seq_analysis

Avery Katko

July 28, 2014

```
rm(list=ls())
library(ggplot2)
library(reshape2)
library(plyr)
library(bootstrap)
#source("useful.R")
d <- read.csv("pragmods_seq.results.tsv",sep="\t")
#head(d)
```

Exclude participants that either failed manipulation check or were rejected.

```
exclude <- d$assignmentstatus == "Rejected" |
  d$Answer.name_check_correct == "\"FALSE\""
sum(exclude)
```

```
## [1] 4
```

```
mean(exclude)
```

```
## [1] 0.04
```

```
d <- subset(d, exclude == FALSE)

d$Answer.choice_correct_1 <- factor(as.logical(d$Answer.choice_correct_1))
d$Answer.choice_correct_2 <- factor(as.logical(d$Answer.choice_correct_2))
d$Answer.choice_correct_3 <- factor(as.logical(d$Answer.choice_correct_3))
```

Restructure data to have trial variable. (working around mturk data submission limitations; can't safely submit arrays, so we have to use separate variables for each trial)

```
d$level_1 <- factor(ifelse(d$Answer.sequence_condition == "\"1w0w1b\"",1,0))
d$level_2 <- factor(ifelse(d$Answer.sequence_condition == "\"1w0w1b\"",0,1))
d$level_3 <- factor(rep(1,nrow(d)))

trial.df <- function(tn){
  df <- data.frame(id = d$workerid,
                   seqCond = d$Answer.sequence_condition,
                   trial = factor(rep(tn,nrow(d))),
                   level = d[sprintf("level_%d",tn)],
                   item = d[sprintf("Answer.item_%d",tn)],
                   targetProp = d[sprintf("Answer.target_prop_%d",tn)],
                   distractorProp = d[sprintf("Answer.distractor_prop_%d",tn)],
                   targetPosition = d[sprintf("Answer.target_position_%d",tn)],
                   distractorPosition = d[sprintf("Answer.distractor_position_%d",tn)],
                   choice = d[sprintf("Answer.choice_%d",tn)],
```

```
                    choiceCorrect = d[sprintf("Answer.choice_correct_%d",tn)])
  names(df) <- c("id","seqCond","trial","level","item","targetProp","distractorProp",
                 "targetPosition","distractorPosition","choice","choiceCorrect")
  return(df)
}

d2 <- rbind(trial.df(1),trial.df(2),trial.df(3))
```

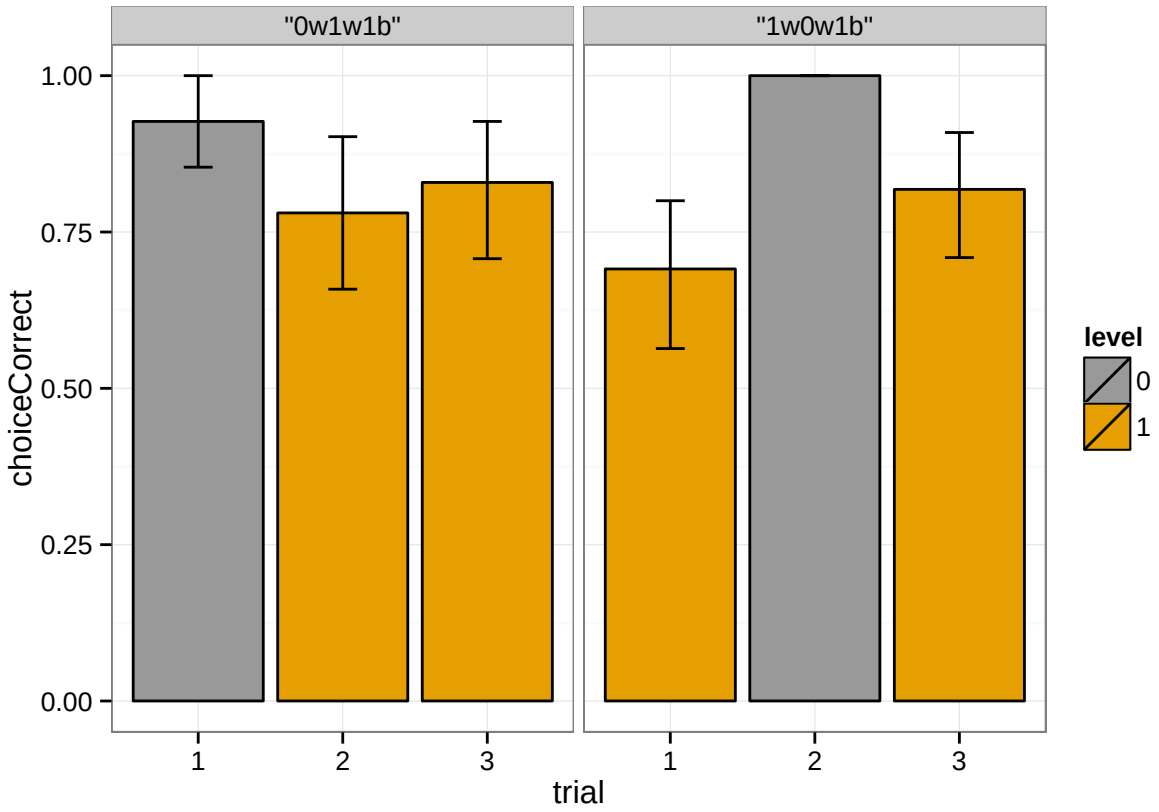
```
#statistics for boolean factors; copied from useful.R, with a slightly different mean function to work
l.mean <- function(...){mean(as.logical(...))}
l.theta <- function(x,xdata,na.rm=T) {l.mean(xdata[x],na.rm=na.rm)}
l.ci.low <- function(x,na.rm=T) {
  l.mean(x,na.rm=na.rm) - quantile(bootstrap(1:length(x),1000,l.theta,x,na.rm=na.rm)$thetastar,.025,na.
l.ci.high <- function(x,na.rm=T) {
  quantile(bootstrap(1:length(x),1000,l.theta,x,na.rm=na.rm)$thetastar,.975,na.rm=na.rm) - l.mean(x,na.

ms <- aggregate(choiceCorrect ~ seqCond + trial + level,data = d2,l.mean)
ms$cil <- aggregate(choiceCorrect ~ seqCond + trial + level, data = d2, l.ci.low)$choiceCorrect
ms$cih <- aggregate(choiceCorrect ~ seqCond + trial + level, data = d2, l.ci.high)$choiceCorrect

#colorblind-friendly color palettes
cbPalette <- c("#999999", "#E69F00", "#56B4E9", "#009E73", "#F0E442", "#0072B2", "#D55E00", "#CC79A7")
cbbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442", "#0072B2", "#D55E00", "#CC79A7")
```

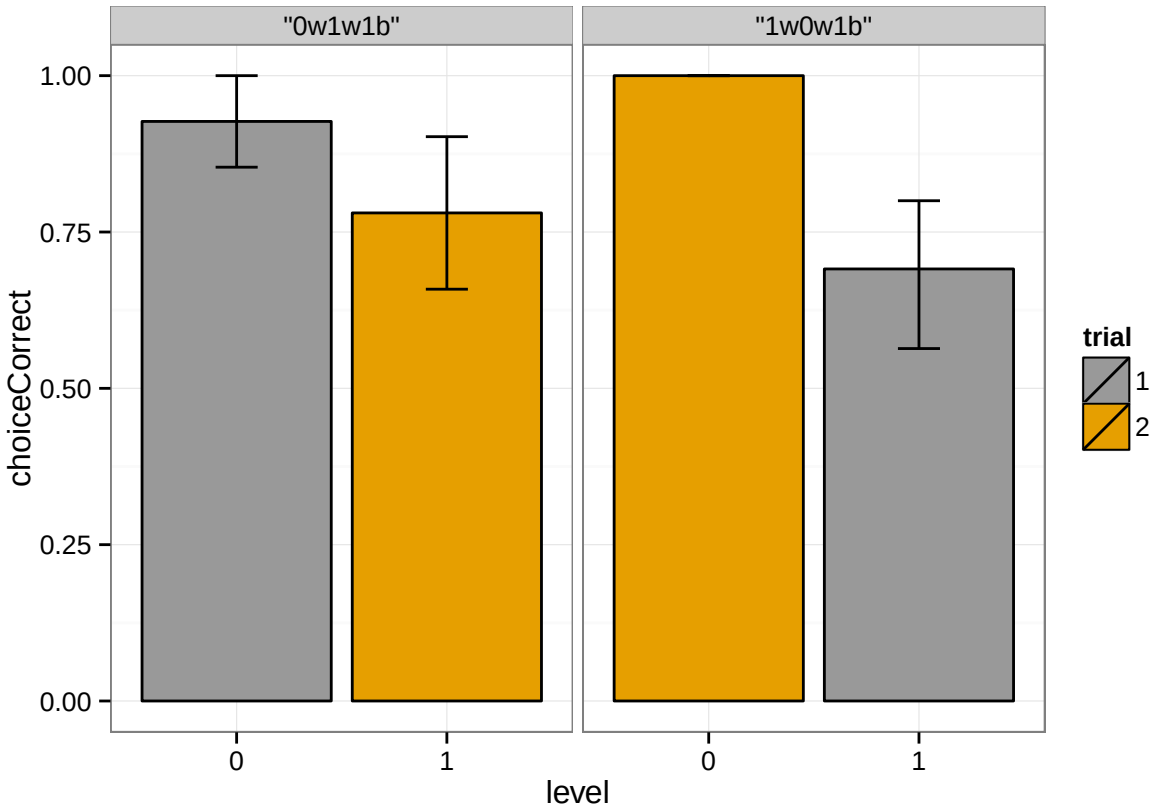
Compare performance on trials, in chronological order; facet by ordering condition.

```
ggplot(data = ms, aes(x = trial, y = choiceCorrect, fill = level)) +
  geom_bar(stat = "identity", color = "black") +
  geom_errorbar(aes(ymin = choiceCorrect - cil, ymax = choiceCorrect + cih), width = 0.2) +
  facet_grid(. ~ seqCond) +
  theme_bw() +
  scale_fill_manual(values=cbPalette)
```



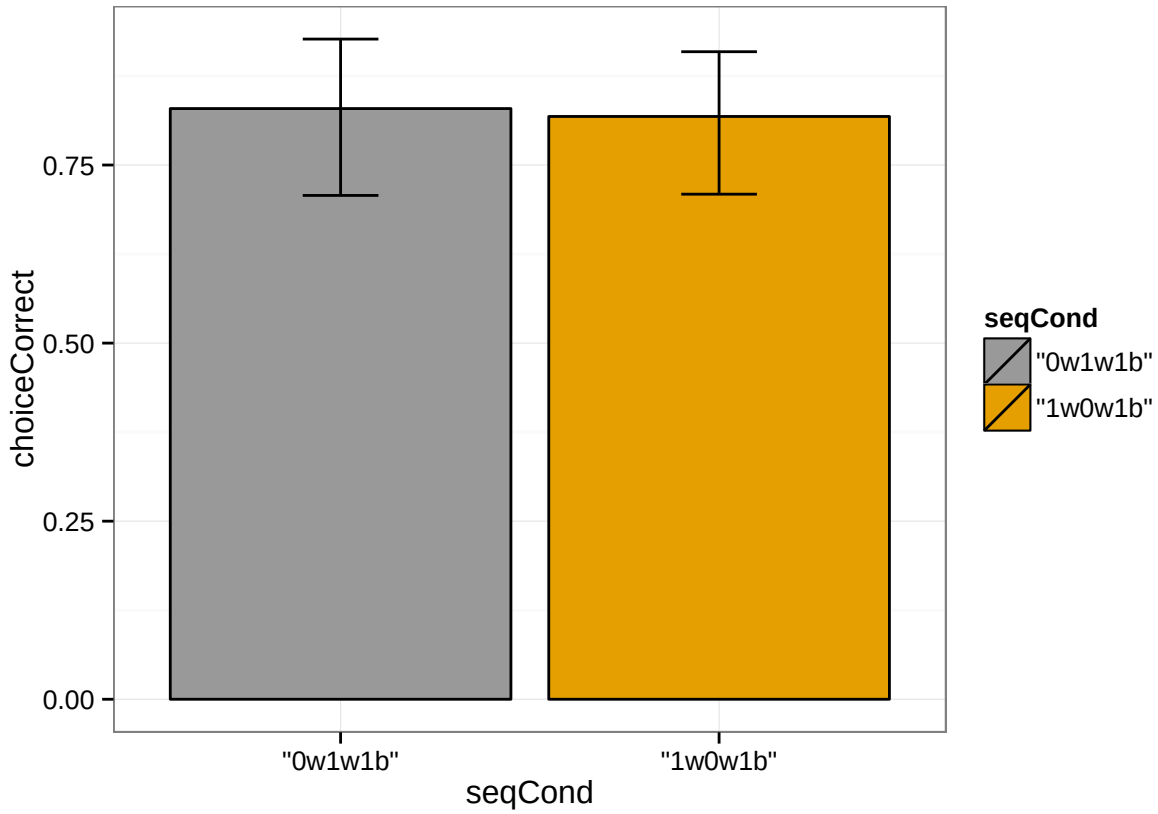
Compare performance on level-0 vs level-1 trials in the two conditions

```
ggplot(data = subset(ms,trial != "3"), aes(x = level, y = choiceCorrect, fill = trial)) +
  geom_bar(stat = "identity", color = "black") +
  geom_errorbar(aes(ymin = choiceCorrect - cil, ymax = choiceCorrect + cih), width = 0.2) +
  facet_grid(. ~ seqCond) +
  theme_bw() +
  scale_fill_manual(values=cbPalette)
```



Compare performance on third trial (level-1, different stimuli)

```
ggplot(data = subset(ms,trial=="3"), aes(x = seqCond, y = choiceCorrect, fill = seqCond)) +
  geom_bar(stat = "identity", color = "black") +
  geom_errorbar(aes(ymin = choiceCorrect - cil, ymax = choiceCorrect + cih), width = 0.2) +
  theme_bw() +
  scale_fill_manual(values=cbPalette)
```


choiceCorrect
0.75
0.50
0.25
0.00
"0w1w1b"
"1w0w1b"
seqCond
seqCond
"0w1w1b"
"1w0w1b"