Characterization of and selection on compound within-individual floral variation in *Vicia americana* (Fabaceae)

Mason W. Kulbaba[*]

June 17, 2025

Contents

[*] St. Mary's University, mason.kulbaba@stmu.ca, https://orcid.org/0000-0003-0619-7089

Abstract

This document provides code to reproduce all results from the manuscript `Characterization of and selection on compound within-individual floral variation in *Vicia americana* (Fabaceae)`. The data file `vicia_final_data.csv` contains all data required to reproduce all results in the manuscript, and is located in the associated Zenodo repository. This study sought to describe the floral traits of *Vicia americana* as compound function-valued traits, and compare standardized linear selection estimates (e.g., β) as per Lande and Arnold (1983), with the functional regression approached used by Kulbaba, Clocher, and Harder (2017) and Harder et al. (2019).

1 R

- The version of R used to make this document is 4.5.0.
- The version of the `rmarkdown` package used to make this document is 2.29.
- The version of the `bookdown` package used to make this document is 0.43.
- The version of the `dplyr` package used to make this document is 1.1.4.
- The version of the `glmmTMB` package used to make this document is 1.1.11.
- The version of the `DHARMa` package used to make this document is 0.4.7.
- The version of the `car` package used to make this document is 3.1.3.
- The version of the `caret` package used to make this document is 7.0.1.
- The version of the `Hmisc` package used to make this document is 5.2.3.
- The version of the `tidyr` package used to make this document is 1.3.1.
- The version of the `viridis` package used to make this document is 0.6.5.
- The version of the `refund` package used to make this document is 0.1.37.
- The version of the `mgcv` package used to make this document is 1.9.3.
- The version of the `tibble` package used to make this document is 3.2.1. Attach packages.

```r
suppressMessages(library("dplyr"))
suppressMessages(library("glmmTMB"))
suppressMessages(library("ggplot2"))
suppressMessages(library("DHARMa"))
suppressMessages(library("car"))
suppressMessages(library("caret"))
suppressMessages(library("Hmisc"))
suppressMessages(library("tidyr"))
suppressMessages(library("viridis"))
suppressMessages(library("refund"))
suppressMessages(library("mgcv"))
suppressMessages(library("tibble"))
```

2 Data

Load data file

```r
data<- read.csv("vicia_final_data.csv")
```

where the variables are

- `PlantID` is a unique numerical identifier for each individual in the study (1-40).
- `Branch` is a unique numerical identifier for each sequentially produced raceme (1-10). The first raceme to flower was designated as 1, and was the most basal.
- `PosSeq` is the sequential flower position (1-49) across all sequentially flowering racemes.
- `BPos` is a composite of `Branch` and `Pos` (see below), indicating the raceme-specific flower position.
- `Pos` is the individual flower position within each raceme.
- `FL` is the length of flower.
- `FD` is the diameter of the flower where the banner petal attaches.
- `B` is the length (height) of the banner petal.
- `Date` is the date of flower opening, and when the three floral measurements were made.
- `flw_date` is the numerical day of the flowering season (1-17) the flower opened.
- `FlwFate` is whether or not a flower produced fruit (0 = no, 1 = yes).
- `seeds` is the number of seed produced in a given fruit.
- `aborted` is the number of aborted embryos.
- `unfert` is the number of unfertilized ovules.
- `Notes` records any specific notes for a given flower.
- `flw_vol` is flower volume as approximated as a cone ($V = \frac{1}{3}\pi\frac{FD^2}{2}FL$)

3 Standardized Linear Selection (e.g., Lande and Arnold (1983))

3.1 Relative fitness (seeds)

```r
#make sure PlantID is a factor
data$PlantID<- as.factor(data$PlantID)

#calculate total seed set (fitness) at plant level
plant.seeds<- aggregate(data$seeds, by=list(data$PlantID), sum)

#reset column names
colnames(plant.seeds)<- c("PlantID", "tot_seeds")

#calculate relative fitness
```

```
plant.seeds$rel_seeds<- plant.seeds$tot_seeds/(mean(plant.seeds$tot_seeds, na.rm=T))

#Check
head(plant.seeds)
```

```
##   PlantID tot_seeds rel_seeds
## 1       1        16  1.412804
## 2       2        21  1.854305
## 3       3        21  1.854305
## 4       4         0  0.000000
## 5       5         0  0.000000
## 6       6         0  0.000000
```

3.2 Standardized traits

First need to calculate mean values for each floral trait, and then subtract the mean and divide by the trait standard deviation to standardize each traits for each individual plant.

```
#First calculate mean trait value for each trait (yes, not efficient, but I like to see the steps)
mean.B<- aggregate(data$B, by=list(data$PlantID), mean, na.rm=T)
mean.B$Group.1 <- NULL
colnames(mean.B)<- "mean.B"

mean.FL<- aggregate(data$FL, by=list(data$PlantID), mean, na.rm=T)
mean.FL$Group.1 <- NULL
colnames(mean.FL)<- "mean.FL"

mean.FD<- aggregate(data$FD, by=list(data$PlantID), mean, na.rm=T)
colnames(mean.FD)<- c("PlantID", "mean.FD")
```

Merge into a single dataframe (I know this is not efficient, I like to see the steps) with relative seed set

```
traits<- cbind(mean.B, mean.FL, mean.FD)

# add relative seed set
sel.data<- merge(traits, plant.seeds)

#check
sel.data
```

```
##    PlantID   mean.B   mean.FL  mean.FD tot_seeds rel_seeds
## 1        1 4.384872  8.513333 2.654615        16 1.4128035
## 2       10 6.617778  8.874444 3.336667         5 0.4415011
## 3       11 6.621667  8.786667 3.090000         0 0.0000000
## 4       12 7.206667 10.253333 3.389333         0 0.0000000
## 5       13 6.604706  9.202353 3.228235         0 0.0000000
## 6       14 5.826667  7.611667 3.036667         0 0.0000000
## 7       15 5.108750  8.624750 2.938750        45 3.9735099
## 8       16 6.948000  9.286000 3.758000         0 0.0000000
## 9       17 5.388571  9.061905 2.906190         0 0.0000000
## 10      18 4.353095  8.365714 2.548571        14 1.2362031
## 11      19 4.856389  8.720278 2.658056        19 1.6777042
## 12       2 5.508214  9.956071 3.398929        21 1.8543046
## 13      20 6.661429  9.056250 2.978750         5 0.4415011
## 14      21 6.302857  9.892857 3.171429        19 1.6777042
## 15      22 5.687857  9.230000 2.846429        10 0.8830022
## 16      23 6.930909 10.366364 3.241818         6 0.5298013
## 17      24 5.751667  9.445833 3.019167         0 0.0000000
## 18      25 6.131818  8.886364 3.013636        36 3.1788079
## 19      26 6.278095  9.806667 3.033333         0 0.0000000
## 20      27 6.824000 10.014000 2.988000        14 1.2362031
## 21      28 5.033333  8.496667 2.744000         7 0.6181015
## 22      29 4.633750  8.248750 2.838750         0 0.0000000
## 23       3 5.716111  9.909167 3.252500        21 1.8543046
## 24      30 5.400000  8.715556 2.953333        15 1.3245033
## 25      31 5.473333  8.485000 2.666667         8 0.7064018
## 26      32 4.132222  8.714815 2.558148        15 1.3245033
## 27      33 4.903200  8.210400 2.783600        10 0.8830022
## 28      34 4.875789  8.818421 2.684211        15 1.3245033
## 29      35 4.104706  7.813529 2.659412        43 3.7969095
## 30      36 4.860000  8.016429 2.556429         0 0.0000000
## 31      37 4.471667  8.850000 2.684444        17 1.5011038
## 32      38 3.898750  8.801250 2.686875        37 3.2671082
## 33      39 3.831429  7.962449 2.443673        32 2.8256071
## 34       4 5.460000 10.010476 3.026190         0 0.0000000
## 35      40 3.098571  7.821429 2.265714         4 0.3532009
## 36       5 6.362000  9.179000 3.379000         0 0.0000000
## 37       6 8.326667 10.480000 4.475000         0 0.0000000
## 38       7 6.560556  9.313333 3.035000        15 1.3245033
## 39       8 6.854706  8.866471 3.434706         0 0.0000000
## 40       9 7.553333  9.436667 3.492222         4 0.3532009
```

Now need to standardize individual plant mean (from above).

```
#Calculate total (population) mean for each trait
sel.data$B_z<- (sel.data$mean.B - mean(sel.data$mean.B, na.rm = T))/sd(sel.data$mean.B, na.rm = T)
sel.data$FL_z<- (sel.data$mean.FL - mean(sel.data$mean.FL, na.rm = T))/sd(sel.data$mean.FL, na.rm = T)
sel.data$FD_z<- (sel.data$mean.FD - mean(sel.data$mean.FD, na.rm = T))/sd(sel.data$mean.FD, na.rm = T)
```

3.3 Covariates

```
# total flowers
tot.flw<- aggregate(data$PosSeq, by=list(data$PlantID), max)
tot.flw$Group.1<- NULL

#total branches (racemes)
tot.branch<- aggregate(data$Branch, by=list(data$PlantID), max)
tot.branch$Group.1<- NULL
```

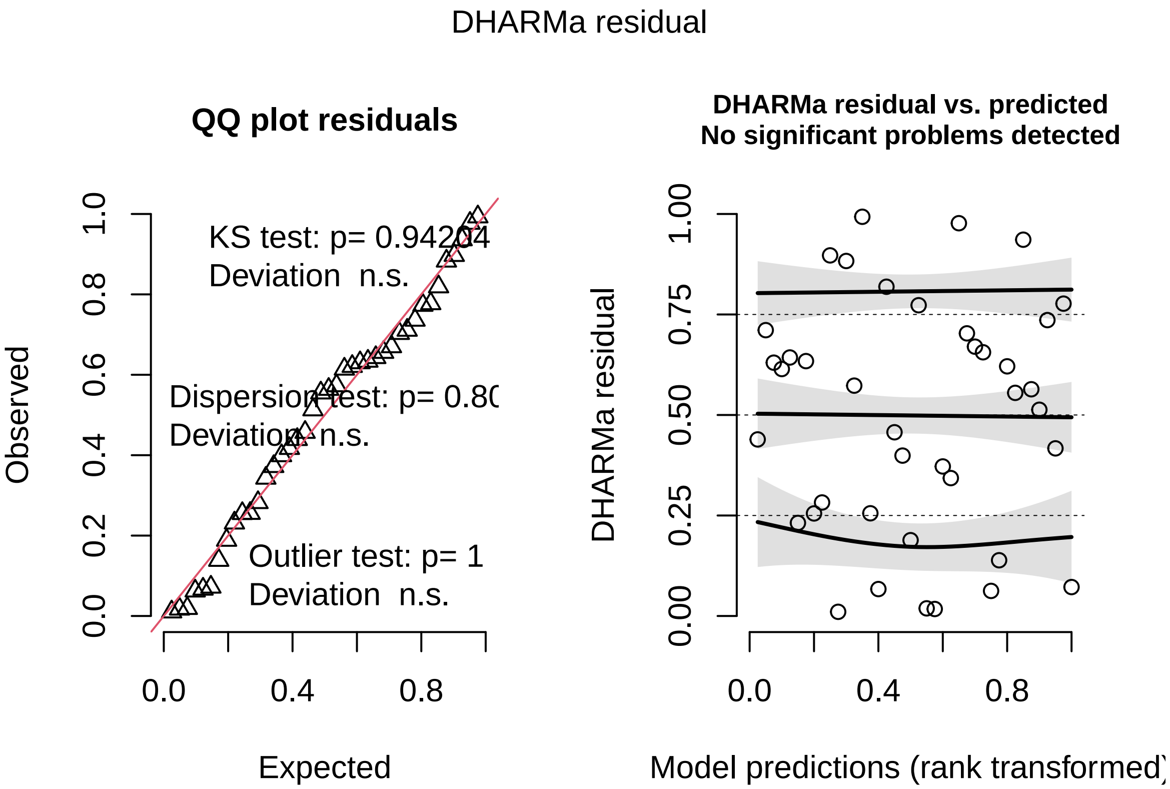
3.4 Estimate (β)

Start with a poisson distribution.

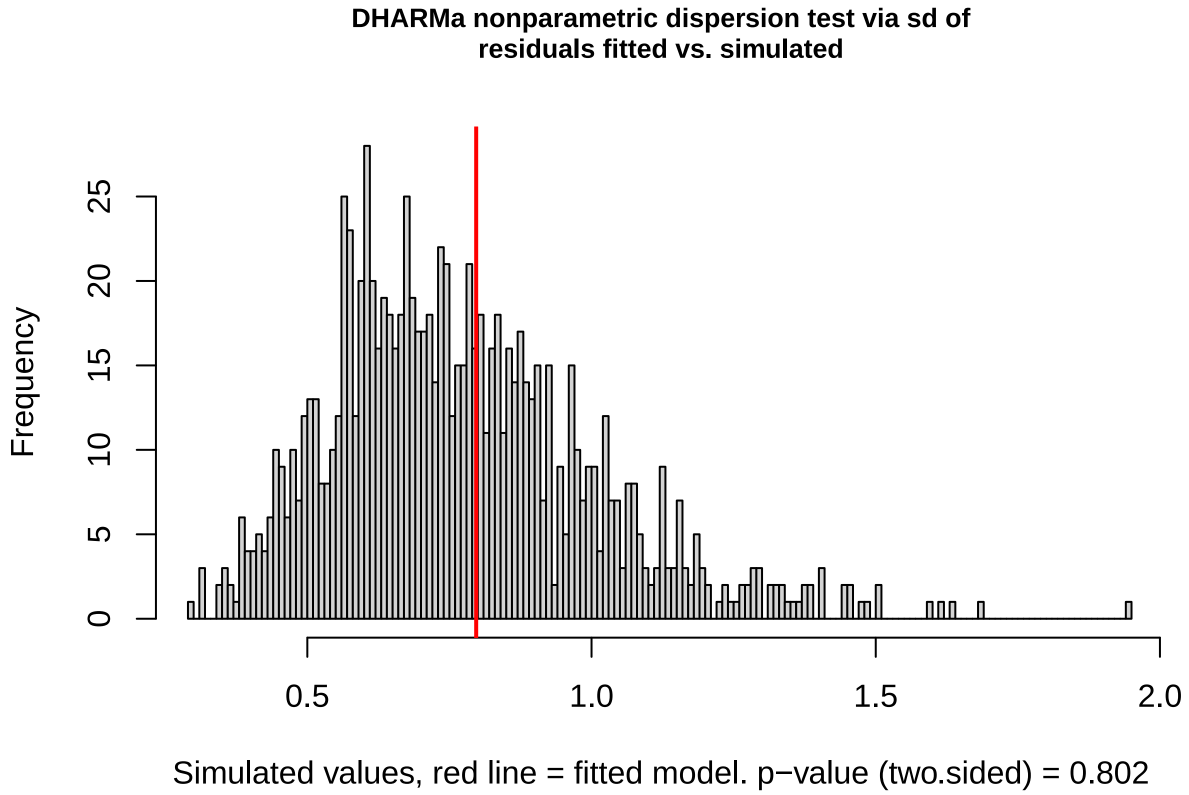
```
# model with standardized traits as fixed effects, and paIntID as random
# Fit Poisson model
fit_pois <- glmmTMB(rel_seeds ~ B_z + FL_z + FD_z,
                    data = sel.data, family = poisson)
```

```
## Warning in glmmTMB(rel_seeds ~ B_z + FL_z + FD_z, data = sel.data, family =
## poisson): non-integer counts in a poisson model
```

```
# Model diagnostics using DHARMa
sim_resid <- simulateResiduals(fit_pois, n = 1000)
plot(sim_resid)
```



```
# Test for overdispersion
testDispersion(sim_resid)
```

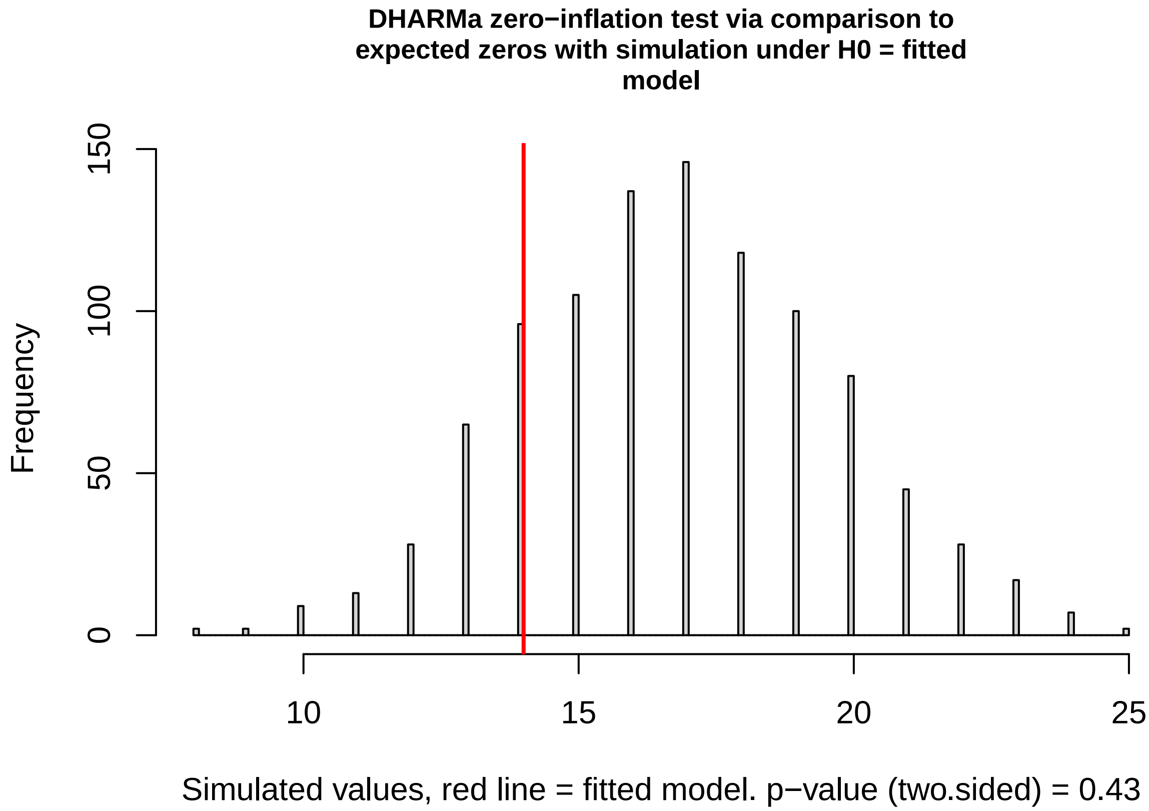


```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## dispersion = 1.0352, p-value = 0.802
## alternative hypothesis: two.sided
```

```
summary(fit_pois)
```

```
##  Family: poisson  ( log )
## Formula:          rel_seeds ~ B_z + FL_z + FD_z
## Data: sel.data
##
##      AIC       BIC    logLik -2*log(L)  df.resid
##    105.4     112.2     -48.7      97.4        36
##
##
## Conditional model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.14262    0.18035  -0.791   0.4290
## B_z         -0.73068    0.36197  -2.019   0.0435 *
## FL_z         0.20030    0.25371   0.789   0.4298
## FD_z         0.08304    0.41486   0.200   0.8414
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Formal test for zero inflation
testZeroInflation(sim_resid) # not significant
```



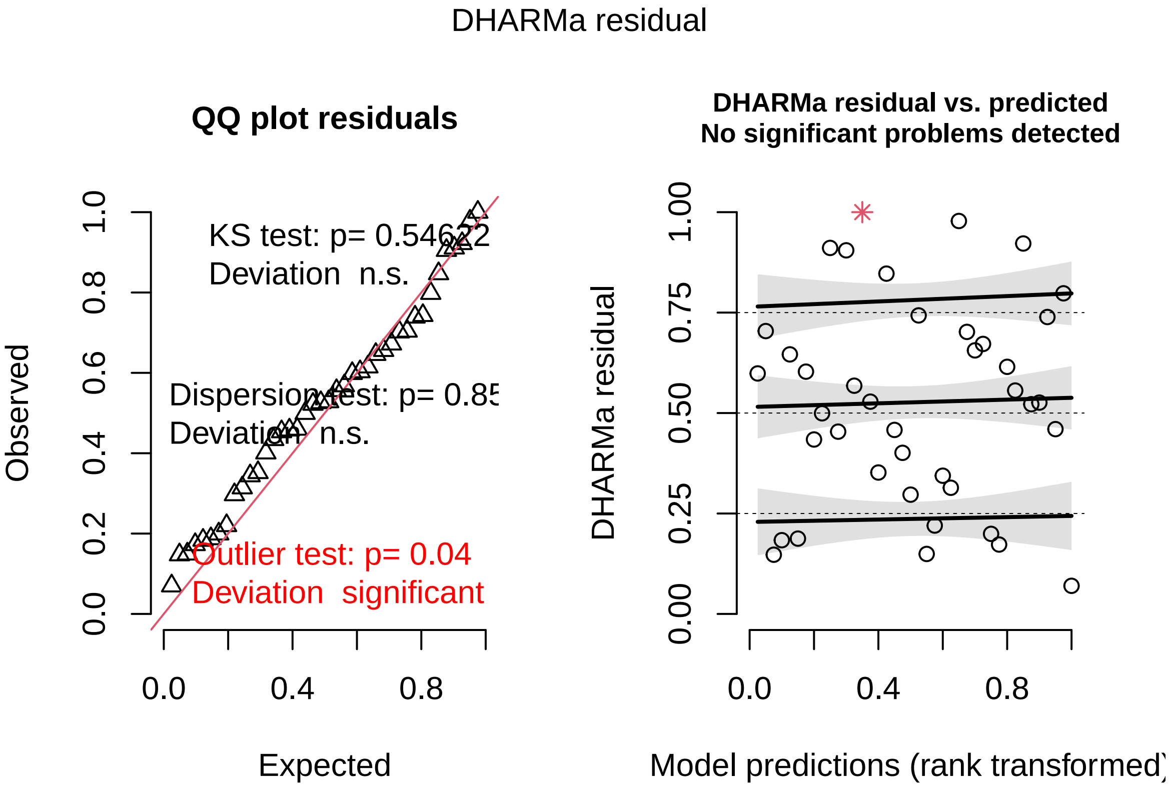
```
##
##  DHARMa zero-inflation test via comparison to expected zeros with
##  simulation under H0 = fitted model
##
## data:  simulationOutput
## ratioObsSim = 0.8316, p-value = 0.43
## alternative hypothesis: two.sided
```

The above model looks like a good fit (according to diagnostics), and not over dispersed. However, try fitting with a negative binomial distribution and compare AIC across two models.

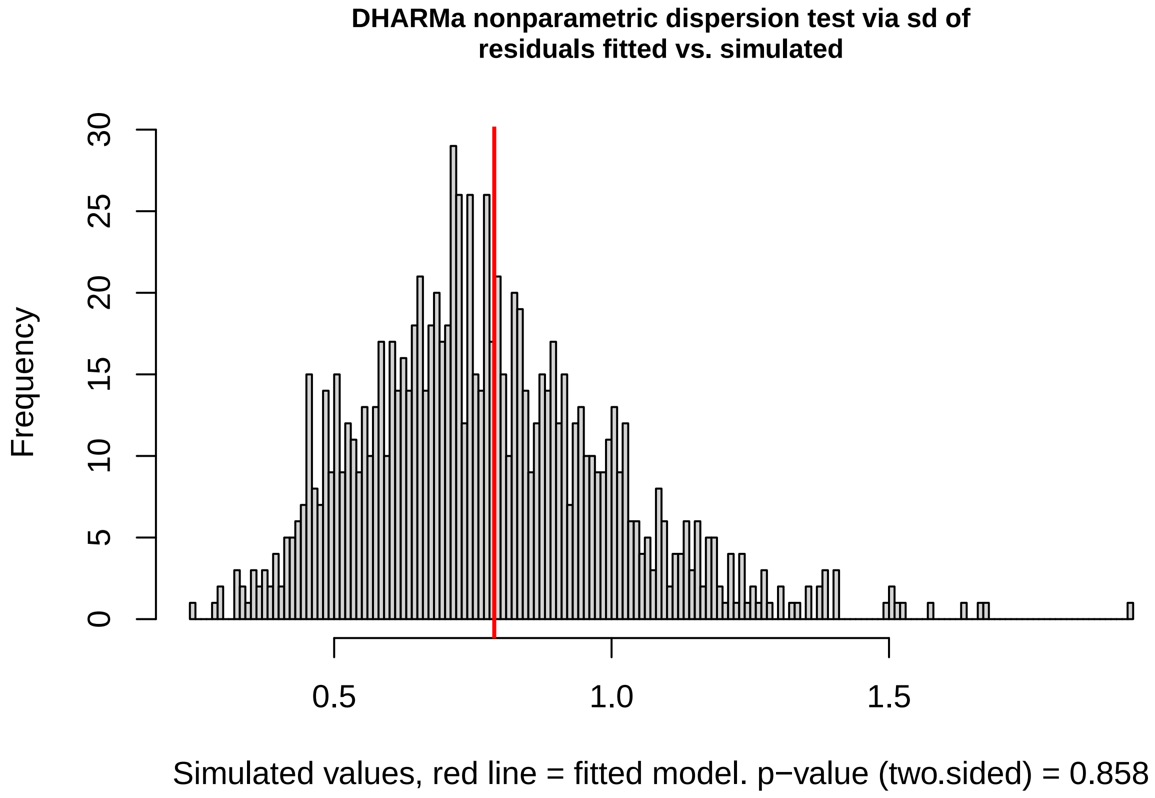
```
# Fit negative binomial model
fit_nb <- glmmTMB(rel_seeds ~ B_z + FL_z + FD_z,
                  data = sel.data, family = nbinom2)
```

```
## Warning in glmmTMB(rel_seeds ~ B_z + FL_z + FD_z, data = sel.data, family =
## nbinom2): non-integer counts in a nbinom2 model
```

```r
# Model diagnostics using DHARMa
sim_resid <- simulateResiduals(fit_nb, n = 1000)
plot(sim_resid)
```



```r
# Test for overdispersion
testDispersion(sim_resid)
```

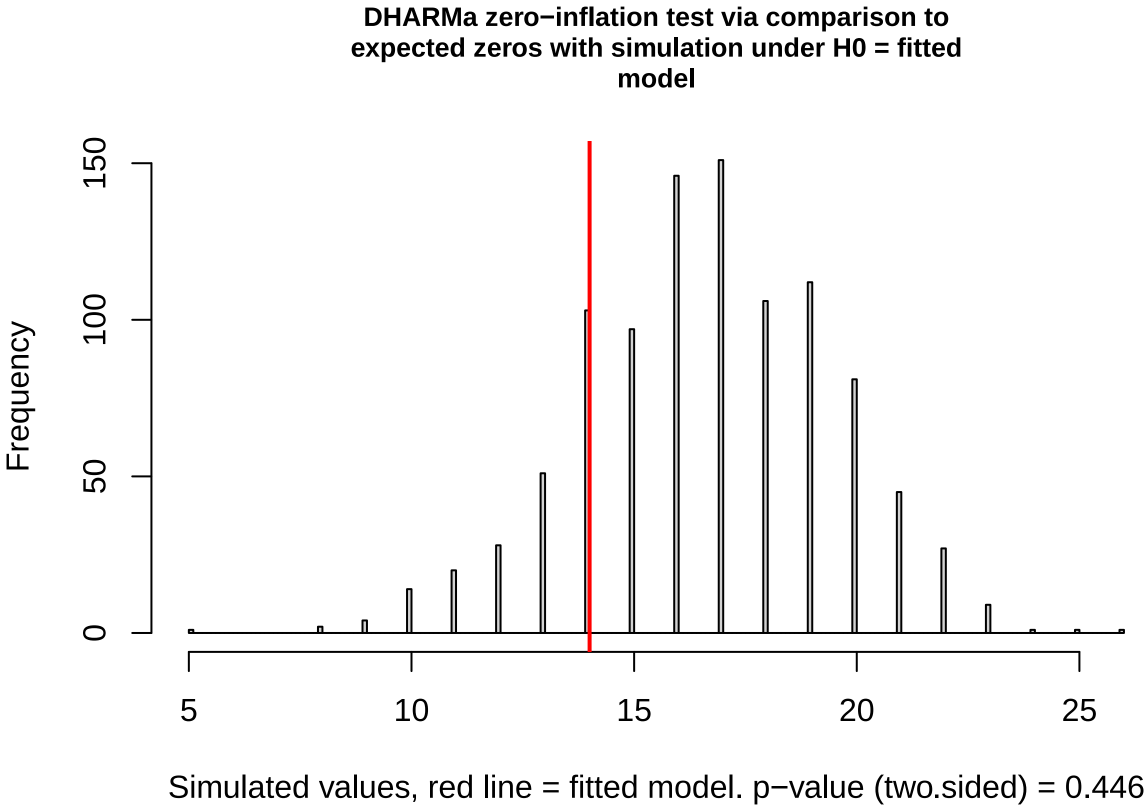


```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## dispersion = 1.0168, p-value = 0.858
## alternative hypothesis: two.sided
```

```
summary(fit_nb)
```

```
##  Family: nbinom2  ( log )
## Formula:          rel_seeds ~ B_z + FL_z + FD_z
## Data: sel.data
##
##      AIC       BIC    logLik -2*log(L)  df.resid
##    107.4     115.8     -48.7      97.4        35
##
##
## Dispersion parameter for nbinom2 family (): 1.32e+08
##
## Conditional model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.14262    0.18035  -0.791   0.4291
## B_z         -0.73068    0.36197  -2.019   0.0435 *
## FL_z         0.20030    0.25371   0.789   0.4298
## FD_z         0.08304    0.41486   0.200   0.8414
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Formal test for zero inflation
testZeroInflation(sim_resid) # not significant
```



```
##
##  DHARMa zero-inflation test via comparison to expected zeros with
##  simulation under H0 = fitted model
##
## data:  simulationOutput
## ratioObsSim = 0.83867, p-value = 0.446
## alternative hypothesis: two.sided
```

```
AIC(fit_pois, fit_nb)
```

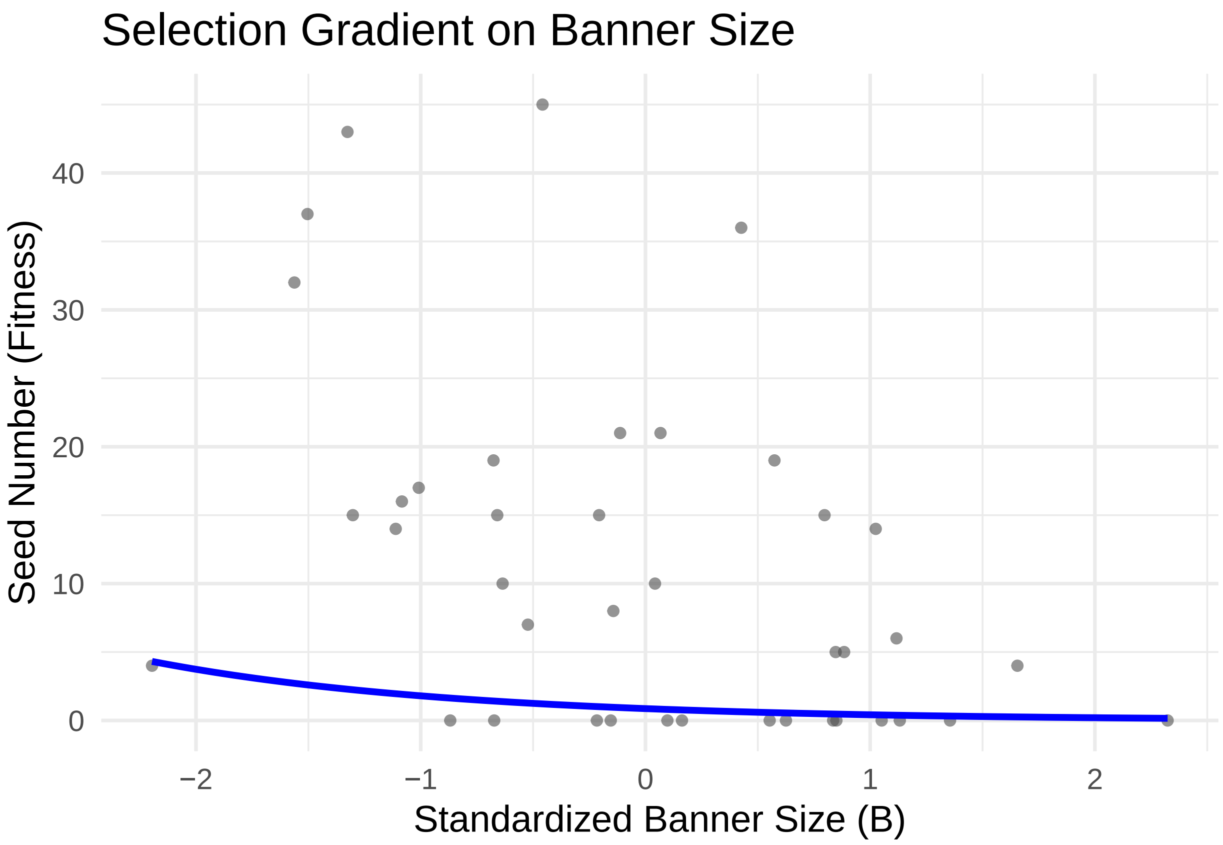
```
##          df      AIC
## fit_pois  4 105.4014
## fit_nb    5 107.4014
```

Both models fit well, and show the same pattern (significant effect of Banner height). As the AIC is slightly smaller with Poisson distribution, use this model.

Now produce a quick plot of the significant effect of banner height. A rather underwhelming figure.

```r
# Create prediction data over the range of standardized B
newdata <- data.frame(
  B_z = seq(min(sel.data$B_z), max(sel.data$B_z), length.out = 100),
  FL_z = 0,  # Hold other traits at their means (0 after standardization)
  FD_z = 0
)

# Predict expected seed number from the Poisson model
newdata$predicted_seeds <- predict(fit_pois, newdata, type = "response")

# Plot observed data and predicted curve
ggplot(sel.data, aes(x = B_z, y = tot_seeds)) +
  geom_point(alpha = 0.6, color = "gray30") +
  geom_line(data = newdata, aes(x = B_z, y = predicted_seeds), color = "blue", size = 1.2) +
  labs(
    x = "Standardized Banner Size (B)",
    y = "Seed Number (Fitness)",
    title = "Selection Gradient on Banner Size"
  ) +
  theme_minimal(base_size = 14)
```

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



4 Floral Integration

Floral integration was described with correlation coefficients among floral traits. To explore if correlations among traits change across racemes, we compared correlation coefficients on racemes 1-5. To facilitate comparison among racemes, the first five flowers were used to calculate these correlations. We calculated both within racemes (first five flowers), and among racemes (same position across first five racemes).

```
# define function to extract, r, se, and P-value
get_cor_stats <- function(x, y) {
  ct <- cor.test(x, y, method = "pearson")
  r <- ct$estimate
  n <- sum(complete.cases(x, y))
  se <- sqrt((1 - r^2) / (n - 2))
  data.frame(correlation = r, se = se, p_value = ct$p.value)
}
```

4.1 Withn raceme integration

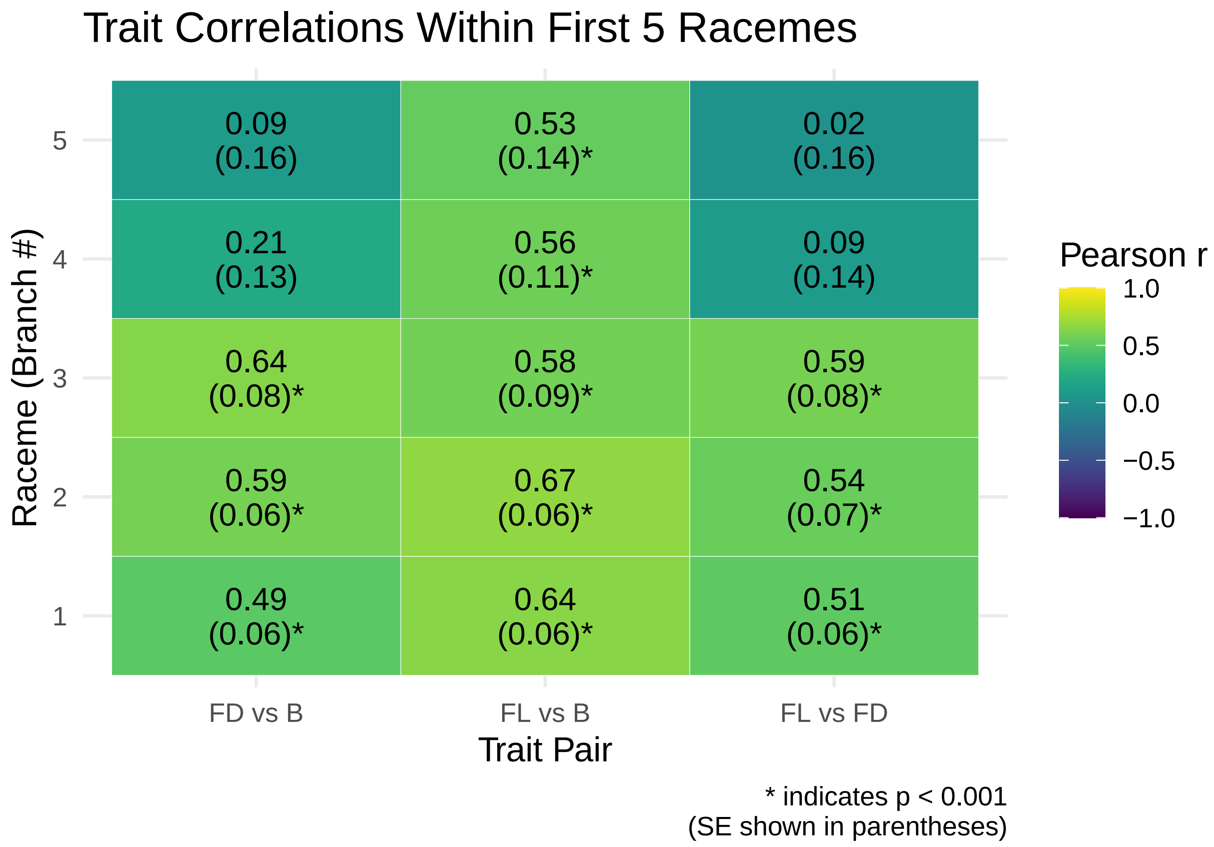
Calculate within-inflorescence (raceme) floral integration.

```
#within raceme integration
within_raceme <- data %>%
  filter(Branch %in% 1:5, Pos %in% 1:5) %>%
  group_by(Branch) %>%
  group_modify(~{
    df <- .
    bind_rows(
      get_cor_stats(df$FL, df$FD) %>% mutate(pair = "FL vs FD"),
      get_cor_stats(df$FL, df$B)  %>% mutate(pair = "FL vs B"),
      get_cor_stats(df$FD, df$B)  %>% mutate(pair = "FD vs B")
    )
  }) %>%
  ungroup() %>%
  select(Branch, pair, correlation, se, p_value)

within_raceme
```

```
## # A tibble: 15 x 5
##    Branch pair     correlation     se  p_value
##     <int> <chr>          <dbl>  <dbl>    <dbl>
##  1      1 FL vs FD      0.508  0.0625 5.48e-14
##  2      1 FL vs B       0.643  0.0555 8.08e-24
##  3      1 FD vs B       0.491  0.0632 4.86e-13
##  4      2 FL vs FD      0.542  0.0671 1.57e-13
##  5      2 FL vs B       0.671  0.0591 3.44e-22
##  6      2 FD vs B       0.586  0.0646 4.60e-16
##  7      3 FL vs FD      0.588  0.0843 4.51e-10
##  8      3 FL vs B       0.578  0.0851 1.04e- 9
##  9      3 FD vs B       0.636  0.0804 5.53e-12
## 10      4 FL vs FD      0.0902 0.136  5.09e- 1
## 11      4 FL vs B       0.562  0.113  6.43e- 6
## 12      4 FD vs B       0.209  0.133  1.22e- 1
## 13      5 FL vs FD      0.0176 0.164  9.15e- 1
## 14      5 FL vs B       0.531  0.139  5.02e- 4
## 15      5 FD vs B       0.0908 0.164  5.82e- 1
```

Now make a nice little table with heatmap features to show pattern of floral integration with racemes, across the first five racemes.

```
# prepare standard errors for inclusoin in table/heatmap
within_raceme <- within_raceme %>%
  mutate(sig = ifelse(p_value < 0.05, "*", ""),
         label = sprintf("%.2f\n(%.2f)%s", correlation, se, sig))

ggplot(within_raceme, aes(x = pair, y = factor(Branch), fill = correlation)) +
  geom_tile(color = "white") +
  geom_text(aes(label = label), color = "black", size = 4.2, lineheight = 0.9) +
  scale_fill_viridis(name = "Pearson r", limits = c(-1, 1)) +
  labs(
    title = "Trait Correlations Within First 5 Racemes",
    x = "Trait Pair", y = "Raceme (Branch #)",
    caption = "* indicates p < 0.001\n(SE shown in parentheses)"
  ) +
  theme_minimal(base_size = 13)
```



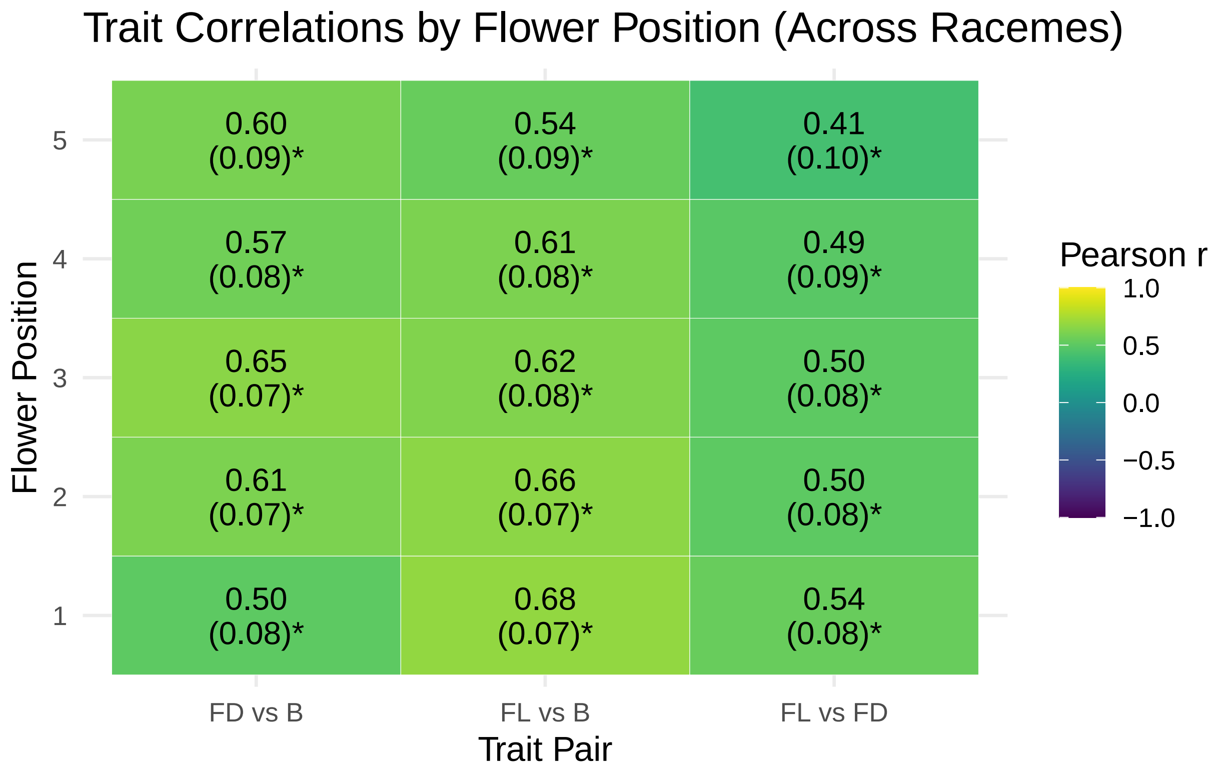
4.2 Among raceme integration

```
# among racemes
across_pos <- data %>%
  filter(Branch %in% 1:5, Pos %in% 1:5) %>%
  group_by(Pos) %>%
  group_modify(~{
    df <- .
    bind_rows(
      get_cor_stats(df$FL, df$FD) %>% mutate(pair = "FL vs FD"),
      get_cor_stats(df$FL, df$B)  %>% mutate(pair = "FL vs B"),
      get_cor_stats(df$FD, df$B)  %>% mutate(pair = "FD vs B")
    )
  }) %>%
  ungroup() %>%
  select(Pos, pair, correlation, se, p_value)

across_pos
```

```
## # A tibble: 15 x 5
##       Pos pair     correlation     se  p_value
##     <int> <chr>          <dbl>  <dbl>    <dbl>
##  1      1 FL vs FD       0.539 0.0785 3.54e-10
##  2      1 FL vs B        0.677 0.0686 5.25e-17
##  3     1 FD vs B        0.502 0.0806 7.90e- 9
##  4     2 FL vs FD       0.502 0.0803 6.83e- 9
##  5     2 FL vs B        0.657 0.0700 6.79e-16
##  6     2 FD vs B        0.607 0.0738 3.32e-13
##  7     3 FL vs FD       0.502 0.0836 2.67e- 8
##  8     3 FL vs B        0.623 0.0756 4.62e-13
##  9     3 FD vs B        0.651 0.0734 1.84e-14
## 10     4 FL vs FD       0.486 0.0853 1.11e- 7
## 11     4 FL vs B        0.607 0.0776 4.27e-12
## 12     4 FD vs B        0.570 0.0802 1.52e-10
## 13     5 FL vs FD       0.407 0.0979 7.58e- 5
## 14     5 FL vs B        0.536 0.0905 6.20e- 8
## 15     5 FD vs B        0.596 0.0861 7.09e-10
```

Similar table as before, but now for same flower position (1-5) position across subsequently produced racemes.

```
across_pos <- across_pos %>%
  mutate(sig = ifelse(p_value < 0.05, "*", ""),
         label = sprintf("%.2f\n(%.2f)%s", correlation, se, sig))

ggplot(across_pos, aes(x = pair, y = factor(Pos), fill = correlation)) +
  geom_tile(color = "white") +
  geom_text(aes(label = label), color = "black", size = 4.2, lineheight = 0.9) +
  scale_fill_viridis(name = "Pearson r", limits = c(-1, 1)) +
  labs(
    title = "Trait Correlations by Flower Position (Across Racemes)",
    x = "Trait Pair", y = "Flower Position",
    caption = "* indicates p < 0.001\n(SE shown in parentheses)"
  ) +
  theme_minimal(base_size = 13)
```


Trait Correlations by Flower Position (Across Racemes)

* indicates p < 0.001
(SE shown in parentheses)

Floral integration at same flower position (1-5) across racemes (1-5) was moderate and consistent. However, floral integration within racemes disappeared with successive racemes for FL-FD and FD-B pairs. Only FL-B maintained a consistent and significant correlation.

5 Describe Subindividual Variation in Floral Traits

The goal is to describe the general pattern of within-individual floral trait variation, across all flower positions and racemes. Therefore, we will be using the variable `PosSeq` that describes the continuous flower position across all racemes on a plant. Generalized additive models (`gam`) from the package `mgcv` will be used to fit spline function to describe trait variation across continuous flower positions.

First, need to standardize the flower positions within each plant. This is done as all plants do not have the same number of flowers. This will be important when applying functional regression, when the functional predictor has a variable domain. Note, we are using thin-plate splines, which are appropriate for variable domains. This will be ***required*** for functional regression, so it is appropriate to describe the patterns with the same basis type. Traditional b-splines have a standardized number of knots across all individuals, which is not appropriate when the number of data points across individuals is variable. Thin-plate splines (`bs="tp"`) allows for more flexibility (with an appropriate penalty) in knot placement, to accommodate the varible number of flowers across individual plants.

```
dat <- data %>%
  group_by(PlantID) %>%
  mutate(pos_scaled = (PosSeq - min(PosSeq)) / (max(PosSeq) - min(PosSeq))) %>%
  ungroup()

dat$PlantID<- as.factor(dat$PlantID)
```

Next, fit GAM models for each trait across PosSeq using plant-specific smooths.

```
# Flower Length (FL)
gam_FL <- gam(FL ~ s(pos_scaled, by = PlantID, bs = "tp") + PlantID, data = dat)



# Flower Diameter (FD)
gam_FD <- gam(FD ~ s(pos_scaled, by = PlantID, bs = "tp") + PlantID, data = dat)



# Banner height (B)
gam_B <- gam(B ~ s(pos_scaled, by = PlantID, bs = "tp") + PlantID, data = dat)
```

5.1 Variation in individual plant trajectories

The above shows that the GAM models describe significant within-individual variation in floral traits. Now see if individual plant trajectories (e.g., patterns of within-individual variation) significantly varies.

Start by fitting only global models (no individual plant effects).

```
# Model 1: Global spline only
mod_global.fl <- gam(FL ~ s(pos_scaled, bs="tp"), data = dat,
                 family = Gamma(link = "log"))

mod_global.fd <- gam(FD ~ s(pos_scaled, bs="tp"), data = dat,
                    family = Gamma(link = "log"))

mod_global.b <- gam(B ~ s(pos_scaled, bs="tp"), data = dat,
                   family = Gamma(link = "log"))
```

Now add individual plant effects.

```
# Model 2: Global spline + plant-specific deviations
mod_plant.fl <- gam(FL ~ s(pos_scaled, bs="tp") + s(pos_scaled, by=PlantID, bs = "tp"), data = dat,
                family = Gamma(link = "log"))

mod_plant.fd <- gam(FD ~ s(pos_scaled, bs="tp") + s(pos_scaled, by=PlantID, bs = "tp"), data = dat,
                   family = Gamma(link = "log"))

mod_plant.b <- gam(B ~ s(pos_scaled, bs="tp") + s(pos_scaled, by=PlantID, bs = "tp"), data = dat,
                  family = Gamma(link = "log"))
```

Now compare global and global + individual plant models

```
# Compare models
anova(mod_global.fl, mod_plant.fl, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: FL ~ s(pos_scaled, bs = "tp")
## Model 2: FL ~ s(pos_scaled, bs = "tp") + s(pos_scaled, by = PlantID, bs = "tp")
##   Resid. Df Resid. Dev   Df Deviance  Pr(>Chi)
## 1    716.38    10.8951
## 2    638.98     8.6972 77.4   2.1979 2.323e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(mod_global.fd, mod_plant.fd, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: FD ~ s(pos_scaled, bs = "tp")
## Model 2: FD ~ s(pos_scaled, bs = "tp") + s(pos_scaled, by = PlantID, bs = "tp")
##   Resid. Df Resid. Dev     Df Deviance  Pr(>Chi)
## 1    715.46     23.064
## 2    631.06     14.179 84.398   8.8846 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(mod_global.b, mod_plant.b, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: B ~ s(pos_scaled, bs = "tp")
## Model 2: B ~ s(pos_scaled, bs = "tp") + s(pos_scaled, by = PlantID, bs = "tp")
##   Resid. Df Resid. Dev     Df Deviance  Pr(>Chi)
## 1    717.36     47.769
## 2    656.67     39.032 60.692   8.7363 8.926e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

All three floral traits show significant differences among individual plant trajectories.

Now include total flowers for each plant as a covariate.

```
# Compute total flowers per PlantID
dat <- data %>%
  group_by(PlantID) %>%
  mutate(total_flw = n()) %>%
  ungroup()

dat2 <- dat %>%
  group_by(PlantID) %>%
  mutate(pos_scaled = (PosSeq - min(PosSeq)) / (max(PosSeq) - min(PosSeq))) %>%
  mutate(total_flw = n()) %>%
  ungroup()

head(dat2)
```

```
## # A tibble: 6 x 18
##   PlantID Branch PosSeq Bpos    Pos    FL    FD     B Date    flw_date FlwFate
##   <fct>    <int>  <int> <chr> <int> <dbl> <dbl> <dbl> <chr>      <int>   <int>
## 1 1             1      1 1-1       1 12.1   6.66  7.63 12-Jun        1       0
## 2 1             1      2 1-2       2  9.07  2.11  5.13 13-Jun        2       0
## 3 1             1      3 1-3       3  9.26  2     5.11 13-Jun        2       0
## 4 1             1      4 1-4       4  7.15  2.65  6.32 13-Jun        2       0
## 5 1             2      5 2-1       1  8.92  3.03  6.1  13-Jun        2       0
## 6 1             2      6 2-2       2  8.77  3.02  5.11 13-Jun        2       0
## # i 7 more variables: seeds <int>, aborted <int>, unfert <int>, Notes <chr>,
## #   flw_vol <dbl>, total_flw <int>, pos_scaled <dbl>
```

Rerun models with covariate of total number of flowers.

```
mod.fl <- gam(FL ~ s(pos_scaled) + s(pos_scaled, by=PlantID, bs = "tp") + total_flw,
           data = dat2,
           family = Gamma(link = "log"))

mod.fd <- gam(FD ~ s(pos_scaled) + s(pos_scaled, by=PlantID, bs = "tp") + total_flw,
           data = dat2,
           family = Gamma(link = "log"))

mod.b <- gam(B ~ s(pos_scaled) + s(pos_scaled, by=PlantID, bs = "tp") + total_flw, data = dat2,
             family = Gamma(link = "log"))
```

Generate predicted values for plotting. Start with a grid to place predicted values.

```
# Generate prediction grid
unique_plants <- unique(data$PlantID)

# Sequence of positions from 0 to 1 (because pos_scaled is standardized)
pos_grid <- seq(0, 1, length.out = 100)

# Total flower counts for each plant
plant_info <- dat2 %>%
  group_by(PlantID) %>%
  summarise(total_flw = first(total_flw))  # or n(), same result here

# Expand grid for predictions for each floral trait
newdata.fl <- expand_grid(
  PlantID = unique(data$PlantID),
  pos_scaled = pos_grid
) %>%
  left_join(plant_info, by = "PlantID")


newdata.fd <- expand_grid(
  PlantID = unique(data$PlantID),
  pos_scaled = pos_grid
) %>%
  left_join(plant_info, by = "PlantID")


newdata.b <- expand_grid(
  PlantID = unique(data$PlantID),
  pos_scaled = pos_grid
) %>%
  left_join(plant_info, by = "PlantID")
```

Now add predicted values from each trait-specific GAM model.

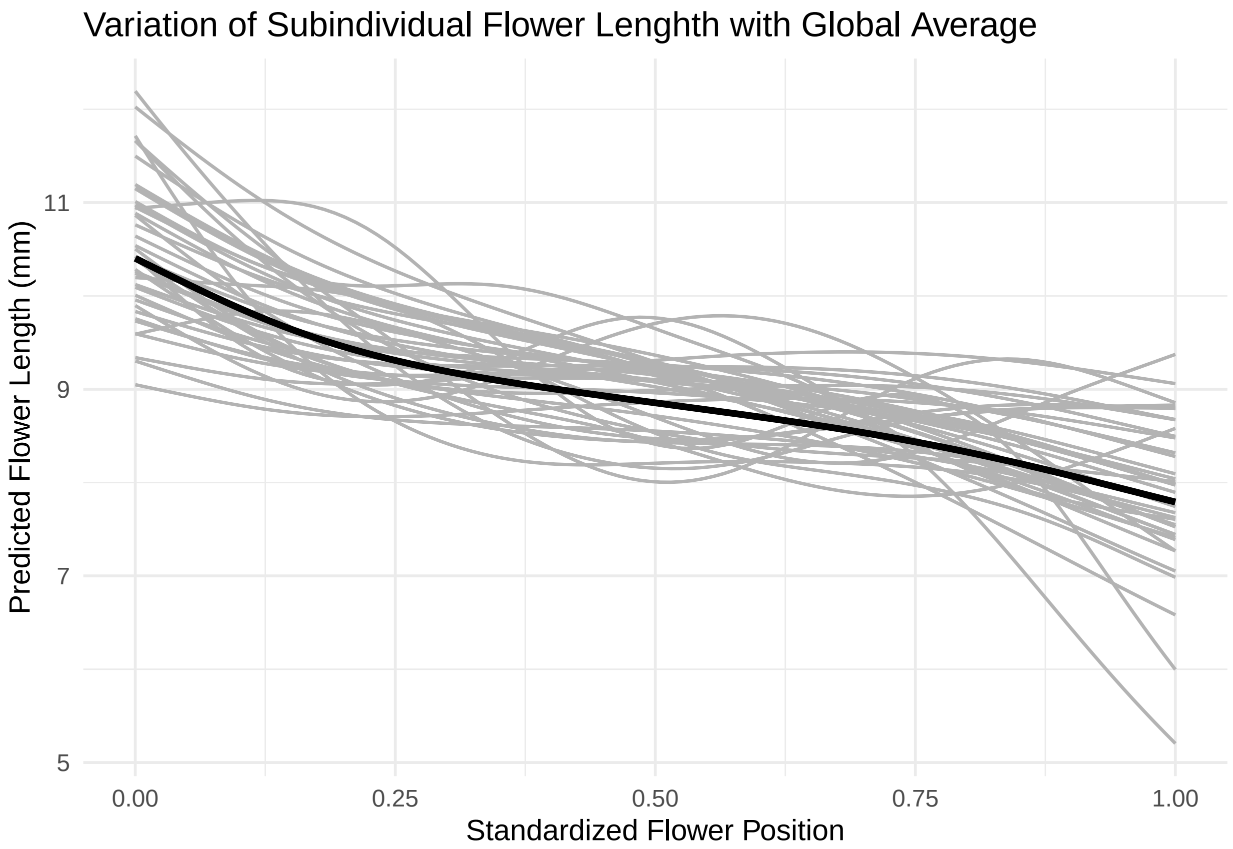
```
newdata.fl$FL_pred <- predict(mod.fl, newdata = newdata.fl, type = "response")

newdata.fd$FD_pred<- predict(mod.fd, newdata = newdata.fd, type = "response")

newdata.b$B_pred<- predict(mod.b, newdata = newdata.b, type = "response")
```

Refit global GAM models (now with total flower number as covariate), and generate predicted values (use mean total flower number during prediction) for each floral trait.

```
# 1. Global spline dataset (no PlantID)
mod_global <- gam(FL ~ s(pos_scaled) + total_flw, data = dat2,
                  family = Gamma(link = "log"))

global_data.fl <- data.frame(
  pos_scaled = pos_grid,
  total_flw = mean(dat2$total_flw)  # average value
)
global_data.fl$FL_pred <- predict(mod_global, newdata = global_data.fl, type = "response")



mod_global.fd <- gam(FD ~ s(pos_scaled) + total_flw, data = dat2,
                     family = Gamma(link = "log"))

global_data.fd <- data.frame(
  pos_scaled = pos_grid,
  total_flw = mean(dat2$total_flw)  # average value
)
global_data.fd$FD_pred <- predict(mod_global.fd, newdata = global_data.fd, type = "response")


mod_global.b <- gam(B ~ s(pos_scaled) + total_flw, data = dat2,
                    family = Gamma(link = "log"))

global_data.b <- data.frame(
  pos_scaled = pos_grid,
  total_flw = mean(dat2$total_flw)  # average value
)
global_data.b$B_pred <- predict(mod_global.b, newdata = global_data.b, type = "response")
```

Plot example of individual and global splines. Only showing plot for FL to save space. All data was written to .csv files, and plotted in SigmaPlot. Output .csv files for all three traits are found in the Results folder of the Zenodo repository.

```
ggplot() +
  geom_line(data = newdata.fl, aes(x = pos_scaled, y = FL_pred, group = PlantID),
            color = "grey70", size = 0.6) +
  geom_line(data = global_data.fl, aes(x = pos_scaled, y = FL_pred),
            color = "black", size = 1.2) +
  labs(
    x = "Standardized Flower Position",
    y = "Predicted Flower Length (mm)",
    title = "Variation of Subindividual Flower Lenghth with Global Average"
  ) +
  theme_minimal()
```



6 Coefficient of Variation

Calculate mean, variance, and CV ($\frac{\sigma}{\bar{x}}$) for all traits.

```
trait_stats <- data %>%
  group_by(PlantID) %>%
  summarise(
    FL_mean = mean(FL, na.rm = TRUE),
    FL_var  = var(FL, na.rm = TRUE),
    FL_cv   = sqrt(FL_var) / FL_mean,

    FD_mean = mean(FD, na.rm = TRUE),
    FD_var  = var(FD, na.rm = TRUE),
    FD_cv   = sqrt(FD_var) / FD_mean,
    B_mean  = mean(B, na.rm = TRUE),
    B_var   = var(B, na.rm = TRUE),
    B_cv    = sqrt(B_var) / B_mean
  )
```

In order for CV to be informative, the mean must scale with the standard deviation. Therefore, perform a simple linear regression of trait variance on trait mean. See Herrera (2009) for details.

```
# FL: variance ~ mean
lm_FL <- lm(FL_var ~ FL_mean, data = trait_stats)
summary(lm_FL)
```

```
##
## Call:
## lm(formula = FL_var ~ FL_mean, data = trait_stats)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4907 -0.4750 -0.1636  0.4975  3.3997
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.1227     1.6848  -0.666    0.509
## FL_mean       0.2790     0.1865   1.496    0.143
##
## Residual standard error: 0.8599 on 38 degrees of freedom
## Multiple R-squared:  0.05561,    Adjusted R-squared:  0.03076
## F-statistic: 2.238 on 1 and 38 DF,  p-value: 0.143
```

```
# FD: variance ~ mean
lm_FD <- lm(FD_var ~ FD_mean, data = trait_stats)
summary(lm_FD)
```

```
##
## Call:
## lm(formula = FD_var ~ FD_mean, data = trait_stats)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.53704 -0.20074 -0.08179  0.02654  2.29954
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.1929     0.5594  -2.133  0.03949 *
## FD_mean       0.5098     0.1851   2.755  0.00896 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.4662 on 38 degrees of freedom
## Multiple R-squared:  0.1665, Adjusted R-squared:  0.1446
## F-statistic:  7.59 on 1 and 38 DF,  p-value: 0.008959
```

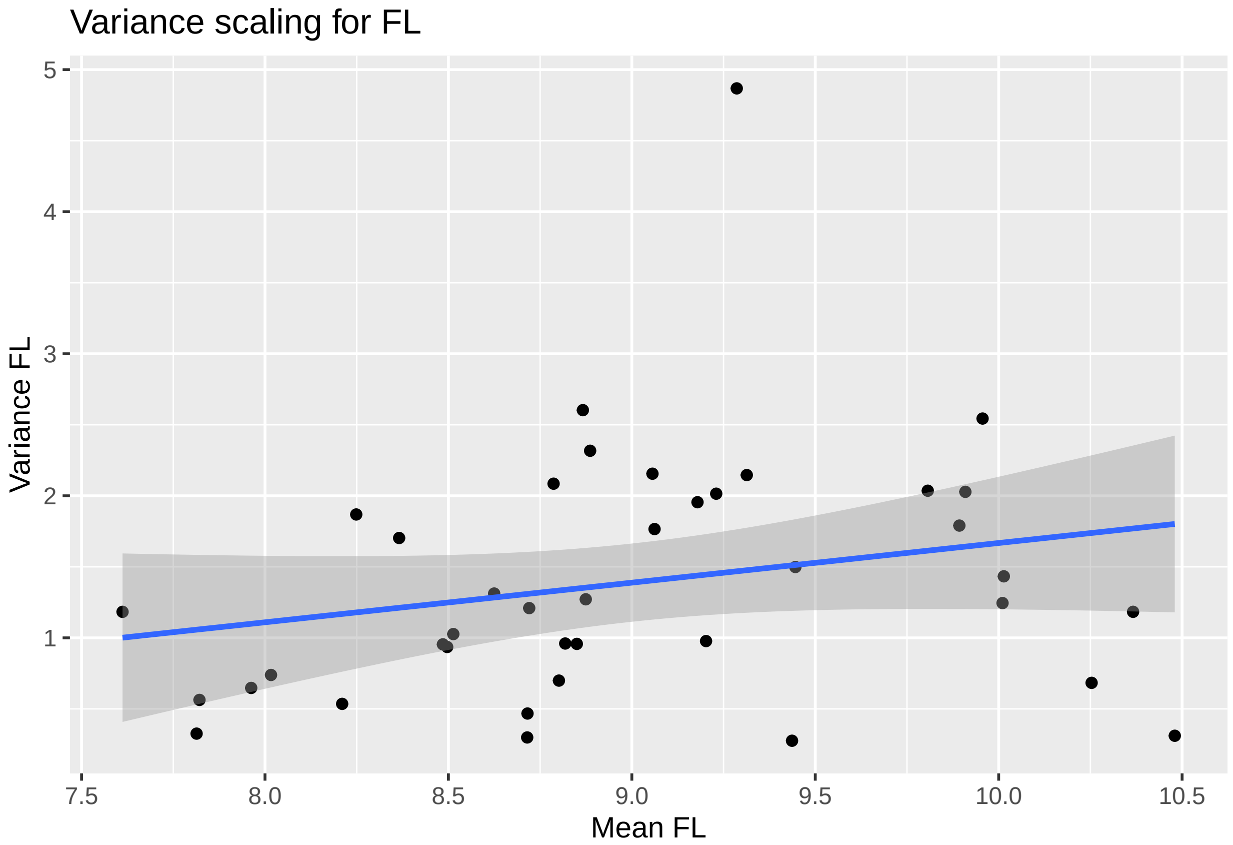
```r
# B: variance ~ mean
lm_B <- lm(B_var ~ B_mean, data = trait_stats)
summary(lm_B)
```

```
## 
## Call:
## lm(formula = B_var ~ B_mean, data = trait_stats)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -1.8378 -0.6537 -0.1917  0.6207  2.6705 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  -1.0971     0.7997  -1.372 0.178156    
## B_mean        0.5389     0.1390   3.877 0.000406 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## Residual standard error: 1.004 on 38 degrees of freedom
## Multiple R-squared:  0.2834, Adjusted R-squared:  0.2645 
## F-statistic: 15.03 on 1 and 38 DF,  p-value: 0.0004064
```

Make a quick plot to show relationship with flower length as an example.

```r
ggplot(trait_stats, aes(x = FL_mean, y = FL_var)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Variance scaling for FL", x = "Mean FL", y = "Variance FL")
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



Yes, positive relationship and therefore justification for using CV to describe variability.

References

Harder, Lawrence D., Marina M. Strelin, Ilona C. Clocher, Mason W. Kulbaba, and Marcelo A. Aizen. 2019. "The Dynamic Mosaic Phenotypes of Flowering Plants." *New Phytologist* 224 (3): 1021–34. https://doi.org/10.1111/nph.15916.

Herrera, Carlos M. 2009. *Multiplicity in Unity*. University of Chicago Press, Chicago.

Kulbaba, Mason W., Ilona C. Clocher, and Lawrence D. Harder. 2017. "Inflorescence Characteristics as Function-Valued Traits: Analysis of Heritability and Selection on Architectural Effects." *Journal of Systematics and Evolution* 55 (6): 559–65. https://doi.org/10.1111/jse.12252.

Lande, Russell, and Stevan J. Arnold. 1983. "The Measurement of Selection on Correlated Characters." *Evolution* 37 (6): 1210–26. https://doi.org/10.1111/j.1558-5646.1983.tb00236.x.