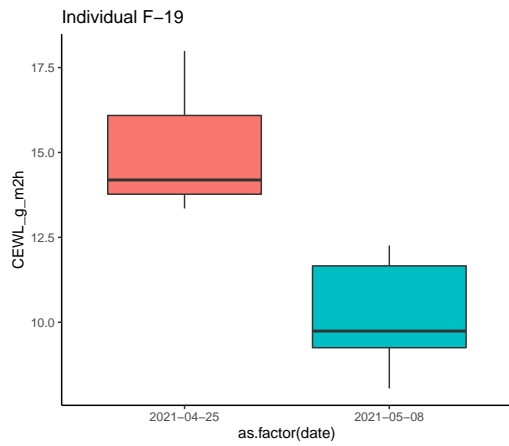

```
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
outliers_found
```

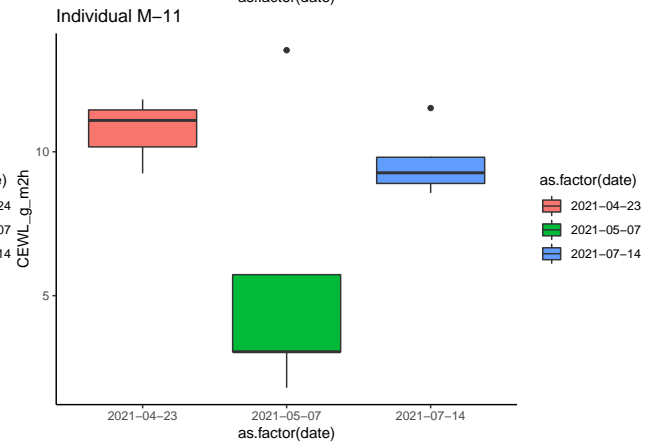
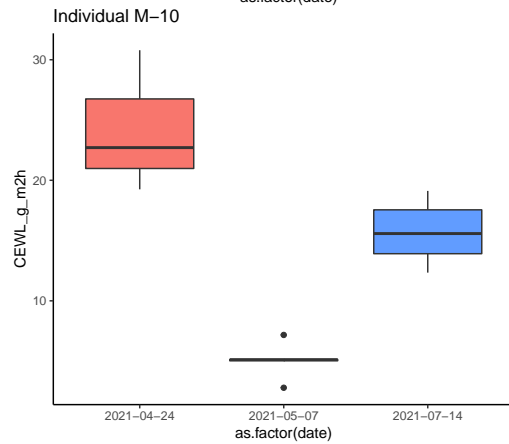
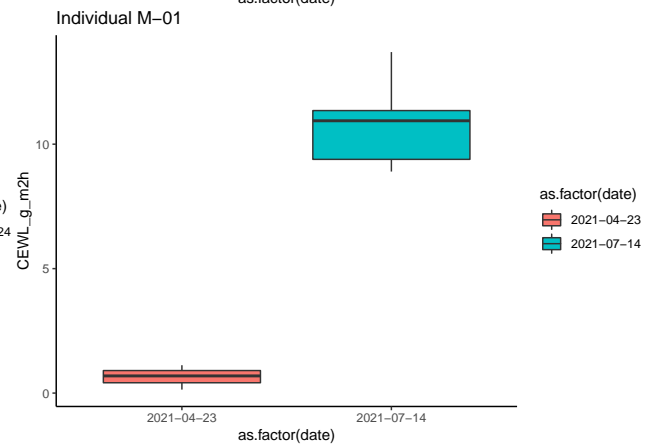
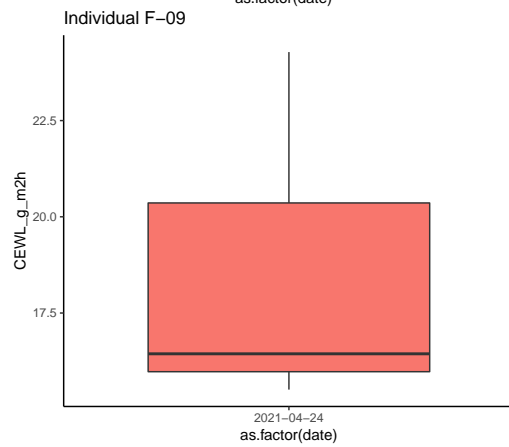
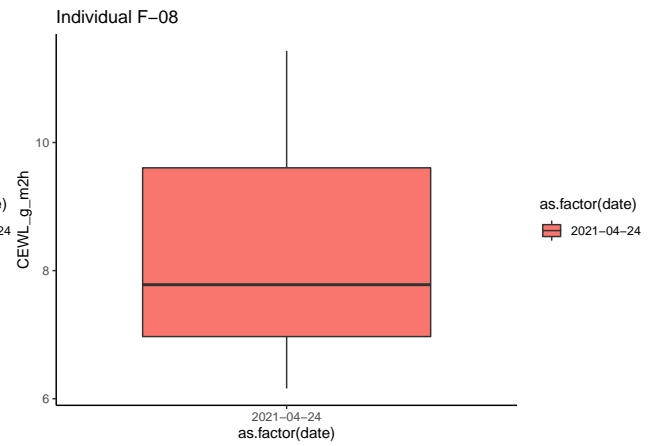
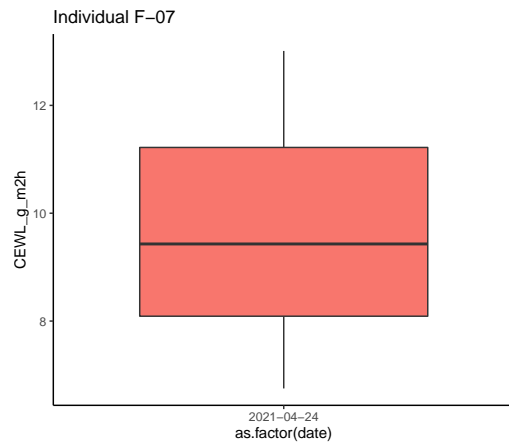
```
## # A tibble: 24 x 3
## # Groups:   individual_ID, date [18]
##   individual_ID date      outs
##   <fct>         <date>    <dbl>
## 1 F-13          2021-05-08 41.9
## 2 F-06          2021-05-08 13.8
## 3 M-10          2021-05-07  7.17
## 4 M-10          2021-05-07  2.79
## 5 M-11          2021-05-07 13.5
## 6 M-11          2021-07-14 11.5
## 7 M-13          2021-07-14 17.1
## 8 M-13          2021-07-14 11.3
## 9 M-19          2021-07-14 17.9
## 10 M-20         2021-07-14 11.3
## # ... with 14 more rows
```

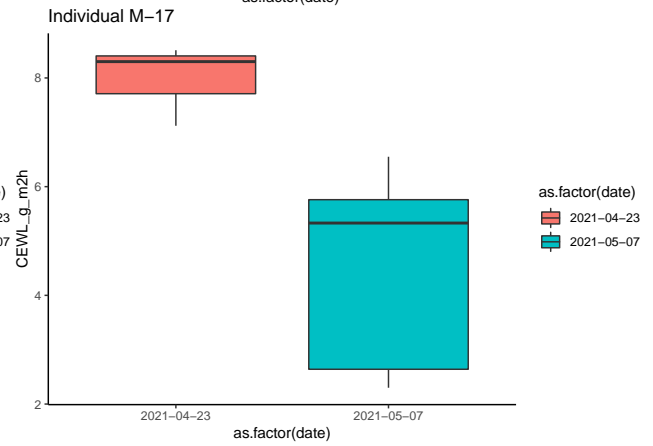
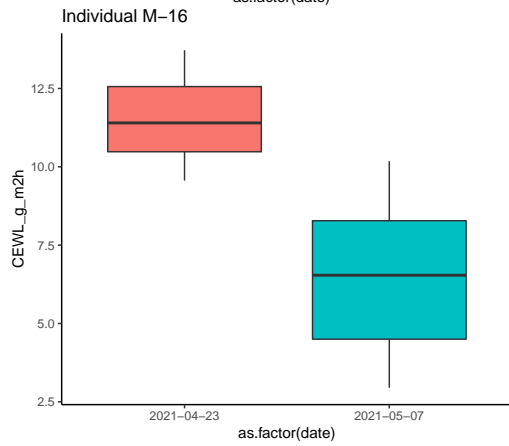
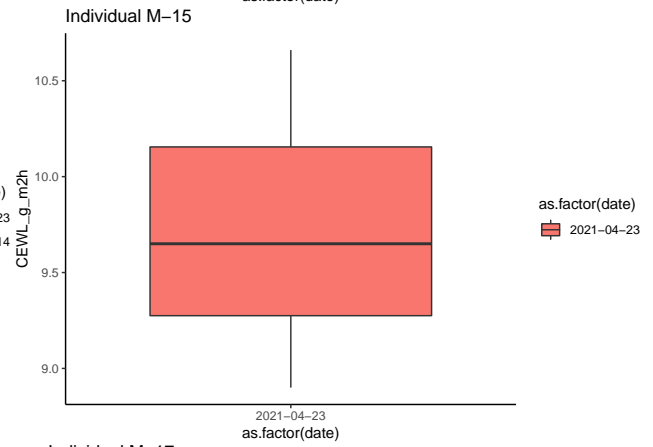
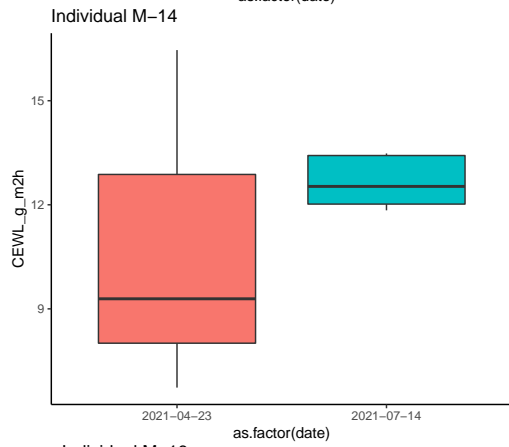
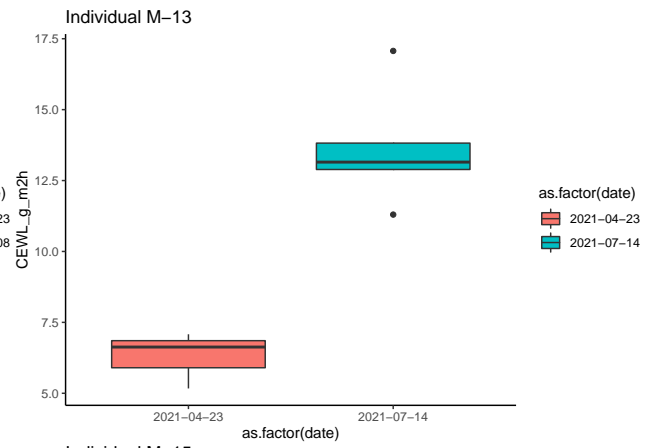
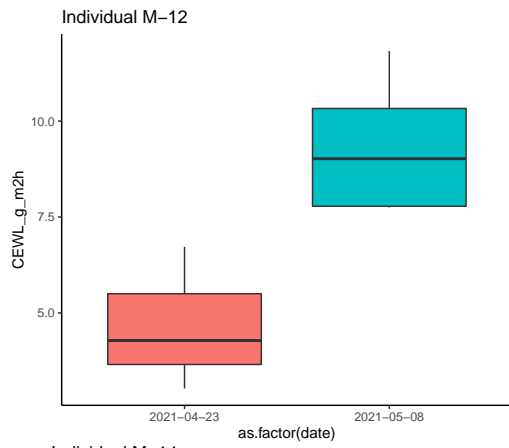
```
par(mfrow = c(1, 1))
```

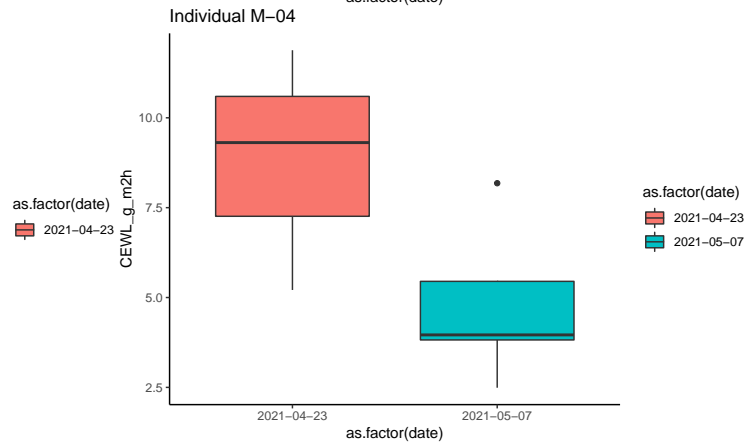
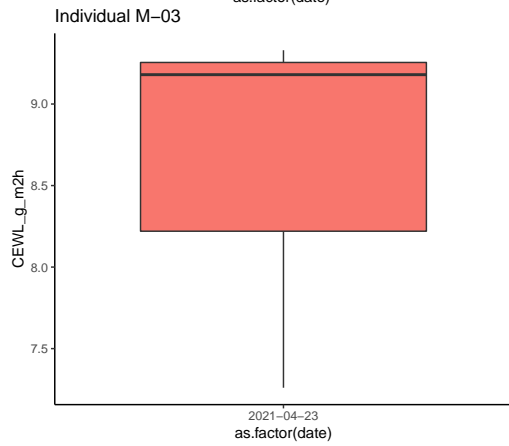
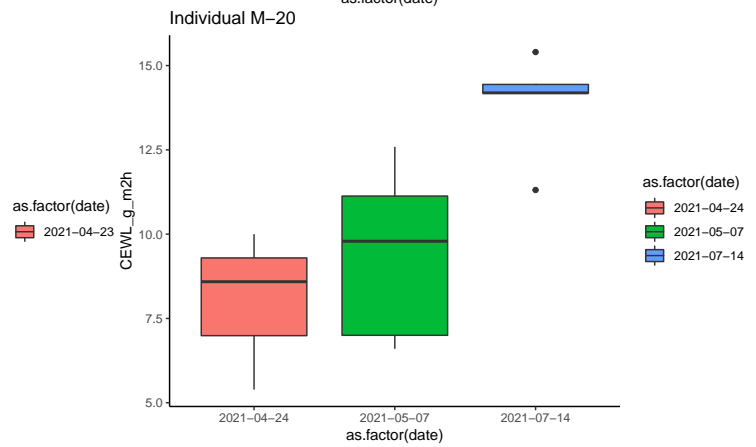
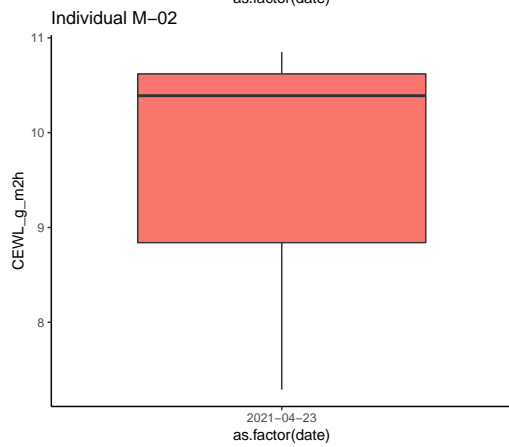
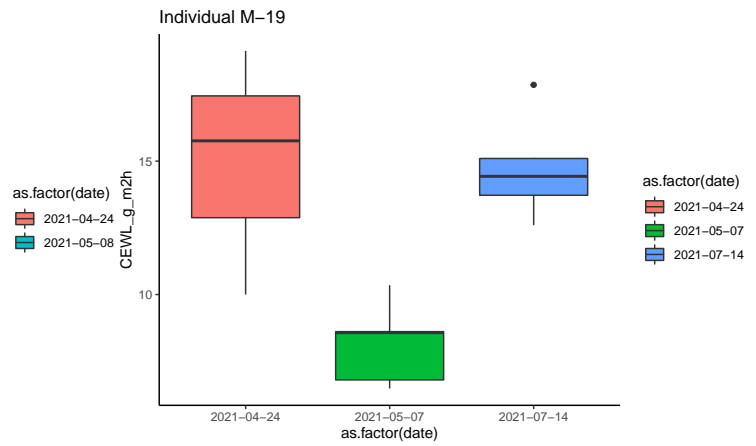
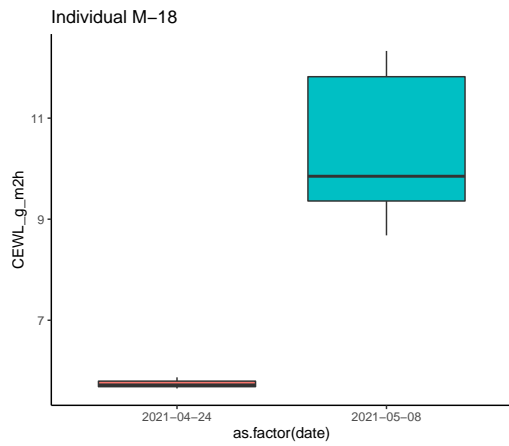


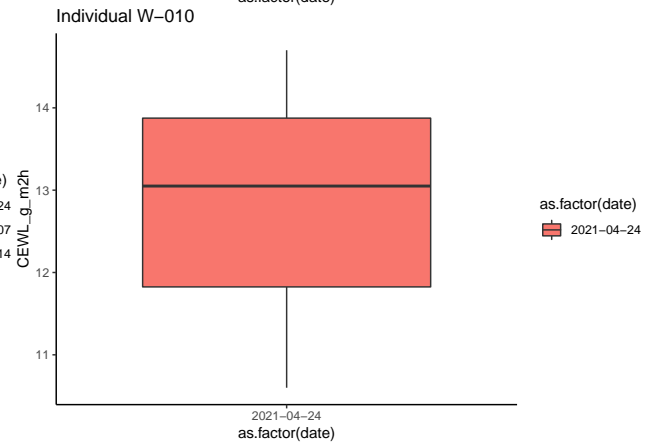
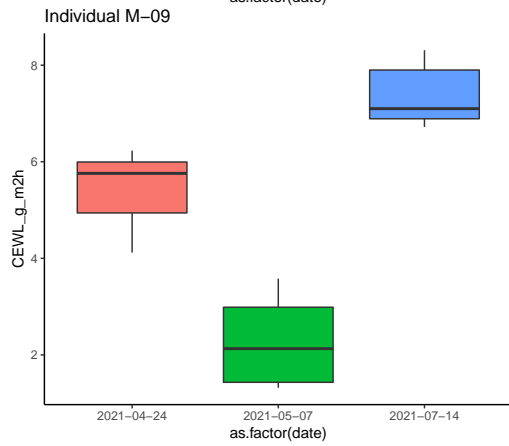
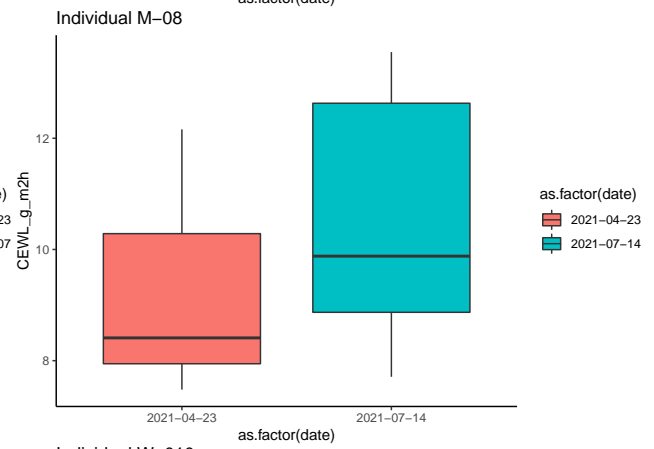
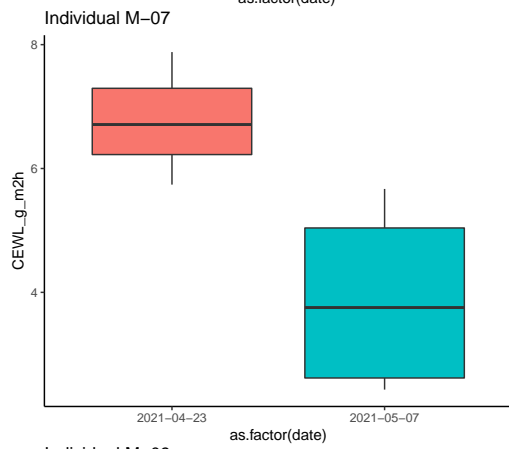
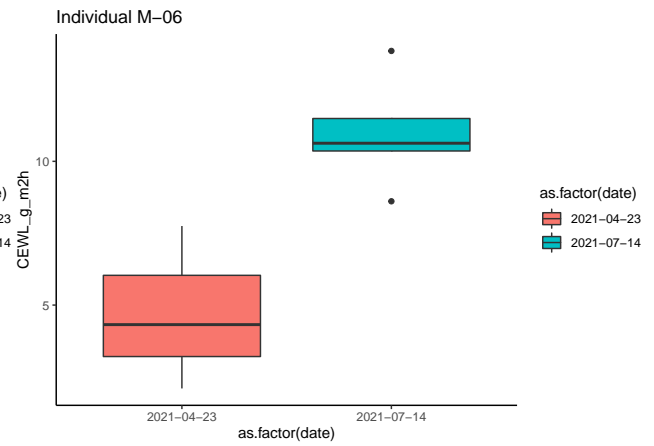
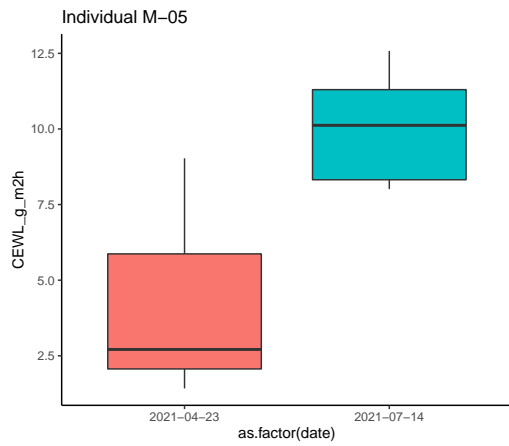


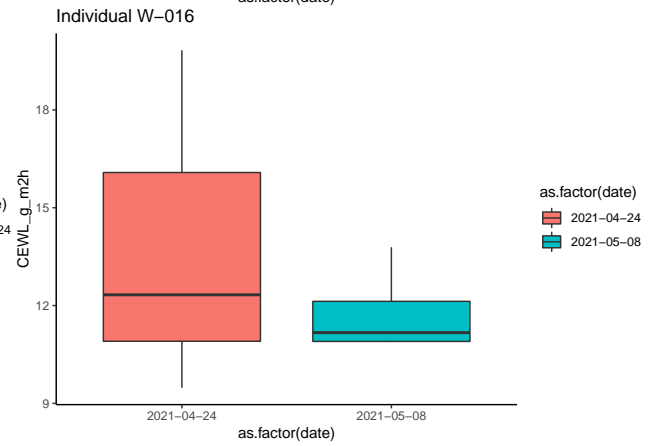
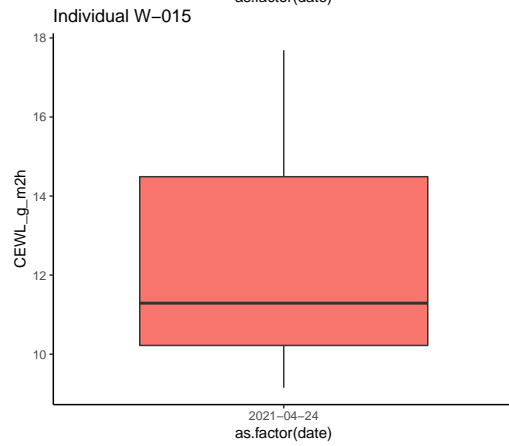
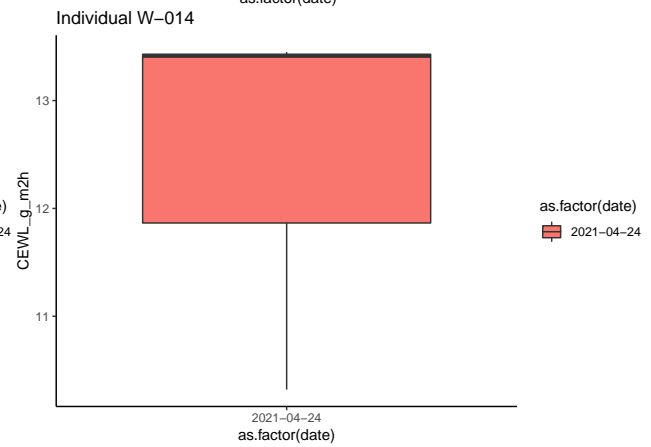
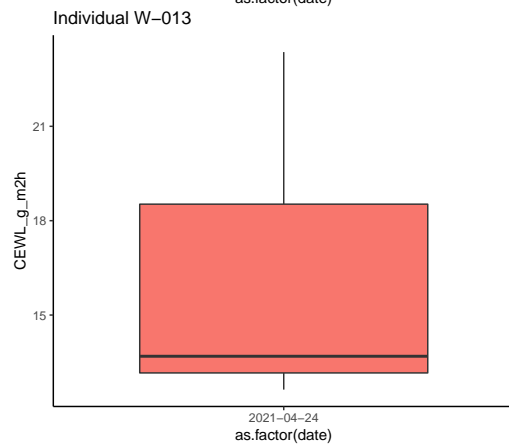
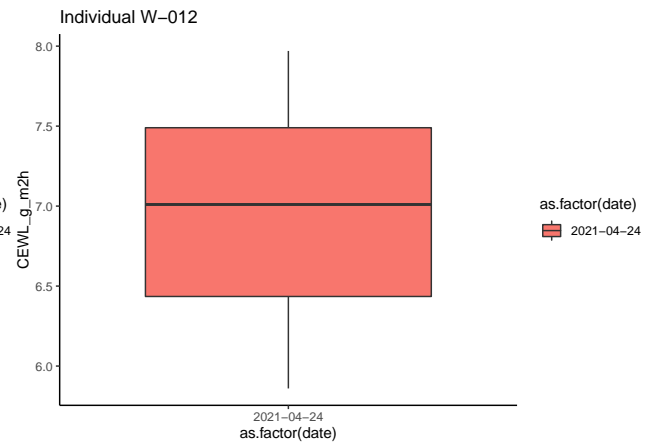
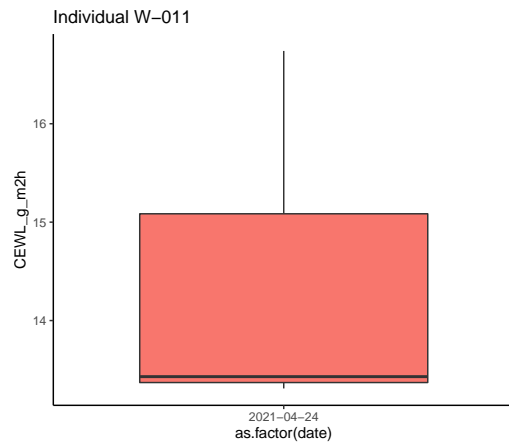


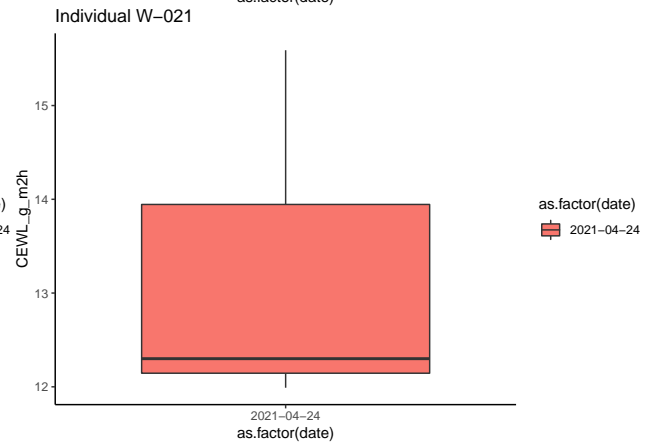
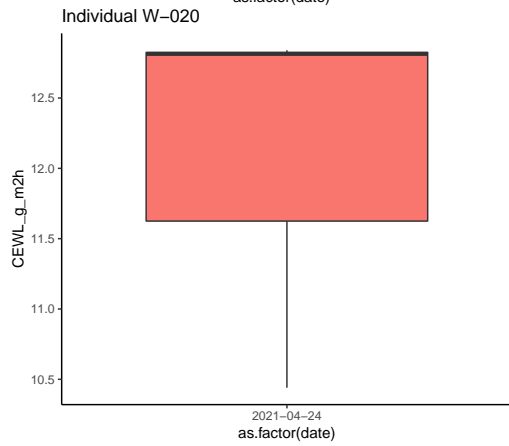
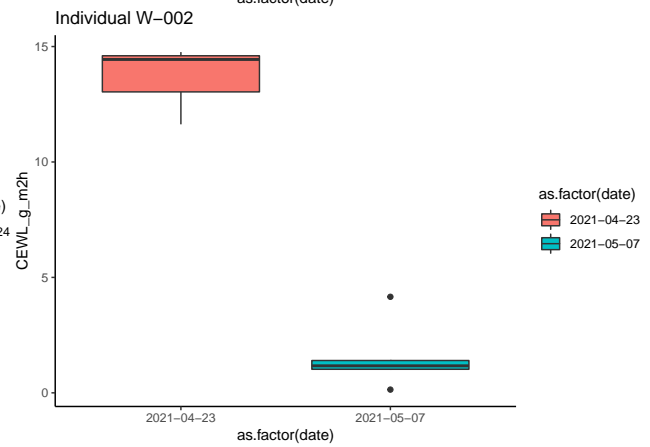
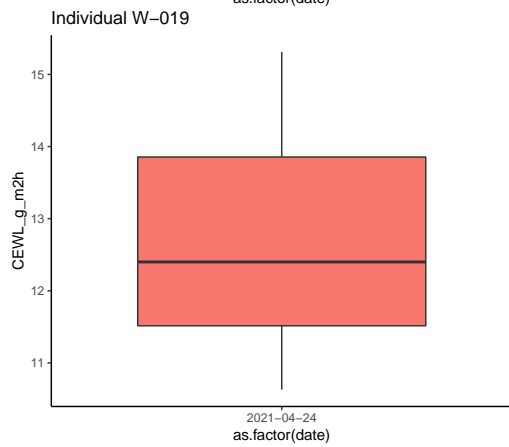
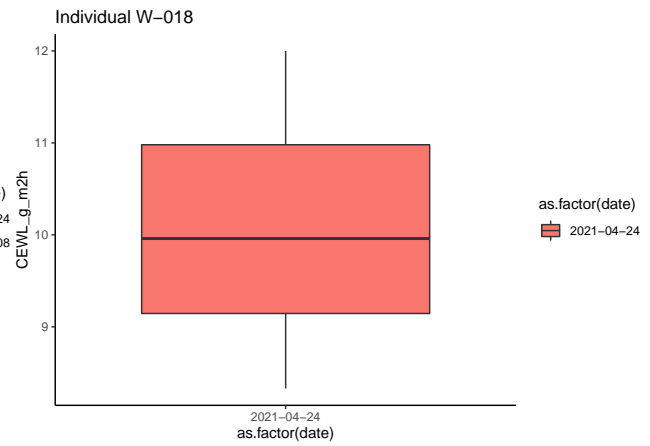
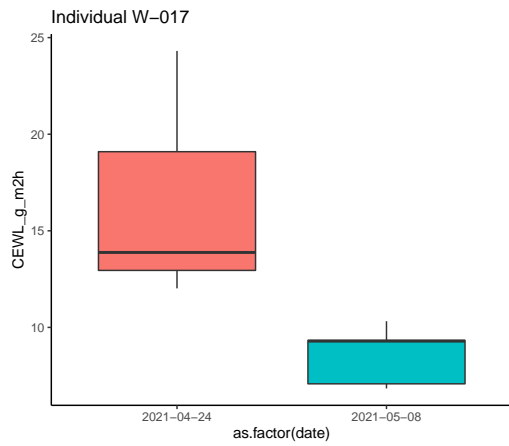


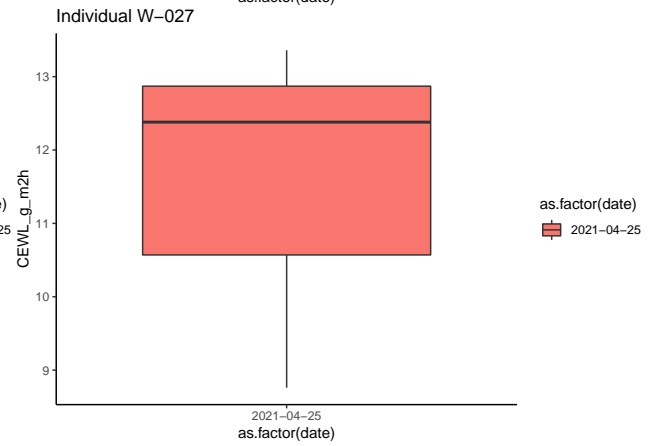
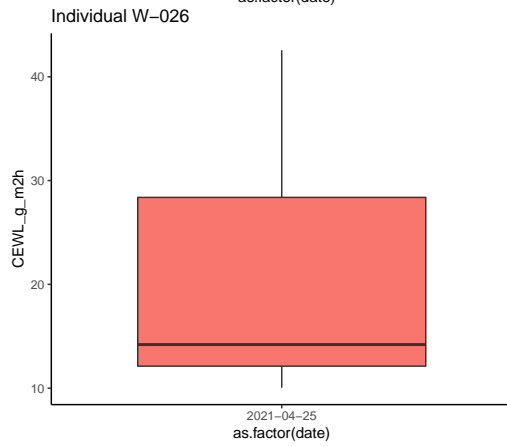
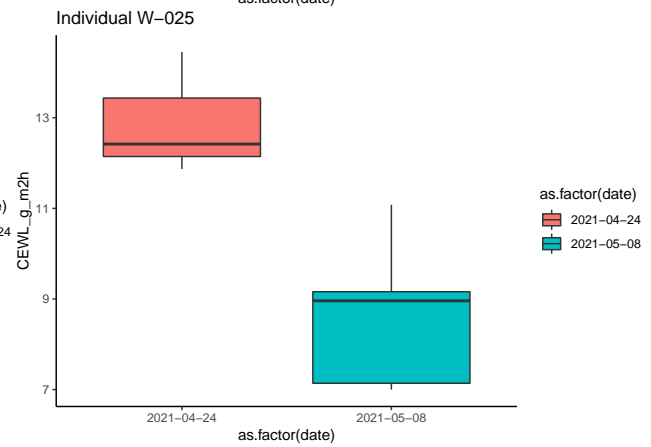
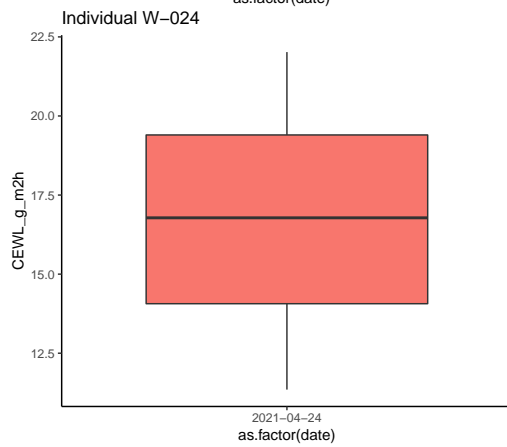
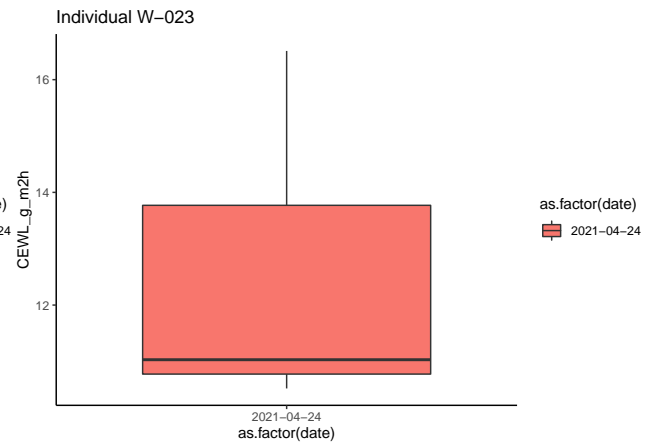
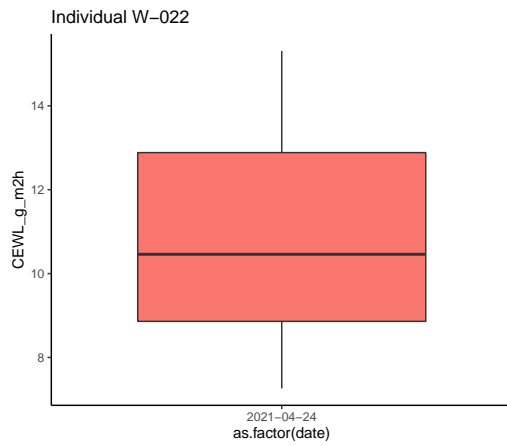


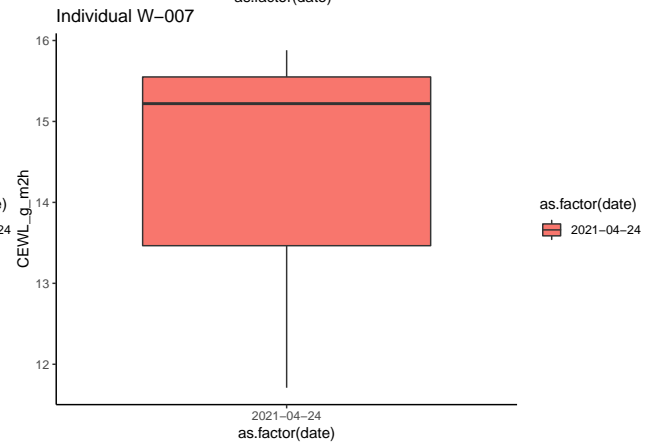
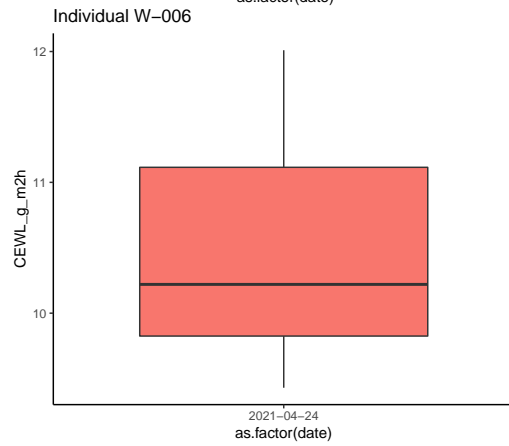
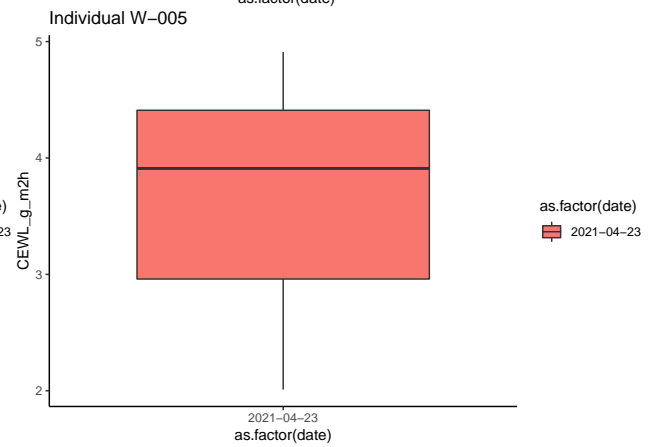
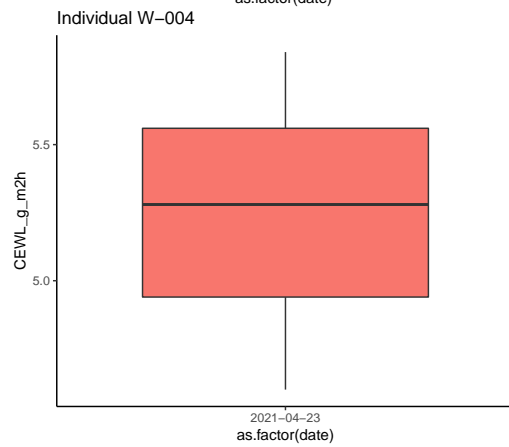
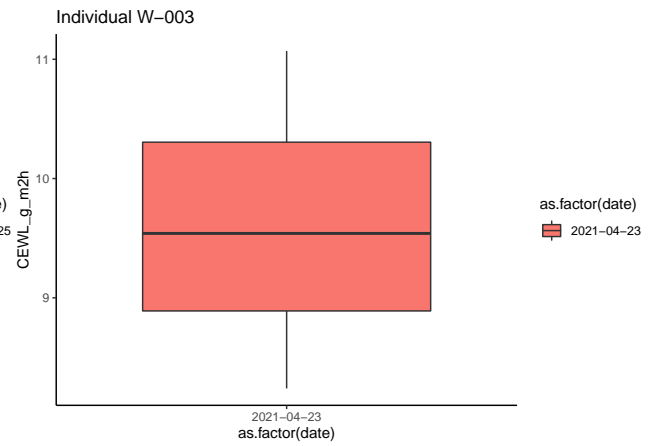
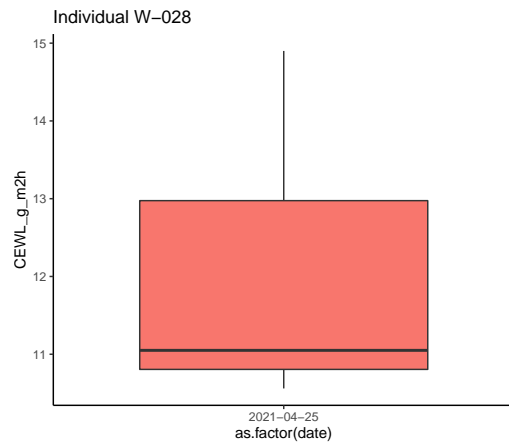


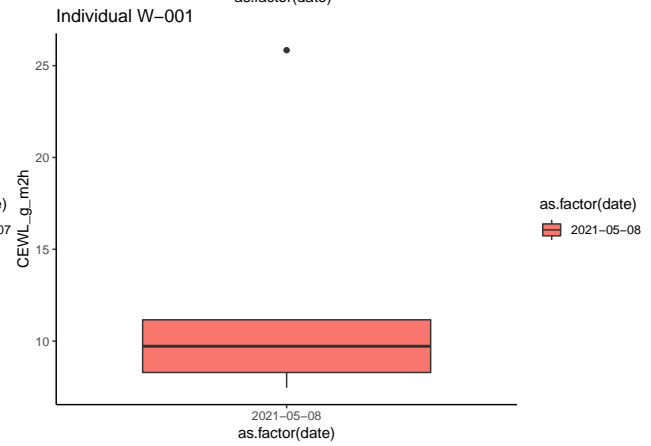
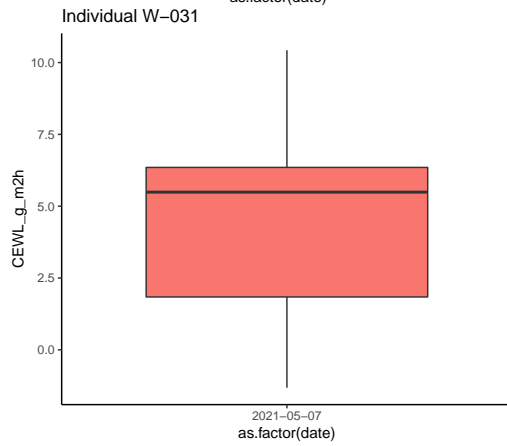
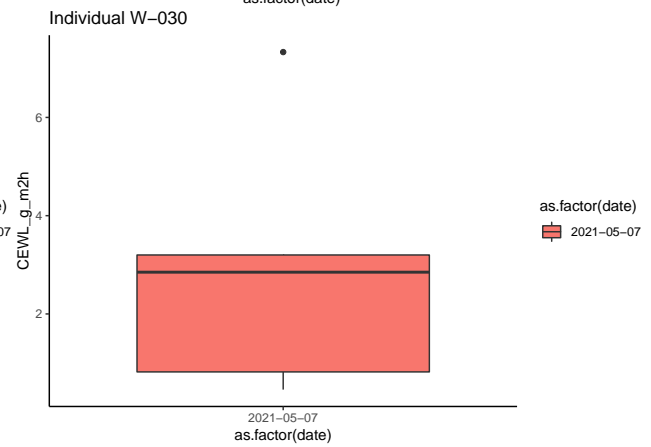
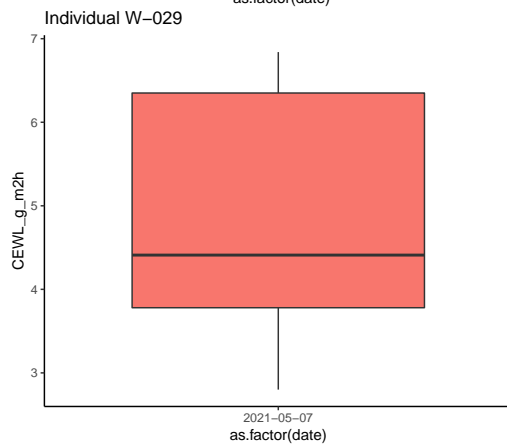
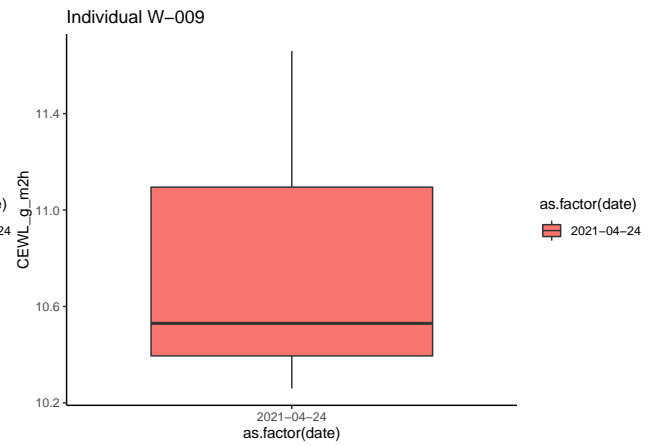
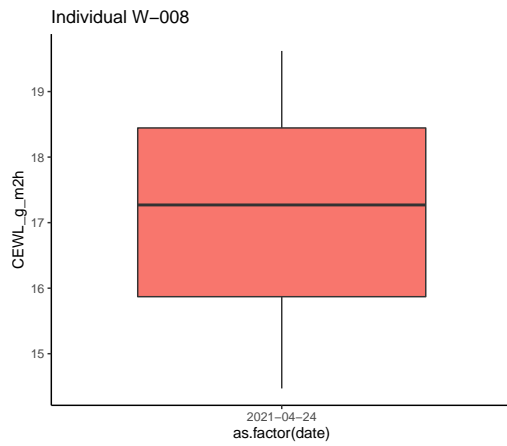


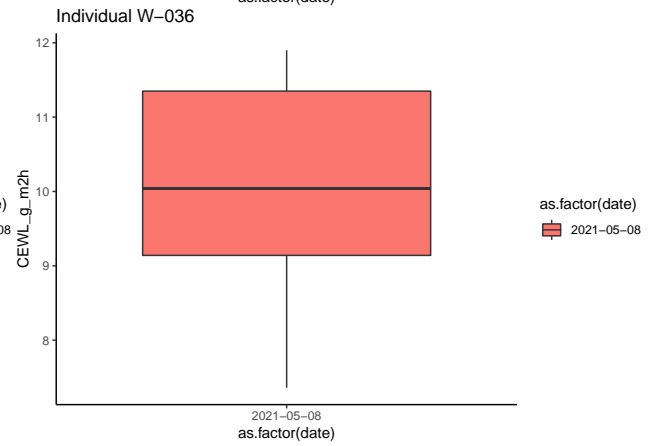
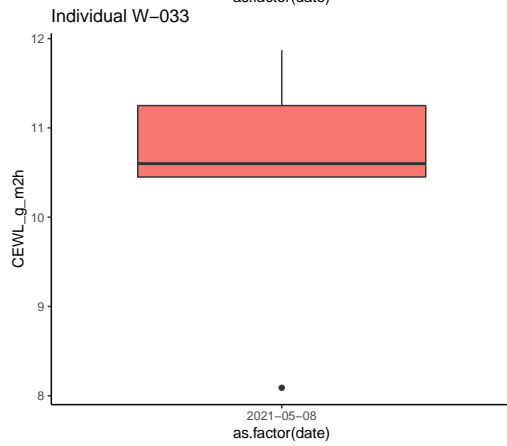
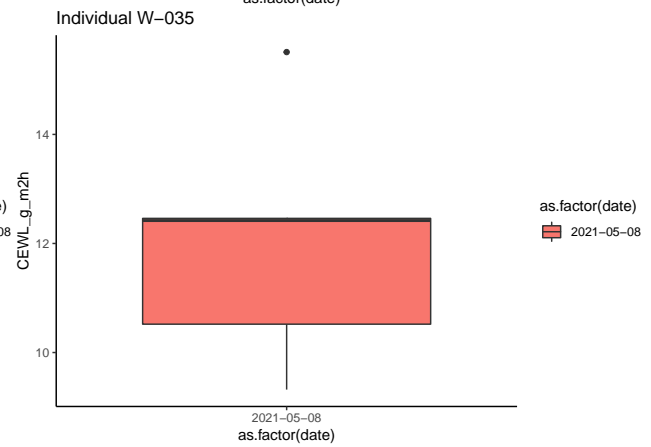
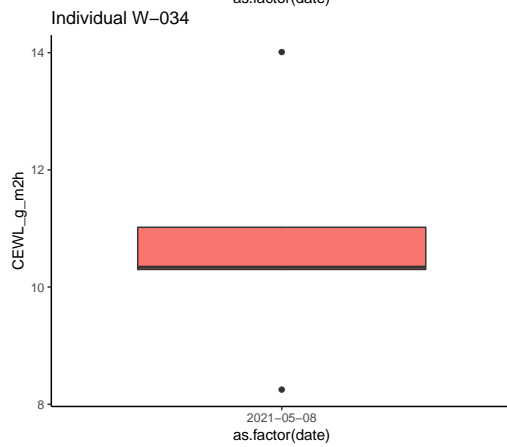
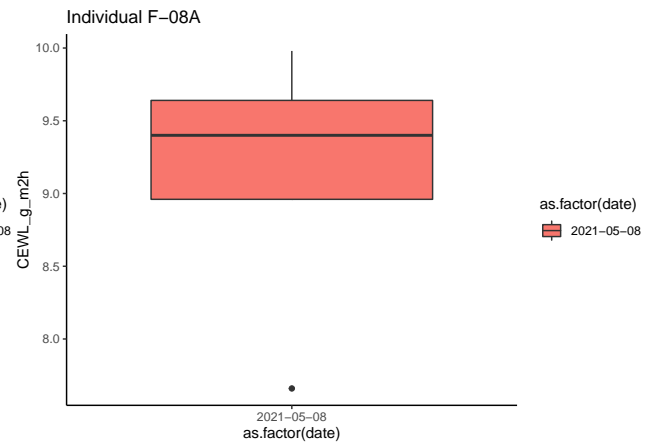
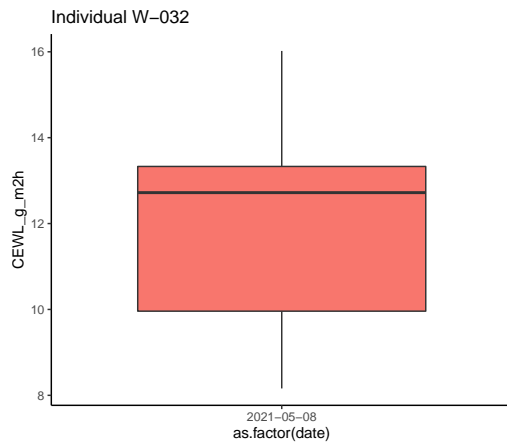


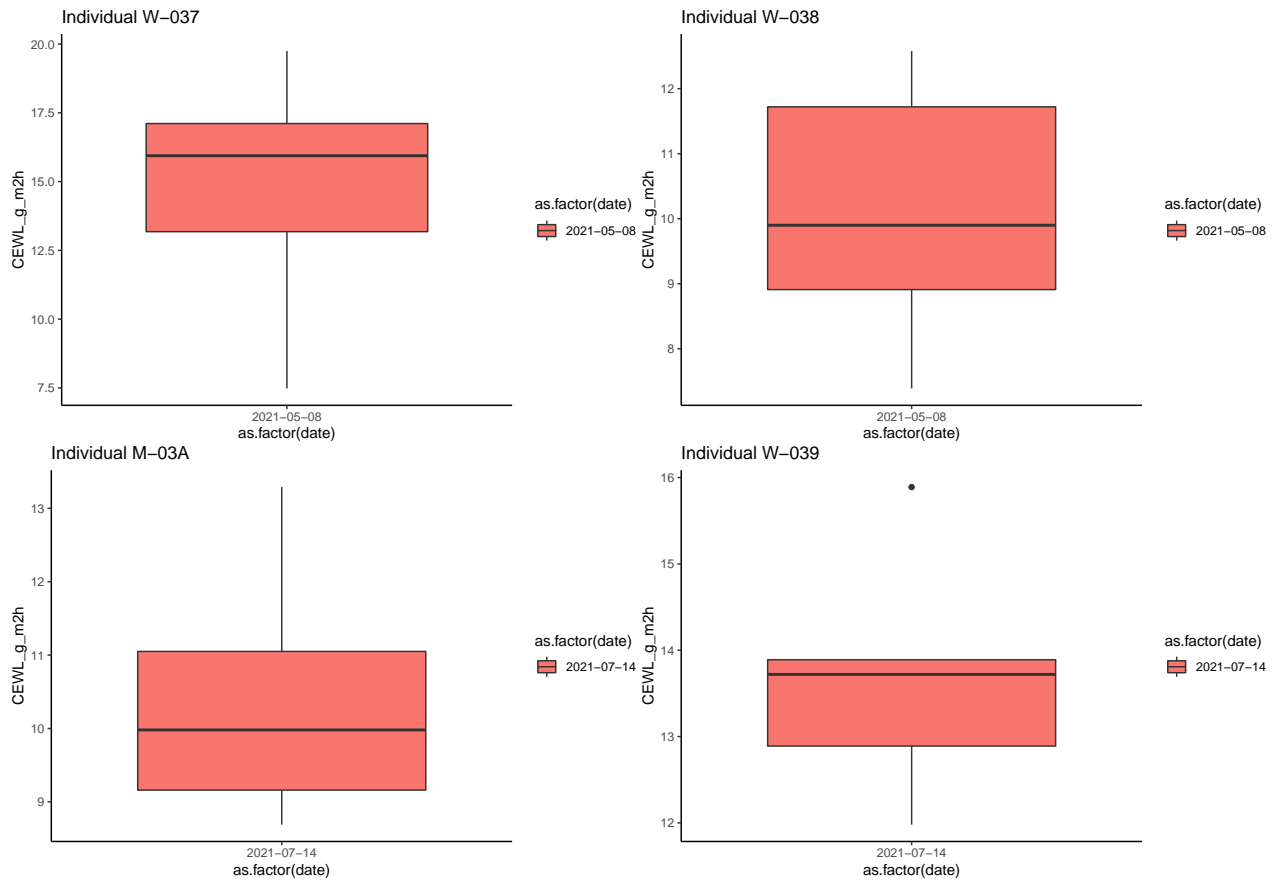












Based on the plots, the dataframe of outliers I compiled is correct. (yay!)

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)

```

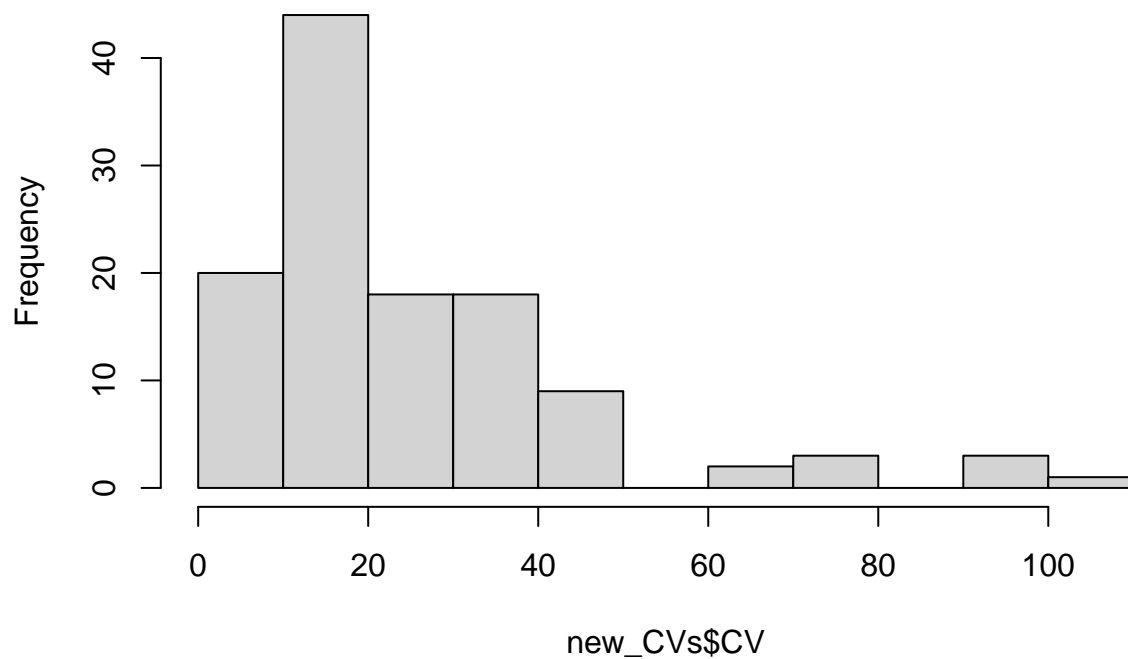
```
## `summarise()` regrouping output by 'individual_ID' (override with ` .groups ` argument)
```

```
summary(new_CVs)
```

```
## individual_ID      date      mean      SD
## F-12   : 3   Min.   :2021-04-23   Min.   : 0.650   Min.   : 0.05508
## M-09   : 3   1st Qu.:2021-04-24   1st Qu.: 8.486   1st Qu.: 1.21719
## M-10   : 3   Median :2021-04-24   Median :10.421   Median : 1.85776
## M-11   : 3   Mean    :2021-05-08   Mean    :10.682   Mean    : 2.65196
## M-19   : 3   3rd Qu.:2021-05-08   3rd Qu.:13.239   3rd Qu.: 2.88268
## M-20   : 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
##
```

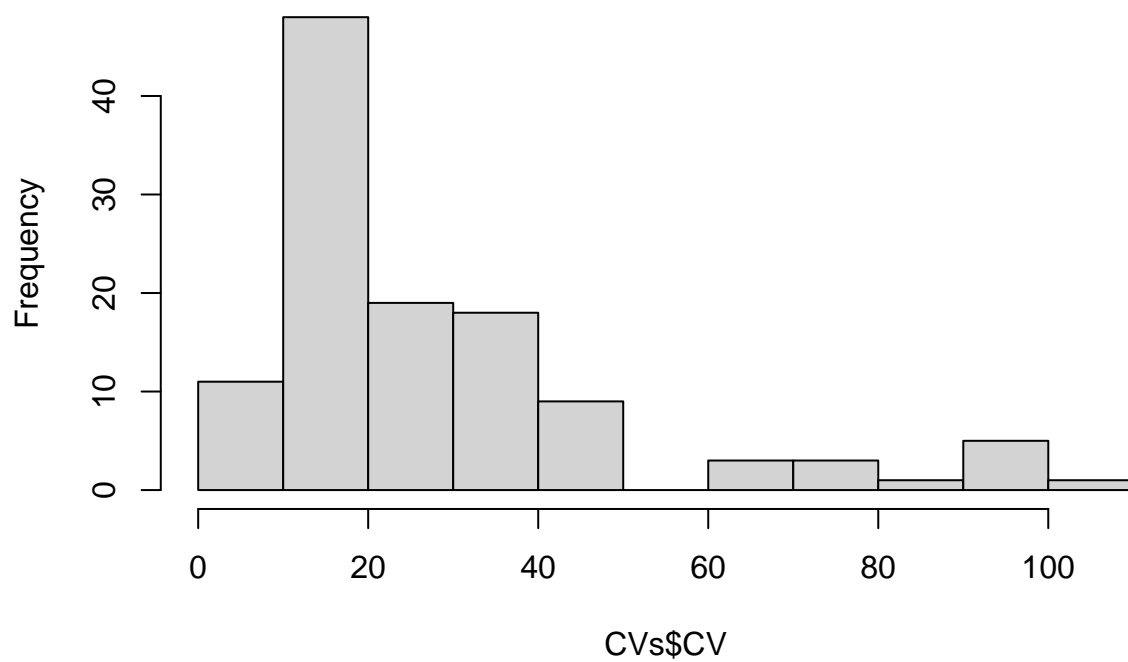
```
hist(new_CVs$CV)
```

Histogram of new_CVs\$CV



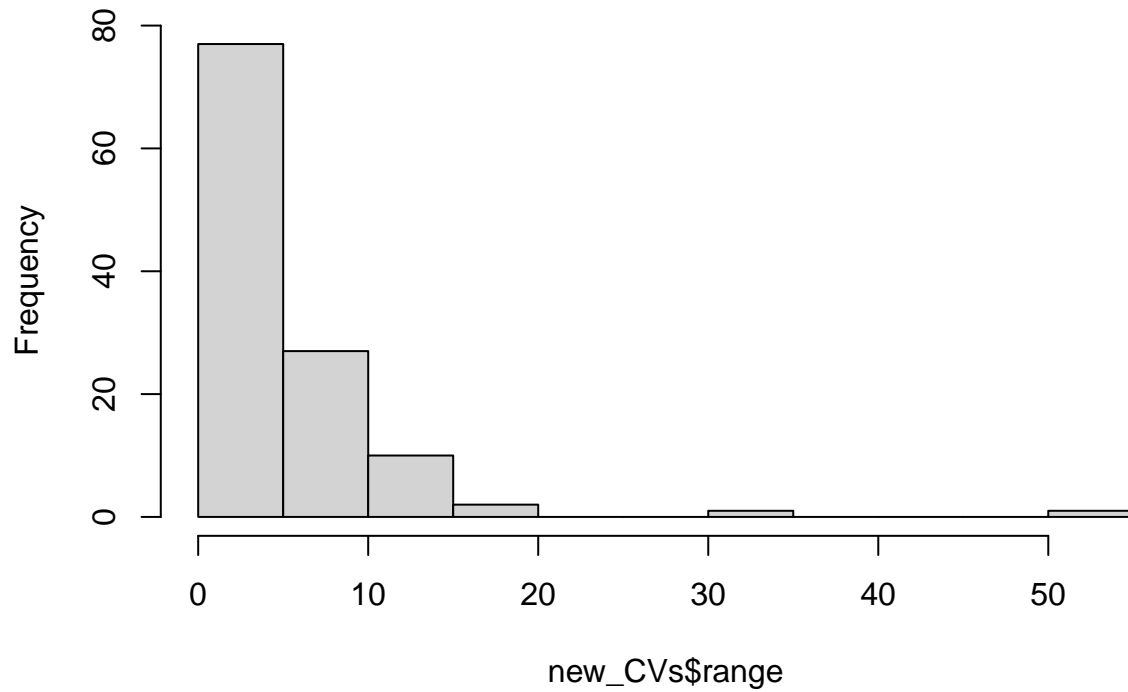
```
hist(CVs$CV)
```

Histogram of CVs\$CV



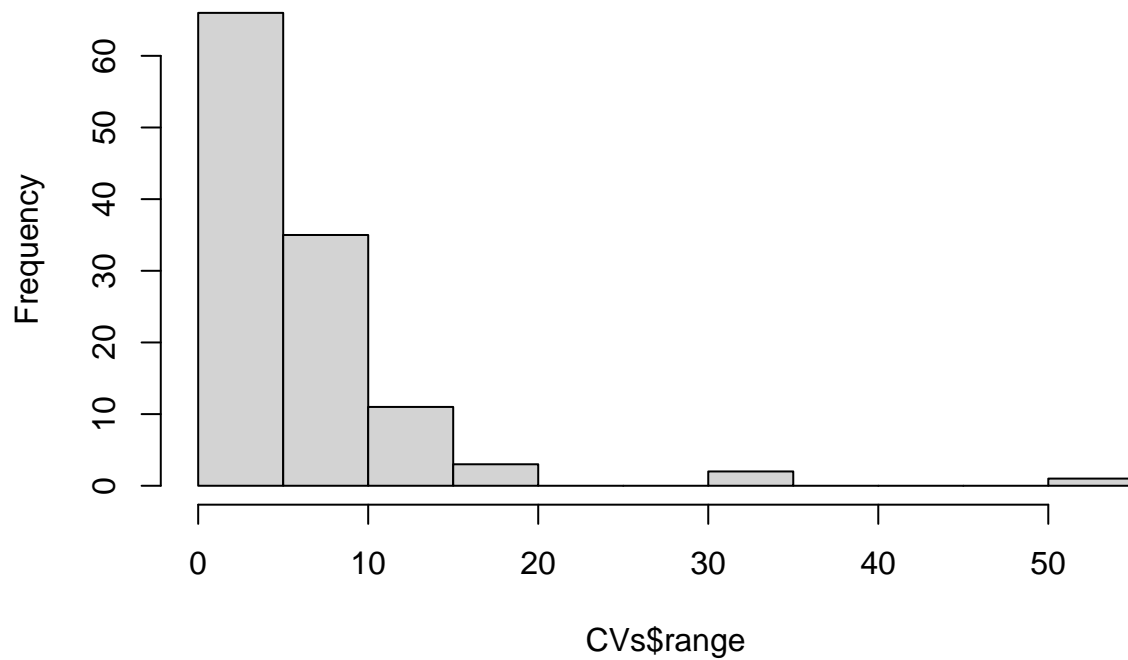
```
hist(new_CVs$range)
```


Histogram of new_CVs\$range



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

Histogram of CVs\$range



This definitely improved things, but unfortunately, CVs are still skewed to the right. I think the replicate groups with only 3 replicates are harder to find outliers in, so let's compute pairwise CVs and see if we can still minimize the larger range/CV values.

Find “Outliers”

Determine which replicates lead to an increased CV...

```
# which individuals have how many reps
n_reps <- outliers_omitted %>%
  group_by(individual_ID, date) %>%
  summarise(n = (n()))

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

# prep dfs for computing CVs
oo_small_345 <- outliers_omitted %>%
  dplyr::select(individual_ID, date, CEWL_g_m2h, replicate_no)

oo_small_45 <- oo_small_345 %>%
  left_join(n_reps, by = c('individual_ID', 'date')) %>%
  dplyr::filter(n %in% c(4, 5))

oo_small_5 <- oo_small_45 %>%
  dplyr::filter(n == 5)

# test excluding different replicates
CV_excl1 <- oo_small_345 %>%
  dplyr::filter(replicate_no != 1) %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100) %>%
  mutate(rep_excluded = "1")

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

CV_excl2 <- oo_small_345 %>%
  dplyr::filter(replicate_no != 2) %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100) %>%
  mutate(rep_excluded = "2")

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

CV_excl3 <- oo_small_345 %>%
  dplyr::filter(replicate_no != 3) %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100) %>%
  mutate(rep_excluded = "3")

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

CV_excl4 <- oo_small_45 %>%
  dplyr::filter(replicate_no != 4) %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
```

```

      CV = (SD/mean) *100) %>%
mutate(rep_excluded = "4")

## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
CV_excl5 <- oo_small_5 %>%
  dplyr::filter(replicate_no != 5) %>%
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100) %>%
  mutate(rep_excluded = "5")

## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
# figure out what replicate inflates CV (and range)
compare <- oo_small_345 %>%
  # first compute CV again with ALL replicates
  group_by(individual_ID, date) %>%
  summarise(mean = mean(CEWL_g_m2h),
            SD = sd(CEWL_g_m2h),
            CV = (SD/mean) *100) %>%
  mutate(rep_excluded = "none") %>%
  # attach other CVs with sub-setted rep numbers
  rbind(CV_excl1) %>%
  rbind(CV_excl2) %>%
  rbind(CV_excl3) %>%
  rbind(CV_excl4) %>%
  rbind(CV_excl5) %>%
  # compare which group of reps gives the lowest CV for each individual
  group_by(individual_ID, date) %>%
  dplyr::mutate(none_CV = case_when(rep_excluded == "none" ~ CV,
                                   rep_excluded != "none" ~ NA_real_),
               min_CV = min(CV),
               none_vs_min = none_CV - min_CV,
               rep_excluded = as.factor(rep_excluded))

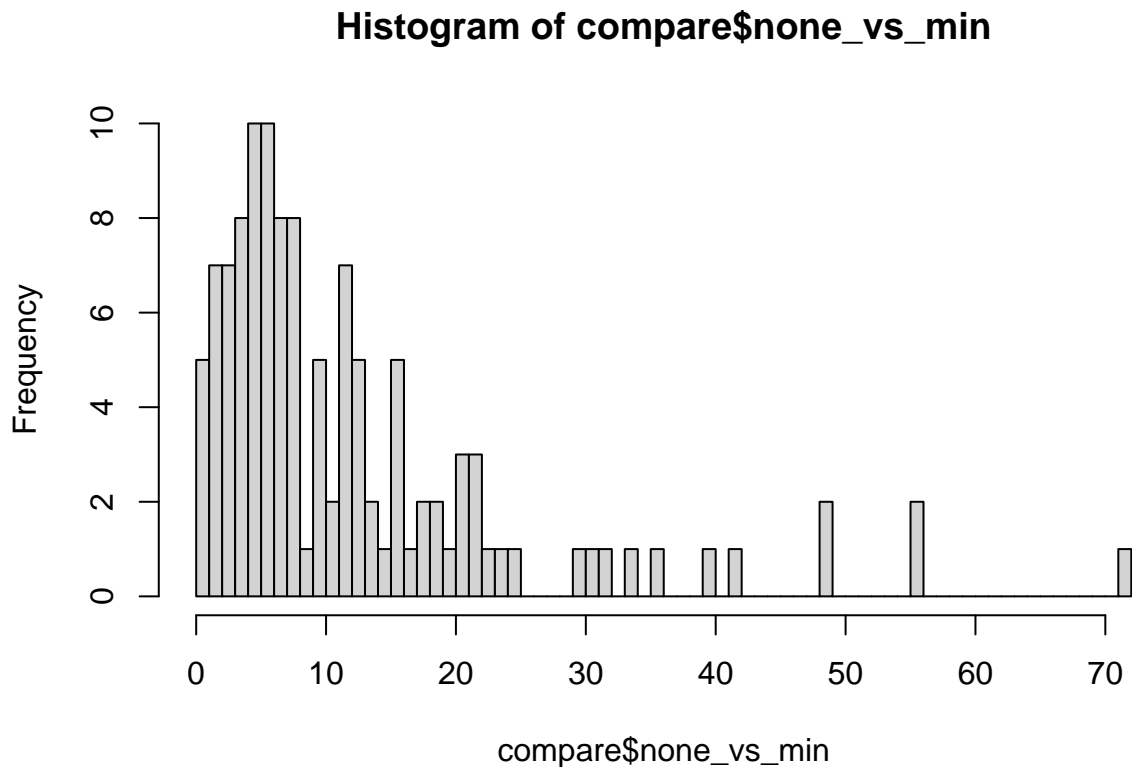
## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
summary(compare)

```

##	individual_ID	date	mean	SD
##	F-12 : 16	Min. :2021-04-23	Min. : 0.415	Min. : 0.00707
##	M-09 : 15	1st Qu.:2021-04-24	1st Qu.: 8.385	1st Qu.: 1.00698
##	M-19 : 15	Median :2021-05-07	Median :10.396	Median : 1.79953
##	M-10 : 14	Mean :2021-05-10	Mean :10.527	Mean : 2.42046
##	M-11 : 14	3rd Qu.:2021-05-08	3rd Qu.:12.891	3rd Qu.: 2.69628
##	M-20 : 14	Max. :2021-07-14	Max. :38.860	Max. :37.40595
##	(Other):462			
##	CV	rep_excluded	none_CV	min_CV
##	Min. : 0.04985	1 :118	Min. : 1.032	Min. : 0.04985
##	1st Qu.: 11.40235	2 :118	1st Qu.: 13.432	1st Qu.: 4.64184
##	Median : 19.11119	3 :118	Median : 19.265	Median :11.66153
##	Mean : 24.07846	none:118	Mean : 25.543	Mean :14.13556
##	3rd Qu.: 31.21640	4 : 46	3rd Qu.: 33.435	3rd Qu.:19.11642
##	Max. :122.85266	5 : 32	Max. :105.713	Max. :68.71924

```
##                                     NA's      :432
##   none_vs_min
##   Min.      : 0.000
##   1st Qu.:  4.295
##   Median :  7.624
##   Mean  :12.132
##   3rd Qu.:15.433
##   Max.    :71.159
##   NA's     :432
```

```
hist(compare$none_vs_min, breaks = 100)
```



Remove Outliers

```
# take the "none" removed rep avg when none_vs_min <5
no_cleaning_needed <- compare %>%
  dplyr::filter(none_vs_min <= 5) %>%
  dplyr::select(individual_ID, date, rep_excluded)

# get rest
needs_cleaning_IDs <- compare %>%
  dplyr::filter(none_vs_min > 5) %>%
  dplyr::mutate(keep = "KEEP") %>%
  dplyr::select(individual_ID, date, keep)

# check number of data obs
test_n <- outliers_omitted %>%
  group_by(individual_ID, date) %>%
  summarise(n = n())
```

```

## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
nrow(test_n) == (nrow(no_cleaning_needed) + nrow(needs_cleaning_IDs))

## [1] TRUE

# designate what to remove / not
how_cleaning <- compare %>%
  left_join(needs_cleaning_IDs) %>%
  dplyr::filter(keep == "KEEP") %>%
  dplyr::filter(CV == min_CV) %>%
  dplyr::select(individual_ID, date, rep_excluded) %>%
  rbind(no_cleaning_needed)

## Joining, by = c("individual_ID", "date")

# check
nrow(test_n) == (nrow(how_cleaning))

## [1] TRUE

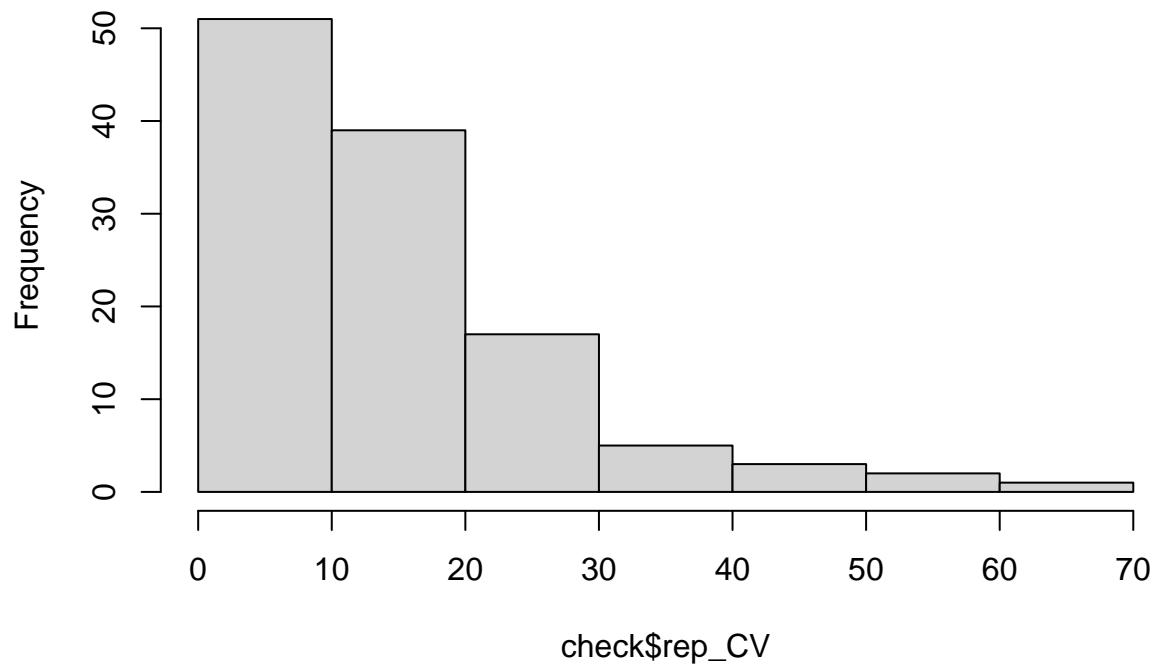
# remove!! :D
cleaned_using_CVs <- outliers_omitted %>%
  left_join(how_cleaning, by = c("individual_ID", "date")) %>%
  dplyr::filter(replicate_no != as.character(rep_excluded))

# check that we improved things
check <- cleaned_using_CVs %>%
  group_by(individual_ID, date) %>%
  summarise(CEWL_g_m2h_mean = mean(CEWL_g_m2h),
            rep_SD = sd(CEWL_g_m2h),
            rep_CV = (rep_SD/CEWL_g_m2h_mean) *100,
            rep_min = min(CEWL_g_m2h),
            rep_max = max(CEWL_g_m2h),
            rep_range = rep_max - rep_min
  )

## `summarise()` regrouping output by 'individual_ID' (override with `.groups` argument)
hist(check$rep_CV)

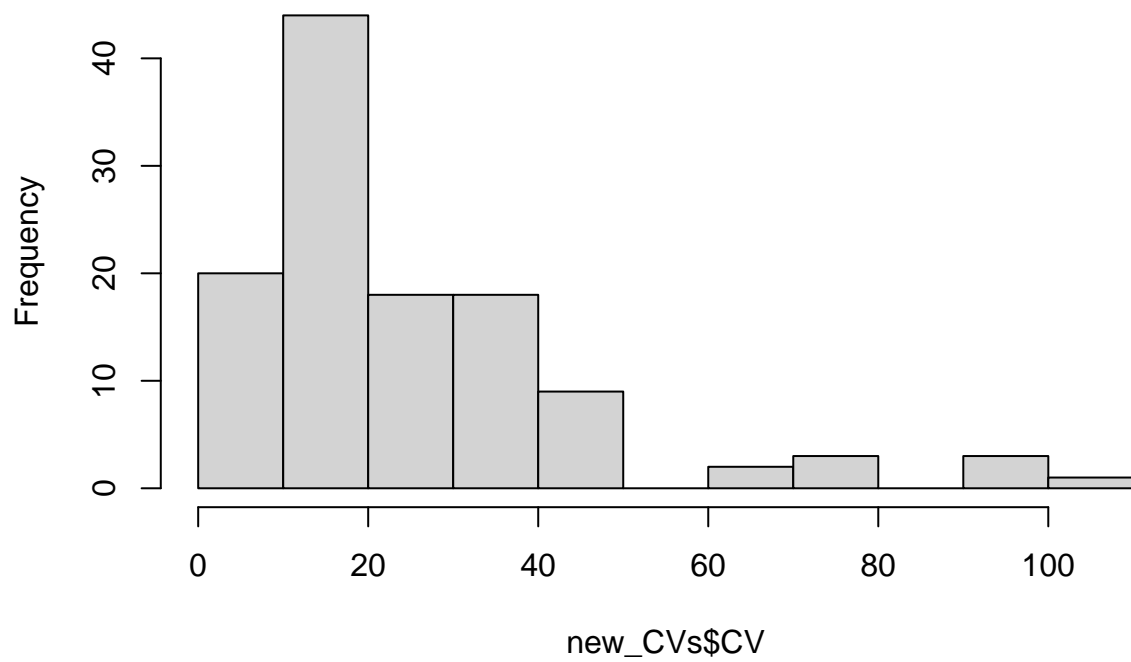
```

Histogram of check\$rep_CV



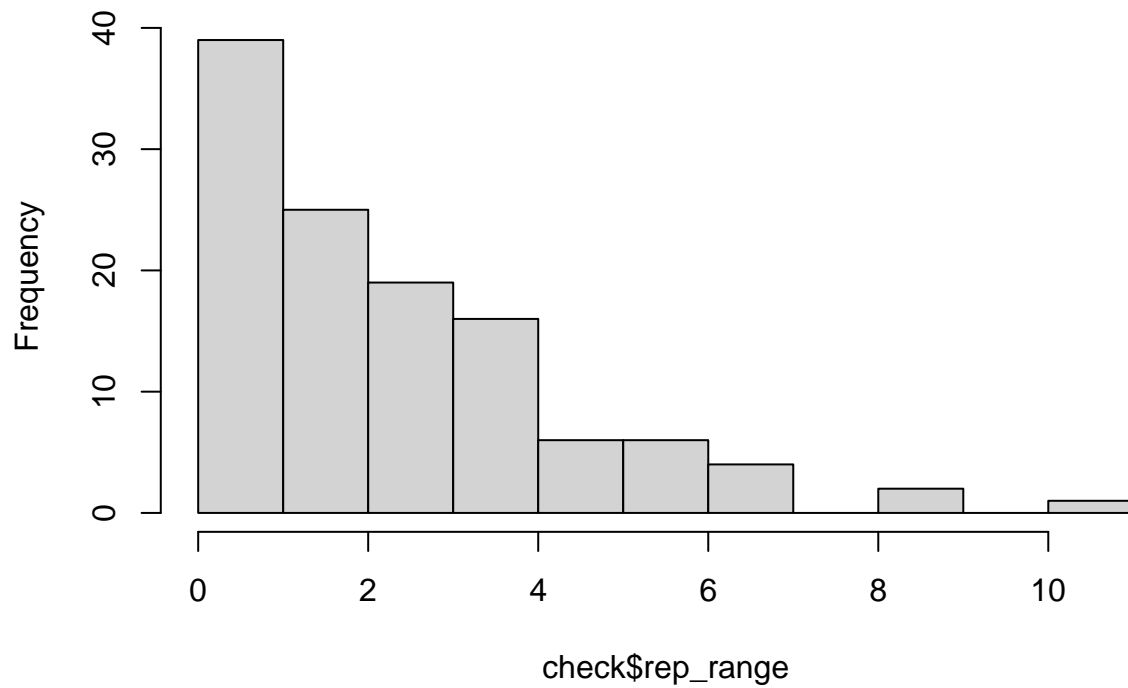
```
hist(new_CVs$CV)
```

Histogram of new_CVs\$CV



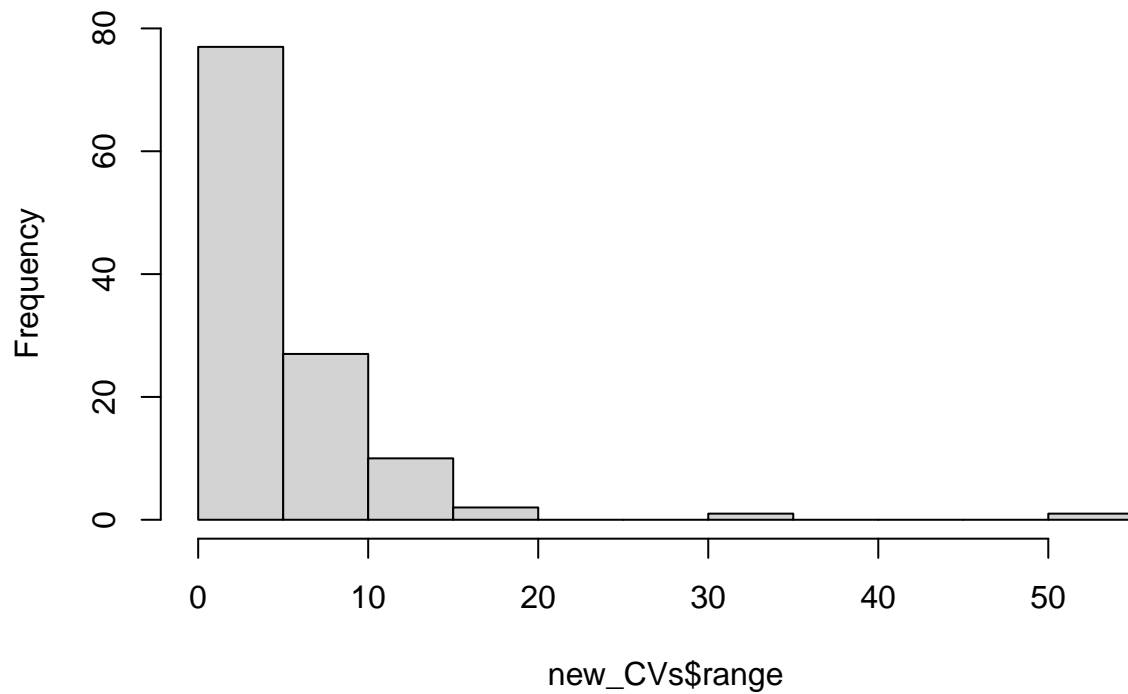
```
hist(check$rep_range)
```

Histogram of check\$rep_range



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

Histogram of new_CVs\$range



Yes, we decreased the CVs and ranges even more, so I think worth the data scrubbing.

Average Replicates (outliers removed)

```
CEWL_final <- cleaned_using_CVs %>%
  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 F-01           8.18        31.6           11.8
## 2 2021-04-23 F-02          15.4        33.6           16.8
## 3 2021-04-23 F-03           8.40        32.0           14.2
## 4 2021-04-23 F-04           9.21        25.0           26.0
## 5 2021-04-23 F-11          10.6        32.3           13.8
## 6 2021-04-23 F-12           0.59        32.3           13.5
```

Final Synthesis

Re-Check Data

Check that we still have data for every individual.

I can check this by comparing original individual IDs to the individual IDs in our final dataset, then selecting/printing the IDs used that are in one but not the other.

```
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:

```
cleaned_using_CVs %>%
  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 F-01          2021-04-23         2
## 2 F-02          2021-04-23         2
## 3 F-05          2021-04-24         2
## 4 F-06          2021-04-24         2
## 5 F-07          2021-04-24         2
## 6 F-08          2021-04-24         2
## 7 F-09          2021-04-24         2
## 8 F-10          2021-04-24         2
## 9 F-11          2021-04-23         2
## 10 F-12         2021-04-23         2
## # ... with 108 more rows
```

Between 2-5. So, We were able to delete a point if it was off from the other two in a group of 3.

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 105 measurements from our CEWL dataset (465 - 351): 1 replicate was removed for most individuals. We used the `boxplot.stats` function in R to extract outliers from each set of technical replicates, and 24 points were removed this way. The remaining 81 removed replicates were taken out because they increased the CV for their replicate group by >5% compared to the CV without that point. After data cleaning, every individual still had 2-5 technical replicates for each of their measurement dates. The distribution of coefficient of variation values was much better after both data cleaning steps than before.