Statistical analysis for: 'Personality and the collective: Bold homing pigeons occupy higher leadership ranks in flocks'

Takao Sasaki[1], Richard P. Mann[2], Katherine N. Warren[1], Tristian Herbert[1], Tara Wilson[3] & Dora Biro[1]

1. Department of Zoology, University of Oxford, South Parks Road, OX1 3PS Oxford, UK
2. Department of Statistics, School of Mathematics, University of Leeds, LS2 9JT Leeds, UK
3. University College London, Gower Street, London, WC1E 6BT, UK

Repeatability Analysis

First we establish whether personality (i.e. boldness) is a repeated and consistent across trials. We will do this both for boldness defined in a binary sense, using a binomial test, and for boldness rank via a Spearman rank correlation test. We do not consider the raw response times as many trials were stopped at 600s with no response

```
###Set a threshold response time (in seconds) to be considered 'bold' in a binary sense

boldness_threshold =309.5 #median is 309.5
```

Free flight data

```
#Load results for free flight data
personality_results = read.xlsx("./finalData/TakaDataMaster.xls", sheetName="solo")
```

Process data:

```
#Create binary personality variables for each of the 3 tests,
#and also rank by mean time to respond. Remove any birds without all 3 trials present

#Rescale weight and age to zero mean, unit sd
personality_results$test1_bin = (personality_results$test1 < boldness_threshold)
personality_results$test2_bin = (personality_results$test2 < boldness_threshold)
personality_results$test3_bin = (personality_results$test3 < boldness_threshold)
personality_results$meantime = (personality_results$test1 + personality_results$test2
                                + personality_results$test3)/3
personality_results$meantime_rank =
  as.ordered(rank(personality_results$meantime, ties.method = 'average'))
personality_results = personality_results[complete.cases(personality_results),]
personality_results[, c('Weight_scaled', 'Age_scaled')] =
 scale(personality_results[, c('Weight', 'Age')])
```

Binary

First do binomial tests for repeatability from trial 1 to 2, 2 to 3 and 1 to 3

```
#Count repeaters for free flight data
N_free = nrow(personality_results)
num_repeat12_free = sum(personality_results$test1_bin == personality_results$test2_bin)
num_repeat23_free = sum(personality_results$test2_bin == personality_results$test3_bin)
num_repeat123_free = sum((personality_results$test1_bin == personality_results$test2_bin) &
                         (personality_results$test2_bin == personality_results$test3_bin))

#Binomial tests
p_12_free = 1-pbinom(num_repeat12_free-1, size=N_free, prob=0.5)
p_23_free = 1-pbinom(num_repeat23_free-1, size=N_free, prob=0.5)
p_123_free = 1-pbinom(num_repeat123_free-1, size=N_free, prob=0.25)
```

Test results:

```
## [1] "Num. repeaters 1-2: 42 of 50"

## [1] "P = 5.81777902297631e-07"

## [1] "Num. repeaters 2-3: 42 of 50"

## [1] "P = 5.81777902297631e-07"

## [1] "Num. repeaters 1-2-3: 37 of 50"

## [1] "P = 5.02931030155196e-13"
```

Ordinal

Calculate Spearman rank correlations between ranks on subsequent trials.

```
corr_12_free = cor.test(personality_results$test1,
                        personality_results$test2, method = 'spearman')
corr_23_free = cor.test(personality_results$test2,
                        personality_results$test3, method='spearman')
```

Test results:

```
## [1] "Spearman rho 1-2: 0.720805962264438"

## [1] "P =3.57147832310267e-09"

## [1] "Spearman rho 2-3: 0.800259185530169"

## [1] "P = 3.09256476152417e-12"
```

Navigational data (familiar and unfamiliar)

Load data for personality trials

```
#Load results for personality data
personality_results_navigational = read.xlsx("./finalData/personality_test_2016_R.xlsx",
                                        sheetName="personality")
```

Process data:

```
#Create binary personality variables for each of the 2 tests,
#and also rank by mean time to respond.

#Scale weigth and age to 0 mean,sd1
personality_results_navigational$test1_bin =
  (personality_results_navigational$test1 < boldness_threshold)
personality_results_navigational$test2_bin =
  (personality_results_navigational$test2 < boldness_threshold)
personality_results_navigational$meantime =
  (personality_results_navigational$test1 + personality_results_navigational$test2)/2
personality_results_navigational$meantime_rank =
  as.ordered(rank(personality_results_navigational$meantime, ties.method = 'average'))

personality_results_navigational[, c('Weight_scaled', 'Age_scaled')] =
  scale(personality_results_navigational[, c('Weight', 'Age')])
```

Binary

Do binomial tests for repeatability from trial 1 to 2

```
#Count repeaters
N_nav = nrow(personality_results_navigational)
num_repeat12_nav =
  sum(personality_results_navigational$test1_bin == personality_results_navigational$test2_bin)

#Binomial tests
p_12_nav = 1-pbinom(num_repeat12_nav-1, size=N_nav, prob=0.5)
```

Test results:

```
## [1] "Num. repeaters 1-2: 34 of 45"

## [1] "P = 0.000412041179799871"
```

Ordinal

Calculate Spearman rank correlations between ranks on subsequent trials.

```
corr_12_nav = cor.test(personality_results_navigational$test1,
                       personality_results_navigational$test2, method = 'spearman')
```

Test results:

```
## [1] "Spearman rho 1-2: 0.519782324614081"

## [1] "P =0.000252919308828249"
```

Effect of age and weight on personality

We use a logistic regression to assess whether age and weigth affect boldness in a binary sense, and we use an ordinal regression (cumulative link model) to test whether age and weight affect boldness ranking

Free flight data

Binary

Logistic regression is a generalised linear model with binomial family. Dependent variable is proprtion of trial responses which were 'bold'

```
#Proportion of bold responses
personality_results$rho = (personality_results$test1_bin+personality_results$test2_bin
                           +personality_results$test3_bin)/3
#Weight matrix of number of trials
W = matrix(3, nrow=nrow(personality_results), ncol=1)
glm_model_free = glm(rho ~ Weight_scaled + Age_scaled,
                     data = personality_results, family=binomial, weights = W)
```

Model results:

```
##                     Estimate Std. Error       z value  Pr(>|z|)
## (Intercept)   -2.440871e-05  0.1648762 -0.0001480426 0.9998819
## Weight_scaled  6.317121e-03  0.1735056  0.0364087369 0.9709564
## Age_scaled     2.836586e-01  0.1749873  1.6210242738 0.1050124
```

Ordinal

Ordinal regression is a cumulative link model (GLM for ranked data). Dependent variable is the rank ordering of boldness

```
clm_personality_free = clm(meantime_rank ~ Age_scaled + Weight_scaled, data = personality_results)
```

Model results. There is no intercept in a ranking model, and rank thresholds are excluded for brevity

```
##                   Estimate Std. Error    z value  Pr(>|z|)
## Weight_scaled  0.09842715  0.2583512  0.3809819 0.7032167
## Age_scaled    -0.40705276  0.2834908 -1.4358587 0.1510426
```

Summary for free flight data

No effect of age or weight on personality either in binary or ordinal analyses

Navigational data

Binary

```
#Proportion of bold responses
personality_results_navigational$rho =(personality_results_navigational$test1_bin
                                  +personality_results_navigational$test2_bin)/2

#Weight matrix of number of trials (2)
W_nav =matrix(2, nrow=nrow(personality_results_navigational), ncol=1)
#Logistic regression
glm_model_nav = glm(rho ~ Weight_scaled + Age_scaled,
                    data = personality_results_navigational, family=binomial, weights = W_nav)
```

Model results:

```
##                 Estimate Std. Error    z value    Pr(>|z|)
## (Intercept)   0.01195355  0.2224632 0.05373271 0.957148124
## Weight_scaled 0.70113425  0.2606419 2.69002902 0.007144581
## Age_scaled    0.17130365  0.2241033 0.76439596 0.444631317
```

Ordinal

Same as above but for navigational trials personality data

```
clm_personality_nav =
  clm(meantime_rank ~ Age_scaled + Weight_scaled, data = personality_results_navigational)
```

Model results. There is no intercept in a ranking model, and rank thresholds are excluded for brevity

```
##                  Estimate Std. Error    z value    Pr(>|z|)
## Weight_scaled -0.87525675  0.2939045 -2.9780315 0.002901062
## Age_scaled    -0.05855039  0.2684901 -0.2180728 0.827372413
```

Summary for navigational data

In both binary and ordinal analyses there is a significant effect of weight: heavier birds are bolder. No effect of age in either analysis

Combined data

Merge the two data sets and define a new set of rankings by mean response time

```
#Merge
all_personality_data = rbind.fill(personality_results, personality_results_navigational)
all_personality_data[, c('Weight_scaled', 'Age_scaled')] =
 scale(all_personality_data[, c('Weight', 'Age')])
W_both = rbind(W, W_nav)
#Rank boldness
all_personality_data$meantime_rank =
  as.ordered(rank(all_personality_data$meantime, ties.method = 'average'))
```

Binary

Logistic regression on combined data

```
glm_model_all =
  glm(rho ~ Weight_scaled + Age_scaled, data = all_personality_data, family=binomial, weights = W_both)
```

Model results:

```
##                  Estimate Std. Error   z value   Pr(>|z|)
## (Intercept)   0.03289237  0.1321778 0.2488495 0.80347723
## Weight_scaled 0.17797345  0.1377426 1.2920724 0.19633207
## Age_scaled    0.23078055  0.1378963 1.6735805 0.09421307
```

Ordinal

Ordinal regression on combined data

```
clm_personality_all = clm(meantime_rank ~ Age_scaled + Weight_scaled, data = all_personality_data)
```

Model results. There is no intercept in a ranking model, and rank thresholds are excluded for brevity

```
##                  Estimate Std. Error    z value  Pr(>|z|)
## Weight_scaled -0.2875306  0.1923230 -1.4950399 0.1349040
## Age_scaled    -0.1544286  0.1846705 -0.8362383 0.4030209
```

Summary for combined data

No significant effect of age or weight on personality either in binary or ordinal analyses, though both are not far from significant in the binary analysis

Summary for effect of age and weight on personality

No consistent effect of either age or weight. In the navigational data weight has a significant effect, but not overall considering both sets of data. This is true for both binary and ordinal boldness

Effect of age, weight, boldness and release number on flight characteristics in solo training (CH site)

We test for effects on flight speed, efficiency and fidelity.

Speed

Load data and make aggregate object of flights

```
speed_results = read.xlsx("./finalData/solo_data2 fixed.xlsx", sheetName="speed")
agg_data1 = data.frame(Speed = matrix(NA, nrow=10000, ncol=1),
                       ID = matrix(NA, nrow=10000, ncol=1),
                       Bold = matrix(NA, nrow=10000, ncol=1),
                       Release = matrix(NA, nrow=10000, ncol=1),
                       meantime_rank = matrix(NA, nrow=1000, ncol=1))


count = 0
speed_results$ID = as.character(speed_results$ID)
for (j in 1:13){
  for (k in 1:23){
    count = count +1
    agg_data1$Speed[count] = speed_results[k, 1+j]
    agg_data1$ID[count] = speed_results$ID[k]
    agg_data1$Bold[count] = speed_results$Bold[k]
    agg_data1$Weight[count] = speed_results$Weight[k]
    agg_data1$Age[count]= speed_results$Age[k]
    agg_data1$Release[count] = j
    agg_data1$meantime_rank[count] =
      personality_results_navigational$meantime_rank[which(personality_results_navigational$ID
                                                           == agg_data1$ID[count])]


  }
}
agg_data1 = agg_data1[1:count,]
agg_data1$ID = as.factor(agg_data1$ID)

ur = unique(agg_data1$meantime_rank)
rur = rank(ur)
for (i in 1:length(agg_data1$meantime_rank)){
  agg_data1$meantime_rank[i] = rur[which(ur == agg_data1$meantime_rank[i])]
}
agg_data1$meantime_rank = as.integer(agg_data1$meantime_rank)
agg_data1[, c('Weight', 'Age', 'Release')] = scale(agg_data1[, c('Weight', 'Age', 'Release')] )
```

Binary

Use a linear mixed effects model with stepwise reduction by removal of non-significant effects. Report both full model and reduced model. P-values are determined by likelihood ratio tests (Wilks' theorem) with the relevant fixed effect present or removed.

```
mymodel =stepLMER('Speed', c('Weight', 'Age', 'Bold', 'Release', 'Bold:Release',
                             'Age:Release', 'Weight:Release'), '(1  + Release | ID)', agg_data1)
stepDownSummary(mymodel)
```

```
## [1] "Output: Speed"
## [1] "random Effects: (1  + Release | ID)"
## [1] "The reduced model is: Speed ~ (1  + Release | ID) + Bold + Release"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved       ps
```

```
## [1,] "Bold:Release"   "0.919195975180026"
## [2,] "Weight"         "0.74590408440365"
## [3,] "Age:Release"    "0.484031942448156"
## [4,] "Age"            "0.412501316615666"
## [5,] "Weight:Release" "0.270099237167482"
## [1] "The remaining effect sizes and p-values are:"
##    Effect   Estimate         SE           P
## 1    Bold 0.17153364 0.06606775 1.510087e-02
## 2 Release 0.09436667 0.01559080 3.125414e-06
## [1] "Log Likelihood: 1.84707523170946"
```

Ordinal

Same as above, but using ordinal boldness

```
mymodel =stepLMER('Speed', c('Weight', 'Age', 'meantime_rank', 'Release','meantime_rank:Release',
                             'Age:Release', 'Weight:Release'), '(1 + Release | ID)', agg_data1)

stepDownSummary(mymodel)
```

```
## [1] "Output: Speed"
## [1] "random Effects: (1 + Release | ID)"
## [1] "The reduced model is: Speed ~ (1 + Release | ID) + meantime_rank + Release"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved              ps
## [1,] "Weight"                "0.827489388560485"
## [2,] "meantime_rank:Release" "0.722354554872357"
## [3,] "Age"                   "0.541252417216102"
## [4,] "Age:Release"           "0.495091943722933"
## [5,] "Weight:Release"        "0.259906542924975"
## [1] "The remaining effect sizes and p-values are:"
##          Effect    Estimate          SE           P
## 1 meantime_rank -0.02051814 0.006064458 2.301420e-03
## 2       Release  0.09435171 0.015580688 3.089856e-06
## [1] "Log Likelihood: 3.54080325476494"
```

Efficiency

Test if route efficiency is influenced by age, weight, boldness, release number. We transform raw efficiency to make it amenable to linear modelling by an log inverse transform. Load the data and make aggregate object of all flights

```
eff_results = read.xlsx("./finalData/solo_data2 fixed.xlsx", sheetName="efficiency")

eff_results$ID = as.character(eff_results$ID)


agg_data2 = data.frame(Eff = matrix(NA, nrow=10000, ncol=1),
                       ID = matrix(NA, nrow=10000, ncol=1),
                       Bold = matrix(NA, nrow=10000, ncol=1),
                       Release = matrix(NA, nrow=10000, ncol=1),
                       meantime_rank = matrix(NA, nrow=1000, ncol=1))
```

```
count = 0

for (j in 1:13){
  for (k in 1:23){
    count = count +1
    #log inverse transform
    agg_data2$Eff[count] = log(1/eff_results[k, 1+j]-1)
    agg_data2$ID[count] = eff_results$ID[k]
    agg_data2$Bold[count] = eff_results$Bold[k]
    agg_data2$Release[count] = j
    agg_data2$Weight[count] = eff_results$Weight[k]
    agg_data2$Age[count] = eff_results$Age[k]
    agg_data2$meantime_rank[count] =
      personality_results_navigational$meantime_rank[which(personality_results_navigational$ID
                                                           == agg_data2$ID[count])]


  }
}
agg_data2 = agg_data2[1:count,]
agg_data2$ID = as.factor(agg_data2$ID)

ur = unique(agg_data2$meantime_rank)
rur = rank(ur)
for (i in 1:length(agg_data2$meantime_rank)){
  agg_data2$meantime_rank[i] = rur[which(ur == agg_data2$meantime_rank[i])]
}
agg_data2$meantime_rank = as.integer(agg_data2$meantime_rank)
agg_data2[, c('Weight', 'Age', 'Release')] = scale(agg_data2[, c('Weight', 'Age', 'Release')] )
```

Binary

Use a linear mixed effects model with stepwise reduction by removal of non-significant effects. Report both full model and reduced model. P-values are determined by likelihood ratio tests (Wilks' theorem) with the relevant fixed effect present or removed.

```
mymodel =stepLMER('Eff', c('Weight', 'Age', 'Bold', 'Release', 'Bold:Release',
                           'Age:Release', 'Weight:Release'), '(1 + Release | ID)', agg_data2)

stepDownSummary(mymodel)
```

```
## [1] "Output: Eff"
## [1] "random Effects: (1 + Release | ID)"
## [1] "The reduced model is: Eff ~ (1 + Release | ID) + Release"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved       ps
## [1,] "Weight:Release" "0.717052090016649"
## [2,] "Age"            "0.478647909425873"
## [3,] "Weight"         "0.436657135837934"
## [4,] "Bold:Release"   "0.344639673531073"
## [5,] "Age:Release"    "0.104385556266079"
```

```
## [6,] "Bold"           "0.0525418681012583"
## [1] "The remaining effect sizes and p-values are:"
##    Effect   Estimate         SE            P
## 1 Release -0.4204465 0.04407039 1.375035e-09
## [1] "Log Likelihood: -295.681702147458"
```

Ordinal

Same as above, but using ordinal boldness

```
mymodel =stepLMER('Eff', c('Weight', 'Age', 'meantime_rank', 'Release', 'meantime_rank:Release',
                           'Age:Release', 'Weight:Release'), '(1 + Release | ID)', agg_data2)

stepDownSummary(mymodel)
```

```
## [1] "Output: Eff"
## [1] "random Effects: (1 + Release | ID)"
## [1] "The reduced model is: Eff ~ (1 + Release | ID) + Release"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved              ps
## [1,] "Weight:Release"        "0.653846592051006"
## [2,] "Age"                   "0.655188407170654"
## [3,] "meantime_rank:Release" "0.456245720484676"
## [4,] "Weight"                "0.382860472460453"
## [5,] "meantime_rank"         "0.15469971873383"
## [6,] "Age:Release"           "0.0803902760772555"
## [1] "The remaining effect sizes and p-values are:"
##    Effect   Estimate         SE            P
## 1 Release -0.4204465 0.04407039 1.375035e-09
## [1] "Log Likelihood: -295.681702147458"
```

Fidelity

Test if route fidelity is influenced by age, weight, boldness, release number. We transform raw fidelity to make it amenable to linear modelling by a log inverse transform. Load the data and make aggregate object of all flights

```
fid_results = read.xlsx("./finalData/solo_data2 fixed.xlsx", sheetName="fidelity")
agg_data3 = data.frame(Fid = matrix(NA, nrow=10000, ncol=1),
                       ID = matrix(NA, nrow=10000, ncol=1),
                       Bold = matrix(NA, nrow=10000, ncol=1),
                       Release = matrix(NA, nrow=10000, ncol=1),
                       meantime_rank = matrix(NA, nrow=1000, ncol=1))

count = 0
fid_results$ID = as.character(fid_results$ID)
for (j in 1:12){
  for (k in 1:23){
    count = count +1
    #log inverse transform
    agg_data3$Fid[count] = log(fid_results[k, 1+j])
    agg_data3$ID[count] = fid_results$ID[k]
```

```
    agg_data3$Bold[count] = fid_results$Bold[k]
    agg_data3$Release[count] = j+1
    agg_data3$Weight[count] = fid_results$Weight[k]
    agg_data3$Age[count]= fid_results$Age[k]
    agg_data3$meantime_rank[count] =
      personality_results_navigational$meantime_rank[which(personality_results_navigational$ID
                                                           == agg_data3$ID[count])]
  }
}
agg_data3 = agg_data3[1:count,]
agg_data3$ID = as.factor(agg_data3$ID)

ur = unique(agg_data3$meantime_rank)
rur = rank(ur)
for (i in 1:length(agg_data3$meantime_rank)){
  agg_data3$meantime_rank[i] = rur[which(ur == agg_data3$meantime_rank[i])]
}
agg_data3$meantime_rank = as.integer(agg_data3$meantime_rank)
agg_data3[, c('Weight', 'Age', 'Release')] = scale(agg_data3[, c('Weight', 'Age', 'Release')] )
```

Binary

Use a linear mixed effects model with stepwise reduction by removal of non-significant effects. Report both full model and reduced model. P-values are determined by likelihood ratio tests (Wilks' theorem) with the relevant fixed effect present or removed.

```
mymodel =stepLMER('Fid', c('Weight', 'Age', 'Bold', 'Release', 'Bold:Release',
                           'Age:Release', 'Weight:Release'), '(1 + Release | ID)', agg_data3)

stepDownSummary(mymodel)
```

```
## [1] "Output: Fid"
## [1] "random Effects: (1 + Release | ID)"
## [1] "The reduced model is: Fid ~ (1 + Release | ID) + Release"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved       ps
## [1,] "Bold"           "0.842422073088379"
## [2,] "Bold:Release"   "0.854391558244383"
## [3,] "Weight"         "0.55888232501786"
## [4,] "Age"            "0.356271344166846"
## [5,] "Age:Release"    "0.404961828929119"
## [6,] "Weight:Release" "0.0795021768646972"
## [1] "The remaining effect sizes and p-values are:"
##    Effect   Estimate         SE            P
## 1 Release -0.3494315 0.05754908 3.624687e-06
## [1] "Log Likelihood: -255.087875586632"
```

Ordinal

Same as above, but using ordinal boldness

```
mymodel =stepLMER('Fid', c('Weight', 'Age', 'meantime_rank', 'Release', 'meantime_rank:Release',
                           'Age:Release', 'Weight:Release'), '(1 + Release | ID)', agg_data3)

stepDownSummary(mymodel)
```

```
## [1] "Output: Fid"
## [1] "random Effects: (1 + Release | ID)"
## [1] "The reduced model is: Fid ~ (1 + Release | ID) + Release"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved              ps
## [1,] "meantime_rank:Release" "0.714452784721267"
## [2,] "meantime_rank"         "0.556017456279358"
## [3,] "Weight"                "0.55888232501786"
## [4,] "Age"                   "0.356271344166846"
## [5,] "Age:Release"           "0.404961828929119"
## [6,] "Weight:Release"        "0.0795021768646972"
## [1] "The remaining effect sizes and p-values are:"
##    Effect   Estimate         SE            P
## 1 Release -0.3494315 0.05754908 3.624687e-06
## [1] "Log Likelihood: -255.087875586632"
```

Leadership rankings

Free flight data

Load data from the three groups

```
#load data
group1= read.xlsx("./finalData/TakaDataMaster.xls", sheetName = "1st group")
group2 = read.xlsx("./finalData/TakaDataMaster.xls", sheetName = "2nd group")
group3 = read.xlsx("./finalData/TakaDataMaster.xls", sheetName = "3rd group")
groups = list(group1, group2, group3)
```

Make an aggregated data object across all flights.

```
#Construct aggregated data frame
count = 0;
agg_data_free = data.frame(Rank = matrix(NA, nrow=10000, ncol=1),
                      ID = matrix(NA, nrow=10000, ncol=1),
                      Weight = matrix(NA, nrow=10000, ncol=1),
                      Age = matrix(NA, nrow=10000, ncol=1),
                      Bold = matrix(NA, nrow=10000, ncol=1),
                      meantime_rank = matrix(NA, nrow=1000, ncol=1),
                      meantime = matrix(NA, nrow=1000, ncol=1),
                      Release = matrix(NA, nrow=1000, ncol=1))
for (i in 1:3){

  G = groups[[i]]
  G$ID = as.character(G$ID)
    for (j in 2:(ncol(G)-4)){
      for (k in 1:10){
```

```
        count = count +1
      agg_data_free$Rank[count] = G[k, 1+j]
      agg_data_free$Group[count] = i
      agg_data_free$ID[count] = G$ID[k]
      agg_data_free$Weight[count] = G$Weight[k]
      agg_data_free$Age[count] = G$Age[k]
      agg_data_free$Bold[count] =
        as.double(personality_results$meantime[which(personality_results$ID ==agg_data_free$ID[count])]
      agg_data_free$meantime_rank[count] =
        personality_results$meantime_rank[which(personality_results$ID == agg_data_free$ID[count])]
      agg_data_free$meantime[count] =
        personality_results$meantime[which(personality_results$ID == agg_data_free$ID[count])]
      agg_data_free$Release[count] = j-1
      }
    }
}
agg_data_free = agg_data_free[1:count,]
agg_data_free$ID = as.factor(agg_data_free$ID)
agg_data_free$Rank = as.ordered(agg_data_free$Rank)

ur = unique(agg_data_free$meantime_rank)
rur = rank(ur)
for (i in 1:length(agg_data_free$meantime_rank)){
  agg_data_free$meantime_rank[i] = rur[which(ur == agg_data_free$meantime_rank[i])]
}

agg_data_free[, c('Weight', 'Age', 'Release')] =
  scale(agg_data_free[, c('Weight', 'Age', 'Release')] )
```

Binary

Perform an ordinal regression with random effect on bird ID and fixed effects of age, weight and boldness (binary), along with interactions of these with release, and print model results for each group. We use stepwise removal of non-significant effects and present results from the full and reduced model. P-values are determined by likelihood ratio tests (Wilks' theorem) with the relevant fixed effect present or removed.

```
mymodel =
  stepCLMM('Rank', c('Weight', 'Age', 'Bold', 'Bold:Release',
      'Age:Release', 'Weight:Release', '(Bold | Group)'), '(1 | ID)', agg_data_free)

stepDownSummary(mymodel)
```

```
## [1] "Output: Rank"
## [1] "random Effects: (1 | ID)"
## [1] "The reduced model is: Rank ~ (1 | ID) + Bold"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved       ps
## [1,] "(Bold | Group)" "0.996606782127291"
## [2,] "Weight"         "0.270013630587126"
## [3,] "Age"            "0.272116134357985"
## [4,] "Bold:Release"   "0.233308819762178"
## [5,] "Age:Release"    "0.165483273628336"
```

```
## [6,] "Weight:Release" "0.081242698788125"
## [1] "The remaining effect sizes and p-values are:"
##   Effect Estimate        SE           P
## 1   Bold -2.68307 0.3933666 2.087855e-09
## [1] "Log Likelihood: -390.770589017611"
```

Ordinal

Same as above, but we use boldness rank rather than binary boldness

```
mymodel =
  stepCLMM('Rank', c('Weight', 'Age', 'meantime_rank', 'meantime_rank:Release',
      'Age:Release', 'Weight:Release', '(meantime_rank | Group)'), '(1 | ID)', agg_data_free)

stepDownSummary(mymodel)
```

```
## [1] "Output: Rank"
## [1] "random Effects: (1 | ID)"
## [1] "The reduced model is: Rank ~ (1 | ID) + meantime_rank"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved                ps
## [1,] "(meantime_rank | Group)" "0.902957946252926"
## [2,] "meantime_rank:Release"   "0.36879537267426"
## [3,] "Age"                     "0.310273882174187"
## [4,] "Age:Release"             "0.164652025637229"
## [5,] "Weight:Release"          "0.0908608917054683"
## [6,] "Weight"                  "0.0582227965505357"
## [1] "The remaining effect sizes and p-values are:"
##          Effect  Estimate         SE           P
## 1 meantime_rank 0.2061688 0.02932397 1.824018e-09
## [1] "Log Likelihood: -390.638968052358"
```

Summary of free flight leadership tests

For both binary and ordinal boldness, full models show a strongly significant effect of boldness on leadership rank. Stepwise reduction of non-significant effects in both cases reduces the model to only boldness as a relevant factor.

Navigational flight, unfamiliar site (Noke)

Load data and construct aggregate object of all flights

```
#First download Speed data to add as an individual characteristic where appropriate
speed_results = read.xlsx("./finalData/solo_data2 fixed.xlsx", sheetName="speed")
speed_results$meanspeed = rowMeans(speed_results[, 2:14], na.rm = T)

G = read.xlsx("./finalData/summary_noke_R3 fixed.xlsx", sheetName="Sheet1")

agg_data_noke = data.frame(Rank = matrix(NA, nrow=10000, ncol=1),
                      ID = matrix(NA, nrow=10000, ncol=1),
                      Bold = matrix(NA, nrow=10000, ncol=1),
```

```
                      Age = matrix(NA, nrow=10000, ncol=1),
                      Weight = matrix(NA, nrow=10000, ncol=1),
                      rho = matrix(NA, nrow=10000, ncol=1),
                      meantime_rank = matrix(NA, nrow=10000, ncol=1))


count = 0
G$PigeonID = as.character(G$PigeonID)
G$r1 = as.double(G$r1)
G$r2 = as.double(G$r2)
G$r3 = as.double(G$r3)
G$r4 = as.double(G$r4)
G$r5 = as.double(G$r5)
G$r6 = as.double(G$r6)
G$r7 = as.double(G$r7)
for (j in 1:7){
  for (k in 1:19){
    count = count +1
    agg_data_noke$Rank[count] = G[k, 1+j]
    agg_data_noke$ID[count] = G$PigeonID[k]
    agg_data_noke$Bold[count] = G$Bold[k]
    agg_data_noke$Release[count] = j

    agg_data_noke$Weight[count] = G$Weight[k]
    agg_data_noke$Age[count] = G$Age[k]
    agg_data_noke$meantime_rank[count] =
      personality_results_navigational$meantime_rank[which(
        personality_results_navigational$ID == agg_data_noke$ID[count])]
    agg_data_noke$Bold[count] =
      as.double(personality_results_navigational$meantime[which(
        personality_results_navigational$ID == agg_data_noke$ID[count])] < boldness_threshold)
    agg_data_noke$Speed[count] =
      speed_results$meanspeed[which(speed_results$ID == agg_data_noke$ID[count])]



  }
}

agg_data_noke = agg_data_noke[1:count,]
agg_data_noke$ID = as.factor(agg_data_noke$ID)
agg_data_noke$Rank = as.ordered(agg_data_noke$Rank)

ur = unique(agg_data_noke$meantime_rank)
rur = rank(ur)
for (i in 1:length(agg_data_noke$meantime_rank)){
  agg_data_noke$meantime_rank[i] = rur[which(ur == agg_data_noke$meantime_rank[i])]
}

agg_data_noke[, c('Weight', 'Age', 'Release', 'Speed')] =
  scale(agg_data_noke[, c('Weight', 'Age', 'Release', 'Speed')] )
```

Binary

Perform an ordinal regression with random effect on bird ID and fixed effects of age, weight and boldness (binary), along with interactions of these with release, and print model results for each group. We use stepwise removal of non-significant effects and present results from the full and reduced model. P-values are determined by likelihood ratio tests (Wilks' theorem) with the relevant fixed effect present or removed.

```
mymodel =
  stepCLMM('Rank', c('Weight', 'Age', 'Bold', 'Speed', 'Bold:Release',
          'Age:Release', 'Weight:Release', 'Speed:Release'), '(1 | ID)', agg_data_noke)

stepDownSummary(mymodel)
```

```
## [1] "Output: Rank"
## [1] "random Effects: (1 | ID)"
## [1] "The reduced model is: Rank ~ (1 | ID) + Bold + Speed"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved       ps
## [1,] "Bold:Release"   "0.907835031929207"
## [2,] "Age:Release"    "0.811043848368062"
## [3,] "Age"            "0.765311396773756"
## [4,] "Speed:Release"  "0.507166663715097"
## [5,] "Weight"         "0.352141963091517"
## [6,] "Weight:Release" "0.140293203567661"
## [1] "The remaining effect sizes and p-values are:"
##   Effect   Estimate        SE           P
## 1   Bold -1.8344051 0.4587513 0.000297037
## 2  Speed -0.6921943 0.2279675 0.003126132
## [1] "Log Likelihood: -206.992704835175"
```

Ordinal

Same as above, but we use boldness rank rather than binary boldness

```
mymodel =
  stepCLMM('Rank', c('Weight', 'Age', 'meantime_rank', 'Speed', 'meantime_rank:Release',
        'Age:Release', 'Weight:Release', 'Speed:Release'), '(1 | ID)', agg_data_noke)

stepDownSummary(mymodel)
```

```
## [1] "Output: Rank"
## [1] "random Effects: (1 | ID)"
## [1] "The reduced model is: Rank ~ (1 | ID) + meantime_rank + Speed"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved              ps
## [1,] "Age:Release"           "0.731864471983513"
## [2,] "Speed:Release"         "0.64856477296878"
## [3,] "Age"                   "0.609594772224799"
## [4,] "meantime_rank:Release" "0.496354812460851"
## [5,] "Weight"                "0.394257130735231"
## [6,] "Weight:Release"        "0.171240643961208"
## [1] "The remaining effect sizes and p-values are:"
```

```
##          Effect   Estimate         SE          P
## 1 meantime_rank  0.1816947 0.07237073 0.01635284
## 2         Speed -0.7072088 0.29110198 0.01903521
## [1] "Log Likelihood: -210.654932170271"
```

Navigational flight, familiar site (CH)

Load data and construct aggregate object of all flights

```
#First download Speed data to add as an individual characteristic where appropriate
speed_results = read.xlsx("./finalData/solo_data2 fixed.xlsx", sheetName="speed")
speed_results$meanspeed = rowMeans(speed_results[, 2:14], na.rm = T)

G = read.xlsx("./finalData/summary_ch_R3 fixed.xlsx", sheetName="Sheet1")

agg_data_ch = data.frame(Rank = matrix(NA, nrow=10000, ncol=1),
                         ID = matrix(NA, nrow=10000, ncol=1),
                         rho = matrix(NA, nrow=10000, ncol=1),
                         Bold = matrix(NA, nrow=10000, ncol=1),
                         Weight= matrix(NA, nrow=10000, ncol=1),
                         Age = matrix(NA, nrow=10000, ncol=1),
                         meantime_rank = matrix(NA, nrow=10000, ncol=1),
                         Speed =  matrix(NA, nrow=10000, ncol=1))

count = 0
G$PigeonID = as.character(G$PigeonID)
G$X1 = as.double(G$X1)
G$X2 = as.double(G$X2)
G$X3 = as.double(G$X3)
G$X4 = as.double(G$X4)
for (j in 1:4){
  for (k in 1:23){

    count = count +1
    agg_data_ch$Rank[count] = G[k, 1+j]
    agg_data_ch$ID[count] = G$PigeonID[k]
    agg_data_ch$Weight[count] = G$Weight[k]
    agg_data_ch$Age[count] = G$Age[k]
    agg_data_ch$Release[count] = j
    agg_data_ch$meantime_rank[count] =
      personality_results_navigational$meantime_rank[which(
        personality_results_navigational$ID == agg_data_ch$ID[count])]
    agg_data_ch$Bold[count] =
      as.double(personality_results_navigational$meantime[which(
        personality_results_navigational$ID == agg_data_ch$ID[count])] < boldness_threshold)
    agg_data_ch$Speed[count] =
      speed_results$meanspeed[which(speed_results$ID == agg_data_ch$ID[count])]



  }
}
```

```
agg_data_ch = agg_data_ch[1:count,]
agg_data_ch$ID = as.factor(agg_data_ch$ID)
agg_data_ch$Rank = as.ordered(agg_data_ch$Rank)

ur = unique(agg_data_ch$meantime_rank)
rur = rank(ur)
for (i in 1:length(agg_data_ch$meantime_rank)){
  agg_data_ch$meantime_rank[i] = rur[which(ur == agg_data_ch$meantime_rank[i])]
}

agg_data_ch[, c('Weight', 'Age', 'Release', 'Speed')] =
  scale(agg_data_ch[, c('Weight', 'Age', 'Release', 'Speed')])
```

Binary

Perform an ordinal regression with random effect on bird ID and fixed effects of age, weight and boldness (binary), along with interactions of these with release, and print model results for each group. We use stepwise removal of non-significant effects and present results from the full and reduced model. P-values are determined by likelihood ratio tests (Wilks' theorem) with the relevant fixed effect present or removed.

```
mymodel =
  stepCLMM('Rank', c('Weight', 'Age', 'Bold', 'Speed', 'Bold:Release',
      'Age:Release', 'Weight:Release', 'Speed:Release'), '(1 | ID)', agg_data_ch)

stepDownSummary(mymodel)
```

```
## [1] "Output: Rank"
## [1] "random Effects: (1 | ID)"
## [1] "The reduced model is: Rank ~ (1 | ID) + Bold"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved       ps
## [1,] "Age:Release"    "0.981219864399734"
## [2,] "Weight"         "0.893868988326432"
## [3,] "Speed:Release"  "0.578396140969293"
## [4,] "Age"            "0.528135444663742"
## [5,] "Speed"          "0.361804064034032"
## [6,] "Bold:Release"   "0.276810266550892"
## [7,] "Weight:Release" "0.298619152631559"
## [1] "The remaining effect sizes and p-values are:"
##   Effect  Estimate        SE           P
## 1   Bold -1.693038 0.5429147 0.001750968
## [1] "Log Likelihood: -160.12469706955"
```

Ordinal

Same as above, but we use boldness rank rather than binary boldness

```
mymodel =
  stepCLMM('Rank', c('Weight', 'Age', 'meantime_rank', 'Speed', 'meantime_rank:Release',
     'Age:Release', 'Weight:Release', 'Speed:Release'), '(1 | ID)', agg_data_ch)

stepDownSummary(mymodel)
```

```
## [1] "Output: Rank"
## [1] "random Effects: (1 | ID)"
## [1] "The reduced model is: Rank ~ (1 | ID) + meantime_rank"
## [1] "The following effects were rejected (in order), with associated p-values"
##      effremoved              ps
## [1,] "Age:Release"           "0.942976485169491"
## [2,] "Age"                   "0.848183408833233"
## [3,] "Weight"                "0.671679899926013"
## [4,] "meantime_rank:Release" "0.4773687720148"
## [5,] "Speed"                 "0.442278329356739"
## [6,] "Speed:Release"         "0.337908214707663"
## [7,] "Weight:Release"        "0.271274021854275"
## [1] "The remaining effect sizes and p-values are:"
##          Effect Estimate         SE           P
## 1 meantime_rank 0.153591 0.05497616 0.005179145
## [1] "Log Likelihood: -161.113721130037"
```

Leadership summary

In all 3 contexts, boldness (binary or ordinal) is clearly significant as a factor presicting leadership rank - bolder birds tend to occupy higher leadership positions

Speed selected at Noke (unfamilair site). No other factors selected.

No clear evidence whether binary or ordinal boldness is more predictive

Code to make figures

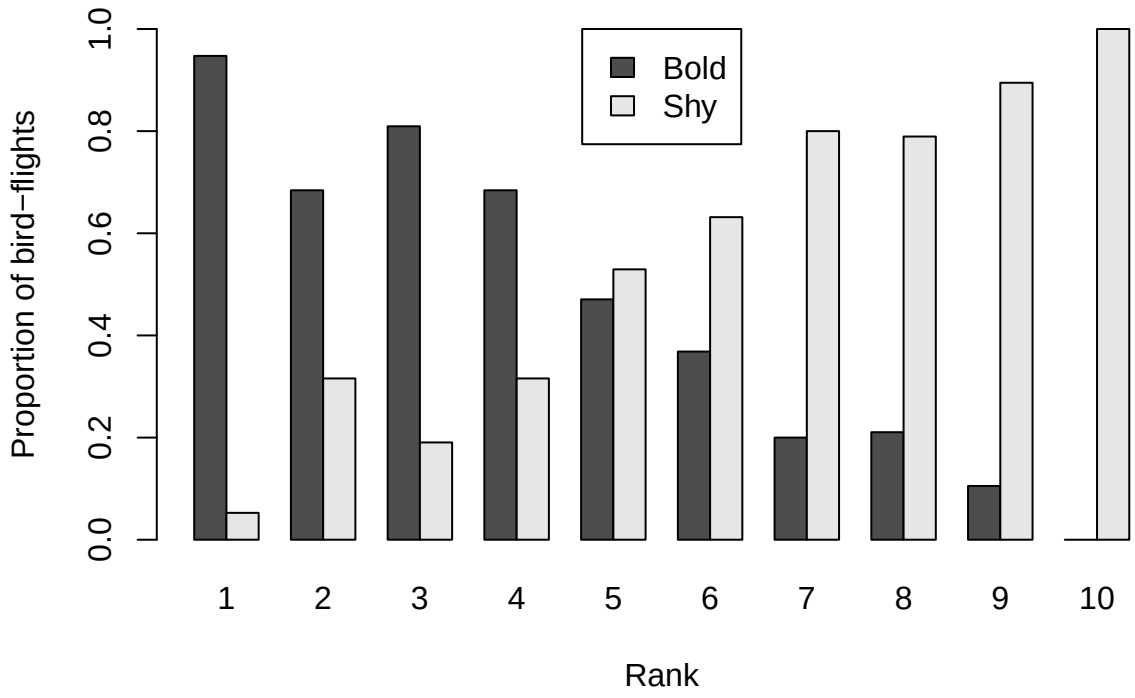
Free flights leadership

```
#Make histogram plot for old (free) data
idx1 = which(agg_data_free$Bold == 1)
idx0 = which(agg_data_free$Bold == 0)
N_Bold = rep(NA, as.double(max(agg_data_free$Rank, na.rm = T)))
N_Shy = N_Bold
for (r in 1:as.double(max(agg_data_free$Rank, na.rm = T))){
  N_Bold[r] = sum(agg_data_free$Rank[idx1] == r, na.rm=TRUE)/
    sum(agg_data_free$Rank == r, na.rm=TRUE)
  N_Shy[r] = sum(agg_data_free$Rank[idx0] == r, na.rm=TRUE)/
    sum(agg_data_free$Rank == r, na.rm=TRUE)
}

Proportions = rbind(N_Bold, N_Shy)
#Create pdf
postscript("./finalPlots/free_bars.eps",
           width=8, height = 8, horizontal = FALSE,
           onefile = FALSE, paper = "special")

barplot(Proportions, beside=TRUE, ylim = c(0, 1),
        names.arg = 1:as.double(max(agg_data_free$Rank, na.rm = T)),
        xlab="Rank", ylab="Proportion of bird-flights", legend=c("Bold", "Shy"),
        args.legend = list(x="top"))
```

```
dev.off()
```

```
## pdf
##   2
```

```
#Plot in documents
barplot(Proportions, beside=TRUE, ylim = c(0, 1),
        names.arg = 1:as.double(max(agg_data_free$Rank, na.rm = T)),
        xlab="Rank", ylab="Proportion of bird-flights", legend=c("Bold", "Shy"),
        args.legend = list(x="top"))
```



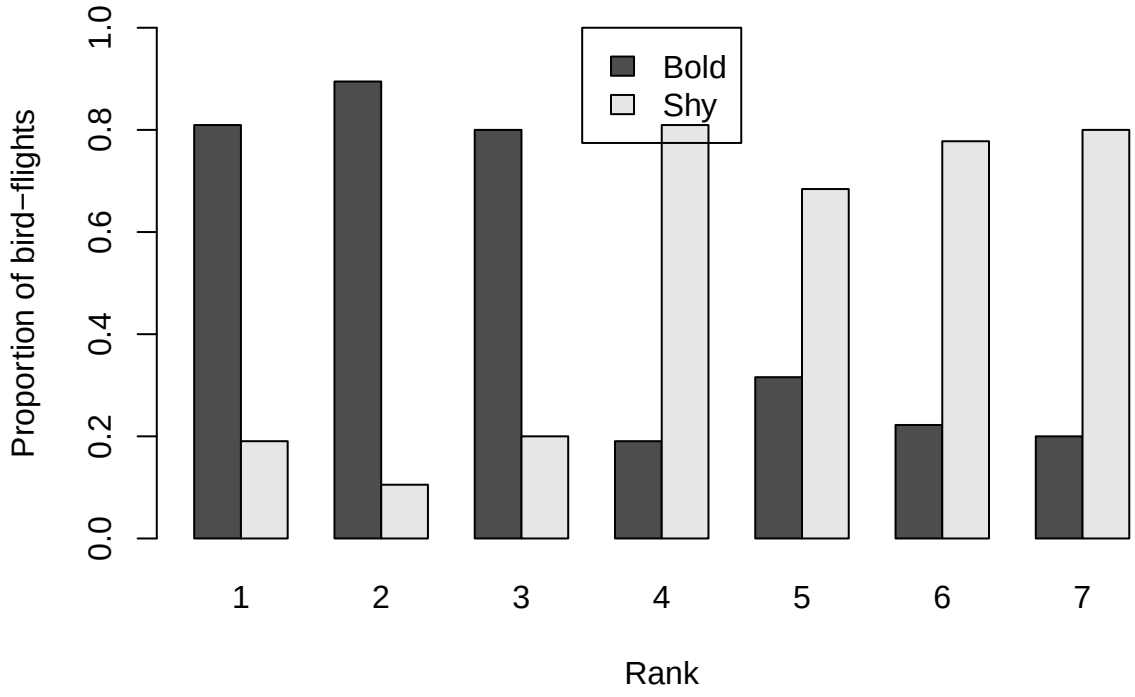
Navigational unfamiliar site (Noke) leadership

```
#Make histogram plot for unfamiliar site data
idx1 = which(agg_data_noke$Bold == 1)
idx0 = which(agg_data_noke$Bold == 0)
N_Bold = rep(NA, as.double(max(agg_data_noke$Rank, na.rm = T)))
N_Shy = N_Bold
for (r in 1:as.double(max(agg_data_noke$Rank, na.rm = T))){
  N_Bold[r] = sum(agg_data_noke$Rank[idx1] == r, na.rm=TRUE)/
    sum(agg_data_noke$Rank == r, na.rm=TRUE)
  N_Shy[r] = sum(agg_data_noke$Rank[idx0] == r, na.rm=TRUE)/
    sum(agg_data_noke$Rank == r, na.rm=TRUE)
}

Proportions = rbind(N_Bold, N_Shy)
#Create pdf
postscript("./finalPlots/noke_bars.eps",
```

```
           width=8, height = 8, horizontal = FALSE,
           onefile = FALSE, paper = "special")

barplot(Proportions, beside=TRUE, ylim = c(0, 1),
        names.arg = 1:as.double(max(agg_data_noke$Rank, na.rm = T)),
        xlab="Rank", ylab="Proportion of bird-flights", legend=c("Bold", "Shy"),
        args.legend = list(x="top"))

dev.off()
```

```
## pdf
##   2
```

```
#plot in document
barplot(Proportions, beside=TRUE, ylim = c(0, 1),
        names.arg = 1:as.double(max(agg_data_noke$Rank, na.rm = T)),
        xlab="Rank", ylab="Proportion of bird-flights", legend=c("Bold", "Shy"),
        args.legend = list(x="top"))
```



Navigational famailiar site (CH) leadership

```
#Make histogram plot for familiar site data
idx1 = which(agg_data_ch$Bold == 1)
idx0 = which(agg_data_ch$Bold == 0)
N_Bold = rep(NA, as.double(max(agg_data_ch$Rank, na.rm = T)))
N_Shy = N_Bold
for (r in 1:as.double(max(agg_data_ch$Rank, na.rm = T))){
  N_Bold[r] = sum(agg_data_ch$Rank[idx1] == r, na.rm=TRUE)/sum(agg_data_ch$Rank == r, na.rm=TRUE)
  N_Shy[r] = sum(agg_data_ch$Rank[idx0] == r, na.rm=TRUE)/sum(agg_data_ch$Rank == r, na.rm=TRUE)
```

```
}

Proportions = rbind(N_Bold, N_Shy)
#Create pdf
postscript("./finalPlots/ch_bars.eps",
           width=8, height = 8, horizontal = FALSE,
           onefile = FALSE, paper = "special")

barplot(Proportions, beside=TRUE, ylim = c(0, 1),
        names.arg = 1:as.double(max(agg_data_ch$Rank, na.rm = T)),
        xlab="Rank", ylab="Proportion of bird-flights", legend=c("Bold", "Shy"),
        args.legend = list(x="top"))

dev.off()
```

```
## pdf
##   2
```

```
#Plot in document
barplot(Proportions, beside=TRUE, ylim = c(0, 1),
        names.arg = 1:as.double(max(agg_data_ch$Rank, na.rm = T)),
        xlab="Rank", ylab="Proportion of bird-flights", legend=c("Bold", "Shy"),
        args.legend = list(x="top"))
```