Modality effects in a signalling game

Intro

This script uses data compiled by *analyseData.R*.

Load libraries

```
library(lme4)
library(sjPlot)
library(ggplot2)
library(lattice)
library(influence.ME)
```

Load data

```
d = read.csv("../../data/FinalSignalData.csv")
```

We don't need info on every signal in each turn, just the trial time. Keep only 1st signal in each trial.

```
d = d[!duplicated(d$trialString),]
```

Descriptive stats

Here is a graph showing the distribution of trial lengths by conditions:

The distribution of trial times is very skewed:

```
hist(d$trialLength)
```

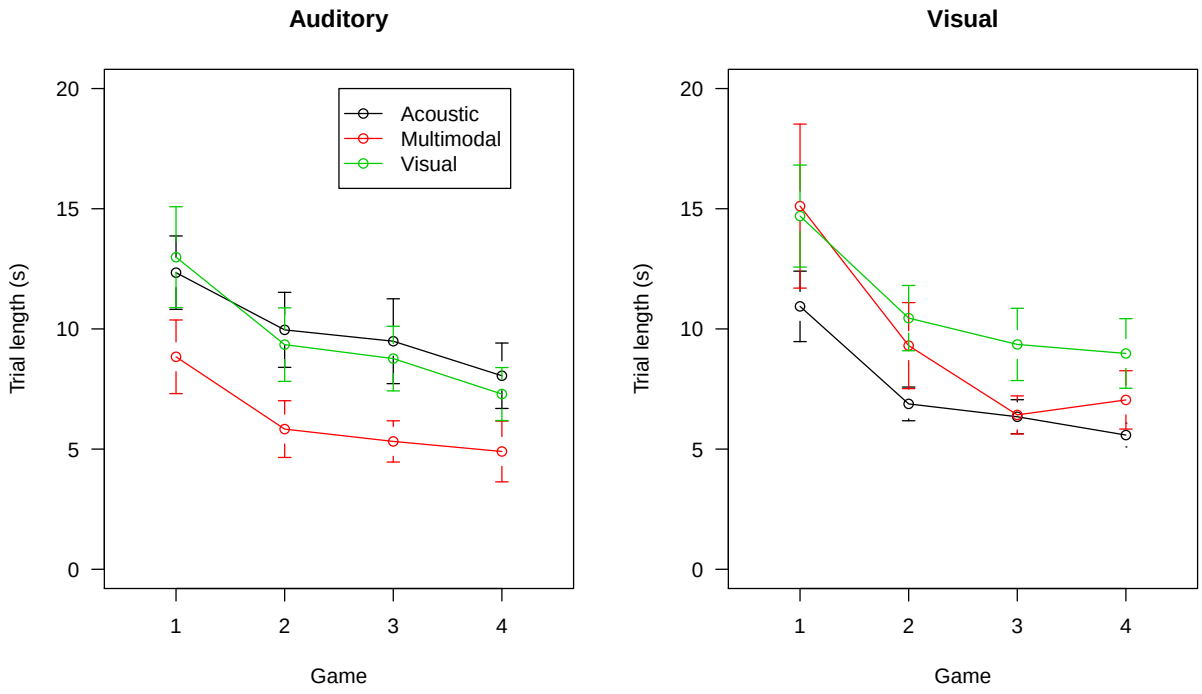
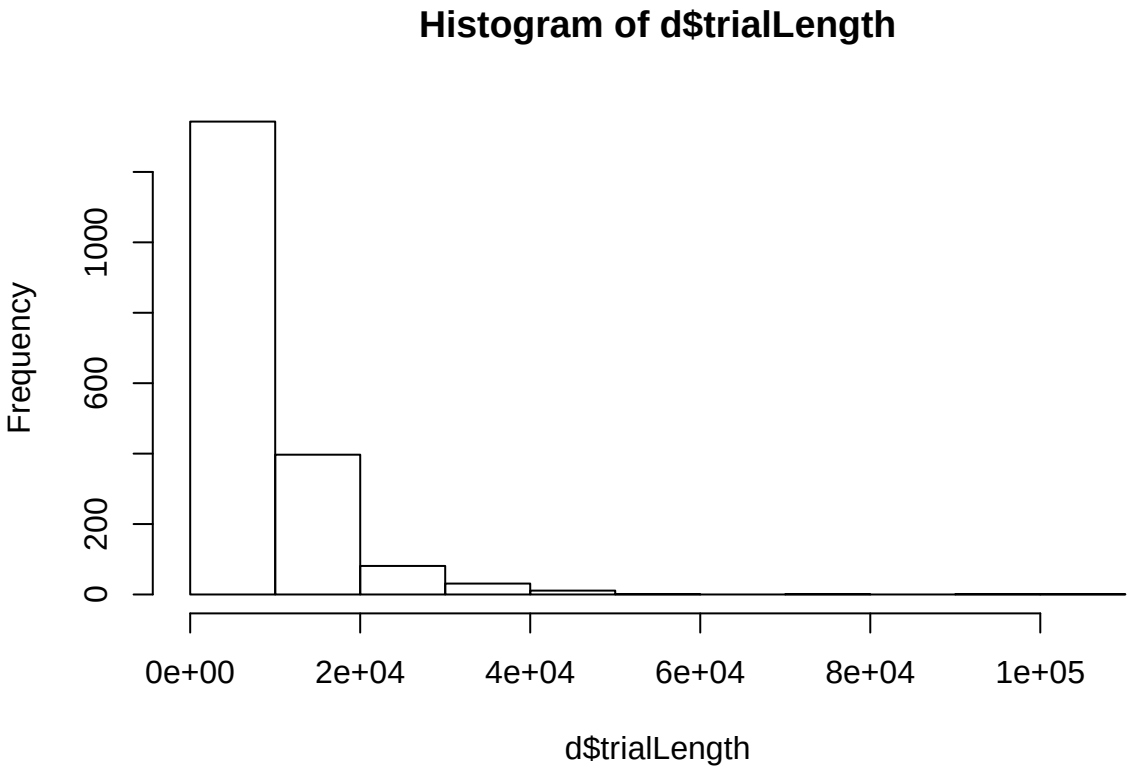
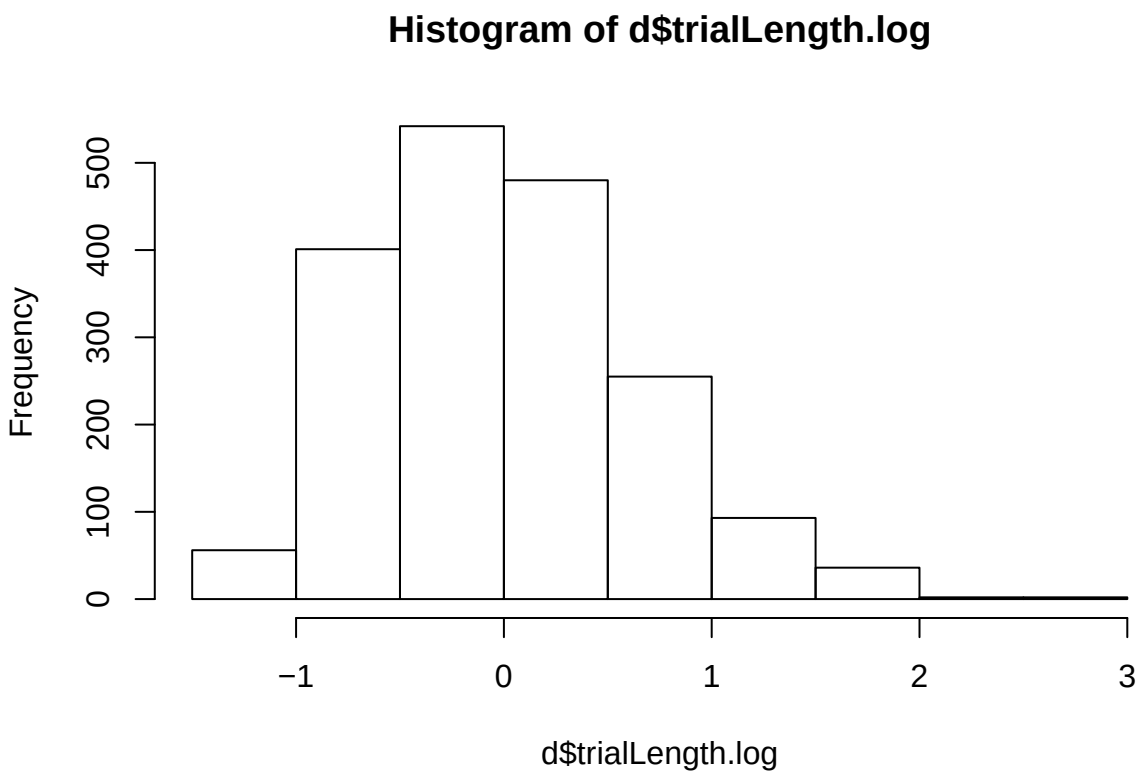


Figure 1: The efficiency of trials in different conditions



So we transform it using a log transform, then center the data.

```
d$trialLength.log = log(d$trialLength)
meanLogTrialLength = mean(d$trialLength.log)
d$trialLength.log = d$trialLength.log - meanLogTrialLength
hist(d$trialLength.log)
```



Make a variable to represent proportion of games played:

```
# Make a variable that represents the number of trials played
d$trialTotal = d$trial + (d$game * (max(d$trial)+1))
# Convert to proportion of games played, so that estimates reflect change per game.
d$trialTotal = d$trialTotal / 16
# Center the trialTotal variable so intercept reflects after the first game
d$trialTotal = d$trialTotal - 1
```

Make a variable for which stimuli the players experienced first.

```
firstBlock = tapply(as.character(d$condition),d$dyadNumber,head,n=1)
d$firstBlock = as.factor(firstBlock[match(d$dyadNumber,names(firstBlock))])
```

Reorder some levels so that the intercept reflects the most frequent condition.

```
d$incorrect = !d$correct
```

Mixed models

Make a series of models with random effects for dyad, director (nested within dyad) and item.

Not all random slopes are appropriate. For example, items are used in only one stimulus condition, so a random slope for condition by item is not appropriate. Similarly, each dyad only plays in one modality condition.

It's reasonable to have a random slope for trial by dyad, but this caused unreliable model convergence, so is not included.

The final random slopes were for condition and incorrectness by dyad/player, and modality condition by item.

```
# No fixed effects
m0 = lmer(trialLength.log ~ 1 +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add modality condition
modality = lmer(trialLength.log ~ 1 + modalityCondition +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add stimulus condition
cond = lmer(trialLength.log ~ 1 + modalityCondition + condition +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add trial total
game = lmer(trialLength.log ~ 1 + modalityCondition + condition + trialTotal +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
```

```
# Add interaction between condition and stimulus condition
modXcond = lmer(trialLength.log ~ 1 + modalityCondition * condition + trialTotal +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add interaction between condition and trial
conXgame = lmer(trialLength.log ~ 1 + (modalityCondition * condition) + trialTotal +
            (trialTotal:condition) +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add interaction between modality and trial
moXcoXga = lmer(trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add whether the response was incorrect
incor = lmer(trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
            incorrect +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
```

```
# Add the interaction between modality and incorrectness
moXincor = lmer(trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
            incorrect + (modalityCondition:incorrect) +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add the quadratic effect of trial
gamQuad = lmer(trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
            I(trialTotal^2) +
            incorrect + (modalityCondition:incorrect) +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add interaction between quadratic effect of trial and modality
modXgamQ = lmer(trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
            I(trialTotal^2) +
            incorrect + (modalityCondition:incorrect) +
            (modalityCondition:I(trialTotal^2)) +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add block order
block = lmer(trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
            I(trialTotal^2) +
            incorrect + (modalityCondition:incorrect) +
            (modalityCondition:I(trialTotal^2)) +
            firstBlock +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
# Add interaction between block order and modality
blocXmod = lmer(trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
            I(trialTotal^2) +
            incorrect + (modalityCondition:incorrect) +
            (modalityCondition:I(trialTotal^2)) +
            firstBlock * modalityCondition +
            (1 + condition + incorrect |dyadNumber/playerId) +
            (1 + modalityCondition|itemId),
          data=d)
```

Results

Compare the fit of the models:

```
anova(m0,modality,cond,game,modXcond,conXgame,
      moXcoXga,incor,moXincor,gamQuad,modXgamQ,
      block, blocXmod)
```

```
## refitting model(s) with ML (instead of REML)
```

```
## Data: d
## Models:
## m0: trialLength.log ~ 1 + (1 + condition + incorrect | dyadNumber/playerId) +
## m0:     (1 + modalityCondition | itemId)
## modality: trialLength.log ~ 1 + modalityCondition + (1 + condition + incorrect |
## modality:     dyadNumber/playerId) + (1 + modalityCondition | itemId)
## cond: trialLength.log ~ 1 + modalityCondition + condition + (1 + condition +
## cond:     incorrect | dyadNumber/playerId) + (1 + modalityCondition |
## cond:     itemId)
## game: trialLength.log ~ 1 + modalityCondition + condition + trialTotal +
## game:     (1 + condition + incorrect | dyadNumber/playerId) + (1 +
## game:     modalityCondition | itemId)
## modXcond: trialLength.log ~ 1 + modalityCondition * condition + trialTotal +
## modXcond:     (1 + condition + incorrect | dyadNumber/playerId) + (1 +
## modXcond:     modalityCondition | itemId)
## conXgame: trialLength.log ~ 1 + (modalityCondition * condition) + trialTotal +
## conXgame:     (trialTotal:condition) + (1 + condition + incorrect | dyadNumber/playerId) +
## conXgame:     (1 + modalityCondition | itemId)
## moXcoXga: trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
## moXcoXga:     (1 + condition + incorrect | dyadNumber/playerId) + (1 +
## moXcoXga:     modalityCondition | itemId)
## incor: trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
## incor:     incorrect + (1 + condition + incorrect | dyadNumber/playerId) +
## incor:     (1 + modalityCondition | itemId)
## moXincor: trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
## moXincor:     incorrect + (modalityCondition:incorrect) + (1 + condition +
## moXincor:     incorrect | dyadNumber/playerId) + (1 + modalityCondition |
## moXincor:     itemId)
## gamQuad: trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
## gamQuad:     I(trialTotal^2) + incorrect + (modalityCondition:incorrect) +
## gamQuad:     (1 + condition + incorrect | dyadNumber/playerId) + (1 +
## gamQuad:     modalityCondition | itemId)
## modXgamQ: trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
## modXgamQ:     I(trialTotal^2) + incorrect + (modalityCondition:incorrect) +
## modXgamQ:     (modalityCondition:I(trialTotal^2)) + (1 + condition + incorrect |
## modXgamQ:     dyadNumber/playerId) + (1 + modalityCondition | itemId)
## block: trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
## block:     I(trialTotal^2) + incorrect + (modalityCondition:incorrect) +
## block:     (modalityCondition:I(trialTotal^2)) + firstBlock + (1 + condition +
## block:     incorrect | dyadNumber/playerId) + (1 + modalityCondition |
## block:     itemId)
## blocXmod: trialLength.log ~ 1 + modalityCondition * condition * trialTotal +
## blocXmod:     I(trialTotal^2) + incorrect + (modalityCondition:incorrect) +
## blocXmod:     (modalityCondition:I(trialTotal^2)) + firstBlock * modalityCondition +
## blocXmod:     (1 + condition + incorrect | dyadNumber/playerId) + (1 +
```

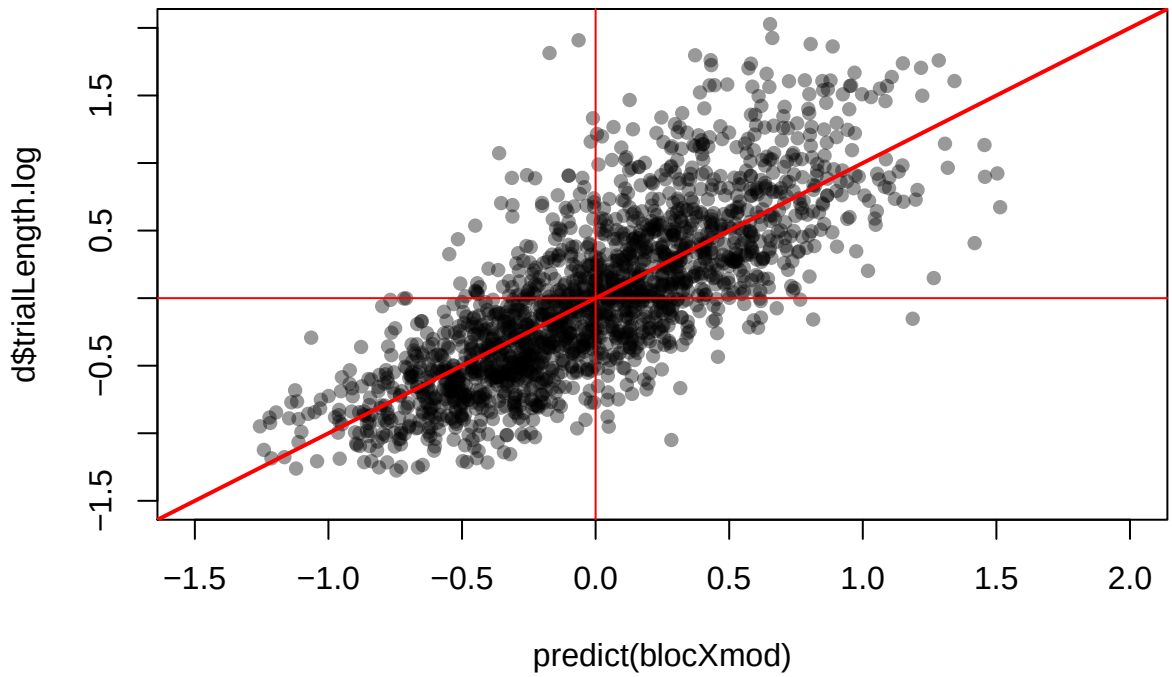
```
## blocXmod:     modalityCondition | itemId)
##           Df    AIC    BIC  logLik deviance    Chisq Chi Df Pr(>Chisq)
## m0        20 2743.3 2853.9 -1351.7   2703.3
## modality  22 2745.1 2866.8 -1350.6   2701.1   2.1746      2  0.3371278
## cond      23 2746.4 2873.6 -1350.2   2700.4   0.7327      1  0.3920072
## game      24 2366.7 2499.4 -1159.3   2318.7 381.7015      1  < 2.2e-16 ***
## modXcond  26 2357.6 2501.4 -1152.8   2305.6  13.0703      2  0.0014515 **
## conXgame  27 2359.3 2508.7 -1152.7   2305.3   0.3118      1  0.5765803
## moXcoXga  31 2359.4 2530.9 -1148.7   2297.4   7.8964      4  0.0954477 .
## incor     32 2347.5 2524.5 -1141.8   2283.5  13.8998      1  0.0001928 ***
## moXincor  34 2346.8 2534.9 -1139.4   2278.8   4.7073      2  0.0950216 .
## gamQuad   35 2273.9 2467.5 -1102.0   2203.9  74.9002      1  < 2.2e-16 ***
## modXgamQ  37 2270.8 2475.5 -1098.4   2196.8   7.1049      2  0.0286546 *
## block     38 2272.4 2482.6 -1098.2   2196.4   0.4365      1  0.5088031
## blocXmod  40 2275.6 2496.8 -1097.8   2195.6   0.7977      2  0.6710839
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Pick final model for estimates:

```
finalModel = blocXmod
```

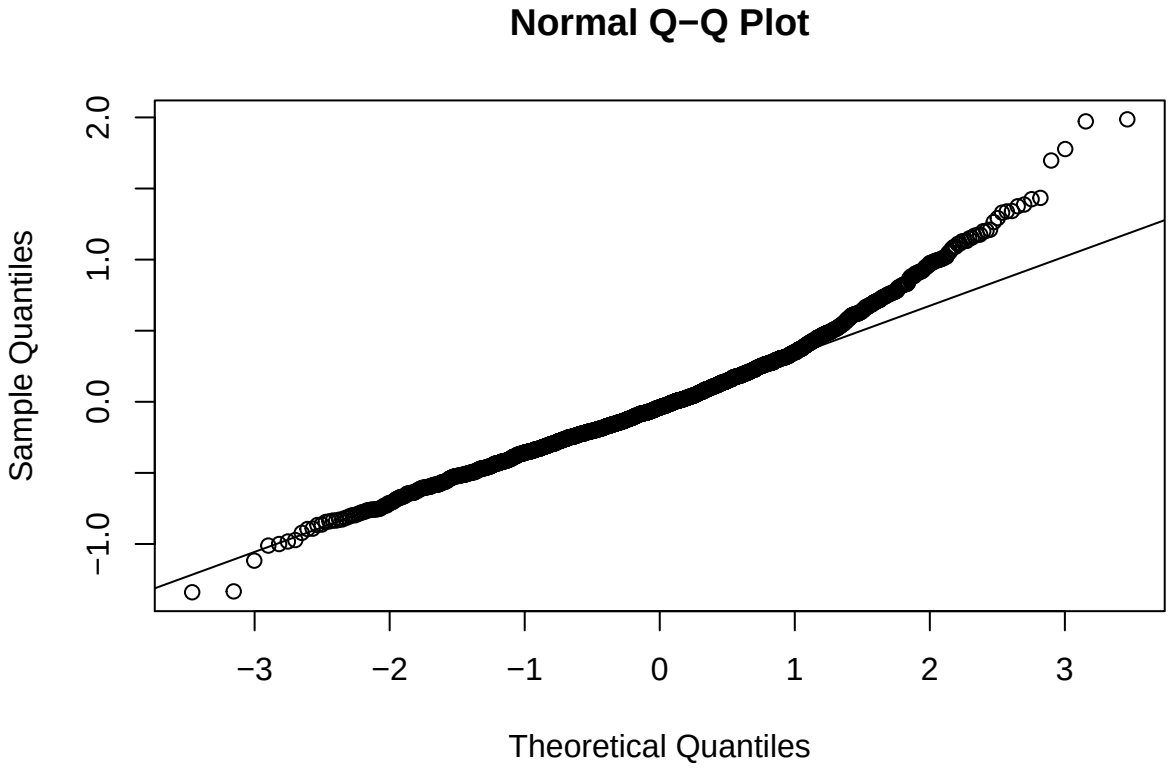
Check model predictions. The model predictions are in the right range and direction, fitting linear quite well:

```
plot(predict(blocXmod),d$trialLength.log, pch=16, col=rgb(0,0,0,0.4),
     ylim=c(-1.5,2),xlim=c(-1.5,2))
abline(a=0,b=1, col=2, lwd=2)
abline(h=0, col=2)
abline(v=0, col=2)
```



The residuals are ok, though it tends to do worse at higher values. This is expected from using the log scale.

```
qqnorm(resid(blocXmod))
qqline(resid(blocXmod))
```



Plot the fixed effects

Relabel the effects:

```
feLabels = matrix(c(
"(Intercept)"              ,"Intercept"      ,
"modalityConditionvisual" ,"Visual modality",
"modalityConditionvocal"  , "Acoustic modality",
"conditionVisual" , "Visual stimuli",
"trialTotal"              , "Game",
"modalityConditionvisual:conditionVisual"  , "Visual modality:Visual stimuli",
"modalityConditionvocal:conditionVisual" , "Acoustic modality:Visual stimuli",
"modalityConditionvisual:trialTotal"    , "Visual modality:Game",
"modalityConditionvocal:trialTotal"     , "Acoustic modality:Game",
"conditionVisual:trialTotal"             , "Visual stimuli:Game",
"modalityConditionvisual:conditionVisual:trialTotal", "Visual modality:Visual stimuli:Game",
"modalityConditionvocal:conditionVisual:trialTotal", "Acoustic modality:Visual stimuli:Game",
"incorrectTRUE","Inorrect",
"modalityConditionvisual:incorrectTRUE","Visual modality:Incorrect",
"modalityConditionvocal:incorrectTRUE","Acoustic modality:Incorrect",
"modalityConditionvisual:I(trialTotal^2)", "Visual modality:Game^2",
"modalityConditionvocal:I(trialTotal^2)", "Acoustic modality:Game^2",
"I(trialTotal^2)","Game^2"
), ncol=2, byrow = T)

feLabels2 = as.vector(feLabels[match(names(fixef(finalModel)),feLabels[,1]),2])
```

Plot the strength of the fixed effects:

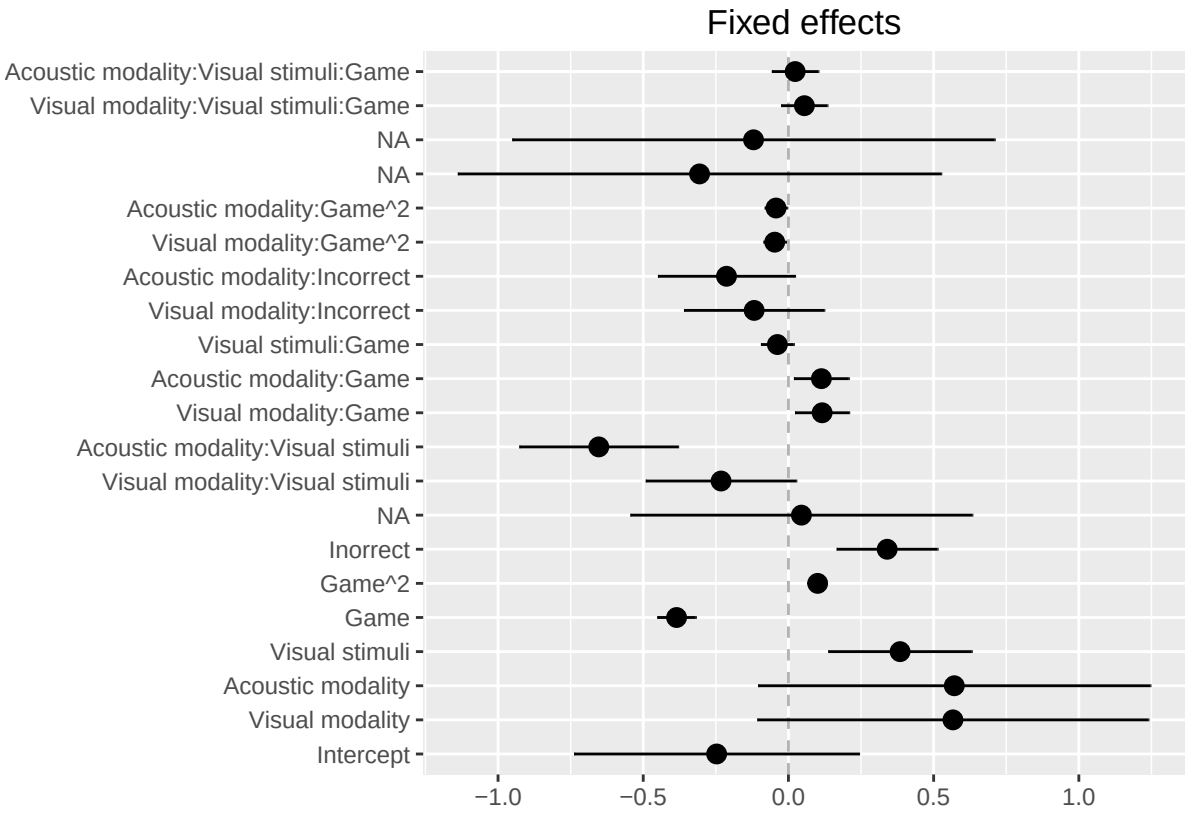
```
x = sjp.lmer(finalModel, 'fe',
          show.intercept = T,
          sort.est=NULL,
          axis.labels = feLabels2[2:length(feLabels2)],
          xlab="Trial time (ms)",
          geom.colors = c(1,1),
          show.p=F,
          show.values = F,
          string.interc="Intercept")
```

```
## Warning: replacing previous import 'lme4::sigma' by 'stats::sigma' when
## loading 'pbkrtest'
```

```
## Computing p-values via Kenward-Roger approximation. Use `p.kr = FALSE` if computation takes too long
```

```
## Warning in deviance.merMod(object, ...): deviance() is deprecated for REML
## fits; use REMLcrit for the REML criterion or deviance(.,REML=FALSE) for
## deviance calculated at the REML fit
```

```
## Warning: Deprecated, use tibble::rownames_to_column() instead.
```



Attempt plot with axes in milliseconds.

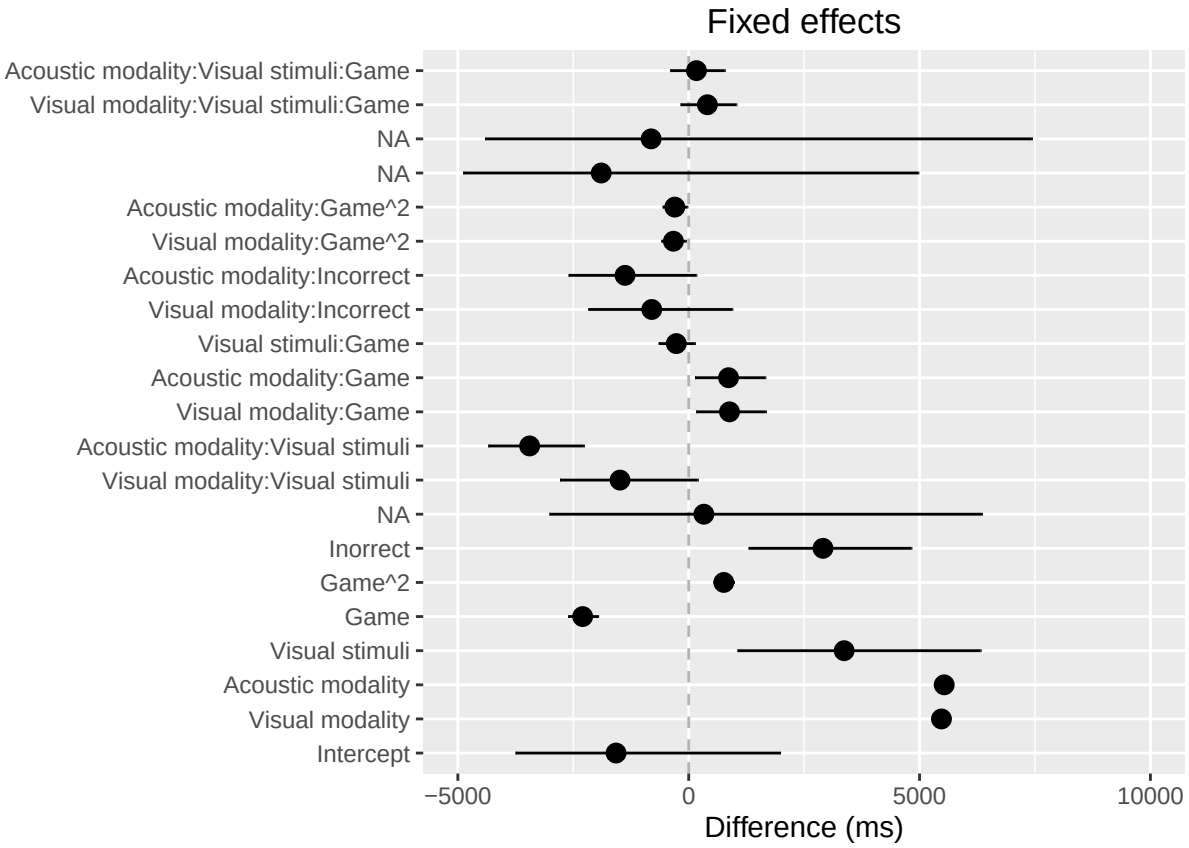
```
convertEst = function(X){
  exp(meanLogTrialLength+X) - exp(meanLogTrialLength)
}

x$plot.list[[1]]$data$estimate =convertEst(x$plot.list[[1]]$data$estimate)
x$plot.list[[1]]$data$conf.low = convertEst(x$plot.list[[1]]$data$conf.low)
x$plot.list[[1]]$data$conf.high =  convertEst(x$plot.list[[1]]$data$conf.high)
```

```
x$plot.list[[1]] +
  scale_y_continuous(limits=c(-5000,10000), name="Difference (ms)") +
  scale_x_discrete(labels=feLabels2)
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which
## will replace the existing scale.
```

```
## Warning: Removed 2 rows containing missing values (geom_errorbar).
```

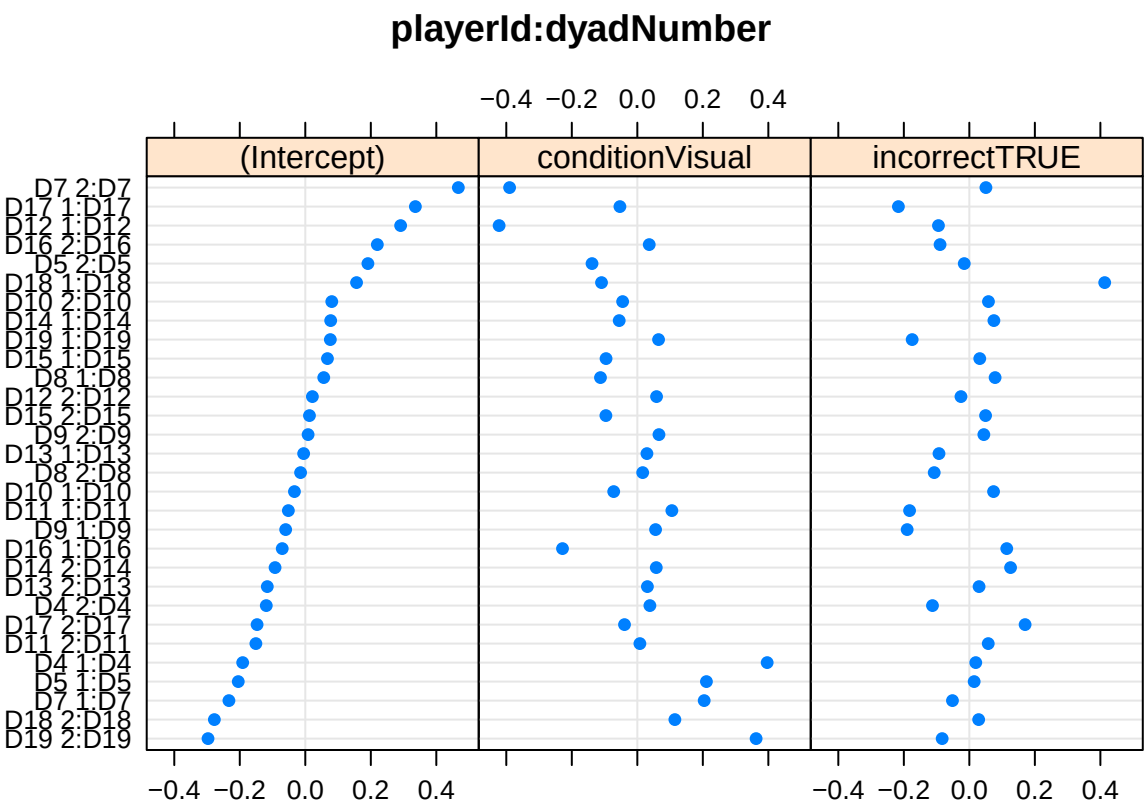


Random effects

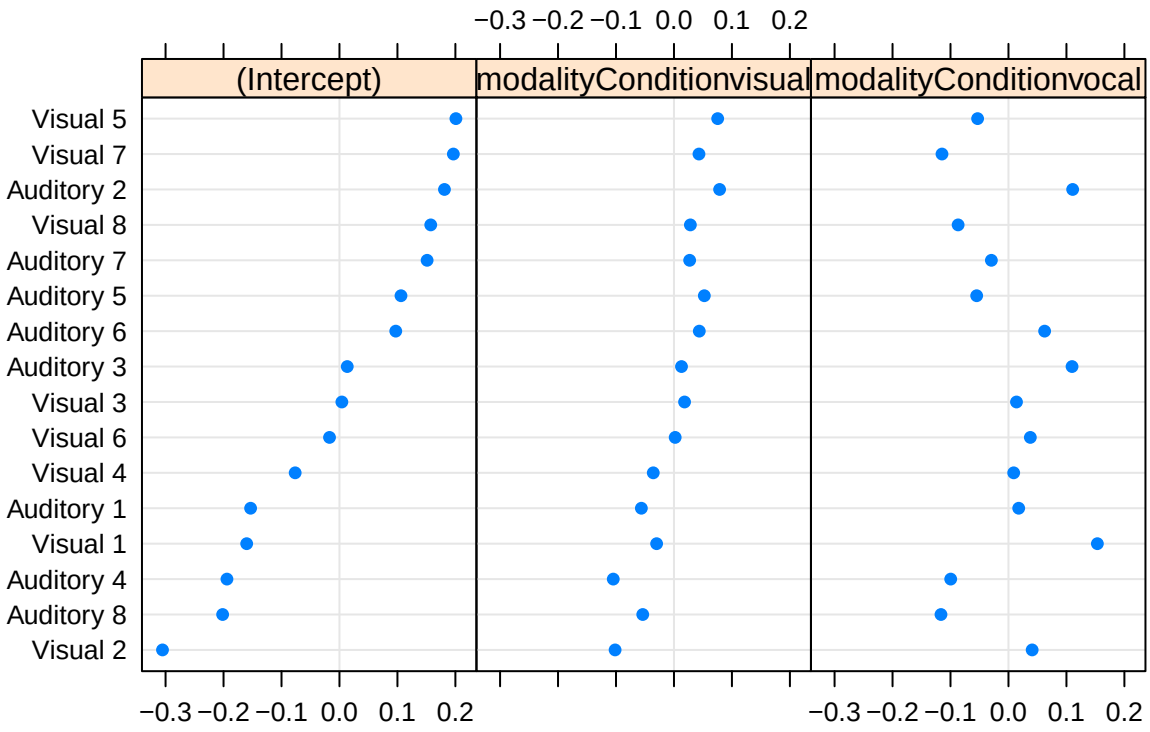
There is a reasonable amount of variaition in the random effects, suggesting that dyads and players differ. This justifies the use of mixed effects modelling.

```
dotplot(ranef(finalModel))
```

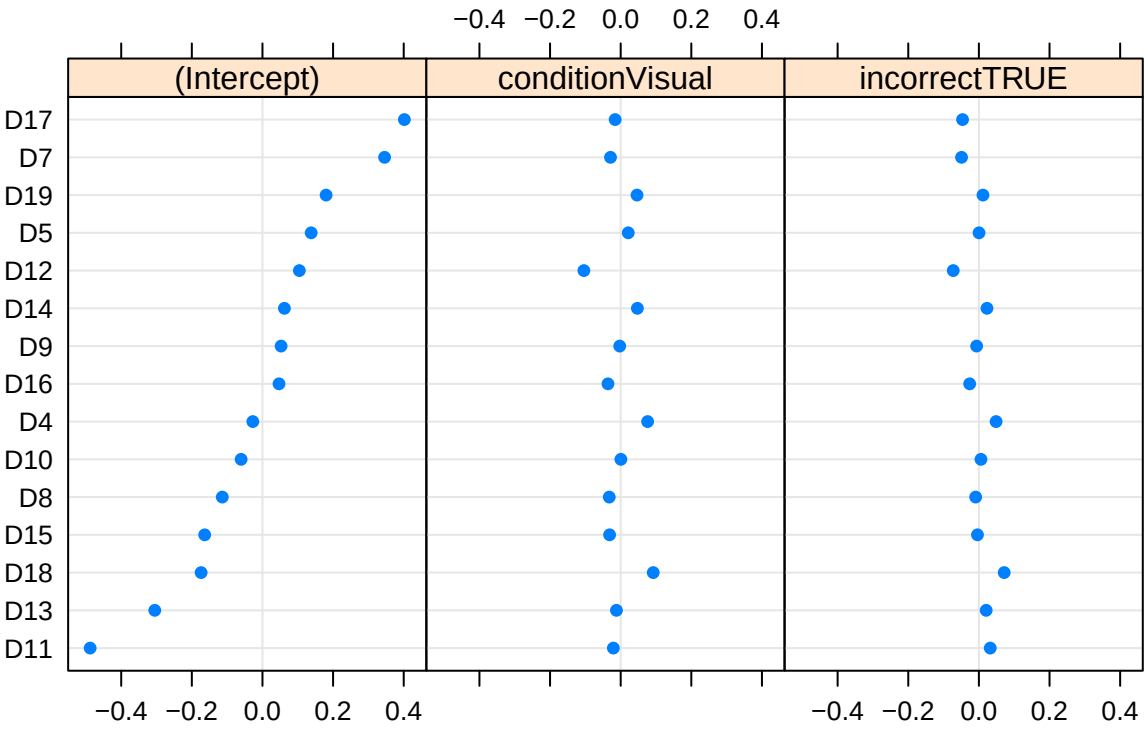
```
## $`playerId:dyadNumber`
```




```
##
## $itemId
```


itemId
(Intercept)
modalityConditionvisual
modalityConditionvocal
Visual 5
Visual 7
Auditory 2
Visual 8
Auditory 7
Auditory 5
Auditory 6
Auditory 3
Visual 3
Visual 6
Visual 4
Auditory 1
Visual 1
Auditory 4
Auditory 8
Visual 2
−0.3 −0.2 −0.1 0.0 0.1 0.2

```
##
## $dyadNumber
```


dyadNumber
(Intercept)
conditionVisual
incorrectTRUE
D17
D7
D19
D5
D12
D14
D9
D16
D4
D10
D8
D15
D18
D13
D11
−0.4 −0.2 0.0 0.2 0.4