ABER Data Analysis

Elise

2024-06-09

Set the right working directory.

```
setwd("C:/Users/elise/Documents/Mémoire/Main/Data/Templates/ABER")
```

Data importation

Import the data sets extracted from the Data Preparation R Markdown.

```
list.files()
```

```
## [1] "endpoint.txt"   "plant_info.txt"
```

```
plant_info <- read.table("plant_info.txt", header = TRUE, sep = "\t")
endpoint <- read.table("endpoint.txt", header = TRUE, sep = "\t")
```

Convert the columns to factor and date formats.

```
# plant_info
plant_info <- lapply(plant_info, factor)

# endpoint
matching_cols <- intersect(names(endpoint), names(plant_info))
endpoint[, matching_cols] <- lapply(endpoint[, matching_cols], factor)
endpoint$Date <- date(endpoint$Date)
endpoint$Timestamp <- NA

# timeseries
# No data for ABER

# S_timeseries
# No data

# T_timeseries
# No data
```

Collect the variables of every data template and print the names of the variables. This serves as a double check.

```
platform <- "ABER"

# endpoint
df <- endpoint[,colSums(is.na(endpoint))<nrow(endpoint)]
genotype_index <- which(colnames(df) == "Genotype")
variables <- colnames(df[, c(3:(genotype_index - 1))]) # We remove the two first column
s that are "Unit.ID" and "Date"

# timeseries
# no data

# S_timeseries
# no data

# T_timeseries
# no data

print(paste(platform, ": The variables for endpoint are", paste(variables, collapse =
", "), sep = " "))
```

```
## [1] "ABER : The variables for endpoint are DW_shoot_g, FW_shoot_g, DW_root_g, FW_roo
t_g, Leaf_number"
```

Add a column Plant_type with three levels, H L and T. This variable is useful to test for heterosis effects.

```
endpoint$Plant_type <- substr(endpoint$Genotype, nchar(as.character(endpoint$Genotyp
e)), nchar(as.character(endpoint$Genotype)))
```

1. Endpoint dataframe

A. Exploration of data

Exploration tables using the rstatix, janitor and skimr packages

```
endpoint %>%
  count(Genotype)
```

```
##   Genotype n
## 1   EPPN_T 6
## 2  EPPN1_H 6
## 3  EPPN1_L 6
## 4  EPPN2_H 6
## 5  EPPN2_L 6
## 6  EPPN3_H 6
## 7  EPPN3_L 6
## 8  EPPN4_H 6
## 9  EPPN4_L 6
```

```
endpoint %>%
  tabyl(Genotype, Column) %>%
  adorn_totals("row") %>%
  adorn_percentages("row") %>%
  adorn_pct_formatting() %>%
  adorn_ns() %>%
  adorn_title("combined")
```

```
##  Genotype/Column         1         2         3         4         5         6
##           EPPN_T  0.0% (0)  0.0% (0)  0.0% (0) 16.7% (1) 16.7% (1)  0.0% (0)
##          EPPN1_H  0.0% (0) 33.3% (2) 16.7% (1) 16.7% (1) 16.7% (1)  0.0% (0)
##          EPPN1_L 33.3% (2) 16.7% (1)  0.0% (0)  0.0% (0)  0.0% (0)  0.0% (0)
##          EPPN2_H 16.7% (1) 33.3% (2)  0.0% (0)  0.0% (0) 33.3% (2) 16.7% (1)
##          EPPN2_L  0.0% (0) 16.7% (1)  0.0% (0)  0.0% (0)  0.0% (0)  0.0% (0)
##          EPPN3_H 16.7% (1)  0.0% (0)  0.0% (0) 16.7% (1) 16.7% (1)  0.0% (0)
##          EPPN3_L  0.0% (0)  0.0% (0) 33.3% (2)  0.0% (0)  0.0% (0) 50.0% (3)
##          EPPN4_H 16.7% (1)  0.0% (0) 33.3% (2)  0.0% (0)  0.0% (0) 16.7% (1)
##          EPPN4_L  0.0% (0)  0.0% (0)  0.0% (0) 33.3% (2) 16.7% (1)  0.0% (0)
##            Total  9.3% (5) 11.1% (6)  9.3% (5)  9.3% (5) 11.1% (6)  9.3% (5)
##          7         8         9        10
##   0.0% (0) 16.7% (1) 33.3% (2) 16.7% (1)
##  16.7% (1)  0.0% (0)  0.0% (0)  0.0% (0)
##  16.7% (1)  0.0% (0)  0.0% (0) 33.3% (2)
##   0.0% (0)  0.0% (0)  0.0% (0)  0.0% (0)
##  33.3% (2) 16.7% (1) 16.7% (1) 16.7% (1)
##   0.0% (0) 33.3% (2) 16.7% (1)  0.0% (0)
##   0.0% (0)  0.0% (0) 16.7% (1)  0.0% (0)
##   0.0% (0) 16.7% (1)  0.0% (0) 16.7% (1)
##  33.3% (2) 16.7% (1)  0.0% (0)  0.0% (0)
##  11.1% (6) 11.1% (6)  9.3% (5)  9.3% (5)
```

```
endpoint %>%
  tabyl(Genotype, Row) %>%
  adorn_totals("row") %>%
  adorn_percentages("row") %>%
  adorn_pct_formatting() %>%
  adorn_ns() %>%
  adorn_title("combined")
```

```
##  Genotype/Row         1         2          3         4         5         6
##        EPPN_T 33.3% (2)  0.0% (0)  0.0%  (0) 33.3% (2) 16.7% (1) 16.7% (1)
##       EPPN1_H 16.7% (1) 16.7% (1) 16.7%  (1)  0.0% (0) 33.3% (2) 16.7% (1)
##       EPPN1_L 33.3% (2) 16.7% (1) 16.7%  (1) 16.7% (1)  0.0% (0) 16.7% (1)
##       EPPN2_H 16.7% (1)  0.0% (0) 33.3%  (2) 16.7% (1) 33.3% (2)  0.0% (0)
##       EPPN2_L  0.0% (0) 33.3% (2) 16.7%  (1)  0.0% (0) 33.3% (2) 16.7% (1)
##       EPPN3_H 16.7% (1) 16.7% (1) 33.3%  (2) 16.7% (1)  0.0% (0) 16.7% (1)
##       EPPN3_L  0.0% (0) 16.7% (1) 16.7%  (1) 33.3% (2) 16.7% (1) 16.7% (1)
##       EPPN4_H 16.7% (1) 33.3% (2)  0.0%  (0) 16.7% (1) 16.7% (1) 16.7% (1)
##       EPPN4_L 16.7% (1) 16.7% (1) 33.3%  (2) 16.7% (1)  0.0% (0) 16.7% (1)
##         Total 16.7% (9) 16.7% (9) 18.5% (10) 16.7% (9) 16.7% (9) 14.8% (8)
```

```
endpoint %>%
  count(Genotype)
```

```
##   Genotype n
## 1   EPPN_T 6
## 2  EPPN1_H 6
## 3  EPPN1_L 6
## 4  EPPN2_H 6
## 5  EPPN2_L 6
## 6  EPPN3_H 6
## 7  EPPN3_L 6
## 8  EPPN4_H 6
## 9  EPPN4_L 6
```

```
get_summary_stats(data = endpoint,
                  variables,
                  type = "common")
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## ℹ Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(variables)
##
##   # Now:
##   data %>% select(all_of(variables))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## # A tibble: 5 × 10
##   variable        n   min   max median   iqr  mean     sd    se    ci
##   <fct>       <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1 DW_shoot_g     54   4.5  32.8   13.7  4.47  15.0  5.61  0.764 1.53
## 2 FW_shoot_g     54  45.6  98.8   73.4 12.6   73.8 10.4   1.42  2.84
## 3 DW_root_g      54   7    39.3   17.2  6.8   17.6  5.57  0.758 1.52
## 4 FW_root_g      54  28.5  77.1   43.4 11.8   44.3  9.15  1.25  2.50
## 5 Leaf_number    54   8    10      9    0      9    0.673 0.092 0.184
```

```
skim(endpoint[variables])
```

Data summary

Name	endpoint[variables]
Number of rows	54
Number of columns	5
_____	
Column type frequency:	
numeric	5
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
DW_shoot_g	0	1	14.99	5.61	4.5	12.10	13.70	16.58	32.8	
FW_shoot_g	0	1	73.79	10.42	45.6	67.88	73.45	80.47	98.8	
DW_root_g	0	1	17.59	5.57	7.0	14.27	17.25	21.08	39.3	
FW_root_g	0	1	44.28	9.15	28.5	38.27	43.40	50.05	77.1	
Leaf_number	0	1	9.00	0.67	8.0	9.00	9.00	9.00	10.0	

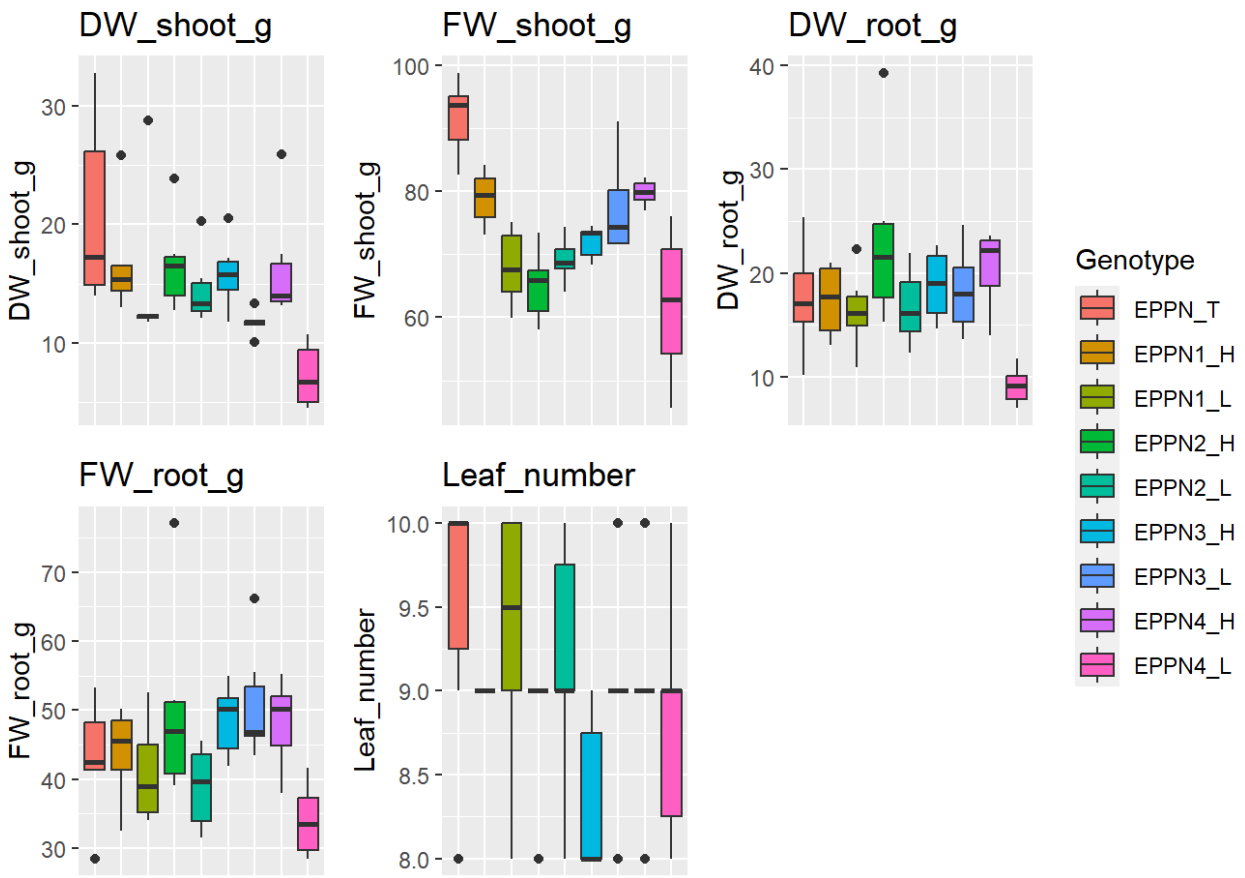
Data visualization

Using several functions that are located in the functions.R script

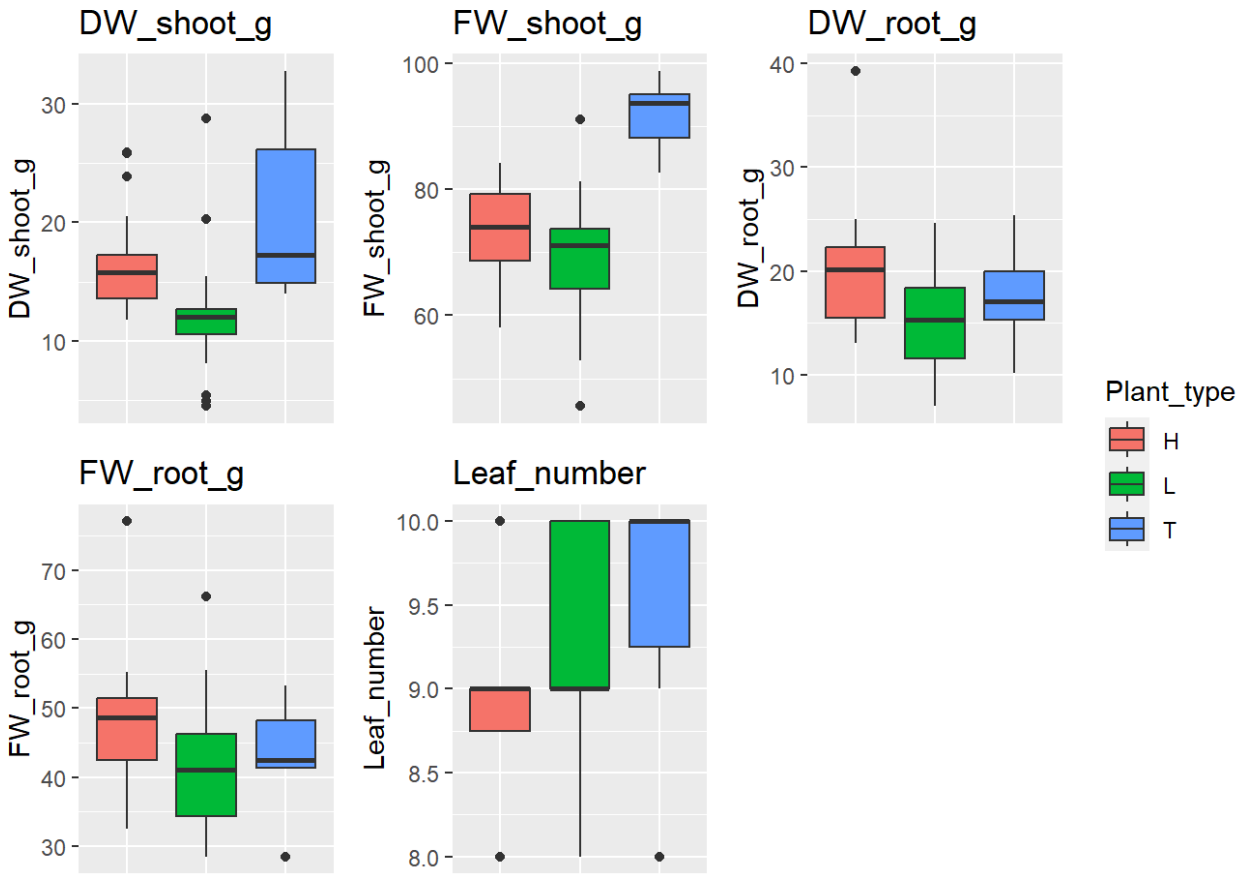
Boxplots

```
create_boxplots(endpoint, variables, "Genotype")
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## ℹ Please use tidy evaluation idioms with `aes()`.
## ℹ See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



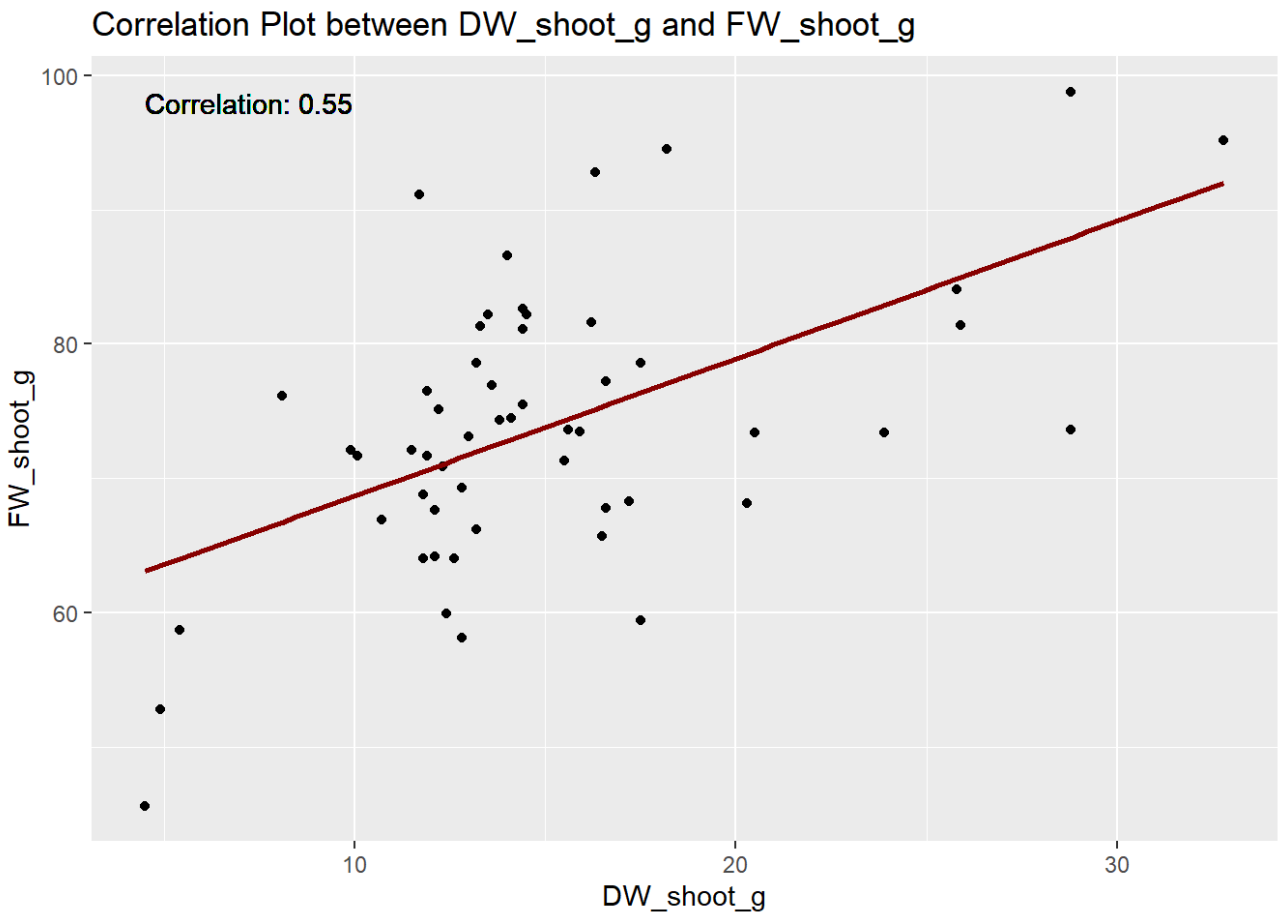

```
create_boxplots(endpoint, variables, "Plant_type")
```



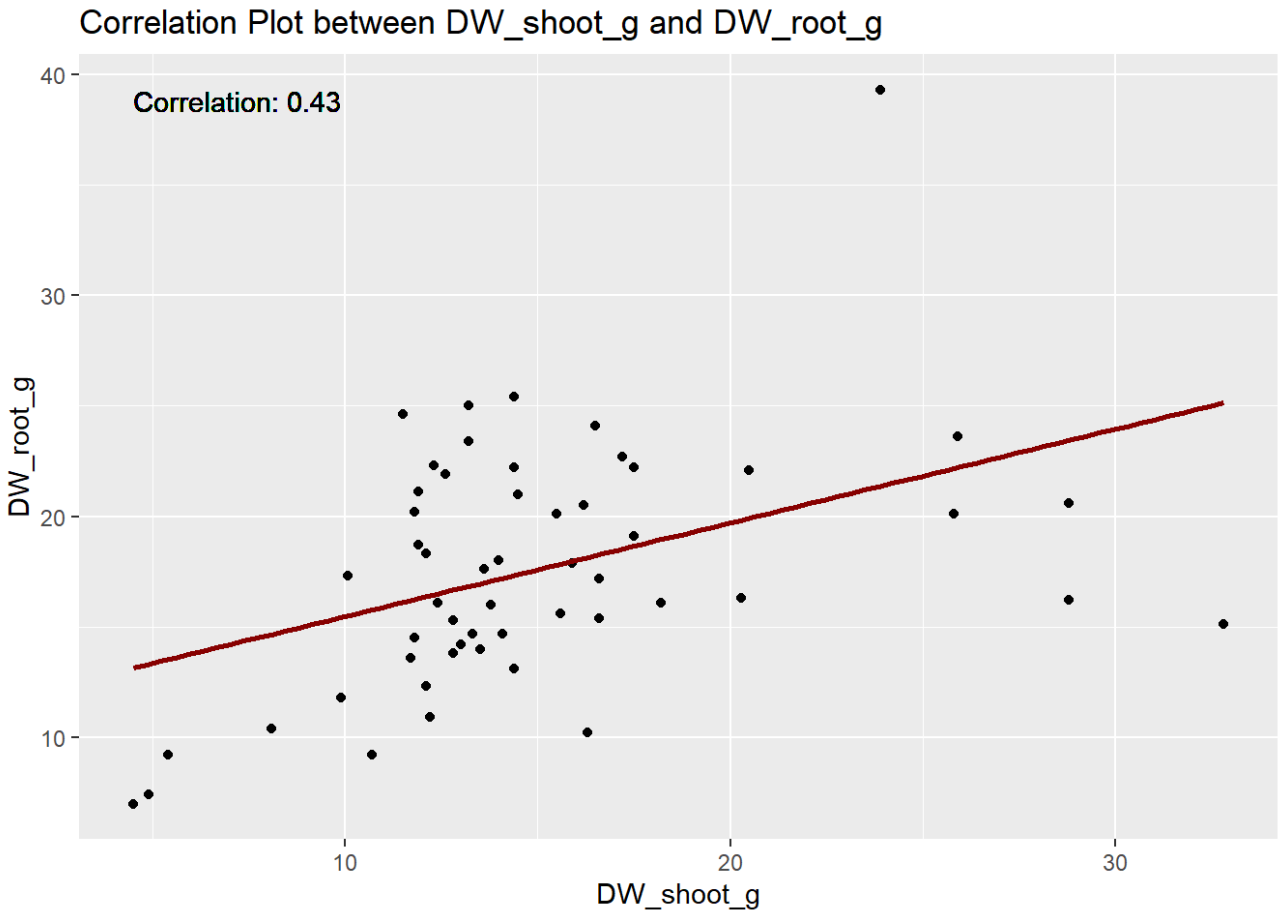
Correlation plots

```
for (i in 1:(length(variables) - 1)) {
  for (j in (i + 1):length(variables)) {
    calculate_correlation_plot(endpoint, variables[i], variables[j])
  }
}
```

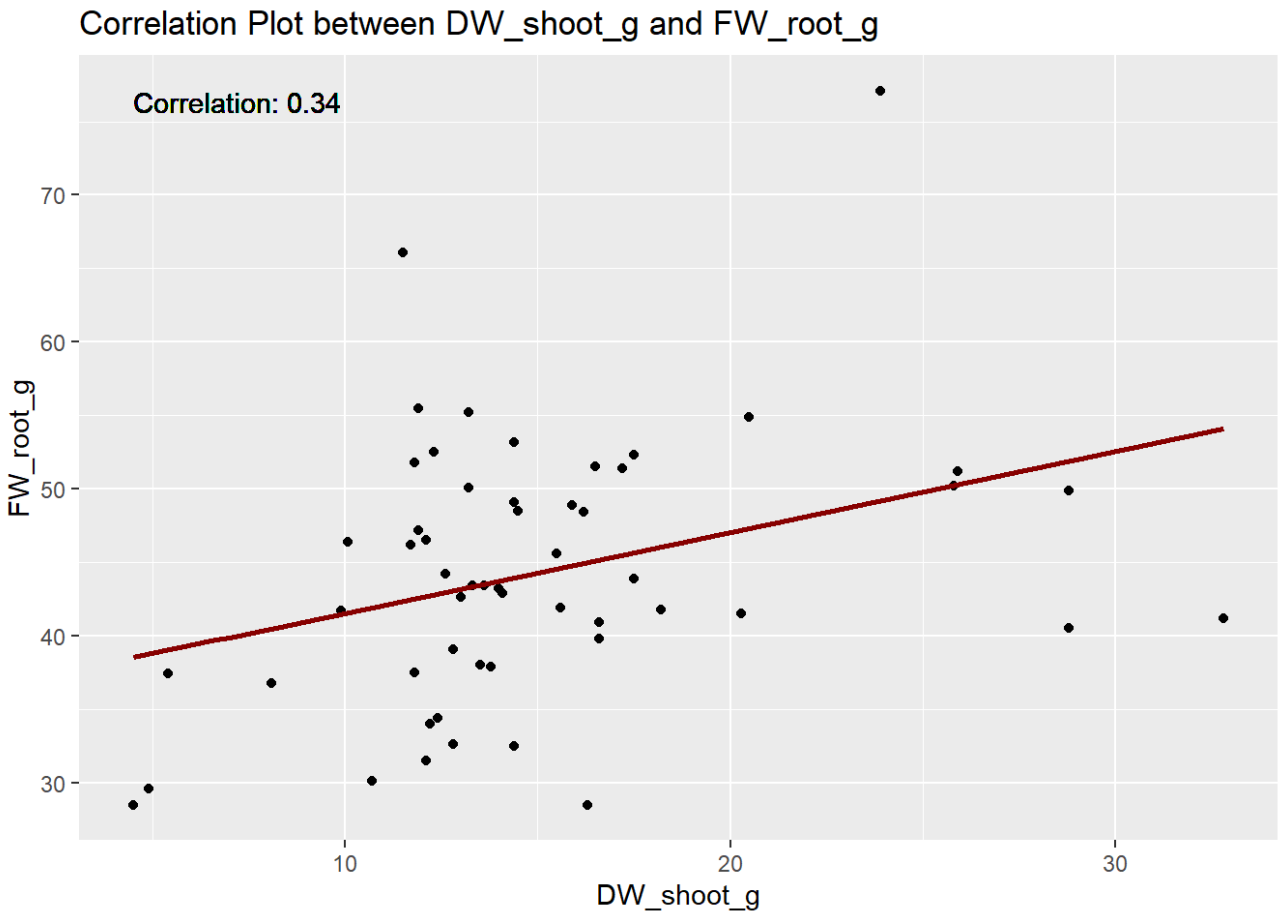
```
## `geom_smooth()` using formula = 'y ~ x'
```



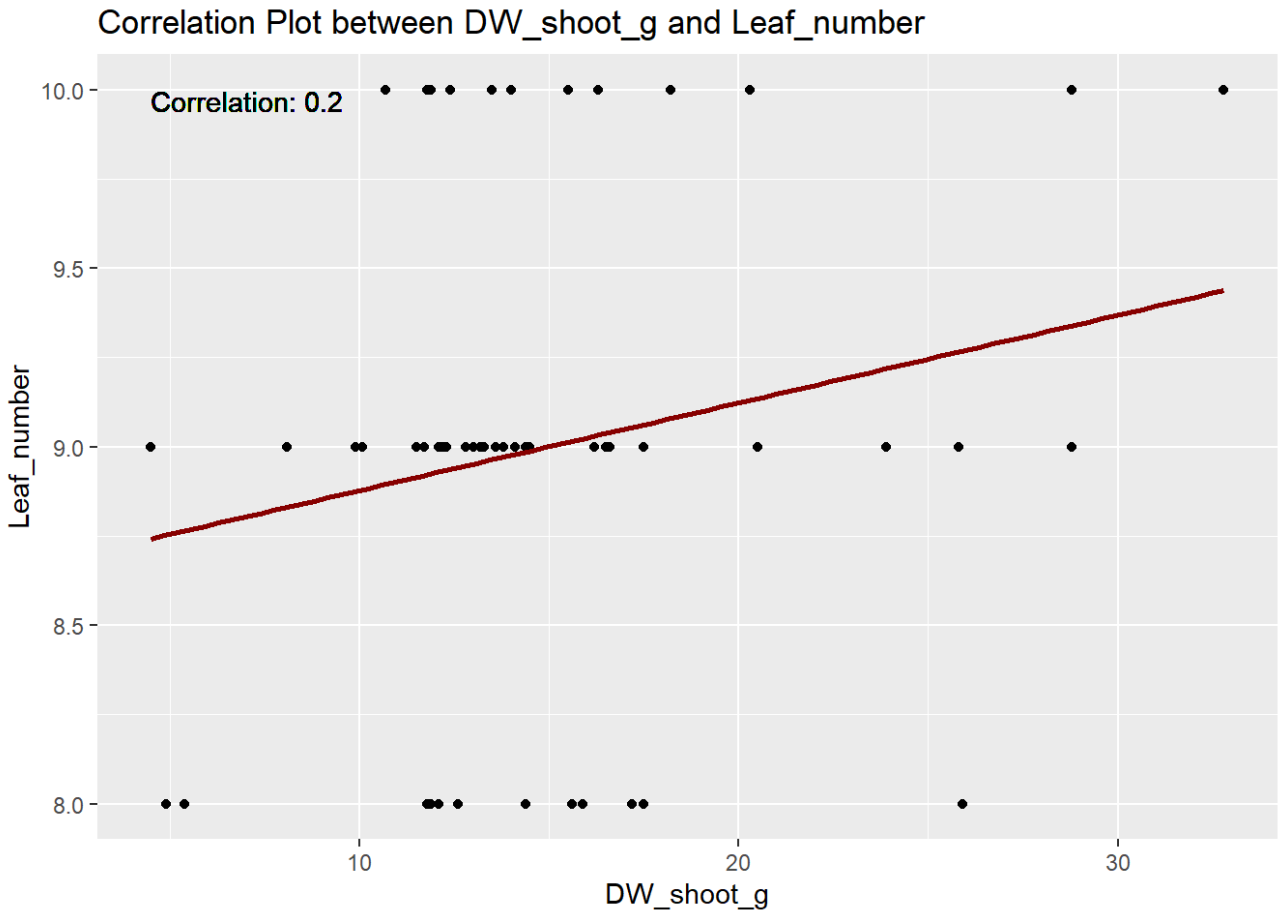

```
## `geom_smooth()` using formula = 'y ~ x'
```



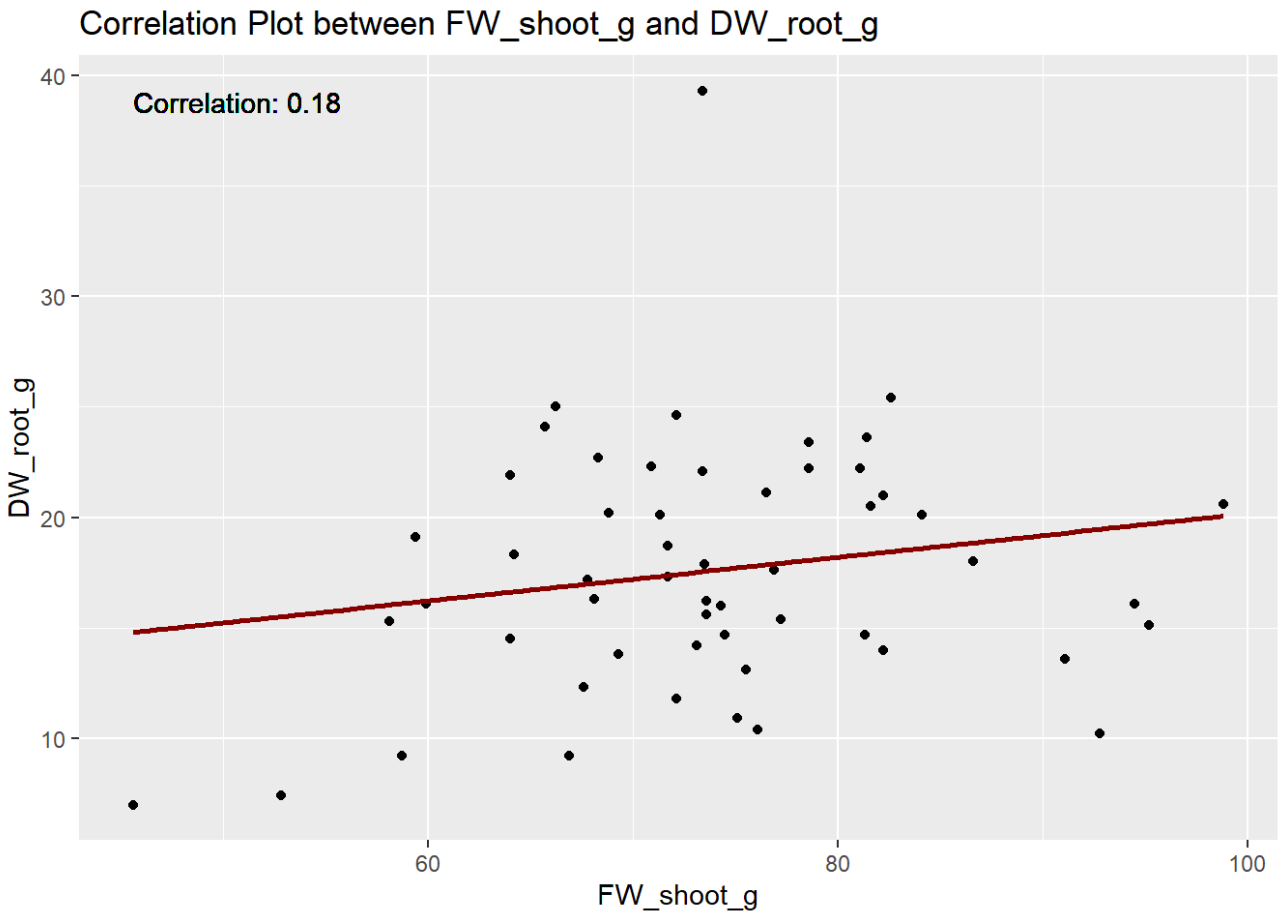

```
## `geom_smooth()` using formula = 'y ~ x'
```



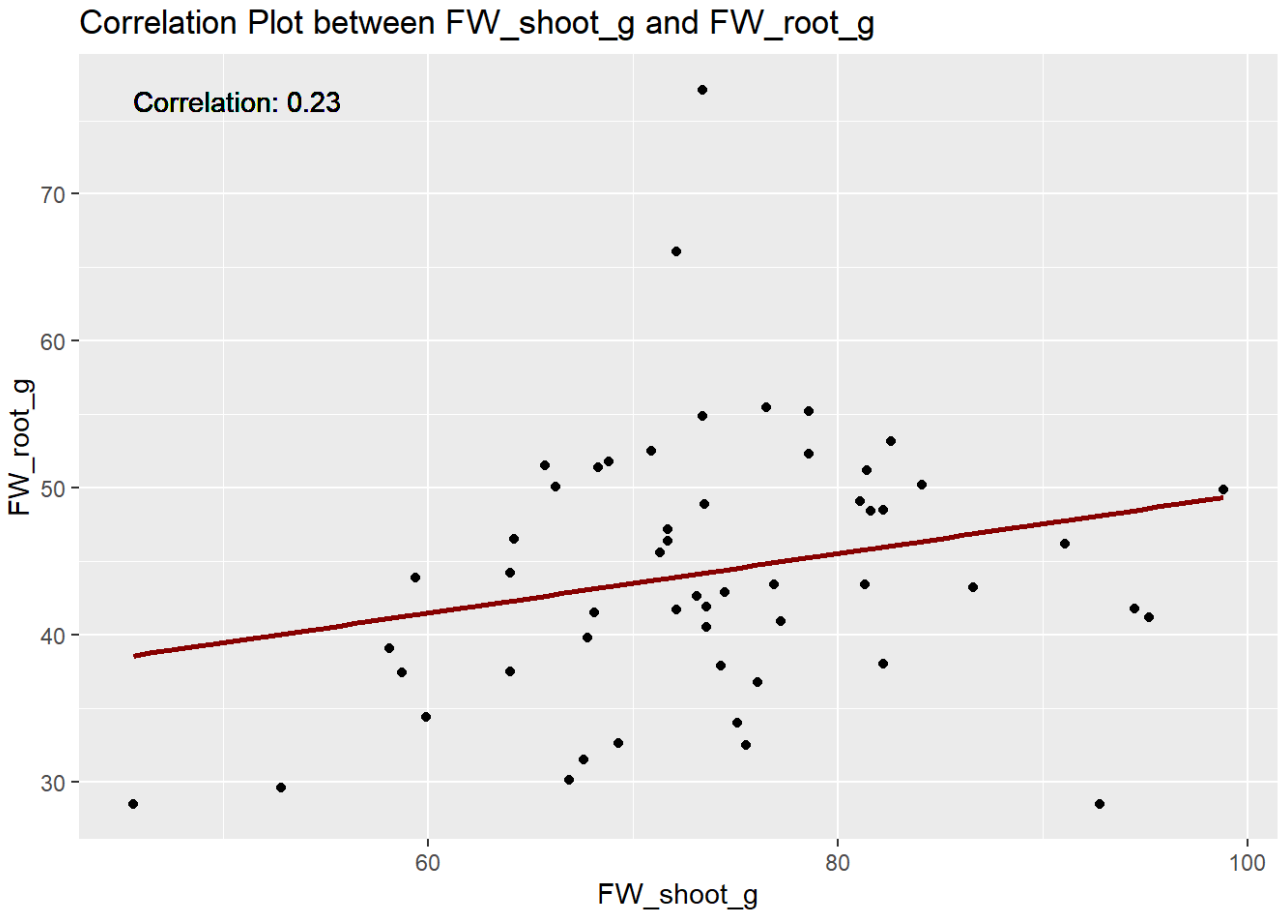

```
## `geom_smooth()` using formula = 'y ~ x'
```



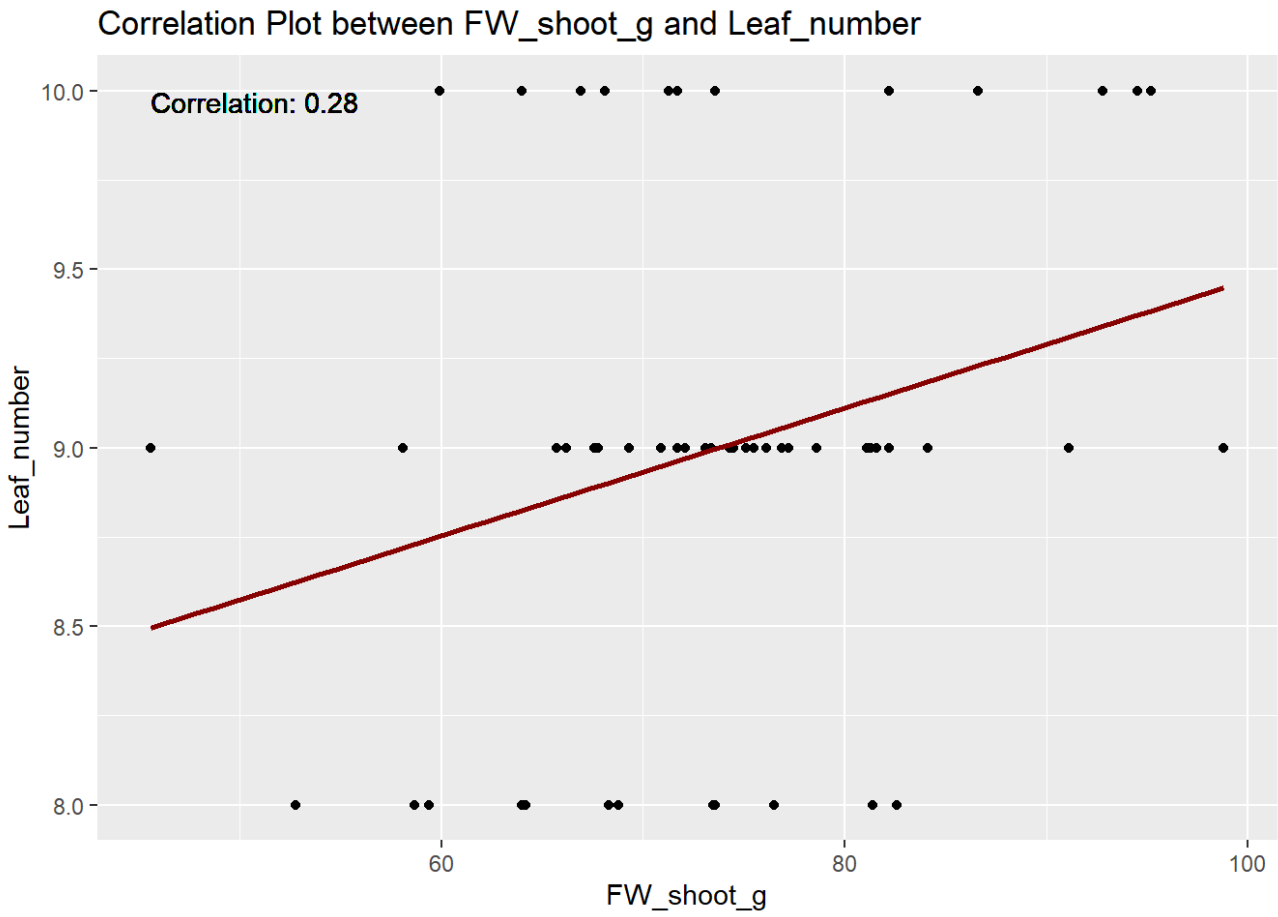

```
## `geom_smooth()` using formula = 'y ~ x'
```



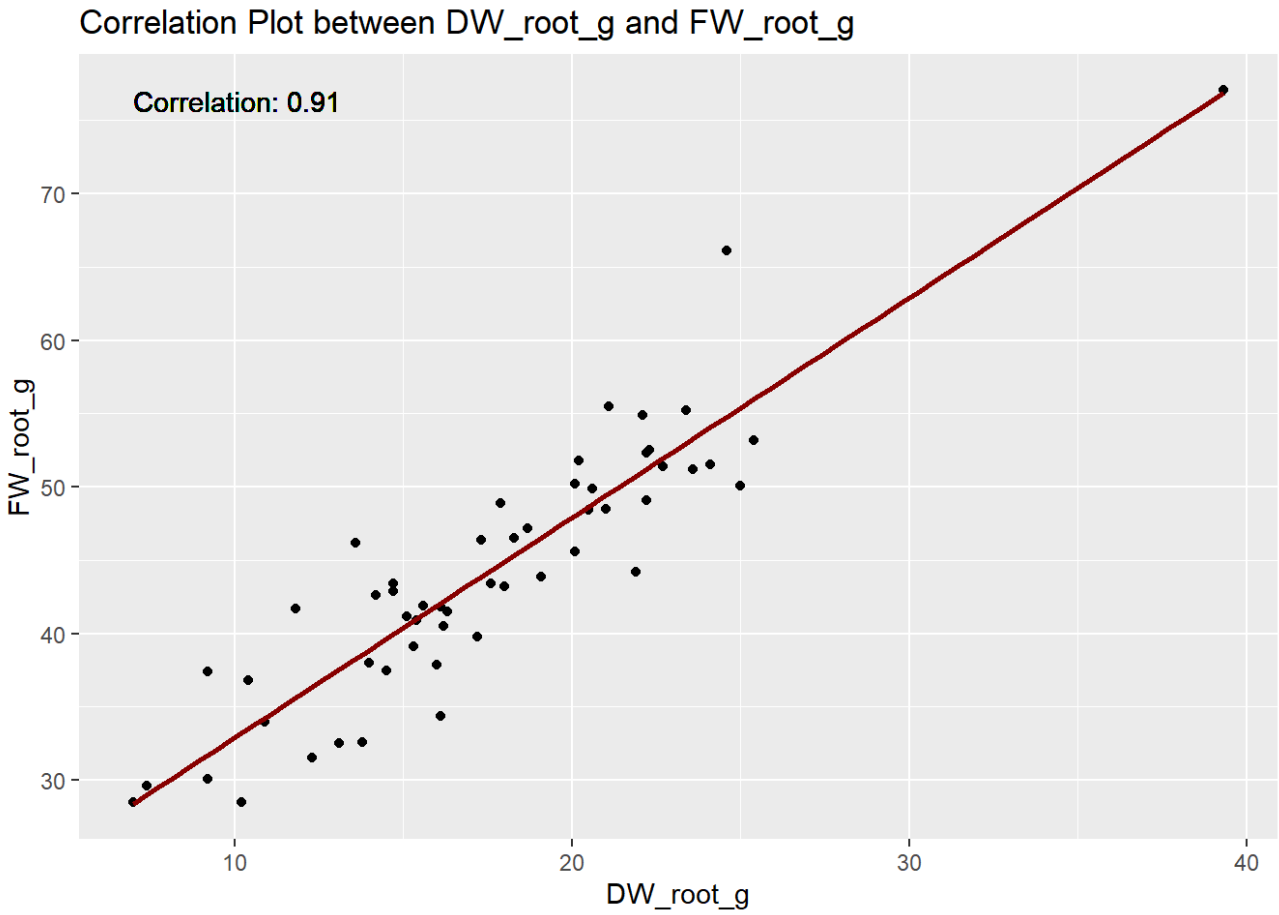

```
## `geom_smooth()` using formula = 'y ~ x'
```



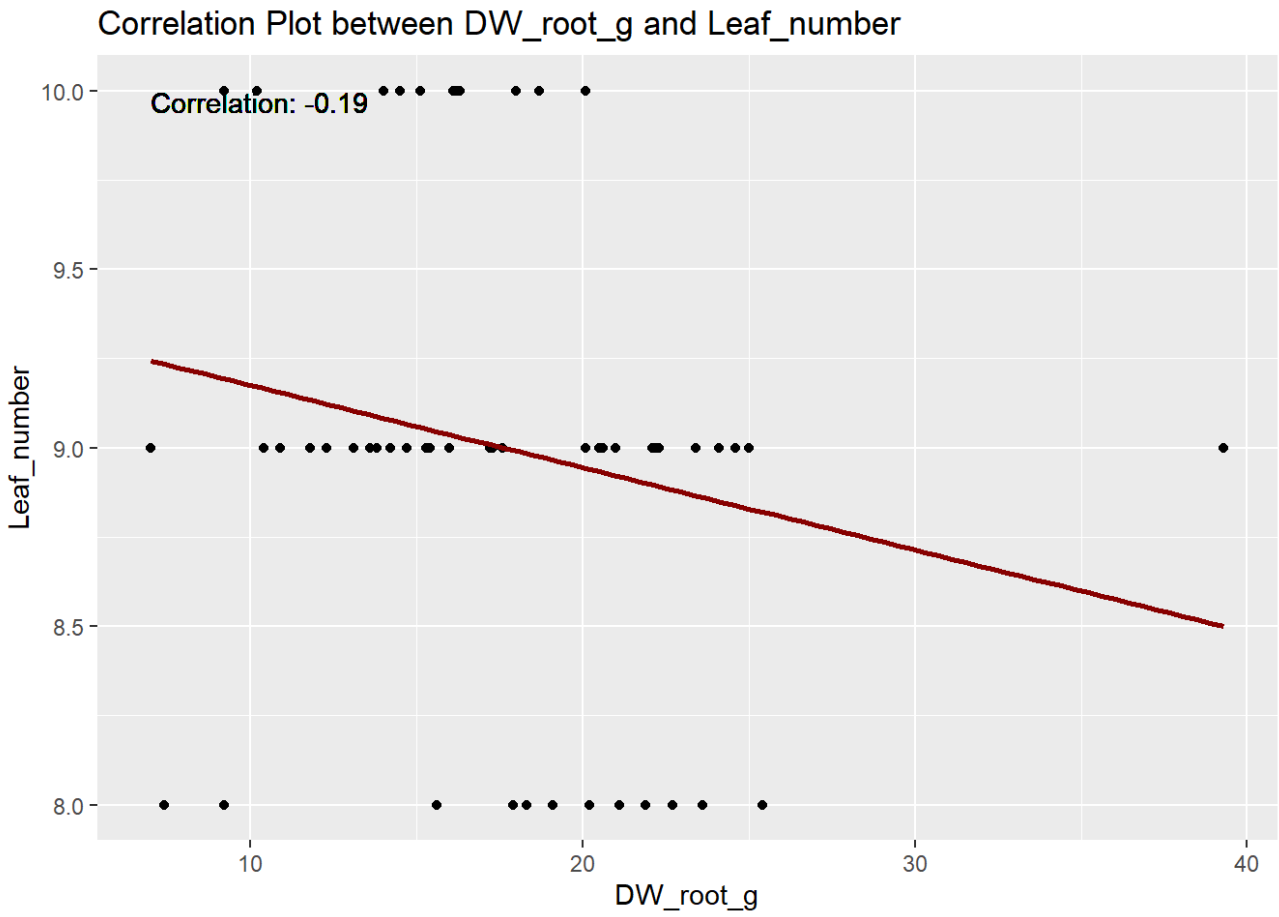

```
## `geom_smooth()` using formula = 'y ~ x'
```



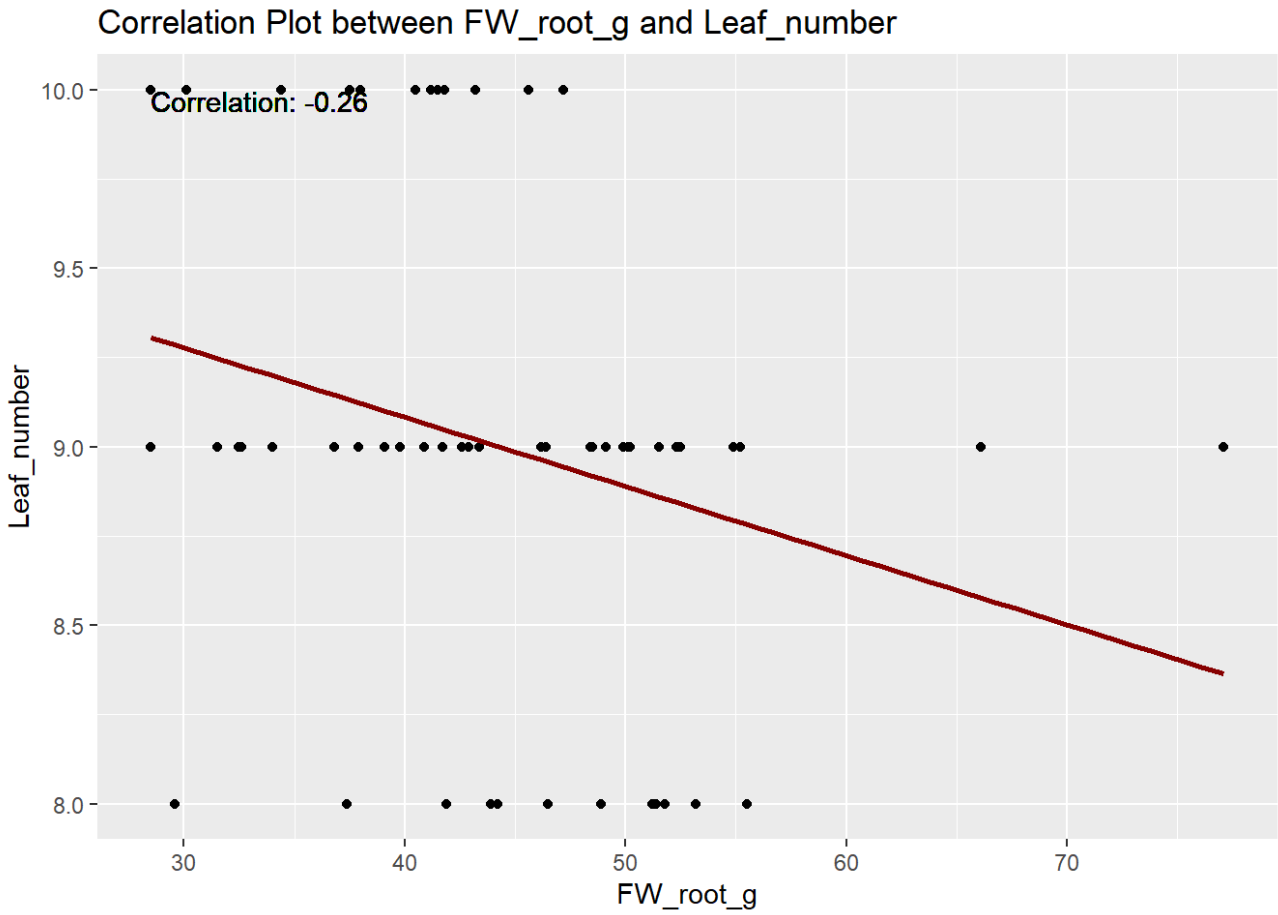

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
## `geom_smooth()` using formula = 'y ~ x'
```




```
## `geom_smooth()` using formula = 'y ~ x'
```

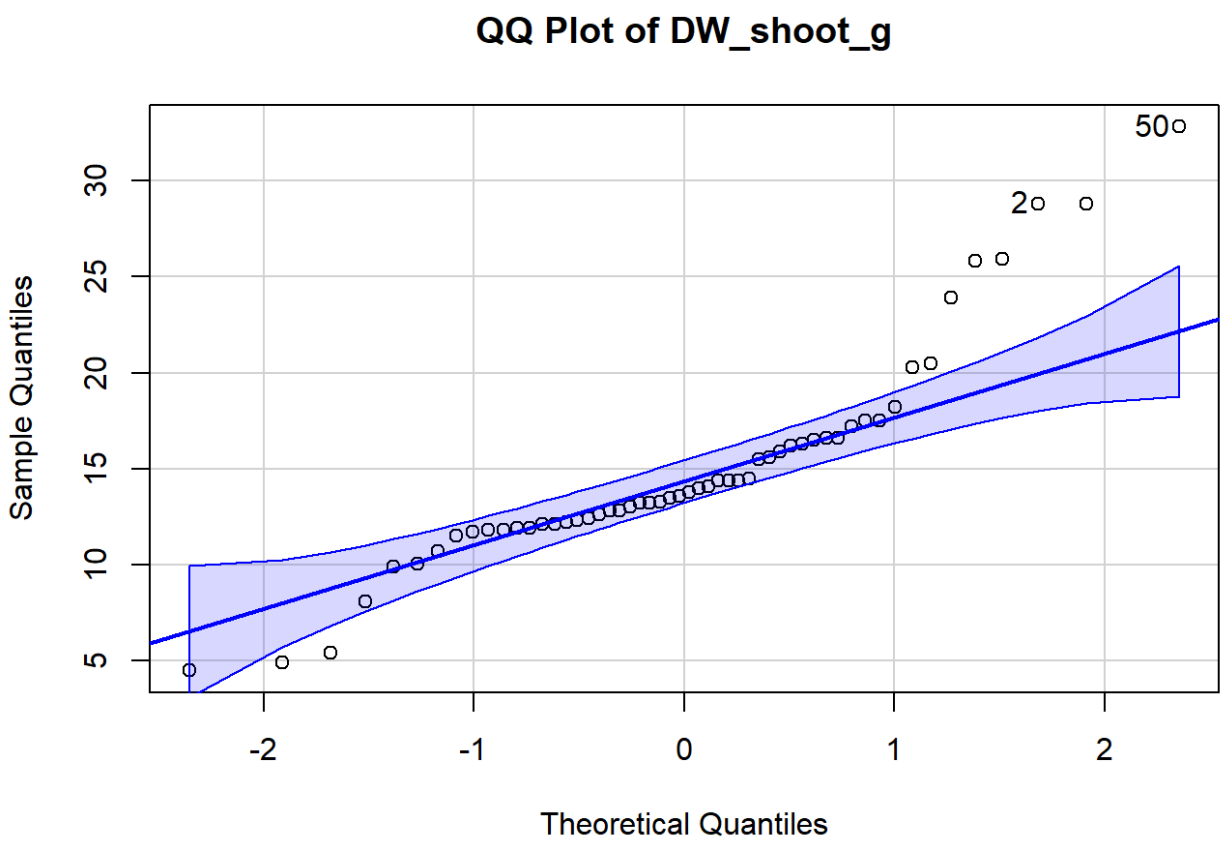


B. Normality hypothesis and outlier detection

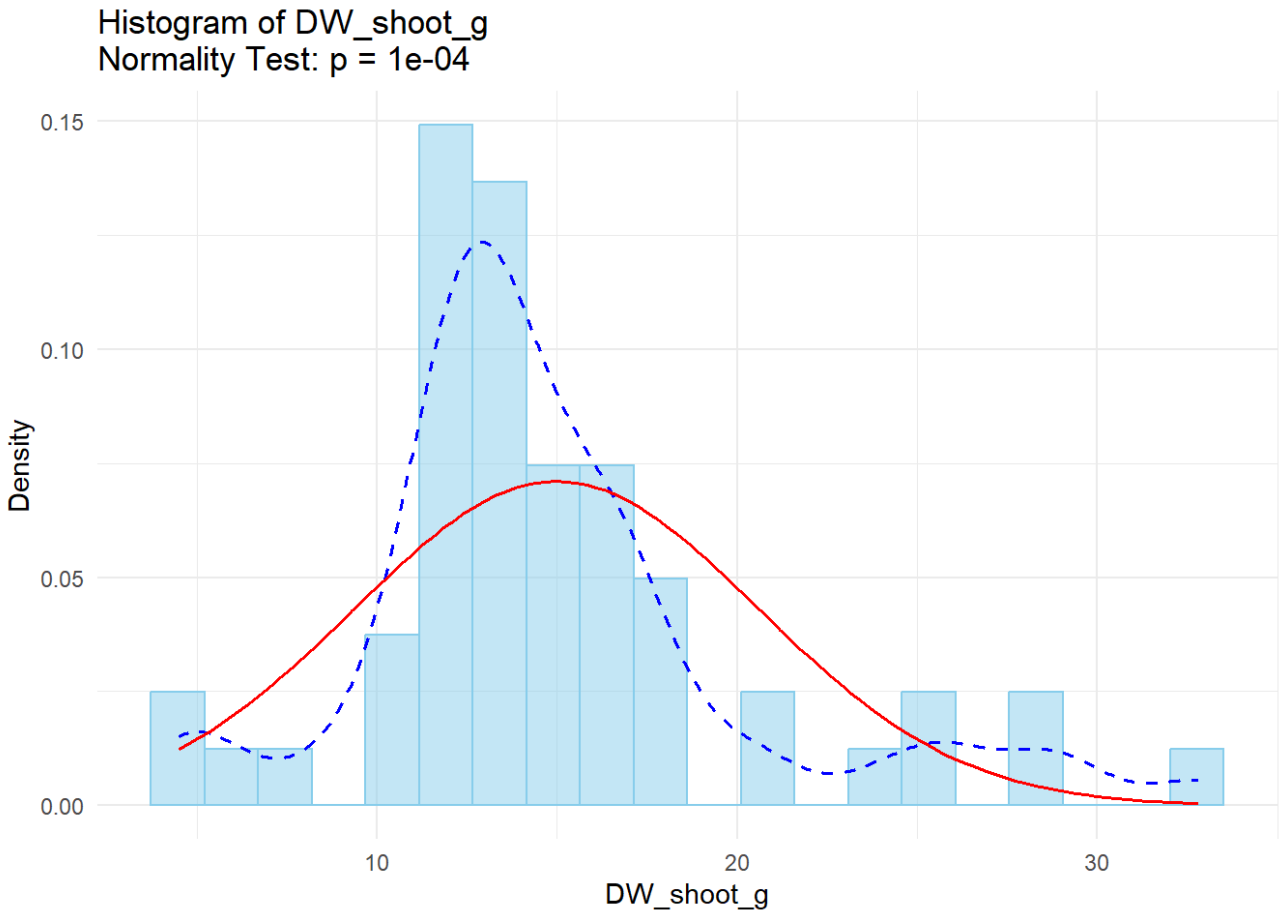
Test for normality hypothesis and plot density histogram. The red curve is the normal distribution, the blue dotted curve is the data density curve.

```
normality_results <- normality_test_histogram(endpoint)
```

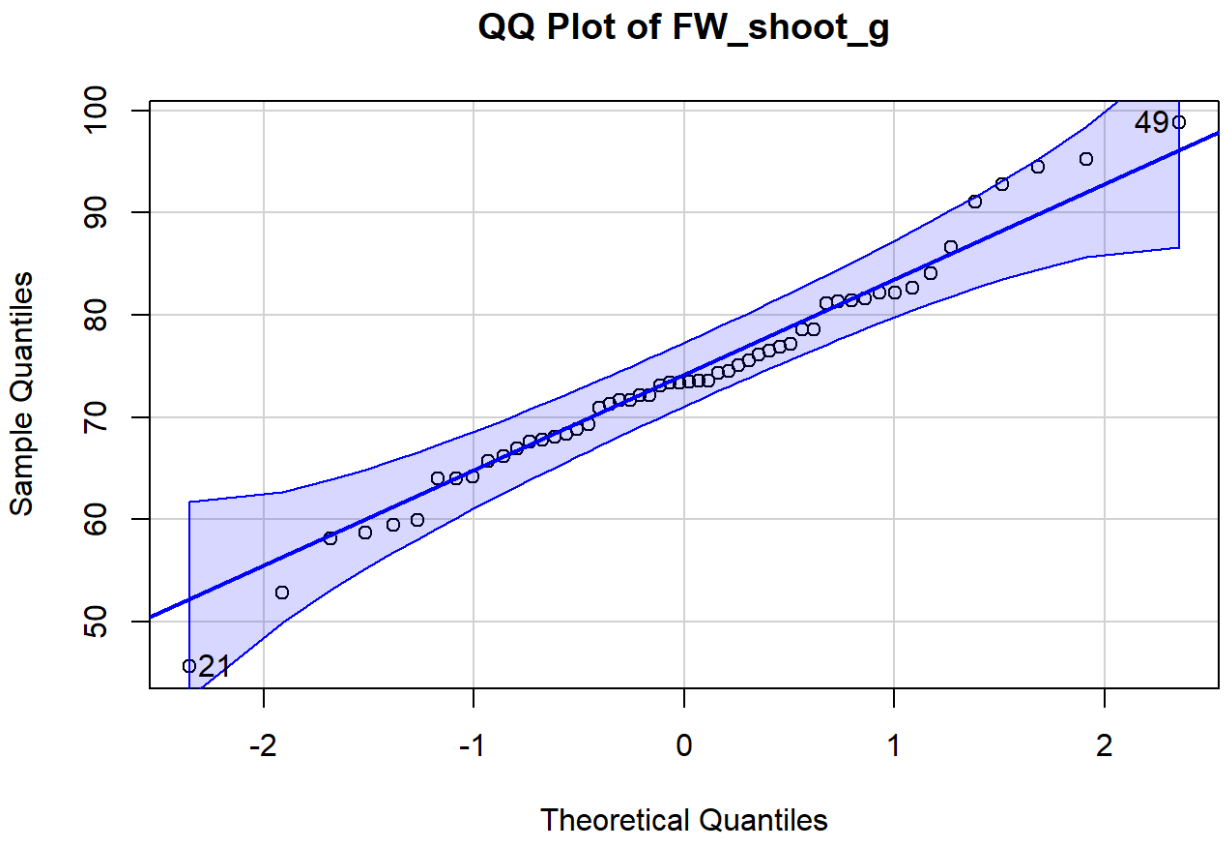
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

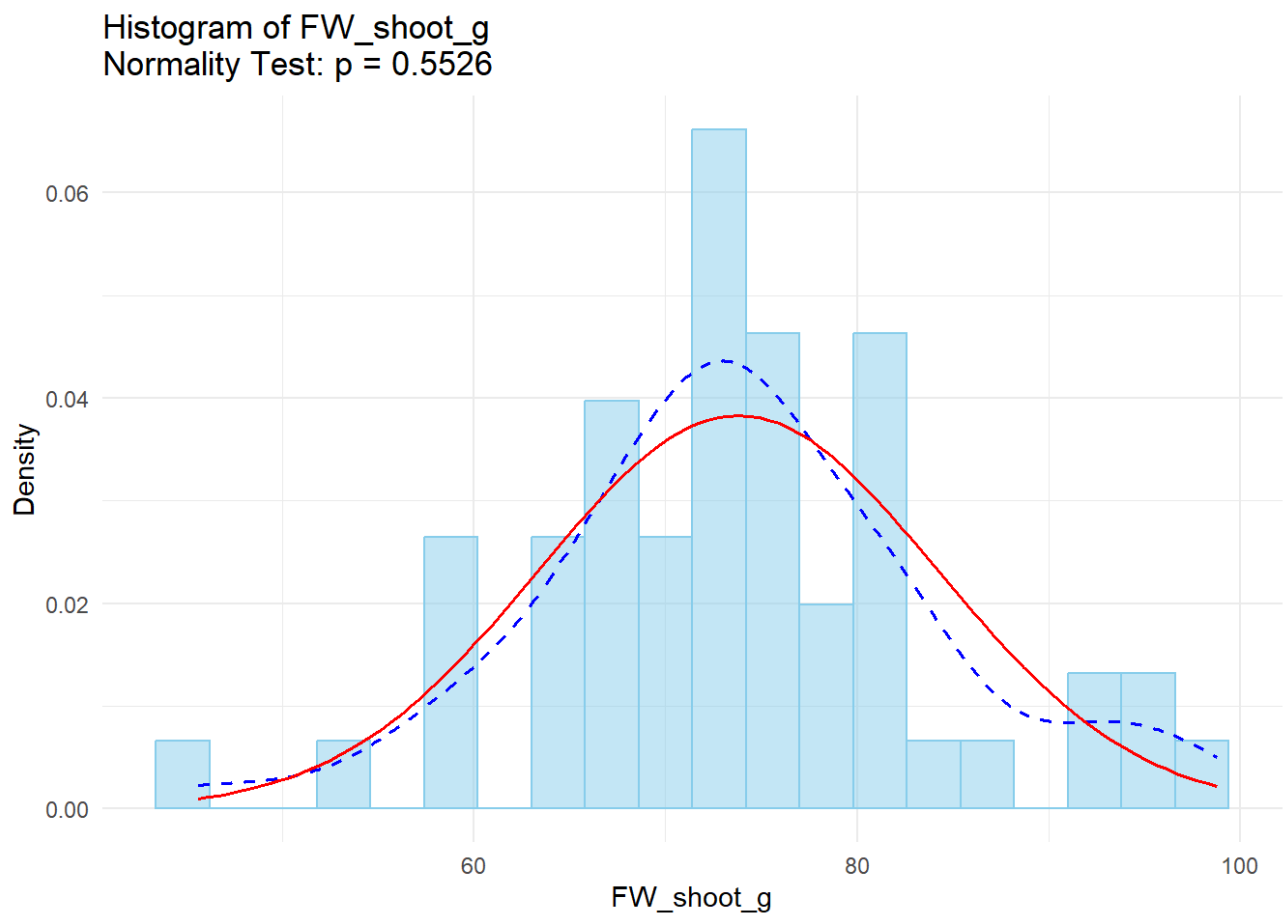



```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## ℹ Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

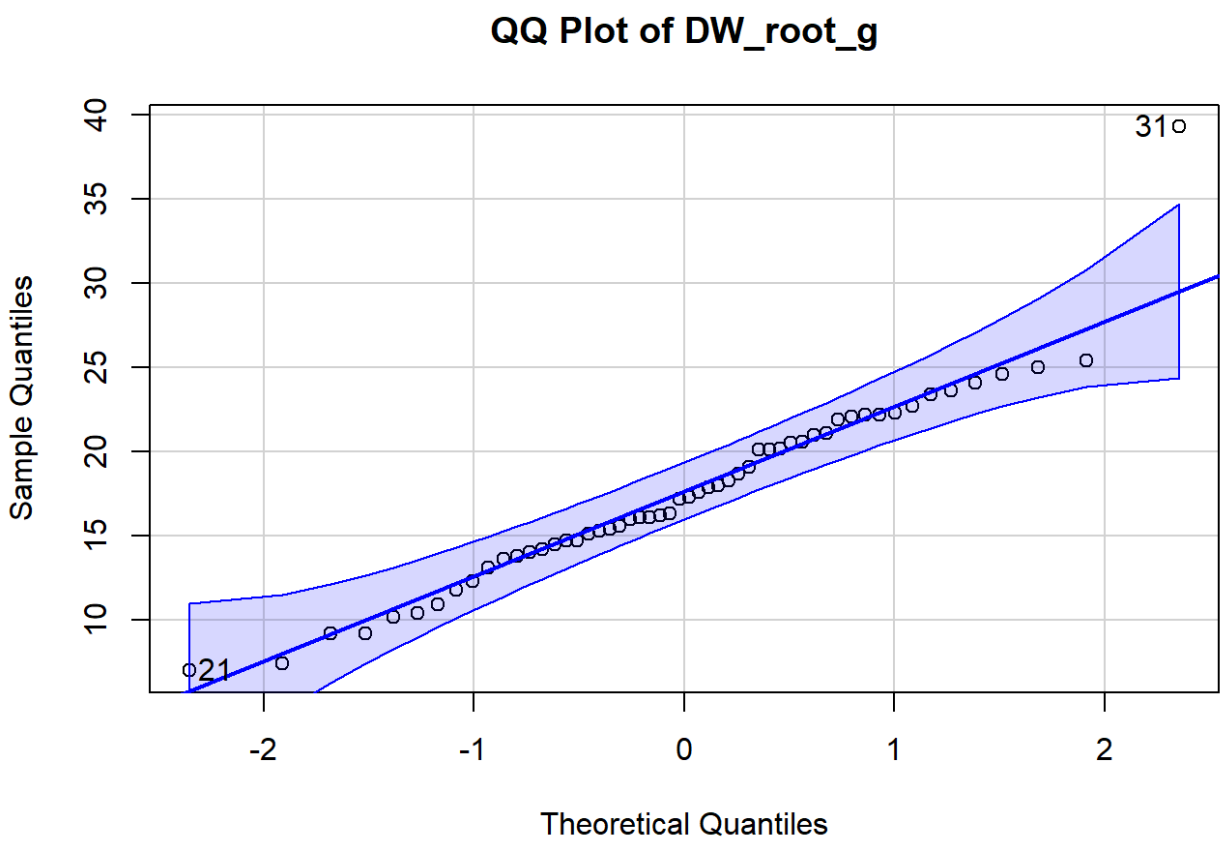

Histogram of DW_shoot_g
Normality Test: p = 1e-04
0.15
0.10
0.05
0.00
Density
10
20
30
DW_shoot_g

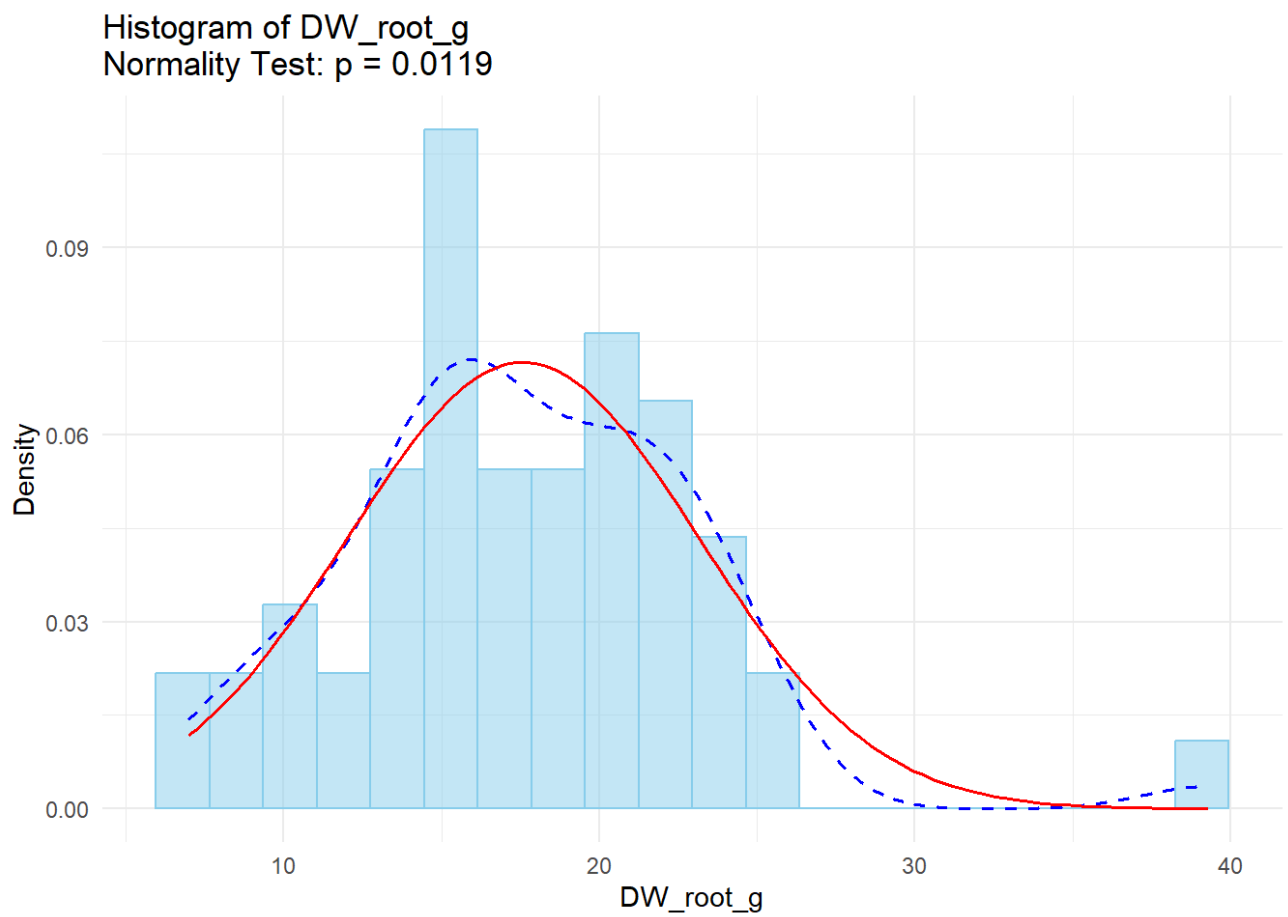
```
## [1] 50  2
```


QQ Plot of FW_shoot_g
Sample Quantiles
Theoretical Quantiles
50
60
70
80
90
100
-2
-1
0
1
2
21
49

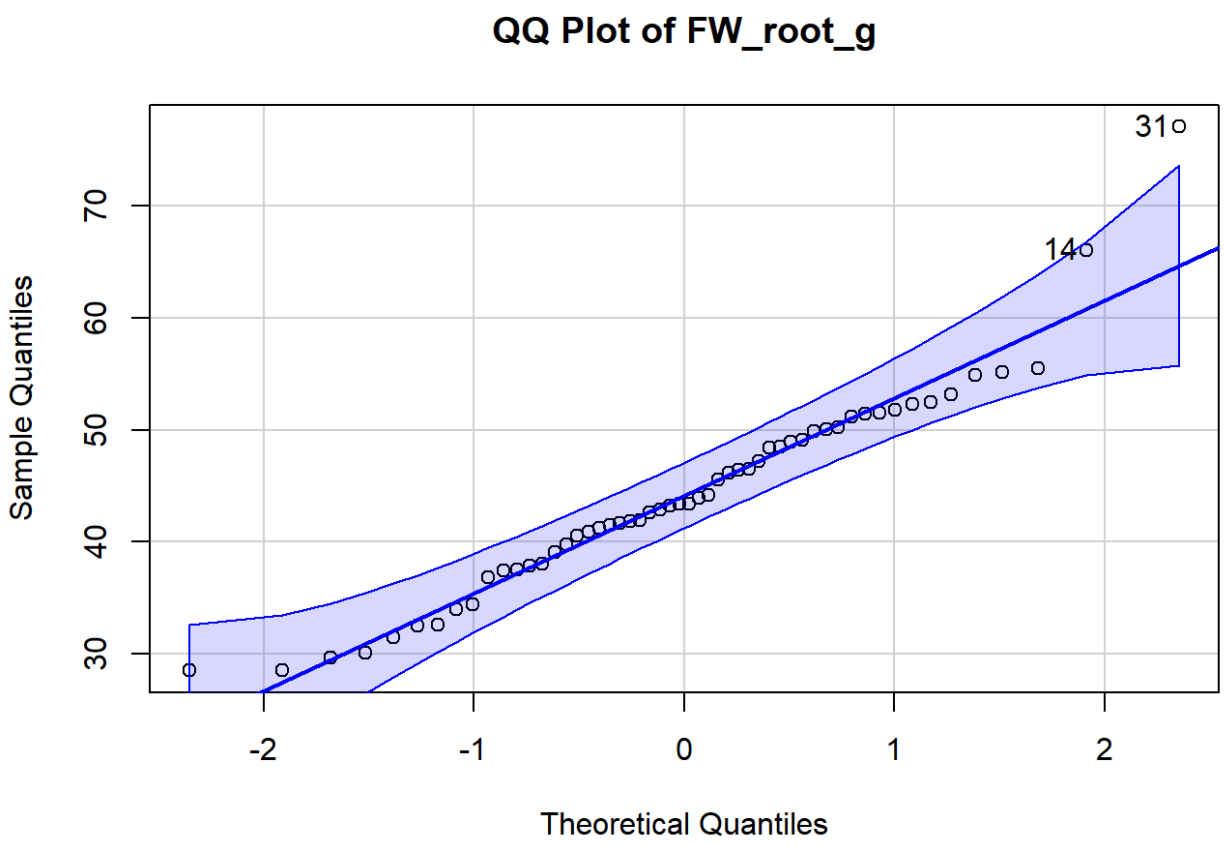

Histogram of FW_shoot_g
Normality Test: p = 0.5526
Density
FW_shoot_g
0.00
0.02
0.04
0.06
60
80
100

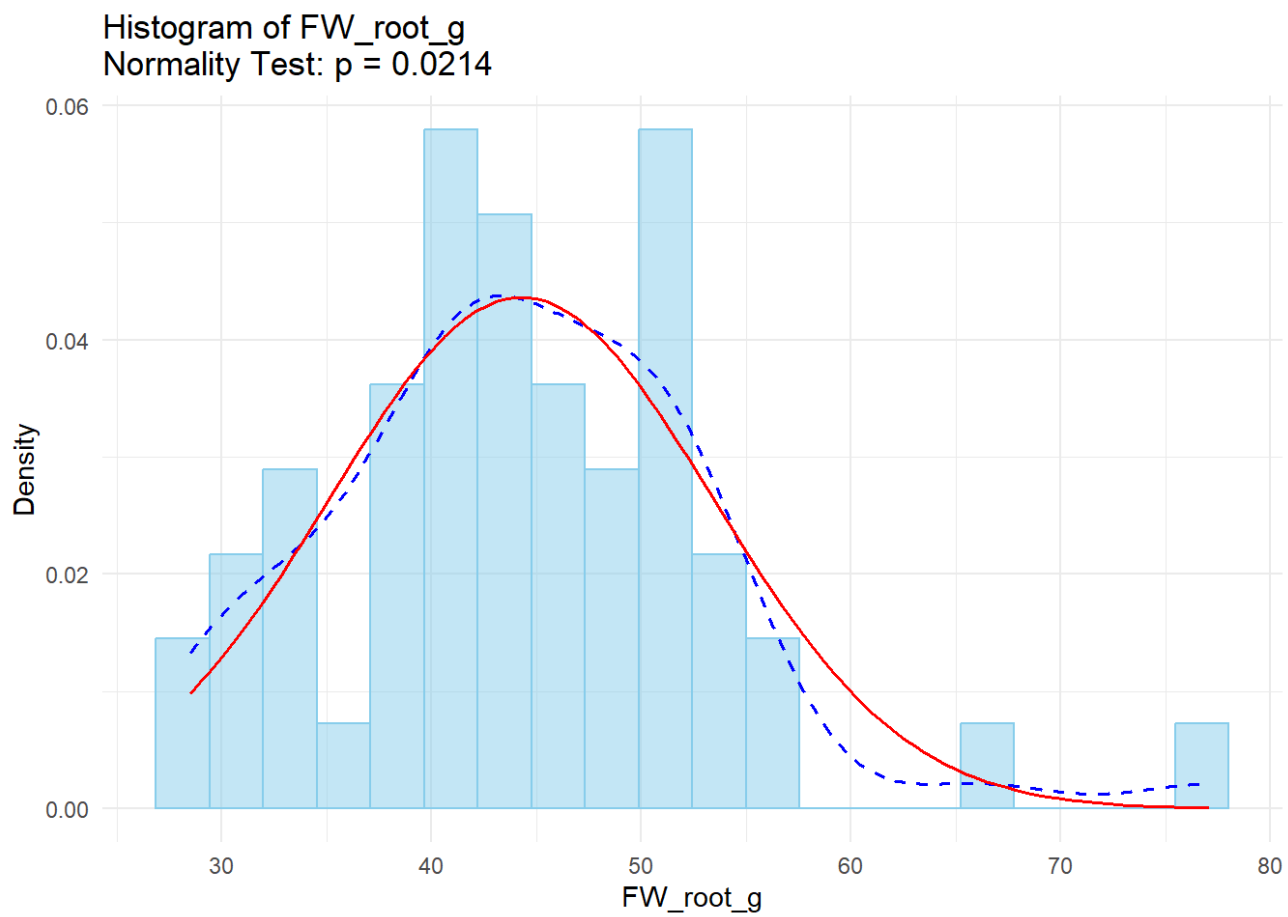
```
## [1] 21 49
```


QQ Plot of DW_root_g
Sample Quantiles
Theoretical Quantiles
31
21
10
15
20
25
30
35
40
-2
-1
0
1
2

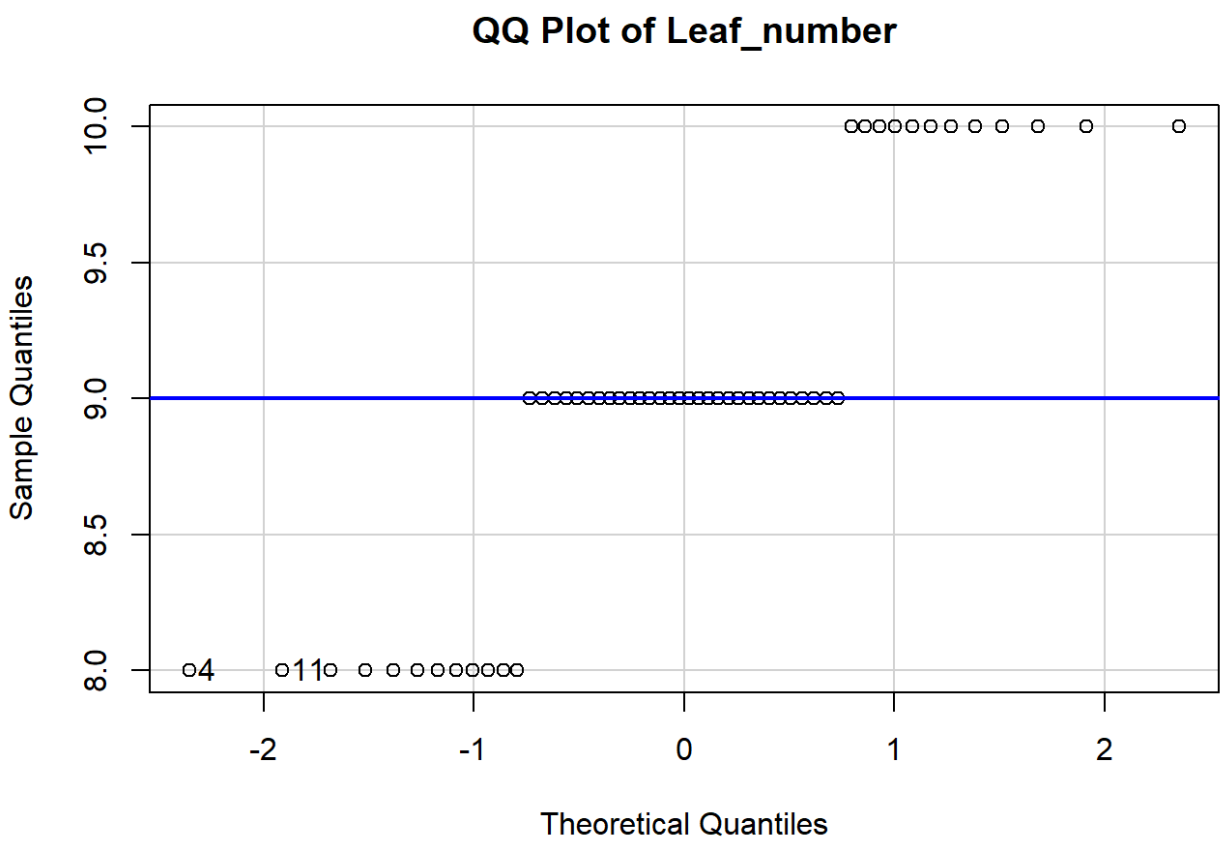

Histogram of DW_root_g
Normality Test: p = 0.0119
Density
DW_root_g
0.00
0.03
0.06
0.09
10
20
30
40

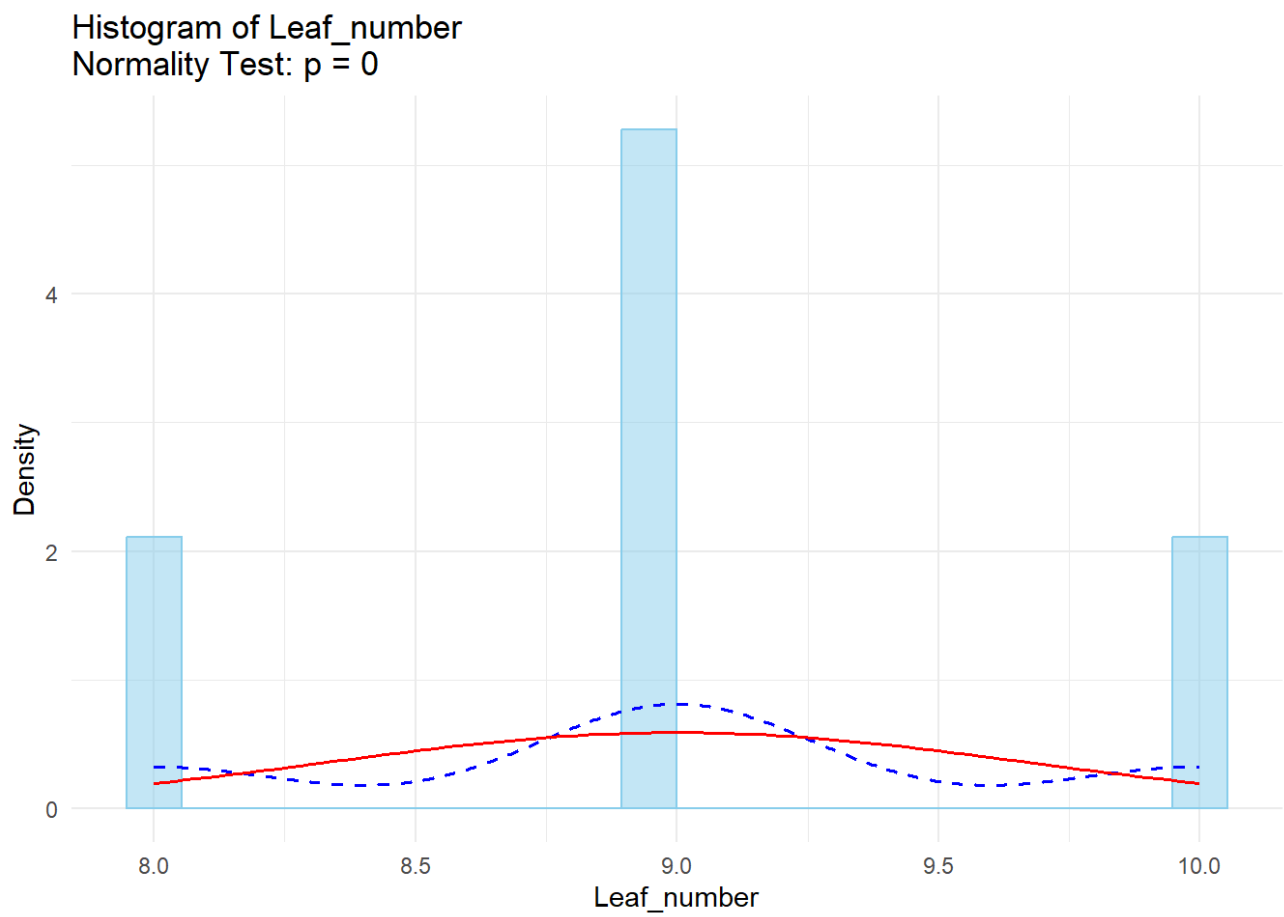
```
## [1] 31 21
```


QQ Plot of FW_root_g
Sample Quantiles
Theoretical Quantiles
31
14
30
40
50
60
70
-2
-1
0
1
2


Histogram of FW_root_g
Normality Test: p = 0.0214
Density
FW_root_g
0.00
0.02
0.04
0.06
30
40
50
60
70
80

```
## [1] 31 14
```






```
## [1]  4 11
```

Remove the outliers, replacing them with NULL values and normality visual verification.

The function detect_replace_ouliers_by_genotype checks for outlying values, using the Tukey method.

Then run the function on all variables of the dataset.

```
endpoint_clean <- endpoint
# Run the function on the dataset for all the variables
endpoint_clean <- detect_replace_outliers_by_genotype(endpoint_clean)
```

Boxplots after outlier detection

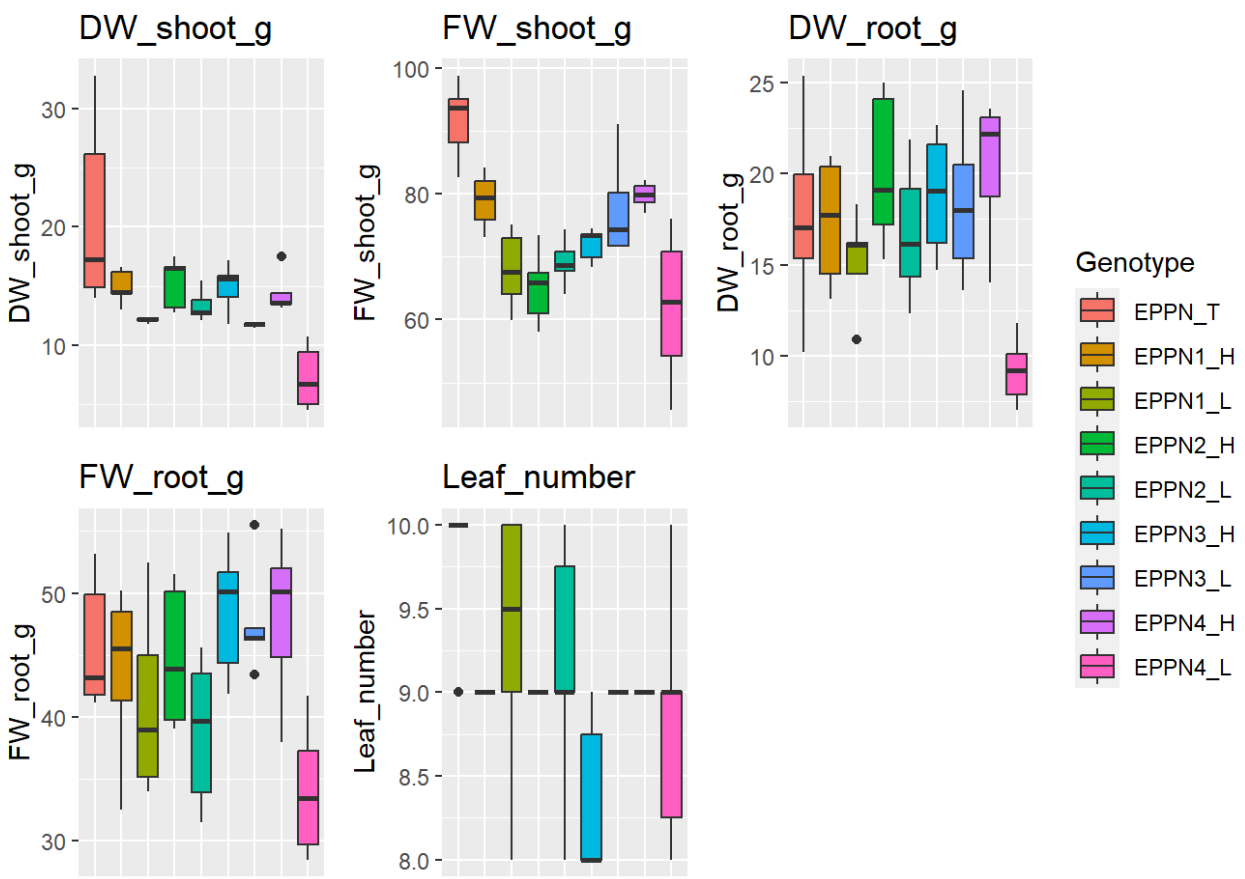
```
create_boxplots(endpoint_clean, variables, "Genotype")
```

```
## Warning: Removed 8 rows containing non-finite values (`stat_boxplot()`).
```

```
## Warning: Removed 2 rows containing non-finite values (`stat_boxplot()`).
```

```
## Warning: Removed 3 rows containing non-finite values (`stat_boxplot()`).
```

```
## Warning: Removed 6 rows containing non-finite values (`stat_boxplot()`).
```



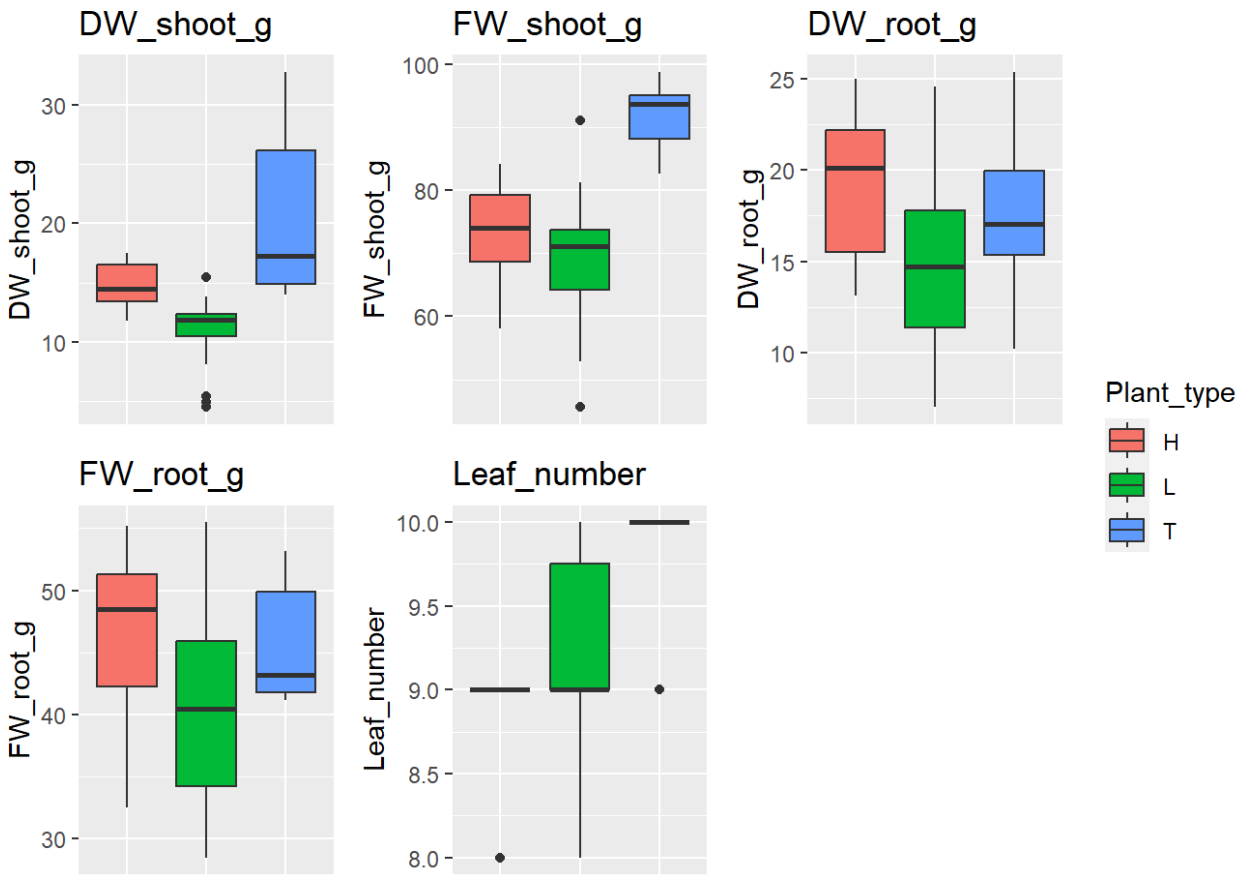

```
create_boxplots(endpoint_clean, variables, "Plant_type")
```

```
## Warning: Removed 8 rows containing non-finite values (`stat_boxplot()`).
```

```
## Warning: Removed 2 rows containing non-finite values (`stat_boxplot()`).
```

```
## Warning: Removed 3 rows containing non-finite values (`stat_boxplot()`).
```

```
## Warning: Removed 6 rows containing non-finite values (`stat_boxplot()`).
```



Violin and sina plots after outlier detection

```
create_violin_plots(endpoint_clean, variables, "Genotype")
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## ℹ Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: Removed 8 rows containing non-finite values (`stat_ydensity()`).
```

```
## Warning: Removed 8 rows containing non-finite values (`stat_sina()`).
```

```
## Warning: Removed 2 rows containing non-finite values (`stat_ydensity()`).
```

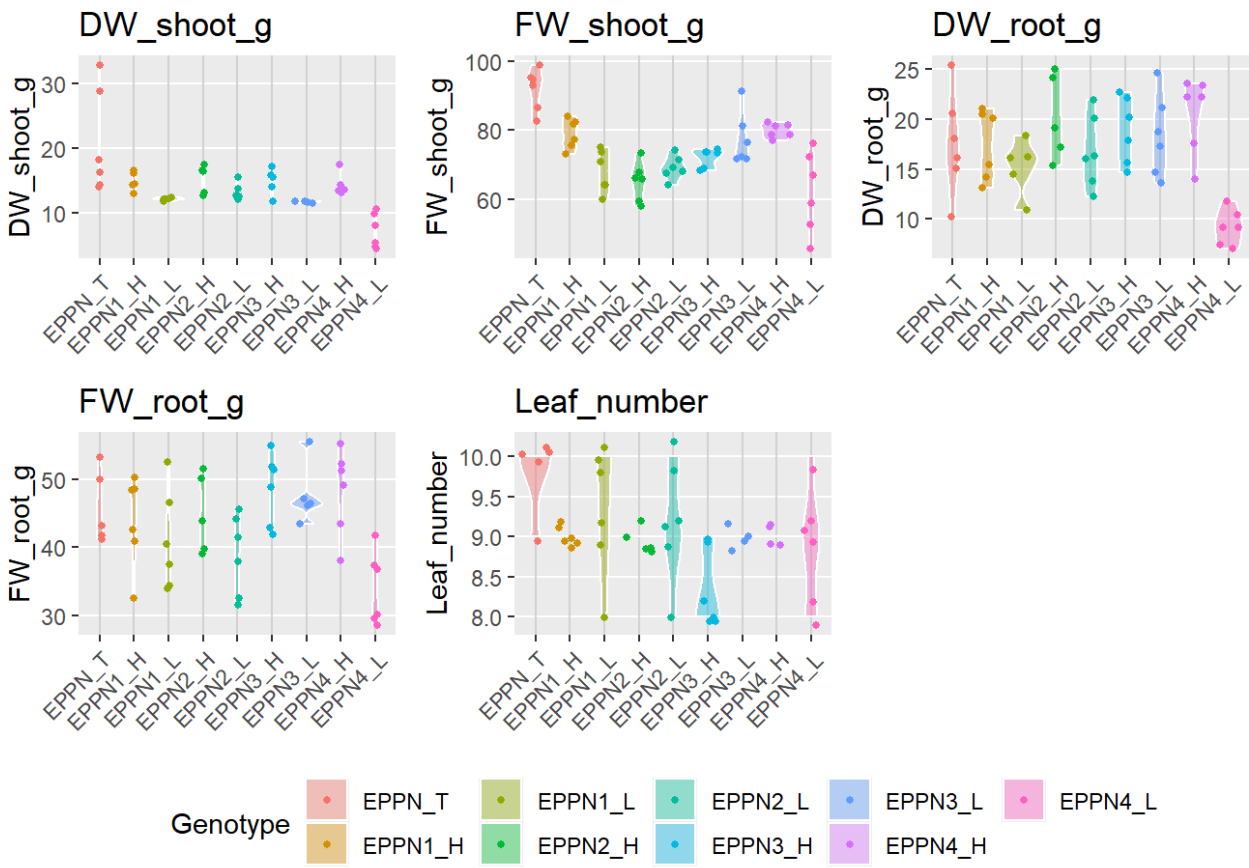
```
## Warning: Removed 2 rows containing non-finite values (`stat_sina()`).
```

```
## Warning: Removed 3 rows containing non-finite values (`stat_ydensity()`).
```

```
## Warning: Removed 3 rows containing non-finite values (`stat_sina()`).
```

```
## Warning: Removed 6 rows containing non-finite values (`stat_ydensity()`).
```

```
## Warning: Removed 6 rows containing non-finite values (`stat_sina()`).
```




```
create_violin_plots(endpoint_clean, variables, "Plant_type")
```

```
## Warning: Removed 8 rows containing non-finite values (`stat_ydensity()`).
```

```
## Warning: Removed 8 rows containing non-finite values (`stat_sina()`).
```

```
## Warning: Removed 2 rows containing non-finite values (`stat_ydensity()`).
```

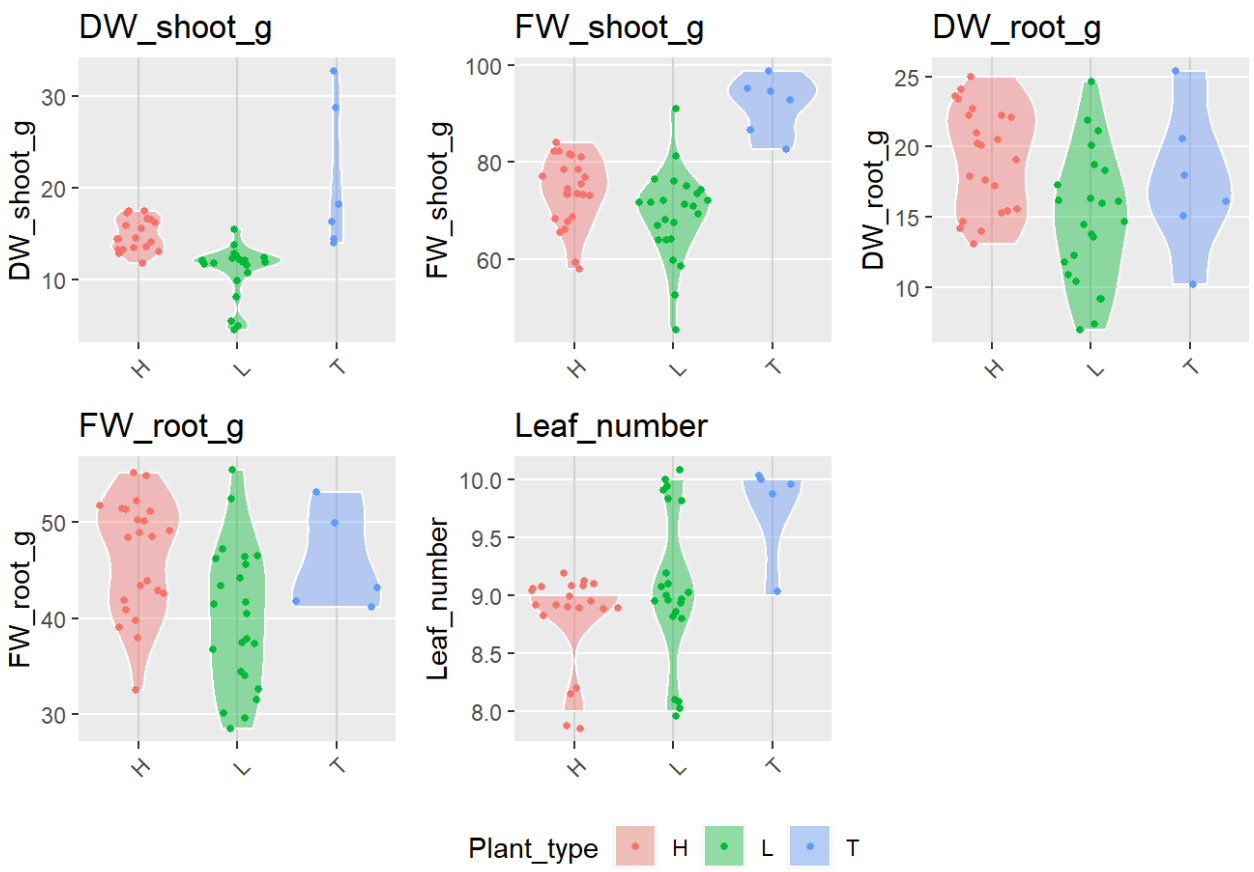
```
## Warning: Removed 2 rows containing non-finite values (`stat_sina()`).
```

```
## Warning: Removed 3 rows containing non-finite values (`stat_ydensity()`).
```

```
## Warning: Removed 3 rows containing non-finite values (`stat_sina()`).
```

```
## Warning: Removed 6 rows containing non-finite values (`stat_ydensity()`).
```

```
## Warning: Removed 6 rows containing non-finite values (`stat_sina()`).
```



Exploration statistics for the variables after outlier detection

```
skim(endpoint_clean[variables])
```

Data summary

Name	endpoint_clean[variables]
Number of rows	54
Number of columns	5
_____	
Column type frequency:	
numeric	5
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
DW_shoot_g	8	0.85	13.93	4.78	4.5	11.95	13.35	15.83	32.8	
FW_shoot_g	0	1.00	73.79	10.42	45.6	67.88	73.45	80.47	98.8	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
DW_root_g	2	0.96	17.08	4.72	7.0	14.15	16.75	20.70	25.4	
FW_root_g	3	0.94	43.52	7.21	28.5	38.55	43.40	49.50	55.5	
Leaf_number	6	0.89	9.04	0.62	8.0	9.00	9.00	9.00	10.0	

```
for (var in variables) {
  cat("\nSummary for:", var, "\n")
  endpoint_clean %>%
    group_by(Genotype) %>%
    summarize(mean    = mean(get(var), na.rm = TRUE),
              std.dev = sd(get(var), na.rm = TRUE),
              n_missing  = sum(is.na(get(var)))) %>%
    arrange(desc(mean)) %>%
    print(n = Inf)
}
```

```
##
## Summary for: DW_shoot_g
## # A tibble: 9 × 4
##   Genotype  mean std.dev n_missing
##   <fct>    <dbl>   <dbl>     <int>
## 1 EPPN_T   20.8    8.03          0
## 2 EPPN2_H  15.3    2.16          1
## 3 EPPN1_H  14.9    1.47          1
## 4 EPPN3_H  14.9    2.06          1
## 5 EPPN4_H  14.4    1.77          1
## 6 EPPN2_L  13.4    1.35          1
## 7 EPPN1_L  12.2    0.230         1
## 8 EPPN3_L  11.8    0.191         2
## 9 EPPN4_L   7.25   2.69          0
##
## Summary for: FW_shoot_g
## # A tibble: 9 × 4
##   Genotype  mean std.dev n_missing
##   <fct>    <dbl>   <dbl>     <int>
## 1 EPPN_T    91.8    6.01         0
## 2 EPPN4_H   79.8    2.06         0
## 3 EPPN1_H   79.0    4.32         0
## 4 EPPN3_L   77.4    7.70         0
## 5 EPPN3_H   72.0    2.72         0
## 6 EPPN2_L   69.1    3.50         0
## 7 EPPN1_L   68.0    6.10         0
## 8 EPPN2_H   65.1    5.64         0
## 9 EPPN4_L   62.0   11.7          0
##
## Summary for: DW_root_g
## # A tibble: 9 × 4
##   Genotype  mean std.dev n_missing
##   <fct>    <dbl>   <dbl>     <int>
## 1 EPPN4_H  20.5     3.86         0
## 2 EPPN2_H  20.1     4.26         1
## 3 EPPN3_H  18.9     3.34         0
## 4 EPPN3_L  18.3     4.09         0
## 5 EPPN_T   17.6     5.16         0
## 6 EPPN1_H  17.4     3.54         0
## 7 EPPN2_L  16.7     3.66         0
## 8 EPPN1_L  15.2     2.76         1
## 9 EPPN4_L   9.17    1.80         0
##
## Summary for: FW_root_g
## # A tibble: 9 × 4
##   Genotype  mean std.dev n_missing
##   <fct>    <dbl>   <dbl>     <int>
## 1 EPPN3_H   48.6    5.20         0
## 2 EPPN4_H   48.2    6.37         0
## 3 EPPN3_L   47.7    4.57         1
## 4 EPPN_T    45.9    5.37         1
## 5 EPPN2_H   44.9    5.73         1
## 6 EPPN1_H   43.8    6.66         0
## 7 EPPN1_L   40.9    7.31         0
## 8 EPPN2_L   38.9    5.92         0
```

```
## 9 EPPN4_L   34.0    5.36         0
##
## Summary for: Leaf_number
## # A tibble: 9 × 4
##   Genotype  mean std.dev n_missing
##   <fct>    <dbl>   <dbl>     <int>
## 1 EPPN_T    9.8    0.447         1
## 2 EPPN1_L   9.33   0.816         0
## 3 EPPN2_L   9.17   0.753         0
## 4 EPPN1_H   9      0             0
## 5 EPPN2_H   9      0             1
## 6 EPPN3_L   9      0             2
## 7 EPPN4_H   9      0             2
## 8 EPPN4_L   8.83   0.753         0
## 9 EPPN3_H   8.33   0.516         0
```

2. Exploration of the timeseries data

In this part, we look at the timeseries, S_timeseries and T_timeseries datasets, also using several functions, located in the functions.R script.

Number of data observations per day for the traits of the timeseries datasets

```
print(paste0("No data for", platform))
```

```
## [1] "No data forABER"
```

A. Exploration of the timeseries dataframe

```
print(paste0("No data for", platform))
```

```
## [1] "No data forABER"
```

B. Exploration of the S_timeseries dataframe

Scatter plots by Genotype

```
print(paste0("No data for", platform))
```

```
## [1] "No data forABER"
```

Scatterplots for all genotypes by Plant type (Hybride, Line, EPPN20_T) with smooth line.

```
print(paste0("No data for", platform))
```

```
## [1] "No data forABER"
```

Scatter plots for all genotypes by water treatment

```
print(paste0("No data for", platform))
```

```
## [1] "No data forABER"
```

C. Exploration of the T_timeseries dataframe

```
print(paste0("No data for", platform))
```

```
## [1] "No data forABER"
```