Predicting Visual Search Slopes

Dr Alasdair Clarke, Senior Lecturer in Psychology, University of Essex, UK & Dr Anna Hughes, Lecturer in Psychology, University of Essex, UK

Answer 1:

```
d <- read.csv("data/accuracy_rt_data.txt")
```

Answer 2:

```
head(d)
```

```
                                  imageFileName observer block feature
n
1 images/exp2a_trial_240_colour1_distractors4_l.png        4    2a       1
0
2 images/exp2c_trial_102_colour2_distractors4_r.png        4    2c       2
1
3 images/exp2a_trial_177_colour1_distractors9_l.png        4    2a       1
2
4 images/exp1a_trial_272_colour1_distractors4_l.png        4    1a       1
3
5 images/exp1a_trial_46_colour3_distractors31_r.png        4    1a       3
4
6 images/exp2b_trial_161_colour1_distractors4_r.png        4    2b       1
5
  distractor_no        rt accuracy
1             4 0.5825252        1
2             4 0.5564872        1
3             9 0.5042450        1
4             4 0.7859964        1
5            31 0.6468332        1
6             4 0.6718330        1
```

Answer 3:

```
summary(d)
```

```
 imageFileName         observer        block             feature
 Length:64000       Min.   : 1.00   Length:64000       Min.   :0.000
 Class :character   1st Qu.:10.75   Class :character   1st Qu.:1.000
 Mode  :character   Median :20.50   Mode  :character   Median :2.000
                    Mean   :20.50                      Mean   :1.875
                    3rd Qu.:30.25                      3rd Qu.:3.000
                    Max.   :40.00                      Max.   :3.000
       n          distractor_no         rt              accuracy
 Min.   :   0.0   Min.   : 0.00   Min.   : 0.00217   Min.   :0.0000
 1st Qu.: 399.8   1st Qu.: 3.25   1st Qu.: 0.60274   1st Qu.:1.0000
 Median : 799.5   Median : 9.00   Median : 0.70336   Median :1.0000
 Mean   : 799.5   Mean   :12.00   Mean   : 0.78076   Mean   :0.9742
 3rd Qu.:1199.2   3rd Qu.:19.00   3rd Qu.: 0.85354   3rd Qu.:1.0000
 Max.   :1599.0   Max.   :31.00   Max.   :55.37298   Max.   :1.0000
```

```
# We have 40 participants, and each completed 1600 trials.
# Multiplying these together gives us the full length of our dataset (6400
0 rows):

40*1600

[1] 64000

# So we can conclude that all participants did complete all trials.
```

Answer 4:

```
d <- subset(d, select = -imageFileName)
```

Answer 5:

```
d$n <- d$n + 1
```

Answer 6:

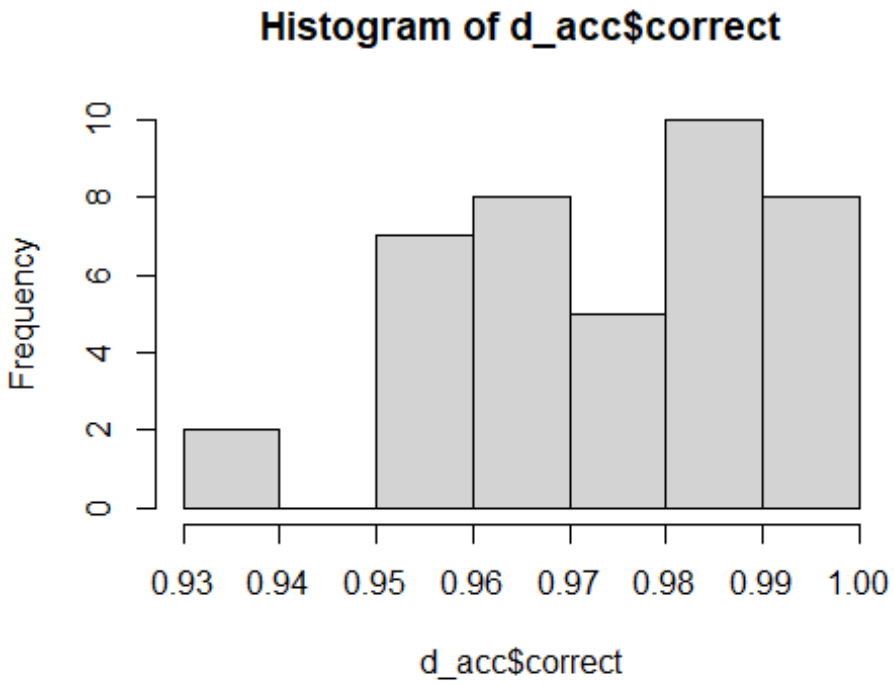
```
names(d) <- c("observer", "block", "feature", "trial", "nd", "rt", "correc
t")

# check that everything looks good
head(d)

  observer block feature trial nd        rt correct
1        4    2a       1     1  4 0.5825252       1
2        4    2c       2     2  4 0.5564872       1
3        4    2a       1     3  9 0.5042450       1
4        4    1a       1     4  4 0.7859964       1
5        4    1a       3     5 31 0.6468332       1
6        4    2b       1     6  4 0.6718330       1
```

Answer 7:

```
d$log_nd <- log(d$nd + 1)
```

Answer 8:

```
d_acc <- aggregate(correct ~ observer, d, FUN = "mean")
hist(d_acc$correct)
```




```
# All participants had an accuracy over 90%.
```

As we can see from the average accuracy rates, this task was relatively 'easy' to get right. We expect that trials where the participant got the answer wrong are different in some way (perhaps they had a lapse of attention) and therefore we don't want to analyse them in the main models.

Answer 9:

```
d <- subset(d, correct == 1)
```

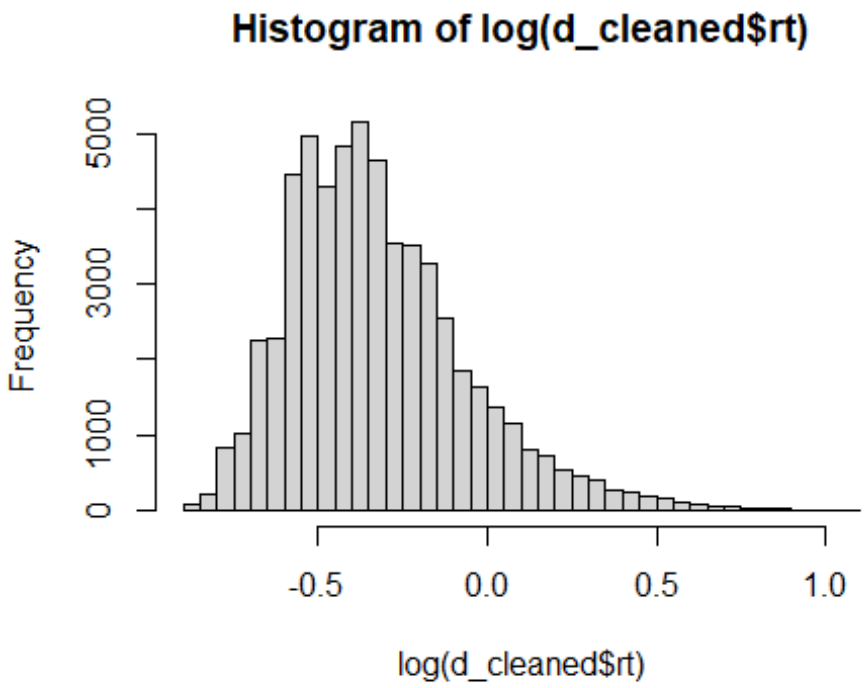
Answer 10:

```
d$log_rt <- log(d$rt)
d2 <- aggregate(log_rt ~ observer, d, FUN = "mean")

# group average
group_average <- mean(d2$log_rt)
group_sd <- sd(d2$log_rt)

min_log_rt <- group_average - 2*group_sd
max_log_rt <- group_average + 2*group_sd

subset(d2, log_rt < min_log_rt)

[1] observer log_rt
<0 rows> (or 0-length row.names)

subset(d2, log_rt > max_log_rt)
```

```
   observer      log_rt
19       19 -0.02269199
24       24 -0.03312403

# we should remove participants 19 and 24
d <- subset(d, !(observer %in% c(19, 24)))
```

Answer 11:

```
d_cleaned <- data.frame()

 for (obs in unique(d$observer)) {

  d_obs <- subset(d, observer == obs)
  limits98 <- quantile(d_obs$rt, c(0.01, 0.99))
  d_obs <- subset(d_obs, rt > limits98[1] & rt < limits98[2])

  d_cleaned <- rbind(d_cleaned, d_obs)
 }

hist(log(d_cleaned$rt), 50)
```




```
d <- d_cleaned
```

Answer 12:

```
# Replicate nD=0 trials
d0 <- subset(d, nd == 0)
summary(d0)

    observer        block              feature      trial              nd
 Min.   : 1.00   Length:3597        Min.   :0   Min.   :   1.0   Min.   :0
```

```
 1st Qu.:10.00   Class :character   1st Qu.:0   1st Qu.: 404.0   1st Qu.:0
 Median :21.00   Mode  :character   Median :0   Median : 804.0   Median :0
 Mean   :20.47                      Mean   :0   Mean   : 800.4   Mean   :0
 3rd Qu.:31.00                      3rd Qu.:0   3rd Qu.:1206.0   3rd Qu.:0
 Max.   :40.00                      Max.   :0   Max.   :1600.0   Max.   :0
       rt            correct      log_nd      log_rt
 Min.   :0.4195   Min.   :1   Min.   :0   Min.   :-0.8687
 1st Qu.:0.5550   1st Qu.:1   1st Qu.:0   1st Qu.:-0.5888
 Median :0.6372   Median :1   Median :0   Median :-0.4507
 Mean   :0.6748   Mean   :1   Mean   :0   Mean   :-0.4244
 3rd Qu.:0.7394   3rd Qu.:1   3rd Qu.:0   3rd Qu.:-0.3020
 Max.   :2.2059   Max.   :1   Max.   :0   Max.   : 0.7911

# The feature column is labelled as 0 (because there are no distractors in
these trials!)
```

Answer 13:

```
d <- subset(d, nd > 0)

d01 <- d0
d01$feature <- 1
d02 <- d0
d02$feature <- 2
d03 <- d0
d03$feature <- 3

d <- rbind(d, d01, d02, d03)
rm(d0, d01, d02, d03)
```

Answer 14:

```
# tidy up experiment and feature labels
d$feature = factor(paste(d$block, d$feature, sep = "-"))

levels(d$feature) <- c("orange", "pink", "purple",
                       "circle", "diamond", "triangle",
                       "pink circle", "orange diamond", "purple triangle",
                       "orange circle", "purple diamond", "pink triangle",
                       "purple circle","pink diamond",  "orange triangle")
```

Answer 15:

```
d1 <- subset(d, block %in% c("1a", "1b"))
```

Answer 16:

```
# first, aggregate by person
d1agg <- aggregate(log_rt ~ feature + log_nd + observer,
                   data = d1, FUN = mean)

head(d1agg)

   feature log_nd observer     log_rt
1   orange      0        1 -0.4620293
```

```
2     pink      0        1 -0.4620293
3   purple      0        1 -0.4620293
4   circle      0        1 -0.3451511
5  diamond      0        1 -0.3451511
6 triangle      0        1 -0.3451511
```

Answer 17:

```
# now over people
d1agg2 <- aggregate(log_rt ~ feature + log_nd,
                    data = d1agg, FUN = mean)
```

Answer 18:

```
plot(d1agg2$log_nd, d1agg2$log_rt,
     xlab = "log(num. distractors + 1)",
     ylab = "log(response time)",
     type = "n")

colours <- c("orange", "pink", "purple", "grey20", "grey40", "grey60")
shapes <- c(3, 3, 3, 1, 5, 2)

conditions <- unique(d1agg2$feature)
n_conditions <- length(conditions)

for (i in 1:n_conditions) {

  dc <- subset(d1agg2, feature == conditions[i])

  lines(dc$log_nd, dc$log_rt,
        col = colours[i])
  points(dc$log_nd, dc$log_rt,
         col = colours[i], pch = shapes[i])

}
```

Answer 19:

```
# log_rt ~ log_nd * feature
```

Answer 20:

```
# (1 | observer)
```

Answer 21:

```
library(lme4)
```

```
Loading required package: Matrix
```

```
m1a <- lmer(log_rt ~ log_nd * feature + (1|observer),
            data = d1)

summary(m1a, corr = F)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: log_rt ~ log_nd * feature + (1 | observer)
   Data: d1

REML criterion at convergence: -2572.6

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.9004 -0.6900 -0.1324  0.5419  4.7648

Random effects:
 Groups   Name        Variance Std.Dev.
 observer (Intercept) 0.01638  0.1280
 Residual             0.05241  0.2289
Number of obs: 25971, groups:  observer, 38

Fixed effects:
                        Estimate Std. Error t value
(Intercept)            -0.395812   0.021705 -18.236
log_nd                  0.068136   0.002869  23.749
featurepink            -0.011270   0.008944  -1.260
featurepurple          -0.023009   0.008924  -2.578
featurecircle           0.014064   0.008899   1.580
featurediamond         -0.007415   0.008892  -0.834
featuretriangle         0.051747   0.008933   5.793
log_nd:featurepink     -0.047060   0.004040 -11.647
log_nd:featurepurple   -0.053302   0.004041 -13.189
log_nd:featurecircle    0.013460   0.004037   3.334
log_nd:featurediamond   0.021853   0.004042   5.406
log_nd:featuretriangle  0.031788   0.004078   7.794
```

Answer 22:

```
m1b <- lmer(log_rt ~  0 + feature + log_nd:feature + (1|observer),
            data = d1)

summary(m1b, corr = F)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: log_rt ~ 0 + feature + log_nd:feature + (1 | observer)
   Data: d1

REML criterion at convergence: -2572.6

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.9004 -0.6900 -0.1324  0.5419  4.7648

Random effects:
 Groups   Name        Variance Std.Dev.
 observer (Intercept) 0.01638  0.1280
 Residual             0.05241  0.2289
Number of obs: 25971, groups:  observer, 38
```

```
Fixed effects:
                         Estimate Std. Error t value
featureorange           -0.395812   0.021705 -18.236
featurepink             -0.407082   0.021707 -18.754
featurepurple           -0.418821   0.021699 -19.302
featurecircle           -0.381748   0.021688 -17.602
featurediamond          -0.403227   0.021685 -18.594
featuretriangle         -0.344065   0.021702 -15.854
featureorange:log_nd     0.068136   0.002869  23.749
featurepink:log_nd       0.021076   0.002845   7.408
featurepurple:log_nd     0.014834   0.002847   5.211
featurecircle:log_nd     0.081596   0.002840  28.732
featurediamond:log_nd    0.089989   0.002847  31.607
featuretriangle:log_nd   0.099924   0.002899  34.473
```

Answer 23:

```
# extract slopes
b <- summary(m1b)$coefficients[7:12, 1]
slopes1 <- data.frame(condition = names(b),
                      b = as.numeric(b))

slopes1
```

```
               condition          b
1   featureorange:log_nd 0.06813613
2     featurepink:log_nd 0.02107611
3   featurepurple:log_nd 0.01483414
4   featurecircle:log_nd 0.08159632
5  featurediamond:log_nd 0.08998888
6 featuretriangle:log_nd 0.09992413
```

Answer 24:

```
d2 <- subset(d, block %in% c("2a", "2b", "2c"))
```

Answer 25:

```
m2 <- lmer(log_rt ~  0 + feature + log_nd:feature + (1|observer),
            data = d2)

summary(m2, corr = F)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: log_rt ~ 0 + feature + log_nd:feature + (1 | observer)
   Data: d2

REML criterion at convergence: -14873.4

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.7236 -0.6699 -0.1332  0.4927  6.0738

Random effects:
 Groups   Name        Variance Std.Dev.
```

```
 observer (Intercept) 0.01492  0.1222
 Residual             0.03968  0.1992
Number of obs: 39222, groups:  observer, 38

Fixed effects:
                              Estimate Std. Error t value
featurepink circle           -0.413935   0.020571 -20.122
featureorange diamond        -0.430879   0.020567 -20.950
featurepurple triangle       -0.434549   0.020569 -21.126
featureorange circle         -0.423114   0.020560 -20.580
featurepurple diamond        -0.426093   0.020567 -20.718
featurepink triangle         -0.423413   0.020565 -20.589
featurepurple circle         -0.419142   0.020560 -20.386
featurepink diamond          -0.437410   0.020560 -21.275
featureorange triangle       -0.415545   0.020558 -20.213
featurepink circle:log_nd     0.009740   0.002480   3.927
featureorange diamond:log_nd  0.056719   0.002474  22.924
featurepurple triangle:log_nd 0.018239   0.002481   7.353
featureorange circle:log_nd   0.043320   0.002465  17.571
featurepurple diamond:log_nd  0.011109   0.002479   4.481
featurepink triangle:log_nd   0.026564   0.002479  10.716
featurepurple circle:log_nd   0.005544   0.002471   2.244
featurepink diamond:log_nd    0.022037   0.002468   8.929
featureorange triangle:log_nd 0.069717   0.002469  28.235
```

Answer 26:

```
b <- summary(m2)$coefficients[10:18, 1]
slopes2 <- data.frame(condition = names(b),
                      b = as.numeric(b))

slopes2

                      condition           b
1     featurepink circle:log_nd 0.009739908
2  featureorange diamond:log_nd 0.056718645
3 featurepurple triangle:log_nd 0.018239356
4   featureorange circle:log_nd 0.043320023
5  featurepurple diamond:log_nd 0.011109424
6   featurepink triangle:log_nd 0.026563836
7   featurepurple circle:log_nd 0.005543993
8    featurepink diamond:log_nd 0.022036582
9 featureorange triangle:log_nd 0.069716901
```

The next part is slightly fiddly: we need to first change the `slopes2` dataset so that there are two extra columns for `colour` and `shape` so that we know which colour and shape was in each double feature combination. Run the code below to see how this works.

```
library(stringr) # this library allows us to manipulate strings

slopes2$condition <- str_remove_all(slopes2$condition, "feature|:log_nd")

# break up condition
slopes2$colour <- str_extract(slopes2$condition, "pink|orange|purple")
```

```
slopes2$shape <- str_extract(slopes2$condition, "circle|diamond|triangle")

slopes2
```

```
        condition           b colour    shape
1     pink circle 0.009739908   pink   circle
2  orange diamond 0.056718645 orange  diamond
3 purple triangle 0.018239356 purple triangle
4   orange circle 0.043320023 orange   circle
5  purple diamond 0.011109424 purple  diamond
6   pink triangle 0.026563836   pink triangle
7   purple circle 0.005543993 purple   circle
8    pink diamond 0.022036582   pink  diamond
9 orange triangle 0.069716901 orange triangle
```

Next, we are going to make two new columns in our `slopes2` dataset: one called `bc` which will contain the slope for that specific *colour* in Experiment 1, and one called `bs` which will contain the slope for that specific *shape* in Experiment 1. Again, run the code below, and check you understand what it is doing.

```
# merge - HARD!!!!
slopes2$bc <- NA
slopes2$bs <- NA

# tidy up slopes1 labelling in the same way as we did in the previous sect
ion for slopes2
slopes1$condition <- str_remove_all(slopes1$condition, "feature|:log_nd")


for (i in 1:nrow(slopes2)) {

  slopes2$bc[i] = slopes1$b[slopes1$condition == slopes2$colour[i]]
  slopes2$bs[i] = slopes1$b[slopes1$condition == slopes2$shape[i]]
}

slopes2
```

```
        condition           b colour    shape         bc         bs
1     pink circle 0.009739908   pink   circle 0.02107611 0.08159632
2  orange diamond 0.056718645 orange  diamond 0.06813613 0.08998888
3 purple triangle 0.018239356 purple triangle 0.01483414 0.09992413
4   orange circle 0.043320023 orange   circle 0.06813613 0.08159632
5  purple diamond 0.011109424 purple  diamond 0.01483414 0.08998888
6   pink triangle 0.026563836   pink triangle 0.02107611 0.09992413
7   purple circle 0.005543993 purple   circle 0.01483414 0.08159632
8    pink diamond 0.022036582   pink  diamond 0.02107611 0.08998888
9 orange triangle 0.069716901 orange triangle 0.06813613 0.09992413
```

To predict the double feature slopes from the single feature slopes, we are going to use an *orthogonal contrast* model, which we can define in the following way:

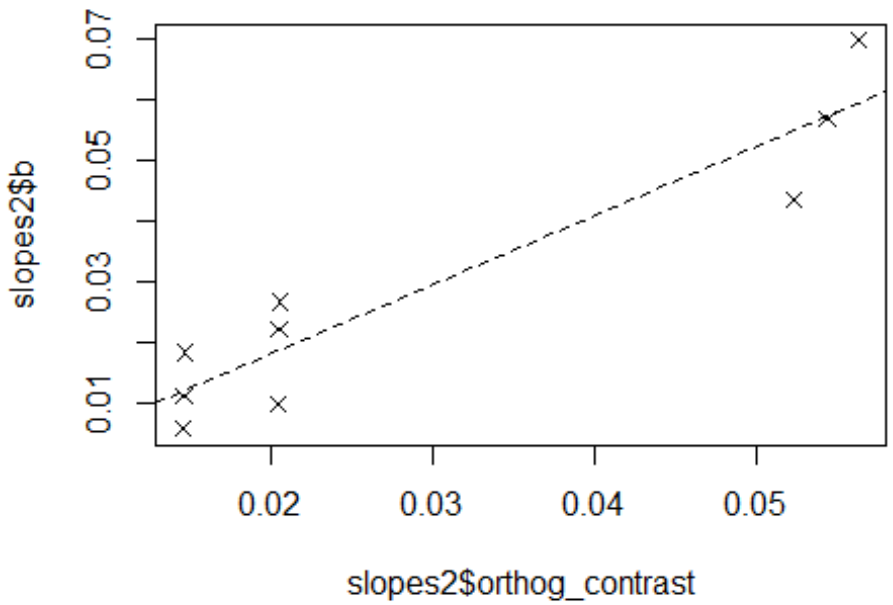
$$\frac{1}{\sqrt{((1/b_c)^2 + (1/b_s)^2)}}$$

Answer 27:

```
# orthogonal contrast

slopes2$orthog_contrast <- 1 / sqrt((1/slopes2$bc)^2 + (1/slopes2$bs)^2)
```

Answer 28:

```
# plot

plot(slopes2$orthog_contrast, slopes2$b, pch = 4)

# check correlation
cor.test(slopes2$b, slopes2$orthog_contrast)


    Pearson's product-moment correlation

data:  slopes2$b and slopes2$orthog_contrast
t = 7.3022, df = 7, p-value = 0.0001625
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.7350027 0.9876328
sample estimates:
      cor
0.9401888

# compute best fit line
mp <- summary(lm(b ~ orthog_contrast, slopes2))$coefficients

abline(a = mp[1,1], b = mp[2,1], lty = 2)
```



```
# There is a high correlation between these two measures.
# This suggests that if we know the single feature slope values,
#applying the orthogonal contrast model to them allows us to make
# a good prediction of the double feature slope values.
```