

Appendix B: Modeling Flight Response

for ‘paper_title’; Bernat, AV, Cenzer, ML

Contents

| | | |
|----------|---|-----------|
| 1 | Details of the Analyses | 3 |
| 1.1 | Description of the Data | 3 |
| 1.2 | Abbreviations Used in the Data and Code | 3 |
| 1.3 | Data Transformations | 4 |
| 2 | Winter 2020 Data Cleaning | 4 |
| 2.1 | Read Libraries | 4 |
| 2.2 | Read Source Files | 4 |
| 2.3 | Read the Data | 5 |
| 3 | Across-Trial Flight Response (T1 & T2) | 6 |
| 3.1 | Experimental Effects | 6 |
| 3.1.1 | Trial Type, Days from Start & Trial Start Time | 6 |
| 3.2 | Binomial Modeling | 7 |
| 3.2.1 | Average Days Since Start | 8 |
| 3.2.2 | Single-Variate Effects | 9 |
| 3.2.3 | Single-Variate Effects (sex, mass, & wing-to-body ratio) | 9 |
| 3.2.4 | Multi-Variate Models | 10 |
| 3.2.5 | Multi-Variate Models Split By Sex | 12 |
| 4 | Between-Trial Flight Response (T1 vs. T2) | 16 |
| 4.1 | Read Libraries | 16 |
| 4.2 | Read Source Files | 16 |
| 4.3 | Read the Data | 16 |
| 4.4 | Encodings & Signs | 16 |
| 4.5 | Multinomial Modeling | 17 |
| 4.5.1 | Baseline | 17 |
| 4.5.2 | Null Model | 17 |
| 4.5.3 | Compare Models - predictors: % mass, sex, host | 17 |
| 4.5.4 | Best Fit | 18 |
| 4.5.5 | Compare Models - predictors: % mass, sex, wing2body | 19 |
| 4.5.6 | Best Fit | 19 |
| 4.5.7 | Prediction Equations | 20 |
| 4.5.8 | Visualize Significant Multinomial Functions | 20 |
| 4.5.9 | Plot Predicted Probabilities | 22 |
| 4.6 | Multinomial Modeling (Females Only) | 23 |
| 4.6.1 | Egg Case | 23 |
| 4.6.2 | Baseline | 23 |
| 4.6.3 | Null model | 23 |
| 4.6.4 | Comparing Models - predictors: % mass, egg diff, host | 23 |
| 4.6.5 | Comparing Models - predictors: % mass, egg diff, wing2body | 24 |

| | | |
|----------|---|-----------|
| 4.6.6 | Best Fit | 25 |
| 4.6.7 | Prediction Equations | 25 |
| 4.6.8 | Visualize Significant Multinomial Functions | 25 |
| 4.6.9 | Barplot | 26 |
| 4.6.10 | Plot Predicted Probabilities | 27 |
| 5 | Fall 2019 Data Cleaning | 27 |
| 5.1 | Read Libraries | 27 |
| 5.2 | Read Source Files | 27 |
| 5.3 | Read the Data | 27 |
| 6 | Flight Response Predictions | 28 |
| 6.1 | Compute predicted probabilities | 28 |
| 6.2 | Plot predicted probabilities | 28 |
| 6.3 | Overall and Grouped Accuracies | 29 |
| 6.4 | Confusion Matrix | 29 |
| 6.5 | Females | 30 |
| 6.5.1 | Compute predicted probabilities | 30 |
| 6.5.2 | Plot predicted probabilities | 31 |
| 6.5.3 | Overall and Grouped Accuracies | 31 |
| 6.5.4 | Confusion Matrix | 31 |

1 Details of the Analyses

This document was generated by R Markdown on 2021-12-09 using R version 4.0.5 (2021-03-31). The document provides the step-by-step analytical methods used in the manuscript by Anastasia Bernat (AVB) and Meredith Censer (MLC). Multiple draft scripts were written by AVB and MLC between 2020-03-01 and 2021-07-26 until being distilled and compiled by AVB and code reviewed by MLC at the University of Chicago into this comprehensive script. All draft scripts can be viewed in the GitHub repository, SBB-dispersal (<https://github.com/mlcenser/SBB-dispersal>), within the directory `avbernat > Dispersal > Winter_2020 > stats` .

All code and output from the statistical analyses are shown. Code for data cleaning and the generation of plots is not displayed, but can be viewed in the `appendix_B-flight_summary.Rmd` file and its accompanying sourced scripts. To repeat analyses and the generation of plots, all data files and sourced scripts should follow the directory structure presented in the SBB-dispersal repository.

1.1 Description of the Data

Soapberry bugs, *Jadera haematoloma*, were flight tested in the Fall 2019 (2019-10-15 to 2019-11-08) and Winter 2020 (2020-02-17 to 2020-03-10) seasons using a flight mill machine. Soapberry bugs were flight tested twice for either set time increments or multiple hours in the flight mill and observed from 8 AM to (5-8 PM) each day. For each trial, the mass, flight response, egg-laying response, distance, duration, average speed, and max speed of each soapberry bug was recorded and then processed. All Python scripts used to process the flight records are located in the GitHub repository within the directory `avbernat > Dispersal > Winter_2020 > windaq_processing` . After trials, morphology measurements were taken for each bug. There are four morphology measurements: beak length, thorax width, wing length, and body length. The sex, wing morph (long-winged, shot-winged, or ambiguously-winged), host plant, and population of each bug were also recorded.

As a result of the experimental design, this document analyzes two main types of datasets: a full dataset and a unique dataset. A **full dataset** is a dataset where each row has a unique ID and trial type combination. A **unique dataset** is a dataset where each row has a unique ID because each trial has been grouped by ID. Examples are provided below. The advantage of generating a unique dataset is that changes between trials can be observed and analyzed.

1.2 Abbreviations Used in the Data and Code

- **SBB** - soapberry bug, *Jadera haematoloma*
- **S** - short-winged morph
- **L** - long-winged morph
- **LS** or **SL** - ambiguous wing morph
- **T1** - trial 1 of flight testing
- **T2** - trial 2 of flight testing
- **EWM** - eggs when massed, binary response (yes or no)
- **host_** - the host plant soapberry bugs were collected from, which was either *Koeleria elegans* or *Cardiospermum corindum*, occasionally called (and abbreviated) as goldenrain tree (GRT) or balloon vine (BV), respectively
- **sym_dist** - distance from the local sympatric zone, which is demarked as Homestead
- **wing2body** - a computed and unitless column that calculates the wing length divided by the body length of a soapberry bug
- **sd** - standard deviation
- **se** - standard error
- **w_** - a column name that starts with **w_** is abbreviated from “wing”. Example column: **w_morph** is “wing morph”

1.3 Data Transformations

- **_b** - a column name that ends in **_b** is a column that has been recodified into binary data (0's and 1's). Example columns: `flew_b`, `eggs_b`
- **_c** - a column name that ends in **_c** is a column that has been centered. Example columns: `sex_c`, `host_c`, `avg_days_c`
- **_s** - a column name that ends in **_s** is a column that has been standardized. Example columns: `wing2body_s`, `sym_dist_s`, `thorax_s`
- **avg_** - a column name that starts in **avg_** is a column that has been averaged across trial 1 (T1) and trial 2 (T2). Example columns: `avg_mass`, `avg_days`, `avg_time_start`, `avg_rec_dur` (exception: `average_speed`)
- **_diff** - a column name that ends in **_diff** is a column that is the difference between T1 and T2 data values. Formula: data values
- **_per** - a column name that ends in **_per** is a column that is the percent change between T1 and T2 (T2-T1) data values. Formula: $(T2-T1)/T1 * 100$.
- **_logsqrt** - a column name that ends in **_logsqrt** is a column that has been normalized using a log-square-root transformation. Formula: $\log(\sqrt{\langle data_column \rangle}) - \text{mean}(\log(\sqrt{\langle data_column \rangle}))$. Example columns: `avg_mass_logsqrt`
- **_logsqrt_i** - a column name that ends in **_logsqrt_i** is a column that has been normalized using a log-square-root transformation but its sign is the inverse of the column. Formula: $\log(\sqrt{0.85 - \text{column}}) - \text{mean}(\log(\sqrt{0.85 - \text{column}}))$. Example columns: `wing2body_logsqrt_i`

2 Winter 2020 Data Cleaning

2.1 Read Libraries

The flight response of *J. haematoloma* was analyzed using multivariate, generalized linear modeling (GLM) as implemented in the R packages `lme4` and `binom`. All plots were generated using base R and supplemented with the `popbio` package to display logistic regressions and the `rethinking` package to display 95% confidence intervals of linear regressions. Additional R packages not show below but embedded in the sourced scripts are `lubridate`, `chron`, and `dplyr`. `lubridate` and `chron` both aid in datetime manipulation while `dplyr` pipelines data grouping processes.

```
library(lme4)           # fit regressions
library(rethinking)     # Bayesian data analysis and plotting
library(popbio)         # logistic regression plotting
library(binom)          # binomial confidence intervals
```

2.2 Read Source Files

Each sourced script below aides in either data cleaning (`read_flight_data()`, `center_data()`) or multivariate GLM (`model_comparisonsAIC()`, `get_model_probs()`, `catch_warnings()`). Additionally, the function `model_comparisonsAIC()` takes in the path of a generic multi-factor script specific to the GLM family and link function needed to build the predictive models. All aforementioned, sourced scripts are located in the **Rscr** folder.

```
source_path = paste0(dir, "/Rsrc/")

script_names = c("center_flight_data.R", # 1 function: center_data()
                 "clean_flight_data.R",  # 1 function: clean_flight_data()
                 "unique_flight_data.R",  # 1 function: create_delta_data()
```

```

      "compare_models.R",      # 1 function: model_comparisonsAIC()
      "get_Akaike_weights.R",  # 1 function: get_model_probs()
      "get_warnings.R")        # 1 function: catch_warnings()

for (script in script_names) {
  path = paste0(source_path, script)
  source(path)
}

```

2.3 Read the Data

The flight performance data read directly below are only from Winter 2020 flight trials. The `read_flight_data()` function standardizes data types and names of the ID, trial type, host plant, flight response, egg-laying response, sex, population, and wing morph inputs. The date, start time, and end time of trials are also converted into datetimes. Variables of interest like wing-to-body ratio are also calculated and centered. Using the `clean_flight_data()` function, all morphology, mass, and flight performance measurements are centered and/or standardized within the `read_flight_data()` function. Then, what is returned is a full dataset (n=758) that includes all bugs collected during Winter 2020 and a subset of the full dataset (n=614) that includes only bugs tested from the Winter 2020 collection.

The `create_delta_data()` function generates the unique dataset by grouping by ID. The function also computes trial differences, percent differences, and averages for variables of interest such as mass, flight response, egg-laying response, distance, and speed. Then, the unique data variables are centered and/or standardized.

```

data_path = paste0(dir, "/Dispersal/Winter_2020/stats/data/all_flight_data-Winter2020.csv")

data = read_flight_data(data_path) # centers each subset of data
data_all = data[[1]]              # full dataset
data_tested = data[[2]]           # subset of data_all, contains only bugs flight tested

# create the unique dataset
d <- create_delta_data(data_tested, remove_bugs_tested_once = FALSE)

# keep all bugs (even bugs only tested once), then re-center
dc <- center_data(d, is_not_unique_data = FALSE)

```

Example of a **full dataset** (each row has unique ID and trial type):

```
data_tested[c(1:2, 400:401), c("ID", "trial_type")]
```

```
##      ID trial_type
## 1   114         T1
## 2   318         T1
## 400 316         T2
## 401 416         T2
```

Example of a **unique dataset** (each row has unique ID):

```
dc[c(1:2, 295:296), c("ID", "trial_type")]
```

```
## # A tibble: 4 x 2
```

```
## # Groups:   ID [4]
##   ID   trial_type
##   <fct> <list>
## 1 1     <fct [2]>
## 2 2     <fct [2]>
## 3 400    <fct [2]>
## 4 401    <fct [2]>
```

The datatype of the `trial_type` column is a list because when expanded out, it would show `list(T1, T2)`.

3 Across-Trial Flight Response (T1 & T2)

Flight response (yes flew or did not fly) was recorded and modeled. Multivariate, GLM was performed using the `glm()` function in the `lme4` package. Models were compared using Akaike Information Criterion (AIC) and model selection was determined using Akaike weights. Model fit was further evaluated between two models using the `anova()` function.

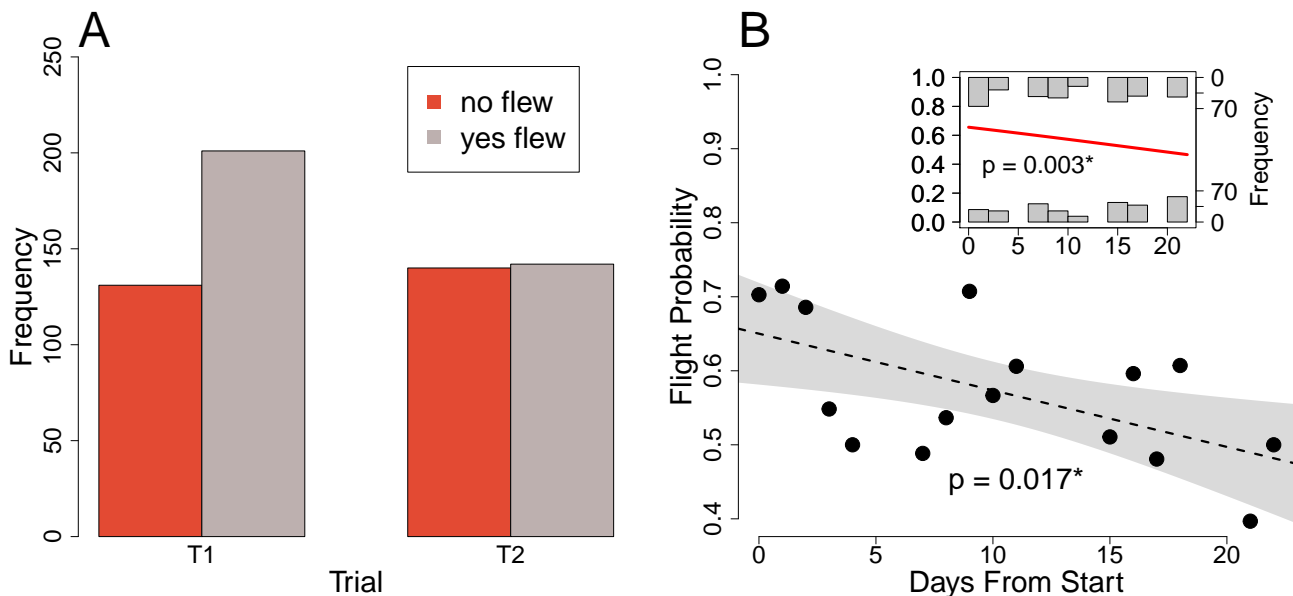
3.1 Experimental Effects

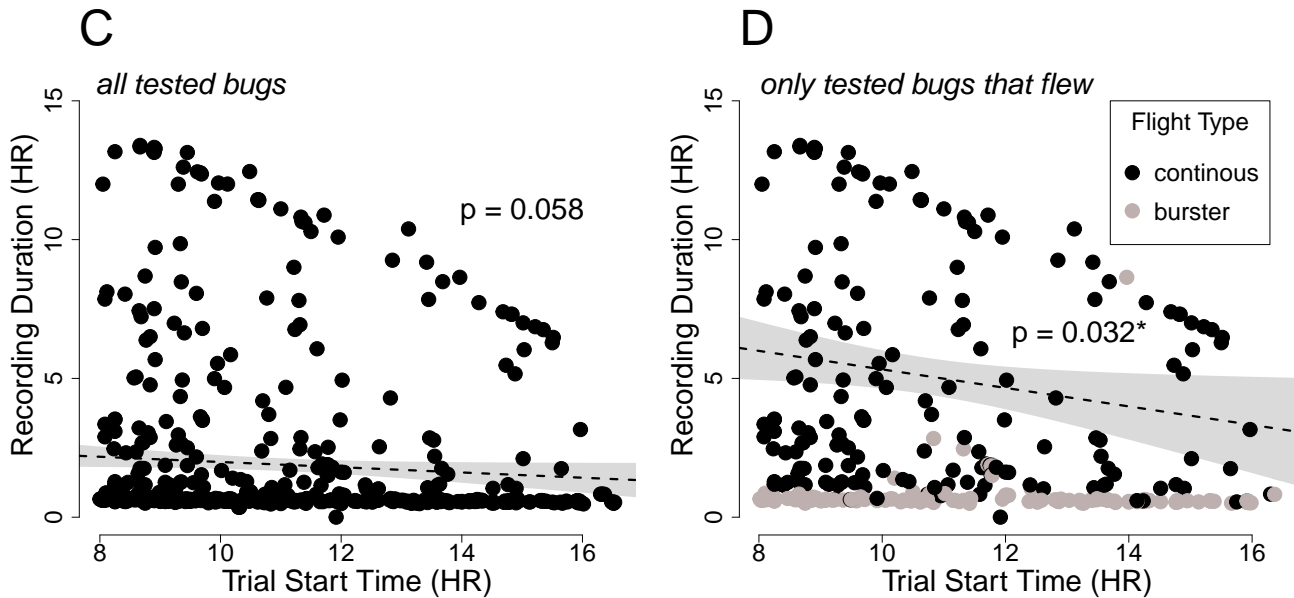
To determine how the design of the experiment affected flight response and/or performance, three design factors were modeled: trial type (T1 vs. T2), days from start, and trial start time.

3.1.1 Trial Type, Days from Start & Trial Start Time

```
# compute how many times flew yes or no per trial
binary_counts = table(data_tested$flew_b, data_tested$trial_type)[,2:3]

# aggregate by days since the trials began and flight response to determine flight prob
dd = aggregate(flew_b ~ days_from_start, data=data_tested, FUN=mean)
```





A & B. There was a negative effect of day a bug was tested (since the start of trials) on flight probability, but there was a significant effect only when the full dataset is considered. It is not significant for the unique dataset because days from start had to be averaged between trials. This is explored in the next section of the report. **C & D.** There was a negative effect of the trial start time on flight duration but only after removing bugs that did not fly ($p = 0.031$). Continuous flyers are driving this significant relationship (**D**).

3.2 Binomial Modeling

To understand SBB flight response, flight response across trials was modeled against sex, host plant, distance from the sympatric zone, wing-to-body ratio, and mass. This was done using the unique dataset.

Because the unique dataset was used, there exist multiple recorded counts of the number of times a SBB flew and did not fly between trials, T1 and T2. For that reason, we used `cbind(num_flew, num_notflew)` when modeling in order to account for all flight successes and failures for each individual.

Finally, we tested whether the data was over-dispersed, which could be resolved using a Quasibinomial:

```
# calculate the confidence interval for the mean of the data (Binomial vs. Quasibinomial)
fit <- glm(cbind(num_flew, num_notflew) ~ 1, family = binomial, data = dc)
plogis(confint(fit))
```

```
##      2.5 %      97.5 %
## 0.5191729 0.5976031
```

```
fit_q <- glm(cbind(num_flew, num_notflew) ~ 1, family = quasibinomial, data = dc)
plogis(confint(fit_q))
```

```
##      2.5 %      97.5 %
## 0.5108508 0.6056992
```

```
# estimate the dispersion parameter
summary(fit_q)$dispersion
```

```
## [1] 1.464596
```

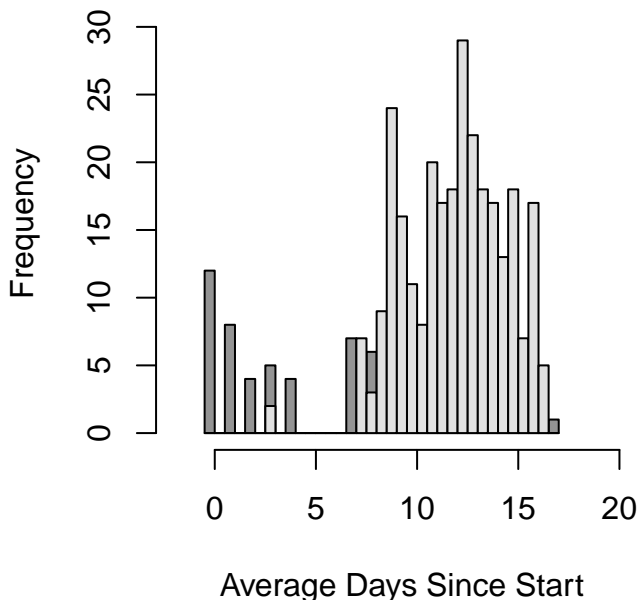
If the dispersion parameter is close to 1, the data is not over-dispersed, so there is not much of a necessity to apply a Quasibinomial model. Therefore, we selected the family as “binomial”.

3.2.1 Average Days Since Start

For the unique dataset, average days since start (log-square-root transformed) was computed in order to determine how an experimental factor affected flight response. It proved to not be significant:

```
avg_days_model<-glm(cbind(num_flew,num_notflew)~avg_days_c, data=dc, family=binomial)
summary(avg_days_model)
```

```
##
## Call:
## glm(formula = cbind(num_flew, num_notflew) ~ avg_days_c, family = binomial,
##      data = dc)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.8429  -1.7825  -0.1593   1.5162   1.5795
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.22942    0.08236   2.785  0.00535 **
## avg_days_c    0.01087    0.02354   0.462  0.64430
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 668.05  on 332  degrees of freedom
## Residual deviance: 667.84  on 331  degrees of freedom
## AIC: 759.17
##
## Number of Fisher Scoring iterations: 3
```



Average days since start accounts for bugs who died before they could be tested twice, which would most likely lead to an early average day value. The rest of the bugs that were tested twice would most likely have a later average day value. This testing regime shapes the bimodal distribution seen in the histogram. Additionally, the advantage of this computed variable is that it controls for the fact that some bugs were tested once late, and some by chance had been tested twice early. In turn, because we randomized test day, when repeated measures for each individual are combined across days, they

balance each other out. Thus, average days since start allows the multi-variate models, which control for repeated tests per ID number, to converge, and we can be confident that non-random mortality is not impacting flight response. [[AB FLAG: “can be confident...” not so founded...]]

3.2.2 Single-Variate Effects

3.2.3 Single-Variate Effects (sex, mass, & wing-to-body ratio)

We used aggregated datasets for single-variate modeling.

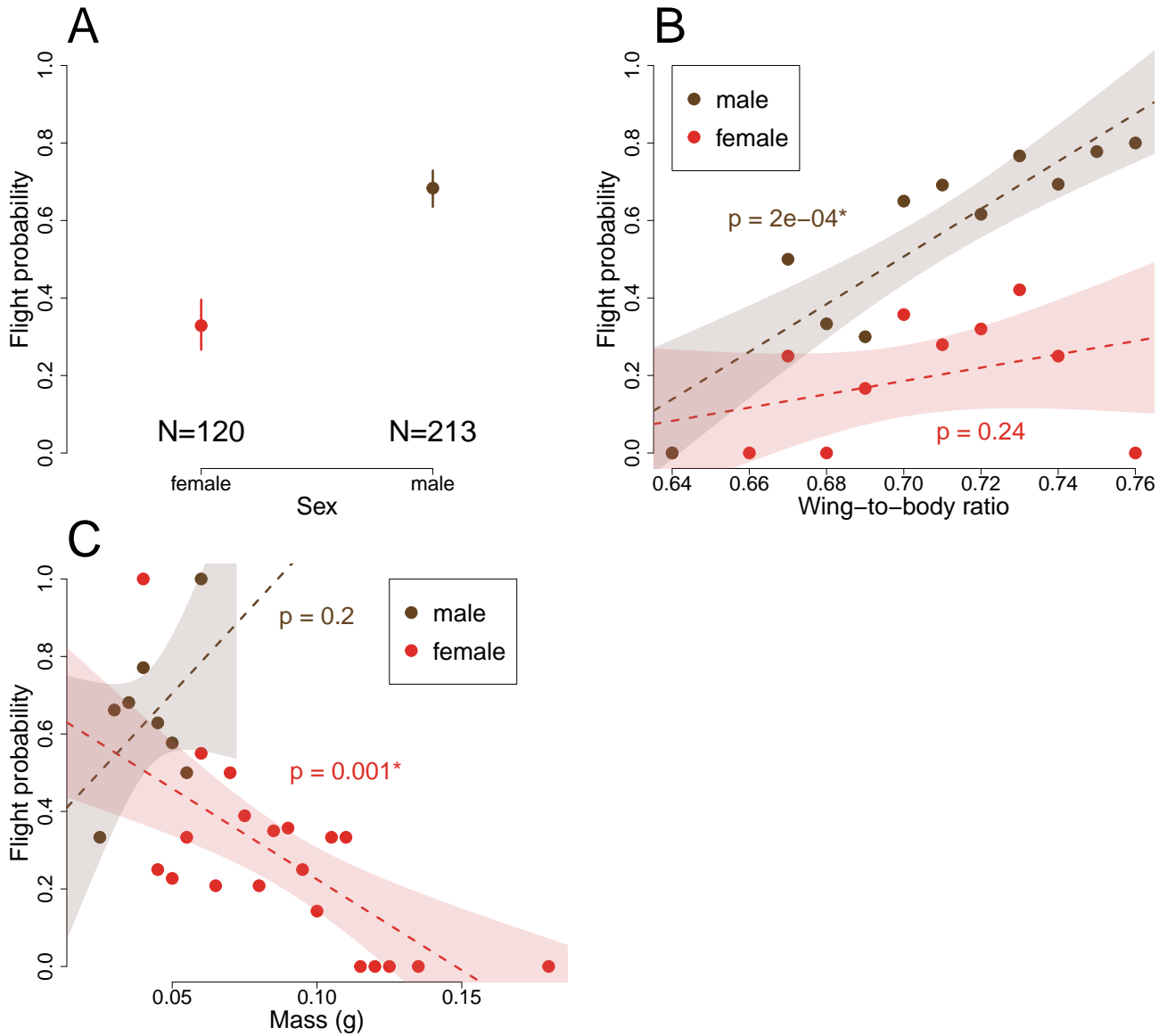
```
# tailoring variables for plotting
d$mass_block=round(d$avg_mass/0.005)*0.005      # 0.005 g blocks
d$wing2body_block=round(d$wing2body, digits=2)    # 0.01 blocks
d$days_block=round(d$avg_days, digits=0)         # integer blocks

# aggregate data for plotting
dt=aggregate(flew_prob~sex, data=d, FUN=mean)
dt$trials=c(sum(d$num_flew[d$sex=="F"]+d$num_notflew[d$sex=="F"]),
            sum(d$num_flew[d$sex=="M"]+d$num_notflew[d$sex=="M"]))

ds=aggregate(flew_prob~sex*wing2body_block, data=d, FUN=mean)
ds$n=aggregate(flew_prob~sex*wing2body_block, data=d, FUN=length)$flew_prob

dm=aggregate(flew_prob~sex*mass_block, data=d, FUN=mean)
dm$n=aggregate(flew_prob~sex*mass_block, data=d, FUN=length)$flew_prob

# calculate binomial confidence interval
dt$successes = c(sum(d$num_flew[d$sex=="F"]), sum(d$num_flew[d$sex=="M"]))
dt$CI = binom.confint(dt$successes, dt$trials, methods="exact")
```



3.2.4 Multi-Variate Models

We used the unique dataset for multi-variate modeling.

```
data<-data.frame(R1 = dc$num_flew,
                 R2 = dc$num_notflew,
                 A = dc$host_c,
                 B = dc$sex_c,
                 C = dc$sym_dist,
                 D = dc$avg_mass_logsqr,
                 E = dc$avg_days_c)
```

```
model_script = paste0(source_path,"generic models-binomial glm 2R ~ 4-FF + E.R")
errors = catch_warnings(model_comparisonsAIC(model_script))
cat("Number of models that failed to converge: ", length(errors$warnings))
```

```
##      [,1]      [,2]      [,3]
## AICs 683.3791 683.95 684.4483
## models 85      63      50
## probs 0.08873418 0.06669969 0.05198815
##
## m85 glm(formula = cbind(R1, R2) ~ A * D + B * D + C * D + E, family = binomial,
```

```
##      data = data)
## m63  glm(formula = cbind(R1, R2) ~ A * D + C * D + B + E, family = binomial,
##      data = data)
## m50  glm(formula = cbind(R1, R2) ~ A * D + B * D + E, family = binomial,
##      data = data)
## Number of models that failed to converge: 0
```

```
anova(m63, m85, test="Chisq") # Adding B*D does not improve fit
anova(m63, m36, test="Chisq") # Adding C*D does improve fit
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * D + C * D + B + E
## Model 2: cbind(R1, R2) ~ A * D + B * D + C * D + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         325       580.61
## 2         324       578.04  1    2.5709  0.1088
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * D + C * D + B + E
## Model 2: cbind(R1, R2) ~ A * D + B + C + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         325       580.61
## 2         326       585.11 -1   -4.4988  0.03392 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
M1 <- glm(cbind(num_flew, num_notflew) ~ host_c * avg_mass_logsqrt
          + sym_dist_s * avg_mass_logsqrt + sex_c + avg_days_c, data=d, family=binomial)
summary(M1)
```

3.2.4.1 Best Fit

```
##
## Call:
## glm(formula = cbind(num_flew, num_notflew) ~ host_c * avg_mass_logsqrt +
##   sym_dist_s * avg_mass_logsqrt + sex_c + avg_days_c, family = binomial,
##   data = d)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54691 -1.08562 -0.03924  1.17713  2.41023
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.03733    0.11152   0.335  0.73785
## host_c        -0.14197    0.13044  -1.088  0.27643
## avg_mass_logsqrt -1.00302    0.88134  -1.138  0.25510
## sym_dist_s     -0.03954    0.12352  -0.320  0.74890
## sex_c         -0.46077    0.16797  -2.743  0.00609 **
## avg_days_c      0.01138    0.02596   0.438  0.66111
## host_c:avg_mass_logsqrt  1.85594    0.59204   3.135  0.00172 **
```

```
## avg_mass_logsqrt:sym_dist_s -1.36386    0.66258  -2.058  0.03955 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 668.05  on 332  degrees of freedom
## Residual deviance: 580.61  on 325  degrees of freedom
## AIC: 683.95
##
## Number of Fisher Scoring iterations: 4
```

3.2.5 Multi-Variate Models Split By Sex

```
data_fem <- dc[dc$sex=="F",]
data_fem <- center_data(data_fem, is_not_unique_data = FALSE)
```

```
data<-data.frame(R1 = data_fem$num_flew,
                 R2 = data_fem$num_notflew,
                 A = data_fem$host_c,
                 B = data_fem$sym_dist,
                 C = data_fem$avg_mass_logsqrt,
                 D = data_fem$wing2body_logsqrt_i,
                 E = data_fem$avg_days_c)
```

```
model_script = paste0(source_path,"generic models-binomial glm 2R ~ 4-FF + E.R")
errors = catch_warnings(model_comparisonsAIC(model_script))
cat("Number of models that failed to converge: ", length(errors$warnings))
```

3.2.5.1 Females

```
##      [,1]      [,2]      [,3]
## AICs 238.8713 239.0635 239.8444
## models 45      25      10
## probs 0.08178418 0.07429069 0.0502761
##
## m45 glm(formula = cbind(R1, R2) ~ A * C + A * D + E, family = binomial,
##      data = data)
## m25 glm(formula = cbind(R1, R2) ~ A * C + D + E, family = binomial,
##      data = data)
## m10 glm(formula = cbind(R1, R2) ~ C + D + E, family = binomial, data = data)
## Number of models that failed to converge: 0
```

```
anova(m25, m45, test='Chisq') #adding A*D does not improve fit
anova(m25, m13, test='Chisq') #adding A*C improves fit
anova(m25, m17, test="Chisq") #adding D improves fit
anova(m25, m45, test="Chisq") #adding D improves fit
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * C + D + E
```

```
## Model 2: cbind(R1, R2) ~ A * C + A * D + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      114      202.11
## 2      113      199.92  1    2.1922   0.1387
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * C + D + E
## Model 2: cbind(R1, R2) ~ A + C + D + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      114      202.11
## 2      115      206.87 -1    -4.764   0.02906 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * C + D + E
## Model 2: cbind(R1, R2) ~ A * C + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      114      202.11
## 2      115      206.35 -1    -4.243   0.03941 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * C + D + E
## Model 2: cbind(R1, R2) ~ A * C + A * D + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      114      202.11
## 2      113      199.92  1    2.1922   0.1387
```

```
M2 <- glm(cbind(num_flew, num_notflew) ~ host_c * avg_mass_logsqrt + wing2body_logsqrt_i +
          avg_days_c, data=data_fem, family=binomial)
summary(M2)
```

3.2.5.1.1 Best Fit

```
##
## Call:
## glm(formula = cbind(num_flew, num_notflew) ~ host_c * avg_mass_logsqrt +
##   wing2body_logsqrt_i + avg_days_c, family = binomial, data = data_fem)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.1849  -1.1189  -0.7523   1.1182   2.7357
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -0.15913    0.34121  -0.466   0.6409
## host_c        -0.61037    0.33019  -1.849   0.0645 .
## avg_mass_logsqrt -2.08700    1.45468  -1.435   0.1514
## wing2body_logsqrt_i -5.37017    2.66359  -2.016   0.0438 *
## avg_days_c      0.11558    0.04757   2.430   0.0151 *
## host_c:avg_mass_logsqrt 3.02237    1.39976   2.159   0.0308 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 223.66  on 119  degrees of freedom
## Residual deviance: 202.11  on 114  degrees of freedom
## AIC: 239.06
##
## Number of Fisher Scoring iterations: 4
```

```
data_male <- dc[dc$sex=="M",]
data_male <- center_data(data_male, is_not_unique_data = FALSE)
```

3.2.5.2 Males

```
## Warning in d$latitude - sym_zone: longer object length is not a multiple of
## shorter object length
```

```
data<-data.frame(R1 = data_male$num_flew,
                 R2 = data_male$num_notflew,
                 A = data_male$host_c,
                 B = data_male$sym_dist,
                 C = data_male$avg_mass_logsqr,
                 D = data_male$wing2body_logsqr_i,
                 E = data_male$avg_days_c)
```

```
model_script = paste0(source_path,"generic models-binomial glm 2R ~ 4-FF + E.R")
errors = catch_warnings(model_comparisonsAIC(model_script))
cat("Number of models that failed to converge: ", length(errors$warnings))
```

```
##      [,1]      [,2]      [,3]
## AICs 427.3929 427.649 428.1156
## models 105      50      83
## probs 0.08393807 0.07384843 0.05848274
##
## m105      glm(formula = cbind(R1, R2) ~ A * D + B * C + B * D + C * D +
##      E, family = binomial, data = data)
## m50      glm(formula = cbind(R1, R2) ~ A * D + B * D + E, family = binomial,
##      data = data)
## m83      glm(formula = cbind(R1, R2) ~ A * D + B * C + B * D + E, family = binomial,
##      data = data)
## Number of models that failed to converge: 0
```

```
anova(m83, m105, test="Chisq") # adding C*D marginally improves fit
anova(m83, m62, test="Chisq") # adding B*C marginally improves fit
anova(m50, m62, test="Chisq") # adding C does not improve fit
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * D + B * C + B * D + E
## Model 2: cbind(R1, R2) ~ A * D + B * C + B * D + C * D + E
```

```
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         204       347.73
## 2         203       345.01  1   2.7227  0.09893 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * D + B * C + B * D + E
## Model 2: cbind(R1, R2) ~ A * D + B * D + C + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         204       347.73
## 2         205       351.01 -1  -3.2786  0.07019 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Analysis of Deviance Table
##
## Model 1: cbind(R1, R2) ~ A * D + B * D + E
## Model 2: cbind(R1, R2) ~ A * D + B * D + C + E
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         206       351.27
## 2         205       351.01  1   0.25488  0.6137
```

```
M3<-glm(cbind(num_flew, num_notflew)~host_c*wing2body_logsqrt_i + sym_dist*wing2body_logsqrt_i
+ avg_days_c, family=binomial, data=data_male)
summary(M3)
```

3.2.5.2.1 Best Fit

```
##
## Call:
## glm(formula = cbind(num_flew, num_notflew) ~ host_c * wing2body_logsqrt_i +
##   sym_dist * wing2body_logsqrt_i + avg_days_c, family = binomial,
##   data = data_male)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.6331  -0.7526   0.8309   1.1667   2.0726
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.46487    0.22478   2.068  0.0386 *
## host_c         -0.38219    0.18897  -2.023  0.0431 *
## wing2body_logsqrt_i -15.20652    5.22315  -2.911  0.0036 **
## sym_dist        0.11229    0.13852   0.811  0.4176
## avg_days_c     -0.03316    0.03421  -0.969  0.3323
## host_c:wing2body_logsqrt_i -9.46041    4.27524  -2.213  0.0269 *
## wing2body_logsqrt_i:sym_dist  6.31131    3.02643   2.085  0.0370 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 372.15  on 212  degrees of freedom
```

```
## Residual deviance: 351.27 on 206 degrees of freedom
## AIC: 427.65
##
## Number of Fisher Scoring iterations: 4
```

4 Between-Trial Flight Response (T1 vs. T2)

4.1 Read Libraries

```
library(dplyr) # data manipulation
library(zoo) # data manipulation
library(nnet) # multinomial modeling
library(kableExtra) # table formatting
library(plot.matrix) # enables matrix/heatmap plotting
```

4.2 Read Source Files

```
script_names = c("multinom_functions.R") # 4 relevant functions:
                                         # calculate_P2(), calculate_P3(),
                                         # get_significant_models(),
                                         # get_significant_modelsf(),

for (script in script_names) {
  path = paste0(source_path, script)
  source(path)
}
```

4.3 Read the Data

```
# only keep bugs tested twice
d = create_delta_data(data_tested, remove_bugs_tested_once=TRUE)
```

4.4 Encodings & Signs

We aimed to model the probability of different delta flight response cases with sex, host plant, percent changes in mass, and percent changes in egg-laying response as predictors. Since the outcomes (or response variables) were no longer binomial, we used multi-categorical logit models. Below are the categorical encodings and/or signs used. See the Appendix for additional explanations and examples of computing multi-categorical logit models.

| Delta Flight Response Key | |
|---------------------------|----------|
| Event | Encoding |
| flew in both trials | 2 |
| flew in T2 only | 1 |
| flew in neither trials | 0 |
| flew in T1 only | -1 |

| Delta Percent Mass Key (%) | |
|---------------------------------|------|
| Event | Sign |
| gained % mass from T1 to T2 | + |
| no % mass change between trails | 0 |
| lost % mass from T1 to T2 | - |

| Host Plant Key | |
|------------------------|----------|
| Host | Encoding |
| Golden Rain Tree (GRT) | 1 |
| Balloon Vine (BV) | -1 |

| Sex Key | |
|---------|----------|
| Sex | Encoding |
| Female | 1 |
| Male | -1 |

4.5 Multinomial Modeling

4.5.1 Baseline

```
# remove any missing values for flight case or mass percent change between trials
df = d[with(d, !is.na(flight_case) & !is.na(mass_per)),]

# order the dataset by ascending mass percent change values
df = df[with(df, order(mass_per)),]

# relevel the flight case factors so as to set 0 as the first level.
df$flight_case = relevel(as.factor(df$flight_case), ref = "0")
```

4.5.2 Null Model

```
null = multinom(flight_case ~ 1, data = df)
```

```
## # weights: 8 (3 variable)
## initial value 385.389832
## iter 10 value 319.269929
## final value 319.269680
## converged
```

4.5.3 Compare Models - predictors: % mass, sex, host

```
data = data.frame(R = df$flight_case,
  A = df$mass_per,
  B = df$sex_c,
  C = df$host_c)
```

```
model_script = paste0(source_path,"generic multinomial models- multinom 1RF + 3 FF.R")
model_comparisonsAIC(model_script)
```

```
##          [,1]      [,2]      [,3]      [,4]
## AICs    587.5607  591.9016  592.3168  592.4231
## models 4          7          13          12
## probs   0.7141852 0.0815063 0.06622882 0.06280119
##
## m4    multinom(formula = R ~ A + B, data = data, trace = FALSE)
## m7    multinom(formula = R ~ A + B + C, data = data, trace = FALSE)
## m13   multinom(formula = R ~ B * C + A, data = data, trace = FALSE)
## m12   multinom(formula = R ~ A * C + B, data = data, trace = FALSE)
```

```
anova(m4, m7, test="Chisq") # Adding C (host plant) does not improve fit
anova(m4, m8, test="Chisq") # Adding A*B does not improve fit
```

```
## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev   Test    Df LR stat.   Pr(Chi)
## 1      A + B      825    569.5607
## 2 A + B + C      822    567.9016 1 vs 2     3 1.659076 0.6460701
## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev   Test    Df LR stat.   Pr(Chi)
## 1 A + B      825    569.5607
## 2 A * B      822    569.4209 1 vs 2     3 0.1398496 0.9866598
```

4.5.4 Best Fit

```
M4 = multinom(flight_case ~ mass_per + sex_c, data = df)
model_table4 = calculate_P2(M4, "mass_per", "sex_c")
```

```
## # weights: 16 (9 variable)
## initial value 385.389832
## iter 10 value 286.869825
## iter 20 value 284.809036
## iter 30 value 284.797822
## final value 284.780360
## converged
##
## AIC: 587.5607
##      (Intercept) mass_per sex_c DF   SEi   SE1   SE2      zi      z1      z2
## -1      -1.015    0.043 -0.692  9 0.239 0.010 0.203  -4.248  4.390  -3.408
## 1      -6.820    -0.009 -5.626  9 0.183 0.026 0.183 -37.245 -0.348 -30.721
## 2       0.124     0.019 -0.902  9 0.167 0.008 0.159   0.742  2.334  -5.684
##      waldi wald1   wald2 Pi > |z| P1 > |z| P2 > |z|
## -1    18.049 19.272 11.617  0.000  0.000  0.001
## 1   1387.197  0.121 943.764  0.000  0.728  0.000
## 2     0.551  5.447 32.310  0.458  0.020  0.000
```

Host plant was not a significant predictor, so we tested the wing-to-body ratio as a predictor next.

4.5.5 Compare Models - predictors: % mass, sex, wing2body

```
df$wing2body_c = df$wing2body - mean(df$wing2body) # re-center the w2b predictor

data = data.frame(R = df$flight_case,
                  A = df$mass_per,
                  B = df$sex_c,
                  C = df$wing2body_c)
model_script = paste0(source_path, "generic multinomial models- multinom 1RF + 3 FF.R")
model_comparisonsAIC(model_script)

##           [,1]      [,2]      [,3]
## AICs    582.2678  585.1197  587.133
## models  7         12         13
## probs   0.6671688 0.1603139 0.05858546
##
## m7    multinom(formula = R ~ A + B + C, data = data, trace = FALSE)
## m12   multinom(formula = R ~ A * C + B, data = data, trace = FALSE)
## m13   multinom(formula = R ~ B * C + A, data = data, trace = FALSE)

anova(m7, m12, test="Chisq") # adding A*C does not improve fit
anova(m7, m13, test="Chisq") # Adding B*C does not improve fit

## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev  Test    Df LR stat.   Pr(Chi)
## 1 A + B + C      822    558.2678
## 2 A * C + B      819    555.1197 1 vs 2     3 3.148182 0.3693379
## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev  Test    Df LR stat.   Pr(Chi)
## 1 A + B + C      822    558.2678
## 2 B * C + A      819    557.1330 1 vs 2     3 1.134887 0.7686596
```

4.5.6 Best Fit

```
M5 = multinom(flight_case ~ mass_per + sex_c + wing2body_c, data = df)
model_table5 = calculate_P3(M5)

## # weights:  20 (12 variable)
## initial  value 385.389832
## iter   10 value 286.740091
## iter   20 value 280.436850
## iter   30 value 279.437125
## iter   40 value 279.174660
## iter   50 value 279.134087
## final   value 279.133921
## converged
```

```
##
## AIC: 582.2678
## (Intercept) mass % sex wing2body DF SEi SE1 SE2 SE3 zi
## -1 -0.935 0.041 -0.571 23.739 12 0.243 0.010 0.212 12.059 -3.854
## 1 -8.177 -0.005 -6.954 -6.595 12 0.187 0.025 0.187 18.786 -43.767
## 2 0.201 0.018 -0.760 28.094 12 0.172 0.008 0.166 9.718 1.173
## z1 z2 z3 waldi wald1 wald2 wald3 Pi>|z| P1>|z| P2>|z|
## -1 4.254 -2.698 1.969 14.850 18.096 7.278 3.875 0.000 0.000 0.007
## 1 -0.215 -37.102 -0.351 1915.510 0.046 1376.550 0.123 0.000 0.830 0.000
## 2 2.141 -4.590 2.891 1.375 4.585 21.071 8.357 0.241 0.032 0.000
## P3>|z|
## -1 0.049
## 1 0.726
## 2 0.004
```

4.5.7 Prediction Equations

```
get_prediction_eq = function(tb, table_rowA, table_rowB, var_lab1, var_lab2, var_lab3,
                             log_lab, title_lab) {
  I = (tb[table_rowA,1] - tb[table_rowB,1])
  M = (tb[table_rowA,2] - tb[table_rowB,2])
  S = (tb[table_rowA,3] - tb[table_rowB,3])
  W = (tb[table_rowA,4] - tb[table_rowB,4])
  EQ = paste0(log_lab, round(I, 2), " + ", round(M,2), var_lab1, " + ", round(S, 2),
              var_lab2, " + ", round(W, 2), var_lab3, title_lab)
  print(EQ)

  return(EQ)
}

EQ1 = get_prediction_eq(model_table5, 1, 2, " Mass %", " Sex", " Wing-to-Body",
                        "log(pi_-1 / pi_1) = ", " Flew in T1, not T2")
EQ2 = get_prediction_eq(model_table5, 3, 1, " Mass %", " Sex", " Wing-to-Body",
                        "log(pi_2 / pi_-1) = ", " Flew in both, not T1" )
EQ3 = get_prediction_eq(model_table5, 3, 2, " Mass %", " Sex", " Wing-to-Body",
                        "log(pi_2 / pi_1) = ", " Flew in both, not T2" )
```

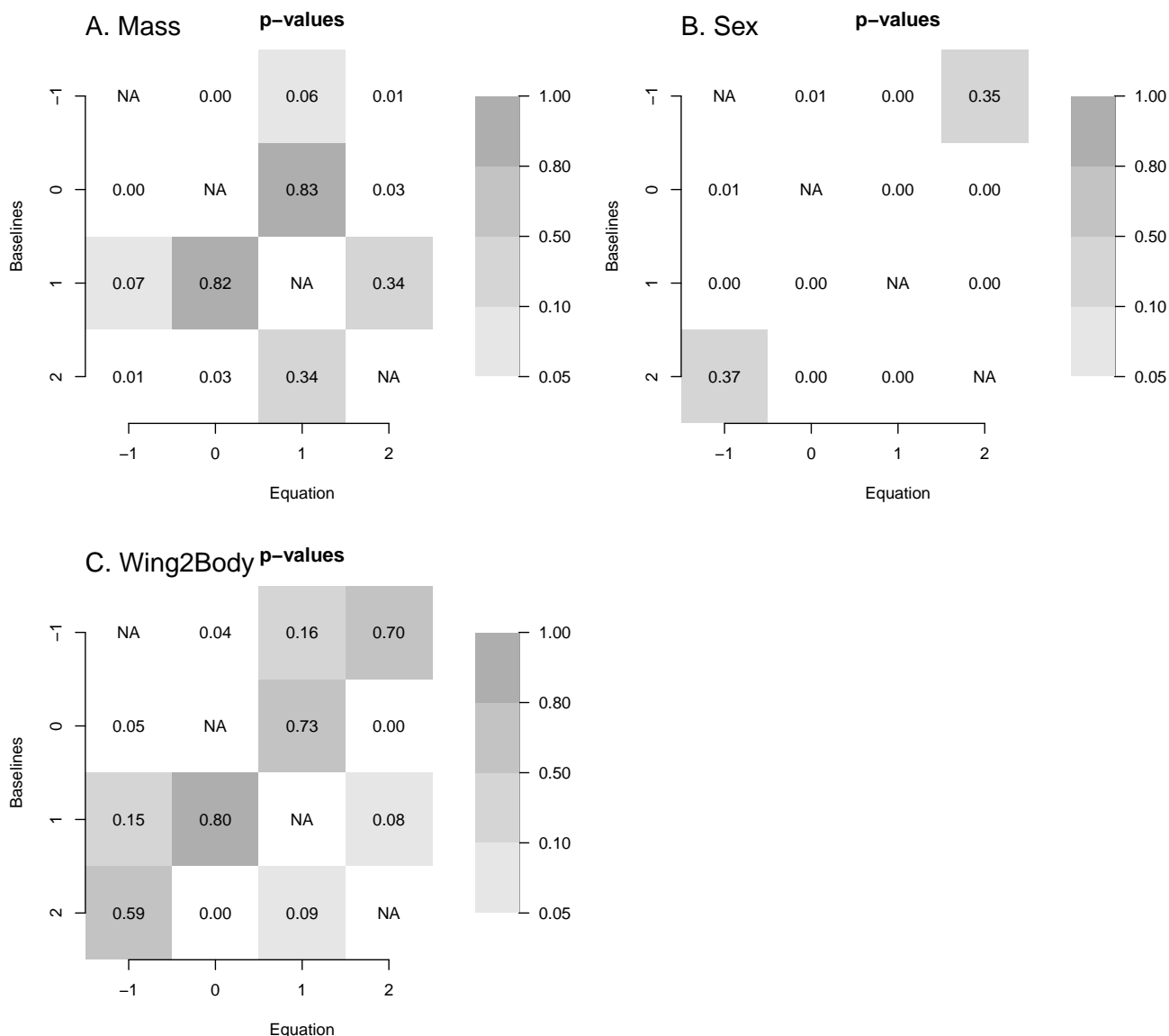
```
## [1] "log(pi_-1 / pi_1) = 7.24 + 0.05 Mass % + 6.38 Sex + 30.33 Wing-to-Body Flew in T1, not T2"
## [1] "log(pi_2 / pi_-1) = 1.14 + -0.02 Mass % + -0.19 Sex + 4.36 Wing-to-Body Flew in both, not T1"
## [1] "log(pi_2 / pi_1) = 8.38 + 0.02 Mass % + 6.19 Sex + 34.69 Wing-to-Body Flew in both, not T2"
```

4.5.8 Visualize Significant Multinomial Functions

```
# define a run_multinom_model function based on the best fit model
run_multinom_model = function(d) {
  m = multinom(flight_case ~ mass_per + sex_c + wing2body_c, trace=FALSE, data = d)
  model_table = calculate_P3(m, print_table=FALSE)
  return(model_table)
}

# determine which multinomial model equations are significant with a plot
par(mfrow=c(2,2))
mass_per_ML = get_significant_models(19) # % mass
mtext("A. Mass", side=3, adj=0, line=0.5, cex=1.3, font=1)
```

```
sex_ML = get_significant_models(20) # sex
mtext("B. Sex", side=3, adj=0, line=0.5, cex=1.3, font=1)
w2b_ML = get_significant_models(21) # wing2body
mtext("C. Wing2Body", side=3, adj=0, line=0.5, cex=1.3, font=1)
```

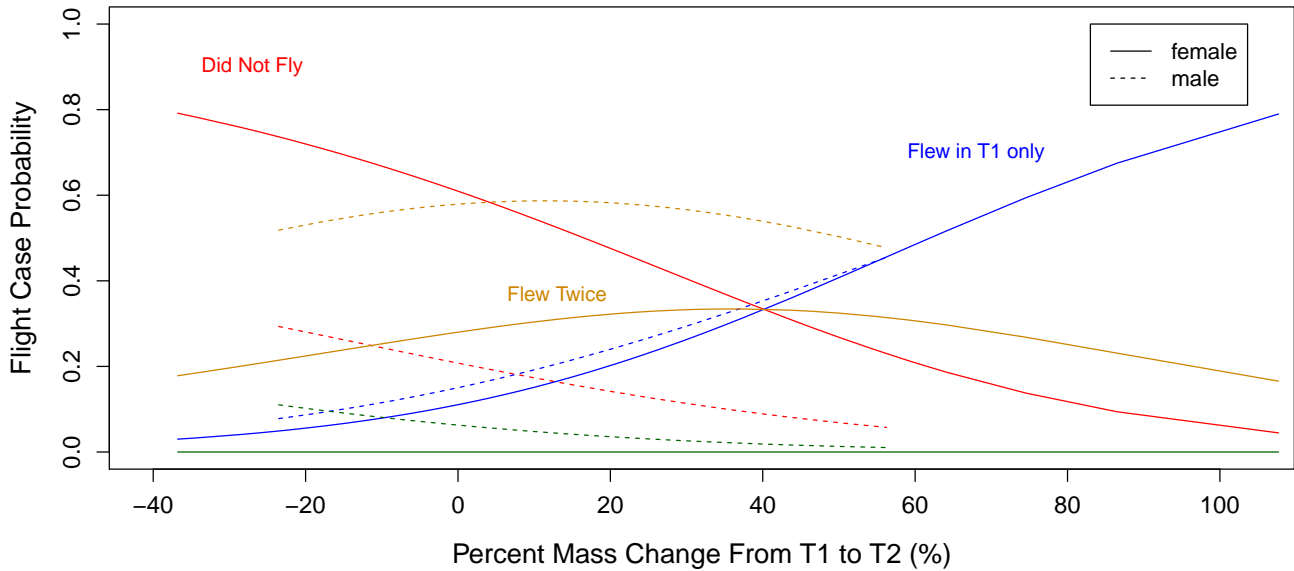


```
##           [,1]           [,2]           [,3]           [,4]
## [1,]          NA -0.041168533 -0.047132139 -0.02329390
## [2,] 0.04121460              NA -0.005450321 0.01790830
## [3,] 0.04699220 0.005882702              NA 0.02378072
## [4,] 0.02331345 -0.017835748 -0.023938109              NA
##           [,1]           [,2]           [,3]           [,4]
## [1,]          NA -0.041168533 -0.047132139 -0.02329390
## [2,] 0.04121460              NA -0.005450321 0.01790830
## [3,] 0.04699220 0.005882702              NA 0.02378072
## [4,] 0.02331345 -0.017835748 -0.023938109              NA
##           [,1]           [,2]           [,3]           [,4]
## [1,]          NA -0.041168533 -0.047132139 -0.02329390
## [2,] 0.04121460              NA -0.005450321 0.01790830
## [3,] 0.04699220 0.005882702              NA 0.02378072
## [4,] 0.02331345 -0.017835748 -0.023938109              NA
```

4.5.9 Plot Predicted Probabilities

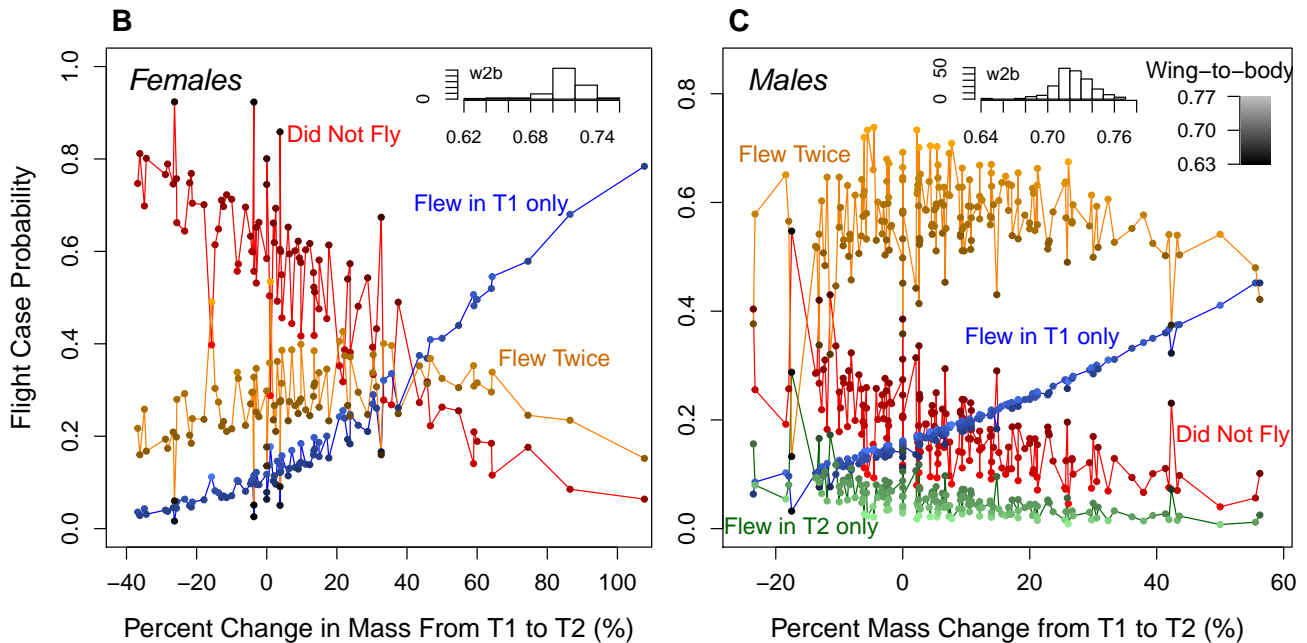
```
head(pp <- fitted(M4), 3) # compute fitted values of the best fit model without wing-to-body ratio
```

```
##           0           -1           1           2
## 1 0.7917303 0.03003973 4.362037e-06 0.1782256
## 2 0.7894639 0.03073036 4.325625e-06 0.1798015
## 3 0.7844677 0.03228066 4.247094e-06 0.1832474
```



```
head(pp <- fitted(M5), 3) # compute fitted values of the best fit model with wing-to-body ratio
```

```
##           0           -1           1           2
## 1 0.7470322 0.03581826 2.459149e-07 0.2171493
## 2 0.8116776 0.02845344 2.925412e-07 0.1598686
## 3 0.6983200 0.04316854 2.166785e-07 0.2585113
```



4.6 Multinomial Modeling (Females Only)

4.6.1 Egg Case

| Delta Egg Response Key | |
|-----------------------------|----------|
| Event | Encoding |
| laid eggs in both trials | 2 |
| laid eggs in T2 only | 1 |
| laid eggs in neither trials | 0 |
| laid eggs in T1 only | -1 |

4.6.2 Baseline

```
# filter for females &
# remove any missing values for flight case, mass percent change, and egg case between trials
df = d[with(d, !is.na(flight_case) & !is.na(mass_per) & !is.na(egg_case) & sex=="F"),]

# order the dataset by ascending mass percent change values
df = df[with(df, order(mass_per)),]

# relevel the flight case factors so as to set 0 as the first level.
df$flight_case = relevel(as.factor(df$flight_case), ref = "0")

unique(df$flight_case) # no female bug only flew in T2, so can drop factor "1"
df$flight_case = droplevels(df$flight_case)

## [1] 2 0 -1
## Levels: 0 -1 1 2
```

4.6.3 Null model

```
null <- multinom(flight_case ~ 1, data = df)

## # weights: 6 (2 variable)
## initial value 102.170943
## final value 93.055466
## converged
```

4.6.4 Comparing Models - predictors: % mass, egg diff, host

```
data <- data.frame(R = df$flight_case,
  A = df$egg_case,
  B = df$mass_per,
  C = df$host_c)
model_script = paste0(source_path, "generic multinomial models- multinom 1RF + 3 FF.R")
model_comparisonsAIC(model_script)

##      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## AICs 164.3817 165.6054 166.336 167.5638 167.9891 168.3593
## models 7      4      13      11      16      12
## probs 0.3761191 0.2039899 0.1415644 0.07661927 0.06194208 0.0514745
##
```

```
## m7 multinom(formula = R ~ A + B + C, data = data, trace = FALSE)
## m4 multinom(formula = R ~ A + B, data = data, trace = FALSE)
## m13 multinom(formula = R ~ B * C + A, data = data, trace = FALSE)
## m11 multinom(formula = R ~ A * B + C, data = data, trace = FALSE)
## m16 multinom(formula = R ~ B * C + A * B, data = data, trace = FALSE)
## m12 multinom(formula = R ~ A * C + B, data = data, trace = FALSE)
```

```
anova(m4, m7, test="Chisq") # Adding C does not improve fit
anova(m7, m13, test="Chisq") # Adding mass_per*host does not improve fit
```

```
## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev   Test      Df LR stat.    Pr(Chi)
## 1      A + B      180    153.6054
## 2 A + B + C      178    148.3817 1 vs 2      2  5.223671 0.0733997
## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev   Test      Df LR stat.    Pr(Chi)
## 1 A + B + C      178    148.3817
## 2 B * C + A      176    146.3360 1 vs 2      2  2.045698 0.3595691
```

Host plant was not a significant predictor for females as well, so we tested the wing-to-body ratio as a predictor next.

4.6.5 Comparing Models - predictors: % mass, egg diff, wing2body

```
##      [,1]      [,2]      [,3]      [,4]
## AICs 164.5293 164.9831 165.6054 167.7955
## models 7      13      4      12
## probs 0.3174096 0.2529723 0.1853291 0.06199495
##
## m7 multinom(formula = R ~ A + B + C, data = data, trace = FALSE)
## m13 multinom(formula = R ~ B * C + A, data = data, trace = FALSE)
## m4 multinom(formula = R ~ A + B, data = data, trace = FALSE)
## m12 multinom(formula = R ~ A * C + B, data = data, trace = FALSE)
```

```
anova(m4, m7, test="Chisq") # adding wing2body does not improve fit
anova(m7, m13, test="Chisq") # Adding A*C does not improve fit
anova(m7, m12, test="Chisq") # Adding B*C does not improve fit
```

```
## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev   Test      Df LR stat.    Pr(Chi)
## 1      A + B      180    153.6054
## 2 A + B + C      178    148.5293 1 vs 2      2  5.07612 0.07901956
## Likelihood ratio tests of Multinomial Models
##
```



```
## Response: R
##      Model Resid. df Resid. Dev   Test    Df LR stat.   Pr(Chi)
## 1 A + B + C      178   148.5293
## 2 B * C + A      176   144.9831 1 vs 2     2 3.546174 0.169808
## Likelihood ratio tests of Multinomial Models
##
## Response: R
##      Model Resid. df Resid. Dev   Test    Df LR stat.   Pr(Chi)
## 1 A + B + C      178   148.5293
## 2 A * C + B      176   147.7955 1 vs 2     2 0.7337197 0.6929067
```

4.6.6 Best Fit

```
M6 = multinom(flight_case ~ mass_per + egg_case, data = df) # same top model
model_table6 = calculate_P2(M6, "mass_per", "egg_case")

## # weights: 12 (6 variable)
## initial value 102.170943
## iter 10 value 76.802714
## final value 76.802689
## converged
##
## AIC: 165.6054
##      (Intercept) mass_per egg_case DF   SEi   SE1   SE2     zi    z1    z2 waldi
## -1      -0.950    0.041   -0.533  6 0.617 0.012 0.380 -1.539 3.389 -1.402 2.370
## 2         0.406    0.022   -1.098  6 0.424 0.011 0.297  0.957 2.038 -3.700 0.917
##      wald1  wald2 Pi > |z| P1 > |z| P2 > |z|
## -1 11.488  1.966   0.124   0.001   0.161
## 2   4.154 13.693   0.338   0.042   0.000
```

4.6.7 Prediction Equations

```
get_prediction_eq = function(tb, table_rowA, table_rowB, var_lab1, var_lab2,
                             log_lab, title_lab) {
  I = (tb[table_rowA,1] - tb[table_rowB,1])
  M = (tb[table_rowA,2] - tb[table_rowB,2])
  E = (tb[table_rowA,3] - tb[table_rowB,3])
  EQ = paste0(log_lab, round(I, 2), " + ", round(M,2), var_lab1, " + ", round(E, 2),
              var_lab2, title_lab)
  print(EQ)

  return(EQ)
}

EQ = get_prediction_eq(model_table6, 1, 2, " Mass %", " Egg Case",
                       "log(pi_-1 / pi_1) = ", "   Flew in T1, not T2")

## [1] "log(pi_-1 / pi_1) = -1.36 + 0.02 Mass % + 0.56 Egg Case   Flew in T1, not T2"
```

4.6.8 Visualize Significant Multinomial Functions

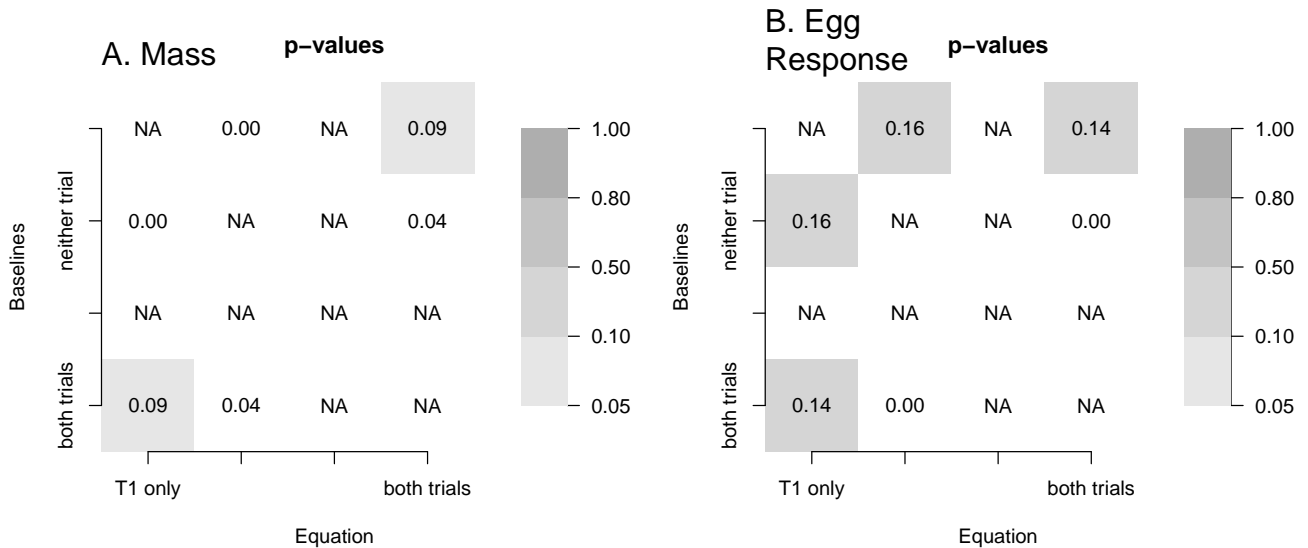
```
# define a run_multinom_model function based on the best fit model
run_multinom_model = function(d) {
  m <- multinom(flight_case ~ mass_per + egg_case, trace=FALSE, data = d)
```

```

model_table = calculate_P2(m, "mass_per", "egg_case", print_table=FALSE)
return(model_table)
}

# determine which multinomial model equations are significant with a plot
par(mfrow=c(1,2))
mass_per_ML = get_significant_models(15) # mass_per
mtext("A. Mass", side=3, adj=0, line=0.5, cex=1.6, font=1)
egg_case_ML = get_significant_models(16) # egg_case
mtext("B. Egg \nResponse", side=3, adj=0, line=0.3, cex=1.6, font=1)

```

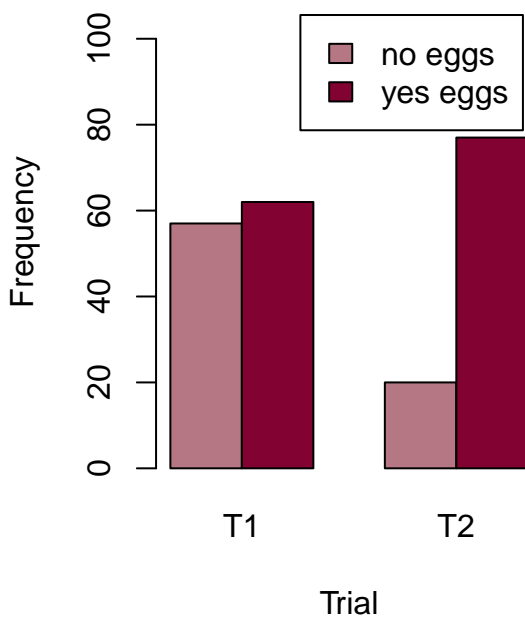


4.6.9 Barplot

```

data_fem = data_tested[data_tested$sex=="F",]
binary_counts <- table(data_fem$eggs_b, data_fem$trial_type)[,2:3]

```

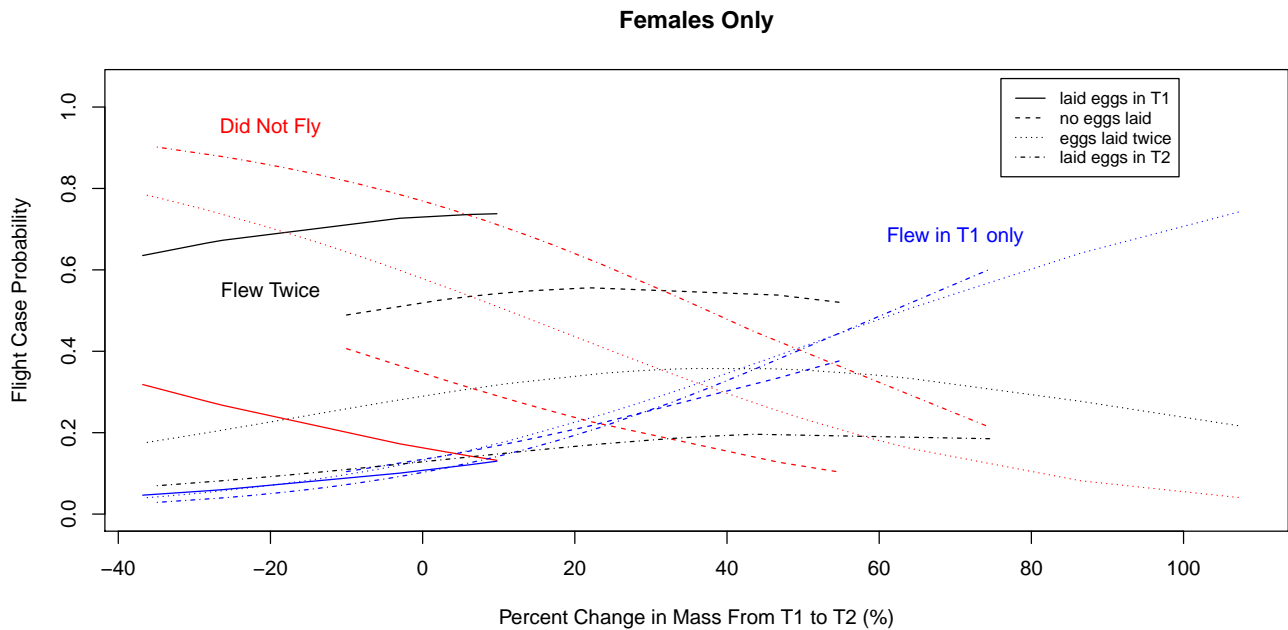


Notice that female bugs were laying more during the second trial (T2) than the first trial (T1).

4.6.10 Plot Predicted Probabilities

```
head(pp <- fitted(M6),3)
```

```
##           0           -1           2
## 1 0.3182776 0.04652361 0.63519881
## 2 0.7833277 0.04039434 0.17627792
## 3 0.9015654 0.02877502 0.06965959
```



5 Fall 2019 Data Cleaning

5.1 Read Libraries

```
library(cvms) # cross-validating regressions
```

5.2 Read Source Files

```
script_names = c("clean_flight_data-Fall.R", # 1 function: clean_flight_data.Fall()
                 "unique_flight_data-Fall.R", # 1 function: create_delta_data.Fall()
                 "prediction_accuracy.R",      # 1 function: calculate_accuracy()
                 "confusion_matrix.R")        # 1 function: get_confusion_matrix()

for (script in script_names) {
  path = paste0(source_path, script)
  source(path)
}
```

5.3 Read the Data

```
data_path = paste0(dir,"/Dispersal/Winter_2020/stats/data/full_data-Fall2019.csv")
dataFall = clean_flight_data.Fall(data_path)
```

```
# extract sets with an experimental design similar to the Winter tests
```

```
ongoing_data = dataFall[with(dataFall,!is.na(mass) & set_number > 71),]

# create delta data
d = create_delta_data.Fall(ongoing_data)
```

6 Flight Response Predictions

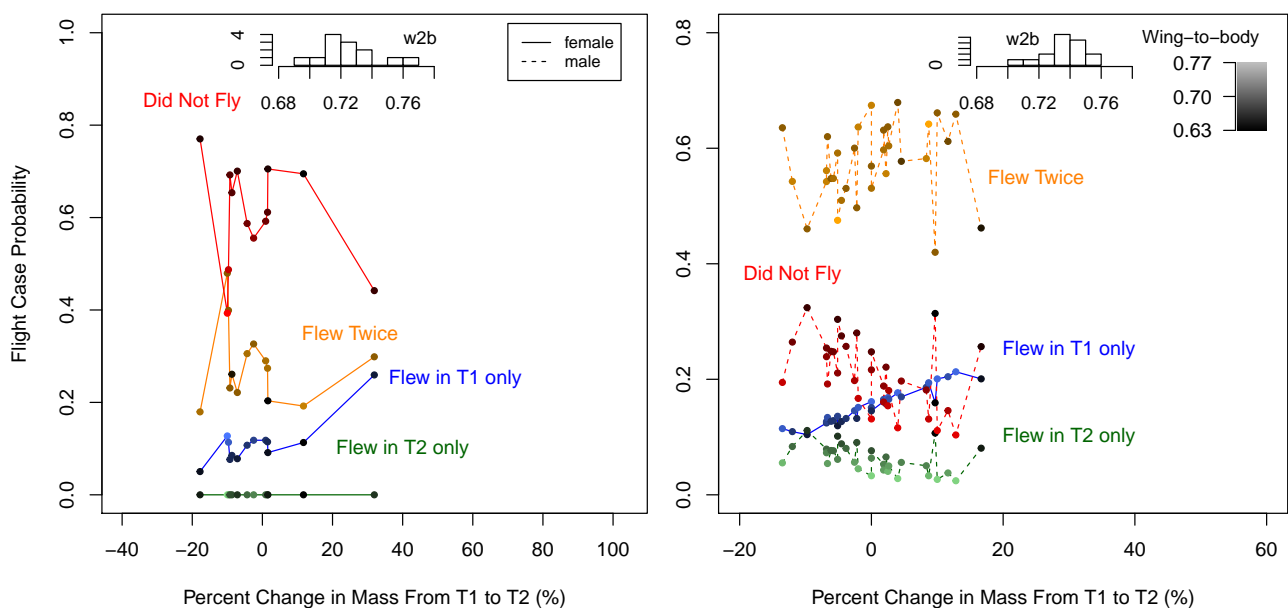
6.1 Compute predicted probabilities

```
d <- d[with(d, order(mass_per)),]

neither = c()
T1_rather_than_none = c()
T2_rather_than_none = c()
both_rather_than_none = c()

for (i in 1:nrow(d)) {
  m = d$mass_per[[i]]
  s = d$sex_c[[i]]
  w = d$wing2body_c[i]
  # extract effects from the best fit model
  top0 = exp(0) # equals 1
  top1 = exp(model_table5[1,1] + model_table5[1,2]*m + model_table5[1,3]*s + model_table5[1,4]
  top2 = exp(model_table5[2,1] + model_table5[2,2]*m + model_table5[2,3]*s + model_table5[2,4]
  top3 = exp(model_table5[3,1] + model_table5[3,2]*m + model_table5[3,3]*s + model_table5[3,4]
  bottom = top0 + top1 + top2 + top3
  # calculate predicted probabilities
  neither = c(neither, top0/bottom)
  T1_rather_than_none = c(T1_rather_than_none, top1/bottom)
  T2_rather_than_none = c(T2_rather_than_none, top2/bottom)
  both_rather_than_none = c(both_rather_than_none, top3/bottom)
}
```

6.2 Plot predicted probabilities



6.3 Overall and Grouped Accuracies

```
probs = round(cbind(neither, T1_rather_than_none, T2_rather_than_none, both_rather_than_none),
summary_probs = cbind(as.character(d$flight_case), as.character(d$sex), probs)
colnames(summary_probs) = c("event", "sex", "none", "T1", "T2", "both")
dataframe = as.data.frame(summary_probs)
nrow(dataframe)

## [1] 45

# overall
acc = calculate_accuracy(dataframe,3,6)
paste("Overall prediction accuracy, ", round(acc,2))

# by sex
femdata = dataframe[dataframe$sex=="F",]
maledata = dataframe[dataframe$sex=="M",]

accF = calculate_accuracy(femdata,3,6)
paste("Female prediction accuracy, ", round(accF,2))
accM = calculate_accuracy(maledata,3,6)
paste("Male prediction accuracy, ", round(accM,2))

## [1] "Overall prediction accuracy, 0.6"
## [1] "Female prediction accuracy, 0.38"
## [1] "Male prediction accuracy, 0.69"
```

6.4 Confusion Matrix

```
acc_table = get_confusion_matrix(dataframe,3,6)
acc_table[,1:5]

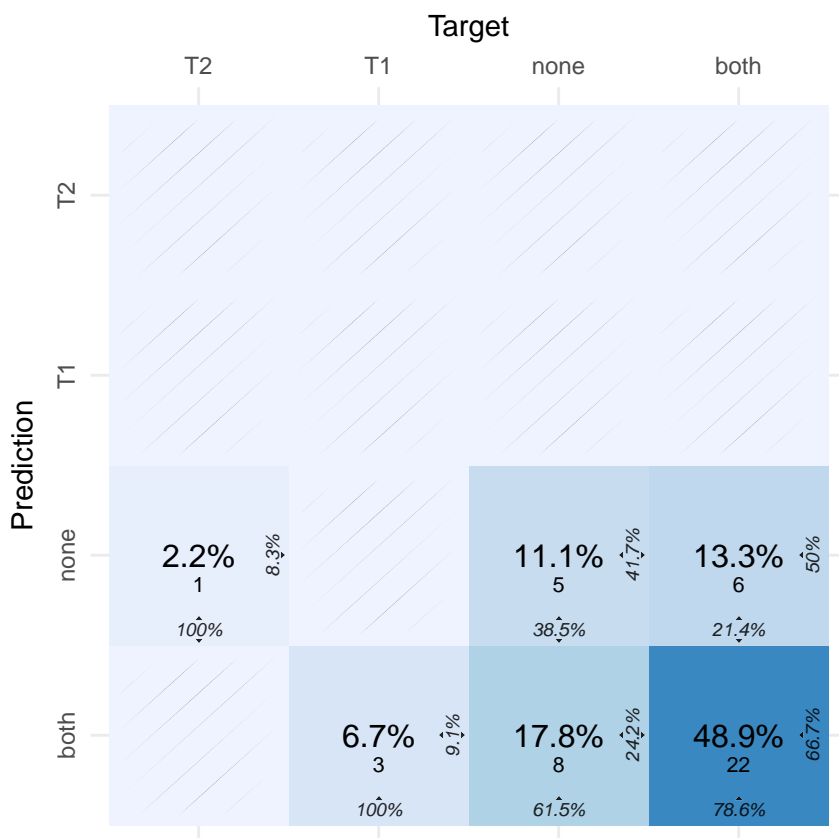
## # A tibble: 1 x 5
##   `Overall Accuracy` `Balanced Accuracy`    F1 Sensitivity Specificity
##   <dbl>             <dbl> <dbl>         <dbl>         <dbl>
## 1         0.6         0.538   NaN         0.293         0.784

confusion_matrix <- acc_table$'Confusion Matrix'[[1]]
confusion_matrix

## # A tibble: 16 x 3
##   Prediction Target    N
##   <chr>         <chr> <int>
## 1 both         both    22
## 2 none         both     6
## 3 T1           both     0
## 4 T2           both     0
## 5 both         none     8
## 6 none         none     5
## 7 T1           none     0
## 8 T2           none     0
## 9 both         T1       3
## 10 none        T1       0
## 11 T1          T1       0
## 12 T2          T1       0
## 13 both        T2       0
```

```
## 14 none      T2      1
## 15 T1        T2      0
## 16 T2        T2      0
```

```
plot_confusion_matrix(confusion_matrix, add_sums=FALSE)
```



6.5 Females

```
dfem = d[d$sex=="F",]
```

```
dfem <- dfem[with(dfem, order(mass_per)),]
```

```
neither = c()
T1_rather_than_none = c()
both_rather_than_none = c()
for (i in 1:nrow(dfem)) {
  M = dfem$mass_per[[i]]
  EC = dfem$egg_diff[[i]]
  top0 = exp(0) # equals 1
  top1 = exp(model_table6[1,1] + model_table6[1,2]*M + model_table6[1,3]*EC)
  top2 = exp(model_table6[2,1] + model_table6[2,2]*M + model_table6[2,3]*EC)
  bottom = top0 + top1 + top2
  neither = c(neither, top0/bottom)
  T1_rather_than_none = c(T1_rather_than_none, top1/bottom)
  both_rather_than_none = c(both_rather_than_none, top2/bottom)
}
```

6.5.1 Compute predicted probabilities

```

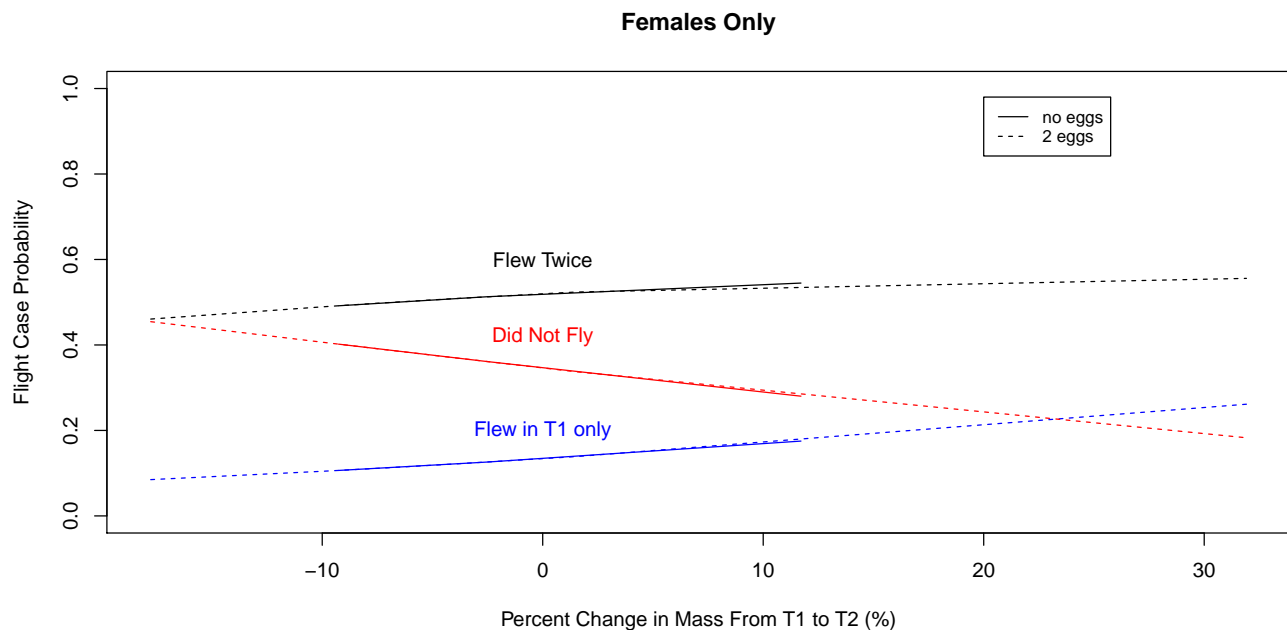
probs = round(cbind(neither, T1_rather_than_none, both_rather_than_none),2)
summary_probs = cbind(as.character(dfem$flight_case), as.character(dfem$egg_diff), probs)
colnames(summary_probs) = c("event", "egg_diff", "none", "T1", "both")

egg2 = c(1,2,3,5,6,7,9,10,11,13)
noegg = c(4,8,12)

dataframe = as.data.frame(summary_probs)
dataframe$egg_cat = c(2,2,2,0,2,2,2,0,2,2,2,0,2)

```

6.5.2 Plot predicted probabilities



6.5.3 Overall and Grouped Accuracies

```

accF_eggs = calculate_accuracy(dataframe,3,5)
paste("Female prediction accuracy for mass diff and egg model, ", round(accF_eggs,2))

## [1] "Female prediction accuracy for mass diff and egg model, 0.46"

```

6.5.4 Confusion Matrix

```

acc_table = get_confusion_matrix(dataframe,3,5)
acc_table[,1:5]

## # A tibble: 1 x 5
##   `Overall Accuracy` `Balanced Accuracy` F1 Sensitivity Specificity
##   <dbl>             <dbl> <dbl>      <dbl>      <dbl>
## 1      0.462         0.5    NaN      0.333      0.667

confusion_matrix <- acc_table$'Confusion Matrix'[[1]]
confusion_matrix

## # A tibble: 9 x 3
##   Prediction Target    N
##   <chr>          <chr> <int>
## 1 both          both     6

```

```
## 2 none      both      0
## 3 T2        both      0
## 4 both      none      6
## 5 none      none      0
## 6 T2        none      0
## 7 both      T2        1
## 8 none      T2        0
## 9 T2        T2        0
```

```
plot_confusion_matrix(confusion_matrix, add_sums=FALSE)
```

