STATS604 Project 3

1. Experiment design

Study Objective

Assess how **light** and **watering** conditions affect green onion **height** and **freshness** across three independent labs (An, Dhruba, Chandler).

Materials & Sourcing

- **72** green onions were purchased from the same store (Way 1 supermarket). After excluding unhealthy bulbs (e.g., roots are too small), **59** onions remained for the study. All onions were trimmed to **10 cm** before assignment.

Experimental Units, Labs, and Treatment Subgroups

- The experiment is replicated in **three labs**: **An**, **Dhruba**, and **Chandler**.
- Treatment **subgroups (conditions)** considered across labs:
  SW — Sunlight + Water; **SN** — Sunlight + No Water; **DW** — Shade + Water;
  DN — Shade + No Water; **FN** — Fridge + No Water; **GL** — Grow Light + Water.

Allocation by Lab

- **An (n = 16)**: SW (4), DW (4), DN (4), FN (4).
- **Dhruba (n = 20)**: SW (4), SN (4), DW (4), DN (4), FN (4).

- **Chandler (n = 23)**: GL (3), SW (4), SN (4), DW (4), DN (4), FN (4).

Randomization

- Onions within each lab were **numbered 1..n_lab** and randomly permuted in R.
  The first block of indices was assigned to **SW**, the next to **SN**, and so on, until all onions were assigned to prespecified subgroup sizes. This ensured **within-lab random allocation** to conditions.

Measurement Schedule & Outcomes

- **Schedule**: Measurements were taken at **Day 0, 3, 6, and 9**.
- **Primary outcomes**:
  1. **Height** (cm) at each time point (with emphasis on **Day 9** for primary tests).
  2. **Freshness** (binary), determined on day 9 via a pre-specified rule (below).

Freshness Rule

A plant is labeled **fresh = 1** if: 1. **Leaf color** is good (vibrant/dark green), and
2. At least **two of three** visual checks are true: (i) **tip length 5 mm**, (ii) **no dead leaf**, (iii) **bulb is clear & white**.

Otherwise, **fresh = 0**.

2. Data load and cleaning

2.1. Setup

```
# Core
library(dplyr)
library(tidyr)
library(ggplot2)

# Helpers used in functions below
library(stringr)   # str_extract, str_to_title
library(forcats)   # fct_explicit_na
library(rlang)     # ensym, as_label
library(scales)    # percent_format
library(knitr)     # kable
```

2.2. Data load

```
# Load
proj3_data <- read.csv("../data/final_dataset.csv")
glimpse(proj3_data)
```

```
## Rows: 59
## Columns: 12
## $ lab        <chr> "C", "C", "C", "C", "C", "C", "C", "C", "C", "C", "C", "C",~
## $ label      <chr> "GL1", "GL2", "GL3", "SW1", "SW2", "SW3", "SW4", "SN1", "SN~
## $ day_0_len  <int> 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,~
## $ day_3_len  <dbl> 12.9, 13.2, 13.3, 12.5, 11.0, 13.7, 13.6, 11.3, 10.5, 11.8,~
## $ day_6_len  <dbl> 18.1, 18.0, 19.7, 14.5, 14.1, 17.2, 17.6, 12.5, 11.1, 12.4,~
## $ day_9_len  <dbl> 23.3, 20.7, 27.2, 17.6, 18.7, 19.8, 22.4, 13.6, 11.3, 12.9,~
## $ circumf    <dbl> 4.9, 4.5, 5.7, 4.7, 4.5, 3.9, 4.2, 4.6, 3.9, 4.7, 4.4, 4.6,~
## $ leaf_color <int> 2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 2, 1, 1, 1, 1, 2, 3, 3, 3, 3,~
## $ bulb_color <int> 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1,~
## $ tip        <dbl> 0.5, 2.0, 3.0, 2.0, 1.0, 3.0, 1.0, 3.0, 6.0, 2.0, 2.0, 0.5,~
## $ dead_leaf  <int> 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ freshness  <int> 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,~
```

Variables in the Dataset

The original dataset (`proj3_data`) contains **59 rows** × **12 columns**, with the following variables:

Variable	Type	Meaning	Possible Values / Range
lab	factor	Lab where the plant was grown	`"C"` for Chandler, `"A"` for An, `"D"` for Dhruba
label	factor	Unique plant ID including treatment code + serial number	e.g. `"GL1"`, `"SW3"`, `"DN2"`
day_0_len, **day_3_len**, **day_6_len**, **day_9_len**	numeric	Plant height (cm) measured over time	~10 cm initially, up to ~42 cm by Day 9
circumf	numeric	Stem circumference (cm)	~3.0 – 5.9 cm
leaf_color	factor	Visual rating of leaf color	`"1"` for vibrant green, `"2"` for dark green, `"3"` for yellowish/brownish
bulb_color	factor	Whether the lower bulb area is clear and white	`"1"` for clear & white, `"0"` for not clear/white
tip	numeric	Length of dried/brown leaf tip (mm)	~0.5 – 11.0 mm
dead_leaf	factor	Presence of dried/dead leaf tissue	`"1"` for Yes or `"0"` for No
freshness	factor	Rule-based computed freshness	`"1"` for Yes or `"0"` for No

---

```r
# Ensure numeric-typed columns for computable fields (safe even if already numeric)
tmp <- proj3_data %>%
  mutate(
    leaf_color_num = as.numeric(leaf_color),
    bulb_color_num = as.numeric(bulb_color),
    tip_num        = as.numeric(tip),
    dead_leaf_num  = as.numeric(dead_leaf),
    freshness_num  = as.numeric(freshness)
  )

# Recompute freshness based on the project rule:
# If (leaf_color <= 2) AND at least 3 of {leaf_color<=2, bulb_color==1, tip<=5, dead_leaf==0} are true
cond_leaf_ok <- !is.na(tmp$leaf_color_num) & tmp$leaf_color_num <= 2
cond_bulb    <- !is.na(tmp$bulb_color_num) & tmp$bulb_color_num == 1
cond_tip     <- !is.na(tmp$tip_num)        & tmp$tip_num <= 5
cond_dead    <- !is.na(tmp$dead_leaf_num)  & tmp$dead_leaf_num == 0
```

```r
sum_conds <- rowSums(cbind(cond_leaf_ok, cond_tip, cond_dead, cond_bulb), na.rm = TRUE)
freshness_calc <- ifelse(cond_leaf_ok & sum_conds >= 3, 1L, 0L)

# Compare with provided freshness (if present). We report *true mismatches* (exclude provided NAs).
provided <- tmp$freshness_num
mism_idx <- which(!is.na(provided) & (freshness_calc != provided))
message("Freshness check: ",
        sum(freshness_calc == provided, na.rm = TRUE), " match; ",
        length(mism_idx), " mismatch (excluding provided NAs).")
if (length(mism_idx) > 0) {
  print(dplyr::tibble(row = mism_idx,
                      provided = provided[mism_idx],
                      computed = freshness_calc[mism_idx]))
}

# Final typed dataset: numeric where numeric, categorical as factors
df <- tmp %>%
  mutate(
    day_0_len = as.numeric(day_0_len),
    day_3_len = as.numeric(day_3_len),
    day_6_len = as.numeric(day_6_len),
    day_9_len = as.numeric(day_9_len),
    circumf   = as.numeric(circumf),
    tip       = as.numeric(tip_num),  # keep numeric for analysis
    lab   = factor(lab, levels = c("C","A","D"), labels = c("Chandler","An","Dhruba")),
    label = factor(label),  # keeps GL/DN/DW/SN/SW/FN prefixes with serial numbers
    # Human-readable factor encodings
    leaf_color = factor(as.integer(leaf_color_num),
                        levels = c(1,2,3),
                        labels = c("vibrant green","dark green","yellowish/brownish")),
    bulb_color = factor(as.integer(bulb_color_num),
                        levels = c(0,1),
                        labels = c("not clear/white","clear & white")),
    dead_leaf  = factor(as.integer(dead_leaf_num),
                        levels = c(0,1),
                        labels = c("No","Yes")),
    freshness  = factor(ifelse(is.na(freshness_num), NA, as.integer(freshness_num)),
                        levels = c(0,1),
                        labels = c("No","Yes")),
    freshness_calc = factor(as.integer(freshness_calc), levels = c(0,1), labels = c("No","Yes"))
  ) %>%
  select(-ends_with("_num"))

# Derive label group (GL/DN/DW/SN/SW/FN) and serial number
df <- df %>%
  mutate(
    label_chr    = as.character(label),
    label_grp    = stringr::str_extract(label_chr, "^[A-Za-z]+"),
    label_serial = as.integer(stringr::str_extract(label_chr, "(\\d+)$"))
  ) %>%
  mutate(label_grp = factor(label_grp)) %>%
  select(-label_chr)
```

```r
glimpse(df)
```

```
## Rows: 59
## Columns: 15
## $ lab            <fct> Chandler, Chandler, Chandler, Chandler, Chandler, Chand~
## $ label          <fct> GL1, GL2, GL3, SW1, SW2, SW3, SW4, SN1, SN2, SN3, SN4, ~
## $ day_0_len      <dbl> 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,~
## $ day_3_len      <dbl> 12.9, 13.2, 13.3, 12.5, 11.0, 13.7, 13.6, 11.3, 10.5, 1~
## $ day_6_len      <dbl> 18.1, 18.0, 19.7, 14.5, 14.1, 17.2, 17.6, 12.5, 11.1, 1~
## $ day_9_len      <dbl> 23.3, 20.7, 27.2, 17.6, 18.7, 19.8, 22.4, 13.6, 11.3, 1~
## $ circumf        <dbl> 4.9, 4.5, 5.7, 4.7, 4.5, 3.9, 4.2, 4.6, 3.9, 4.7, 4.4, ~
## $ leaf_color     <fct> dark green, dark green, dark green, dark green, dark gr~
## $ bulb_color     <fct> not clear/white, clear & white, clear & white, clear & ~
## $ tip            <dbl> 0.5, 2.0, 3.0, 2.0, 1.0, 3.0, 1.0, 3.0, 6.0, 2.0, 2.0, ~
## $ dead_leaf      <fct> Yes, Yes, Yes, Yes, Yes, No, Yes, Yes, No, Yes, No, Yes~
## $ freshness      <fct> No, Yes, Yes, Yes, Yes, Yes, Yes, Yes, No, No, Yes, Yes~
## $ freshness_calc <fct> No, Yes, Yes, Yes, Yes, Yes, Yes, Yes, No, No, Yes, Yes~
## $ label_grp      <fct> GL, GL, GL, SW, SW, SW, SW, SN, SN, SN, SN, DW, DW, DW,~
## $ label_serial   <int> 1, 2, 3, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4~
```

Note (derived fields). `label_grp` and `label_serial` are **not in the original dataset**; they are created during cleaning from the `label` column.
1. `label_grp`: alphabetical treatment prefix extracted from `label` (GL, DN, DW, SN, SW, FN).
2. `label_serial`: numeric suffix extracted from `label` (1, 2, 3, …).

3. `freshness_calc`: This is used only as a **consistency check** against the originally recorded `freshness` field.

Treatment Conditions (`label_grp`) Each plant belongs to one of **six treatment combinations** based on lighting and watering conditions:

Code	Meaning
SW	Sunlight + Water
SN	Sunlight + No Water
DW	Shade + Water
DN	Shade + No Water
GL	Grow Light + Water
FN	Fridge + No Water

---

3. Exploratory Data Analysis

3.1. Treatment structure & counts

```r
lab_label_table <- df %>%
  count(lab, label_grp) %>%
  pivot_wider(names_from = label_grp, values_from = n, values_fill = 0) %>%
  arrange(lab)

lab_label_table %>% knitr::kable(caption = "Counts by Lab × Label Group")
```

Table 3: Counts by Lab × Label Group

lab	DN	DW	FN	GL	SN	SW
Chandler	4	4	4	3	4	4
An	4	4	4	0	0	4
Dhruba	4	4	4	0	4	4

Interpretation. The table shows sample sizes for each **Lab** $\times$ **Treatment** cell. **Chandler** is the only one who ran the **Grow Light (GL)** condition, so those experiments were conducted there. **An** did not have sufficient onions to populate all five light $\times$ water groups, so some cells are sparse or missing.

3.2. Variance analysis

Circumference by Lab

```r
lab_var <- df %>%
  group_by(lab) %>%
  summarise(
    n_circ    = sum(!is.na(circumf)),
    mean_circ= mean(circumf, na.rm = TRUE),
    var_circ = var(circumf,  na.rm = TRUE),
    sd_circ   = sqrt(var_circ),
    .groups = "drop"
  ) %>%
  arrange(lab)

lab_var %>% knitr::kable(caption = "Circumference summary by Lab")
```

Table 4: Circumference summary by Lab

lab	n_circ	mean_circ	var_circ	sd_circ
Chandler	23	4.473913	0.3456522	0.5879219
An	16	4.212500	0.5865000	0.7658329
Dhruba	20	4.125000	0.1409211	0.3753945

```r
# Quick diagnostic: spread by lab
lab_sd <- df %>%
  group_by(lab) %>%
  summarise(n = sum(!is.na(circumf)),
            sd_circ = sd(circumf, na.rm = TRUE),
            .groups = "drop")
ratio_max_min <- with(lab_sd, max(sd_circ, na.rm = TRUE) / min(sd_circ, na.rm = TRUE))
print(ratio_max_min)  # ~1 means very similar spreads; > ~1.5-2 suggests heteroskedasticity
```

```
## [1] 2.040075
```

Interpretation. The lab-wise standard deviations of circumference differ by a factor of **2.04** (max/min). As a rule of thumb, ratios above 1.5–2.0 suggest **heteroskedasticity** (non-constant variance) that may reflect **batch/lab effects**.

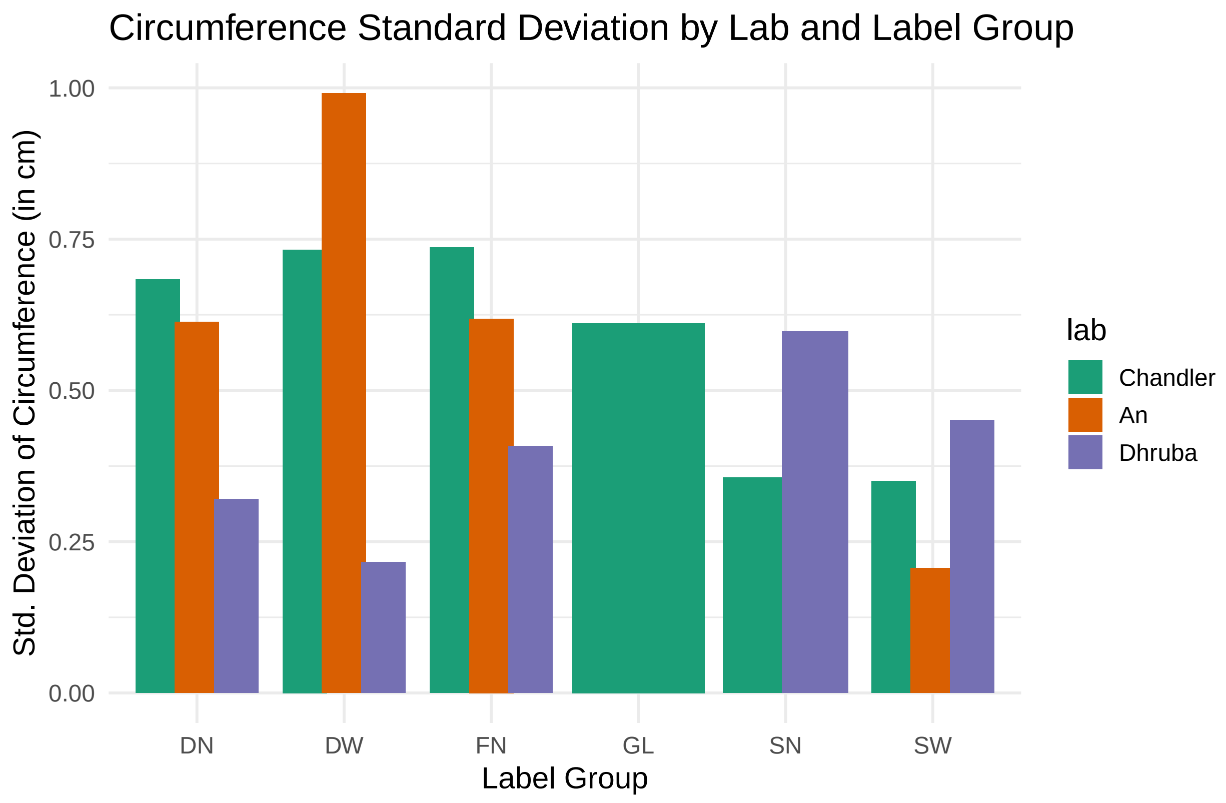
Circumference by Lab × Label Group

```r
lab_label_var <- df %>%
  group_by(lab, label_grp) %>%
  summarise(
    n_circ    = sum(!is.na(circumf)),
    mean_circ= mean(circumf, na.rm = TRUE),
    var_circ = var(circumf,  na.rm = TRUE),
    sd_circ  = sqrt(var_circ),
    .groups = "drop"
  ) %>%
  arrange(lab, label_grp)

lab_label_var %>% knitr::kable(caption = "Circumference summary by Lab × Label Group")
```

Table 5: Circumference summary by Lab × Label Group

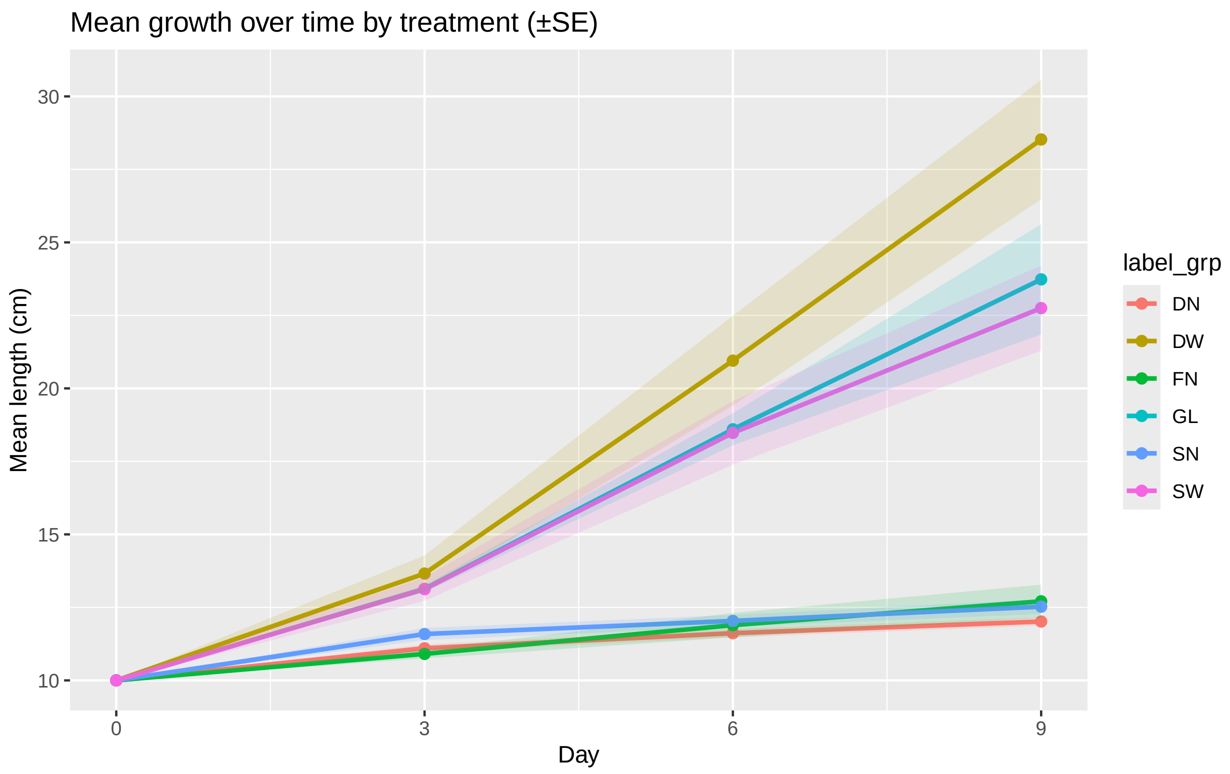
lab	label_grp	n_circ	mean_circ	var_circ	sd_circ
Chandler	DN	4	4.600000	0.4666667	0.6831301
Chandler	DW	4	4.450000	0.5366667	0.7325754
Chandler	FN	4	4.175000	0.5425000	0.7365460
Chandler	GL	3	5.033333	0.3733333	0.6110101
Chandler	SN	4	4.400000	0.1266667	0.3559026
Chandler	SW	4	4.325000	0.1225000	0.3500000
An	DN	4	4.575000	0.3758333	0.6130525
An	DW	4	4.325000	0.9825000	0.9912114
An	FN	4	4.525000	0.3825000	0.6184658
An	SW	4	3.425000	0.0425000	0.2061553
Dhruba	DN	4	4.125000	0.1025000	0.3201562
Dhruba	DW	4	4.200000	0.0466667	0.2160247
Dhruba	FN	4	4.100000	0.1666667	0.4082483
Dhruba	SN	4	4.050000	0.3566667	0.5972158
Dhruba	SW	4	4.150000	0.2033333	0.4509250

```r
ggplot(lab_label_var, aes(x = label_grp, y = sd_circ, fill = lab)) +
  geom_col(position = position_dodge(width = 0.8)) +
  labs(
    title = "Circumference Standard Deviation by Lab and Label Group",
    x = "Label Group",
    y = "Std. Deviation of Circumference (in cm)"
  ) +
  theme_minimal(base_size = 14) +
  scale_fill_brewer(palette = "Dark2")
```



Interpretation. Within *Light* × *Water* treatment groups (e.g., **SW** = Sunlight+Water; **SN** = Sunlight+No Water; **DW** = Shade+Water; **DN** = Shade+No Water; **GL** = Grow Light; **FN** = Fridge+No Water), the **spread of circumference** varies across labs. Treatments with larger **sd** may indicate greater biological variability or measurement variability under that condition and lab.

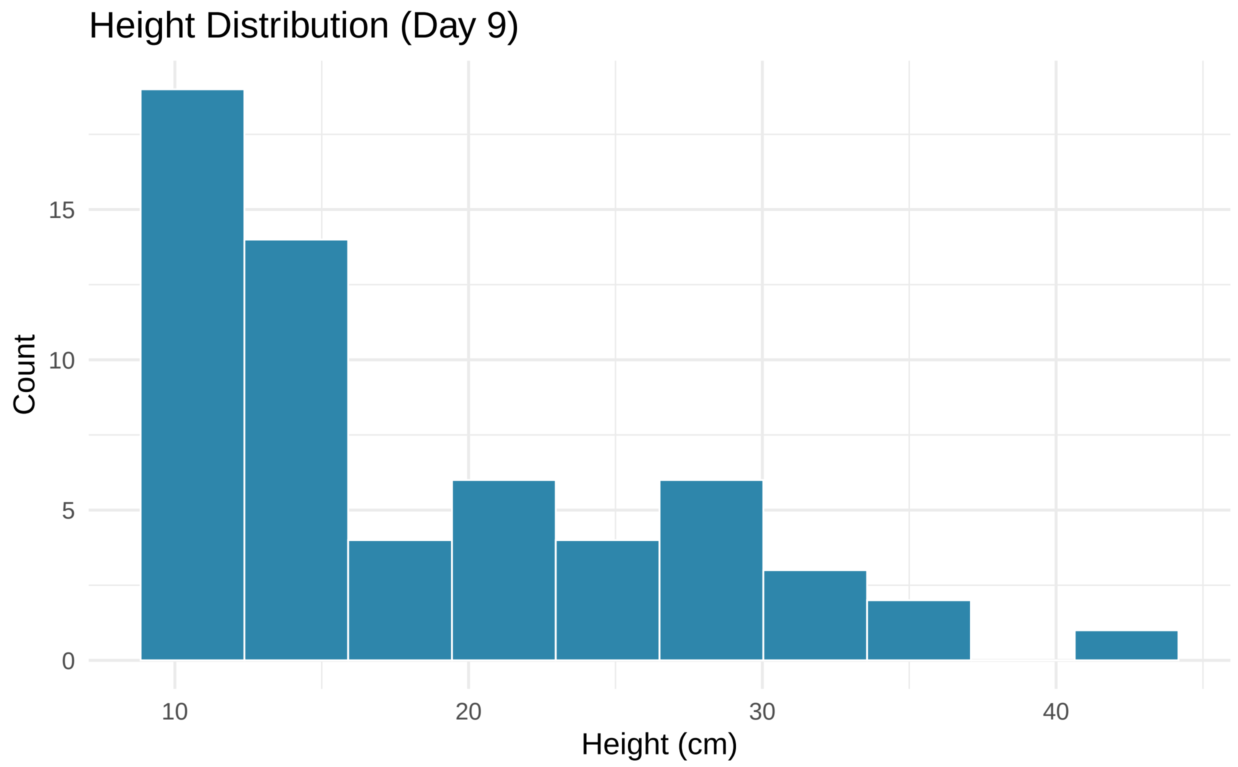
3.3. Growth over Time



Interpretation All treatment groups start at roughly the same initial height on Day 0, but their growth trajectories begin to separate clearly over time. The DW (Shade + Water) group shows the fastest and most sustained growth, ending with the highest mean height by Day 9, followed by GL (Grow Light + Water) and SW (Sunlight + Water). In contrast, the SN (Sunlight + No water), DN (Shade + No Water) and FN (Fridge + No Water) groups show very limited growth, indicating that the absence of water strongly restricts development regardless of light conditions. The widening gap between watered and non-watered treatments over time reinforces that watering drives cumulative growth.

3.4. Distribution of Height on day 9.

```
# Histogram helper (accepts "day_9" or "day_9_len")
plot_height_hist_day <- function(df, day, bins = 20) {
  stopifnot(is.character(day), length(day) == 1)
  day_col <- if (day %in% names(df)) {
    day
  } else if (paste0(day, "_len") %in% names(df)) {
    paste0(day, "_len")
  } else {
    stop(sprintf("Column '%s' (or '%s_len') not found in df.", day, day))
  }
  lab <- stringr::str_to_title(gsub("_len$", "", day_col))
  lab <- gsub("day_(\\d+)", "Day \\1", tolower(lab), perl = TRUE)
  dat <- df %>% transmute(height_cm = suppressWarnings(as.numeric(.data[[day_col]]))) %>% filter(!is.na
  ggplot(dat, aes(x = height_cm)) +
    geom_histogram(bins = bins, boundary = 0, closed = "left",
                   fill = "#2E86AB", color = "white", linewidth = 0.4) +
    labs(title = sprintf("Height Distribution (%s)", lab), x = "Height (cm)", y = "Count") +
    theme_minimal(base_size = 14)
}

print(plot_height_hist_day(df, "day_9_len", bins = 10))
```



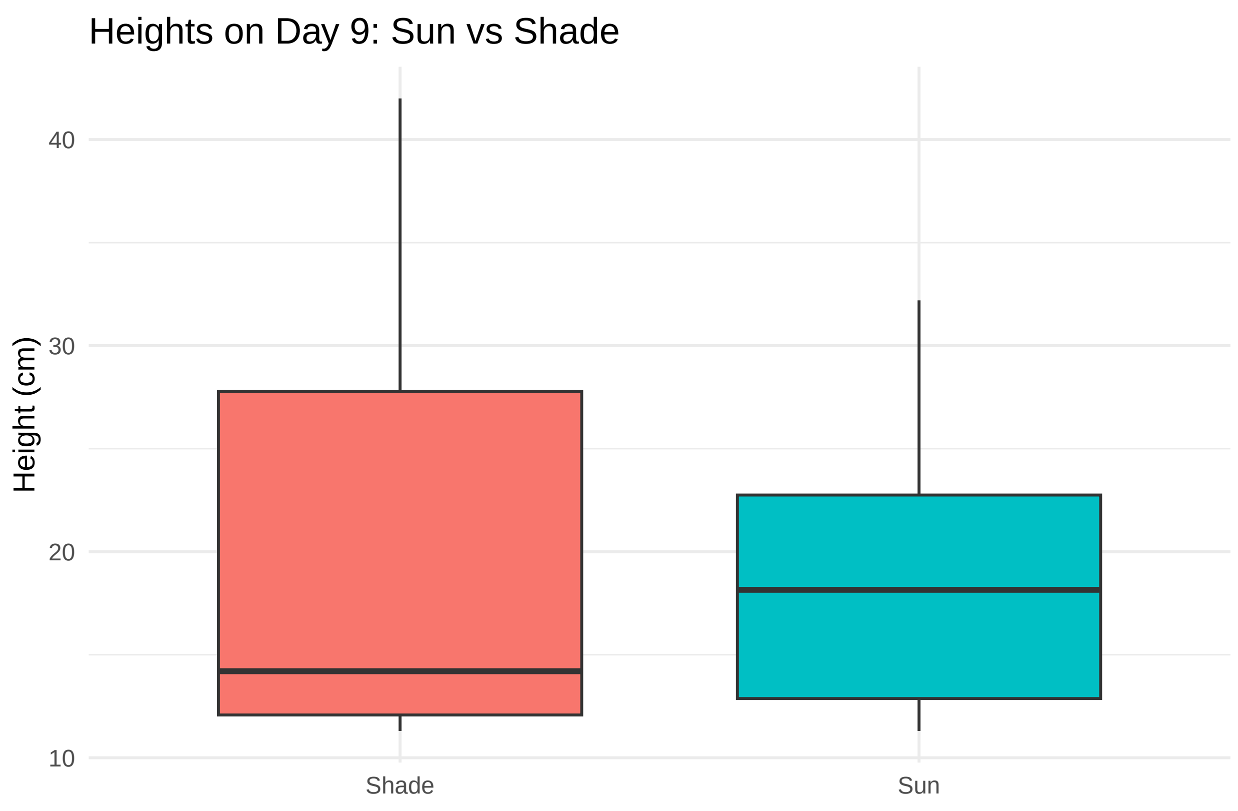
3.5. Height comparisons by light and watering

```
# Sun vs Shade for a given day column
plot_sunshade_day <- function(df, day_col) {
  stopifnot(is.character(day_col), length(day_col) == 1)
  if (!day_col %in% names(df)) stop(sprintf("Column '%s' not found in df.", day_col))

  dat <- df %>%
    transmute(
      sun_shade = dplyr::case_when(
        label_grp %in% c("SN","SW") ~ "Sun",
        label_grp %in% c("DN","DW") ~ "Shade",
        TRUE ~ NA_character_
      ),
      height_cm = .data[[day_col]]
    ) %>%
    filter(!is.na(sun_shade), !is.na(height_cm))

  day_no <- stringr::str_to_title(gsub("_len$", "", day_col))
```

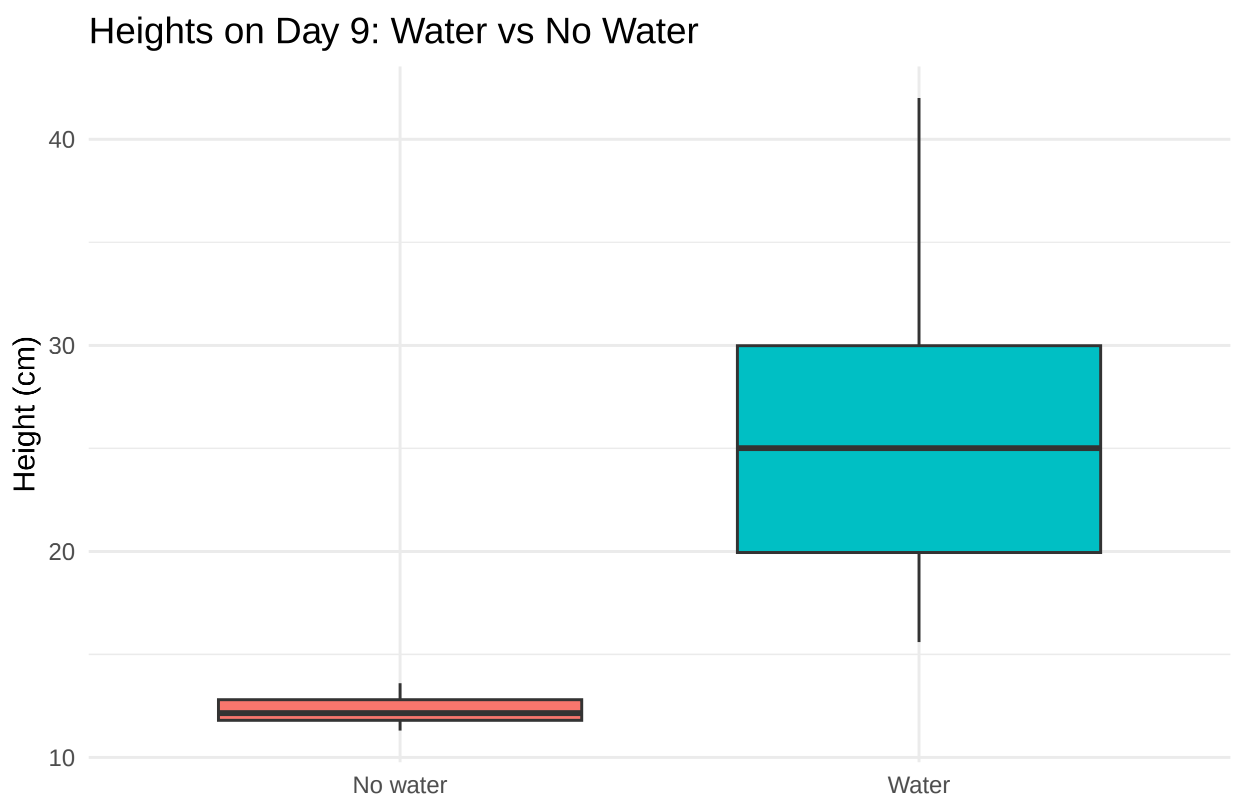
```r
  day_no <- gsub("day_(\\d+)", "Day \\1", tolower(day_no), perl = TRUE)

  ggplot(dat, aes(x = sun_shade, y = height_cm, fill = sun_shade)) +
    geom_boxplot(outlier.alpha = 0.25, width = 0.7) +
    labs(
      title = sprintf("Heights on %s: Sun vs Shade", day_no),
      x = NULL, y = "Height (cm)"
    ) +
    theme_minimal(base_size = 14) +
    theme(legend.position = "none")
}

# Water vs No Water for a given day column
plot_water_day <- function(df, day_col) {
  stopifnot(is.character(day_col), length(day_col) == 1)
  if (!day_col %in% names(df)) stop(sprintf("Column '%s' not found in df.", day_col))

  dat <- df %>%
    transmute(
      water = dplyr::case_when(
        label_grp %in% c("DW","SW") ~ "Water",
        label_grp %in% c("DN","SN") ~ "No water",
        TRUE ~ NA_character_ # GL excluded (watering unspecified)
      ),
      height_cm = .data[[day_col]]
    ) %>%
    filter(!is.na(water), !is.na(height_cm))

  day_no <- stringr::str_to_title(gsub("_len$", "", day_col))
  day_no <- gsub("day_(\\d+)", "Day \\1", tolower(day_no), perl = TRUE)

  ggplot(dat, aes(x = water, y = height_cm, fill = water)) +
    geom_boxplot(outlier.alpha = 0.25, width = 0.7) +
    labs(
      title = sprintf("Heights on %s: Water vs No Water", day_no),
      x = NULL, y = "Height (cm)"
    ) +
    theme_minimal(base_size = 14) +
    theme(legend.position = "none")
}

# Lab × Water and Lab × Sun/Shade for a given day column
plot_day_labXwater_sunshade <- function(df, day_col) {
  stopifnot(is.character(day_col), length(day_col) == 1)
  if (!day_col %in% names(df)) stop(sprintf("Column '%s' not found in df.", day_col))
  if (!("label_grp" %in% names(df))) stop("df must contain 'label_grp'.")
  if (!("lab" %in% names(df))) stop("df must contain 'lab'.")

  # Water vs No Water (exclude GL)
  dat_water <- df %>%
    transmute(
      lab,
      water = dplyr::case_when(
        label_grp %in% c("DW","SW") ~ "Water",
        label_grp %in% c("DN","SN") ~ "No water",
        TRUE ~ NA_character_
      ),
      height_cm = .data[[day_col]]
    ) %>%
    filter(!is.na(water), !is.na(height_cm), !is.na(lab)) %>%
    mutate(water = factor(water, levels = c("No water","Water")))

  day_no <- stringr::str_to_title(gsub("_len$", "", day_col))
  day_no <- gsub("day_(\\d+)", "Day \\1", tolower(day_no), perl = TRUE)

  p_water <- ggplot(dat_water, aes(x = lab, y = height_cm, fill = water)) +
    geom_boxplot(position = position_dodge(width = 0.75), width = 0.65, outlier.alpha = 0.25) +
    labs(
      title = sprintf("Heights on %s: Lab × Water vs No Water", day_no),
      x = "Lab", y = "Height (cm)", fill = "Water"
    ) +
    theme_minimal(base_size = 14)

  # Sun vs Shade (exclude GL & FN)
  dat_sunshade <- df %>%
    transmute(
      lab,
      sun_shade = dplyr::case_when(
        label_grp %in% c("SN","SW") ~ "Sun",
        label_grp %in% c("DN","DW") ~ "Shade",
        TRUE ~ NA_character_
      ),
      height_cm = .data[[day_col]]
    ) %>%
    filter(!is.na(sun_shade), !is.na(height_cm), !is.na(lab)) %>%
    mutate(sun_shade = factor(sun_shade, levels = c("Shade","Sun")))

  p_sunshade <- ggplot(dat_sunshade, aes(x = lab, y = height_cm, fill = sun_shade)) +
    geom_boxplot(position = position_dodge(width = 0.75), width = 0.65, outlier.alpha = 0.25) +
    labs(
      title = sprintf("Heights on %s: Lab × Sun vs Shade", day_no),
      x = "Lab", y = "Height (cm)", fill = "Light"
    ) +
    theme_minimal(base_size = 14)

  list(water_plot = p_water, sunshade_plot = p_sunshade)
}
```

Day 9 height comparisons

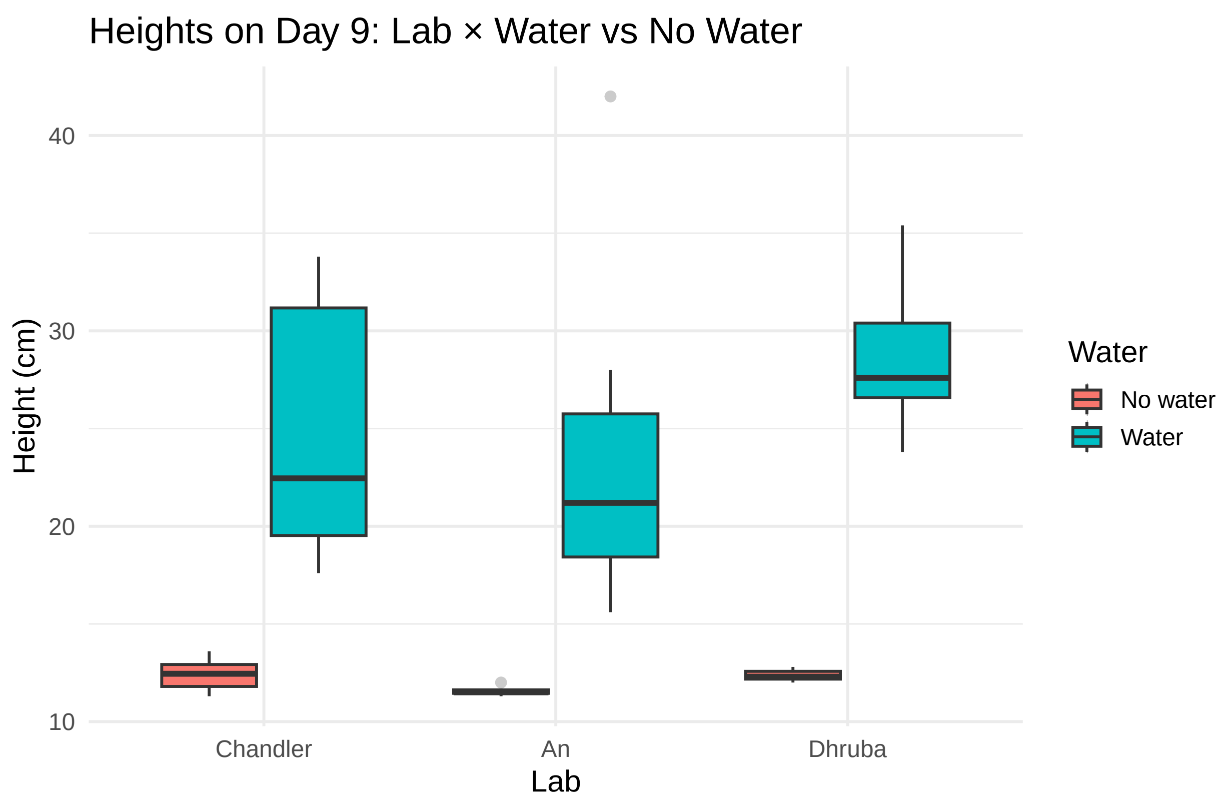
```r
print(plot_sunshade_day(df, "day_9_len"))
```



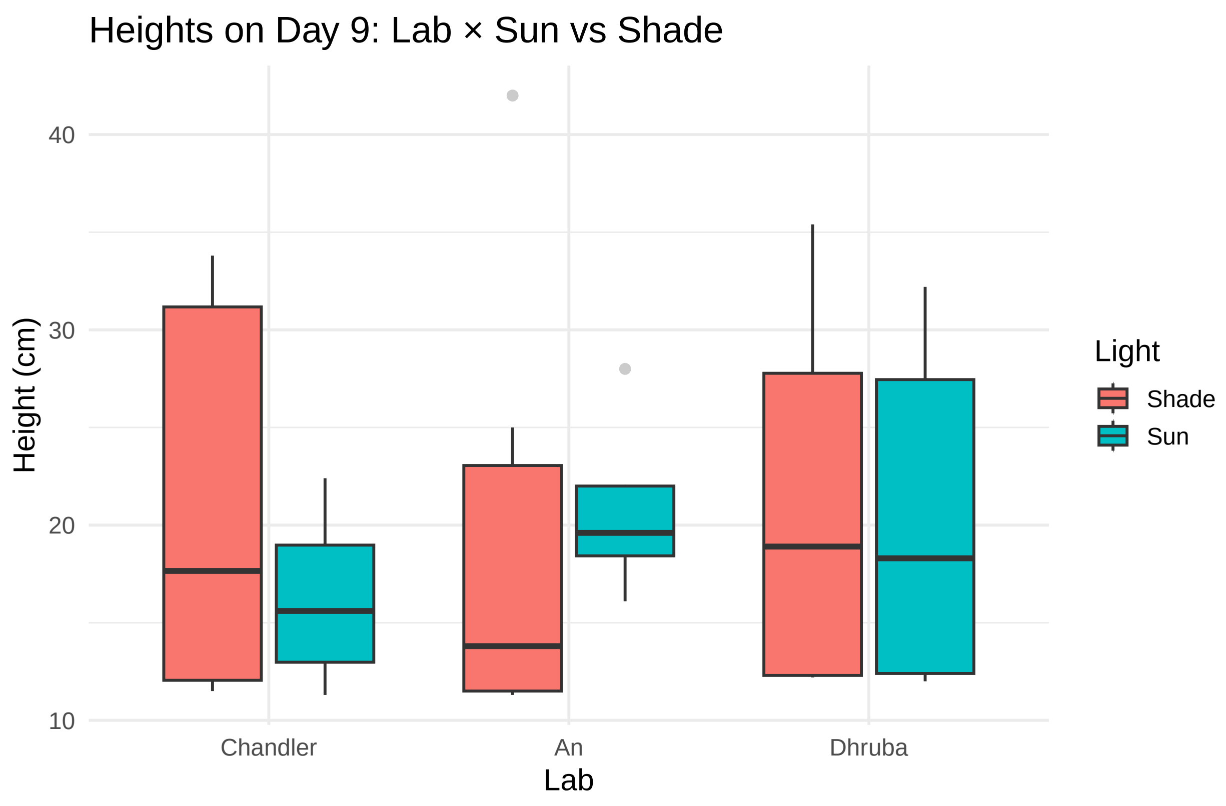
```r
print(plot_water_day(df,    "day_9_len"))
```



```r
figs <- plot_day_labXwater_sunshade(df, "day_9_len")
print(figs$water_plot)
```



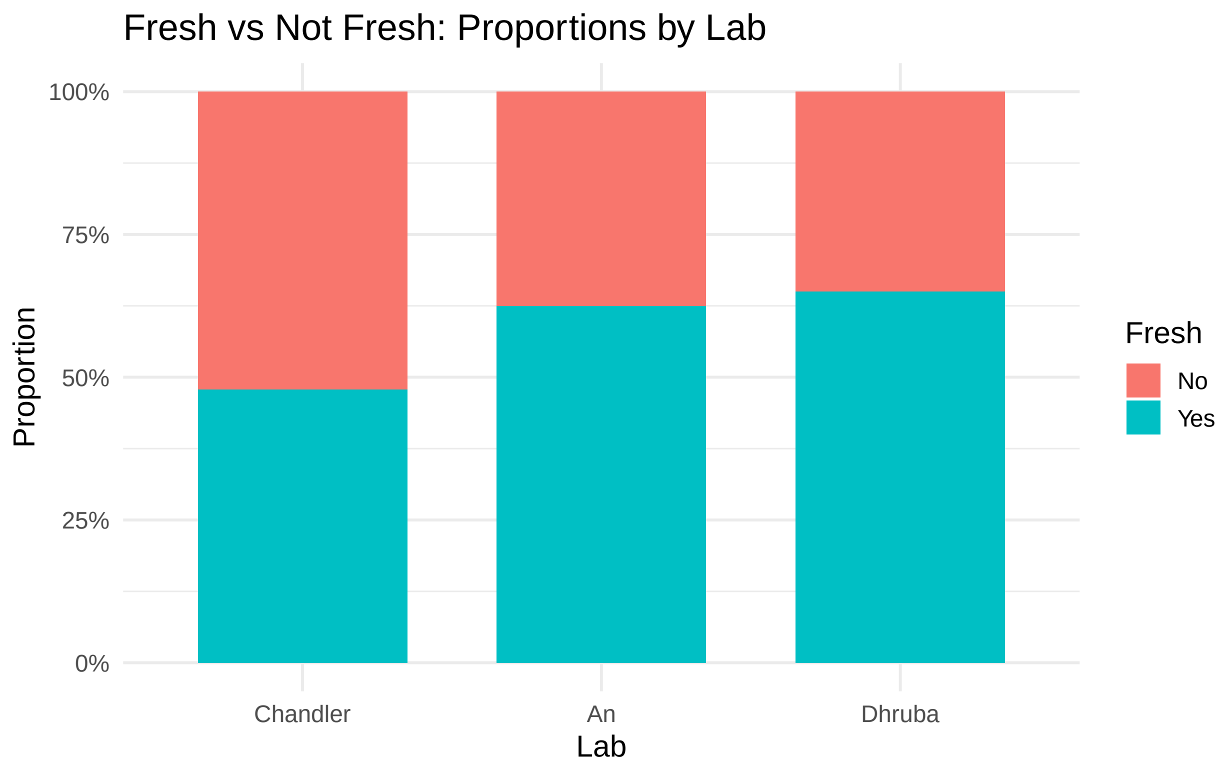
```r
print(figs$sunshade_plot)
```



Interpretation. Across all four plots, watering emerges as the clearest and most consistent driver of plant height on Day 9: watered plants are markedly taller than non-watered plants in every lab, with little overlap in their distributions. In contrast, the Sun vs Shade comparison shows a weaker and less stable pattern, with the direction and magnitude of the light effect varying across labs. The lab-wise plots further show that absolute height levels differ by lab, and the effect of light is not uniform—indicating a potential interaction between lab conditions and light treatment. Overall, the results suggest that watering has a strong and biologically robust effect on growth, whereas the influence of light is weaker.

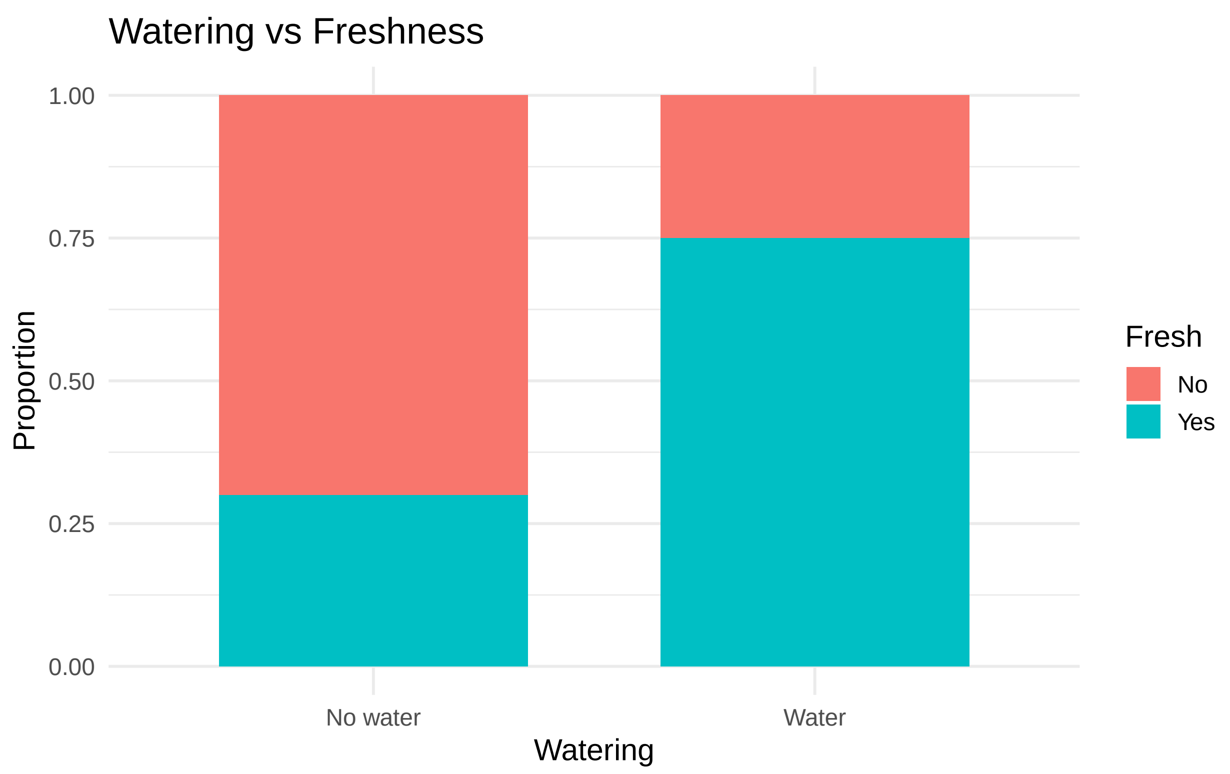
3.6. Freshness: distribution & associations

```r
# Unified freshness field preferring computed when available
df <- df %>%
  mutate(fresh_use = dplyr::coalesce(as.character(freshness_calc), as.character(freshness))) %>%
  mutate(fresh_use = factor(fresh_use, levels = c("No","Yes")))

# Proportions by lab
ggplot(df, aes(x = lab, fill = fresh_use)) +
  geom_bar(width = 0.7, position = "fill") +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(title = "Fresh vs Not Fresh: Proportions by Lab",
       x = "Lab", y = "Proportion", fill = "Fresh") +
  theme_minimal(base_size = 14)
```

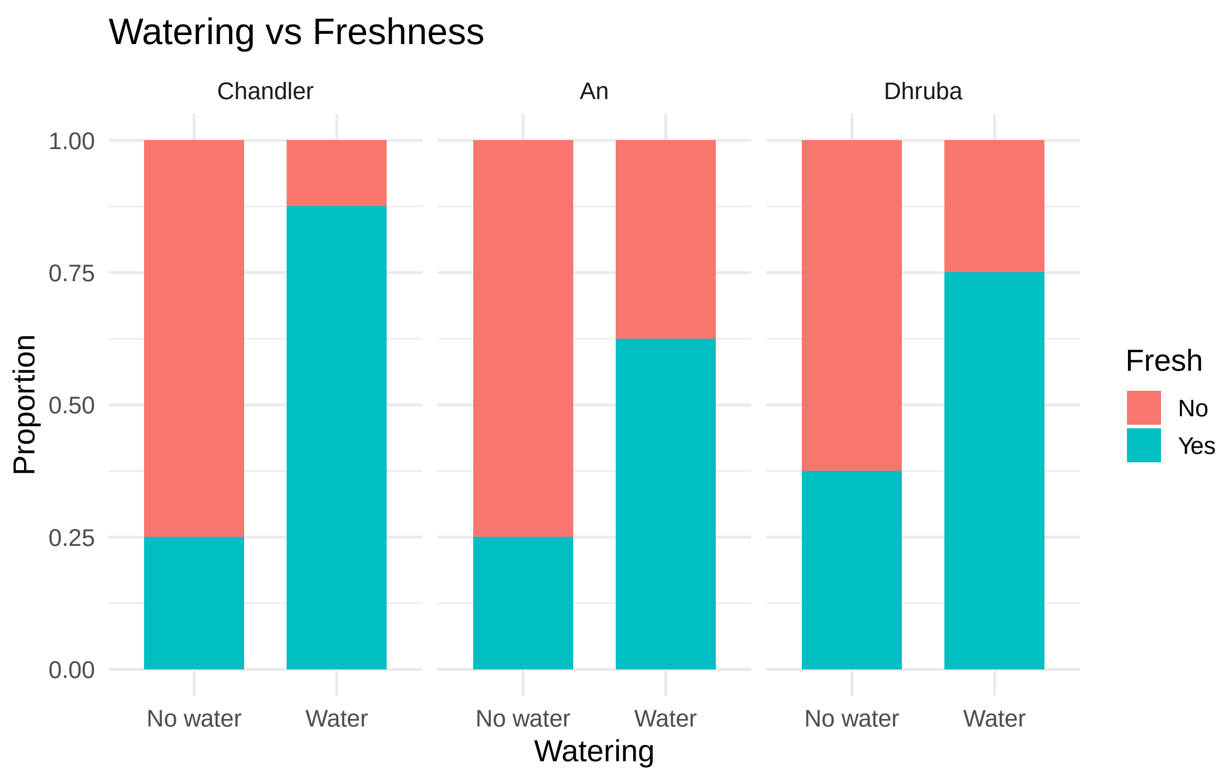


Interpretation Across labs, the proportion of plants classified as “fresh” varies, with Dhruba showing the highest share of fresh samples and Chandler the lowest, suggesting differences in growing conditions or handling across locations.

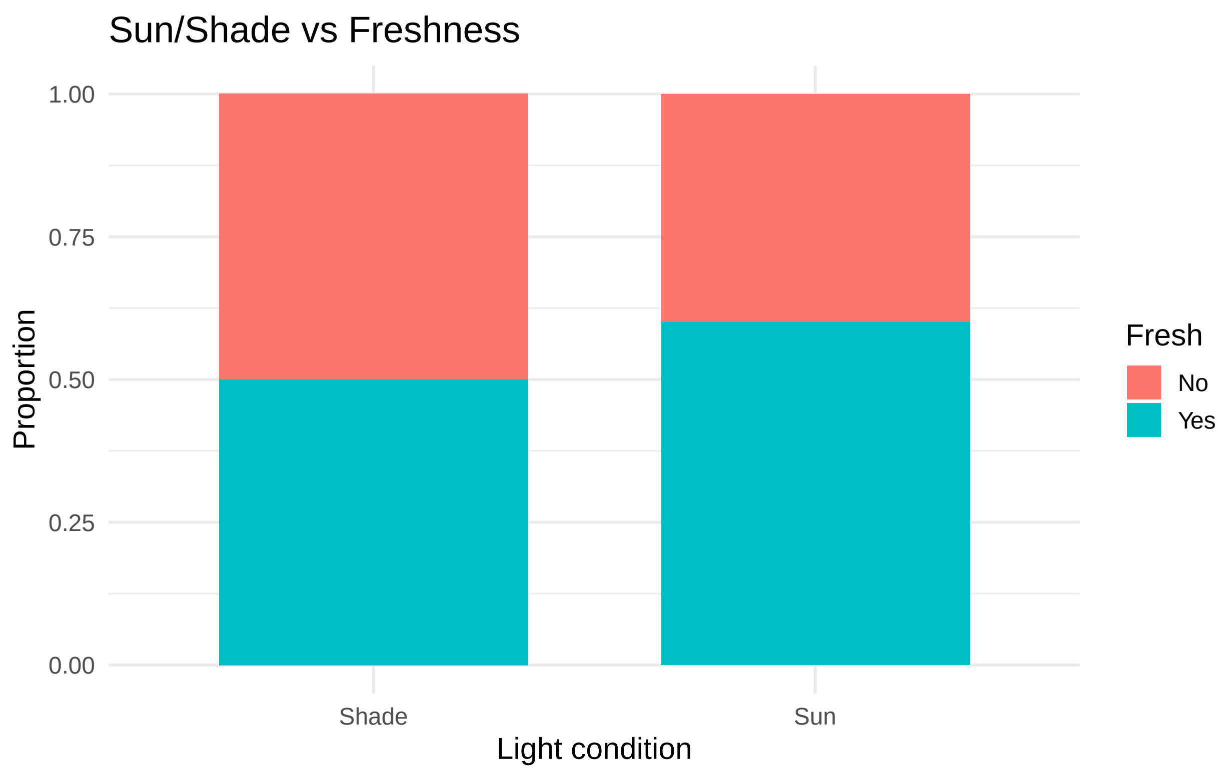
```r
.fresh_use <- function(df) {
  if ("freshness_calc" %in% names(df)) {
    factor(dplyr::coalesce(as.character(df$freshness_calc), as.character(df$freshness)),
           levels = c("No","Yes"))
  } else if ("freshness" %in% names(df)) {
    factor(as.character(df$freshness), levels = c("No","Yes"))
  } else stop("df must contain 'freshness_calc' or 'freshness'.")
}

# 1) Sun/Shade vs Freshness (exclude GL and FN)
plot_sunshade_vs_freshness <- function(df, facet_lab = FALSE, percent = TRUE) {
  stopifnot("label_grp" %in% names(df))
  dat <- df %>%
    mutate(
      freshness = .fresh_use(df),
      sun_shade = dplyr::case_when(
        label_grp %in% c("SN","SW") ~ "Sun",
        label_grp %in% c("DN","DW") ~ "Shade",
        TRUE ~ NA_character_
      )
    ) %>%
    filter(!is.na(sun_shade), !is.na(freshness))

  p <- ggplot(dat, aes(x = sun_shade, fill = freshness)) +
    geom_bar(position = if (percent) "fill" else "stack", width = 0.7) +
    labs(
      title = "Sun/Shade vs Freshness",
      x = "Light condition", y = if (percent) "Proportion" else "Count",
      fill = "Fresh"
    ) +
    theme_minimal(base_size = 14)

  if (facet_lab && "lab" %in% names(df)) p <- p + facet_wrap(~ lab, nrow = 1)
  p
}

# 2) Water vs No Water vs Freshness (exclude GL)
plot_water_vs_freshness <- function(df, facet_lab = FALSE, percent = TRUE) {
  stopifnot("label_grp" %in% names(df))
  dat <- df %>%
    mutate(
      freshness = .fresh_use(df),
      water = dplyr::case_when(
        label_grp %in% c("DW","SW") ~ "Water",
        label_grp %in% c("DN","SN") ~ "No water",
        TRUE ~ NA_character_
      )
    ) %>%
    filter(!is.na(water), !is.na(freshness))

  p <- ggplot(dat, aes(x = water, fill = freshness)) +
    geom_bar(position = if (percent) "fill" else "stack", width = 0.7) +
    labs(
      title = "Watering vs Freshness",
      x = "Watering", y = if (percent) "Proportion" else "Count",
      fill = "Fresh"
    ) +
    theme_minimal(base_size = 14)

  if (facet_lab && "lab" %in% names(df)) p <- p + facet_wrap(~ lab, nrow = 1)
  p
}

print(plot_water_vs_freshness(df))
```



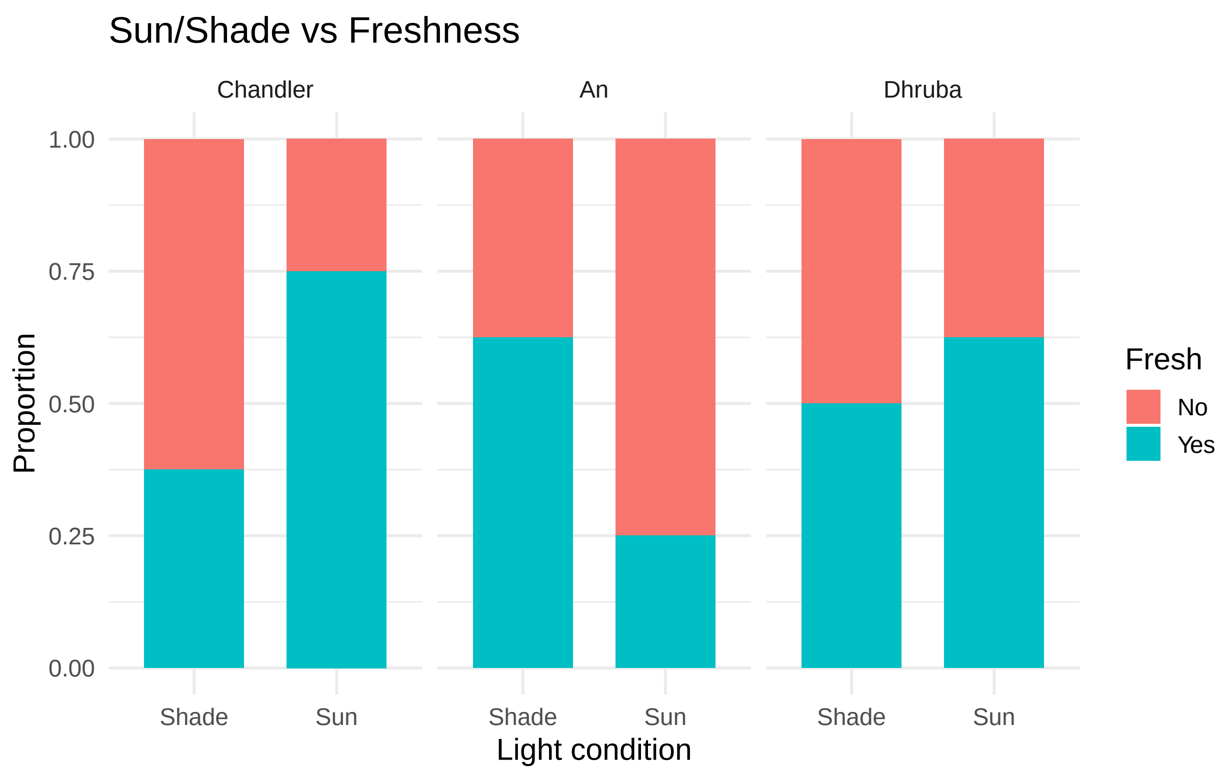
```
print(plot_water_vs_freshness(df, facet_lab = TRUE))
```



```r
print(plot_sunshade_vs_freshness(df))
```

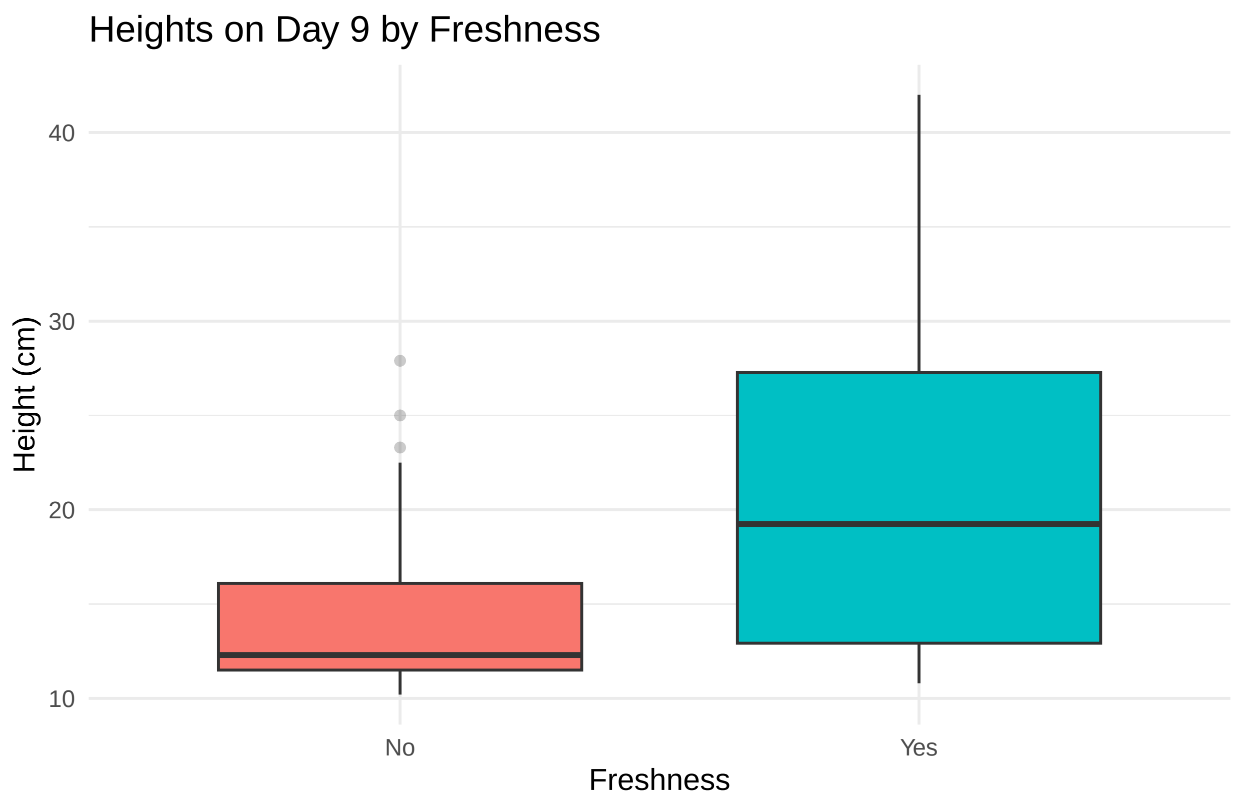


```r
print(plot_sunshade_vs_freshness(df, facet_lab = TRUE))
```

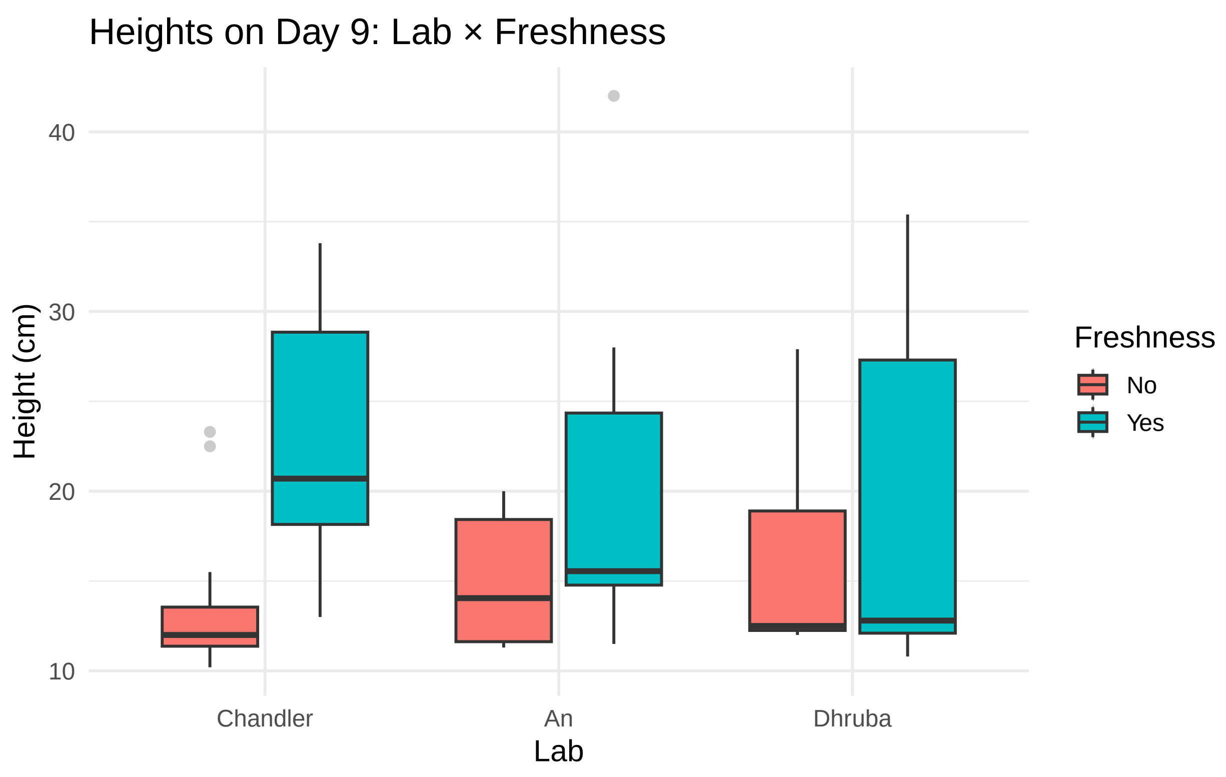


Interpretation Watering has a clear and consistent relationship with freshness: across all labs, the proportion of "fresh" plants is much higher in the watered group than in the no-water group, indicating that water availability is the main driver of whether a plant satisfies the visual freshness criteria. In contrast, the Sun vs Shade comparison does not show a strong overall difference in freshness, and the lab-wise breakdown reveals that the direction and magnitude of the light effect varies by location. Notably, in case of **An** the Sun–Shade height relationship appears **reversed** relative to the other labs—likely because An did not have all subgroups represented under Sun, which can shift the apparent distribution.This suggests that while watering has a dominant and fairly uniform influence on plant quality, the effect of light is more context-dependent and may interact with lab-specific growing conditions.

```r
# Boxplot of height vs freshness for a given day column
plot_freshness_day <- function(df, day_col) {
  stopifnot(is.character(day_col), length(day_col) == 1)
  if (!day_col %in% names(df)) stop(sprintf("Column '%s' not found in df.", day_col))

  dat <- df %>%
    mutate(
      fresh_use = dplyr::coalesce(as.character(freshness_calc), as.character(freshness)),
      fresh_use = factor(fresh_use, levels = c("No", "Yes"))
    ) %>%
    transmute(
      freshness = fresh_use,
      height_cm = .data[[day_col]]
    ) %>%
    filter(!is.na(freshness), !is.na(height_cm))
  day_no <- stringr::str_to_title(gsub("_len$", "", day_col))
  day_no <- gsub("day_(\\d+)", "Day \\1", tolower(day_no), perl = TRUE)

  ggplot(dat, aes(x = freshness, y = height_cm, fill = freshness)) +
    geom_boxplot(width = 0.7, outlier.alpha = 0.25) +
    labs(
      title = sprintf("Heights on %s by Freshness", day_no),
      x = "Freshness",
      y = "Height (cm)"
    ) +
    theme_minimal(base_size = 14) +
    theme(legend.position = "none")
}

print(plot_freshness_day(df, "day_9_len"))
```



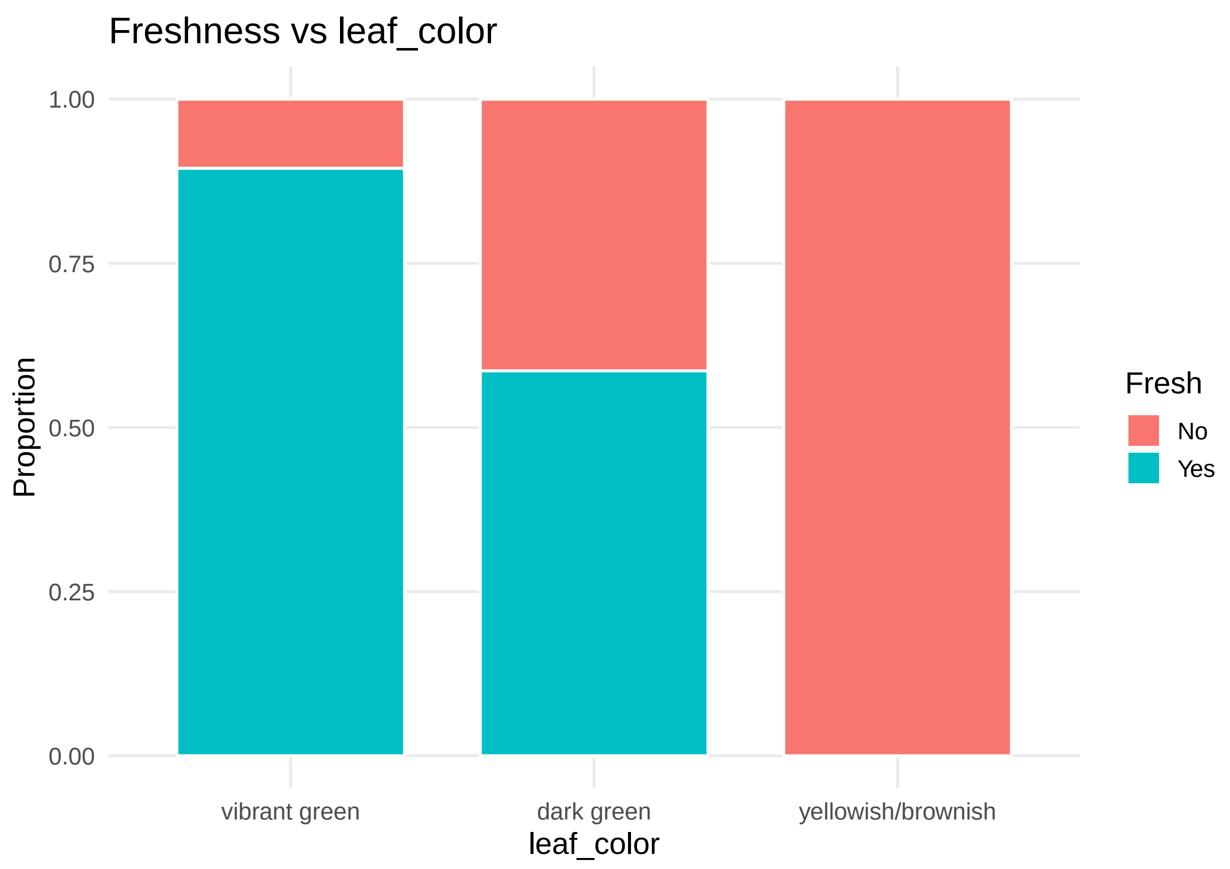
```
# Lab × Freshness height boxplots for a given day column
plot_lab_freshness_day <- function(df, day_col, layout = c("dodge","facet")) {
  layout <- match.arg(layout)
  stopifnot(is.character(day_col), length(day_col) == 1)
  if (!day_col %in% names(df)) stop(sprintf("Column '%s' not found in df.", day_col))
  if (!("lab" %in% names(df))) stop("df must contain 'lab'.")

  dat <- df %>%
    mutate(
      fresh_use = dplyr::coalesce(as.character(freshness_calc), as.character(freshness)),
      fresh_use = factor(fresh_use, levels = c("No","Yes"))
    ) %>%
    transmute(
      lab,
      freshness = fresh_use,
      height_cm = .data[[day_col]]
    ) %>%
    filter(!is.na(lab), !is.na(freshness), !is.na(height_cm))
  day_no <- str_to_title(gsub("_len$", "", day_col))
  day_no <- gsub("day_(\\d+)", "Day \\1", tolower(day_no), perl = TRUE)
  if (layout == "dodge") {
    # one plot; x = lab, grouped by freshness
    ggplot(dat, aes(x = lab, y = height_cm, fill = freshness)) +
      geom_boxplot(position = position_dodge(width = 0.75), width = 0.65, outlier.alpha = 0.25) +
      labs(
        title = sprintf("Heights on %s: Lab × Freshness", day_no),
        x = "Lab", y = "Height (cm)", fill = "Freshness"
      ) +
      theme_minimal(base_size = 14)
  } else {
    # facet: one panel per lab; x = freshness
    ggplot(dat, aes(x = freshness, y = height_cm, fill = freshness)) +
      geom_boxplot(width = 0.7, outlier.alpha = 0.25) +
      facet_wrap(~ lab, nrow = 1) +
      labs(
        title = sprintf("Heights on %s by Freshness (Faceted by Lab)", day_no),
        x = "Freshness", y = "Height (cm)"
      ) +
      theme_minimal(base_size = 14) +
      theme(legend.position = "none")
  }
}

print(plot_lab_freshness_day(df, "day_9_len"))
```

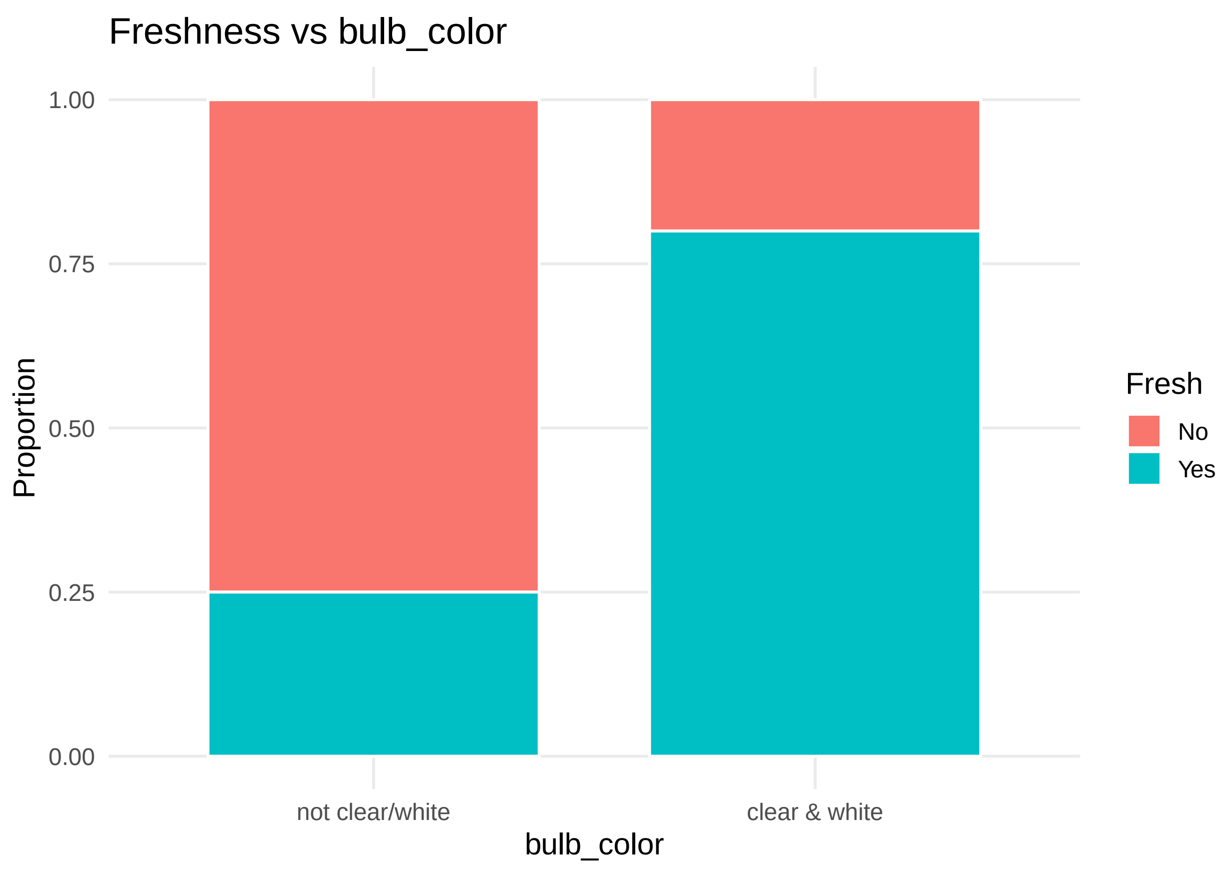


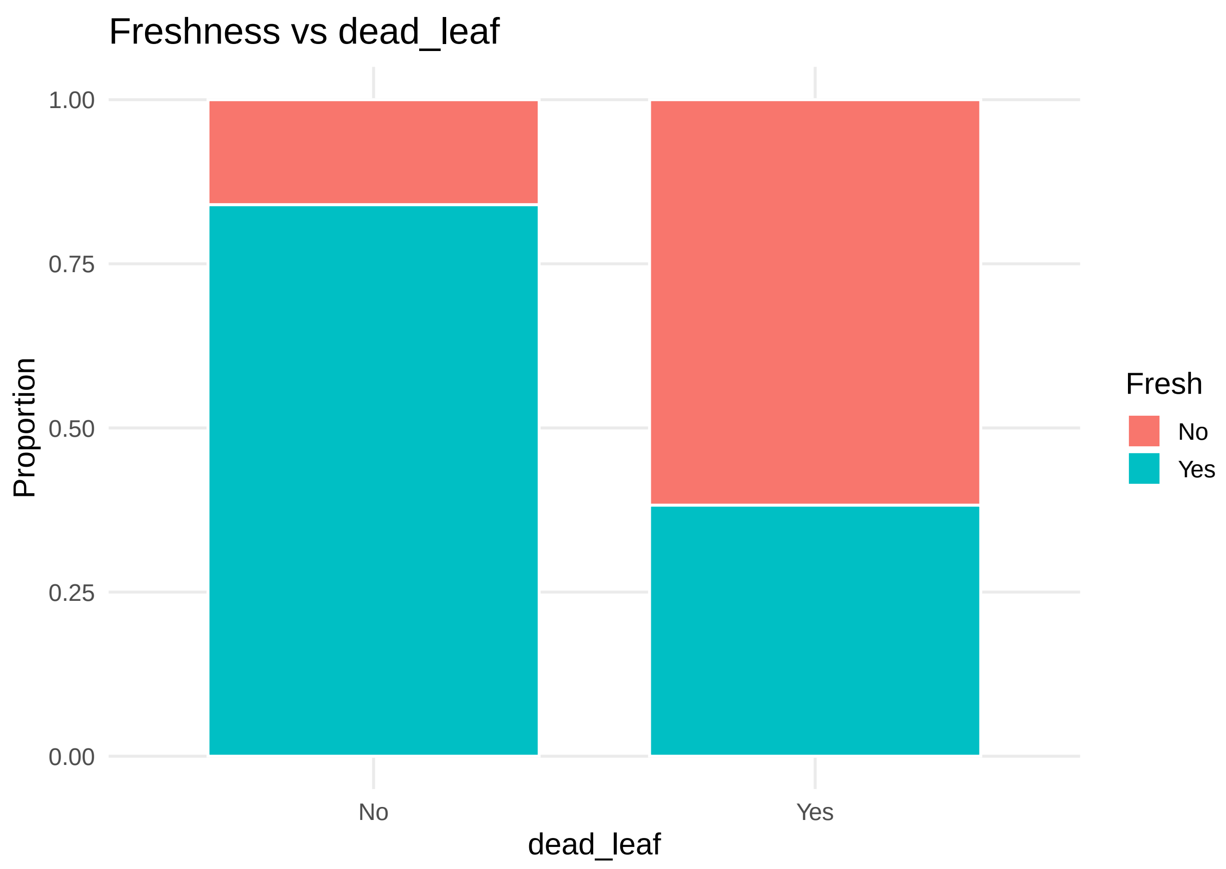
Interpretation The boxplot comparing Day 9 heights by freshness shows a clear separation: fresh plants are noticeably taller on average and have a wider range of growth, while non-fresh plants cluster at lower heights with little variation. This confirms that the rule-based freshness measure aligns well with actual plant vigor, as taller and healthier plants are more likely to satisfy the visual criteria used to define freshness. Across all three labs, plants labeled fresh tend to be taller on Day 9 than those labeled not fresh, but the separation between the two groups varies by lab. In case of Chandler and An, the height difference between fresh and non-fresh plants is large and visually distinct, while in Dhruba the two groups overlap more, suggesting weaker discrimination there.

Freshness vs individual components

```r
plot_pct_stack_fresh_y <- function(df, var) {
  var_sym <- rlang::ensym(var)

  dat <- df %>%
    mutate(
      freshness = dplyr::coalesce(as.character(freshness_calc), as.character(freshness)),
      freshness = factor(freshness, levels = c("No","Yes")),
      !!var_sym := forcats::fct_explicit_na(factor(!!var_sym), na_level = "(Missing)")
    ) %>%
    filter(!is.na(freshness))

  ggplot(dat, aes(x = !!var_sym, fill = freshness)) +
    geom_bar(position = "fill", width = 0.75, color = "white") +
    labs(
      title = paste("Freshness vs", rlang::as_label(var_sym)),
      x = rlang::as_label(var_sym),
      y = "Proportion",
      fill = "Fresh"
    ) +
    theme_minimal(base_size = 14) +
    theme(panel.grid.minor = element_blank())
}

p_leaf  <- plot_pct_stack_fresh_y(df, leaf_color)
p_bulb  <- plot_pct_stack_fresh_y(df, bulb_color)
p_dead  <- plot_pct_stack_fresh_y(df, dead_leaf)

print(p_leaf); print(p_bulb); print(p_dead)
```




Freshness vs bulb_color
1.00
0.75
0.50
0.25
0.00
Proportion
not clear/white
clear & white
bulb_color
Fresh
No
Yes



3.7. Count tables for freshness factors

```r
fresh_counts <- df %>%
  count(fresh_use, name = "n") %>%
  mutate(prop = scales::percent(n / sum(n)))

leaf_counts <- df %>%
  count(leaf_color, name = "n") %>%
  mutate(prop = scales::percent(n / sum(n)))

bulb_counts <- df %>%
  count(bulb_color, name = "n") %>%
  mutate(prop = scales::percent(n / sum(n)))

dead_counts <- df %>%
  count(dead_leaf, name = "n") %>%
  mutate(prop = scales::percent(n / sum(n)))

# Print nice tables
print(knitr::kable(fresh_counts, caption = "Freshness counts"))
```

```
##
```

```
##
## Table: Freshness counts
##
## |fresh_use |  n|prop |
## |:---------|--:|:----|
## |No        | 25|42%  |
## |Yes       | 34|58%  |
```

```
print(knitr::kable(leaf_counts,  caption = "Leaf color counts"))
```

```
##
##
## Table: Leaf color counts
##
## |leaf_color         |  n|prop |
## |:------------------|--:|:----|
## |vibrant green      | 19|32%  |
## |dark green         | 29|49%  |
## |yellowish/brownish | 11|19%  |
```

```
print(knitr::kable(bulb_counts,  caption = "Bulb color counts"))
```

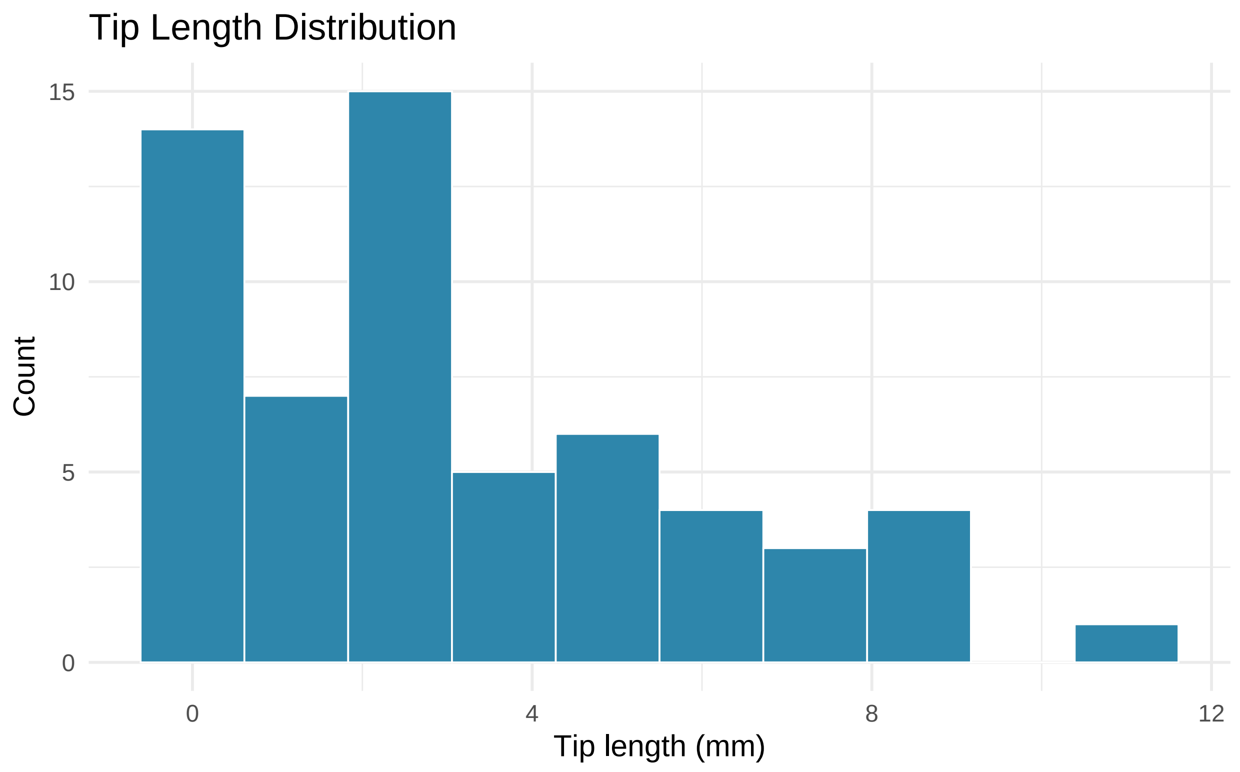
```
##
##
## Table: Bulb color counts
##
## |bulb_color      |  n|prop |
## |:---------------|--:|:----|
## |not clear/white | 24|41%  |
## |clear & white   | 35|59%  |
```

```
print(knitr::kable(dead_counts,  caption = "Dead leaf counts"))
```

```
##
##
## Table: Dead leaf counts
##
## |dead_leaf |  n|prop |
## |:---------|--:|:----|
## |No        | 25|42%  |
## |Yes       | 34|58%  |
```

3.8. Distribution of Tip length

```
df$tip <- suppressWarnings(as.numeric(df$tip))
ggplot(df %>% filter(!is.na(tip)), aes(x = tip)) +
  geom_histogram(bins = 10, boundary = 0, closed = "left",
                 fill = "#2E86AB", color = "white", linewidth = 0.4) +
  labs(title = "Tip Length Distribution", x = "Tip length (mm)", y = "Count") +
  theme_minimal(base_size = 14)
```



3.9. Key takeaways

- **Variance structure:** Lab-wise SD ratio = **2.04**; evidence of heteroskedasticity across labs.

- **Watering vs Light effects:** Watering has a large and consistent impact on growth across labs, with watered plants clearly taller than non-watered plants in every location. In contrast, the Sun vs Shade comparison shows a weaker and more lab-dependent pattern, with the direction and magnitude of the light effect varying across Chandler, An, and Dhruba. This indicates that water is the dominant growth driver.
- **Freshness:** Patterns of height and component criteria (leaf/bulb/dead-leaf) generally align with the rule-based freshness measure.
- **Lab-specific conditions:** Differences in cell counts are due to experimental logistics rather than missing data—e.g., only Chandler ran the Grow Light condition, and An had limited onions for all treatment combinations. These structural differences should be accounted for when comparing treatments across labs.

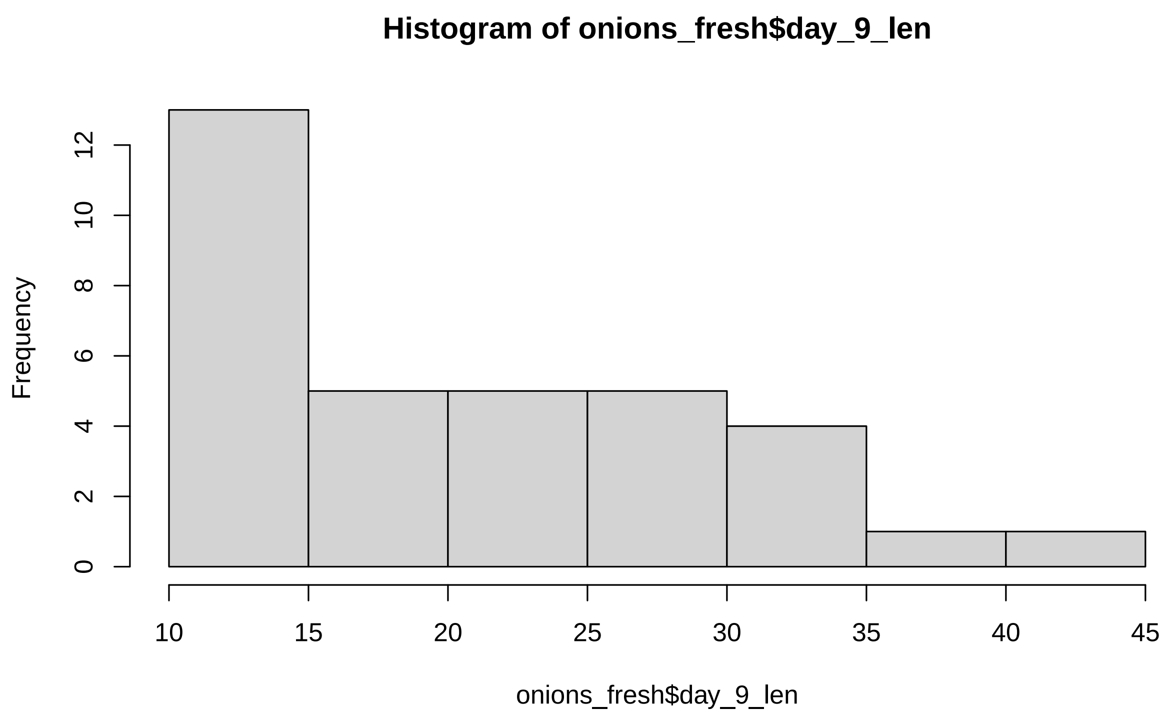
---

4. Freshness Permutation Tests

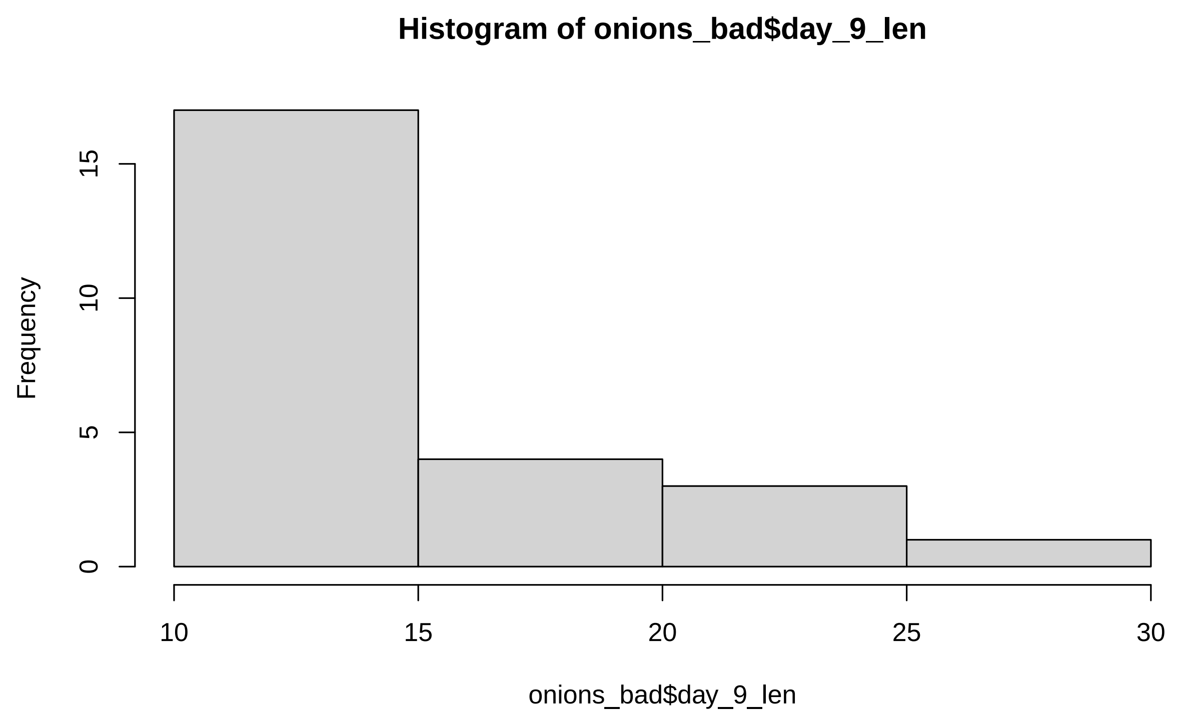
4.1. EDA

Separating data into fresh/not fresh:

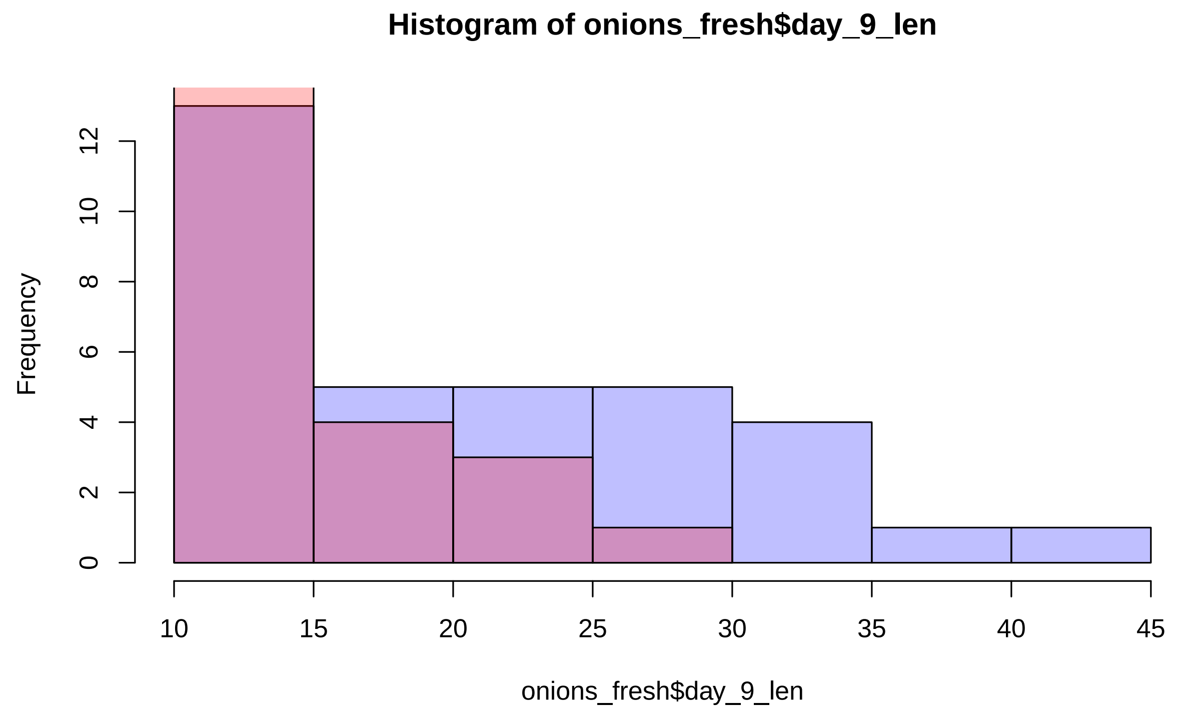
```r
onions_fresh <- proj3_data[proj3_data["freshness"] == 1, ]
onions_bad <- proj3_data[proj3_data["freshness"] == 0, ]
p1 <- hist(onions_fresh$day_9_len)
```



```r
p2 <- hist(onions_bad$day_9_len)
```



```
plot(p1, col = rgb(0, 0, 1, 1/4))
plot(p2, col = rgb(1, 0, 0, 1/4), add=T)
```



4.2. Difference in Proportions Test - Water Effect

Consider the statistic

$$\text{diff}_s = \text{Prop}(\text{fresh} = 1 \mid \text{water} = 1, \text{sunlight} = s) - \text{Prop}(\text{fresh} = 1 \mid \text{water} = 0, \text{sunlight} = s)$$

Performing difference in proportions test while using the statistic

$$T_{obs} = \frac{n_{\text{dark}} \cdot \text{diff}_{\text{dark}} + n_{\text{sun}} \cdot \text{diff}_{\text{sun}}}{n_{\text{dark}} + n_{\text{sun}}}$$

I will also pool labs. First, I will one-hot-encode the sunlight/water variables. We have the following:

```r
# Keeping rows only with SW/SN/DW/DN
experimental_condition <- grepl("^[SD][WN][0-9]+$", proj3_data$label)
df_exp <- proj3_data[experimental_condition, , drop = FALSE]
df_exp$sunlight <- ifelse(substr(df_exp$label, 1, 1) == "S", 1, 0)
df_exp$water <- ifelse(substr(df_exp$label, 2, 2) == "W", 1, 0)
```

Next, I will define a function to compute the water effect controlling for sunlight:

```r
water_effect_stat <- function(input_data) {

  # Calculate difference in proportion for each sunlight group
  sun_levels <- sort(unique(input_data$sunlight))
  diffs <- numeric(length(sun_levels))
  wts <- numeric(length(sun_levels))

  for (i in seq_along(sun_levels)) {
    s <- sun_levels[i]
    sub <- input_data[input_data$sunlight == s, , drop=FALSE]

    p_water <- mean(sub$freshness[sub$water == 1])
    p_nowater <- mean(sub$freshness[sub$water == 0])

    diffs[i] <- p_water - p_nowater
    wts[i] <- nrow(sub)

  }

  sum(diffs * wts) / sum(wts)
}

T_obs <- water_effect_stat(df_exp)
T_obs
```

```
## [1] 0.4393939
```

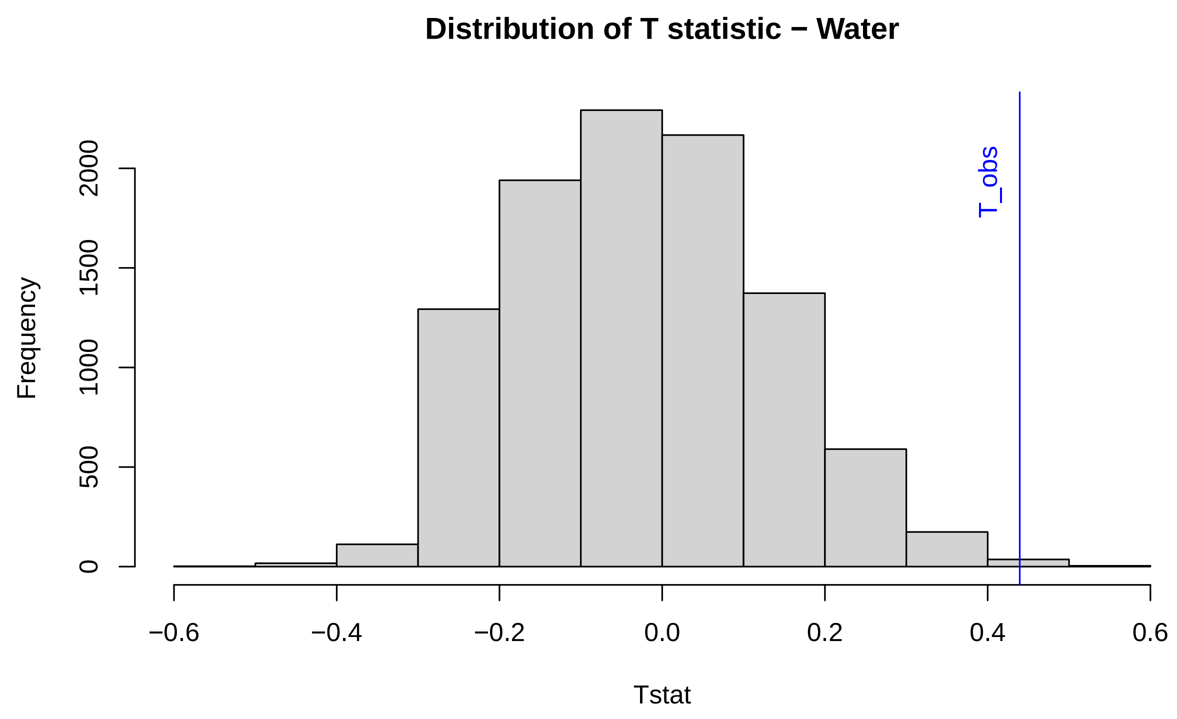
Now I can run the permutation test:

```r
set.seed(82803)
B <- 10000
T_perm <- numeric(B)

sun_levels <- sort(unique(df_exp$sunlight))

for (b in seq_len(B)) {
  dat_perm <- df_exp

  # Permute water inside each sunlight stratum
  for (s in sun_levels) {
    light_ind <- which(df_exp$sunlight == s)
    dat_perm$water[light_ind] <- sample(df_exp$water[light_ind])
  }

  T_perm[b] <- water_effect_stat(dat_perm)
}

# One sided p-value (water increases freshness)
p_one_sided <- (sum(T_perm >= T_obs) + 1) / (B + 1)
print(p_one_sided)
```

```
## [1] 0.00409959
```

The above is the one-sided p-value. We can also produce a histogram for the distribution of permuted statistics. This is illustrated below:

```r
hist(T_perm,
     main = "Distribution of T statistic - Water",
     xlab = "Tstat")
abline(v = T_obs, col="blue")
text(x = 0.3844848, y = 1750, labels = "T_obs", srt = 90, pos = 4, col = "blue")
```



4.3. Difference in Proportions Test - Sunlight Effect

We will now perform the same test as before, but now we will condition on the water and test whether sunlight was significant.

```r
sun_effect_stat <- function(input_data) {

  # Calculate difference in proportion for each water group
  water_levels <- sort(unique(input_data$water))
  diffs <- numeric(length(water_levels))
  wts <- numeric(length(water_levels))

  for (i in seq_along(water_levels)) {
    s <- water_levels[i]
    sub <- input_data[input_data$water == s, , drop=FALSE]

    p_sun <- mean(sub$freshness[sub$sunlight == 1])
    p_nosun <- mean(sub$freshness[sub$sunlight == 0])

    diffs[i] <- p_sun - p_nosun
    wts[i] <- nrow(sub)

  }

  sum(diffs * wts) / sum(wts)
}
```

```r
T_obs <- sun_effect_stat(df_exp)
T_obs
```

```
## [1] 0.06060606
```

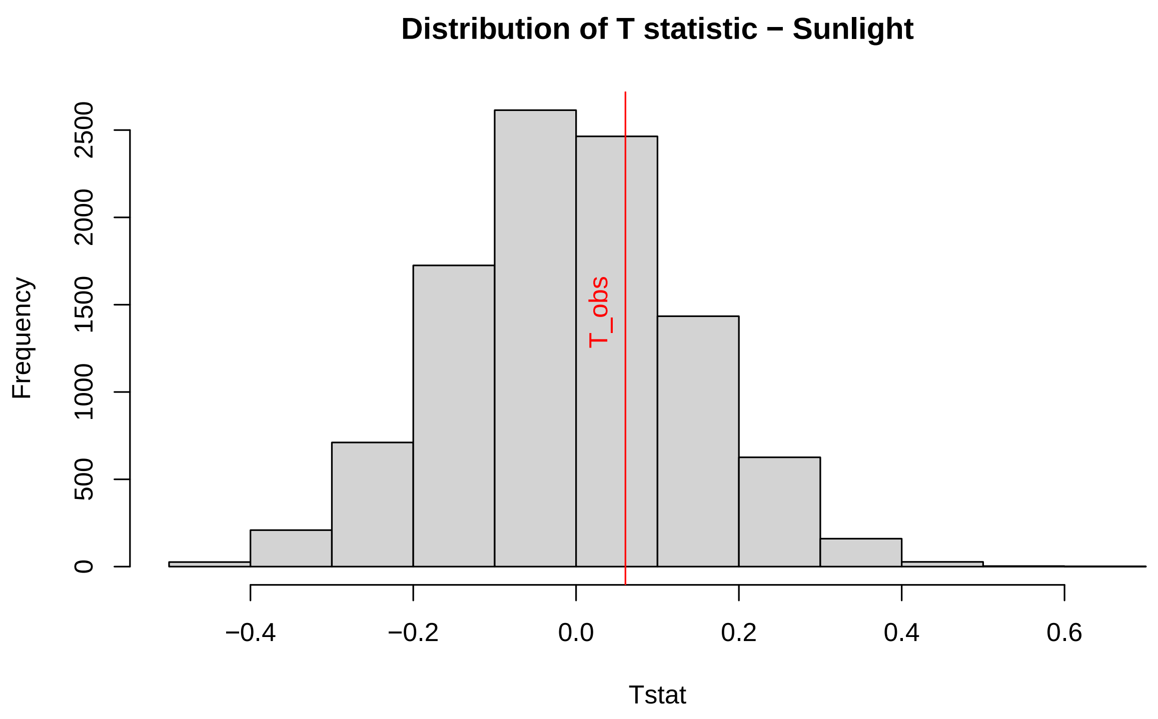
Running permutation test to assess effect of sunlight:

```r
set.seed(82803)
B <- 10000
T_perm <- numeric(B)

water_levels <- sort(unique(df_exp$water))

for (b in seq_len(B)) {
  dat_perm <- df_exp

  # Permute water inside each water stratum
  for (s in water_levels) {
    water_ind <- which(df_exp$water == s)
    dat_perm$sunlight[water_ind] <- sample(df_exp$sunlight[water_ind])
  }

  T_perm[b] <- sun_effect_stat(dat_perm)
}

# One sided p-value (sun increases freshness)
p_one_sided <- (sum(T_perm >= T_obs) + 1) / (B + 1)
print(p_one_sided)
```

```
## [1] 0.2568743
```

Evidently, sunlight is not associated with an increase in freshness. Once again, we produce a histogram illustrating the distribution of our statistic and our observed statistic value:

```r
hist(T_perm,
     main = "Distribution of T statistic - Sunlight",
     xlab = "Tstat")
abline(v = T_obs, col="red")
text(x = 0.0115152, y = 1250, labels = "T_obs", srt = 90, pos = 4, col = "red")
```



4.4. Difference in Proportions Test - Fridge/Darkness no water

Typically, people think it is best to store plants in the fridge as opposed to one of the experimental setups considered in this lab. Thus, as a final permutation test, we consider the difference in proportions between those green onions from the fridge with those from the seemingly best condition: darkness with no water.

We first subset the data for these labels:

```r
is_fridge_nowater   <- grepl("^FN[0-9]+$", proj3_data$label)
is_darkness_water <- grepl("^DW[0-9]+$", proj3_data$label)

df_final <- proj3_data[is_fridge_nowater | is_darkness_water, , drop = FALSE]

# Create a two-level group variable
df_final$group <- ifelse(grepl("^FN", df_final$label),
                        "fridge",
                        "dark_water")
```

Next, we can compute our observed difference in proportions

```r
mean_fridge <- mean(df_final$freshness[df_final$group == "fridge"])
mean_dark   <- mean(df_final$freshness[df_final$group == "dark_water"])

T_obs <- mean_dark - mean_fridge
T_obs
```

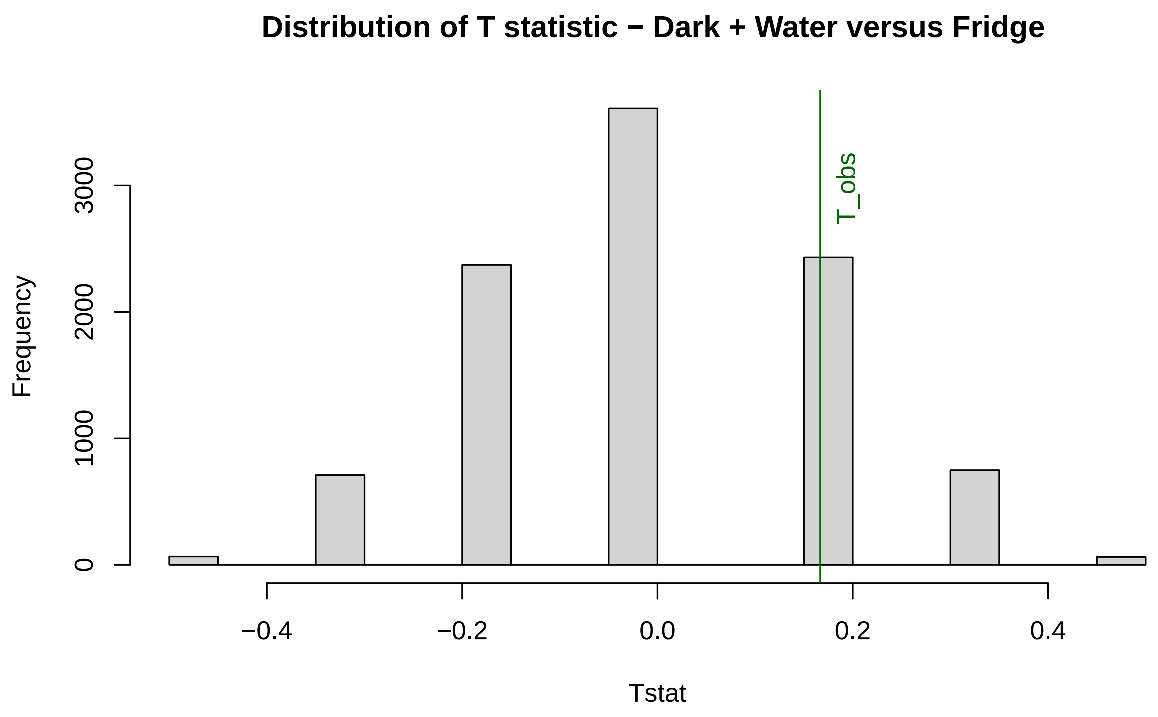
```
## [1] 0.1666667
```

```
set.seed(123)
B <- 10000
T_perm <- numeric(B)

for (b in seq_len(B)) {
  # randomly permute which samples are called "fridge" vs "dark_water"
  permuted_group <- sample(df_final$group)

  mean_fridge_b <- mean(df_final$freshness[permuted_group == "fridge"])
  mean_dark_b   <- mean(df_final$freshness[permuted_group == "dark_water"])

  T_perm[b] <- mean_dark_b - mean_fridge_b
}

p_one_sided <- (sum(T_perm >= T_obs) + 1) / (B + 1)
p_one_sided
```

```
## [1] 0.3243676
```

Our observed difference suggests that green onions left in the darkness with water are more fresh than those green onions in the fridge with no water. However, the p-value suggests that there is not a statistically significant difference between these groups.

```
hist(T_perm,
     main = "Distribution of T statistic - Dark + Water versus Fridge ",
     xlab = "Tstat")
abline(v = T_obs, col="darkgreen")
text(x = 0.18, y = 2690, labels = "T_obs", srt = 90, pos = 4, col = "darkgreen")
```



```r
cond_table <- aggregate(
  freshness ~ sunlight + water,
  data = df_exp,
  FUN = function(x) c(mean = mean(x), n = length(x))
)

# aggregate() packs mean and n into a matrix column; unpack it nicely:
cond_summary <- data.frame(
  sunlight = cond_table$sunlight,
  water    = cond_table$water,
  n        = cond_table$freshness[ , "n"],
  fresh_rate = cond_table$freshness[ , "mean"]
)

# Add readable labels
cond_summary$light_label <- ifelse(cond_summary$sunlight == 1, "Sunlight", "Darkness")
cond_summary$water_label <- ifelse(cond_summary$water == 1, "Water", "No Water")

cat("Freshness rates by condition:\n")
```

```
## Freshness rates by condition:
```

```r
for (i in seq_len(nrow(cond_summary))) {
  cat(sprintf(
    "%s, %s: n=%d, mean freshness=%.3f\n",
    cond_summary$light_label[i],
    cond_summary$water_label[i],
    cond_summary$n[i],
    cond_summary$fresh_rate[i]
  ))
}
```

```
## Darkness, No Water: n=12, mean freshness=0.167
## Sunlight, No Water: n=8, mean freshness=0.500
## Darkness, Water: n=12, mean freshness=0.833
## Sunlight, Water: n=12, mean freshness=0.667
```

Benjamini Hochberg for Multiple Tests

To determine significance, we run our Benjamini-Hochberg procedure to account for the fact that three different nulls were assessed. We have the following:

```r
p_values <- c(
  water_effect = 0.00409959,
  sunlight_effect = 0.2568743,
  dw_vs_fn = 0.3243676
)

# Apply Benjamini-Hochberg correction for FDR
alpha <- 0.05
```

```r
m <- length(p_values)
p_sorted <- sort(p_values)
ranks <- rank(p_values)

critical_values <- (1:m / m) * alpha

reject <- p_sorted <= critical_values

print(reject)
```

```
##    water_effect sunlight_effect        dw_vs_fn
##            TRUE           FALSE           FALSE
```

Thus, after correcting for the False Discovery Rate, we conclude that water did have a significant affect on the green onions after controlling for sunlight.

4.5 Controlling Site-Level Environment

As Dhruba identified in our EDA, there appear to be site-level effects. Indeed, Chandler's girlfriend's refrigerator broke recently, and we suspect that affected the results of the green onions in the fridge relative to Dhruba and An. We repeat the permutation tests above while additionally conditioning on lab.

Conditioning on Lab and Sunlight

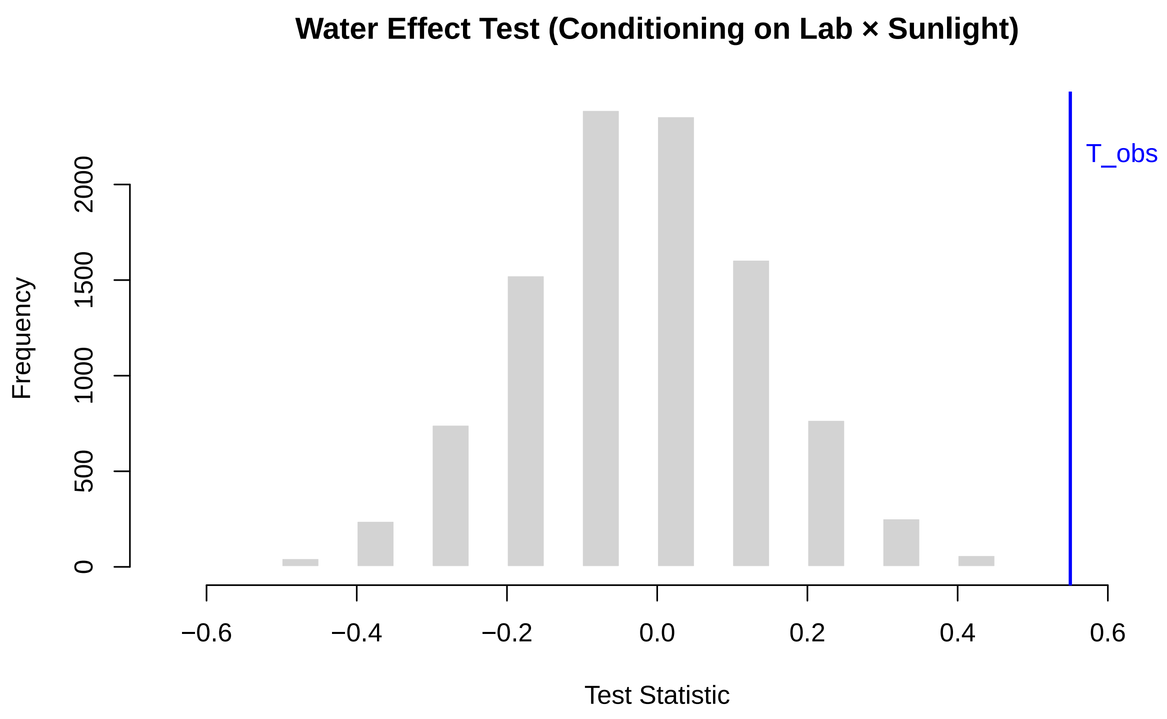
```r
# Load data and filter to experimental conditions
experimental_condition <- grepl("^[SD][WN][0-9]+$", proj3_data$label)
df_exp <- proj3_data[experimental_condition, , drop = FALSE]
df_exp$sunlight <- ifelse(substr(df_exp$label, 1, 1) == "S", 1, 0)
df_exp$water <- ifelse(substr(df_exp$label, 2, 2) == "W", 1, 0)

# Test statistic: weighted average of differences in proportions
# Now stratified by lab × sunlight
water_effect_stat_by_lab <- function(input_data) {
  # Create lab × sunlight strata
  input_data$stratum <- interaction(input_data$lab, input_data$sunlight, drop = TRUE)
  strata <- unique(input_data$stratum)

  diffs <- numeric(length(strata))
  wts <- numeric(length(strata))

  for (i in seq_along(strata)) {
    sub <- input_data[input_data$stratum == strata[i], , drop = FALSE]

    # Skip if stratum doesn't have both water levels
    if (length(unique(sub$water)) < 2) {
      diffs[i] <- NA
      wts[i] <- 0
      next
    }
    p_water <- mean(sub$freshness[sub$water == 1], na.rm = TRUE)
    p_nowater <- mean(sub$freshness[sub$water == 0], na.rm = TRUE)

    diffs[i] <- p_water - p_nowater
    wts[i] <- nrow(sub)
  }

  # Remove strata without both water levels
  valid <- !is.na(diffs) & wts > 0
  sum(diffs[valid] * wts[valid]) / sum(wts[valid])
}

# Observed test statistic
T_obs <- water_effect_stat_by_lab(df_exp)
cat("Observed T:", T_obs, "\n")
```

```
## Observed T: 0.55
```

```r
# Permutation test: permute water within each lab × sunlight stratum
set.seed(82803)
B <- 10000
T_perm <- numeric(B)

df_exp$stratum <- interaction(df_exp$lab, df_exp$sunlight, drop = TRUE)
strata_levels <- unique(df_exp$stratum)

for (b in seq_len(B)) {
  dat_perm <- df_exp

  # Permute water within each lab × sunlight stratum
  for (stratum in strata_levels) {
    stratum_ind <- which(df_exp$stratum == stratum)
    dat_perm$water[stratum_ind] <- sample(df_exp$water[stratum_ind])
  }

  T_perm[b] <- water_effect_stat_by_lab(dat_perm)
}

# One-sided p-value
p_value <- (sum(T_perm >= T_obs) + 1) / (B + 1)
cat("One-sided p-value:", p_value, "\n")
```

```
## One-sided p-value: 0.0009999
```

```r
# Plot
hist(T_perm, breaks = 30,
     main = "Water Effect Test (Conditioning on Lab × Sunlight)",
     xlab = "Test Statistic",
     col = "lightgray",
     border = "white")
abline(v = T_obs, col = "blue", lwd = 2)
text(x = T_obs, y = max(hist(T_perm, plot = FALSE)$counts) * 0.9,
     labels = paste0("T_obs"),
     pos = 4, col = "blue")
```



```
print(p_value)
```

```
## [1] 0.0009999
```

As before, we found that water is highly significant. We do the same while conditioning on water.

Conditioning on Lab and Water

```
experimental_condition <- grepl("^[SD][WN][0-9]+$", proj3_data$label)
df_exp <- proj3_data[experimental_condition, , drop = FALSE]
df_exp$sunlight <- ifelse(substr(df_exp$label, 1, 1) == "S", 1, 0)
df_exp$water <- ifelse(substr(df_exp$label, 2, 2) == "W", 1, 0)

# Test statistic: weighted average of differences in proportions
# Now stratified by lab × water
sun_effect_stat_by_lab <- function(input_data) {
  # Create lab × water strata
  input_data$stratum <- interaction(input_data$lab, input_data$water, drop = TRUE)
  strata <- sort(unique(input_data$stratum))

  valid_diffs <- c()
  valid_wts <- c()
```

```r
  for (stratum in strata) {
    sub <- input_data[input_data$stratum == stratum, , drop = FALSE]

    # Check if stratum has both sunlight levels
    sunlight_levels <- unique(sub$sunlight)
    if (length(sunlight_levels) < 2) {
      # Skip this stratum - cannot compute difference
      next
    }

    p_sun <- mean(sub$freshness[sub$sunlight == 1], na.rm = TRUE)
    p_nosun <- mean(sub$freshness[sub$sunlight == 0], na.rm = TRUE)

    valid_diffs <- c(valid_diffs, p_sun - p_nosun)
    valid_wts <- c(valid_wts, nrow(sub))
  }

  # Weighted average across valid strata only
  if (length(valid_diffs) == 0) return(NA)
  sum(valid_diffs * valid_wts) / sum(valid_wts)
}

# Observed test statistic
T_obs <- sun_effect_stat_by_lab(df_exp)
cat("Observed T:", T_obs, "\n")
```

```
## Observed T: 0.05
```

```r
# Print which strata are being used
cat("\nStrata analysis:\n")
```

```
##
## Strata analysis:
```

```r
df_exp$stratum <- interaction(df_exp$lab, df_exp$water, drop = TRUE)
for (stratum in sort(unique(df_exp$stratum))) {
  sub <- df_exp[df_exp$stratum == stratum, , drop = FALSE]
  sunlight_levels <- unique(sub$sunlight)
  lab_name <- sub$lab[1]
  water_name <- ifelse(sub$water[1] == 1, "Water", "NoWater")

  if (length(sunlight_levels) == 2) {
    cat(sprintf("    %s × %s: included (n=%d)\n", lab_name, water_name, nrow(sub)))
  } else {
    cat(sprintf("    %s × %s: EXCLUDED (only sunlight=%d, n=%d)\n",
                lab_name, water_name, sunlight_levels[1], nrow(sub)))
  }
}
```

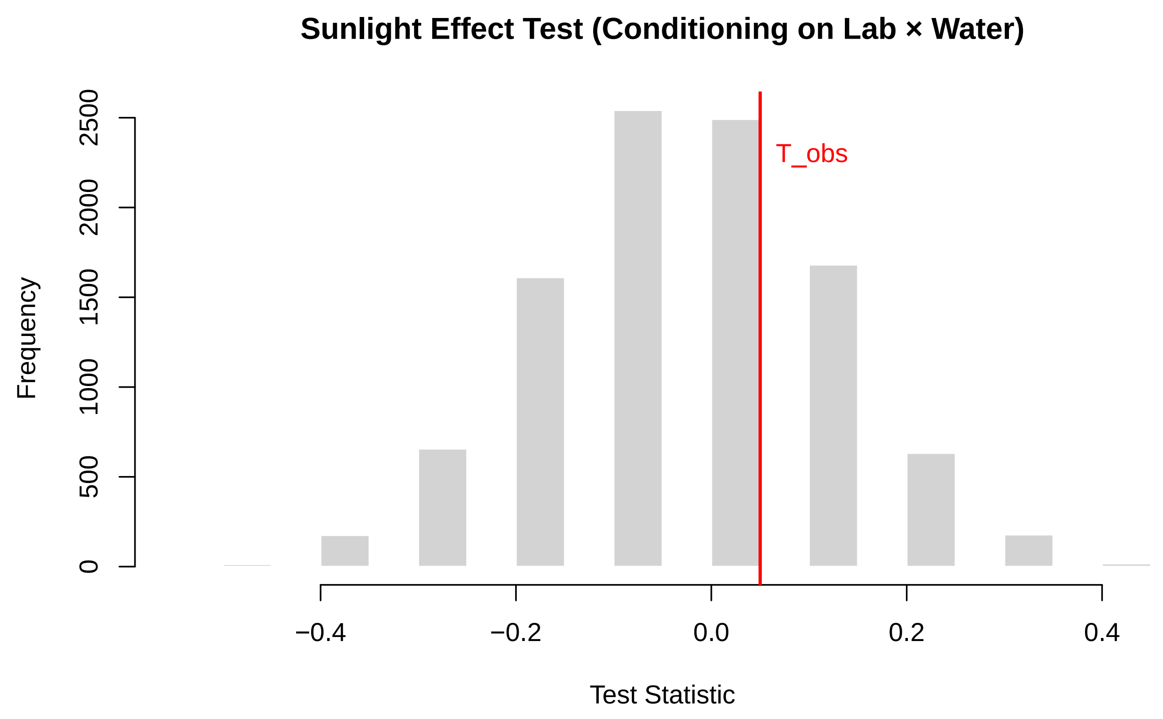
```
##     A × NoWater: EXCLUDED (only sunlight=0, n=4)
##     C × NoWater: included (n=8)
##     D × NoWater: included (n=8)
##     A × Water: included (n=8)
##     C × Water: included (n=8)
##     D × Water: included (n=8)
```

```r
cat("\n")
```

```r
# Permutation test: permute sunlight within each lab × water stratum
set.seed(82803)
B <- 10000
T_perm <- numeric(B)

strata_levels <- unique(df_exp$stratum)

for (b in seq_len(B)) {
  dat_perm <- df_exp

  # Permute sunlight within each lab × water stratum
  for (stratum in strata_levels) {
    stratum_ind <- which(df_exp$stratum == stratum)
    dat_perm$sunlight[stratum_ind] <- sample(df_exp$sunlight[stratum_ind])
  }

  T_perm[b] <- sun_effect_stat_by_lab(dat_perm)
}

# One-sided p-value
p_value <- (sum(T_perm >= T_obs) + 1) / (B + 1)
cat("One-sided p-value:", p_value, "\n")
```

```
## One-sided p-value: 0.50005
```

```r
# Plot
hist(T_perm, breaks = 30,
     main = "Sunlight Effect Test (Conditioning on Lab × Water)",
     xlab = "Test Statistic",
     col = "lightgray",
     border = "white")
abline(v = T_obs, col = "red", lwd = 2)
text(x = T_obs, y = max(hist(T_perm, plot = FALSE)$counts) * 0.9,
     labels = paste0("T_obs"),
     pos = 4, col = "red")
```



```
print(p_value)
```

```
## [1] 0.50005
```

As before, we found that sunlight was not significant after conditioning on lab and water provided. Finally, we repeat our analysis in comparing fridge/darkness without water while conditioning on lab.

Conditioning on Lab - Fridge/Darkness no water

We have the following:

```
is_DW <- grepl("^DW[0-9]+$", proj3_data$label)
is_FN <- grepl("^FN[0-9]+$", proj3_data$label)
df_dwfn <- proj3_data[is_DW | is_FN, , drop = FALSE]

# Create group variable: DW = "dark_water", FN = "fridge"
df_dwfn$group <- ifelse(grepl("^FN", df_dwfn$label), "fridge", "dark_water")

# Test statistic: weighted average of differences in proportions
# Now stratified by lab
dwfn_effect_stat_by_lab <- function(input_data) {
  labs <- sort(unique(input_data$lab))

  valid_diffs <- c()
  valid_wts <- c()

  for (lab in labs) {
    sub <- input_data[input_data$lab == lab, , drop = FALSE]

    # Check if lab has both groups
    groups <- unique(sub$group)
    if (length(groups) < 2) {
      # Skip this lab - cannot compute difference
      next
    }

    p_dark <- mean(sub$freshness[sub$group == "dark_water"], na.rm = TRUE)
    p_fridge <- mean(sub$freshness[sub$group == "fridge"], na.rm = TRUE)

    valid_diffs <- c(valid_diffs, p_dark - p_fridge)
    valid_wts <- c(valid_wts, nrow(sub))
  }

  # Weighted average across valid labs only
  if (length(valid_diffs) == 0) return(NA)
  sum(valid_diffs * valid_wts) / sum(valid_wts)
}

# Observed test statistic
T_obs <- dwfn_effect_stat_by_lab(df_dwfn)
cat("Observed T:", T_obs, "\n")
```

```
## Observed T: 0.1666667
```

```
# Print which labs are being used
cat("\nLab analysis:\n")
```

```
##
## Lab analysis:
```

```
for (lab in sort(unique(df_dwfn$lab))) {
  sub <- df_dwfn[df_dwfn$lab == lab, , drop = FALSE]
  groups <- unique(sub$group)

  if (length(groups) == 2) {
    n_dw <- sum(sub$group == "dark_water")
    n_fn <- sum(sub$group == "fridge")
    cat(sprintf("    Lab %s: included (DW: n=%d, FN: n=%d)\n", lab, n_dw, n_fn))
  } else {
    cat(sprintf("    Lab %s: EXCLUDED (only %s, n=%d)\n",
                lab, groups[1], nrow(sub)))
  }
}
```

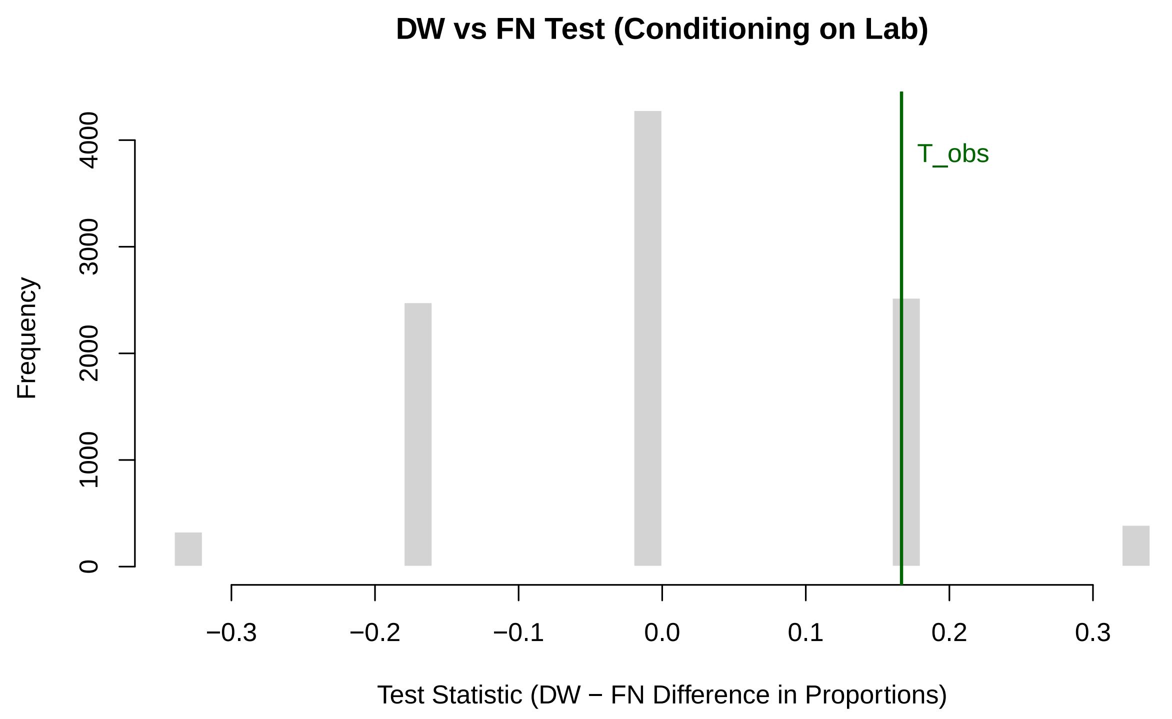
```
##     Lab A: included (DW: n=4, FN: n=4)
##     Lab C: included (DW: n=4, FN: n=4)
##     Lab D: included (DW: n=4, FN: n=4)
```

```r
cat("\n")
```

```r
# Permutation test: permute group within each lab
set.seed(123)
B <- 10000
T_perm <- numeric(B)

labs <- unique(df_dwfn$lab)

for (b in seq_len(B)) {
  dat_perm <- df_dwfn

  # Permute group within each lab
  for (lab in labs) {
    lab_ind <- which(df_dwfn$lab == lab)
    dat_perm$group[lab_ind] <- sample(df_dwfn$group[lab_ind])
  }

  T_perm[b] <- dwfn_effect_stat_by_lab(dat_perm)
}

# One-sided p-value (H1: DW > FN, i.e., darkness+water more fresh than fridge)
p_value <- (sum(T_perm >= T_obs) + 1) / (B + 1)
cat("One-sided p-value:", p_value, "\n")
```

```
## One-sided p-value: 0.2912709
```

```r
# Plot
hist(T_perm, breaks = 30,
     main = "DW vs FN Test (Conditioning on Lab)",
     xlab = "Test Statistic (DW - FN Difference in Proportions)",
     col = "lightgray",
     border = "white")
abline(v = T_obs, col = "darkgreen", lwd = 2)
text(x = T_obs, y = max(hist(T_perm, plot = FALSE)$counts) * 0.9,
     labels = paste0("T_obs"),
     pos = 4, col = "darkgreen")
```



```r
print(p_value)
```

```
## [1] 0.2912709
```

After conditioning on lab, we still found that there is no significant difference between the freshness of green onions that were grown in darkness with water versus those that were left in the fridge. Next, we proceed with permutation tests on the heights of our green onions.

Benjamini Hochberg for Multiple Tests

To determine significance, we run our Benjamini-Hochberg procedure to account for the fact that three different nulls were assessed. We have the following:

```r
p_values <- c(
  water_effect = 0.0009999,
  sunlight_effect = 0.50005,
  dw_vs_fn = 0.2912709
)

# Apply Benjamini-Hochberg correction for FDR
alpha <- 0.05

m <- length(p_values)
p_sorted <- sort(p_values)
ranks <- rank(p_values)

critical_values <- (1:m / m) * alpha
```

```
reject <- p_sorted <= critical_values

print(reject)
```

```
##    water_effect        dw_vs_fn sunlight_effect
##            TRUE           FALSE           FALSE
```

Thus, we conclude that water does have a significant effect on our gren onions after controlling for sunlight and lab.

5. Height Permutation Tests

In addition, we are also interested in the final (Day 9) length of the onions.

```
# Keep only SW/SN/DW/DN rows (pooling all labs)
experimental_condition <- grepl("^(SW|SN|DW|DN)", proj3_data$label)
df_exp <- proj3_data[experimental_condition, , drop = FALSE]

# One-hot style indicators aligned with partner's code
df_exp$sunlight <- ifelse(substr(df_exp$label, 1, 1) == "S", 1L, 0L)     # 1=Sun, 0=Shade
df_exp$water    <- ifelse(substr(df_exp$label, 2, 2) == "W", 1L, 0L)     # 1=Water, 0=NoWater

# Outcome
stopifnot("day_9_len" %in% names(df_exp))
```

5.1. EDA (Height)

Medians of `day_9_len` by cell:

```
df_exp$cell <- paste0(ifelse(df_exp$sunlight==1,"S","D"),
                      ifelse(df_exp$water==1,"W","N"))
tab_n <- table(df_exp$cell)
tab_med <- tapply(df_exp$day_9_len, df_exp$cell, median, na.rm = TRUE)
data.frame(cell = names(tab_n), n = as.integer(tab_n), median_day9 = as.numeric(tab_med))
```

```
##   cell  n median_day9
## 1   DN 12       12.05
## 2   DW 12       28.45
## 3   SN  8       12.65
## 4   SW 12       21.20
```

We ran three permutation tests for the final (Day 9) length of the onions:

- Assess potential difference between green onions that received water and those that did not while conditioning on sunlight.
- Assess potential difference between green onions that received sunlight and those that did not while conditioning on water.

- Assess potential difference between those that received water in the shade versus those that were placed in the fridge without water.

Let Y denote day-9 length (`day_9_len`). For a binary group indicator $G \in \{0, 1\}$ and a binary conditioning variable (stratum) $S \in \{0, 1\}$, define the within-stratum difference in medians

$$\Delta_s \;=\; \text{median}\{Y \mid G = 1,\, S = s\} \;-\; \text{median}\{Y \mid G = 0,\, S = s\}, \qquad s \in \{0, 1\}.$$

With n_s the number of observations in stratum s, the (observed) weighted statistic is

$$T_{\text{obs}} \;=\; \frac{\sum_s n_s\, \Delta_s}{\sum_s n_s}.$$

```r
# HELPERS

diff_in_medians <- function(y, g) {
  y <- as.numeric(y)
  g <- as.integer(g)
  median(y[g==1], na.rm=TRUE) - median(y[g==0], na.rm=TRUE)
}

weighted_strat_median_stat <- function(input_data, grp, strat, outcome) {
  levels_s <- sort(unique(input_data[[strat]]))
  diffs <- numeric(length(levels_s))
  wts   <- numeric(length(levels_s))
  for (i in seq_along(levels_s)) {
    s <- levels_s[i]
    sub <- input_data[input_data[[strat]] == s, , drop = FALSE]
    diffs[i] <- diff_in_medians(sub[[outcome]], sub[[grp]])
    wts[i]    <- nrow(sub)
  }
  sum(diffs * wts) / sum(wts)
}

perm_test_stratified_median <- function(input_data, grp, strat, outcome,
  B = 10000L, seed = 123, alternative = c("greater","less")) {
  alternative <- match.arg(alternative)
  set.seed(seed)

  T_obs <- weighted_strat_median_stat(input_data, grp, strat, outcome)

  T_perm <- numeric(B)
  strata_levels <- sort(unique(input_data[[strat]]))
  for (b in seq_len(B)) {
    dat_perm <- input_data
    for (s in strata_levels) {
      idx <- which(input_data[[strat]] == s)
      dat_perm[[grp]][idx] <- sample(input_data[[grp]][idx])
    }
    T_perm[b] <- weighted_strat_median_stat(dat_perm, grp, strat, outcome)
  }

  if (alternative == "greater") {
    p <- (sum(T_perm >= T_obs) + 1) / (B + 1)
  } else {
    p <- (sum(T_perm <= T_obs) + 1) / (B + 1)
  }

  list(T_obs = T_obs, T_perm = T_perm, p_value = p, alternative = alternative)
}
```

5.2. Test 1: Water Effect (conditioning on Sunlight)

- **Goal:** assess whether watering increases height.
- **Null H_0:** within each sunlight stratum, the median heights for watered and unwatered are equal.
- **Alternative H_1:** watered onions have larger medians (one-sided, "greater").
- **Statistic:** for each sunlight level s, set $G = \mathbb{1}\{\text{Water} = 1\}$, so $\Delta_s = \text{median}_{W=1} - \text{median}_{W=0}$; combine to T_{obs} as above.

Permutation procedure.
Within each sunlight stratum s, randomly permute the water labels (preserving group sizes) and recompute T to obtain $\{T_b\}_{b=1}^{B}$ under H_0.

One-sided p-value (right tail).

$$p = \frac{\#\{T_b \geq T_{\text{obs}}\} + 1}{B + 1}.$$

```r
df_exp$grp_water <- df_exp$water
df_exp$strat_sun <- df_exp$sunlight

set.seed(82803)
B <- 10000L
water_res <- perm_test_stratified_median(
  input_data = df_exp,
  grp = "grp_water",
  strat = "strat_sun",
  outcome = "day_9_len",
  B = B,
  seed = 82803,
  alternative = "greater"
)

cat(sprintf("Water effect - Observed T: %.4f\n", water_res$T_obs))
```

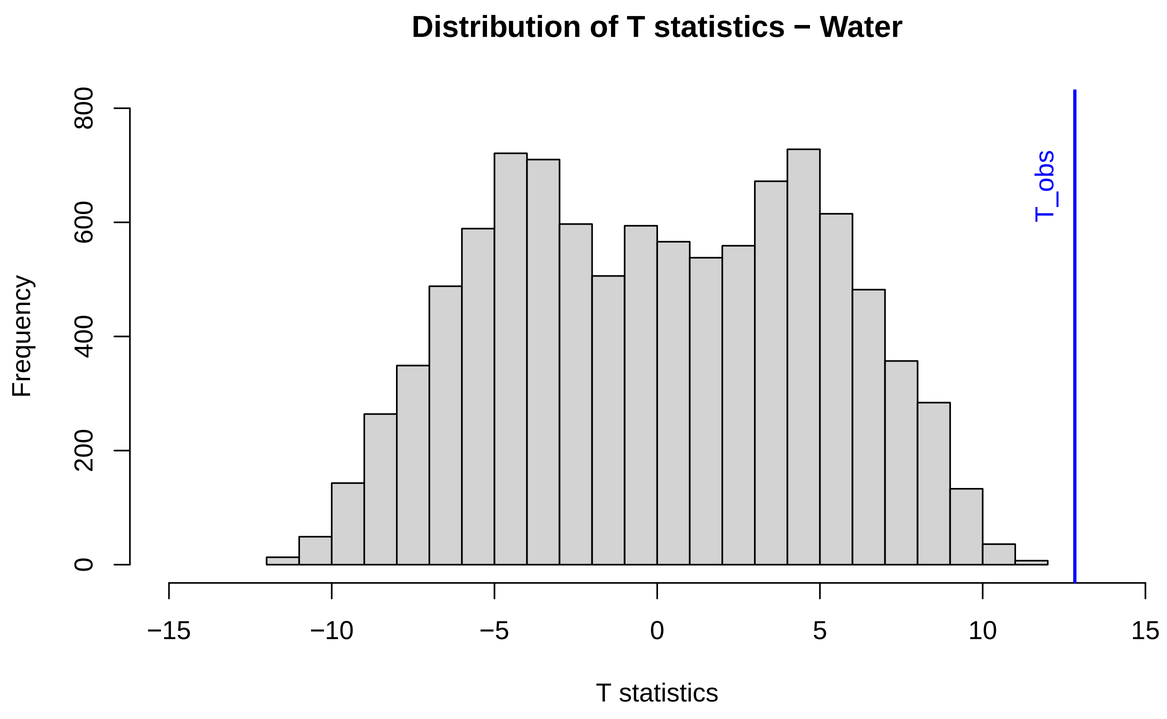
```
## Water effect - Observed T: 12.8318
```

```r
cat(sprintf("Water effect - One-sided p-value: %s\n", sprintf('%.2e', water_res$p_value)))
```

```
## Water effect - One-sided p-value: 1.00e-04
```

The p-value suggests there is very strong evidence that, within each sunlight level, watering increases the median day-9 height. We can also produce a histogram for the distribution of permuted statistics. This is illustrated below:

```r
hist(water_res$T_perm, main = "Distribution of T statistics - Water",
     xlab = "T statistics", breaks=25, xlim = c(-15, 15), ylim = c(0,800))
abline(v = water_res$T_obs, col = "blue", lwd = 2)
text(x = 11.5, y = 600, labels = "T_obs", srt = 90, pos = 4, col = "blue")
```



5.3. Test 2: Sunlight Effect (conditioning on Water)

- **Goal:** assess whether shade yields greater height than sun (conditional on water).
- **Null H_0:** within each water stratum, the median heights for sun and shade are equal.
- **Alternative H_1:** shade has larger medians. Since the statistic is (Sun − Shade), this is a one-sided "less" test.
- **Statistic:** for each water level w, set $G = \mathbb{1}\{\text{Sun} = 1\}$ so that $\Delta_w = \text{median}_{\text{Sun}} - \text{median}_{\text{Shade}}$, and combine as

$$T_{\text{obs}} = \frac{\sum_w n_w \Delta_w}{\sum_w n_w}.$$

Permutation procedure.
Within each water stratum w, randomly permute the sunlight labels, recompute T to get $\{T_b\}_{b=1}^{B}$.

One-sided p-value (left tail).

$$p = \frac{\#\{T_b \le T_{\text{obs}}\} + 1}{B + 1}.$$

```r
df_exp$grp_sun  <- df_exp$sunlight
df_exp$strat_w  <- df_exp$water

set.seed(82803)
B <- 10000L
sun_res <- perm_test_stratified_median(
  input_data = df_exp,
  grp = "grp_sun",
  strat = "strat_w",
  outcome = "day_9_len",
  B = B,
  seed = 82803,
  alternative = "less"
)

cat(sprintf("Sunlight effect - Observed T: %.4f\n", sun_res$T_obs))
```

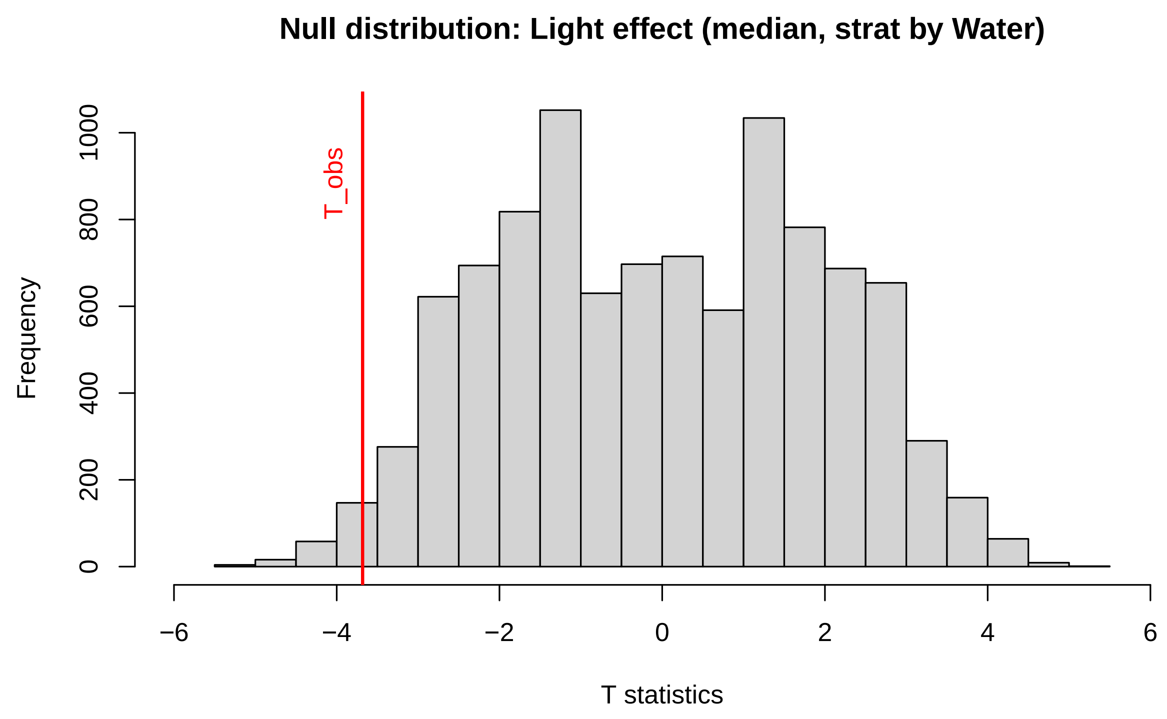
```
## Sunlight effect - Observed T: -3.6818
```

```r
cat(sprintf("Sunlight effect - One-sided p-value: %s\n", sprintf('%.2e', sun_res$p_value)))
```

```
## Sunlight effect - One-sided p-value: 1.32e-02
```

The p-value suggests there is very strong evidence that, within each water condition, sunlight decreases the median day-9 height. We can also produce a histogram for the distribution of permuted statistics. This is illustrated below:

```r
hist(sun_res$T_perm, main = "Null distribution: Light effect (median, strat by Water)",
     xlab = "T statistics", xlim = c(-6,6))
abline(v = sun_res$T_obs, col = "red", lwd = 2)
text(x = -4.2, y = 800, labels = "T_obs", srt = 90, pos = 4, col = "red")
```



5.4. Test 3: DW vs FN comparison

We compare **Darkness + Water (DW)** against **Fridge + No Water (FN)** using the difference in medians and a **right-tailed** permutation test (H1: DW > FN).

```r
## ============================
## Darkness + Water (DW) vs Fridge + No Water (FN)
## ============================

# Subset relevant conditions
is_DW <- grepl("^DW[0-9]+$", proj3_data$label)
is_FN <- grepl("^FN[0-9]+$", proj3_data$label)
df_dwfn <- proj3_data[is_DW | is_FN, , drop = FALSE]

# Create two-level group variable
df_dwfn$group <- ifelse(grepl("^DW", df_dwfn$label), "dark_water", "fridge_nowater")

# Outcome variable
stopifnot("day_9_len" %in% names(df_dwfn))
y <- df_dwfn$day_9_len

# Observed difference in medians (Darkness+Water - Fridge+NoWater)
T_obs <- median(y[df_dwfn$group == "dark_water"], na.rm = TRUE) -
         median(y[df_dwfn$group == "fridge_nowater"], na.rm = TRUE)
cat("\n=== Darkness + Water vs Fridge + No Water ===\n")
```

```
##
## === Darkness + Water vs Fridge + No Water ===
```

```r
cat(sprintf("Observed diff in medians (DW - FN): %.4f\n", T_obs))
```

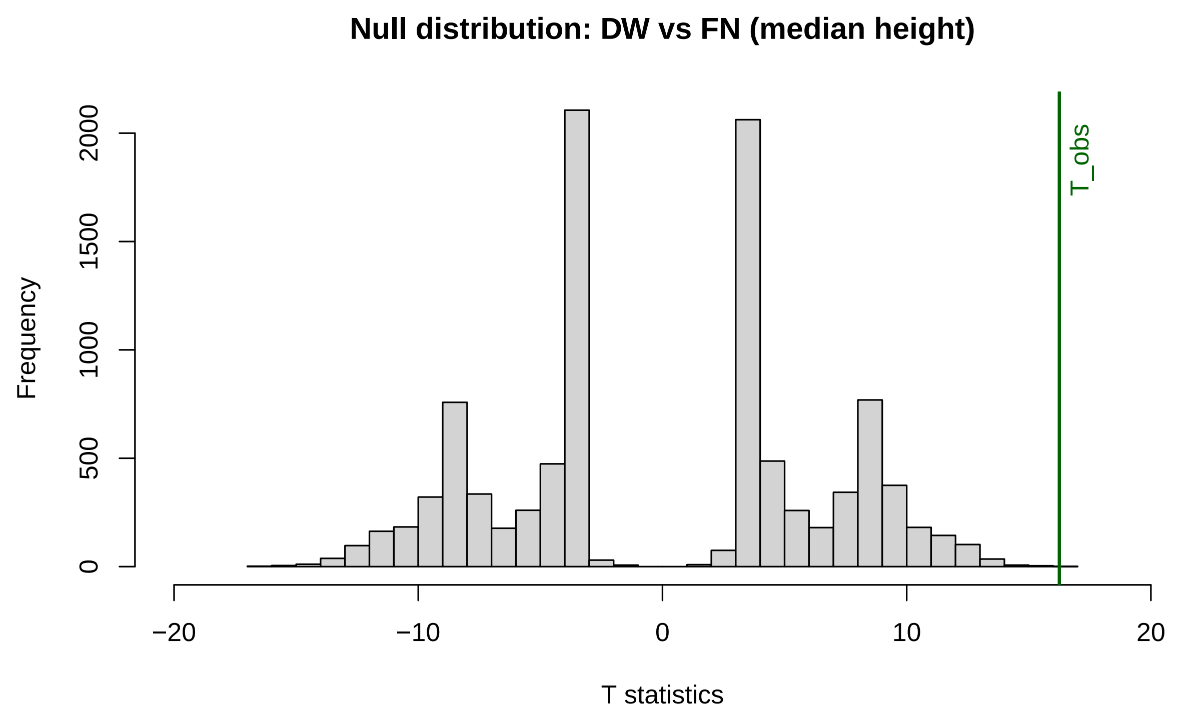
```
## Observed diff in medians (DW - FN): 16.2500
```

```r
# Permutation test (one-sided, H1: DW > FN)
set.seed(123)
B <- 10000
T_perm <- numeric(B)
for (b in seq_len(B)) {
  perm_group <- sample(df_dwfn$group)
  T_perm[b] <- median(y[perm_group == "dark_water"], na.rm = TRUE) -
               median(y[perm_group == "fridge_nowater"], na.rm = TRUE)
}

p_one_sided <- (sum(T_perm >= T_obs) + 1) / (B + 1)
cat(sprintf("One-sided p-value (H1: DW taller than FN): %s\n",
            sprintf('%.2e', p_one_sided)))
```

```
## One-sided p-value (H1: DW taller than FN): 2.00e-04
```

The p-value suggests there is very strong evidence that onions kept in darkness with water (DW) are taller by day 9 than those stored in a fridge with no water (FN). We can also produce a histogram for the distribution of permuted statistics. This is illustrated below:

```r
# Visualization
hist(T_perm,
     main = "Null distribution: DW vs FN (median height)",
     xlab = "T statistics", xlim=c(-20,20),
     breaks = 25)
abline(v = T_obs, col = "darkgreen",lwd = 2)
text(x = T_obs+.3, y = max(hist(T_perm, plot = FALSE)$counts)*0.8,
     labels = "T_obs", srt = 90, pos = 4, col = "darkgreen")
```



5.5. Controlling Site-level environment

To control for site-level environment, we re-conduct the above tests by further stratifying by **Lab**. Strata missing a level are excluded from the statistic. As we shall see below, we still have small p-values for all the tests.

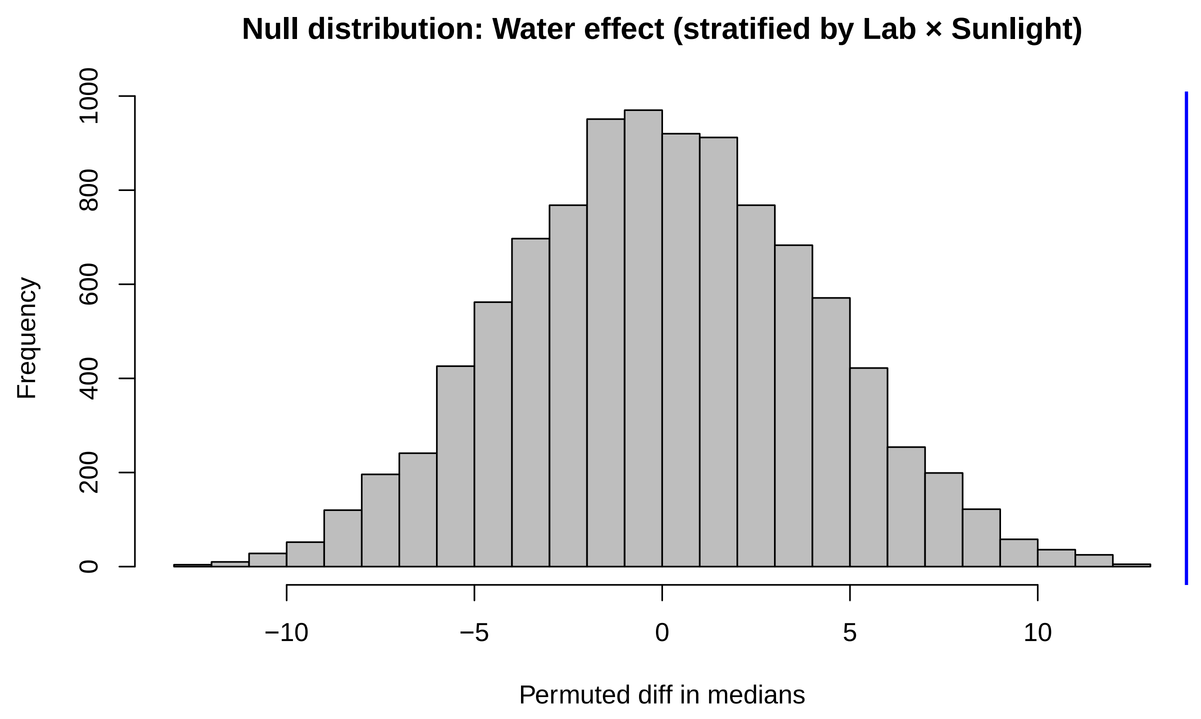
```r
## ---- helper: filter degenerate strata (needs df_exp, grp, strat) ----
filter_valid_strata <- function(dat, grp, strat) {
  # count #obs at each (stratum, group)
  tab <- with(dat, table(dat[[strat]], dat[[grp]]))
  # keep strata with both groups present (>=1 in each column)
  keep_levels <- rownames(tab)[rowSums(tab > 0) == ncol(tab)]
  dropped <- setdiff(levels(factor(dat[[strat]])), keep_levels)
  dat_keep <- dat[dat[[strat]] %in% keep_levels, , drop = FALSE]
  list(data = dat_keep, kept = keep_levels, dropped = dropped)
}
```

```r
## ---- TEST 1: Water effect | Lab × Sunlight (with auto-drop of empty cells) ----
df_exp$grp_water      <- df_exp$water                        # 1 = Water, 0 = NoWater
df_exp$strat_lab_sun <- interaction(df_exp$lab, df_exp$sunlight, drop = TRUE)

# filter out strata that don't have BOTH water levels
flt <- filter_valid_strata(df_exp, grp = "grp_water", strat = "strat_lab_sun")
if (length(flt$dropped)) {
  message("Dropped strata (missing a water level): ",
          paste(flt$dropped, collapse = ", "))
}
```

```r
df_ws <- flt$data

# if nothing left, stop gracefully
if (nrow(df_ws) == 0L || length(unique(df_ws$strat_lab_sun)) == 0L) {
  stop("No valid Lab × Sunlight strata contain both Water and NoWater. Cannot run the test.")
}

set.seed(82803)
water_lab_sun_res <- perm_test_stratified_median(
  input_data = df_ws,
  grp = "grp_water",
  strat = "strat_lab_sun",
  outcome = "day_9_len",
  B = 10000L,
  seed = 82803,
  alternative = "greater"
)

cat(sprintf("Water effect | Lab × Sunlight strata - T_obs: %.4f, p = %s\n",
            water_lab_sun_res$T_obs, sprintf('%.2e', water_lab_sun_res$p_value)))
```

```
## Water effect | Lab × Sunlight strata - T_obs: 13.9600, p = 1.00e-04
```

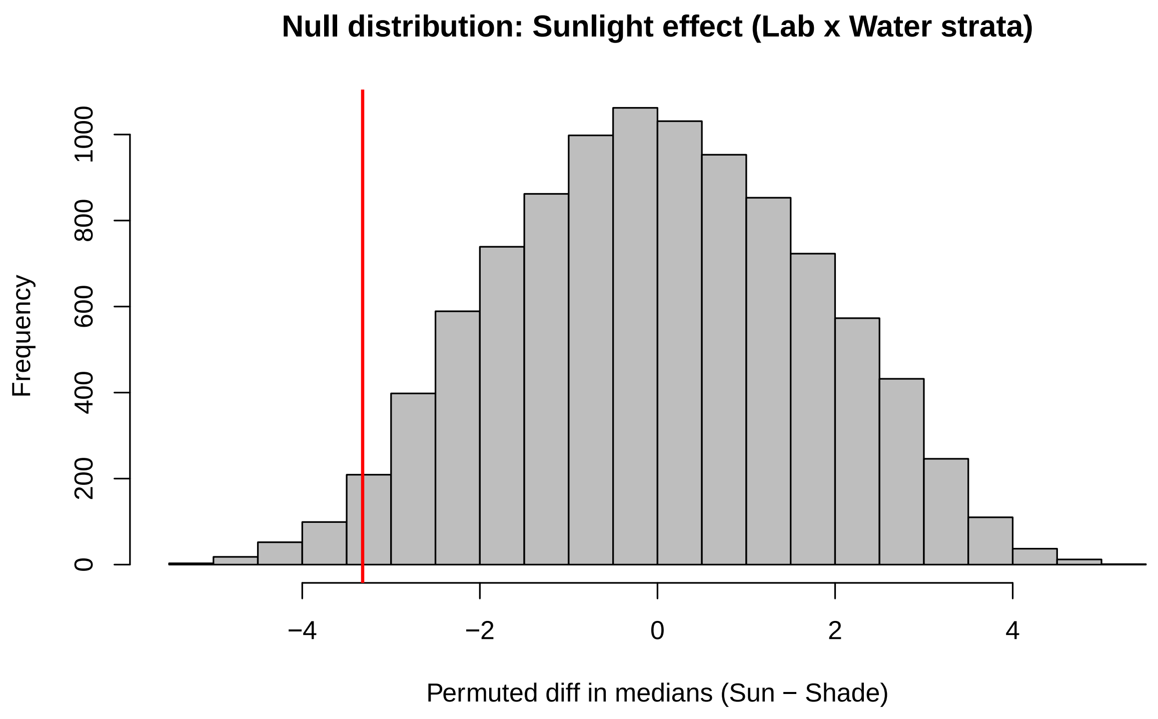
```r
if (length(water_lab_sun_res$T_perm) > 0L && is.finite(water_lab_sun_res$T_obs)) {
  hist(water_lab_sun_res$T_perm,
       main = "Null distribution: Water effect (stratified by Lab × Sunlight)",
       xlab = "Permuted diff in medians", col = "grey", breaks = 25)
  abline(v = water_lab_sun_res$T_obs, col = "blue", lwd = 2)
} else {
  message("Permutation vector is empty after filtering; no histogram drawn.")
}
```



```r
## ---- TEST 2: Sunlight effect | Lab x Water (with auto-drop of empty cells) ----

# Helper (only if not already defined)
if (!exists("filter_valid_strata")) {
  filter_valid_strata <- function(dat, grp, strat) {
    tab <- with(dat, table(dat[[strat]], dat[[grp]]))
    keep_levels <- rownames(tab)[rowSums(tab > 0) == ncol(tab)]
    dropped <- setdiff(levels(factor(dat[[strat]])), keep_levels)
    dat_keep <- dat[dat[[strat]] %in% keep_levels, , drop = FALSE]
    list(data = dat_keep, kept = keep_levels, dropped = dropped)
  }
}

# 1 = Sun, 0 = Shade; alternative "less" since stat = Sun - Shade and H1 is Shade > Sun
df_exp$grp_sun         <- df_exp$sunlight
df_exp$strat_lab_water <- interaction(df_exp$lab, df_exp$water, drop = TRUE)

# Filter out degenerate strata (those missing either Sun or Shade)
flt2 <- filter_valid_strata(df_exp, grp = "grp_sun", strat = "strat_lab_water")
if (length(flt2$dropped)) {
  message("Dropped strata (missing a Sun/Shade level): ",
          paste(flt2$dropped, collapse = ", "))
}
df_sw <- flt2$data

# If no valid strata remain, fail gracefully
if (nrow(df_sw) == 0L || length(unique(df_sw$strat_lab_water)) == 0L) {
  stop("No valid Lab x Water strata contain both Sun and Shade; cannot run the Sunlight test.")
}

set.seed(82803)
B <- 10000L
sun_lab_water_res <- perm_test_stratified_median(
  input_data  = df_sw,
  grp         = "grp_sun",            # 1 = Sun, 0 = Shade
  strat       = "strat_lab_water",    # strata: Lab x Water
  outcome     = "day_9_len",
  B           = B,
  seed        = 82803,
  alternative = "less"                # H1: Shade > Sun
)

cat(sprintf("Sunlight effect | Lab x Water strata - T_obs: %.4f, p = %s\n",
            sun_lab_water_res$T_obs, sprintf('%.2e', sun_lab_water_res$p_value)))
```

```
## Sunlight effect | Lab x Water strata - T_obs: -3.3200, p = 2.38e-02
```

```r
if (length(sun_lab_water_res$T_perm) > 0L && is.finite(sun_lab_water_res$T_obs)) {
  hist(sun_lab_water_res$T_perm,
       main = "Null distribution: Sunlight effect (Lab x Water strata)",
       xlab = "Permuted diff in medians (Sun - Shade)", col = "grey", breaks = 25)
  abline(v = sun_lab_water_res$T_obs, col = "red", lwd = 2)
} else {
  message("Permutation vector is empty after filtering; no histogram drawn.")
}
```



```r
## ---- Test 3: DW vs FN (median length) | Stratified by Lab, one-sided (H1: DW > FN) ----

# Helper (define if not already defined above)
if (!exists("filter_valid_strata")) {
  filter_valid_strata <- function(dat, grp, strat) {
    tab <- with(dat, table(dat[[strat]], dat[[grp]]))
    keep_levels <- rownames(tab)[rowSums(tab > 0) == ncol(tab)]
    dropped <- setdiff(levels(factor(dat[[strat]])), keep_levels)
    dat_keep <- dat[dat[[strat]] %in% keep_levels, , drop = FALSE]
    list(data = dat_keep, kept = keep_levels, dropped = dropped)
  }
}

# Subset to DW and FN rows from the original data
is_DW <- grepl("^DW[0-9]+$", proj3_data$label)
is_FN <- grepl("^FN[0-9]+$", proj3_data$label)
df_dwfn <- proj3_data[is_DW | is_FN, , drop = FALSE]

# Group: 1 = DW (Darkness + Water), 0 = FN (Fridge + No Water)
df_dwfn$grp_dw    <- ifelse(grepl("^DW", df_dwfn$label), 1L, 0L)
df_dwfn$strat_lab <- factor(df_dwfn$lab)

stopifnot("day_9_len" %in% names(df_dwfn))

# Drop labs that don't contain BOTH groups (DW and FN)
flt <- filter_valid_strata(df_dwfn, grp = "grp_dw", strat = "strat_lab")
if (length(flt$dropped)) {
  message("Dropped labs (missing DW or FN): ", paste(flt$dropped, collapse = ", "))
}
df_df <- flt$data

if (nrow(df_df) == 0L || length(unique(df_df$strat_lab)) == 0L) {
  stop("No lab has both DW and FN observations. Cannot run DW vs FN test.")
}

# One-sided stratified permutation test (median), H1: DW > FN  => "greater"
set.seed(123)
B <- 10000L
dw_fn_lab_res <- perm_test_stratified_median(
  input_data  = df_df,
  grp         = "grp_dw",        # 1=DW, 0=FN
  strat       = "strat_lab",     # strata: Lab
  outcome     = "day_9_len",
  B           = B,
  seed        = 123,
  alternative = "greater"
)

cat("\n=== DW vs FN (median length) | stratified by Lab ===\n")
```

```
##
## === DW vs FN (median length) | stratified by Lab ===
```

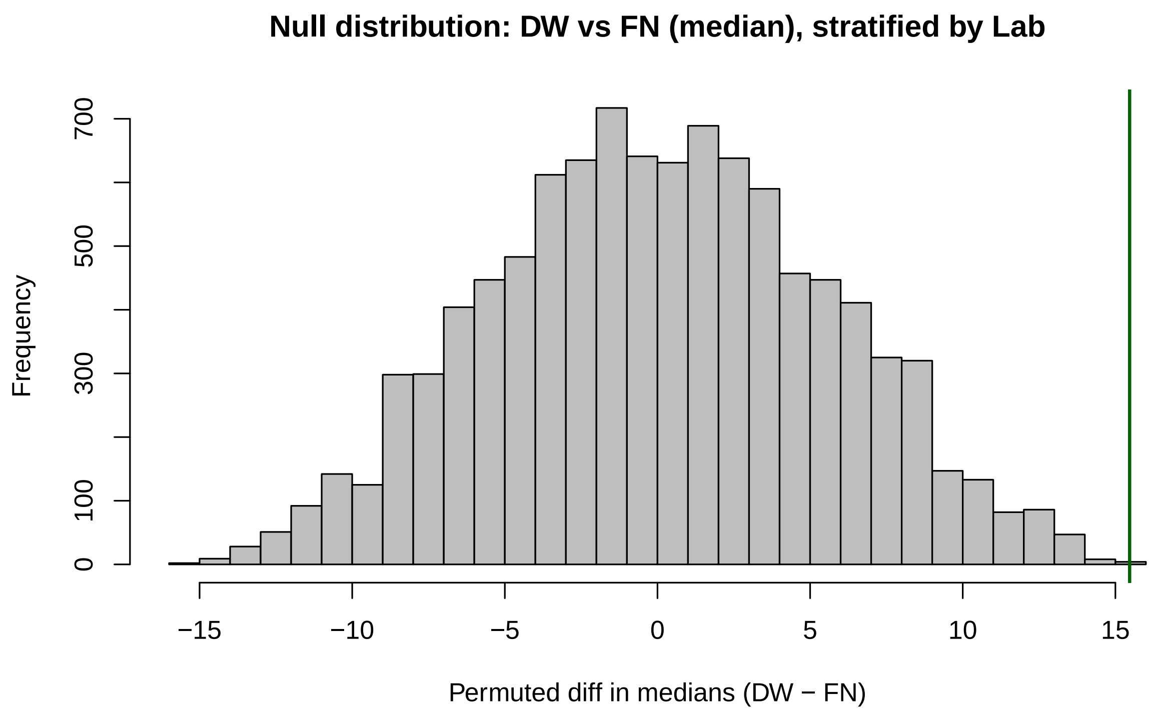
```r
cat(sprintf("Observed diff in medians (DW - FN): %.4f\n", dw_fn_lab_res$T_obs))
```

```
## Observed diff in medians (DW - FN): 15.4667
```

```r
cat(sprintf("One-sided p-value: %s\n", sprintf('%.2e', dw_fn_lab_res$p_value)))
```

```
## One-sided p-value: 1.00e-04
```

```r
# Visualization
if (length(dw_fn_lab_res$T_perm) > 0L && is.finite(dw_fn_lab_res$T_obs)) {
  hist(dw_fn_lab_res$T_perm,
       main = "Null distribution: DW vs FN (median), stratified by Lab",
       xlab = "Permuted diff in medians (DW - FN)", breaks = 25, col = "grey")
  abline(v = dw_fn_lab_res$T_obs, col = "darkgreen", lwd = 2)
} else {
  message("Permutation vector is empty after filtering; no histogram drawn.")
}
```



6. Conclusions:

- From EDA, we observed a clear association between freshness and height — fresher onions tended to be taller.
- Overall, these findings consistently indicate that access to water and protection from direct sunlight both contribute positively to onion growth, while refrigeration (without water) inhibits it.

- We recorded the lengths of the green onions every three days throughout the experiment. While the present analysis focuses on the day-9 measurements, further analyses could be conducted to study growth trajectories over time.
- After conducting our experiment, our group discovered that the refrigerator in Chandler's lab was broken. Thus, we ran further tests while conditioning on lab to account for the broken refrigerator.