```
# hit rates per condition with condition*subjValence interaction

setwd('/home/allgoodguys/Documents/Studying/Lund_PhD/epistles/
005_with-Lima/analysis')
library(lme4)
require(brms)
require(shinystan)

df = read.csv ('data/data_preprocessed.csv')[,-1]
# df = read.csv ('data/data_preprocessed_40sounds.csv')[,-1]

df_target = df[df$type=='target',] # targets only
# df_target = df_target[df_target$emotionBlock != "Amusement",] #
exclude amusement to check if it drives the whole effect   #
length(unique(df_target$item))

# descriptives
t = aggregate(subjArousal~emotionBlock, df_target,
function(x)round(mean(x),1))
t = cbind (t, aggregate(subjValence~emotionBlock, df_target,
function(x)round(mean(x),1))[,2])
colnames(t) = c('Emotion','Arousal','Valence')
t$Arousal = paste(t$Arousal, '±',
aggregate(subjArousal~emotionBlock, df_target,
function(x)round(sd(x),1))[,2])
t$Valence = paste(t$Valence, '±',
aggregate(subjValence~emotionBlock, df_target,
function(x)round(sd(x),1))[,2])
# write.csv (t, 'table_arousal-valence_descriptives.csv')

# models
mod_ar = glmer(hit ~ condition*subjArousal + (1|subject)+(1|item),
family='binomial', data=df_target, nAGQ=0)
summary(mod_ar)
drop1(mod_ar, test='Chisq') # no

mod0 = glmer(hit ~ condition*subjValence + (1|subject)+(1|item),
family='binomial', data=df_target, nAGQ=0)
summary(mod0)
drop1(mod0, test='Chisq') # very weak main effect of subjValence,
but a sizable interaction with condition. Both 192 and 40 sounds

mod = brm(hit ~ condition*subjValence + (1|subject)+(1|item), data =
df_target, warmup=100, iter=500, chains=4, cores=4,
family='bernoulli', ranef=F, prior=set_prior("normal(0,5)"))
# saveRDS(mod, 'models/mod_hit~cond*val.RDS') # saveRDS(mod,
'models/mod_hit~cond*val_40sounds.RDS') # saveRDS(mod,
'mod_hit~cond*val_no-amus.RDS')
# mod = readRDS('models/mod_hit~cond*val.RDS') # mod =
readRDS('models/mod_hit~cond*val_40sounds.RDS') # mod =
readRDS('models/mod_hit~cond*val_no-amus.RDS')
# stancode(mod)

plot(mod)
```

```
summary(mod)
# launch_shiny(mod)

coda = posterior_samples(mod)
colnames(coda)

# for plotting
antilogit = function(x)1/(1+exp(-x))*100
val = seq(min(df_target$subjValence), max(df_target$subjValence),
length.out=100)
deliberated = antilogit(sapply(1:length(val), function(x)coda[,1] +
coda[,5]*val[x]))
fast = antilogit(sapply(1:length(val), function(x)coda[,1]+coda[,2]+
(coda[,5]+coda[,6])*val[x]))
load1 = antilogit(sapply(1:length(val), function(x)coda[,1]+coda[,
3]+(coda[,5]+coda[,7])*val[x]))
load2 = antilogit(sapply(1:length(val), function(x)coda[,1]+coda[,
4]+(coda[,5]+coda[,8])*val[x]))

df_plot = data.frame (subjValence=rep(val,4),

condition=rep(c('Deliberated','Fast','Load1','Load2'),each=100),
                      fit=NA, lwr=NA, upr=NA)
df_plot[,3:5] = rbind(
  t(apply (deliberated, 2, function(x)quantile(x, probs=c(.5, .
025, .975)))),
  t(apply (fast, 2, function(x)quantile(x, probs=c(.5, .025, .
975)))),
  t(apply (load1, 2, function(x)quantile(x, probs=c(.5, .025, .
975)))),
  t(apply (load2, 2, function(x)quantile(x, probs=c(.5, .025, .
975))))
)

# or, on logit scale: df_plot[,3:5] = log(df_plot[,3:5]/(100-
df_plot[,3:5]))
ggplot(df_plot, aes(x=subjValence, y=fit, group=condition,
col=condition, fill=condition, linetype=condition))+
  geom_line(size=2)+
  geom_ribbon(aes(ymin=lwr,ymax=upr), alpha=0.1, col='white') +
  # ylim(c(70,100)) +
  ylab('Hit rate, %') +
  xlab('Subjective valence of sound') +
  theme_bw()

df_plot$subjValence = df_plot$subjValence * 100 / 6  # to %
ggplot(df_plot, aes(x=subjValence, y=fit, group=condition,
col=condition, fill=condition))+
  geom_line(size=2)+
  geom_ribbon(aes(ymin=lwr,ymax=upr), alpha=0.1, col='white') +
  geom_density(data = df_target, aes(x = subjValence * 100 / 6, y
= ..scaled.. * 75), inherit.aes = FALSE, adjust = 2, col = 'gray50',
linetype = 2, show.legend = F) +
  # ylim(c(10,100)) +
```

```
  ylab('Hit rate, %') +
  xlab('Subjective valence of sound, %') +
  theme_bw()


## Contrasts
# Is the difference between deliberated vs rest greater for max
positive vs max negative sounds?
quantile(deliberated[,100] - fast[,100], probs=c(.5, .025, .975)) #
2.8% [1.2, 5.7] - that's for max valence
quantile(deliberated[,1] - fast[,1], probs=c(.5, .025, .975)) # 1.6%
[-2.3, 5.7] - that's for min valence

quantile(deliberated[,100] - (fast[,100]+load1[,100]+load2[,100])/
3 , probs=c(.5, .025, .975)) # 4.1% [2.3, 7.0] - that's for max
valence
quantile ( deliberated[,1] - (fast[,1]+load1[,1]+load2[,1])/3 ,
probs=c(.5, .025, .975)) # 1.0% [-2.4, 4.0] - that's for min valence
quantile((deliberated[,100] - (fast[,100]+load1[,100]+load2[,100])/
3) - (deliberated[,1] - (fast[,1]+load1[,1]+load2[,1])/3) ,
probs=c(.5, .025, .975)) # delta: 3.2% [-0.4, 4.0]

quantile(load1[,100] - load2[,100], probs=c(.5, .025, .975)) # 3.4%
[0.4, 8.1] - that's for max valence
quantile(load1[,1] - load2[,1], probs=c(.5, .025, .975)) # 2.1%
[-1.7, 6.3] - that's for min valence

# For which conditions does subjValence have a non-zero slope?
colnames(coda)
quantile(coda[,5], probs=c(.5, .025, .975)) # deliberated
quantile(coda[,5]+coda[,6], probs=c(.5, .025, .975)) # fast
quantile(coda[,5]+coda[,7], probs=c(.5, .025, .975)) # load1
quantile(coda[,5]+coda[,8], probs=c(.5, .025, .975)) # load2

# What contrasts between conditions depend on valence?
quantile( -(coda[,6]+coda[,7]+coda[,8])/3, probs=c(.5,.025,.975)) #
delib vs rest: 0.30 [0.16, 0.46]
quantile(-coda[,6], probs=c(.5,.025,.975)) # delib vs fast: 0.24
[0.08, 0.41]
quantile(coda[,7]-coda[,8], probs=c(.5,.025,.975)) # load 1 vs load
2: 0.07 [-0.06, 0.20]
quantile( coda[,6]-(coda[,7]+coda[,8])/2, probs=c(.5,.025,.975)) #
fast vs mean(load1, load2): 0.09 [-0.03, 0.21]
# or, as odds ratios or odds ratios over the entire span of valence
(0 to 6):
exp(6*quantile( -(coda[,6]+coda[,7]+coda[,8])/3, probs=c(.
5,.025,.975))) # delib vs rest
exp(6*quantile(-coda[,6], probs=c(.5,.025,.975))) # delib vs fast
exp(6*quantile(coda[,7]-coda[,8], probs=c(.5,.025,.975))) # load 1
vs load 2
exp(6*quantile( coda[,6]-(coda[,7]+coda[,8])/2, probs=c(.
5,.025,.975))) # fast vs mean(load1, load2)
```

```
## Another plot: delib-rest ~ subjValence
# val = val / 6 * 100  # to %
df_plot = data.frame (subjValence=val, fit=NA, lwr=NA, upr=NA)
df_plot[,2:4] = t(apply (deliberated-(fast+load1+load2)/3, 2,
function(x)quantile(x, probs=c(.5, .025, .975))))

p1 = ggplot(df_plot, aes(x=subjValence, y=fit))+
  geom_line(size=2)+
  geom_ribbon(aes(ymin=lwr,ymax=upr),alpha=0.3) +
  annotate ('text', x=3, y=7.5, label='Deliberated vs. rest',
size=5, fontface='bold') +
  ylab('\U2206 hit rate, %') +
  xlab('Subjective valence') +
  xlim(c(0,6)) +
  ylim(c(-8, 8)) +
  geom_hline(yintercept=0, linetype=2, size=1, col='gray50') +
  theme_bw() +
  theme(panel.grid=element_blank())

## delib-fast ~ subjValence
df_plot = data.frame (subjValence=val, fit=NA, lwr=NA, upr=NA)
df_plot[,2:4] = t(apply (deliberated-fast, 2, function(x)quantile(x,
probs=c(.5, .025, .975))))

p2 = ggplot(df_plot, aes(x=subjValence, y=fit))+
  geom_line(size=2)+
  geom_ribbon(aes(ymin=lwr,ymax=upr),alpha=0.3) +
  annotate ('text', x=3, y=7.5, label='Deliberated vs. fast',
size=5, fontface='bold') +
  ylab('\U2206 hit rate, %') +
  xlab('Subjective valence') +
  xlim(c(0,6)) +
  ylim(c(-8, 8)) +
  geom_hline(yintercept=0, linetype=2, size=1, col='gray50') +
  theme_bw() +
  theme(panel.grid=element_blank())

## load1-load2 ~ subjValence
df_plot = data.frame (subjValence=val, fit=NA, lwr=NA, upr=NA)
df_plot[,2:4] = t(apply (load1-load2, 2, function(x)quantile(x,
probs=c(.5, .025, .975))))

p3 = ggplot(df_plot, aes(x=subjValence, y=fit))+
  geom_line(size=2)+
  geom_ribbon(aes(ymin=lwr,ymax=upr),alpha=0.3) +
  annotate ('text', x=3, y=7.5, label='Low vs. high load', size=5,
fontface='bold') +
  ylab('\U2206 hit rate, %') +
  xlab('Subjective valence') +
  xlim(c(0,6)) +
  ylim(c(-8, 8)) +
  geom_hline(yintercept=0, linetype=2, size=1, col='gray50') +
  theme_bw() +
  theme(panel.grid=element_blank())
```

```
## fast-load ~ subjValence
df_plot = data.frame (subjValence=val, fit=NA, lwr=NA, upr=NA)
df_plot[,2:4] = t(apply (fast-(load1+load2)/2, 2,
function(x)quantile(x, probs=c(.5, .025, .975))))

p4 = ggplot(df_plot, aes(x=subjValence, y=fit))+
  geom_line(size=2)+
  geom_ribbon(aes(ymin=lwr,ymax=upr),alpha=0.3) +
  annotate ('text', x=3, y=7.5, label='Fast vs. load', size=5,
fontface='bold') +
  ylab('\U2206 hit rate, %') +
  xlab('Subjective valence') +
  xlim(c(0,6)) +
  ylim(c(-8, 8)) +
  geom_hline(yintercept=0, linetype=2, size=1, col='gray50') +
  theme_bw() +
  theme(panel.grid=element_blank())

source('ggplot_multiple.R')
multiplot(p1, p2, p3, p4, cols = 2)
# ggsave('pix/subjVal.jpg', multiplot(p2, p1, p4, p3, cols = 4),
width=16, height=4, units='cm', dpi=300, scale=2)
# ggsave('pix/subjVal_noAmus.jpg', multiplot(p2, p1, p4, p3, cols =
4), width=16, height=4, units='cm', dpi=300, scale=2)


### Table of beta-coefficients for subjValence in different
conditions, for 192 or 40 sounds
colnames(coda)
out = data.frame (beta = c('Deliberated','Fast','Load 1','Load 2'),
                  median = NA, CI = NA)
t = list(
  'Deliberated' = quantile(coda[,5], probs=c(.5,.025,.975)),
  'Fast' = quantile(coda[,5]+coda[,6], probs=c(.5,.025,.975)),
  'Load 1' = quantile(coda[,5]+coda[,7], probs=c(.5,.025,.975)),
  'Load 2' = quantile(coda[,5]+coda[,8], probs=c(.5,.025,.975))
)

for (i in 1:nrow(out)){
  out[i,2] =  round(t[[i]][1],2)
  out[i,3] = paste0('[', round(t[[i]][2],2), ', ', round(t[[i]][3],
2), ']')
}
# write.csv (out, 'temp.csv')


### Non-Bayesian testing of the significance of delib vs rest for
each emotion separately (8 comparisons)
alpha = .05 / 8  # Bonferroni
df$delib = df$condition == 'Deliberated'
for (em in levels(df$emotionBlock)) {
  temp = df_load[df_load$emotionBlock == em, ]
  mod = glmer(hit ~ subjValence + (1|subject)+(1|item),
```

```
family='binomial', data=temp, nAGQ=0)
  s = summary(mod)$coef[2, 3:4]
  print(em)
  print(s[s[2] < alpha])
}
```