full analysis for 'Very rapid multi-odour discrimination learning in the ant Lasius niger'

Tomer J. Czaczkes

02/07/2020

Contents

Introduction

In this study, we examined differential associative learning in the ant Lasius niger. We attempt to train ants to associate different odours with different food qualities, and ask whether they prefer the odour associated with the highest quality.

Libraries

```
library(readxl)
library(tidyverse)
library (lme4)
library (emmeans) # contrasts
```

Data import

```
roseVblackberry <- read_excel("data.xlsx",
    sheet = "innate pref rose vs blackberry")

orangeVblackberry <- read_excel("data.xlsx",
    sheet = "innate pref orange vs blackberr")

roseVorange <- read_excel("data.xlsx", sheet = "innate pref orange vs rose")

two_odour <- read_excel("data.xlsx", sheet = "2odour 4 training visits")

three_odour <- read_excel("data.xlsx", sheet = "3odour 3 training visits")


four_odour_2visit <- read_excel("data.xlsx",
    sheet = "4odour 2 training visit")

four_odour_1visit <- read_excel("data.xlsx",
    sheet = "4odour 1 training visit")
```

Innate odour preference

rose vs blackberry

```
   rose<- sum (roseVblackberry$went.rose)
   blackberry<- sum (roseVblackberry$went.blackberry)

   binom.test(rose, blackberry + rose, 0.5) # binomial test
```

```
##
##  Exact binomial test
##
## data:  rose and blackberry + rose
## number of successes = 378, number of trials = 820, p-value =
## 0.02774
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.4264408 0.4957919
```

```
## sample estimates:
## probability of success
##              0.4609756
```

ants slightly prefer blackberry to rose

rose vs orange

```
orange<- sum (roseVorange$went.orange)
other<- sum (roseVorange$went.other)

binom.test(orange, orange + other, 0.5) # binomial test
```

```
##
##  Exact binomial test
##
## data:  orange and orange + other
## number of successes = 128, number of trials = 240, p-value =
## 0.3329
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.4680579 0.5977753
## sample estimates:
## probability of success
##              0.5333333
```

no preference between orange and rose

blackberry vs orange

```
orange<- sum (orangeVblackberry$went.orange)
other<- sum (orangeVblackberry$went.other)

binom.test(orange, orange + other, 0.5) # binomial test
```

```
##
##  Exact binomial test
##
## data:  orange and orange + other
## number of successes = 81, number of trials = 180, p-value = 0.205
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.3759101 0.5257639
## sample estimates:
## probability of success
##                   0.45
```

no preference between orange and blackberry

Two odours four visits

First we pull out only visit 10, which is the first choice each ant makes. We recorded many repeated choices, but using them would be pseudoreplication.

```
two_odour_visit10 <- subset (two_odour, visit == 10)
```

Now to look at the data and test it:

```
count (two_odour_visit10, correct.initial)
```

```
## # A tibble: 2 x 2
##   correct.initial     n
##             <dbl> <int>
## 1               0     1
## 2               1    16
```

```
#ok 16 / 17 went in the right direction

 binom.test(16, 17, 0.5)
```

```
##
##  Exact binomial test
##
## data:  16 and 17
## number of successes = 16, number of trials = 17, p-value =
## 0.0002747
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.7131106 0.9985118
## sample estimates:
## probability of success
##              0.9411765
```

```
 #figure not really needed
```

Strong preference 16/17 for correct arm

Three odours three visits

```
three_odour_test <- subset (three_odour, phase == "Testing")

m1 <- lmer (
  correct.final ~
       current.quality * good.side.at.test + (1|colony),
      family = binomial,
      data = three_odour_test)
```

```
## boundary (singular) fit: see ?isSingular
```

```r
summary (m1)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula:
## correct.final ~ current.quality * good.side.at.test + (1 | colony)
##    Data: three_odour_test
## Control:
## structure(list(optimizer = c("bobyqa", "Nelder_Mead"), calc.derivs = TRUE,
##     use.last.params = FALSE, restart_edge = FALSE, boundary.tol = 1e-05,
##     tolPwrss = 1e-07, compDev = TRUE, nAGQ0initStep = TRUE, checkControl = list(
##         check.nobs.vs.rankZ = "ignore", check.nobs.vs.nlev = "stop",
##         check.nlev.gtreq.5 = "ignore", check.nlev.gtr.1 = "stop",
##         check.nobs.vs.nRE = "stop", check.rankX = "message+drop.cols",
##         check.scaleX = "warning", check.formula.LHS = "stop",
##         check.response.not.const = "stop"), checkConv = list(
##         check.conv.grad = list(action = "warning", tol = 0.001,
##             relTol = NULL), check.conv.singular = list(action = "message",
##             tol = 1e-04), check.conv.hess = list(action = "warning",
##             tol = 1e-06)), optCtrl = list()), class = c("glmerControl",
## "merControl"))
##
##      AIC      BIC   logLik deviance df.resid
##     88.0    105.3    -37.0     74.0       80
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.7417  0.2673  0.4082  0.4082  0.7071
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  colony (Intercept) 0        0
## Number of obs: 87, groups:  colony, 5
##
## Fixed effects:
##                                                Estimate Std. Error
## (Intercept)                                      0.6931     0.5477
## current.quality0.2 v 1.8                         1.9459     1.1711
## current.quality0.6 v 1.8                         1.0986     0.9399
## good.side.at.testright                           1.0986     0.9399
## current.quality0.2 v 1.8:good.side.at.testright -1.1727     1.7412
## current.quality0.6 v 1.8:good.side.at.testright -1.8788     1.3445
##                                                z value Pr(>|z|)
## (Intercept)                                      1.266   0.2057
## current.quality0.2 v 1.8                         1.662   0.0966 .
## current.quality0.6 v 1.8                         1.169   0.2424
## good.side.at.testright                           1.169   0.2424
## current.quality0.2 v 1.8:good.side.at.testright -0.674   0.5006
## current.quality0.6 v 1.8:good.side.at.testright -1.397   0.1623
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Correlation of Fixed Effects:
##             (Intr) cr.0.2v1.8 cr.0.6v1.8 gd.s.. c.0.2v1.8:
## crr.0.2v1.8 -0.468
## crr.0.6v1.8 -0.583  0.273
## gd.sd.t.tst -0.583  0.273      0.340
## c.0.2v1.8:.  0.315 -0.673     -0.183     -0.540
## c.0.6v1.8:.  0.407 -0.191     -0.699     -0.699  0.377
## convergence code: 0
## boundary (singular) fit: see ?isSingular
```

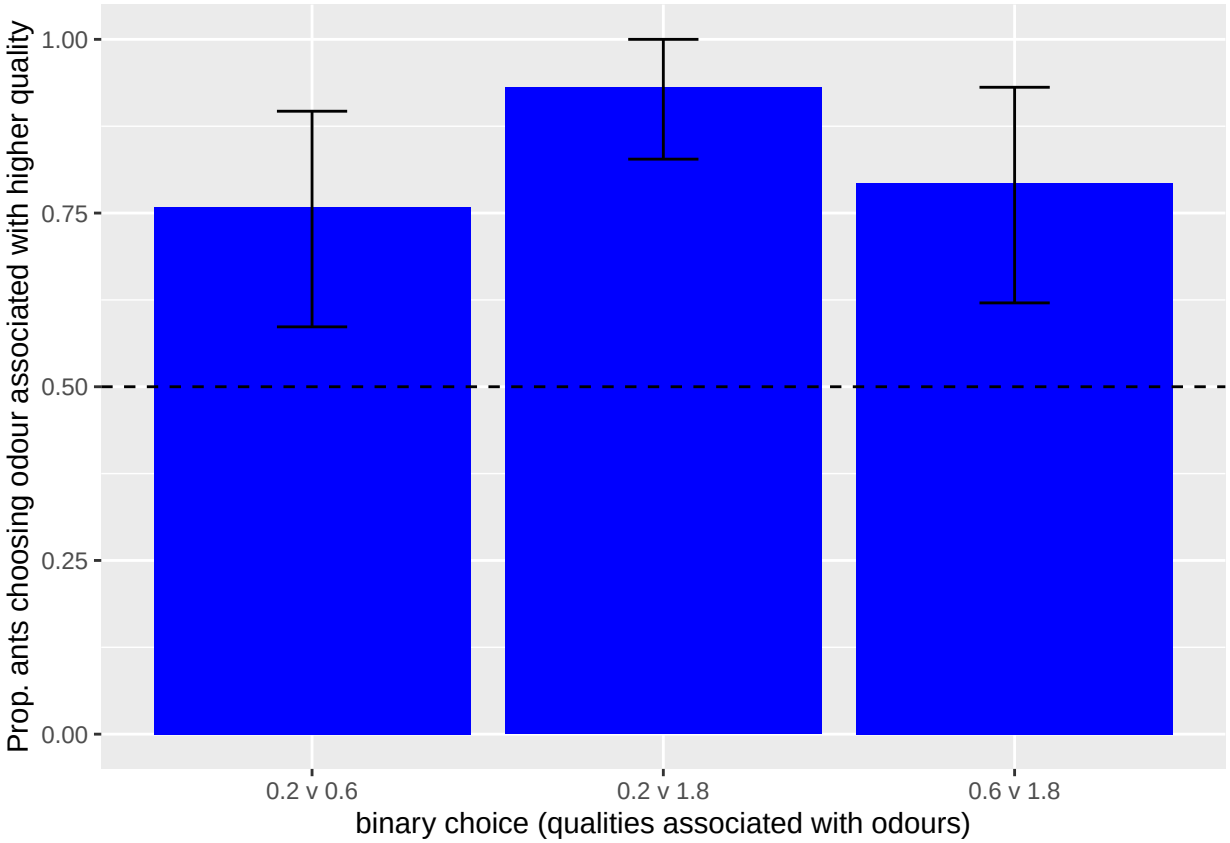
No effect of side, all groups the same, no colony-level differences.

Plots - 3 odours 3 visits

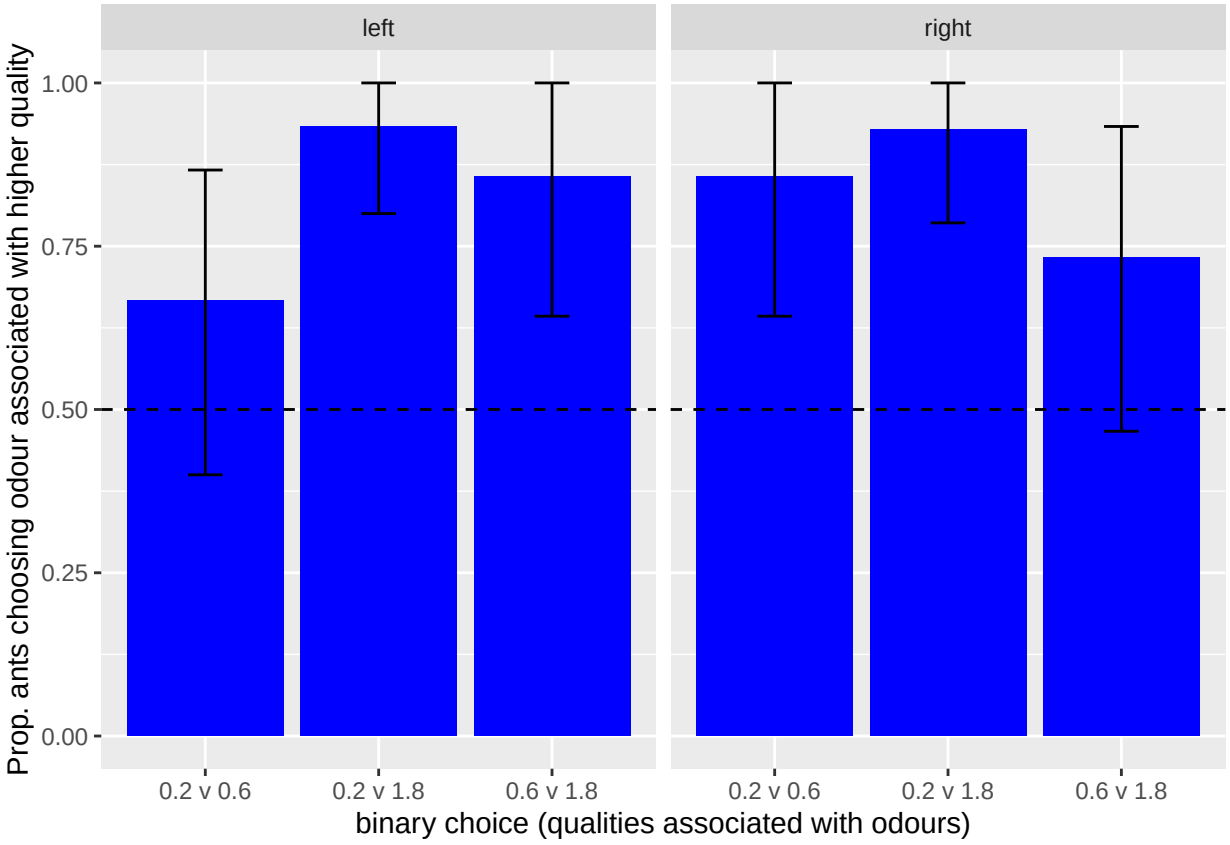
Some plots to see the data clearly, splitting by reward side.

```
threeodour_plot <- ggplot (three_odour_test, aes(x = current.quality, y = correct.final))+
  scale_y_continuous(limits = c(0, 1)) +
  stat_summary(fun.y = "mean", geom = "bar", fill="blue") +
    stat_summary(fun.data = "mean_cl_boot", geom = "errorbar", width = 0.2) +
  geom_abline(intercept = 0.5, slope = 0, color = "black", linetype = 2) +
  xlab("binary choice (qualities associated with odours)") +
  ylab ("Prop. ants choosing odour associated with higher quality")

print (threeodour_plot)
```



```
print (threeodour_plot + facet_wrap (~ good.side.at.test))
```



Binomial tests for each group

```
count (three_odour_test, current.quality, correct.final)
```

```
## # A tibble: 6 x 3
##   current.quality correct.final     n
##   <chr>                   <dbl> <int>
## 1 0.2 v 0.6                   0     7
## 2 0.2 v 0.6                   1    22
## 3 0.2 v 1.8                   0     2
## 4 0.2 v 1.8                   1    27
## 5 0.6 v 1.8                   0     6
## 6 0.6 v 1.8                   1    23
```

0.2 vs 0.6

```
binom.test(22, 29, 0.5)
```

```
##
##  Exact binomial test
```

```
##
## data:  22 and 29
## number of successes = 22, number of trials = 29, p-value = 0.00813
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5645997 0.8970164
## sample estimates:
## probability of success
##              0.7586207
```

0.2 vs 1.8

```
binom.test(27, 29, 0.5)
```

```
##
##  Exact binomial test
##
## data:  27 and 29
## number of successes = 27, number of trials = 29, p-value =
## 1.624e-06
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.7723381 0.9915360
## sample estimates:
## probability of success
##              0.9310345
```

0.6 vs 1.8

```
binom.test(23, 29, 0.5)
```

```
##
##  Exact binomial test
##
## data:  23 and 29
## number of successes = 23, number of trials = 29, p-value =
## 0.002316
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6027531 0.9200582
## sample estimates:
## probability of success
##              0.7931034
```

Four odours one visits

first, let's test whether the colony ID has any effect

```
m4.1 <- lmer (
  correct.final ~
```

```
       current.quality * good.side.at.test + (1|colony),
      family = binomial,
      data = four_odour_1visit)
```

```
## boundary (singular) fit: see ?isSingular
```

```
summary (m4.1)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula:
## correct.final ~ current.quality * good.side.at.test + (1 | colony)
##    Data: four_odour_1visit
## Control:
## structure(list(optimizer = c("bobyqa", "Nelder_Mead"), calc.derivs = TRUE,
##     use.last.params = FALSE, restart_edge = FALSE, boundary.tol = 1e-05,
##     tolPwrss = 1e-07, compDev = TRUE, nAGQ0initStep = TRUE, checkControl = list(
##         check.nobs.vs.rankZ = "ignore", check.nobs.vs.nlev = "stop",
##         check.nlev.gtreq.5 = "ignore", check.nlev.gtr.1 = "stop",
##         check.nobs.vs.nRE = "stop", check.rankX = "message+drop.cols",
##         check.scaleX = "warning", check.formula.LHS = "stop",
##         check.response.not.const = "stop"), checkConv = list(
##         check.conv.grad = list(action = "warning", tol = 0.001,
##             relTol = NULL), check.conv.singular = list(action = "message",
##             tol = 1e-04), check.conv.hess = list(action = "warning",
##             tol = 1e-06)), optCtrl = list()), class = c("glmerControl",
## "merControl"))
##
##      AIC      BIC   logLik deviance df.resid
##    162.3    182.3    -74.1    148.3      122
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.0615 -0.7845  0.5423  0.6124  1.2748
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  colony (Intercept) 0        0
## Number of obs: 129, groups:  colony, 5
##
## Fixed effects:
##                                                 Estimate Std. Error
## (Intercept)                                       0.9808     0.4787
## current.quality0.4 vs 0.8                         0.4661     0.7335
## current.quality0.8 vs1.6                          0.1823     0.7012
## good.side.at.testright                           -1.4663     0.6566
## current.quality0.4 vs 0.8:good.side.at.testright  0.7816     0.9744
## current.quality0.8 vs1.6:good.side.at.testright   1.5270     0.9759
##                                                 z value Pr(>|z|)
## (Intercept)                                       2.049   0.0405 *
## current.quality0.4 vs 0.8                         0.635   0.5251
```

```
## current.quality0.8 vs1.6                            0.260   0.7949
## good.side.at.testright                              -2.233   0.0255 *
## current.quality0.4 vs 0.8:good.side.at.testright     0.802   0.4225
## current.quality0.8 vs1.6:good.side.at.testright      1.565   0.1177
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) cr.0.4v0.8 cr.0.8v1.6 gd.s.. c.0.4v0.8:
## crr.0.4v0.8 -0.653
## crr.0.8v1.6 -0.683  0.446
## gd.sd.t.tst -0.729  0.476      0.498
## c.0.4v0.8:.  0.491 -0.753     -0.335     -0.674
## c.0.8v1.6:.  0.491 -0.320     -0.718     -0.673  0.453
## convergence code: 0
## boundary (singular) fit: see ?isSingular
```

note the estimate of colony's variance = 0, which means that there is no effect of colony as a random effect. We can ignore it, meaning that binomial tests will give us reliable results .

Ants have a left bias

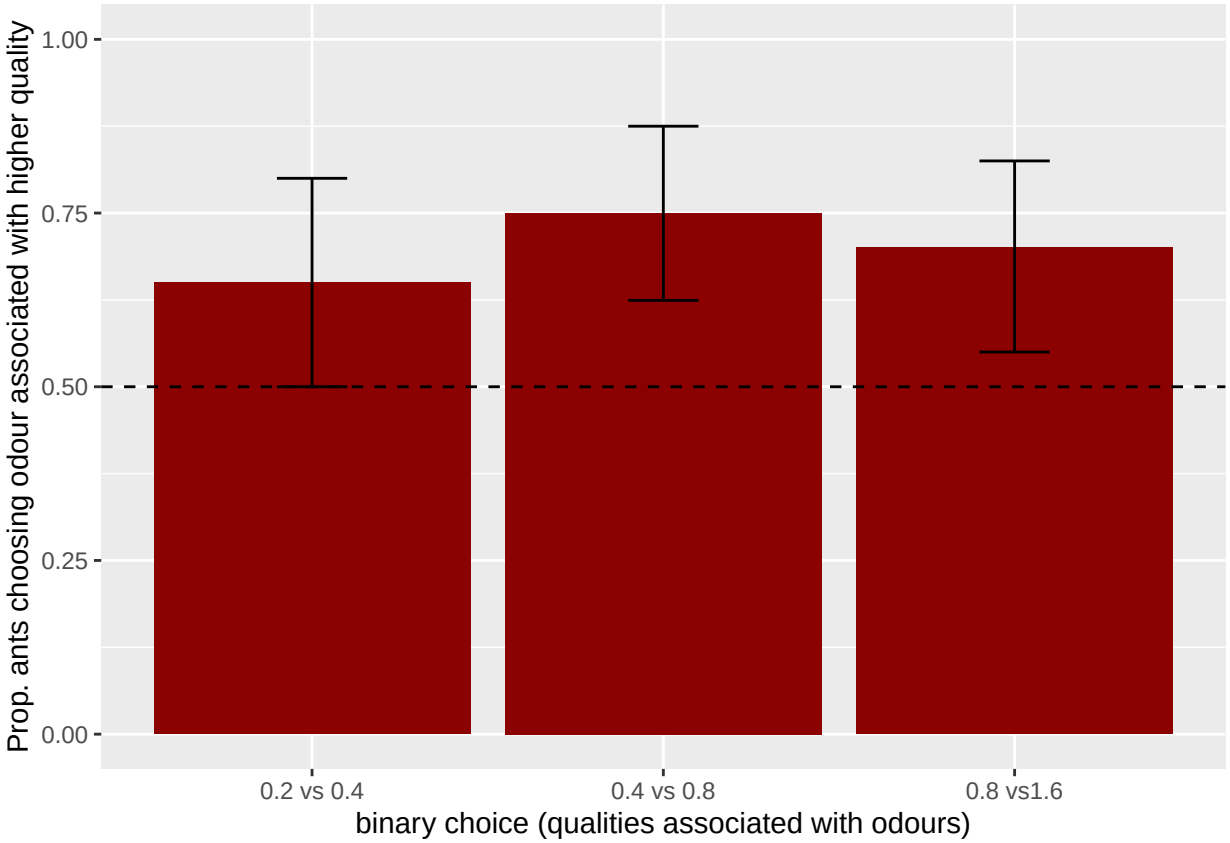
Plots - 4 odours 1 visit

note the left bias

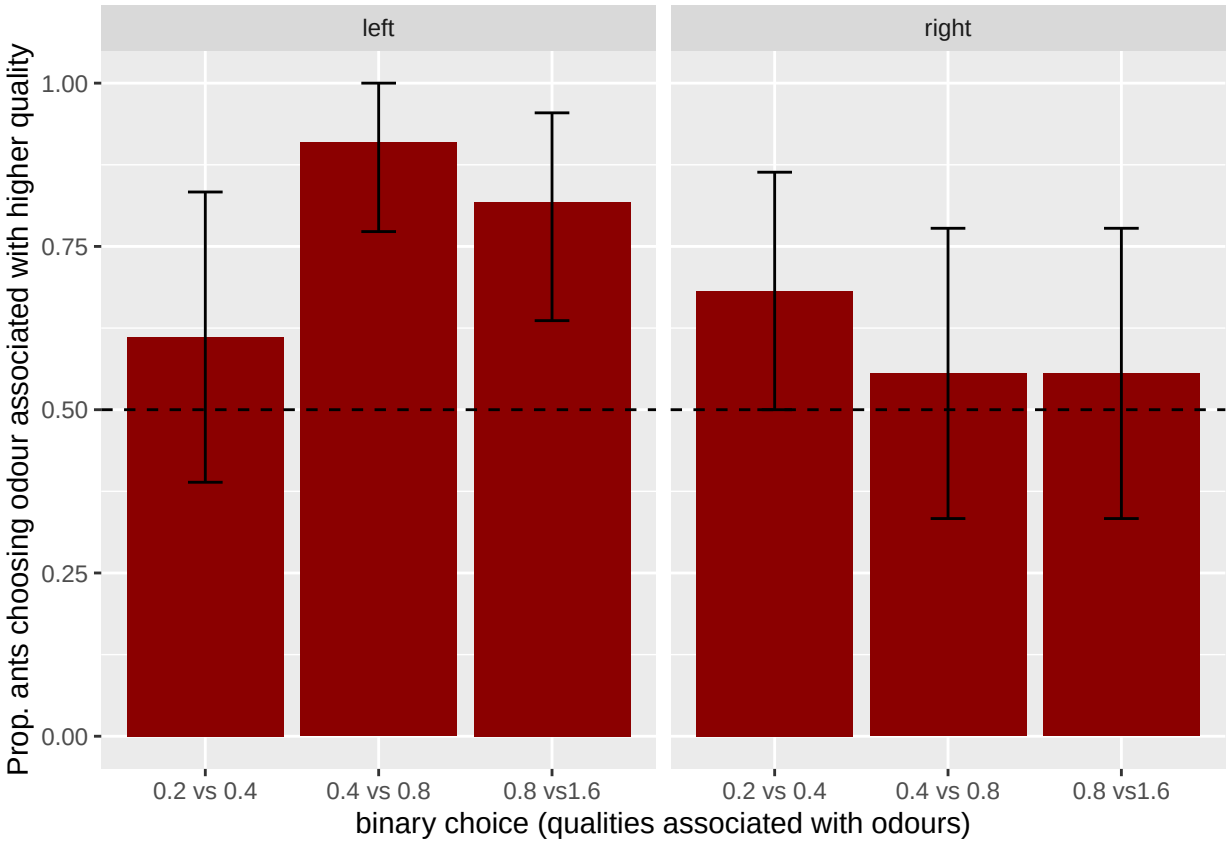
```
fourodour_twovisit_plot <- filter (four_odour_2visit, phase == "Testing") %>% # just need to get rid of

  ggplot (aes(x = current.quality, y = correct.final))+
  scale_y_continuous(limits = c(0, 1)) +                          # to coerce the Y axis to go from 0 to
  stat_summary(fun.y = "mean", geom = "bar", fill="dark red") +
    stat_summary(fun.data = "mean_cl_boot", geom = "errorbar", width = 0.2) +
  geom_abline(intercept = 0.5, slope = 0, color = "black", linetype = 2) +
  xlab("binary choice (qualities associated with odours)") +
  ylab ("Prop. ants choosing odour associated with higher quality")

print (fourodour_twovisit_plot)
```




```
print (fourodour_twovisit_plot + facet_wrap( ~ good.side.at.test))
```



Binomial tests for each group

```
filter (four_odour_2visit, phase == "Testing") %>%
  count (current.quality, correct.final)
```

```
## # A tibble: 6 x 3
##   current.quality correct.final     n
##   <chr>                   <dbl> <int>
## 1 0.2 vs 0.4                  0    14
## 2 0.2 vs 0.4                  1    26
## 3 0.4 vs 0.8                  0    10
## 4 0.4 vs 0.8                  1    30
## 5 0.8 vs1.6                   0    12
## 6 0.8 vs1.6                   1    28
```

0.2 vs 0.4

```
binom.test(26, 40, 0.5)
```

```
##
##  Exact binomial test
##
## data:  26 and 40
```

```
## number of successes = 26, number of trials = 40, p-value = 0.08069
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.4831555 0.7937175
## sample estimates:
## probability of success
##                   0.65
```

0.4 vs 0.8

```
binom.test(30, 40, 0.5)
```

```
##
##  Exact binomial test
##
## data:  30 and 40
## number of successes = 30, number of trials = 40, p-value =
## 0.002221
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5880380 0.8730852
## sample estimates:
## probability of success
##                   0.75
```

0.8 vs 1.6

```
binom.test(28, 40, 0.5)
```

```
##
##  Exact binomial test
##
## data:  28 and 40
## number of successes = 28, number of trials = 40, p-value = 0.01659
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5346837 0.8343728
## sample estimates:
## probability of success
##                    0.7
```

0.2 vs 0.4 is marginal, other two are significant.

Four odours two visits

first, let's test whether the colony ID has any effect

```
m4.2 <- lmer (
  correct.final ~
       current.quality * good.side.at.test +  (1|colony),
      family = binomial,
      data = four_odour_2visit)
```

```
## boundary (singular) fit: see ?isSingular
```

```
summary (m4.2)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula:
## correct.final ~ current.quality * good.side.at.test + (1 | colony)
##    Data: four_odour_2visit
## Control:
## structure(list(optimizer = c("bobyqa", "Nelder_Mead"), calc.derivs = TRUE,
##     use.last.params = FALSE, restart_edge = FALSE, boundary.tol = 1e-05,
##     tolPwrss = 1e-07, compDev = TRUE, nAGQ0initStep = TRUE, checkControl = list(
##         check.nobs.vs.rankZ = "ignore", check.nobs.vs.nlev = "stop",
##         check.nlev.gtreq.5 = "ignore", check.nlev.gtr.1 = "stop",
##         check.nobs.vs.nRE = "stop", check.rankX = "message+drop.cols",
##         check.scaleX = "warning", check.formula.LHS = "stop",
##         check.response.not.const = "stop"), checkConv = list(
##         check.conv.grad = list(action = "warning", tol = 0.001,
##             relTol = NULL), check.conv.singular = list(action = "message",
##             tol = 1e-04), check.conv.hess = list(action = "warning",
##             tol = 1e-06)), optCtrl = list()), class = c("glmerControl",
## "merControl"))
##
##      AIC      BIC   logLik deviance df.resid
##    149.3    168.8    -67.7    135.3      113
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1623 -1.1180  0.4714  0.7977  0.8944
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  colony (Intercept) 0        0
## Number of obs: 120, groups:  colony, 5
##
## Fixed effects:
##                                                 Estimate Std. Error
## (Intercept)                                       0.4520     0.4835
## current.quality0.4 vs 0.8                         1.8506     0.8853
## current.quality0.8 vs1.6                          1.0521     0.7344
## good.side.at.testright                            0.3102     0.6658
## current.quality0.4 vs 0.8:good.side.at.testright -2.3896     1.1038
## current.quality0.8 vs1.6:good.side.at.testright  -1.5911     0.9868
##                                                 z value Pr(>|z|)
## (Intercept)                                       0.935   0.3499
## current.quality0.4 vs 0.8                         2.090   0.0366 *
## current.quality0.8 vs1.6                          1.433   0.1520
## good.side.at.testright                            0.466   0.6413
## current.quality0.4 vs 0.8:good.side.at.testright -2.165   0.0304 *
## current.quality0.8 vs1.6:good.side.at.testright  -1.612   0.1069
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation of Fixed Effects:
##             (Intr) cr.0.4v0.8 cr.0.8v1.6 gd.s.. c.0.4v0.8:
## crr.0.4v0.8 -0.546
## crr.0.8v1.6 -0.658  0.360
## gd.sd.t.tst -0.726  0.397      0.478
## c.0.4v0.8:.  0.438 -0.802     -0.288     -0.603
## c.0.8v1.6:.  0.490 -0.268     -0.744     -0.675  0.407
## convergence code: 0
## boundary (singular) fit: see ?isSingular
```

note the estimate of colony's variance = 0, which means that there is no effect of colony as a random effect. We can ignore it, meaning that binimal tests will give us reliable results Significant interaction - 0.4vs0.8 interacts with side. This means (see figure below) that it is different from 0.2vs0.4 but only when the correct answer is on the right. Interestingly, side is not significant on its own. This means that its not always informative.

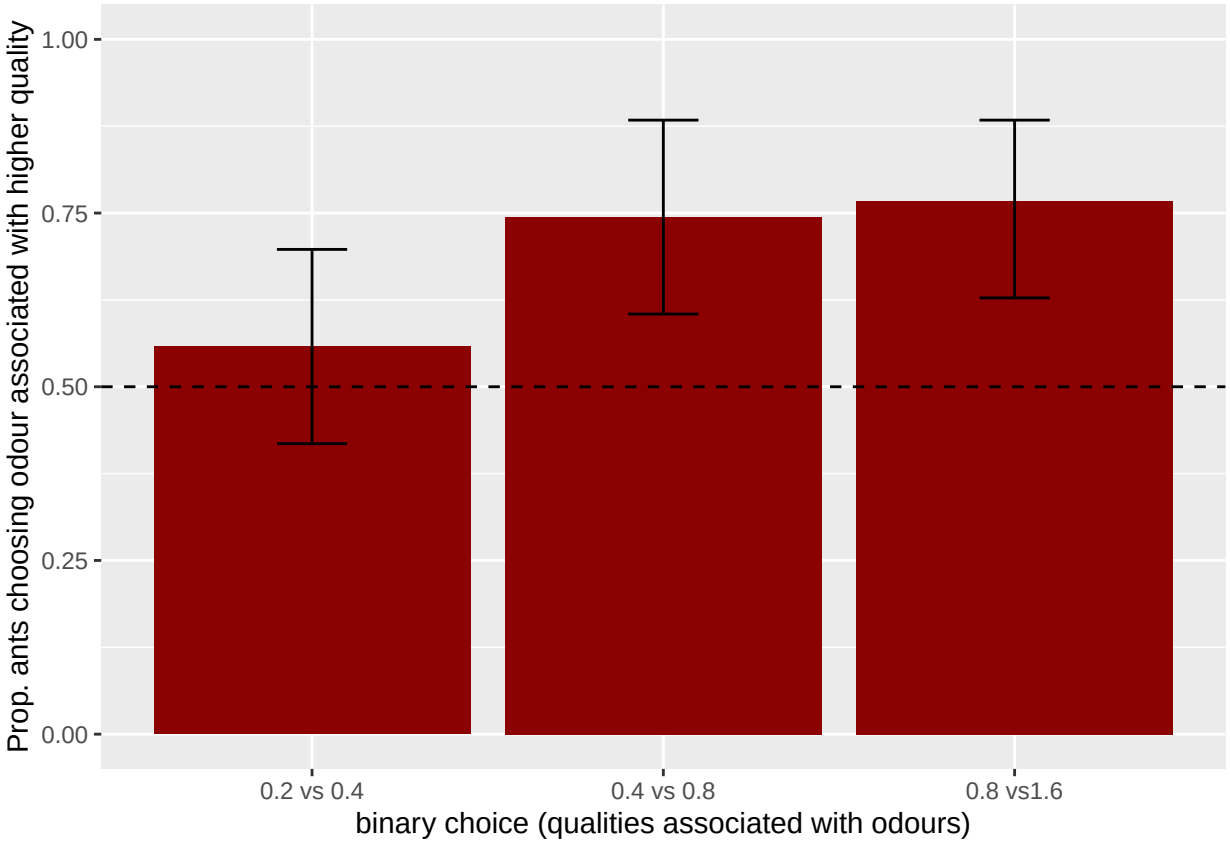
Plots - 4 odours 2 visits

note the side bias

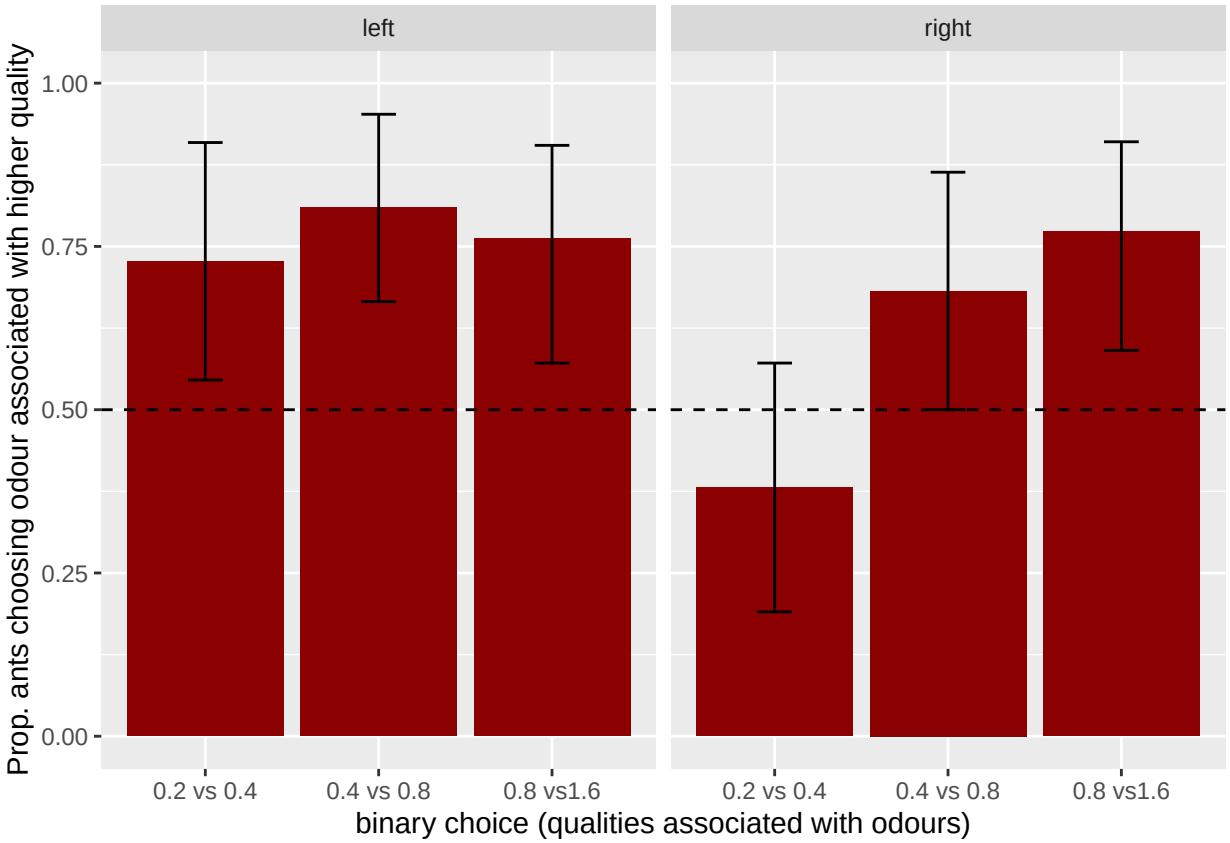
```
fourodour_onevisit_plot <- filter (four_odour_1visit, phase == "Testing") %>% # just need to get rid of

  ggplot (aes(x = current.quality, y = correct.final))+
  scale_y_continuous(limits = c(0, 1)) +                          # to coerce the Y axis to go from 0 to
  stat_summary(fun.y = "mean", geom = "bar", fill="dark red") +
    stat_summary(fun.data = "mean_cl_boot", geom = "errorbar", width = 0.2) +
  geom_abline(intercept = 0.5, slope = 0, color = "black", linetype = 2) +
  xlab("binary choice (qualities associated with odours)") +
  ylab ("Prop. ants choosing odour associated with higher quality")

print (fourodour_onevisit_plot)
```




```
print (fourodour_onevisit_plot + facet_wrap( ~ good.side.at.test))
```



Binomial tests for each group

```
filter (four_odour_1visit, phase == "Testing") %>%
  count (current.quality, correct.final)
```

```
## # A tibble: 6 x 3
##   current.quality correct.final     n
##   <chr>                   <dbl> <int>
## 1 0.2 vs 0.4                  0    19
## 2 0.2 vs 0.4                  1    24
## 3 0.4 vs 0.8                  0    11
## 4 0.4 vs 0.8                  1    32
## 5 0.8 vs1.6                   0    10
## 6 0.8 vs1.6                   1    33
```

0.2 vs 0.4

```
binom.test(24, 43, 0.5)
```

```
##
##  Exact binomial test
##
## data:  24 and 43
```

```
## number of successes = 24, number of trials = 43, p-value = 0.5424
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.3987539 0.7092188
## sample estimates:
## probability of success
##              0.5581395
```

0.4 vs 0.8

```
binom.test(32, 43, 0.5)
```

```
##
##  Exact binomial test
##
## data:  32 and 43
## number of successes = 32, number of trials = 43, p-value =
## 0.001914
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5882843 0.8648140
## sample estimates:
## probability of success
##               0.744186
```

0.8 vs 1.6

```
binom.test(33, 43, 0.5)
```

```
##
##  Exact binomial test
##
## data:  33 and 43
## number of successes = 33, number of trials = 43, p-value =
## 0.0006061
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.6136918 0.8824464
## sample estimates:
## probability of success
##              0.7674419
```

0.2 vs 0.4 is not significant , other two are significant.

Session info

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18363)
##
```

```
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252
## [2] LC_CTYPE=English_United Kingdom.1252
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] emmeans_1.4.4   lme4_1.1-21     Matrix_1.2-17   forcats_0.4.0
##  [5] stringr_1.4.0   dplyr_0.8.3     purrr_0.3.3     readr_1.3.1
##  [9] tidyr_1.0.0     tibble_2.1.3    ggplot2_3.2.1   tidyverse_1.3.0
## [13] readxl_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-140        fs_1.3.1            lubridate_1.7.4
##  [4] RColorBrewer_1.1-2  httr_1.4.1          tools_3.6.1
##  [7] backports_1.1.4     utf8_1.1.4          R6_2.4.0
## [10] rpart_4.1-15        Hmisc_4.3-0         DBI_1.0.0
## [13] lazyeval_0.2.2      colorspace_1.4-1    nnet_7.3-12
## [16] withr_2.1.2         tidyselect_0.2.5    gridExtra_2.3
## [19] compiler_3.6.1      cli_1.1.0           rvest_0.3.5
## [22] htmlTable_1.13.3    xml2_1.2.2          sandwich_2.5-1
## [25] labeling_0.3        scales_1.0.0        checkmate_1.9.4
## [28] mvtnorm_1.0-11      digest_0.6.20       foreign_0.8-71
## [31] minqa_1.2.4         rmarkdown_1.15      base64enc_0.1-3
## [34] jpeg_0.1-8.1        pkgconfig_2.0.2     htmltools_0.4.0
## [37] dbplyr_1.4.2        htmlwidgets_1.5.1   rlang_0.4.2
## [40] rstudioapi_0.10     generics_0.0.2      zoo_1.8-6
## [43] jsonlite_1.6        acepack_1.4.1       magrittr_1.5
## [46] Formula_1.2-3       Rcpp_1.0.2          munsell_0.5.0
## [49] fansi_0.4.0         lifecycle_0.1.0     stringi_1.4.3
## [52] multcomp_1.4-11     yaml_2.2.0          MASS_7.3-51.4
## [55] grid_3.6.1          crayon_1.3.4        lattice_0.20-38
## [58] haven_2.2.0         splines_3.6.1       hms_0.5.2
## [61] zeallot_0.1.0       knitr_1.24          pillar_1.4.2
## [64] boot_1.3-22         estimability_1.3    codetools_0.2-16
## [67] reprex_0.3.0        glue_1.3.1          evaluate_0.14
## [70] latticeExtra_0.6-29 data.table_1.12.2   modelr_0.1.5
## [73] vctrs_0.2.0         png_0.1-7           nloptr_1.2.1
## [76] cellranger_1.1.0    gtable_0.3.0        assertthat_0.2.1
## [79] xfun_0.9            xtable_1.8-4        broom_0.5.2
## [82] survival_2.44-1.1   cluster_2.1.0       TH.data_1.0-10
```