

BumblebeeRobbing

Daniel

2025-06-09

Behavioral assays done on bumblebees at Rocky Mountains Biology Laboratory (RMBL) field season 2023. Observations were carried out by Nick Dabagia and Daniel Souto for *Corydalis* and *Mertensia* species. Y-Tube assays carried out by Oriana Gutierrez in 2024.

Methods:

Refer to Souto-Vilarós et al. ‘Yeast volatiles promote larceny in bumble bee behavior.’

Data analysis:

The dataset is arranged as date, plant species, bee individual, bee species, whether or not a choice was made, stalk number, treatment, whether or not the bee tried to legitimately visit the flower, whether it robbed the flower, time to rob, time feeding, flower number (total for that bout) and additional notes.

Note that not all packages are used in the current analysis

We need to wrangle the data a bit. Namely: Filter the data to include only cases where a flower was robbed (column CHOICE, “Yes”), Remove an outlier which was inactive on a flower over 3 minutes. Make sure all data is numeric and there are no missing data.

This final data set includes: 57 total successful robber trials (out of 116 bee trials transcribed) 42 *Corydalis* 2° robbers (22 *bifarius*, 16 *flavifrons*, 4 *mixtus*) and 15 *Mertensia* 2° robbers (all *flavifrons*).

Bombus mixtus removed from final analysis due to low sample size.

```
robbed_flowers <- visitation %>%
  filter(robbed == "yes") %>% #note that this will reduce total number of bees which made a choice since
  filter(species != "mixtus") %>%
  filter(timetorob != 154.7)

robbed_flowers <- robbed_flowers %>%
  mutate(
    timetorob = as.numeric(timetorob),
    timefeeding = as.numeric(timefeeding)
  ) %>%
  filter(!is.na(timetorob) & !is.na(timefeeding))

robbed_flowers <- robbed_flowers %>%
  mutate(trytolegit_binary = ifelse(trytolegit == "yes", 1, 0))

#How many bees per bee species per flower species in the original and the filtered datasets?

visitation %>% group_by(sample, plantsp, species) %>% count()
```

```
## # A tibble: 120 x 4
```

```
## # Groups:   sample, plantsp, species [120]
##   sample    plantsp  species      n
##   <chr>      <chr>    <chr>    <int>
##  1 CORYBIF001 corydalis bifarius     5
##  2 CORYBIF002 corydalis bifarius     1
##  3 CORYBIF003 corydalis bifarius     2
##  4 CORYBIF004 corydalis bifarius    16
##  5 CORYBIF005 corydalis bifarius     1
##  6 CORYBIF006 corydalis bifarius    11
##  7 CORYBIF007 corydalis bifarius    10
##  8 CORYBIF008 corydalis bifarius    10
##  9 CORYBIF009 corydalis bifarius     1
## 10 CORYBIF010 corydalis bifarius     1
## # i 110 more rows
```

```
robbed_flowers %>% group_by(sample, plantsp, species, bee) %>% count()
```

```
## # A tibble: 49 x 5
## # Groups:   sample, plantsp, species, bee [49]
##   sample    plantsp  species    bee      n
##   <chr>      <chr>    <chr>    <int> <int>
##  1 CORYBIF001 corydalis bifarius     3     3
##  2 CORYBIF004 corydalis bifarius     3    15
##  3 CORYBIF006 corydalis bifarius     7    11
##  4 CORYBIF007 corydalis bifarius     8    10
##  5 CORYBIF008 corydalis bifarius     2     9
##  6 CORYBIF011 corydalis bifarius     7    12
##  7 CORYBIF012 corydalis bifarius     8    19
##  8 CORYBIF014 corydalis bifarius     5    20
##  9 CORYBIF017 corydalis bifarius    11     9
## 10 CORYBIF019 corydalis bifarius     2     6
## # i 39 more rows
```

Simple summary statistics for time to rob (mean, SE) broken down by bumblebee species and plant sp. Note that time to rob is longer for control flowers, but also SE higher for controls. Consistent response for Mreu plants. Tactic switch, I just converted into a binary column yes = 1, no = 0. Feeding time is higher in both treatments, with high SE but consistent throughout.

```
summary_stats <- robbed_flowers %>%
  group_by(species, treatment, plantsp) %>%
  summarize(
    Mean_TimeToRob = mean(timetorob, na.rm = TRUE),
    SE_TimeToRob = sd(timetorob, na.rm = TRUE) / sqrt(n()),
    Mean_TimeFeeding = mean(timefeeding, na.rm = TRUE),
    SE_TimeFeeding = sd(timetorob, na.rm = TRUE) / sqrt(n()),
    Mean_tacticswitch = mean(trytolegit_binary, na.rm = TRUE),
    SE_tacticswitch = sd(trytolegit_binary, na.rm = TRUE) / sqrt(n())
  ) %>%
  arrange(plantsp, treatment)
```

```
## 'summarise()' has grouped output by 'species', 'treatment'. You can override
## using the '.groups' argument.
```

```
print(summary_stats)
```

```
## # A tibble: 6 x 9
## # Groups:   species, treatment [4]
##   species    treatment plantsp   Mean_TimeToRob SE_TimeToRob Mean_TimeFeeding
##   <chr>      <chr>      <chr>         <dbl>         <dbl>         <dbl>
## 1 bifarius   control    corydalis      5.42          0.637          7.76
## 2 flavifrons control    corydalis      5.24          0.672          8.04
## 3 bifarius   mreu      corydalis      4.16          0.343          7.09
## 4 flavifrons mreu      corydalis      3.20          0.272          8.04
## 5 flavifrons control    mertensia      7.12          1.08           9.25
## 6 flavifrons mreu      mertensia      3.97          0.426          8.06
## # i 3 more variables: SE_TimeFeeding <dbl>, Mean_tacticswitch <dbl>,
## #   SE_tacticswitch <dbl>
```

```
write.table(summary_stats, "results/summary_feeding_stats.csv", row.names = FALSE)
```

```
# Summarize the data to calculate the number of visits
```

```
summary_data <- robbed_flowers %>%
  group_by(treatment, species, plantsp) %>%
  summarize(
    number_of_visits = n(), # Count the number of visits
    .groups = "drop"
  )
```

```
# Calculate the mean, stdev, and se for each treatment group (control vs mreu)
```

```
visit_summary <- summary_data %>%
  group_by(species, plantsp) %>%
  summarize(
    mean_visits_control = mean(number_of_visits[treatment == "control"], na.rm = TRUE),
    mean_visits_treated = mean(number_of_visits[treatment == "mreu"], na.rm = TRUE),
    stdev_diff = sd(number_of_visits[treatment == "mreu"] - number_of_visits[treatment == "control"], na.rm = TRUE),
    se_diff = stdev_diff / sqrt(n()),
    .groups = "drop"
  )
```

```
# View the resulting summary table
```

```
print(visit_summary)
```

```
## # A tibble: 3 x 6
##   species    plantsp mean_visits_control mean_visits_treated stdev_diff se_diff
##   <chr>      <chr>         <dbl>         <dbl>         <dbl>   <dbl>
## 1 bifarius   corydal~      114          141           NA      NA
## 2 flavifrons corydal~      47          128           NA      NA
## 3 flavifrons mertens~      39           78           NA      NA
```

```
# Perform t-tests for each species and plantsp combination
```

```
t_test_results <- robbed_flowers %>%
  group_by(species, plantsp) %>%
  summarize(
    t_test = list(
      t.test(
```

```

    x = treatment == "mreu",      # Logical vector for mreu
    y = treatment == "control"   # Logical vector for control
  )
),
.groups = "drop"
)

# Extract p-values and t-statistics from the t-test results
t_test_summary <- t_test_results %>%
  mutate(
    p_value = sapply(t_test, function(x) x$p.value),      # Extract p-value
    statistic = sapply(t_test, function(x) x$statistic)    # Extract t-statistic
  )

# Select only necessary columns
t_test_summary <- t_test_summary %>%
  dplyr::select(species, plantsp, p_value, statistic)

# View the summarized results
print(t_test_summary)

```

```

## # A tibble: 3 x 4
##   species    plantsp   p_value statistic
##   <chr>      <chr>      <dbl>      <dbl>
## 1 bifarius  corydalis 1.68e- 2      2.40
## 2 flavifrons corydalis 5.52e-20      9.74
## 3 flavifrons mertensia 1.77e- 7      5.39

```

```

# Extract p-values, t-statistics, and degrees of freedom from the t-test results
t_test_summary <- t_test_results %>%
  mutate(
    p_value = sapply(t_test, function(x) x$p.value),      # Extract p-value
    statistic = sapply(t_test, function(x) x$statistic),  # Extract t-statistic
    df = sapply(t_test, function(x) x$parameter)          # Extract degrees of freedom
  )

# Explicitly use dplyr::select()
t_test_summary <- t_test_summary %>%
  dplyr::select(species, plantsp, p_value, statistic, df)

# View the summarized results
print(t_test_summary)

```

```

## # A tibble: 3 x 5
##   species    plantsp   p_value statistic    df
##   <chr>      <chr>      <dbl>      <dbl> <dbl>
## 1 bifarius  corydalis 1.68e- 2      2.40  508
## 2 flavifrons corydalis 5.52e-20      9.74  348
## 3 flavifrons mertensia 1.77e- 7      5.39  232

```

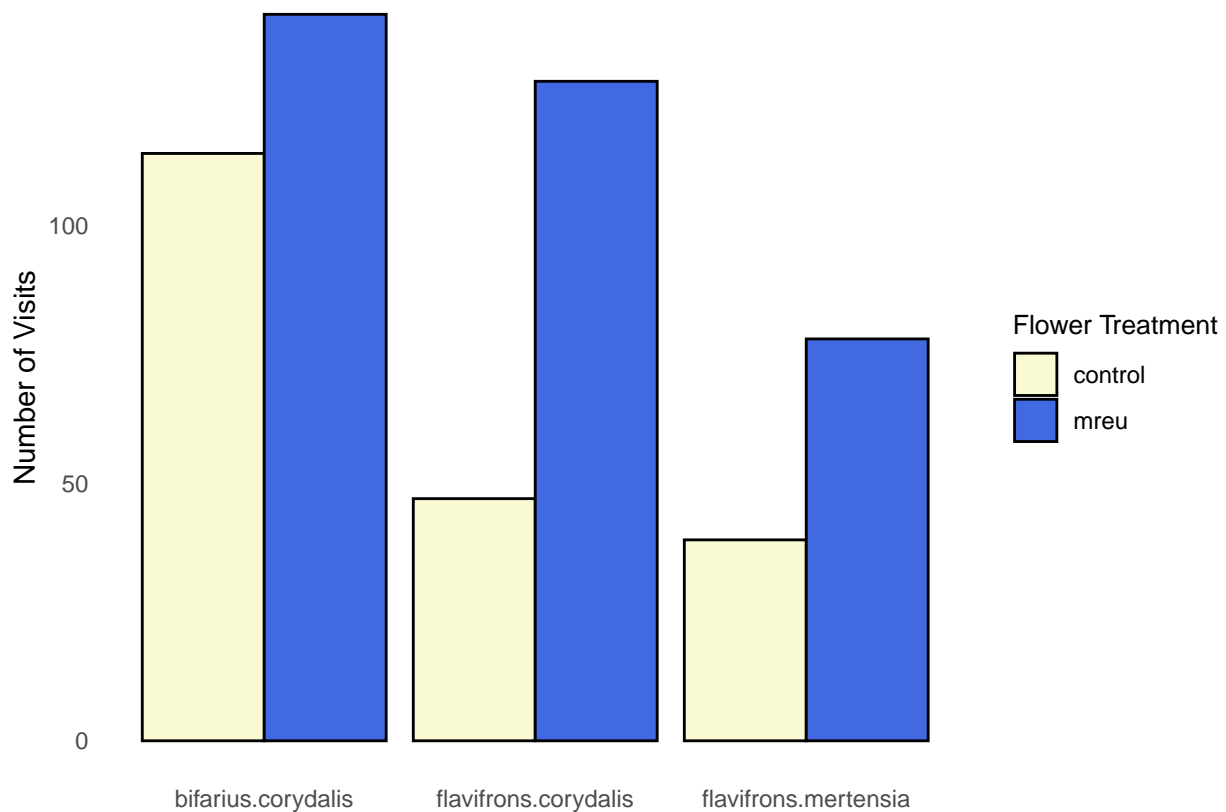
```

# Create a histogram
ggplot(robbed_flowers, aes(x = interaction(species, plantsp), fill = treatment)) +

```

```
geom_bar(position = "dodge", color = "black") +
labs(x = NULL,
     y = "Number of Visits",
     fill = "Flower Treatment"
) +
scale_fill_manual(
  values = c("mreu" = "#4169E1",
            "control" = "#FAFAD2")
)+
theme_minimal() +
theme(
  panel.grid.major = element_blank(),
  panel.grid.minor = element_blank(),
  legend.title = element_text(size = 10),
  legend.text = element_text(size = 9)
) -> plot
```

plot



```
ggsave("results/bee_visits_histogram.pdf", plot = plot, width = 8, height = 6)
```

```
summary_df <- robbed_flowers %>%
  group_by(group = interaction(species, plantsp), treatment) %>%
  summarise(count = n(), .groups = "drop") %>%
  group_by(group) %>%
```

```

summarise(
  max_y = max(count),
  x_pos = as.numeric(factor(group)), # for proper placement on x axis
  .groups = "drop"
) %>%
mutate(label = "*")

```

```

## Warning: Returning more (or less) than 1 row per 'summarise()' group was deprecated in
## dplyr 1.1.0.
## i Please use 'reframe()' instead.
## i When switching from 'summarise()' to 'reframe()', remember that 'reframe()'
## always returns an ungrouped data frame and adjust accordingly.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.

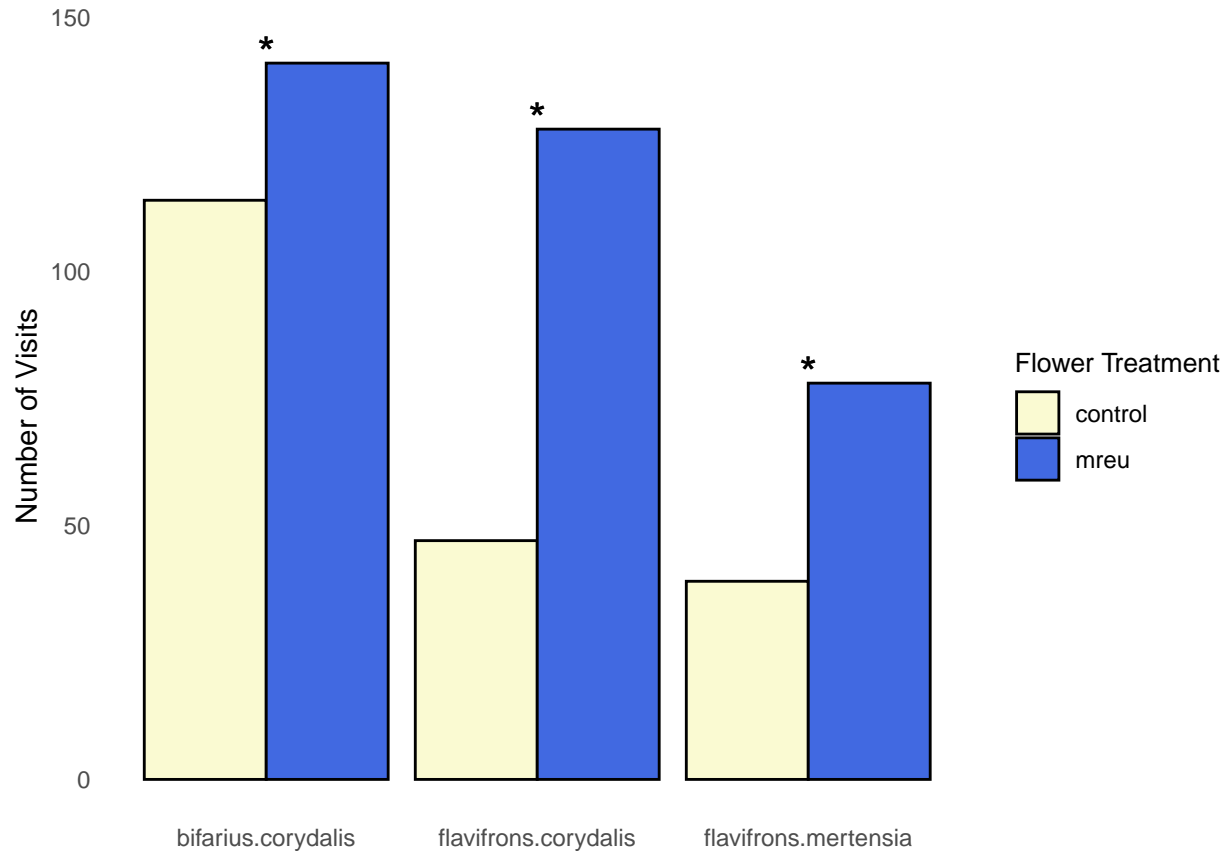
```

```

ggplot(robbed_flowers, aes(x = interaction(species, plantsp), fill = treatment)) +
  geom_bar(position = position_dodge(width = 0.9), color = "black") +
  geom_text(
    data = summary_df,
    aes(x = group, y = max_y + 2, label = label),
    inherit.aes = FALSE,
    size = 6
  ) +
  labs(x = NULL,
       y = "Number of Visits",
       fill = "Flower Treatment") +
  scale_fill_manual(
    values = c("mreu" = "#4169E1", "control" = "#FAFAD2")
  ) +
  theme_minimal() +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 9)
  ) -> plot2

```

```
plot2
```



```
mean_se <- function(x) {
  m <- mean(x, na.rm = TRUE)
  se <- sd(x, na.rm = TRUE) / sqrt(length(na.omit(x)))
  return(paste0(round(m, 2), " ± ", round(se, 2)))
}

timetorob_box_species <- ggplot(robbed_flowers, aes(x = interaction(species, plantsp), y = timetorob, f
  geom_boxplot(outlier.shape = NA, position = position_dodge(width = 0.75)) +
  geom_point(position = position_jitterdodge(jitter.width = 0.1, dodge.width = 0.75),
    aes(shape = treatment, color = treatment),
    size = 1, alpha = 0.75) +
  stat_summary(
    fun.data = function(x) {
      m <- mean(x)
      se <- sd(x) / sqrt(length(x))
      data.frame(y = m + se * -1.5, label = sprintf("%.2f ± %.2f", m, se))
    },
    geom = "text",
    position = position_dodge(width = 0.75),
    size = 3,
    vjust = 0,
    color = "black"
  ) +
  scale_fill_manual(
    values = c("mreu" = "#4169E1",
      "control" = "#FAFAD2")
  )
}
```

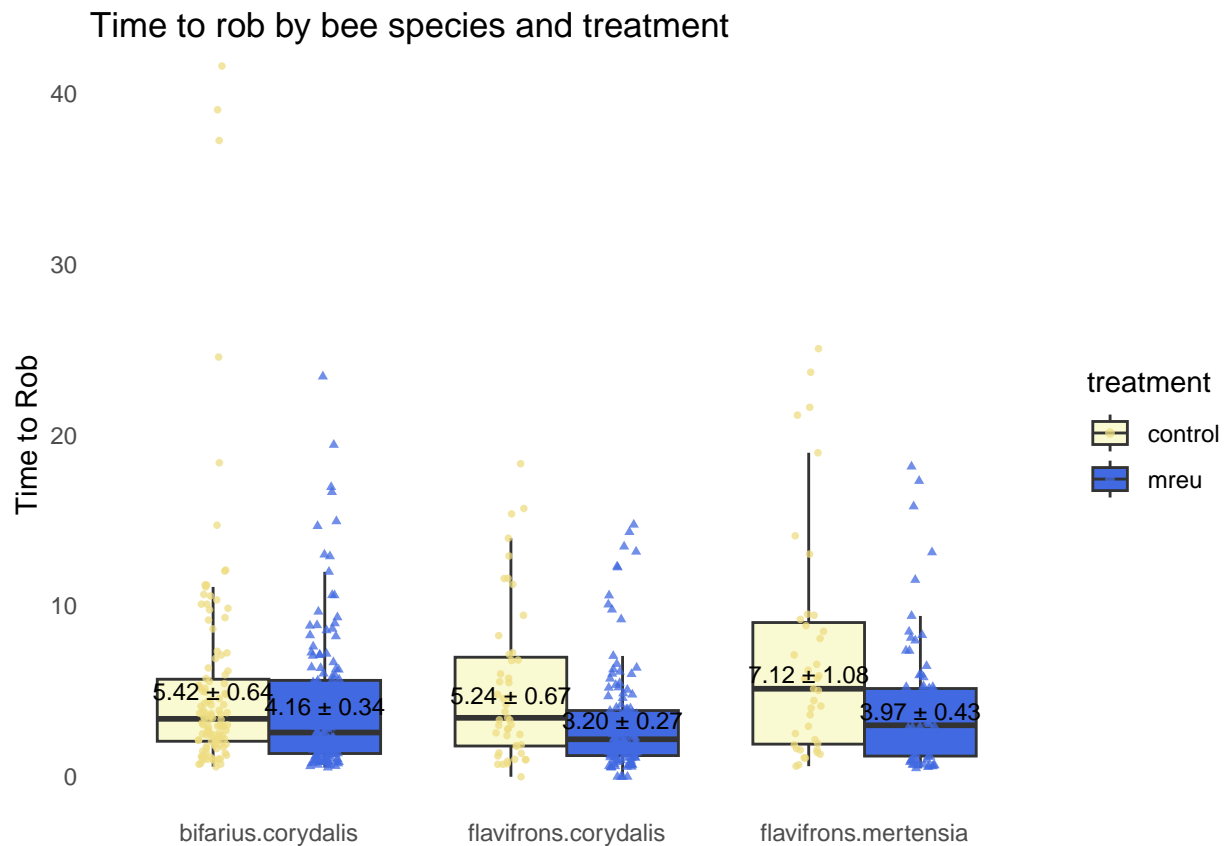
```

)+
scale_color_manual(
  values = c("mreu" = "#4169E1",
             "control" = "#EEDD82")
)+
labs(
  title = "Time to rob by bee species and treatment",
  x = NULL,
  y = "Time to Rob"
) +
#scale_fill_discrete(name = "Treatment") +
#scale_color_discrete(name = "Treatment") +
scale_shape_manual(name = "Treatment", values = c(16, 17), guide = "none") +
theme_minimal()+
theme(
  panel.grid.major = element_blank(),
  panel.grid.minor = element_blank()
)

timetorob <- timetorob_box_species +
  coord_cartesian(ylim = c(0, 40))

timetorob

```




```
ggsave("results/timetorob_pub.pdf", plot = timetorob, width = 8, height = 6)
```

ANOVA time to rob on bee species (1) and plant species (2) - in both cases treatment is highly significant, but interaction between either plant or bee species and treatment is not. Suggests instead that only the nectar treatment significantly different

```
anova_result <- aov(timetorob ~ species + treatment, data = robbed_flowers)
summary(anova_result)
```

```
##              Df Sum Sq Mean Sq F value    Pr(>F)
## species        1      29      29.2   1.249    0.264
## treatment      1     455     455.1  19.479 1.23e-05 ***
## Residuals    544    12711      23.4
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova_result_plant <- aov(timetorob ~ plantsp + treatment, data = robbed_flowers)
summary(anova_result_plant)
```

```
##              Df Sum Sq Mean Sq F value    Pr(>F)
## plantsp        1      45      44.6   1.918    0.167
## treatment      1     491     491.4  21.115 5.38e-06 ***
## Residuals    544    12659      23.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To make sure this significance holds, I did wilcox test only on the mean time to rob. Again, significantly different ($p = < 0.05$)

```
treatment_control <- subset(robbed_flowers, treatment == "control") #control subset
treatment_mreu <- subset(robbed_flowers, treatment == "mreu") #inoculated flowers subset

# Perform the Wilcoxon rank-sum test
wilcox_test_result <- wilcox.test(treatment_control$timetorob, treatment_mreu$timetorob) #wilcox test

# Print the test result
print(wilcox_test_result) #wilcox result - highly significant difference of mean time to rob between tr

##
## Wilcoxon rank sum test with continuity correction
##
## data:  treatment_control$timetorob and treatment_mreu$timetorob
## W = 42406, p-value = 1.502e-05
## alternative hypothesis: true location shift is not equal to 0
```

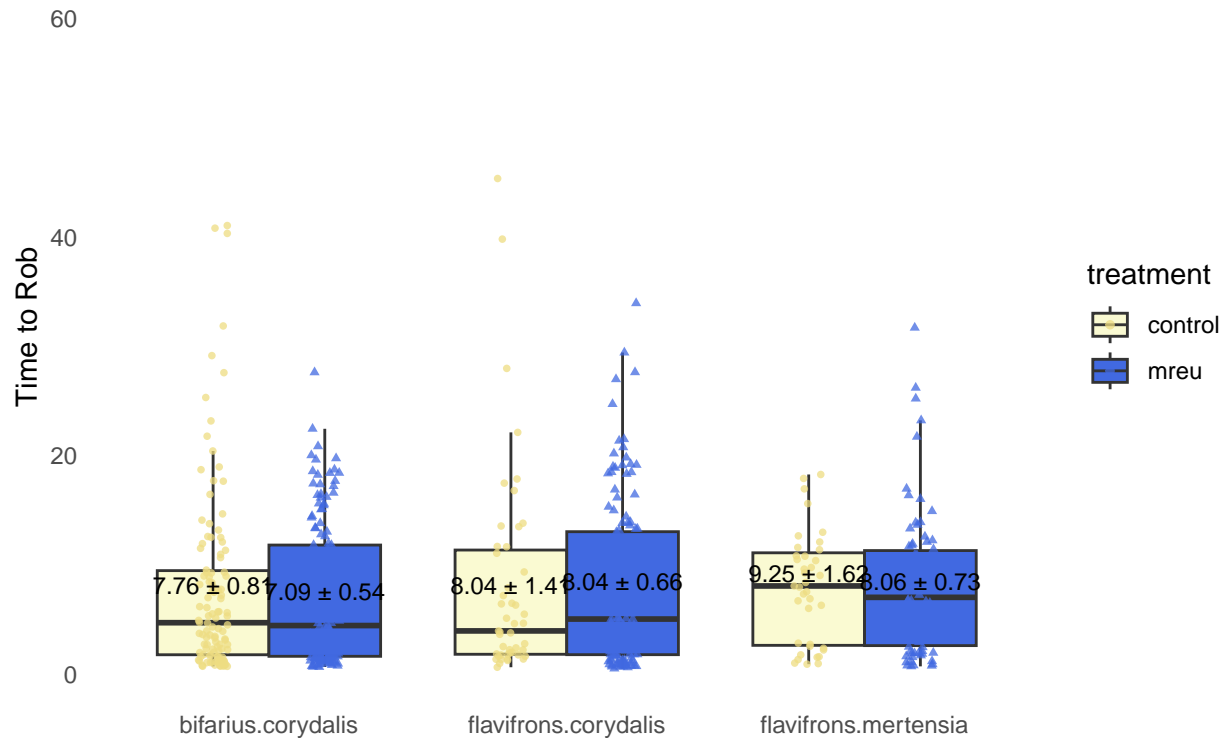
We do the same analysis for feedint time and show that there is no significant difference between treatments on the feeding time of bees. It seems that yeast does not have a significant flavor(?) difference or not not different enough or nasty enough for bees to care. They spend the same amount of time feeding on sterile or yeast inoculated nectar

```

feedingtime_box_species <- ggplot(robbed_flowers, aes(x = interaction(species, plantsp), y = timefeeding)) +
  geom_boxplot(outlier.shape = NA, position = position_dodge(width = 0.75)) +
  geom_point(position = position_jitterdodge(jitter.width = 0.1, dodge.width = 0.75),
    aes(shape = treatment, color = treatment),
    size = 1, alpha = 0.75) +
  stat_summary(
    fun.data = function(x) {
      m <- mean(x)
      se <- sd(x) / sqrt(length(x))
      data.frame(y = m + se * -.5, label = sprintf("%.2f ± %.2f", m, se))
    },
    geom = "text",
    position = position_dodge(width = 0.75),
    size = 3,
    vjust = 0,
    color = "black"
  ) +
  scale_fill_manual(
    values = c("mreu" = "#4169E1",
      "control" = "#FAFAD2")
  ) +
  scale_color_manual(
    values = c("mreu" = "#4169E1",
      "control" = "#EEDD82")
  ) +
  labs(
    title = "Feeding time by bee species and treatment",
    x = NULL,
    y = "Time to Rob"
  ) +
  #scale_fill_discrete(name = "Treatment") +
  #scale_color_discrete(name = "Treatment") +
  scale_shape_manual(name = "Treatment", values = c(16, 17), guide = "none") +
  theme_minimal() +
  theme(panel.grid.major = element_blank(),
    panel.grid.minor = element_blank()
  )
feedingtime_box_species

```

Feeding time by bee species and treatment



```
ggsave("results/feeding_pub.pdf", plot = feedingtime_box_species, width = 8, height = 6)
```

ANOVA and wilcox test confirm. No difference on time spent feeding depending on treatment of nectar.

```
anova_result_feeding <- aov(timefeeding ~ species + treatment, data = robbed_flowerns)
```

```
# Print the ANOVA table
summary(anova_result_feeding)
```

```
##           Df Sum Sq Mean Sq F value Pr(>F)
## species    1     91   90.71   1.518  0.218
## treatment  1     46   46.07   0.771  0.380
## Residuals 544  32510   59.76
```

```
# Perform the Wilcoxon rank-sum test
```

```
wilcox_test_result_feeding <- wilcox.test(treatment_control$timefeeding, treatment_mreu$timefeeding)
```

```
# Print the test result
```

```
print(wilcox_test_result_feeding)
```

```
##
```

```
## Wilcoxon rank sum test with continuity correction
```

```
##
```

```
## data: treatment_control$timefeeding and treatment_mreu$timefeeding
```

```
## W = 34333, p-value = 0.8367
## alternative hypothesis: true location shift is not equal to 0
```

This included all plant species combined. Below I break down everything per plant species (Corydalis and Mertensia)

First subset data

```
corydalis <- subset(robbed_flowers, plantsp == "corydalis")
mertensia <- subset(robbed_flowers, plantsp == "mertensia")
```

Since for Corydalis we have 2 bee species, I still did ANOVA. For Mertensia, there is only one bee species so makes more sense to do wilcox. In the case of Corydalis, the two bee species take different times to find the robbing hole ($p = 0.03$), while time to find the hole for different treatments is still highly significant ($p = 0.001$). We could do different analysis for each bee species, if worth it.

```
anova_result_robbing_corydalis <- aov(timotorob ~ species + treatment, data = corydalis)
# Print the ANOVA table
summary(anova_result_robbing_corydalis)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## species      1     98   98.34    4.309 0.0385 *
## treatment    1    230  230.28   10.091 0.0016 **
## Residuals   427   9744   22.82
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova_result_feeding_corydalis <- aov(timefeeding ~ species + treatment, data = corydalis)
# Print the ANOVA table
summary(anova_result_feeding_corydalis)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## species      1     44   43.61    0.733 0.392
## treatment    1     19   18.60    0.313 0.576
## Residuals   427  25394   59.47
```

For Corydalis, significant difference both for species and treatment, but not the interaction. So, bee species show difference in the time it takes to find a robbing hole ($p = 0.03$), and significantly different according to treatment ($p = 0.001$).

Wilcox test for mertensia, time to rob significantly different between treatments ($P = 0.006$)

```
wilcox_mertensia_timotorob <- wilcox.test(timotorob ~ treatment, data = mertensia)
wilcox_mertensia_timotorob
```

```
##
## Wilcoxon rank sum test with continuity correction
##
## data: timotorob by treatment
## W = 1988, p-value = 0.006991
## alternative hypothesis: true location shift is not equal to 0
```

```
legit_summary <- corydalis %>%
  group_by(unique_name) %>%
  summarise(trytolegit_binary = sum(trytolegit_binary))

print(legit_summary)
```

```
## # A tibble: 35 x 2
##   unique_name          trytolegit_binary
##   <chr>                <dbl>
## 1 Aug-01_corydalis_3_bifarius           2
## 2 Aug-01_corydalis_7_bifarius           1
## 3 Aug-01_corydalis_8_bifarius           2
## 4 Aug-01_corydalis_9_flavifrons        11
## 5 Aug-02_corydalis_12_flavifrons         2
## 6 Aug-02_corydalis_1_flavifrons         0
## 7 Aug-02_corydalis_2_bifarius           1
## 8 Aug-02_corydalis_7_bifarius           2
## 9 Aug-02_corydalis_8_bifarius           2
## 10 Aug-03_corydalis_11_bifarius          3
## # i 25 more rows
```

```
visit_summary <- corydalis %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))

visit_summary
```

```
## # A tibble: 35 x 3
##   unique_name          total_visits legit_visits
##   <chr>                <int>         <int>
## 1 Aug-01_corydalis_3_bifarius          15           2
## 2 Aug-01_corydalis_7_bifarius          11           1
## 3 Aug-01_corydalis_8_bifarius          10           2
## 4 Aug-01_corydalis_9_flavifrons         15          11
## 5 Aug-02_corydalis_12_flavifrons         15           2
## 6 Aug-02_corydalis_1_flavifrons          1           0
## 7 Aug-02_corydalis_2_bifarius           9           1
## 8 Aug-02_corydalis_7_bifarius          12           2
## 9 Aug-02_corydalis_8_bifarius          19           2
## 10 Aug-03_corydalis_11_bifarius          9           3
## # i 25 more rows
```

```
y_tube <- read.csv("data/y_tube_ori_2025.csv", header = T)

table(y_tube$Initial_Choice)
```

```
##
## C T
## 9 13
```

```

chisq.test(table(y_tube$Initial_Choice))

##
## Chi-squared test for given probabilities
##
## data:  table(y_tube$Initial_Choice)
## X-squared = 0.72727, df = 1, p-value = 0.3938

t <- t.test(y_tube$Yeast_time, y_tube$Control_time, paired = TRUE, alternative = "greater")
t

##
## Paired t-test
##
## data:  y_tube$Yeast_time and y_tube$Control_time
## t = 2.4177, df = 21, p-value = 0.0124
## alternative hypothesis: true mean difference is greater than 0
## 95 percent confidence interval:
##  12.99907      Inf
## sample estimates:
## mean difference
##      45.09091

corydalis_bifarius <- subset(corydalis, species == "bifarius")
corydalis_flavifrons <- subset(corydalis, species == "flavifrons")

summary_corydalis <- corydalis %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))

# Create a new column indicating whether trytolegit_binary > 0
legit_summary <- legit_summary %>%
  mutate(trytolegit_gt_0 = ifelse(trytolegit_binary > 0, "Trytolegit > 0", "Trytolegit = 0"))

# Summarize the data by the new column
summary_data <- legit_summary %>%
  group_by(trytolegit_gt_0) %>%
  summarise(total_visits = n())

# Calculate total visits
total_visits <- sum(summary_data$total_visits)

# Calculate percentage of visits for each category
summary_data <- summary_data %>%
  mutate(percentage = total_visits / sum(total_visits))

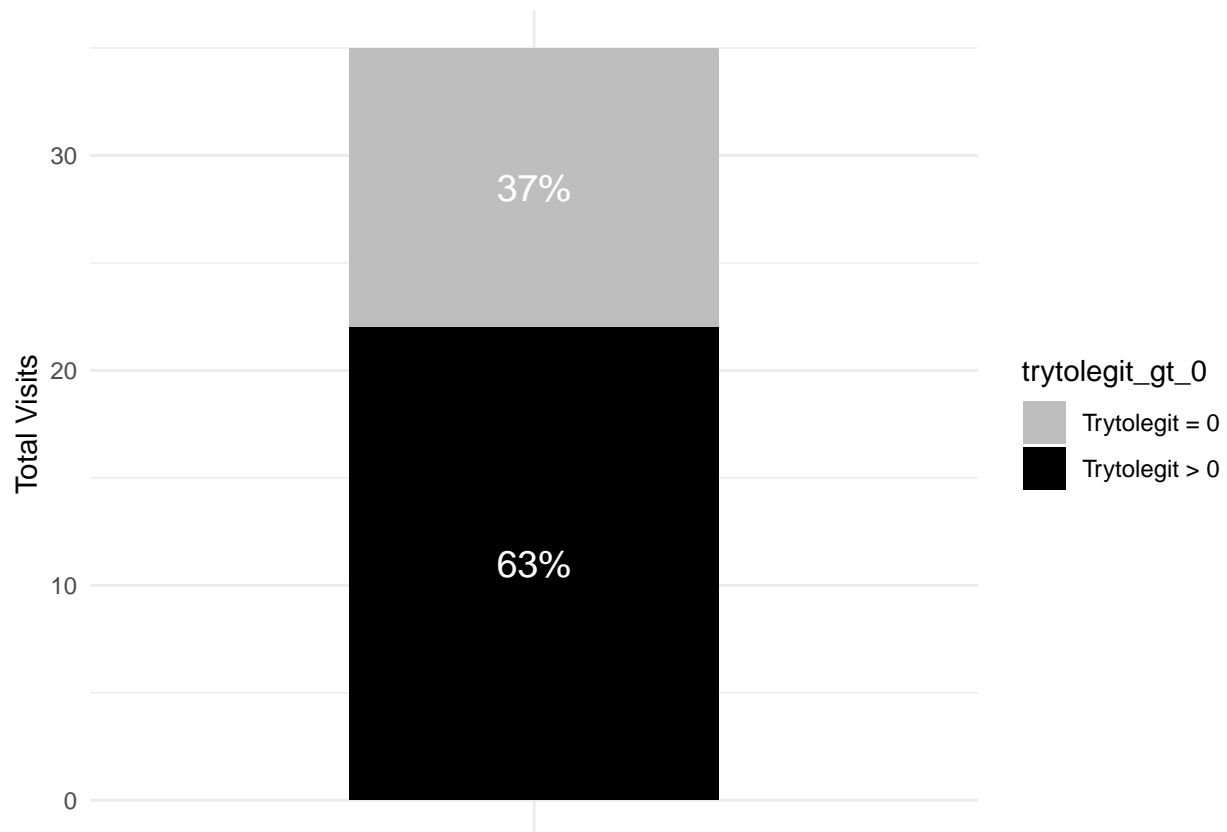
# Plot the barplot
ggplot(summary_data, aes(x = "", y = total_visits, fill = trytolegit_gt_0)) +
  geom_bar(stat = "identity", width = 0.5) +
  geom_text(aes(label = scales::percent(percentage)),

```

```

    position = position_stack(vjust = 0.5),
    color = "white", size = 5) +
labs(x = NULL, y = "Total Visits") +
scale_fill_manual(values = c("Trytolegit = 0" = "gray", "Trytolegit > 0" = "black")) +
theme_minimal() +
theme(axis.text.x=element_blank()) # Hide x-axis label

```



```

legit_summary_mert <- mertensia %>%
  group_by(unique_name) %>%
  summarise(trytolegit_binary = sum(trytolegit_binary))

print(legit_summary_mert)

```

```

## # A tibble: 14 x 2
##   unique_name                trytolegit_binary
##   <chr>                      <dbl>
## 1 Aug-10_mertensia_10_flavifrons      4
## 2 Aug-10_mertensia_3_flavifrons       1
## 3 Aug-10_mertensia_7_flavifrons       4
## 4 Aug-10_mertensia_8_flavifrons       3
## 5 Aug-11_mertensia_1_flavifrons       5
## 6 Aug-11_mertensia_3_flavifrons       2
## 7 Aug-14_mertensia_10_flavifrons      0
## 8 Aug-14_mertensia_1_flavifrons       2
## 9 Aug-14_mertensia_2_flavifrons       1

```

```
## 10 Aug-15_mertensia_1_flavifrons      0
## 11 Aug-15_mertensia_2_flavifrons      0
## 12 Aug-15_mertensia_4_flavifrons      5
## 13 Aug-15_mertensia_6_flavifrons      2
## 14 Aug-15_mertensia_8_flavifrons      0
```

```
visit_summary_mert <- mertensia %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))

visit_summary_mert
```

```
## # A tibble: 14 x 3
##   unique_name      total_visits legit_visits
##   <chr>          <int>      <int>
## 1 Aug-10_mertensia_10_flavifrons      14         4
## 2 Aug-10_mertensia_3_flavifrons       1         1
## 3 Aug-10_mertensia_7_flavifrons      23         4
## 4 Aug-10_mertensia_8_flavifrons      18         3
## 5 Aug-11_mertensia_1_flavifrons       9         5
## 6 Aug-11_mertensia_3_flavifrons       4         2
## 7 Aug-14_mertensia_10_flavifrons       1         0
## 8 Aug-14_mertensia_1_flavifrons      13         2
## 9 Aug-14_mertensia_2_flavifrons       2         1
## 10 Aug-15_mertensia_1_flavifrons       9         0
## 11 Aug-15_mertensia_2_flavifrons       4         0
## 12 Aug-15_mertensia_4_flavifrons      10         5
## 13 Aug-15_mertensia_6_flavifrons       3         2
## 14 Aug-15_mertensia_8_flavifrons       6         0
```

```
summary_mertensia <- mertensia %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))

# Create a new column indicating whether trytolegit_binary > 0
legit_summary_mert <- legit_summary_mert %>%
  mutate(trytolegit_gt_0 = ifelse(trytolegit_binary > 0, "Trytolegit > 0", "Trytolegit = 0"))

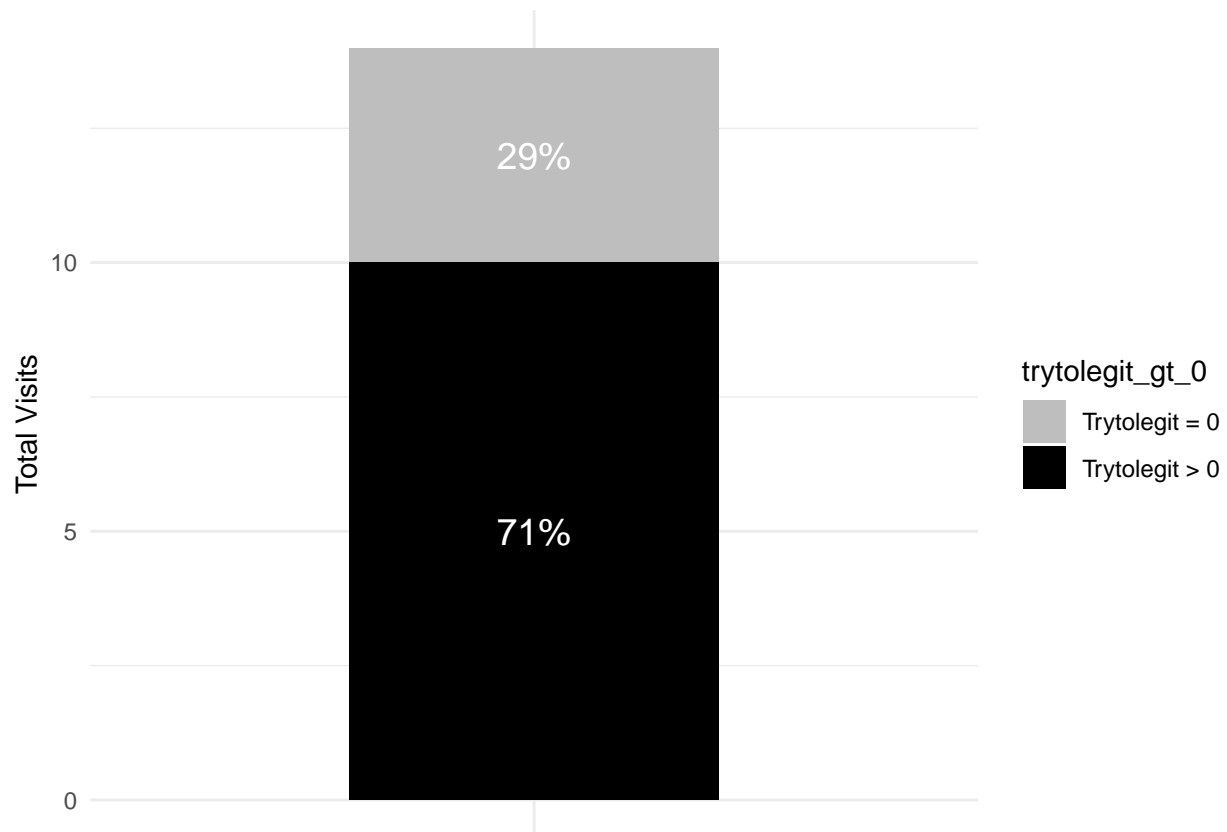
# Summarize the data by the new column
summary_data_mert <- legit_summary_mert %>%
  group_by(trytolegit_gt_0) %>%
  summarise(total_visits = n())

# Calculate total visits
total_visits_mert <- sum(summary_data_mert$total_visits)

# Calculate percentage of visits for each category
summary_data_mert <- summary_data_mert %>%
  mutate(percentage = total_visits / sum(total_visits))
```



```
# Plot the barplot
ggplot(summary_data_mert, aes(x = "", y = total_visits, fill = trytolegit_gt_0)) +
  geom_bar(stat = "identity", width = 0.5) +
  geom_text(aes(label = scales::percent(percentage)),
            position = position_stack(vjust = 0.5),
            color = "white", size = 5) +
  labs(x = NULL, y = "Total Visits") +
  scale_fill_manual(values = c("Trytolegit = 0" = "gray", "Trytolegit > 0" = "black")) +
  theme_minimal() +
  theme(axis.text.x=element_blank()) # Hide x-axis label
```



```
summary_cory_bif <- corydalis_bifarius %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))
legit_summary_corybif <- corydalis_bifarius %>%
  group_by(unique_name) %>%
  summarise(trytolegit_binary = sum(trytolegit_binary))

visit_summary_corybif <- corydalis_bifarius %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))
```

```

# Create a new column indicating whether trytolegit_binary > 0
legit_summary_corybif <- legit_summary_corybif %>%
  mutate(trytolegit_gt_0 = ifelse(trytolegit_binary > 0, "Trytolegit > 0", "Trytolegit = 0"))

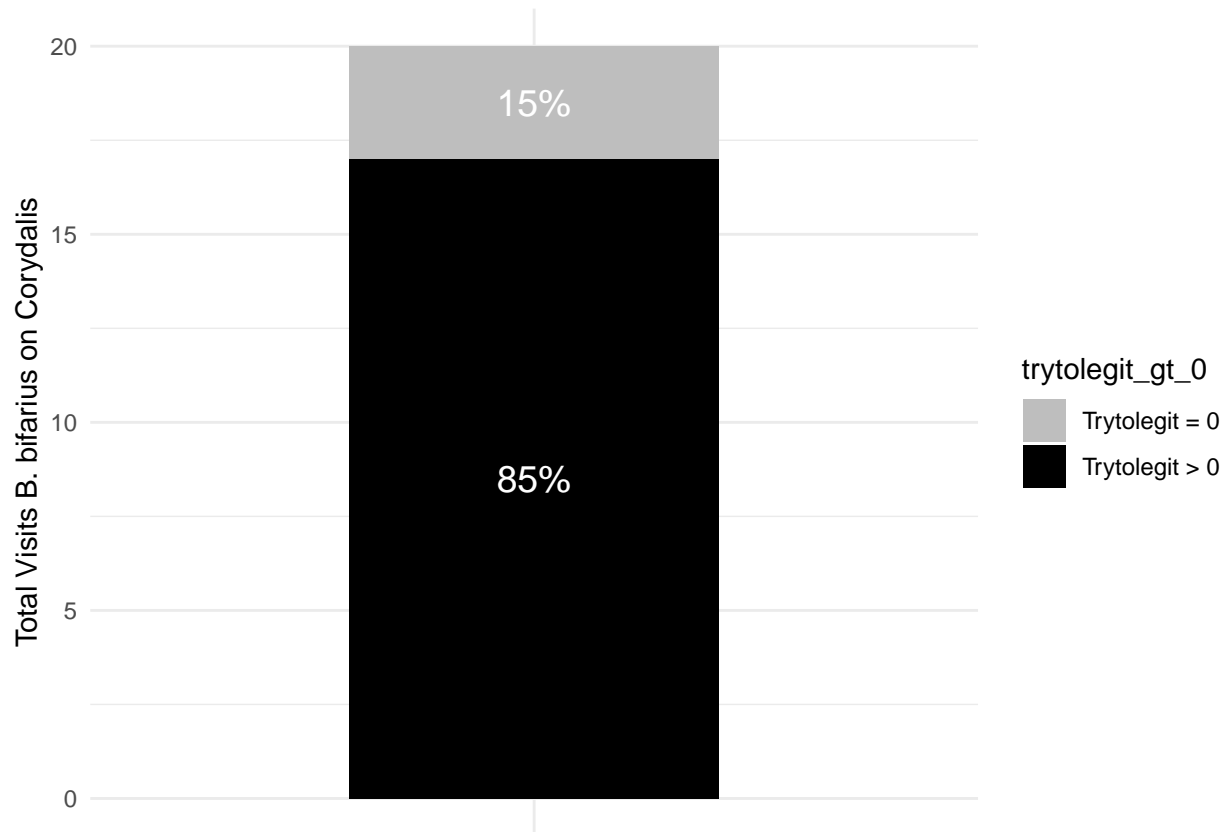
# Summarize the data by the new column
summary_data_corybif <- legit_summary_corybif %>%
  group_by(trytolegit_gt_0) %>%
  summarise(total_visits = n())

# Calculate total visits
total_visits_corybif <- sum(summary_data_corybif$total_visits)

# Calculate percentage of visits for each category
summary_data_corybif <- summary_data_corybif %>%
  mutate(percentage = total_visits / sum(total_visits))

# Plot the barplot
ggplot(summary_data_corybif, aes(x = "", y = total_visits, fill = trytolegit_gt_0)) +
  geom_bar(stat = "identity", width = 0.5) +
  geom_text(aes(label = scales::percent(percentage)),
            position = position_stack(vjust = 0.5),
            color = "white", size = 5) +
  labs(x = NULL, y = "Total Visits B. bifarius on Corydalis") +
  scale_fill_manual(values = c("Trytolegit = 0" = "gray", "Trytolegit > 0" = "black")) +
  theme_minimal() +
  theme(axis.text.x=element_blank()) # Hide x-axis label

```



```
summary_cory_flav <- corydalis_flavifrons %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))
legit_summary_coryflav <- corydalis_flavifrons %>%
  group_by(unique_name) %>%
  summarise(trytolegit_binary = sum(trytolegit_binary))

visit_summary_coryflav <- corydalis_flavifrons %>%
  group_by(unique_name) %>%
  summarise(total_visits = n(),
            legit_visits = sum(trytolegit == 'yes'))

# Create a new column indicating whether trytolegit_binary > 0
legit_summary_coryflav <- legit_summary_coryflav %>%
  mutate(trytolegit_gt_0 = ifelse(trytolegit_binary > 0, "Trytolegit > 0", "Trytolegit = 0"))

# Summarize the data by the new column
summary_data_coryflav <- legit_summary_coryflav %>%
  group_by(trytolegit_gt_0) %>%
  summarise(total_visits = n())

# Calculate total visits
total_visits_coryflav <- sum(summary_data_coryflav$total_visits)
```

```

# Calculate percentage of visits for each category
summary_data_coryflav <- summary_data_coryflav %>%
  mutate(percentage = total_visits / sum(total_visits))

# Plot the barplot
ggplot(summary_data_coryflav, aes(x = "", y = total_visits, fill = trytolegit_gt_0)) +
  geom_bar(stat = "identity", width = 0.5) +
  geom_text(aes(label = scales::percent(percentage)),
            position = position_stack(vjust = 0.5),
            color = "white", size = 5) +
  labs(x = NULL, y = "Total Visits B. flavifrons on Corydalis") +
  scale_fill_manual(values = c("Trytolegit = 0" = "gray", "Trytolegit > 0" = "black")) +
  theme_minimal() +
  theme(axis.text.x=element_blank()) # Hide x-axis label

```

