TIME_Exps_Behavioural_Analysis

2025-01-24

```r
# Load libraries and set path --------------------------------------------------
library(lme4);library(car);
```

```
## Loading required package: Matrix

## Loading required package: carData
```

```r
library(ggplot2);library(emmeans)
```

```
## Welcome to emmeans.
## Caution: You lose important information if you filter this package's results.
## See '? untidy'
```

```r
library(tinytex);library(MKinfer)
library(BayesFactor)
```

```
## Loading required package: coda

## ************
## Welcome to BayesFactor 0.9.12-4.7. If you have questions, please contact Richard Morey (richarddmorey
##
## Type BFManual() to open the manual.
## ************
```

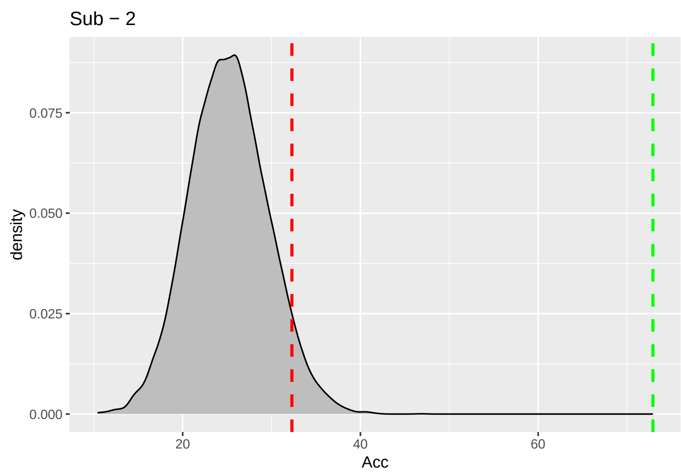
```r
# get data folder paths
# Current directory needs to be set to the script's folder
Exp1DataFolder <- paste(getwd(), "/Exp1Data", sep = "")
Exp2DataFolder <- paste(getwd(), "/Exp2Data", sep = "")
Exp3DataFolder <- paste(getwd(), "/Exp3Data", sep = "")
Exp2SyncDataFolder <- paste(Exp2DataFolder,"/SyncDiscData", sep = "")
Exp3SyncDataFolder <- paste(Exp3DataFolder,"/SyncDiscData", sep = "")

#Colour coding
LongNoFlickerColour <- "#F8766D"
ShortNoFlickerColour <- "#7CAE00"
SyncThetaColour <- "#00BFC4"
AsyncThetaColour <- "#C77CFF"
SyncDeltaColour <- "#FF9913"
SyncDiscColour <- "#CC9999"
```

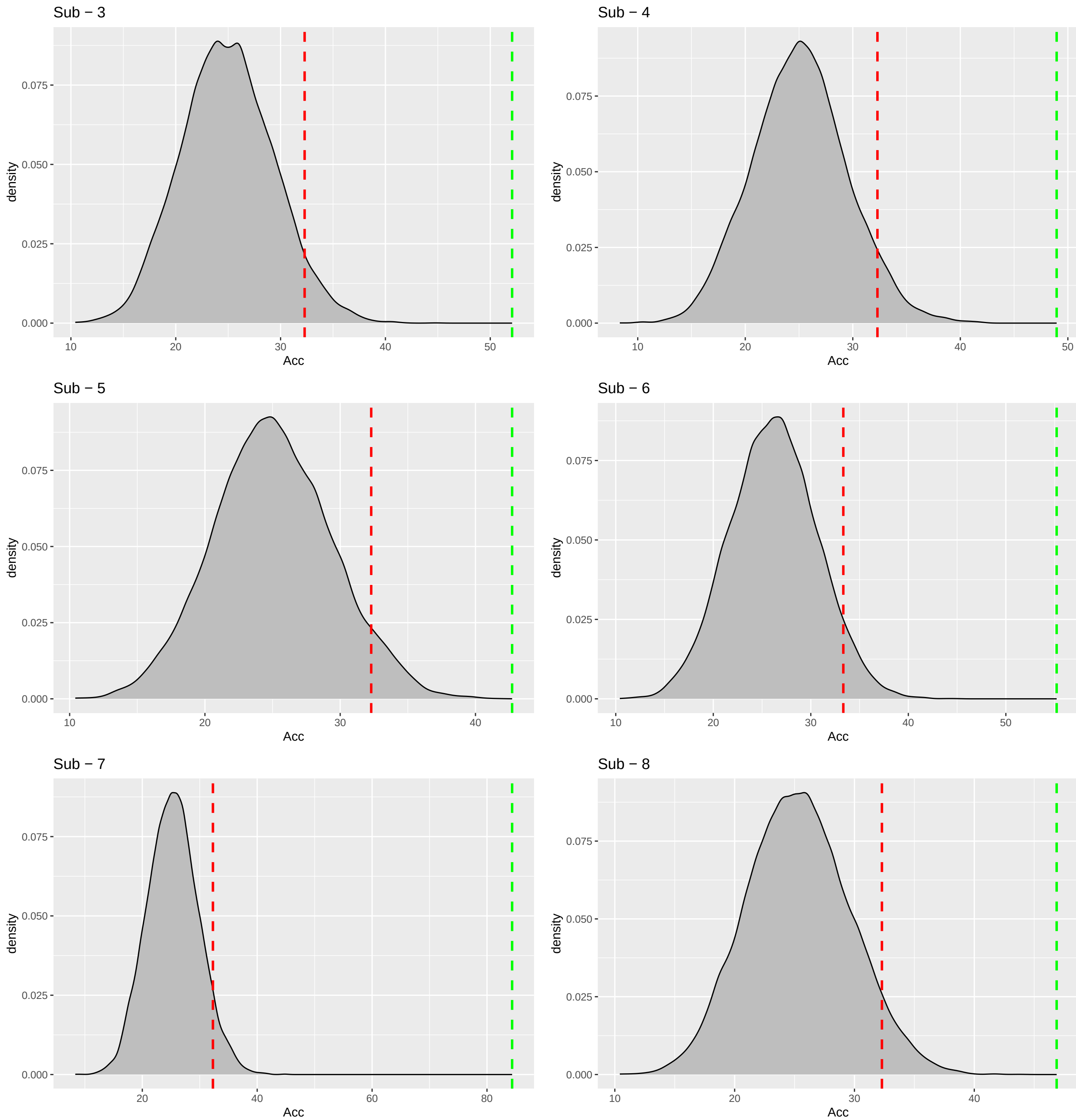
```r
# Arrange the data --------------------------------------------------------

# Set the seed for reproducibility
set.seed(13021996)

# List of subject names
subjects <- c("Sub_01", "Sub_02", "Sub_03", "Sub_04", "Sub_05", "Sub_06", "Sub_07",
              "Sub_08", "Sub_09", "Sub_10", "Sub_11", "Sub_12", "Sub_13", "Sub_14",
              "Sub_15", "Sub_16", "Sub_17", "Sub_18", "Sub_19", "Sub_20", "Sub_21",
              "Sub_22", "Sub_23", "Sub_24", "Sub_25", "Sub_26", "Sub_27")

# Initialize data frame
Exp1Data <- data.frame()

# Loop through subjects to fetch the data
for (iSub in 1:length(subjects)) {
  # Define file name
  File_pattern <- list.files(path = Exp1DataFolder, pattern = subjects[iSub])

  # Read the CSV file
  sub_data <- read.csv(file.path(Exp1DataFolder, File_pattern), header = TRUE)

  # omit flicker conditions because of the experiment error
  sub_data <- sub_data[sub_data$Condition == "NoFlickerHalf" | sub_data$Condition == "NoFlickerFull",]

  # Add subject data into the data data
  Exp1Data <- rbind(Exp1Data, sub_data)
}

Subs <- 1:length(subjects)

# Check if subject has less trials than it is supposed to be
for (iSub in Subs) {

  nRows = na.omit(Exp1Data[Exp1Data$ParticipantID == iSub,])

  if (dim(nRows)[1] < 96) {
    print(paste("Sub", iSub, "has", dim(nRows)[1]), sep = "")
  }
}
```

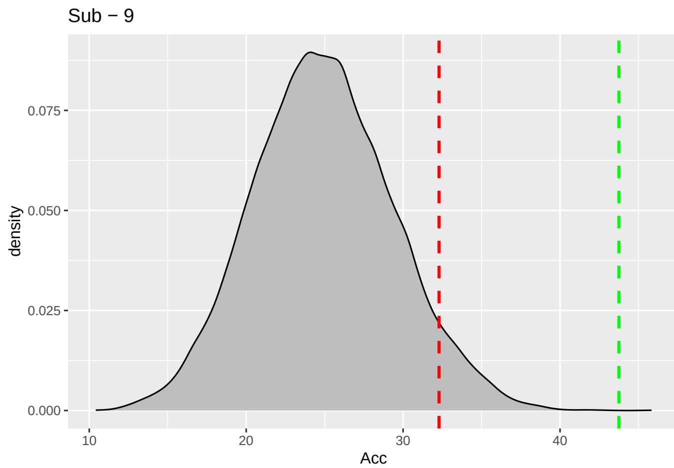
```
## [1] "Sub 5 has 80"
```

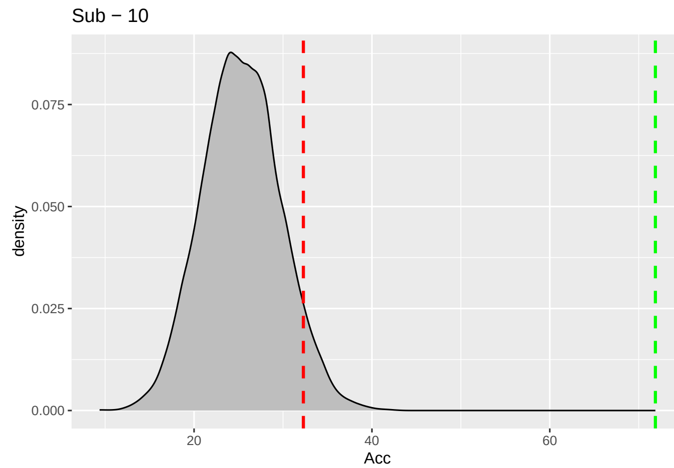
```r
# Calculate the mean accuracy for each subject
Mean_Accs <- tapply(Exp1Data$Accuracy, Exp1Data$ParticipantID , mean)

# parameters
nIterations <- 10000

# Loop through subjects to test whether they performed near chance-level
for (iSub in Subs) {

  NullDist <- data.frame(Acc=rep(NA,nIterations))
```

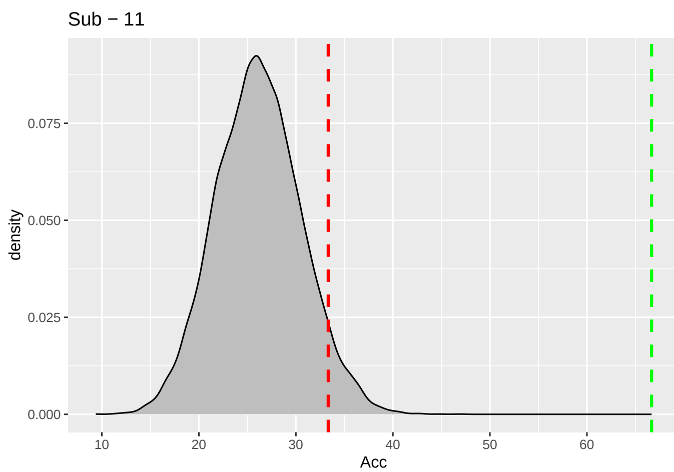
```
  for (i in 1:nIterations) {
    # Generate random responses between 1 and 4
    Responses <- Exp1Data[Exp1Data$ParticipantID == iSub, "Response"]

    # Generate random answers between 1 and 4
    Answers <- sample(Exp1Data[Exp1Data$ParticipantID == iSub, "CorrectKey"])

    NullDist[i,1] <- ((sum(Responses == Answers))/length(Responses))*100
  }

  # Sort distribution
  SortedNullDist <- NullDist[order(NullDist$Acc),]
  Significance_Threshold <- SortedNullDist[length(SortedNullDist)-nIterations*.05]

  print(ggplot(NullDist, aes(x=Acc)) + geom_density(fill="gray") +
          geom_vline(aes(xintercept = Mean_Accs[as.character(iSub)]*100),
                     color="green", linetype="dashed", linewidth=1)+
          geom_vline(aes(xintercept = Significance_Threshold),
                     color="red", linetype="dashed", linewidth=1)+
          ggtitle(paste("Sub -", as.character(iSub))))

  if ((Mean_Accs[as.character(iSub)])*100 <= Significance_Threshold) {
    print(paste("Sub", iSub,' is at chance-level!!!'))
  }
}
```

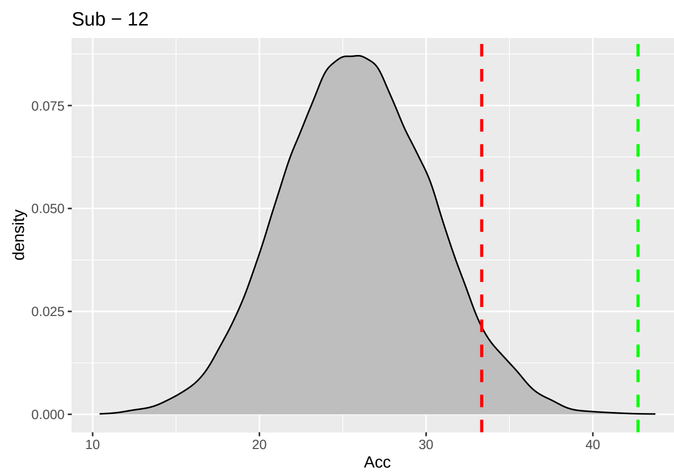


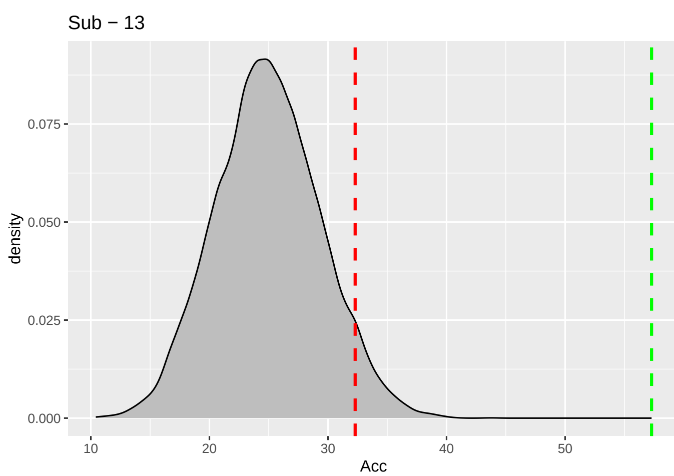

Sub − 3
Sub − 4
Sub − 5
Sub − 6
Sub − 7
Sub − 8
density
Acc
0.000
0.025
0.050
0.075
10
20
30
40
50
60
80

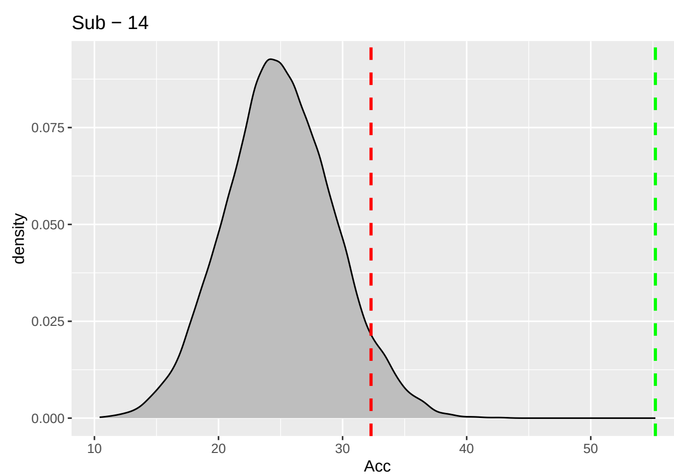

Sub − 9
density
0.000
0.025
0.050
0.075
10
20
30
40
Acc

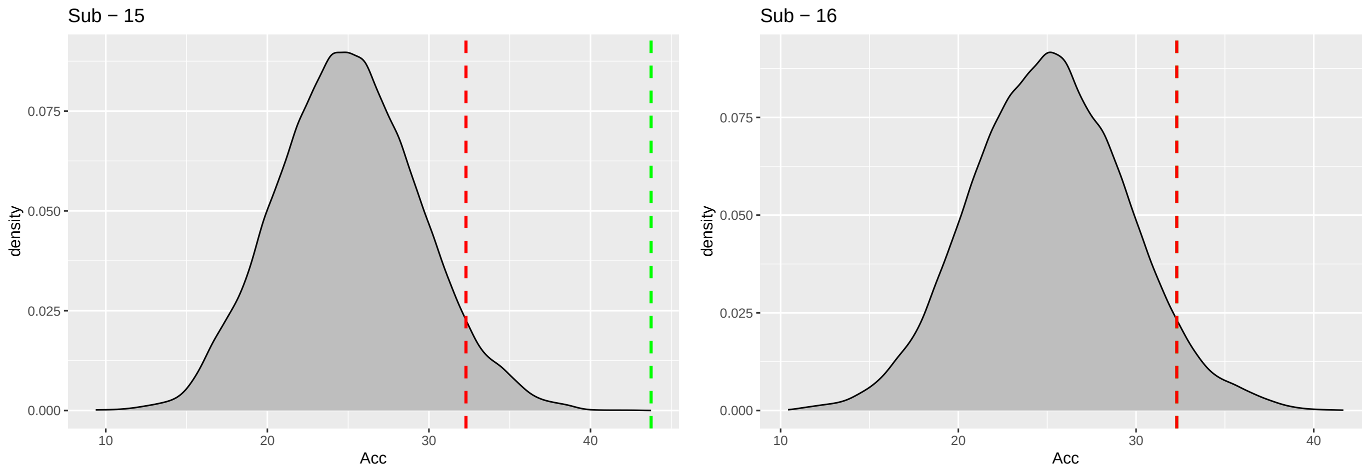

Sub − 10
density
0.000
0.025
0.050
0.075
20
40
60
Acc


Sub − 11
density
0.000
0.025
0.050
0.075
10
20
30
40
50
60
Acc

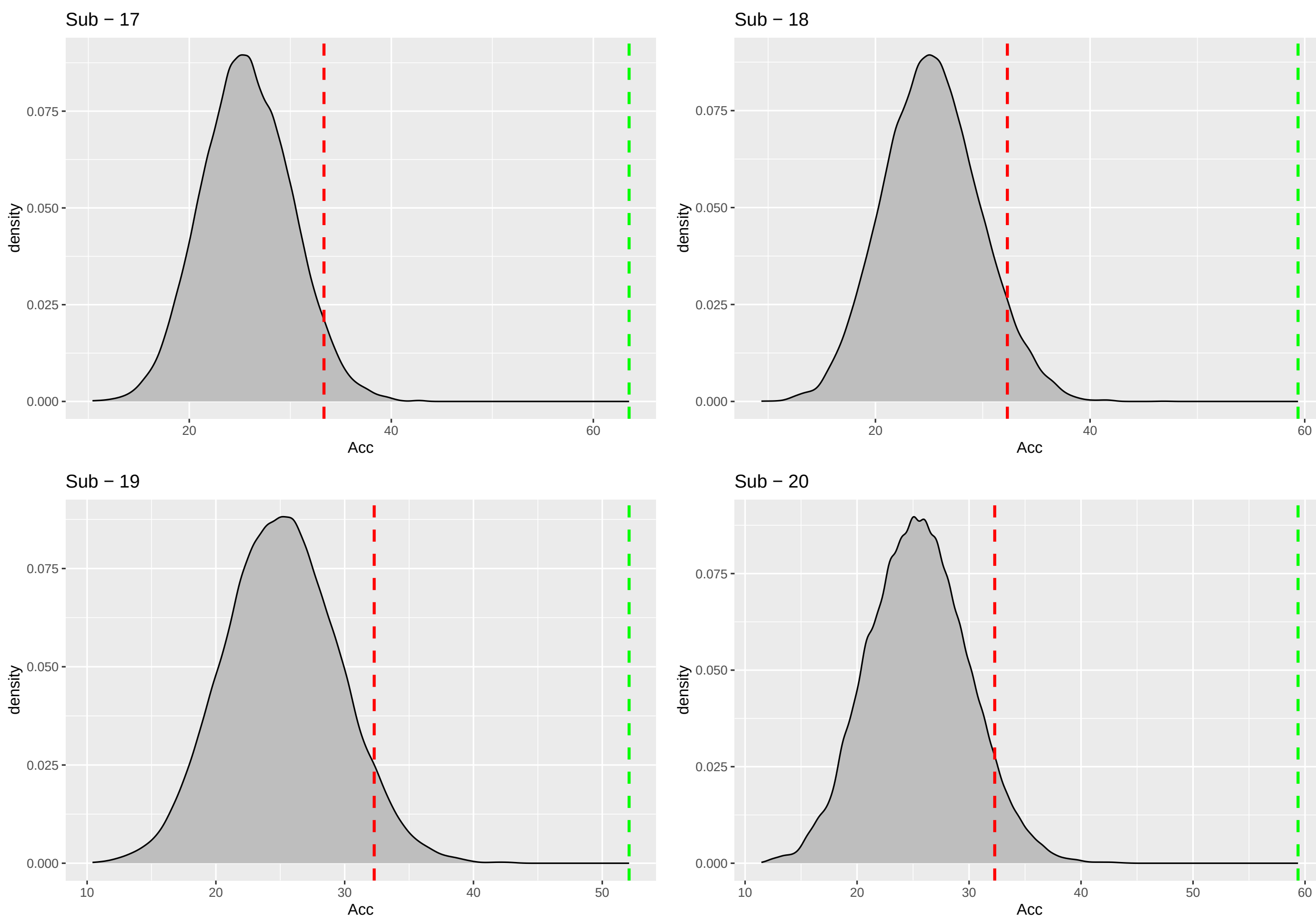

Sub − 12
density
0.000
0.025
0.050
0.075
10
20
30
40
Acc

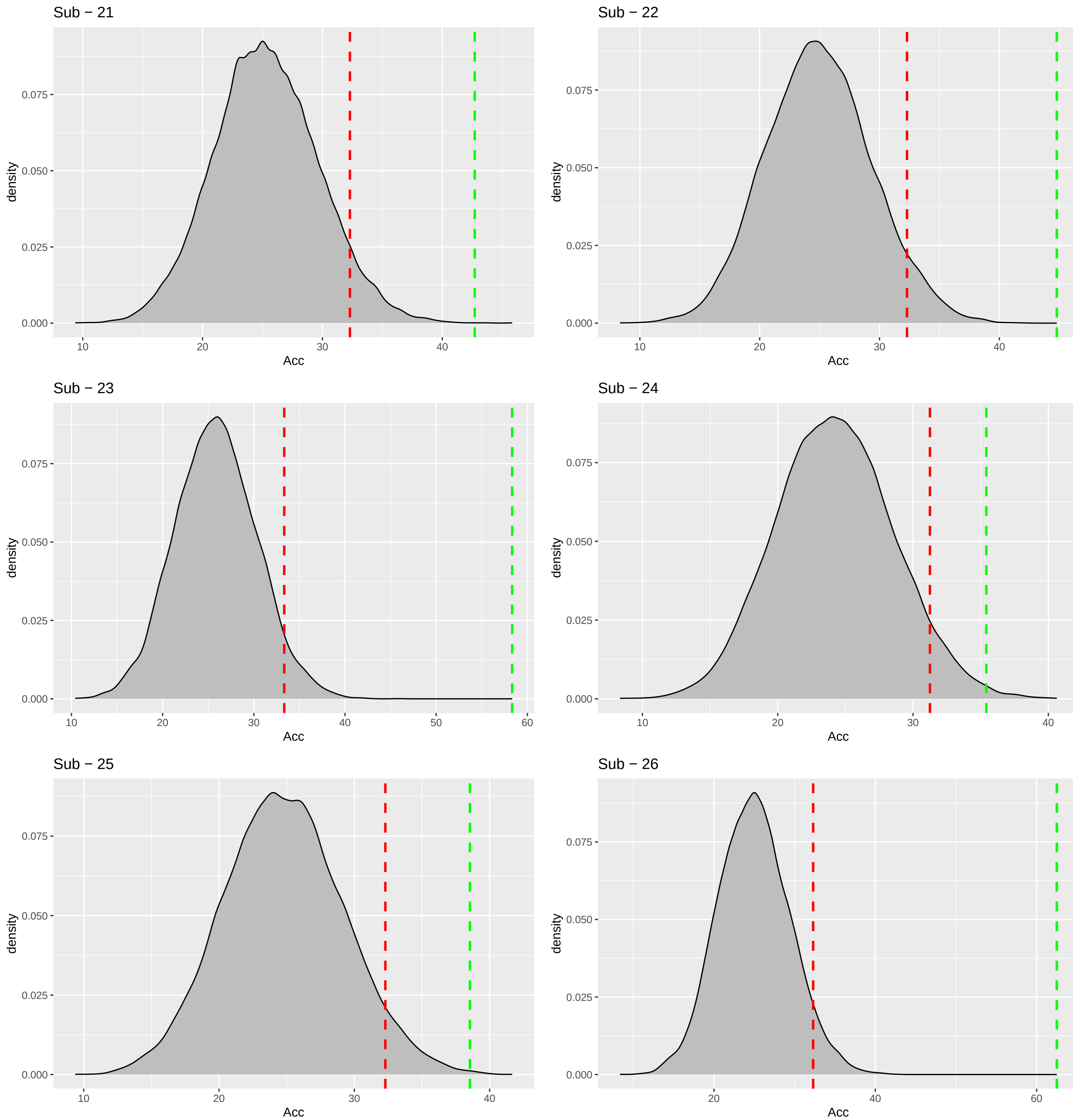

Sub − 13
density
0.000
0.025
0.050
0.075
10
20
30
40
50
Acc

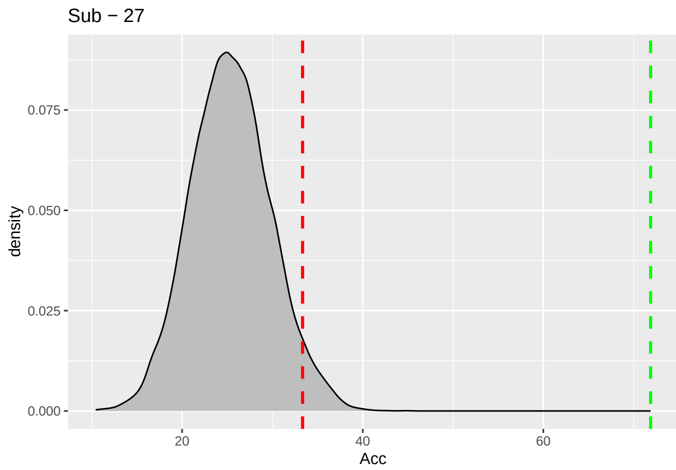

Sub − 14
density
0.000
0.025
0.050
0.075
10
20
30
40
50
Acc




```
## [1] "Sub 16  is at chance-level!!!"
```




Sub − 21
Sub − 22
Sub − 23
Sub − 24
Sub − 25
Sub − 26
density
Acc
0.000
0.025
0.050
0.075
10
20
30
40
50
60




```r
# Set the seed for reproducibility
set.seed(13021996)

# List of subject names without subject 16
subjects <- c("Sub_01", "Sub_02", "Sub_03", "Sub_04", "Sub_05", "Sub_06", "Sub_07",
              "Sub_08", "Sub_09", "Sub_10", "Sub_11", "Sub_12", "Sub_13", "Sub_14",
              "Sub_15", "Sub_17", "Sub_18", "Sub_19", "Sub_20", "Sub_21",
              "Sub_22", "Sub_23", "Sub_24", "Sub_25", "Sub_26", "Sub_27")

# Initialize data frame
Exp1Data <- data.frame()

# Loop through subjects to fetch the data
for (iSub in 1:length(subjects)) {
  # Define file name
  File_pattern <- list.files(path = Exp1DataFolder, pattern = subjects[iSub])

  # Read the CSV file
  sub_data <- read.csv(file.path(Exp1DataFolder, File_pattern), header = TRUE)

  # Add subject data into the data data
  Exp1Data <- rbind(Exp1Data, sub_data)
}

# Run model for only no-flicker conditions
Exp1Data.model = glmer(Accuracy ~ Condition + (1|ClipID) + (1|ParticipantID),
                  data = Exp1Data, family = binomial,
                  subset = Exp1Data$Condition == "NoFlickerHalf" | Exp1Data$Condition == "NoFlickerFul
Anova(Exp1Data.model)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##            Chisq Df Pr(>Chisq)
## Condition 4.9993  1    0.02536 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(Exp1Data.model)
```

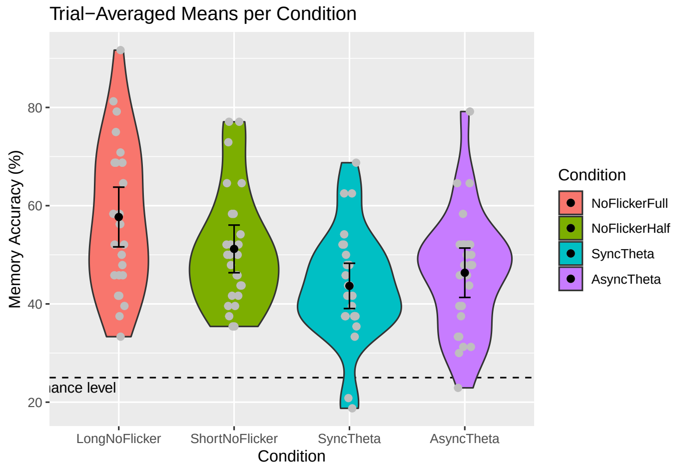
```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition + (1 | ClipID) + (1 | ParticipantID)
##    Data: Exp1Data
##  Subset: Exp1Data$Condition == "NoFlickerHalf" | Exp1Data$Condition ==
##     "NoFlickerFull"
##
##      AIC      BIC   logLik deviance df.resid
##   3271.1   3294.3  -1631.5   3263.1     2492
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.6753 -0.8782  0.4763  0.8173  2.1705
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.5091   0.7135
##  ParticipantID (Intercept) 0.2868   0.5355
## Number of obs: 2496, groups:  ClipID, 192; ParticipantID, 26
##
## Fixed effects:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)              0.3686     0.1423   2.590  0.00959 **
## ConditionNoFlickerHalf  -0.3020     0.1351  -2.236  0.02536 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## CndtnNFlckH -0.478
```

```r
Exp1Data.model.emm <- emmeans(Exp1Data.model, "Condition")
pairs(Exp1Data.model.emm)
```

```
##  contrast                      estimate    SE  df z.ratio p.value
##  NoFlickerFull - NoFlickerHalf    0.302 0.135 Inf   2.236  0.0254
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
# Calculate accuracy for each subject and condition
NoFlicker_pSub <- aggregate(Accuracy ~ ParticipantID + Condition, data = Exp1Data, mean)
NoFlicker_pSub$Condition = factor(NoFlicker_pSub$Condition, levels = c("NoFlickerFull", "NoFlickerHalf"

# Plot trial-averaged means
ggplot(NoFlicker_pSub, aes(x = Condition, y = Accuracy*100, fill = Condition)) +
  geom_hline(yintercept=25,linetype=2)+
  annotate("text", x="NoFlickerFull", y=23, label="chance level                     ",size=3.5)+
  geom_violin(trim=TRUE)+
  geom_point(size = 2, color = "gray", position = position_jitterdodge(jitter.width = .4)) +
```

```
  labs(title="Trial-Averaged Means per Condition", x = "Condition", y = "Memory Accuracy (%)")+
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")+
  scale_fill_manual(values = c("NoFlickerFull" = LongNoFlickerColour, "NoFlickerHalf" = ShortNoFlickerC
  scale_x_discrete(labels=c("NoFlickerFull" = "LongNoFlicker", "NoFlickerHalf" = "ShortNoFlicker"))
```

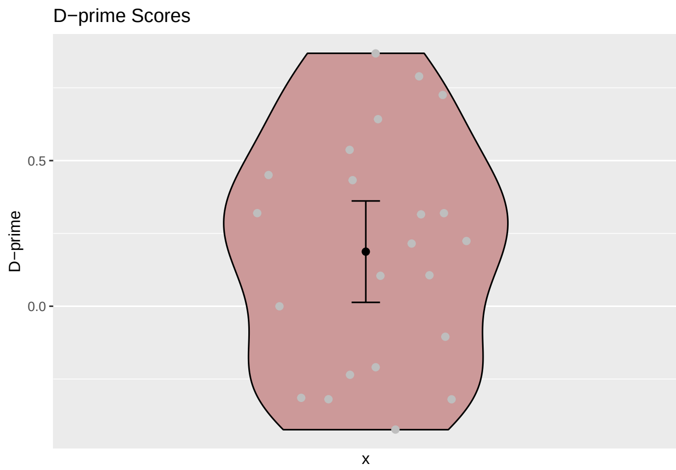



```
# Set the seed for reproducibility
set.seed(13021996)

## Load Exp1 Sync Discrimination task data
# Define file name
Exp1SyncTaskData <- read.csv(file.path(Exp1DataFolder, "Exp1_SyncTaskData.csv"), header = TRUE)

# exclude sub 16
Exp1SyncTaskData <- Exp1SyncTaskData[Exp1SyncTaskData$ParticipantID != 16,]

## Calculate D-prime scores
nSub <- length(unique(Exp1SyncTaskData$ParticipantID))

# Initialize vectors to store results
nHits_pSub <- rep(NA, nSub)
HitRate_pSub <- rep(NA, nSub)
nFAs_pSub <- rep(NA, nSub)
FARate_pSub <- rep(NA, nSub)
Exp1Dprime_pSub <- rep(NA, nSub)
Exp1Sync_subnumsnums <- rep(NA, nSub)

# Loop over subjects
for (iSub in 1:nSub) {

  Exp1SyncData <- Exp1SyncTaskData[Exp1SyncTaskData$ParticipantID == unique(Exp1SyncTaskData$Participan

  # Calculate Hit Rate
  nHits_pSub[iSub] <- sum(Exp1SyncData$SyncRating == 1 & Exp1SyncData$Accuracy == 1)
  HitRate_pSub[iSub] <- nHits_pSub[iSub] / sum(Exp1SyncData$SyncRating == 1)

  # Calculate False Alarm Rate
```

```
  nFAs_pSub[iSub] <- sum(Exp1SyncData$SyncRating == 2 & Exp1SyncData$Accuracy == 0)
  FARate_pSub[iSub] <- nFAs_pSub[iSub] / sum(Exp1SyncData$SyncRating == 2)

  # Calculate d-prime
  Exp1Dprime_pSub[iSub] <- qnorm(HitRate_pSub[iSub]) - qnorm(FARate_pSub[iSub])
  Exp1Sync_subnumsnums[iSub] <- Exp1SyncData$ParticipantID[iSub]
}

# Create a data frame
Exp1DprimeData <- data.frame(ParticipantID = as.factor(Exp1Sync_subnumsnums), Dprime = Exp1Dprime_pSub)

# significance test
(DPrime_out <- boot.t.test(Exp1DprimeData$Dprime, alternative = c("two.sided", "less", "greater"),
                           mu = 0, paired = FALSE, var.equal = FALSE, conf.level = 0.95,
                           R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  Exp1DprimeData$Dprime
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.03562
## bootstrap mean of x (SE) = 0.1873417 (0.08131027)
## 95 percent bootstrap percentile confidence interval:
##  0.02838852 0.34765702
##
## Results without bootstrap:
## t = 2.2379, df = 21, p-value = 0.0362
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.01324902 0.36149390
## sample estimates:
## mean of x
## 0.1873715
```

```
# Create a violin plot
ggplot(data = Exp1DprimeData, aes(x = 1, y = Dprime)) +
  geom_violin(width=1, fill = SyncDiscColour, color = "black") +
  scale_x_discrete( ) +
  geom_point(position = "jitter",size = 2, color = "gray") +
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")+
  labs(title = "D-prime Scores",
       y = "D-prime")
```




```
# Set the seed for reproducibility
set.seed(13021996)

# List of subject names
subjects <- c("Sub_01", "Sub_02", "Sub_03", "Sub_04", "Sub_05", "Sub_06", "Sub_07",
              "Sub_08", "Sub_09", "Sub_10", "Sub_11", "Sub_12", "Sub_13", "Sub_14",
              "Sub_15", "Sub_16", "Sub_17", "Sub_18", "Sub_19", "Sub_20", "Sub_21",
              "Sub_22", "Sub_23", "Sub_24", "Sub_25", "Sub_26", "Sub_27", "Sub_28",
              "Sub_29", "Sub_30", "Sub_31", "Sub_32")

# Initialize data frame
Exp2Data <- data.frame()

# Loop through subjects
for (iSub in 1:length(subjects)) {
  # Define file name
  File_pattern <- paste(subjects[iSub], "_memory.csv", sep = "")

  # Read the CSV file
  sub_data <- read.csv(file.path(Exp2DataFolder, File_pattern), header = TRUE)

  # Combine the subject's runs with the overall combined data
  Exp2Data <- rbind(Exp2Data, sub_data)
}

# Participant Exclusion w/ Permutation ------------------------------------

# Calculate the mean accuracy for each subject
Mean_Accs <- tapply(Exp2Data$Accuracy, Exp2Data$ParticipantID , mean)

# parameters
nIterations <- 10000
Subs <- 1:length(subjects)

for (iSub in Subs) {

  NullDist <- data.frame(Acc=rep(NA,nIterations))
```

```r
  for (i in 1:nIterations) {
    # Generate 192 random responses between 1 and 4
    Responses <- Exp2Data[Exp2Data$ParticipantID == iSub, "Response"]

    # Generate 192 random answers between 1 and 4
    Answers <- sample(Exp2Data[Exp2Data$ParticipantID == iSub, "CorrectKey"])

    NullDist[i,1] <- ((sum(Responses == Answers))/length(Responses))*100
  }

  # Sort distribution
  SortedNullDist <- NullDist[order(NullDist$Acc),]
  Significance_Threshold <- SortedNullDist[length(SortedNullDist)-nIterations*.05]

  # print(ggplot(NullDist, aes(x=Acc)) + geom_density(fill="gray") +
  #         geom_vline(aes(xintercept = Mean_Accs[as.character(iSub)]*100),
  #                     color="green", linetype="dashed", linewidth=1)+
  #         geom_vline(aes(xintercept = Significance_Threshold),
  #                     color="red", linetype="dashed", linewidth=1)+
  #         ggtitle(paste("Sub -", as.character(iSub))))

  if ((Mean_Accs[as.character(iSub)])*100 <= Significance_Threshold) {
    print(c(iSub,' is at chance-level!!!'))
  }
}



# Logistic Mixed-effects Model --------------------------------------------

# recode variable types
Exp2Data$Condition <- as.factor(Exp2Data$Condition)
Exp2Data$ParticipantID <- as.factor(Exp2Data$ParticipantID)
Exp2Data$ClipID <- as.factor(Exp2Data$ClipID)

# registered model
Exp2.model = glmer(Accuracy ~ Condition + (1|ClipID) + (1|ParticipantID),
                   data=Exp2Data, family = binomial)

# summary and plotting
Anova(Exp2.model)
```

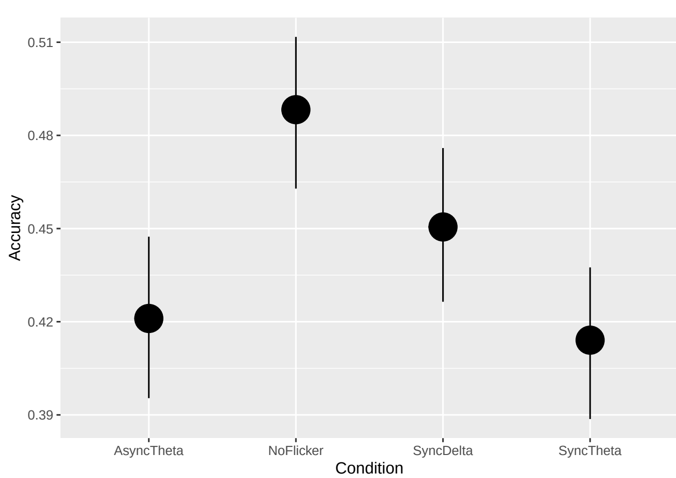
```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##            Chisq Df Pr(>Chisq)
## Condition 22.933  3  4.171e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(Exp2.model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
```

```
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition + (1 | ClipID) + (1 | ParticipantID)
##    Data: Exp2Data
##
##      AIC      BIC   logLik deviance df.resid
##   8179.2   8219.5  -4083.6   8167.2     6122
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.0007 -0.8456 -0.6064  0.9887  2.2851
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.29706  0.5450
##  ParticipantID (Intercept) 0.07318  0.2705
## Number of obs: 6128, groups:  ClipID, 192; ParticipantID, 32
##
## Fixed effects:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -0.34490    0.08226  -4.193 2.75e-05 ***
## ConditionNoFlicker   0.29493    0.07591   3.885 0.000102 ***
## ConditionSyncDelta   0.12941    0.07596   1.704 0.088436 .
## ConditionSyncTheta  -0.02995    0.07629  -0.393 0.694657
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CndtNF CndtSD
## CndtnNFlckr -0.469
## CndtnSyncDl -0.468  0.507
## CndtnSyncTh -0.466  0.505  0.504
```

```r
ggplot(Exp2Data,aes(x=Condition,y=Accuracy))+
  stat_summary(fun.data=mean_cl_boot,size=2)
```



```r
Exp2.emm.s <- emmeans(Exp2.model,"Condition")
#pairs(Exp2.emm.s)

# Conditions within Exp2.emm.s ordered: AsyncTheta, NoFlicker, SyncDelta, SyncTheta
# Planned comparisons for correction .05/3
(out1 <- contrast(Exp2.emm.s, list(SyncTheta.vs.NoFlicker  = c(0, -1, 0, 1))))
```

```
##  contrast               estimate     SE  df z.ratio p.value
##  SyncTheta.vs.NoFlicker   -0.325 0.0757 Inf  -4.289  <.0001
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out2 <- contrast(Exp2.emm.s, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

```
##  contrast                estimate     SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta  -0.0299 0.0763 Inf  -0.393  0.6947
##
## Results are given on the log odds ratio (not the response) scale.
```

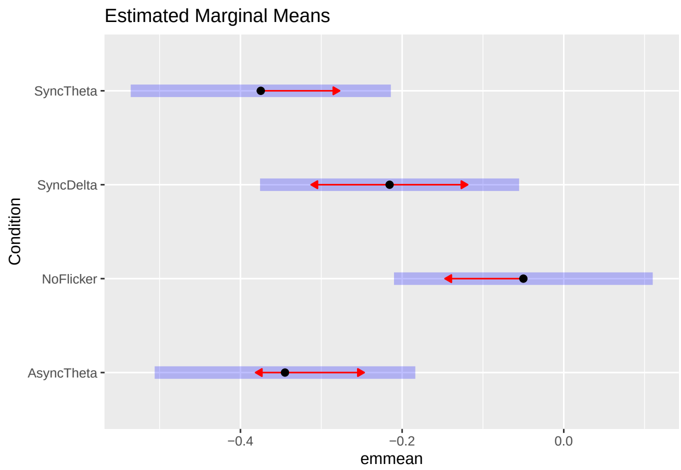
```r
(out3 <- contrast(Exp2.emm.s, list(SyncTheta.vs.SyncDelta  = c(0, 0, -1, 1))))
```

```
##  contrast               estimate     SE  df z.ratio p.value
##  SyncTheta.vs.SyncDelta   -0.159 0.0758 Inf  -2.103  0.0355
##
## Results are given on the log odds ratio (not the response) scale.
```

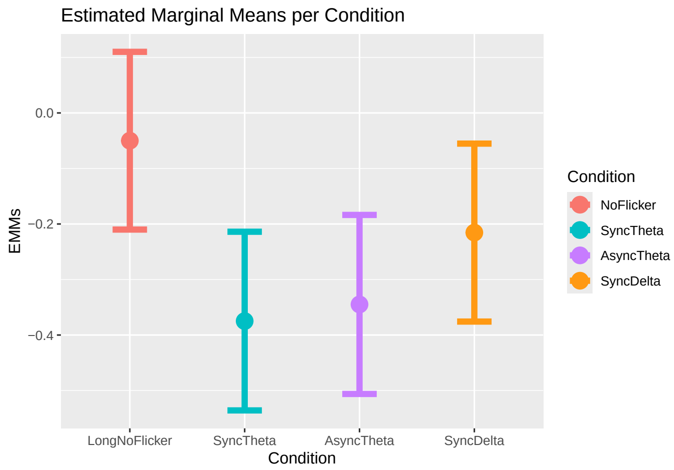
```r
(EffectSizes <- eff_size(Exp2.emm.s, sigma = sigma(Exp2.model), edf = df.residual(Exp2.model), method =
```

```
##  contrast               effect.size     SE  df asymp.LCL asymp.UCL
##  AsyncTheta - NoFlicker     -0.2949 0.0760 Inf   -0.4438   -0.1461
##  AsyncTheta - SyncDelta     -0.1294 0.0760 Inf   -0.2783    0.0195
##  AsyncTheta - SyncTheta      0.0299 0.0763 Inf   -0.1196    0.1795
##  NoFlicker - SyncDelta       0.1655 0.0754 Inf    0.0178    0.3133
##  NoFlicker - SyncTheta       0.3249 0.0758 Inf    0.1763    0.4734
##  SyncDelta - SyncTheta       0.1594 0.0758 Inf    0.0108    0.3079
##
## sigma used for effect sizes: 1
## Confidence level used: 0.95
```

```r
#Data plotting

plot(Exp2.emm.s,comparisons = TRUE) +
  ggtitle("Estimated Marginal Means")
```

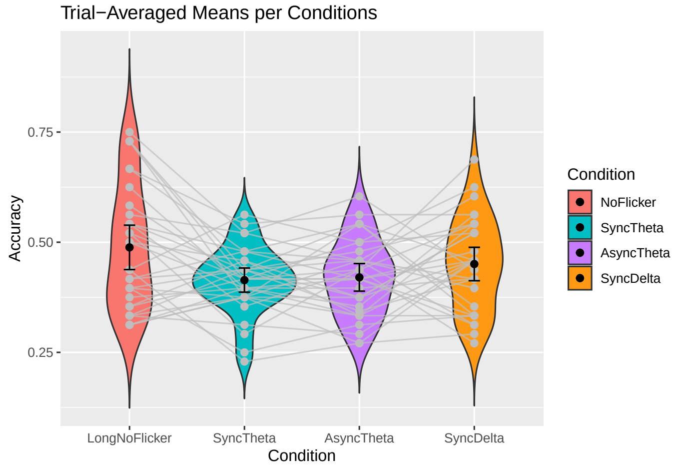



```r
EMFrame <- data.frame(Exp2.emm.s)
EMFrame$Condition = factor(EMFrame$Condition, levels = c("NoFlicker", "SyncTheta", "AsyncTheta", "SyncD

ggplot(EMFrame, aes(x = Condition, y = emmean, color = Condition)) +
  geom_point(size = 5) +
  geom_errorbar(aes(ymin = asymp.LCL, ymax = asymp.UCL), width = 0.3, linewidth = 2) +
  labs(title="Estimated Marginal Means per Condition", x = "Condition", y = "EMMs")+
  scale_color_manual(values = c("NoFlicker" = LongNoFlickerColour, "SyncTheta" = SyncThetaColour, "Asyn
  scale_x_discrete(labels=c("NoFlicker" = "LongNoFlicker"))
```




```r
# Calculate the mean accuracy for each subject
sub_means <- tapply(Exp2Data$Accuracy, Exp2Data$ParticipantID , mean)
# Calculate accuracy for each subject and condition
Exp2pSub_pCondition_acc <- aggregate(Accuracy ~ ParticipantID + Condition, data = Exp2Data, mean)
Exp2pSub_pCondition_acc$Condition = factor(Exp2pSub_pCondition_acc$Condition, levels = c("NoFlicker", "

ggplot(Exp2pSub_pCondition_acc, aes(x = Condition, y = Accuracy, fill = Condition)) +
  geom_violin(trim=FALSE)+
  geom_point(size = 2, color = "gray") +
  geom_line(aes(group = ParticipantID), color = "gray", alpha = .7) +
  labs(title="Trial-Averaged Means per Conditions", x = "Condition", y = "Accuracy") +
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
```

```r
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")+
  scale_fill_manual(values = c("NoFlicker" = LongNoFlickerColour, "SyncTheta" = SyncThetaColour, "Async
  scale_x_discrete(labels=c("NoFlicker" = "LongNoFlicker"))
```




```r
#Trial-averaged test with bootstrapping

# Calculate means
(mAll <- mean(Exp2pSub_pCondition_acc$Accuracy)*100)
```

```
## [1] 44.3278
```

```r
(sdAll <- sd((sub_means)*100))
```

```
## [1] 7.152448
```

```r
(mSyncTheta <- mean(Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition=="SyncTheta"])*10
```

```
## [1] 41.40625
```

```r
(mNoFlicker <- mean(Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition=="NoFlicker"])*10
```

```
## [1] 48.82812
```

```r
(mAsyncTheta <- mean(Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition=="AsyncTheta"])*
```

```
## [1] 42.02474
```

```r
(mSyncDelta <- mean(Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition=="SyncDelta"])*10
```

```
## [1] 45.05208
```

```r
SyncTheta_NoFlicker <- Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition=="SyncTheta"]
  Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition == "NoFlicker"]*100
SyncTheta_AsyncTheta <- Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition=="SyncTheta"
  Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition == "AsyncTheta"]*100
SyncTheta_SyncDelta <- Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition=="SyncTheta"]
  Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition == "SyncDelta"]*100

(SyncTheta_NoFlicker_out <- boot.t.test(SyncTheta_NoFlicker,
                                        alternative = c("two.sided", "less", "greater"),
                                        mu = 0, paired = FALSE, var.equal = FALSE,
                                        conf.level = 0.95, R = 10000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  SyncTheta_NoFlicker
## number of bootstrap samples:  10000
## bootstrap p-value = 0.004
## bootstrap mean of x (SE) = -7.405924 (2.35392)
## 95 percent bootstrap percentile confidence interval:
##  -12.109375  -2.799479
##
## Results without bootstrap:
## t = -3.0885, df = 31, p-value = 0.004221
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -12.323001  -2.520749
## sample estimates:
## mean of x
## -7.421875
```

```r
abs(mean(SyncTheta_NoFlicker)/sd(SyncTheta_NoFlicker)) # Cohen's d
```

```
## [1] 0.5459707
```

```r
(SyncTheta_AsyncTheta_out <- boot.t.test(SyncTheta_AsyncTheta,
                                         alternative = c("two.sided", "less", "greater"),
                                         mu = 0, paired = FALSE, var.equal = FALSE,
                                         conf.level = 0.95, R = 10000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  SyncTheta_AsyncTheta
## number of bootstrap samples:  10000
## bootstrap p-value = 0.614
## bootstrap mean of x (SE) = -0.6159049 (1.265976)
## 95 percent bootstrap percentile confidence interval:
##  -3.125000  1.855469
##
## Results without bootstrap:
```

```
## t = -0.47876, df = 31, p-value = 0.6355
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -3.253223  2.016244
## sample estimates:
##  mean of x
## -0.6184896
```

```r
abs(mean(SyncTheta_AsyncTheta)/sd(SyncTheta_AsyncTheta)) # Cohen's d
```

```
## [1] 0.08463446
```

```r
(SyncTheta_SyncDelta_out <- boot.t.test(SyncTheta_SyncDelta,
                                        alternative = c("two.sided", "less", "greater"),
                                        mu = 0, paired = FALSE, var.equal = FALSE,
                                        conf.level = 0.95, R = 10000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  SyncTheta_SyncDelta
## number of bootstrap samples:  10000
## bootstrap p-value = 0.0824
## bootstrap mean of x (SE) = -3.621315 (2.073809)
## 95 percent bootstrap percentile confidence interval:
##  -7.7473958  0.4557292
##
## Results without bootstrap:
## t = -1.7174, df = 31, p-value = 0.09588
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -7.975437  0.683770
## sample estimates:
## mean of x
## -3.645833
```

```r
abs(mean(SyncTheta_SyncDelta)/sd(SyncTheta_SyncDelta)) # Cohen's d
```

```
## [1] 0.303599
```

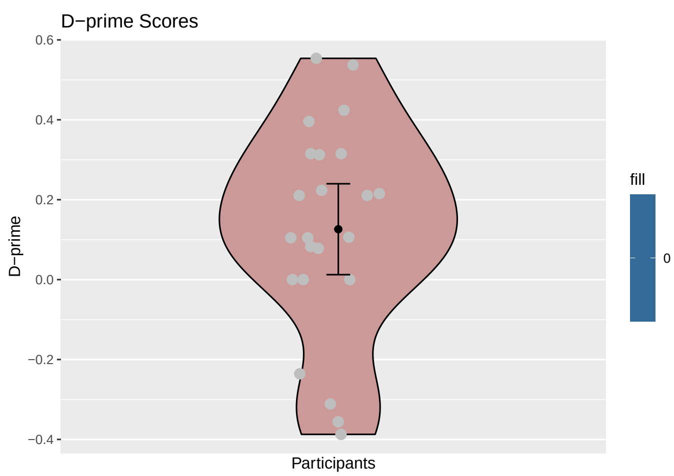
```r
#Synchrony Discrimination Task

# Arrange long-format data

# List of subject names
subjects <- c("Sub_01", "Sub_02", "Sub_03", "Sub_04", "Sub_05", "Sub_06",
              "Sub_09", "Sub_10", "Sub_11", "Sub_14", "Sub_15", "Sub_17",
              "Sub_19", "Sub_20", "Sub_22", "Sub_23", "Sub_24", "Sub_25",
              "Sub_27", "Sub_29", "Sub_30", "Sub_31","Sub_32")
nSub <- length(subjects)
```

```r
# Initialize vectors to store results
nHits_pSub <- rep(NA, nSub)
HitRate_pSub <- rep(NA, nSub)
nFAs_pSub <- rep(NA, nSub)
FARate_pSub <- rep(NA, nSub)
Exp2Dprime_pSub <- rep(NA, nSub)
Exp2Sync_subnumsnums <- rep(NA, nSub)

# Loop over subjects
for (iSub in 1:nSub) {

  # List all files in the current directory that match the pattern
  file_name <- list.files(path = Exp2SyncDataFolder, pattern = subjects[iSub])

  Exp2SyncData <- read.csv(file.path(Exp2SyncDataFolder, file_name), header = TRUE)

  # Calculate Hit Rate
  nHits_pSub[iSub] <- sum(Exp2SyncData$SyncRating == 1 & Exp2SyncData$Accuracy == 1)
  HitRate_pSub[iSub] <- nHits_pSub[iSub] / sum(Exp2SyncData$SyncRating == 1)

  # Calculate False Alarm Rate
  nFAs_pSub[iSub] <- sum(Exp2SyncData$SyncRating == 2 & Exp2SyncData$Accuracy == 0)
  FARate_pSub[iSub] <- nFAs_pSub[iSub] / sum(Exp2SyncData$SyncRating == 2)

  # Calculate d-prime
  Exp2Dprime_pSub[iSub] <- qnorm(HitRate_pSub[iSub]) - qnorm(FARate_pSub[iSub])
  Exp2Sync_subnumsnums[iSub] <- Exp2SyncData$ParticipantID[iSub]
}

# Create a data frame
Exp2DprimeData <- data.frame(ParticipantID = as.factor(Exp2Sync_subnumsnums), Dprime = Exp2Dprime_pSub)

# significance test
(DPrime_out <- boot.t.test(Exp2DprimeData$Dprime, alternative = c("two.sided", "less", "greater"),
                          mu = 0, paired = FALSE, var.equal = FALSE, conf.level = 0.95,
                          R = 10000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  Exp2DprimeData$Dprime
## number of bootstrap samples:  10000
## bootstrap p-value = 0.0518
## bootstrap mean of x (SE) = 0.1259253 (0.05308503)
## 95 percent bootstrap percentile confidence interval:
##  0.01939247 0.22896076
##
## Results without bootstrap:
## t = 2.2992, df = 22, p-value = 0.03136
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.01236129 0.23989307
## sample estimates:
```

```
## mean of x
## 0.1261272
```

```
# Bayes analysis for sync vs async theta
bf = ttestBF(Exp2DprimeData$Dprime, paired = FALSE)

# Create a violin plot
ggplot(data = Exp2DprimeData, aes(x = 0, y = Dprime, fill = 0)) +
  geom_violin(width=1, fill = SyncDiscColour, color = "black") +
  scale_x_discrete() +
  geom_point(size = 3, color = "gray",position = position_jitterdodge()) +
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black") +
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black") +
  labs(title = "D-prime Scores",
       x = "Participants",
       y = "D-prime")
```



Exploratory Analysis on Exp 2

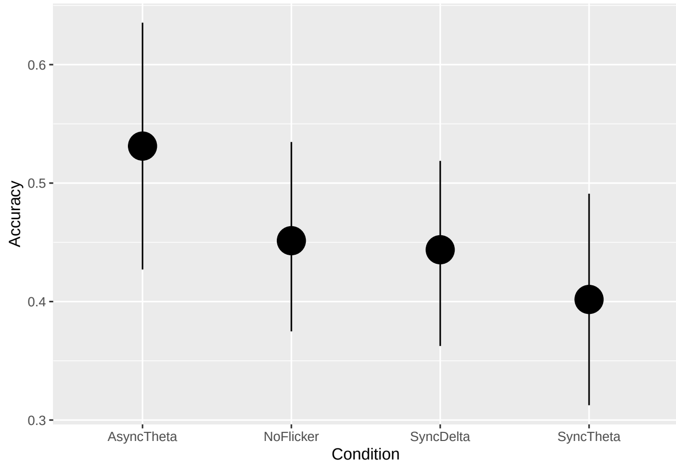
```
# filter data to keep only first blocks
FirstBlockOnly <- Exp2Data[Exp2Data$BlockNumber == 1, ]

FirstBlockOnly.model = glmer(Accuracy ~ Condition + (1|ClipID) + (1|ParticipantID),
                   data=FirstBlockOnly, family = binomial)

# summary and plotting
Anova(FirstBlockOnly.model)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##            Chisq Df Pr(>Chisq)
## Condition 2.8085  3     0.4221
```

```r
summary(FirstBlockOnly.model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition + (1 | ClipID) + (1 | ParticipantID)
##    Data: FirstBlockOnly
##
##      AIC      BIC   logLik deviance df.resid
##    707.8    733.2   -347.9    695.8      506
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.2000 -0.8317 -0.6560  0.9613  1.4530
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.3028   0.5502
##  ParticipantID (Intercept) 0.1340   0.3661
## Number of obs: 512, groups:  ClipID, 176; ParticipantID, 32
##
## Fixed effects:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)           0.1313     0.2690   0.488   0.6254
## ConditionNoFlicker   -0.3546     0.3442  -1.030   0.3029
## ConditionSyncDelta   -0.4053     0.3392  -1.195   0.2322
## ConditionSyncTheta   -0.6051     0.3664  -1.652   0.0986 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CndtNF CndtSD
## CndtnNFlckr -0.766
## CndtnSyncDl -0.776  0.606
## CndtnSyncTh -0.720  0.571  0.584
```

```r
ggplot(FirstBlockOnly,aes(x=Condition,y=Accuracy))+
  stat_summary(fun.data=mean_cl_boot,size=2)
```




```r
FirstBlockOnly.emm.s <- emmeans(FirstBlockOnly.model,"Condition")
#pairs(Exp2.emm.s)

# Conditions within Exp2.emm.s ordered: AsyncTheta, NoFlicker, SyncDelta, SyncTheta
# Planned comparisons for correction .05/3
(out1 <- contrast(FirstBlockOnly.emm.s, list(SyncTheta.vs.NoFlicker  = c(0, -1, 0, 1))))
```

```
##  contrast               estimate   SE  df z.ratio p.value
##  SyncTheta.vs.NoFlicker    -0.25 0.33 Inf  -0.760  0.4472
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out2 <- contrast(FirstBlockOnly.emm.s, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

```
##  contrast                estimate    SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta   -0.605 0.366 Inf  -1.652  0.0986
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out3 <- contrast(FirstBlockOnly.emm.s, list(SyncTheta.vs.SyncDelta  = c(0, 0, -1, 1))))
```

```
##  contrast               estimate    SE  df z.ratio p.value
##  SyncTheta.vs.SyncDelta     -0.2 0.323 Inf  -0.619  0.5359
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(EffectSizes <- eff_size(FirstBlockOnly.emm.s, sigma = sigma(FirstBlockOnly.model),
                         edf = df.residual(FirstBlockOnly.model), method = "pairwise"))
```

```
##  contrast               effect.size    SE  df asymp.LCL asymp.UCL
##  AsyncTheta - NoFlicker      0.3546 0.344 Inf    -0.320     1.029
##  AsyncTheta - SyncDelta      0.4053 0.339 Inf    -0.260     1.070
##  AsyncTheta - SyncTheta      0.6051 0.367 Inf    -0.113     1.324
##  NoFlicker - SyncDelta       0.0506 0.303 Inf    -0.544     0.645
##  NoFlicker - SyncTheta       0.2505 0.330 Inf    -0.396     0.897
```

```
##  SyncDelta - SyncTheta       0.1999 0.323 Inf    -0.433     0.833
##
## sigma used for effect sizes: 1
## Confidence level used: 0.95
```

```r
# Block and Condition interaction model
Exp2.modelBlock = glmer(Accuracy ~ Condition*poly(BlockNumber,2) + (1|ClipID) + (1|ParticipantID),
                   data=Exp2Data, family = binomial)

Anova(Exp2.modelBlock)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##                                  Chisq Df Pr(>Chisq)
## Condition                      24.2145  3  2.253e-05 ***
## poly(BlockNumber, 2)           14.9309  2  0.0005725 ***
## Condition:poly(BlockNumber, 2)  1.8068  6  0.9365835
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(Exp2.modelBlock)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition * poly(BlockNumber, 2) + (1 | ClipID) +
##     (1 | ParticipantID)
##    Data: Exp2Data
##
##      AIC      BIC   logLik deviance df.resid
##   8178.5   8272.6  -4075.3   8150.5     6114
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.1459 -0.8447 -0.6001  0.9861  2.2300
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.29839  0.5462
##  ParticipantID (Intercept) 0.07378  0.2716
## Number of obs: 6128, groups:  ClipID, 192; ParticipantID, 32
##
## Fixed effects:
##                                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)                             -0.34873    0.08248  -4.228 2.36e-05
## ConditionNoFlicker                       0.30558    0.07615   4.013 6.00e-05
## ConditionSyncDelta                       0.13169    0.07607   1.731   0.0834
## ConditionSyncTheta                      -0.02837    0.07645  -0.371   0.7105
## poly(BlockNumber, 2)1                   -3.87155    4.36971  -0.886   0.3756
## poly(BlockNumber, 2)2                   -4.18864    4.24494  -0.987   0.3238
## ConditionNoFlicker:poly(BlockNumber, 2)1 -2.73144    5.85529  -0.466   0.6409
## ConditionSyncDelta:poly(BlockNumber, 2)1  3.11157    6.18643   0.503   0.6150
```

```
## ConditionSyncTheta:poly(BlockNumber, 2)1 -0.33108    6.08055  -0.054   0.9566
## ConditionNoFlicker:poly(BlockNumber, 2)2 -3.93506    5.89411  -0.668   0.5044
## ConditionSyncDelta:poly(BlockNumber, 2)2 -4.84423    6.06601  -0.799   0.4245
## ConditionSyncTheta:poly(BlockNumber, 2)2 -3.35689    6.28897  -0.534   0.5935
##
## (Intercept)                              ***
## ConditionNoFlicker                       ***
## ConditionSyncDelta                       .
## ConditionSyncTheta
## poly(BlockNumber, 2)1
## poly(BlockNumber, 2)2
## ConditionNoFlicker:poly(BlockNumber, 2)1
## ConditionSyncDelta:poly(BlockNumber, 2)1
## ConditionSyncTheta:poly(BlockNumber, 2)1
## ConditionNoFlicker:poly(BlockNumber, 2)2
## ConditionSyncDelta:poly(BlockNumber, 2)2
## ConditionSyncTheta:poly(BlockNumber, 2)2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CndtNF CndtSD CndtST p(BN,2)1 p(BN,2)2 CNF:(BN,2)1
## CndtnNFlckr -0.468
## CndtnSyncDl -0.468  0.507
## CndtnSyncTh -0.465  0.504  0.505
## ply(BlN,2)1  0.012 -0.013 -0.013 -0.013
## ply(BlN,2)2  0.033 -0.036 -0.036 -0.035 -0.079
## CNF:(BN,2)1 -0.008  0.009  0.009  0.009 -0.725    0.058
## CSD:(BN,2)1 -0.009  0.009  0.009  0.009 -0.720    0.053    0.528
## CST:(BN,2)1 -0.008  0.009  0.009  0.013 -0.710    0.060    0.517
## CNF:(BN,2)2 -0.025 -0.003  0.026  0.026  0.052   -0.717   -0.050
## CSD:(BN,2)2 -0.021  0.024  0.032  0.023  0.063   -0.704   -0.041
## CST:(BN,2)2 -0.022  0.024  0.023  0.038  0.052   -0.679   -0.038
##             CSD:(BN,2)1 CST:(BN,2)1 CNF:(BN,2)2 CSD:(BN,2)2
## CndtnNFlckr
## CndtnSyncDl
## CndtnSyncTh
## ply(BlN,2)1
## ply(BlN,2)2
## CNF:(BN,2)1
## CSD:(BN,2)1
## CST:(BN,2)1  0.511
## CNF:(BN,2)2 -0.034      -0.042
## CSD:(BN,2)2 -0.006      -0.048       0.493
## CST:(BN,2)2 -0.037      -0.015       0.482       0.471
## optimizer (Nelder_Mead) convergence code: 0 (OK)
## Model failed to converge with max|grad| = 0.00262627 (tol = 0.002, component 1)
```

```r
# Calculate the mean accuracy for each subjects' theta conditions
ThetaExp2Data <- Exp2Data[Exp2Data$Condition == "SyncTheta" | Exp2Data$Condition == "AsyncTheta",]
ThetaMean_Accs <- tapply(ThetaExp2Data$Accuracy, ThetaExp2Data$ParticipantID , mean)

for (iSub in Subs) {
```

```r
  NullDist <- data.frame(Acc=rep(NA,nIterations))

  for (i in 1:nIterations) {
    # Generate 192 random responses between 1 and 4
    Responses <- Exp2Data[Exp2Data$ParticipantID == iSub, "Response"]

    # Generate 192 random answers between 1 and 4
    Answers <- sample(Exp2Data[Exp2Data$ParticipantID == iSub, "CorrectKey"])

    NullDist[i,1] <- ((sum(Responses == Answers))/length(Responses))*100
  }

  # Sort distribution
  SortedNullDist <- NullDist[order(NullDist$Acc),]
  Significance_Threshold <- SortedNullDist[length(SortedNullDist)-nIterations*.05]

  if ((ThetaMean_Accs[as.character(iSub)])*100 <= Significance_Threshold) {
    print(c(iSub,' is at chance-level!!!'))
  }

}
```

```
## [1] "2"                      " is at chance-level!!!"
## [1] "12"                     " is at chance-level!!!"
## [1] "14"                     " is at chance-level!!!"
```

```r
excludedExp2Data <- Exp2Data[!(Exp2Data$ParticipantID %in% c(2, 12, 14)),]

excludedExp2.model = glmer(Accuracy ~ Condition + (1|ClipID) + (1|ParticipantID),
                   data=excludedExp2Data, family = binomial)

Anova(excludedExp2.model)
```

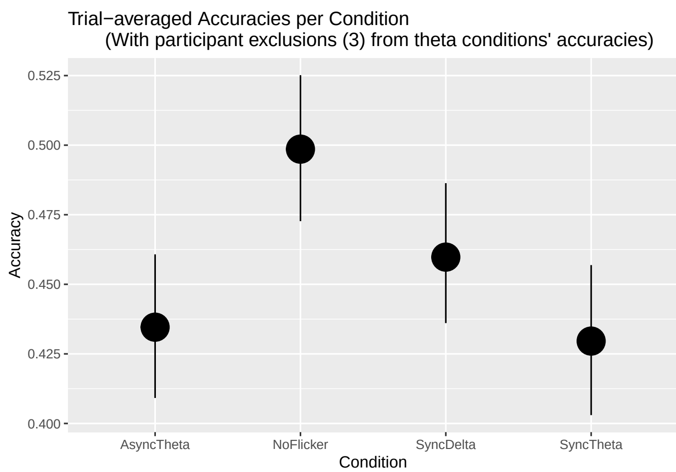
```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##            Chisq Df Pr(>Chisq)
## Condition 16.908  3  0.0007382 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(excludedExp2.model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition + (1 | ClipID) + (1 | ParticipantID)
##    Data: excludedExp2Data
##
##      AIC      BIC   logLik deviance df.resid
##   7465.7   7505.5  -3726.9   7453.7     5546
##
```

```
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.9746 -0.8651 -0.6134  0.9813  2.1851
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.29644  0.5445
##  ParticipantID (Intercept) 0.05097  0.2258
## Number of obs: 5552, groups:  ClipID, 192; ParticipantID, 29
##
## Fixed effects:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -0.28716    0.08066  -3.560 0.000370 ***
## ConditionNoFlicker   0.27455    0.07943   3.457 0.000547 ***
## ConditionSyncDelta   0.11680    0.07957   1.468 0.142128
## ConditionSyncTheta -0.01165    0.07983  -0.146 0.883932
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CndtNF CndtSD
## CndtnNFlckr -0.499
## CndtnSyncDl -0.499  0.505
## CndtnSyncTh -0.497  0.504  0.505
```

```
ggplot(excludedExp2Data,aes(x=Condition, y=Accuracy))+
  stat_summary(fun.data=mean_cl_boot, size=2)+
  labs(title = "Trial-averaged Accuracies per Condition
      (With participant exclusions (3) from theta conditions' accuracies)")
```




```
excludedExp2.emm.s <- emmeans(excludedExp2.model,"Condition")

# Conditions within Exp2.emm.s ordered: AsyncTheta, NoFlicker, SyncDelta, SyncTheta
# Planned comparisons for correction .05/3
(out1 <- contrast(excludedExp2.emm.s, list(SyncTheta.vs.NoFlicker  = c(0, -1, 0, 1))))
```

```
##  contrast                 estimate     SE  df z.ratio p.value
```

```
##  SyncTheta.vs.NoFlicker   -0.286 0.0793 Inf  -3.607  0.0003
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out2 <- contrast(excludedExp2.emm.s, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

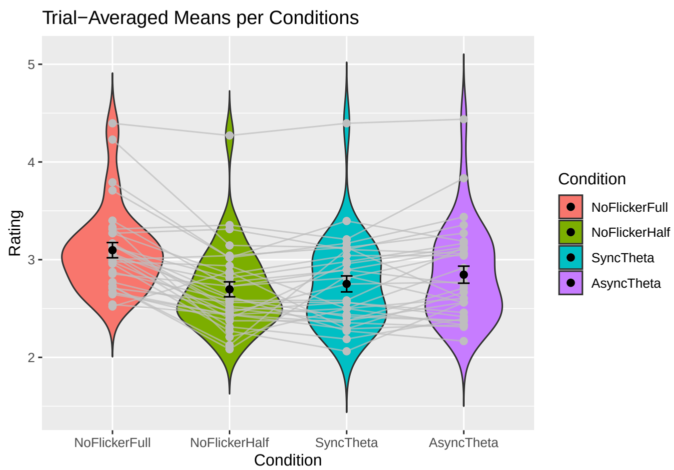
```
##  contrast                estimate     SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta  -0.0117 0.0798 Inf  -0.146  0.8839
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out3 <- contrast(excludedExp2.emm.s, list(SyncTheta.vs.SyncDelta  = c(0, 0, -1, 1))))
```

```
##  contrast               estimate     SE  df z.ratio p.value
##  SyncTheta.vs.SyncDelta   -0.128 0.0793 Inf  -1.621  0.1051
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
# Rating as random intercept----------------------

# Calculate the mean accuracy for each subject
sub_Ratingmeans <- tapply(Exp2Data$Rating, Exp2Data$ParticipantID , mean)
# Calculate accuracy for each subject and condition
pSub_pCondition_rating <- aggregate(Rating ~ ParticipantID + Condition, data = Exp2Data, mean)
pSub_pCondition_rating$Condition = factor(pSub_pCondition_rating$Condition, levels = c("SyncTheta", "NoF

ggplot(pSub_pCondition_rating, aes(x = Condition, y = Rating, fill = Condition)) +
  geom_violin(trim=FALSE)+
  geom_point(size = 2, color = "gray") +
  geom_line(aes(group = ParticipantID), color = "gray", alpha = .7) +
  labs(title="Trial-Averaged Means per Conditions", x = "Condition", y = "Rating") +
  stat_summary(fun.pSub_pCondition_rating = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "
  stat_summary(fun.pSub_pCondition_rating = "mean", geom = "point", size = 2, color = "black")
```

```
## No summary function supplied, defaulting to ‘mean_se()‘
## No summary function supplied, defaulting to ‘mean_se()‘
```



```r
# cor(Exp2Data$Rating, Exp2Data$Accuracy)
#
# ggplot(Exp2Data, aes(x=Rating, y=Accuracy)) +
#   geom_point()+
#   geom_smooth(method=lm)

# Model with Rating as random effect
Exp2.modelofRating = glmer(Rating ~ Condition + (1|ClipID) + (1|ParticipantID),
                   data=Exp2Data)
Anova(Exp2.modelofRating)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Rating
##            Chisq Df Pr(>Chisq)
## Condition 28.766  3  2.508e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(Exp2.modelofRating)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Rating ~ Condition + (1 | ClipID) + (1 | ParticipantID)
##    Data: Exp2Data
##
## REML criterion at convergence: 19424.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.4429 -0.7488  0.0055  0.7295  2.8936
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.4279   0.6542
##  ParticipantID (Intercept) 0.1345   0.3667
##  Residual                  1.2672   1.1257
## Number of obs: 6128, groups:  ClipID, 192; ParticipantID, 32
##
## Fixed effects:
##                       Estimate Std. Error t value
## (Intercept)           2.879196   0.085238  33.778
## ConditionNoFlicker    0.170935   0.040739   4.196
## ConditionSyncDelta  -0.003544   0.040739  -0.087
## ConditionSyncTheta  -0.015914   0.040739  -0.391
##
## Correlation of Fixed Effects:
##             (Intr) CndtNF CndtSD
## CndtnNFlckr -0.240
## CndtnSyncDl -0.240  0.503
## CndtnSyncTh -0.240  0.503  0.503
```

```r
Exp2.emm.sofRating <- emmeans(Exp2.modelofRating,"Condition")
```

```
## Note: D.f. calculations have been disabled because the number of observations exceeds 3000.
## To enable adjustments, add the argument 'pbkrtest.limit = 6128' (or larger)
## [or, globally, 'set emm_options(pbkrtest.limit = 6128)' or larger];
## but be warned that this may result in large computation time and memory use.
## Note: D.f. calculations have been disabled because the number of observations exceeds 3000.
## To enable adjustments, add the argument 'lmerTest.limit = 6128' (or larger)
## [or, globally, 'set emm_options(lmerTest.limit = 6128)' or larger];
## but be warned that this may result in large computation time and memory use.
```

```r
#pairs(Exp2.emm.s)

# Conditions within Exp2.emm.s ordered: AsyncTheta, NoFlicker, SyncDelta, SyncTheta
# Planned comparisons for correction .05/3
(out1 <- contrast(Exp2.emm.sofRating, list(SyncTheta.vs.NoFlicker  = c(0, -1, 0, 1))))
```

```
##  contrast               estimate     SE  df z.ratio p.value
##  SyncTheta.vs.NoFlicker   -0.187 0.0406 Inf  -4.600  <.0001
##
## Degrees-of-freedom method: asymptotic
```

```r
(out2 <- contrast(Exp2.emm.sofRating, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

```
##  contrast                estimate     SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta  -0.0159 0.0407 Inf  -0.391  0.6961
##
## Degrees-of-freedom method: asymptotic
```

```r
(out3 <- contrast(Exp2.emm.sofRating, list(SyncTheta.vs.SyncDelta  = c(0, 0, -1, 1))))
```

```
##  contrast               estimate     SE  df z.ratio p.value
##  SyncTheta.vs.SyncDelta  -0.0124 0.0406 Inf  -0.305  0.7607
##
## Degrees-of-freedom method: asymptotic
```

```r
(EffectSizes <- eff_size(Exp2.emm.sofRating, sigma = sigma(Exp2.modelofRating),
                         edf = df.residual(Exp2.modelofRating), method = "pairwise"))
```

```
##  contrast               effect.size     SE  df asymp.LCL asymp.UCL
##  AsyncTheta - NoFlicker    -0.15185 0.0362 Inf   -0.2228   -0.0809
##  AsyncTheta - SyncDelta     0.00315 0.0362 Inf   -0.0678    0.0741
##  AsyncTheta - SyncTheta     0.01414 0.0362 Inf   -0.0568    0.0851
##  NoFlicker - SyncDelta      0.15500 0.0361 Inf    0.0842    0.2258
##  NoFlicker - SyncTheta      0.16599 0.0361 Inf    0.0952    0.2368
##  SyncDelta - SyncTheta      0.01099 0.0361 Inf   -0.0597    0.0817
##
## sigma used for effect sizes: 1.126
## Degrees-of-freedom method: inherited from asymptotic when re-gridding
## Confidence level used: 0.95
```

```r
Exp2Data$RatingFactor = as.factor(Exp2Data$Rating)

# Model with Rating as random effect
Exp2.modelwRating = glmer(Accuracy ~ Condition*poly(RatingFactor,2) + (1|ClipID) + (1|ParticipantID),
                   data=Exp2Data, family = binomial)

Anova(Exp2.modelwRating)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##                                 Chisq Df Pr(>Chisq)
## Condition                      16.213  3   0.001025 **
## poly(RatingFactor, 2)          85.670  2  < 2.2e-16 ***
## Condition:poly(RatingFactor, 2) 11.763  6   0.067459 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(Exp2.modelwRating)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition * poly(RatingFactor, 2) + (1 | ClipID) +
##     (1 | ParticipantID)
##    Data: Exp2Data
##
##      AIC      BIC   logLik deviance df.resid
##   8098.5   8192.6  -4035.2   8070.5     6114
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.0934 -0.8292 -0.5822  0.9664  2.4200
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.27128  0.5208
##  ParticipantID (Intercept) 0.07868  0.2805
## Number of obs: 6128, groups:  ClipID, 192; ParticipantID, 32
##
## Fixed effects:
##                                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)                           -0.33184    0.08260  -4.018 5.88e-05
## ConditionNoFlicker                     0.25245    0.07654   3.298 0.000973
## ConditionSyncDelta                     0.12024    0.07652   1.571 0.116127
## ConditionSyncTheta                    -0.02482    0.07685  -0.323 0.746696
## poly(RatingFactor, 2)1                 9.74210    4.11621   2.367 0.017944
## poly(RatingFactor, 2)2                12.12605    4.09997   2.958 0.003100
## ConditionNoFlicker:poly(RatingFactor, 2)1  9.03930    5.55950   1.626 0.103966
## ConditionSyncDelta:poly(RatingFactor, 2)1 15.50012    5.67552   2.731 0.006313
## ConditionSyncTheta:poly(RatingFactor, 2)1 13.61255    5.60752   2.428 0.015201
## ConditionNoFlicker:poly(RatingFactor, 2)2 -6.01863    5.67092  -1.061 0.288547
```

```
## ConditionSyncDelta:poly(RatingFactor, 2)2 -2.38447    5.70661  -0.418 0.676061
## ConditionSyncTheta:poly(RatingFactor, 2)2  2.99610    5.87047   0.510 0.609794
##
## (Intercept)                               ***
## ConditionNoFlicker                        ***
## ConditionSyncDelta
## ConditionSyncTheta
## poly(RatingFactor, 2)1                    *
## poly(RatingFactor, 2)2                    **
## ConditionNoFlicker:poly(RatingFactor, 2)1
## ConditionSyncDelta:poly(RatingFactor, 2)1 **
## ConditionSyncTheta:poly(RatingFactor, 2)1 *
## ConditionNoFlicker:poly(RatingFactor, 2)2
## ConditionSyncDelta:poly(RatingFactor, 2)2
## ConditionSyncTheta:poly(RatingFactor, 2)2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CndtNF CndtSD CndtST p(RF,2)1 p(RF,2)2 CNF:(RF,2)1
## CndtnNFlckr -0.467
## CndtnSyncDl -0.466  0.503
## CndtnSyncTh -0.463  0.500  0.500
## ply(RtF,2)1  0.014 -0.022 -0.014 -0.013
## ply(RtF,2)2  0.033 -0.039 -0.037 -0.034  0.037
## CNF:(RF,2)1 -0.009 -0.031  0.010  0.010 -0.668   -0.025
## CSD:(RF,2)1 -0.010  0.010  0.021  0.009 -0.656   -0.021    0.480
## CST:(RF,2)1 -0.009  0.010  0.010  0.019 -0.626   -0.036    0.457
## CNF:(RF,2)2 -0.022 -0.011  0.025  0.024 -0.025   -0.677   -0.016
## CSD:(RF,2)2 -0.023  0.026  0.012  0.023 -0.026   -0.676    0.016
## CST:(RF,2)2 -0.023  0.026  0.025  0.052 -0.032   -0.666    0.027
##             CSD:(RF,2)1 CST:(RF,2)1 CNF:(RF,2)2 CSD:(RF,2)2
## CndtnNFlckr
## CndtnSyncDl
## CndtnSyncTh
## ply(RtF,2)1
## ply(RtF,2)2
## CNF:(RF,2)1
## CSD:(RF,2)1
## CST:(RF,2)1  0.442
## CNF:(RF,2)2  0.016       0.026
## CSD:(RF,2)2  0.014       0.031       0.464
## CST:(RF,2)2  0.013       0.060       0.460       0.457
```

```
Exp2.emm.swRating <- emmeans(Exp2.modelwRating,"Condition")
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
# Conditions within Exp2.emm.s ordered: AsyncTheta, NoFlicker, SyncDelta, SyncTheta
# Planned comparisons for correction .05/3
(out1 <- contrast(Exp2.emm.swRating, list(SyncTheta.vs.NoFlicker  = c(0, -1, 0, 1))))
```

```
##  contrast               estimate    SE  df z.ratio p.value
```

```
##  SyncTheta.vs.NoFlicker   -0.411 0.118 Inf  -3.488  0.0005
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out2 <- contrast(Exp2.emm.swRating, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

```
##  contrast                estimate    SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta  -0.0692 0.113 Inf  -0.612  0.5404
##
## Results are given on the log odds ratio (not the response) scale.
```

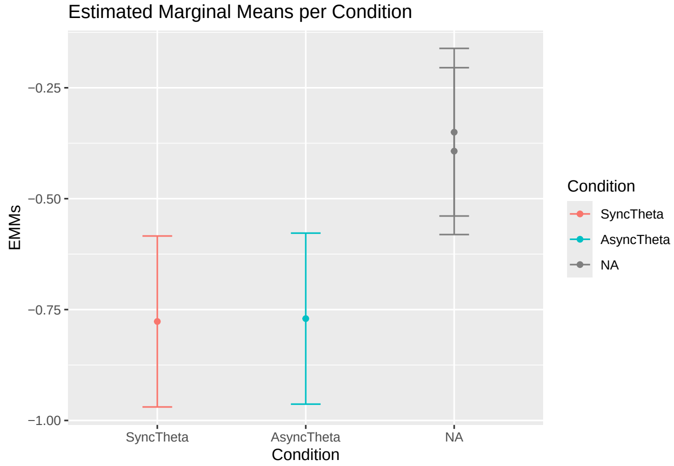
```r
(out3 <- contrast(Exp2.emm.swRating, list(SyncTheta.vs.SyncDelta  = c(0, 0, -1, 1))))
```

```
##  contrast               estimate    SE  df z.ratio p.value
##  SyncTheta.vs.SyncDelta   -0.225 0.117 Inf  -1.923  0.0545
##
## Results are given on the log odds ratio (not the response) scale.
```

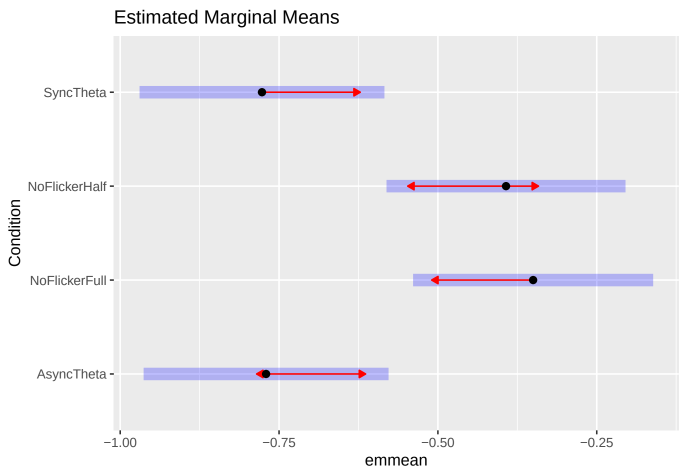
```r
(EffectSizes <- eff_size(Exp2.emm.swRating, sigma = sigma(Exp2.modelwRating),
                        edf = df.residual(Exp2.modelwRating), method = "pairwise"))
```

```
##  contrast               effect.size    SE  df asymp.LCL asymp.UCL
##  AsyncTheta - NoFlicker     -0.3416 0.114 Inf  -0.56559   -0.1176
##  AsyncTheta - SyncDelta     -0.1556 0.113 Inf  -0.37771    0.0666
##  AsyncTheta - SyncTheta      0.0692 0.113 Inf  -0.15231    0.2907
##  NoFlicker - SyncDelta       0.1860 0.119 Inf  -0.04652    0.4186
##  NoFlicker - SyncTheta       0.4108 0.118 Inf   0.17981    0.6417
##  SyncDelta - SyncTheta       0.2247 0.117 Inf  -0.00438    0.4539
##
## sigma used for effect sizes: 1
## Confidence level used: 0.95
```

```r
RatingEMFrame <- data.frame(Exp2.emm.swRating)
RatingEMFrame$Condition = factor(RatingEMFrame$Condition, levels = c("SyncTheta", "NoFlicker", "AsyncThe

ggplot(RatingEMFrame, aes(x = Condition, y = emmean, color = Condition)) +
  geom_point(size = 5) +
  geom_errorbar(aes(ymin = asymp.LCL, ymax = asymp.UCL), width = 0.3, linewidth = 2) +
  labs(title="Estimated Marginal Means per Condition", x = "Condition", y = "EMMs")
```




```r
plot(Exp2.emm.swRating, comparisons = TRUE) +
  ggtitle("Estimated Marginal Means")
```




```r
# Dprime & Accuracy correlation

# select the participants with Dprime
Exp2DatawDprime <- Exp2pSub_pCondition_acc[Exp2pSub_pCondition_acc$ParticipantID %in% c(Exp2Sync_subnums

Theta_means <- aggregate(Accuracy ~ Condition*ParticipantID, data = Exp2DatawDprime, FUN = mean)
STheta_means <- Theta_means[Theta_means == "SyncTheta",]
ATheta_means <- Theta_means[Theta_means == "AsyncTheta",]
Exp2TIMEEffect <- STheta_means$Accuracy - ATheta_means$Accuracy

cor(Exp2DprimeData$Dprime, Exp2TIMEEffect, method = "pearson")
```

```
## [1] 0.2948399
```

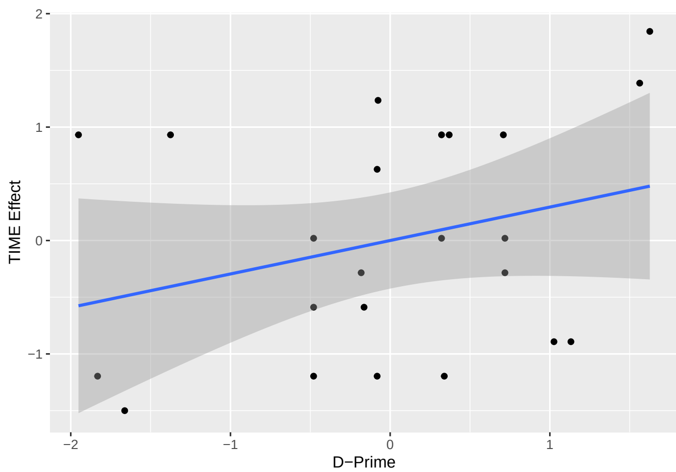
```r
cor.test(Exp2DprimeData$Dprime, Exp2TIMEEffect, method = "pearson")
```

```
##
```

```
##  Pearson's product-moment correlation
##
## data:  Exp2DprimeData$Dprime and Exp2TIMEEffect
## t = 1.414, df = 21, p-value = 0.172
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1335990  0.6304243
## sample estimates:
##       cor
## 0.2948399
```

```
DprimewTIME <- data.frame(DPrime = Exp2DprimeData$Dprime, Exp2TIMEEffect)

ggplot(DprimewTIME, aes(x=scale(DPrime), y=scale(Exp2TIMEEffect))) +
  geom_point()+
  geom_smooth(method=lm)+
  labs(x = "D-Prime", y = "TIME Effect")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



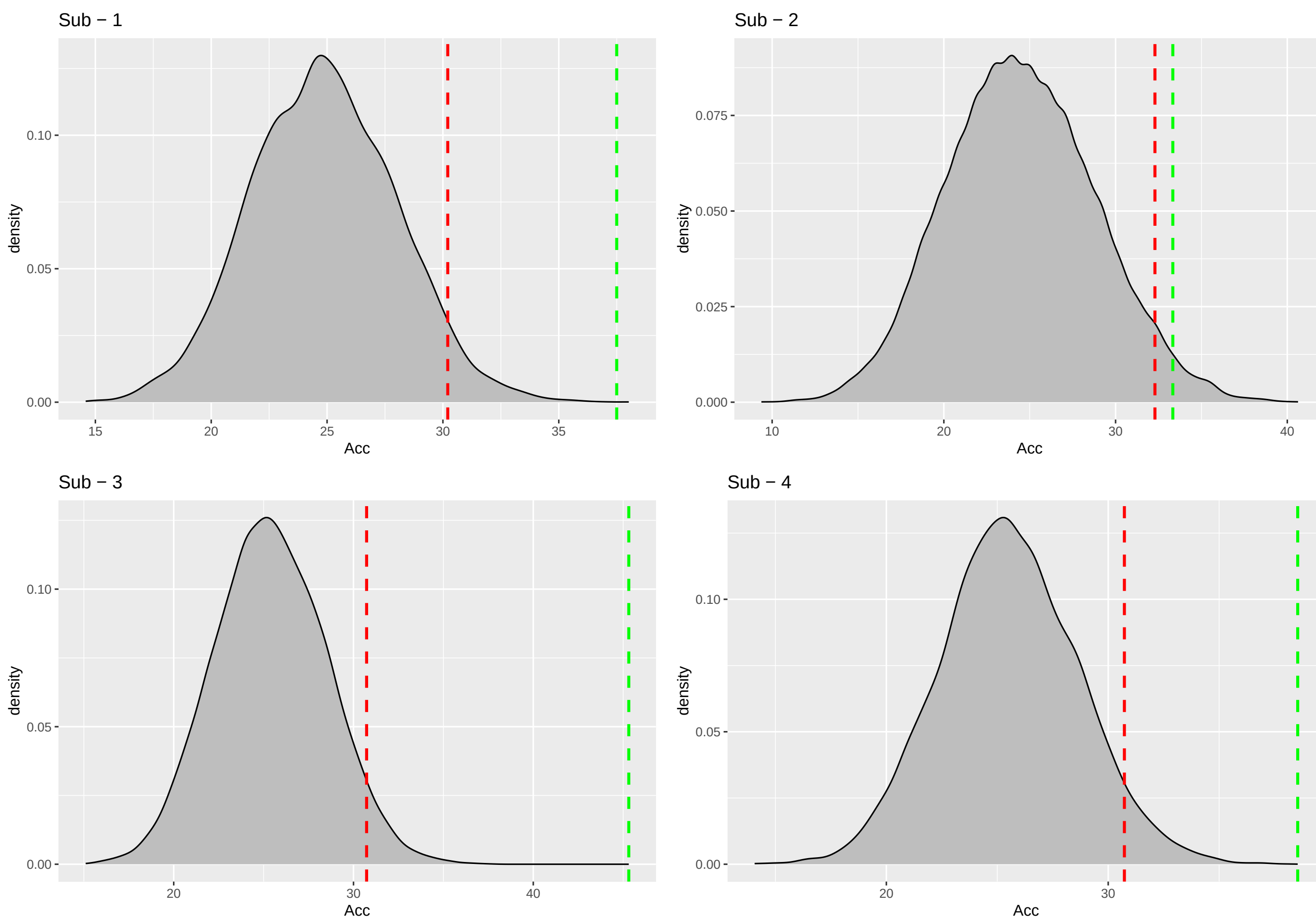

```
# Set the seed for reproducibility
set.seed(13021996)

# List of subject names
subjects <- c("Sub_01", "Sub_02", "Sub_03", "Sub_04", "Sub_05", "Sub_06", "Sub_07",
              "Sub_08", "Sub_09", "Sub_10", "Sub_11", "Sub_12", "Sub_13", "Sub_14",
              "Sub_15", "Sub_16", "Sub_17", "Sub_18", "Sub_19", "Sub_20", "Sub_21",
              "Sub_22", "Sub_23", "Sub_24", "Sub_25", "Sub_26", "Sub_27", "Sub_28",
              "Sub_29", "Sub_30", "Sub_31", "Sub_32", "Sub_33", "Sub_34", "Sub_35",
              "Sub_36", "Sub_37", "Sub_38")

# Initialize data frame
Exp3Data <- data.frame()

# Loop through subjects to fetch the data
for (iSub in 1:length(subjects)) {
  # Define file name
```

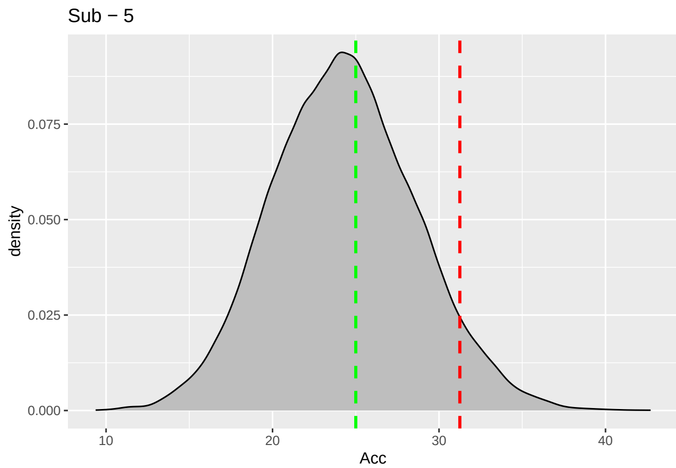
```r
  File_pattern <- list.files(path = Exp3DataFolder, pattern = subjects[iSub])

  # Read the CSV file
  sub_data <- read.csv(file.path(Exp3DataFolder, File_pattern), header = TRUE)

  # Add subject data into the data data
  Exp3Data <- rbind(Exp3Data, sub_data)
}

# Participant Exclusion w/ Permutation -----------------------------------

# Calculate the mean accuracy for each subject
Mean_Accs <- tapply(Exp3Data$Accuracy, Exp3Data$ParticipantID , mean)

# parameters
nIterations <- 10000
Subs <- 1:length(subjects)

# Check if subject has less trials than it is supposed to be
for (iSub in Subs) {

  nRows = na.omit(Exp3Data[Exp3Data$ParticipantID == iSub,])

  if (dim(nRows)[1] < 192) {
    print(paste("Sub", iSub, "has", dim(nRows)[1]), sep = "")
  }
}
```

```
## [1] "Sub 2 has 0"
## [1] "Sub 5 has 96"
## [1] "Sub 12 has 176"
## [1] "Sub 15 has 32"
```

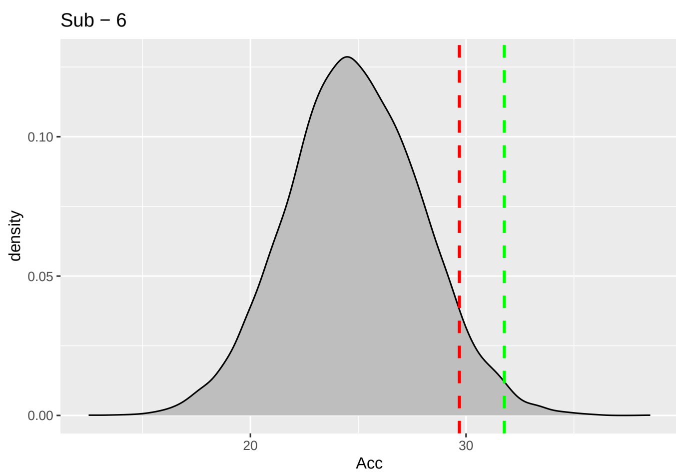
```r
# Loop through subjects to test whether they performed near chance-level
for (iSub in Subs) {

  NullDist <- data.frame(Acc=rep(NA,nIterations))

  for (i in 1:nIterations) {
    # Generate 192 random responses between 1 and 4
    Responses <- na.omit(Exp3Data[Exp3Data$ParticipantID == iSub, "Response"])

    # Generate 192 random answers between 1 and 4
    Answers <- sample(na.omit(Exp3Data[Exp3Data$ParticipantID == iSub, "CorrectKey"]))

    NullDist[i,1] <- ((sum(Responses == Answers))/length(Responses))*100
  }

  # Sort distribution
  SortedNullDist <- NullDist[order(NullDist$Acc),]
  Significance_Threshold <- SortedNullDist[length(SortedNullDist)-nIterations*.05]

  print(ggplot(NullDist, aes(x=Acc)) + geom_density(fill="gray") +
```

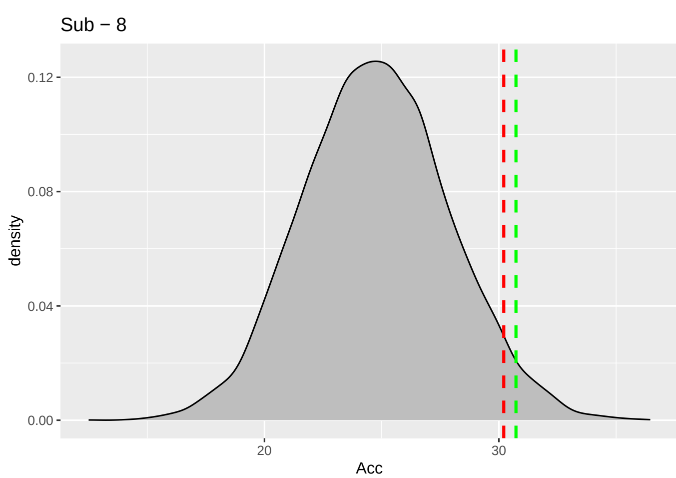
```
geom_vline(aes(xintercept = Mean_Accs[as.character(iSub)]*100),
                   color="green", linetype="dashed", linewidth=1)+
        geom_vline(aes(xintercept = Significance_Threshold),
                   color="red", linetype="dashed", linewidth=1)+
        ggtitle(paste("Sub -", as.character(iSub))))

  if ((Mean_Accs[as.character(iSub)])*100 <= Significance_Threshold) {
    print(paste("Sub", iSub,' is at chance-level!!!'))
  }
}
```

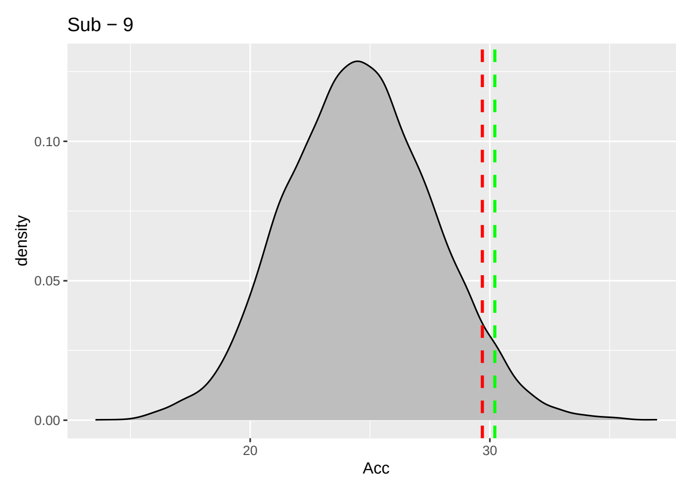


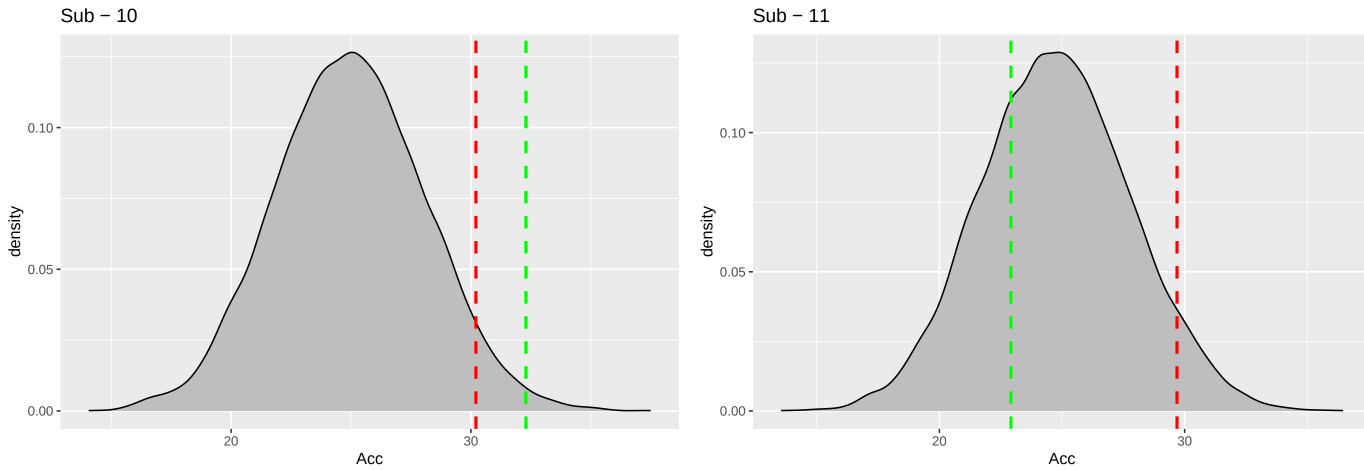



```
## [1] "Sub 5  is at chance-level!!!"
```

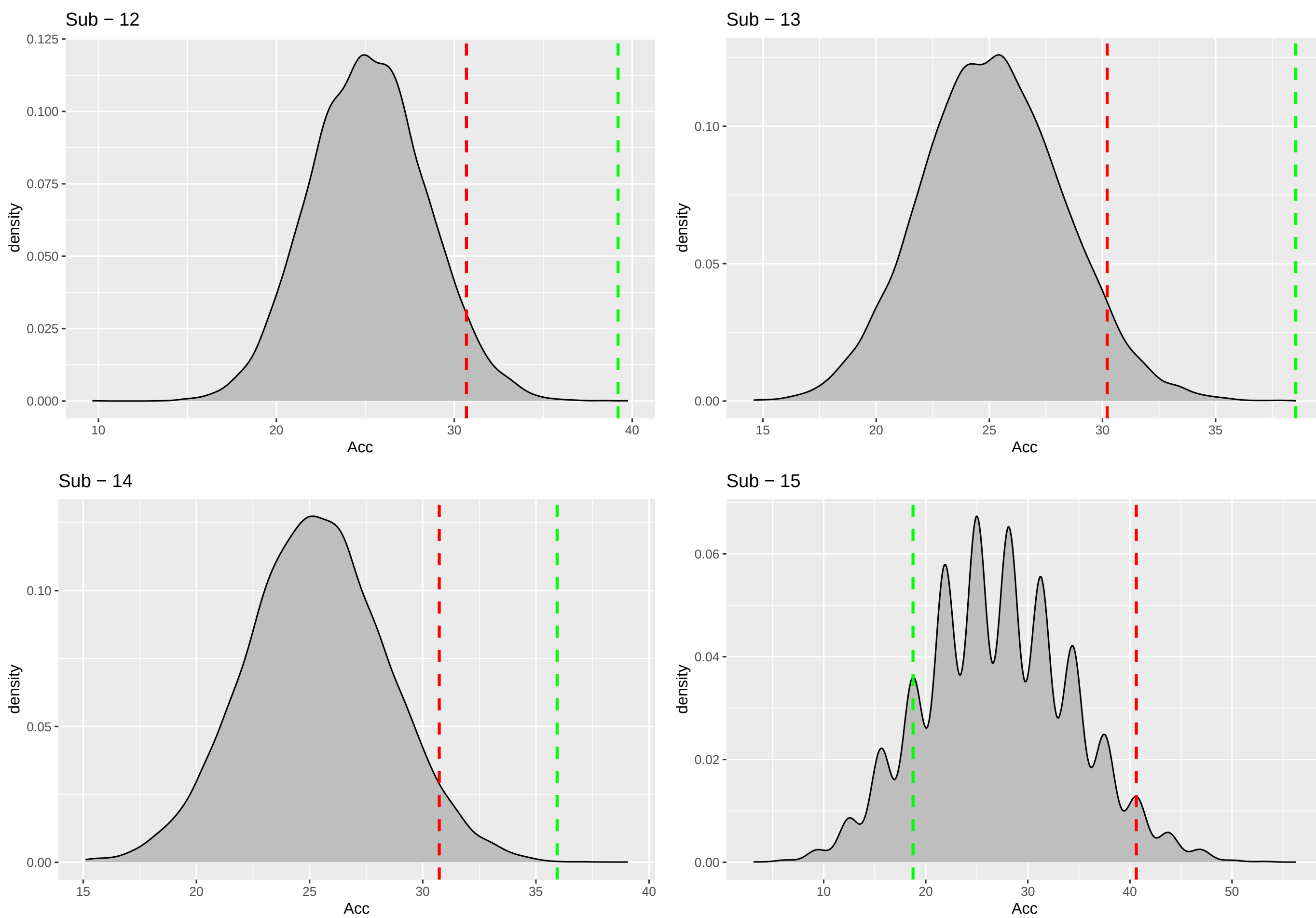




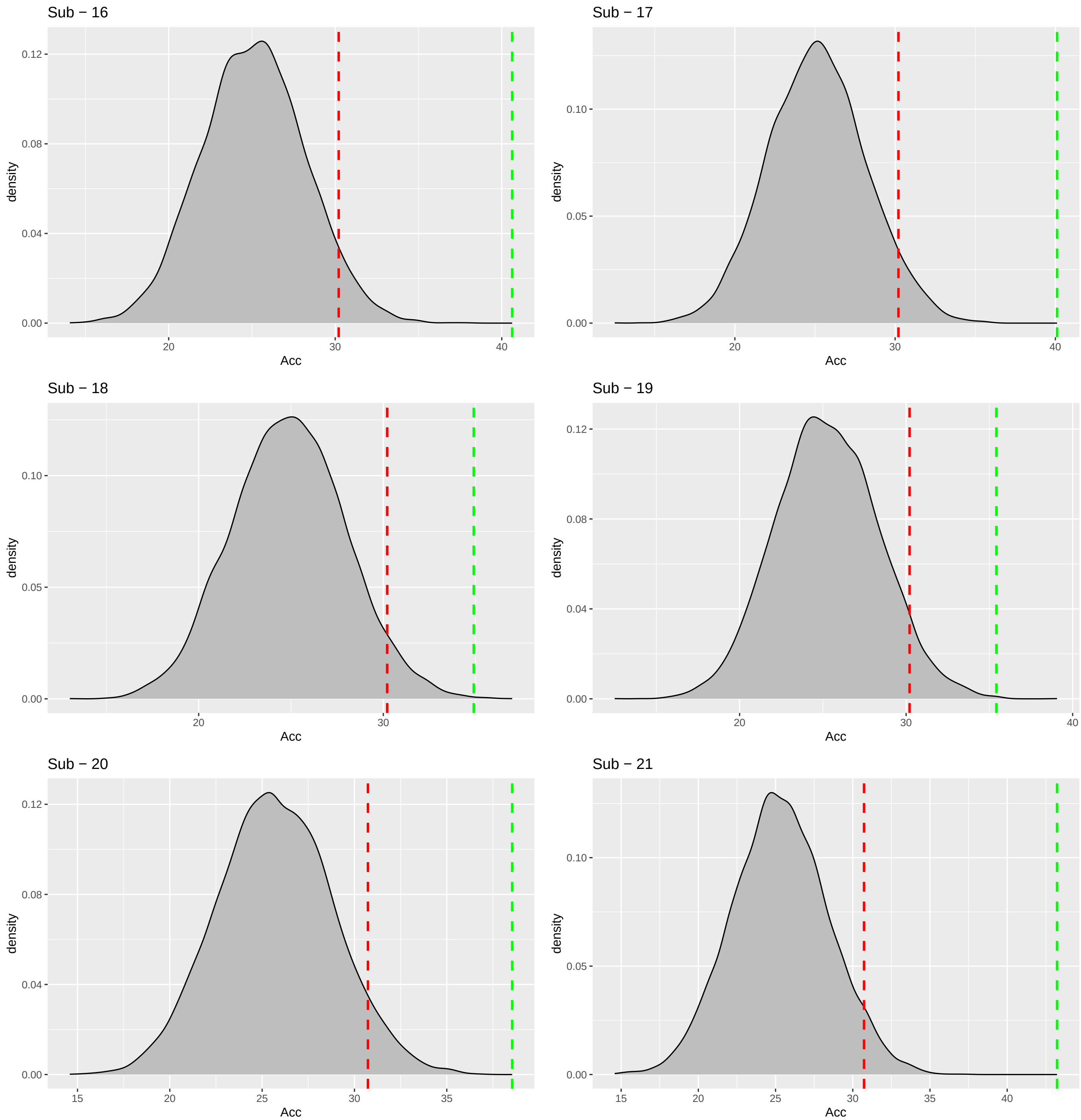


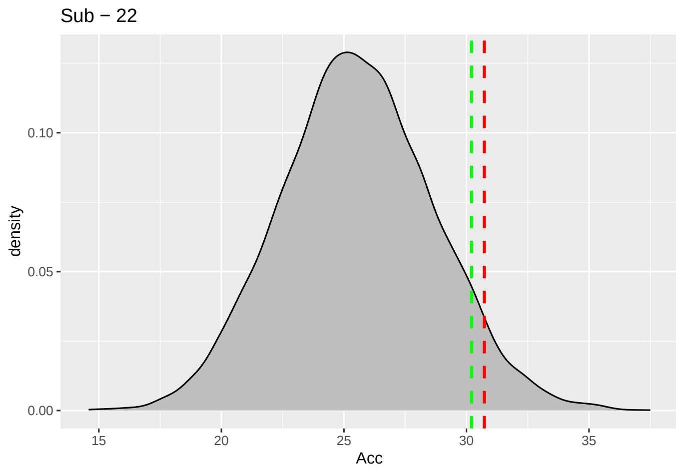



```
## [1] "Sub 11  is at chance-level!!!"
```

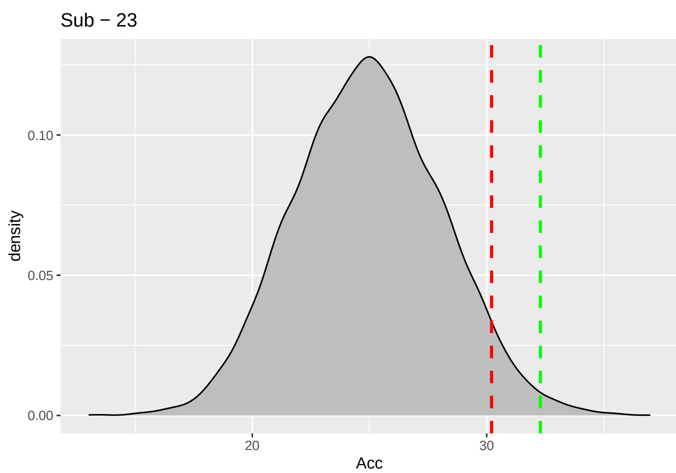


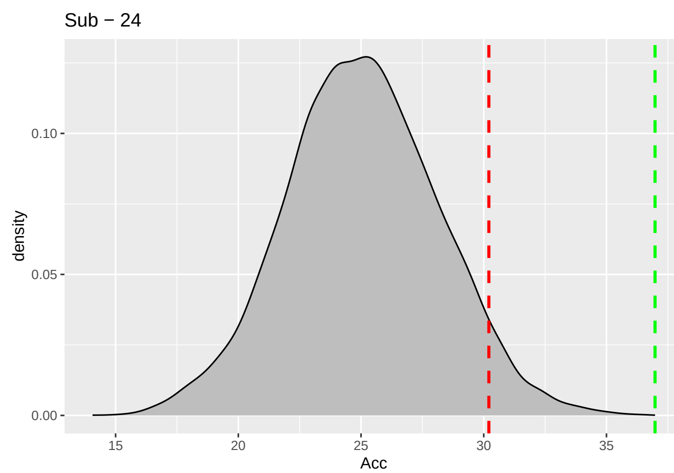

```
## [1] "Sub 15  is at chance-level!!!"
```

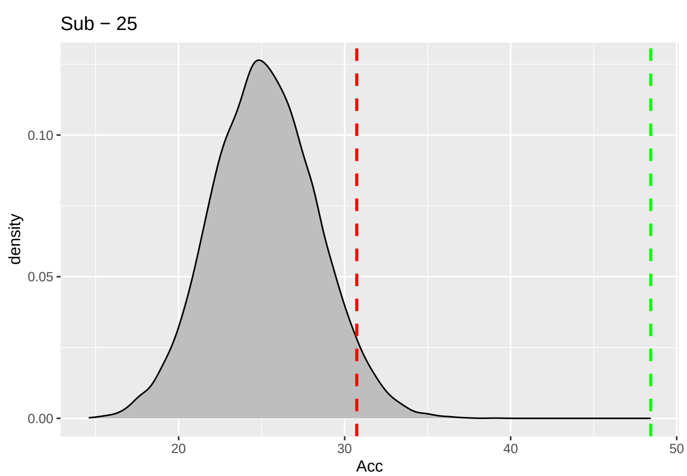

Sub − 16
Sub − 17
Sub − 18
Sub − 19
Sub − 20
Sub − 21
density
Acc

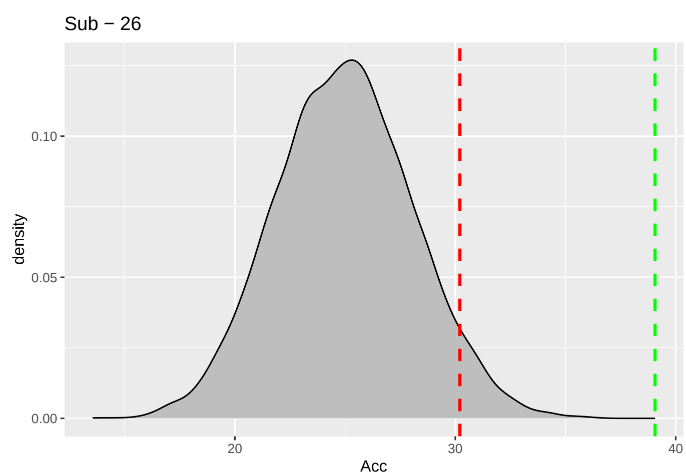


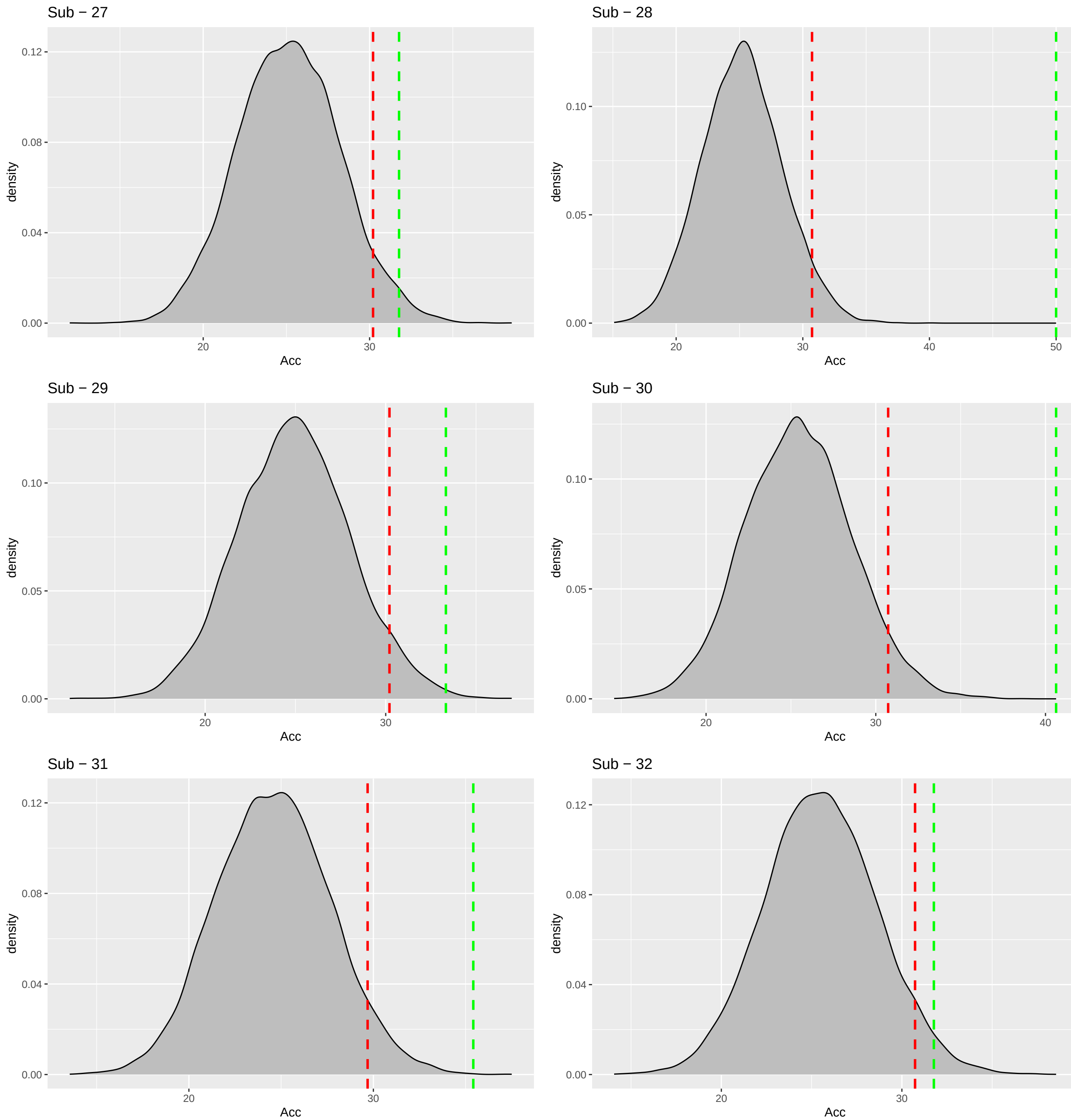

```
## [1] "Sub 22  is at chance-level!!!"
```



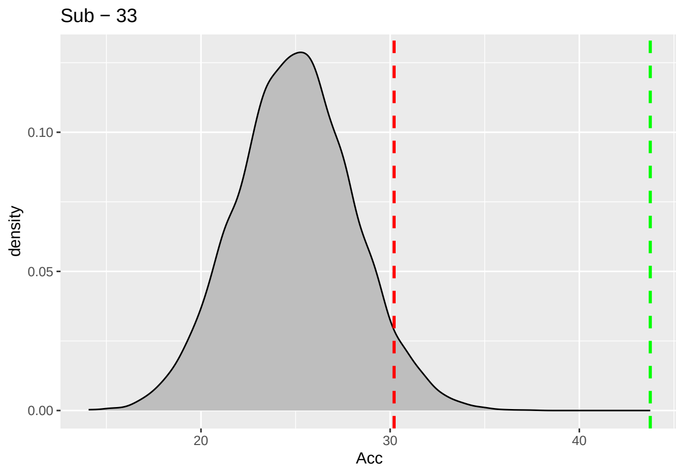


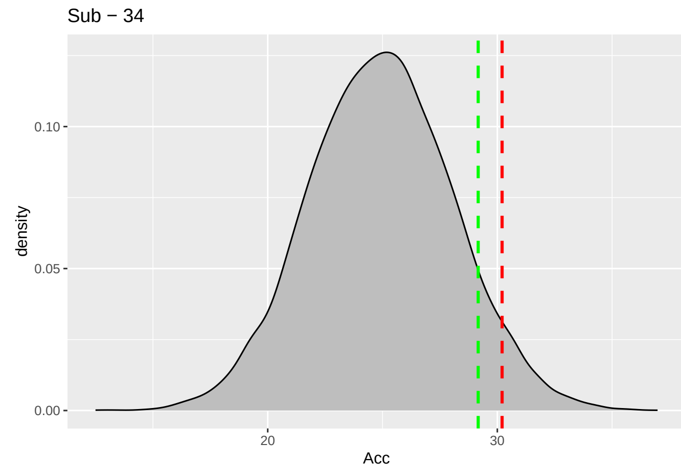




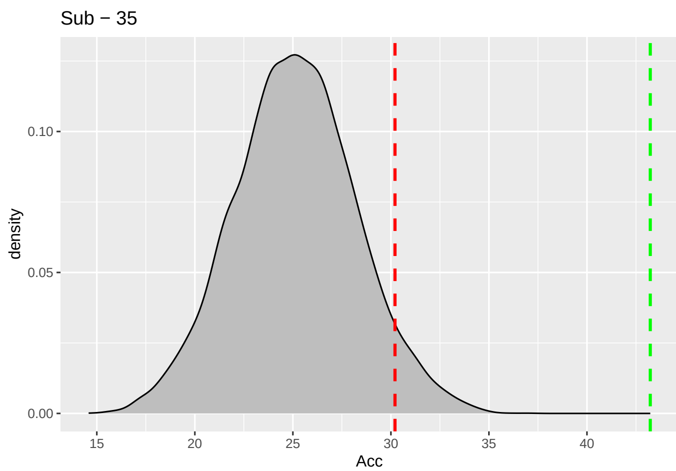


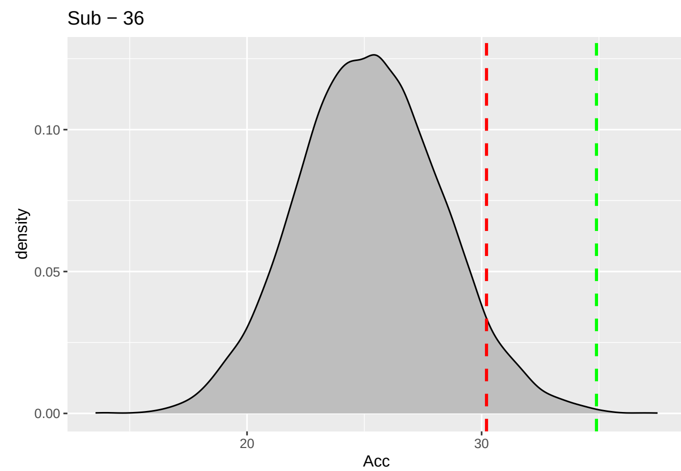
Sub − 27
Sub − 28
Sub − 29
Sub − 30
Sub − 31
Sub − 32
density
Acc

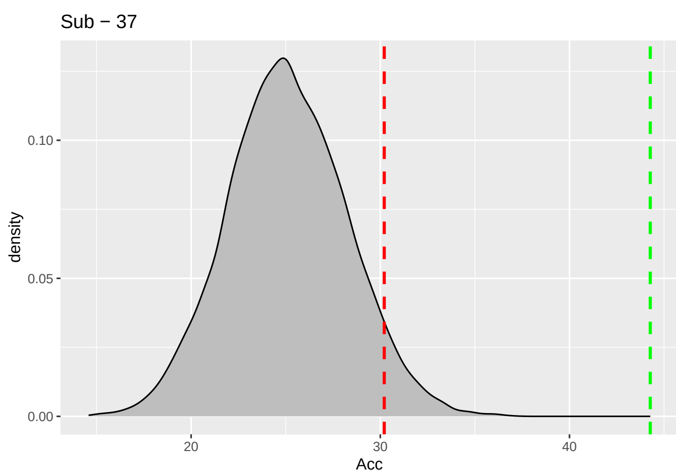


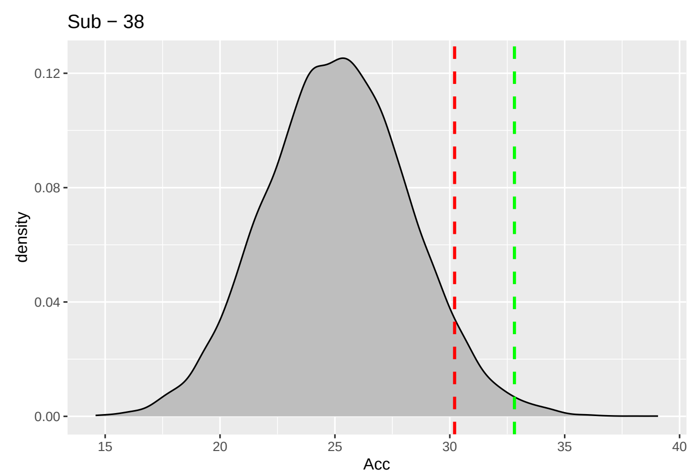



```
## [1] "Sub 34  is at chance-level!!!"
```










```
# Load only non-excluded participants
# 2, 5, and 15 have too many missing data
# 5, 11, 15, 22, and 34 below performance threshold
subjects <- c("Sub_01", "Sub_03", "Sub_04", "Sub_06", "Sub_07",
```

```r
              "Sub_08", "Sub_09", "Sub_10", "Sub_12", "Sub_13", "Sub_14",
              "Sub_16", "Sub_17", "Sub_18", "Sub_19", "Sub_20", "Sub_21",
              "Sub_23", "Sub_24", "Sub_25", "Sub_26", "Sub_27",
              "Sub_28", "Sub_29", "Sub_30", "Sub_31", "Sub_32", "Sub_33",
              "Sub_35", "Sub_36", "Sub_37", "Sub_38")

# Initialize data frame
Exp3Data <- data.frame()

# Load demographic data
DemogData <- read.csv("ParticipantDemographics.csv")

# Loop through subjects
for (iSub in 1:length(subjects)) {
  # Define file name
  File_pattern <- list.files(path = Exp3DataFolder, pattern = subjects[iSub])

  # Read the CSV file
  sub_data <- read.csv(file.path(Exp3DataFolder, File_pattern), header = TRUE)

  # Get demographics
  NRows <- nrow(sub_data)
  Age <- rep(DemogData[iSub, 3], NRows)
  Female <- rep(as.integer(DemogData[iSub, 4] == "Female"), NRows)
  RightHanded <- rep(as.integer(DemogData[iSub, 5] == "Right"), NRows)

  sub_data$Age <- Age
  sub_data$Female <- Female
  sub_data$RightHanded <- RightHanded

  # Combine the subject's runs with the overall combined data
  Exp3Data <- rbind(Exp3Data, sub_data)
}

# Get demographic info
nFemale <- sum(aggregate(Female ~ ParticipantID, data = Exp3Data, median)$Female)
mAge <- mean(aggregate(Age ~ ParticipantID, data = Exp3Data, median)$Age)
sdAge <- sd(aggregate(Age ~ ParticipantID, data = Exp3Data, median)$Age)
rangeAge <- range(aggregate(Age ~ ParticipantID, data = Exp3Data, median)$Age)


# Logistic Mixed-effects Model --------------------------------------------

# Set the seed for reproducibility
set.seed(13021996)

# recode variable types
Exp3Data$Condition <- as.factor(Exp3Data$Condition)
Exp3Data$ParticipantID <- as.factor(Exp3Data$ParticipantID)
Exp3Data$ClipID <- as.factor(Exp3Data$ClipID)

# registered model
TIME.model = glmer(Accuracy ~ Condition + (1|ClipID) + (1|ParticipantID),
```

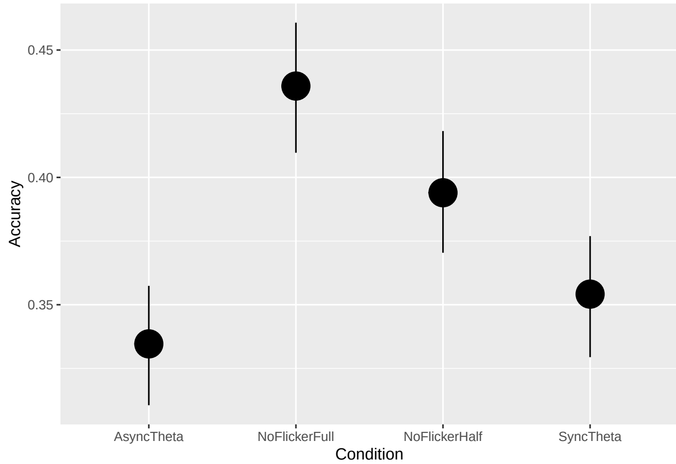
```r
                   data=Exp3Data, family = binomial)

# summary and plotting
Anova(TIME.model)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##            Chisq Df Pr(>Chisq)
## Condition 40.691  3  7.604e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(TIME.model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition + (1 | ClipID) + (1 | ParticipantID)
##    Data: Exp3Data
##
##      AIC      BIC   logLik deviance df.resid
##   8006.5   8046.8  -3997.2   7994.5     6122
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.4629 -0.7698 -0.6317  1.1141  2.0917
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.18503  0.4302
##  ParticipantID (Intercept) 0.03088  0.1757
## Number of obs: 6128, groups:  ClipID, 192; ParticipantID, 32
##
## Fixed effects:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -0.72009    0.07724  -9.323  < 2e-16 ***
## ConditionNoFlickerFull  0.45034    0.07646   5.890 3.87e-09 ***
## ConditionNoFlickerHalf  0.26946    0.09899   2.722  0.00649 **
## ConditionSyncTheta      0.08858    0.09953   0.890  0.37349
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) CndNFF CndNFH
## CndtnNFlckF -0.519
## CndtnNFlckH -0.653  0.404
## CndtnSyncTh -0.649  0.401  0.702
```

```r
ggplot(Exp3Data,aes(x=Condition,y=Accuracy))+
  stat_summary(fun.data=mean_cl_boot,size=2)
```




```r
TIME.emm.s <- emmeans(TIME.model, "Condition")
pairs(TIME.emm.s)
```

```
##  contrast                      estimate     SE  df z.ratio p.value
##  AsyncTheta - NoFlickerFull     -0.4503 0.0765 Inf  -5.890  <.0001
##  AsyncTheta - NoFlickerHalf     -0.2695 0.0990 Inf  -2.722  0.0329
##  AsyncTheta - SyncTheta         -0.0886 0.0995 Inf  -0.890  0.8101
##  NoFlickerFull - NoFlickerHalf   0.1809 0.0976 Inf   1.853  0.2487
##  NoFlickerFull - SyncTheta       0.3618 0.0982 Inf   3.684  0.0013
##  NoFlickerHalf - SyncTheta       0.1809 0.0766 Inf   2.362  0.0846
##
## Results are given on the log odds ratio (not the response) scale.
## P value adjustment: tukey method for comparing a family of 4 estimates
```

```r
# Conditions within TIME.emm.s ordered: AsyncTheta, NoFlickerFull, NoFlickerHalf, SyncTheta
# Planned comparisons
(out1 <- contrast(TIME.emm.s, list(SyncTheta.vs.NoFlickerFull  = c(0, -1, 0, 1))))
```

```
##  contrast                   estimate     SE  df z.ratio p.value
##  SyncTheta.vs.NoFlickerFull   -0.362 0.0982 Inf  -3.684  0.0002
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out2 <- contrast(TIME.emm.s, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

```
##  contrast                estimate     SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta   0.0886 0.0995 Inf   0.890  0.3735
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out3 <- contrast(TIME.emm.s, list(SyncTheta.vs.NoFlickerHalf  = c(0, 0, -1, 1))))
```

```
##  contrast                   estimate     SE  df z.ratio p.value
##  SyncTheta.vs.NoFlickerHalf   -0.181 0.0766 Inf  -2.362  0.0182
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out4 <- contrast(TIME.emm.s, list(AsyncTheta.vs.NoFlickerFull = c(1, -1, 0, 0))))
```

```
##  contrast                    estimate     SE  df z.ratio p.value
##  AsyncTheta.vs.NoFlickerFull    -0.45 0.0765 Inf  -5.890  <.0001
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out5 <- contrast(TIME.emm.s, list(AsyncTheta.vs.NoFlickerHalf  = c(1, 0, -1, 0))))
```

```
##  contrast                    estimate    SE  df z.ratio p.value
##  AsyncTheta.vs.NoFlickerHalf   -0.269 0.099 Inf  -2.722  0.0065
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out6 <- contrast(TIME.emm.s, list(NoFlickerFull.vs.NoFlickerHalf  = c(0, 1, -1, 0))))
```

```
##  contrast                       estimate     SE  df z.ratio p.value
##  NoFlickerFull.vs.NoFlickerHalf    0.181 0.0976 Inf   1.853  0.0639
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(EffectSizes <- eff_size(TIME.emm.s, sigma = sigma(TIME.model), edf = df.residual(TIME.model), method =
```

```
##  contrast                      effect.size     SE  df asymp.LCL asymp.UCL
##  AsyncTheta - NoFlickerFull        -0.4503 0.0766 Inf   -0.6004   -0.3003
##  AsyncTheta - NoFlickerHalf        -0.2695 0.0990 Inf   -0.4635   -0.0754
##  AsyncTheta - SyncTheta            -0.0886 0.0995 Inf   -0.2837    0.1065
##  NoFlickerFull - NoFlickerHalf      0.1809 0.0976 Inf   -0.0105    0.3723
##  NoFlickerFull - SyncTheta          0.3618 0.0983 Inf    0.1692    0.5544
##  NoFlickerHalf - SyncTheta          0.1809 0.0766 Inf    0.0307    0.3310
##
## sigma used for effect sizes: 1
## Confidence level used: 0.95
```

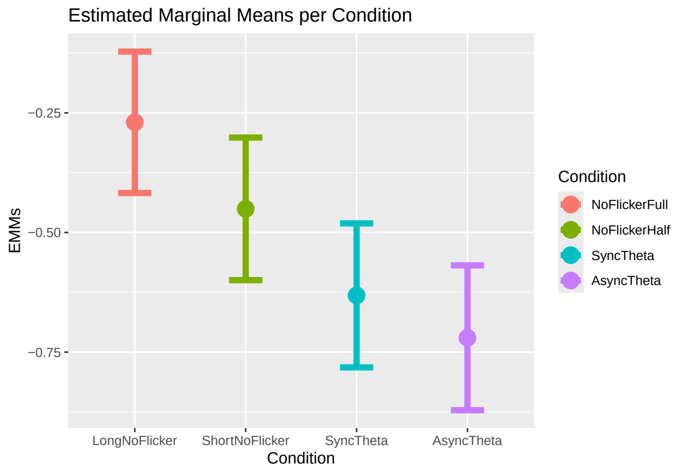
```r
# Calculate the mean accuracy for each subject
sub_means <- tapply(Exp3Data$Accuracy, Exp3Data$ParticipantID , mean)
# Calculate accuracy for each subject and condition
Exp3pSub_pCondition_acc <- aggregate(Accuracy ~ ParticipantID + Condition, data = Exp3Data, mean)
Exp3pSub_pCondition_acc$Condition = factor(Exp3pSub_pCondition_acc$Condition, levels = c( "NoFlickerFul

# Bayes analysis for sync vs async theta
bf = ttestBF(Exp3pSub_pCondition_acc[Exp3pSub_pCondition_acc$Condition == "SyncTheta",]$Accuracy, Exp3pS

1/bf
```

```
## Bayes factor analysis
## --------------
## [1] Null, mu=0 : 2.408496 ±0.03%
##
## Against denominator:
##   Alternative, r = 0.707106781186548, mu =/= 0
## ---
## Bayes factor type: BFoneSample, JZS
```

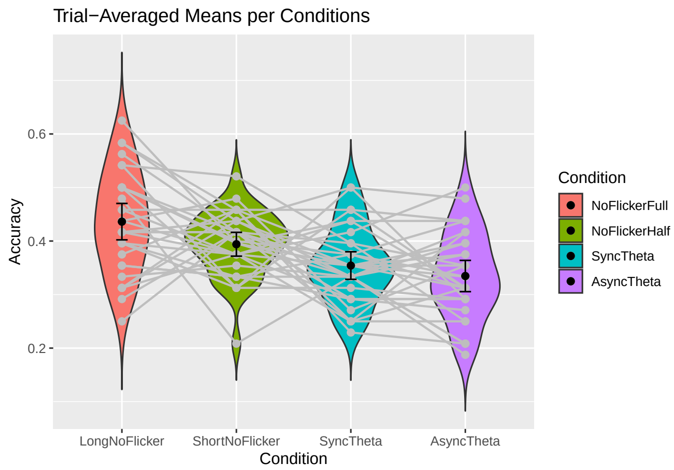
```r
# Bayes analysis for full vs half duration no-flicker
bf = ttestBF(Exp3pSub_pCondition_acc[Exp3pSub_pCondition_acc$Condition == "NoFlickerFull",]$Accuracy, E

bf
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 1.481057 ±0.02%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

```r
# Plot EMMs
EMFrame <- data.frame(TIME.emm.s)
EMFrame$Condition = factor(EMFrame$Condition, levels = c("NoFlickerFull", "NoFlickerHalf", "SyncTheta",

ggplot(EMFrame, aes(x = Condition, y = emmean, color = Condition)) +
  geom_point(size = 5) +
  geom_errorbar(aes(ymin = asymp.LCL, ymax = asymp.UCL), width = 0.3, linewidth = 2) +
  labs(title="Estimated Marginal Means per Condition", x = "Condition", y = "EMMs")+
  scale_color_manual(values = c("NoFlickerFull" = LongNoFlickerColour, "SyncTheta" = SyncThetaColour, "
  scale_x_discrete(labels=c("NoFlickerFull" = "LongNoFlicker", "NoFlickerHalf" = "ShortNoFlicker"))
```




```r
# Plot trial-averaged means
ggplot(Exp3pSub_pCondition_acc, aes(x = Condition, y = Accuracy, fill = Condition)) +
  geom_violin(trim=FALSE)+
  geom_point(size = 2, color = "gray") +
  geom_line(aes(group = ParticipantID), color = "gray", size = .7) +
  labs(title="Trial-Averaged Means per Conditions", x = "Condition", y = "Accuracy") +
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")+
  scale_fill_manual(values = c("NoFlickerFull" = LongNoFlickerColour, "SyncTheta" = SyncThetaColour, "A
  scale_x_discrete(labels=c("NoFlickerFull" = "LongNoFlicker", "NoFlickerHalf" = "ShortNoFlicker"))
```




```
#ggsave("Rawmeans.tiff", dpi = 300)

# Run bootstrapping t-tests
# Set the seed for reproducibility
set.seed(13021996)

# Calculate means
mAll <- mean(Exp3pSub_pCondition_acc$Accuracy)*100
sdAll <- sd((sub_means)*100)
mSyncTheta <- mean(Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="SyncTheta"])*10
mNoFlicker <- mean(Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="NoFlickerFull"]
mAsyncTheta <- mean(Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="AsyncTheta"])*
mSyncDelta <- mean(Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="NoFlickerHalf"]

SyncTheta_NoFlickerFull <- Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="SyncThe
  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "NoFlickerFull"]*100
SyncTheta_AsyncTheta <-
  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="SyncTheta"]*100-
  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "AsyncTheta"]*100
SyncTheta_NoFlickerHalf <- Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="AsyncThe
  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "NoFlickerHalf"]*100
AsyncTheta_NoFlickerHalf <- Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="SyncThe
  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "NoFlickerHalf"]*100
NoFlickerFull_NoFlickerHalf <- Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition=="NoF
  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "NoFlickerHalf"]*100


(SyncTheta_NoFlickerHalf_out <- boot.t.test(SyncTheta_NoFlickerHalf,
                                            alternative = c("two.sided", "less", "greater"),
                                            mu = 0, paired = FALSE, var.equal = FALSE,
                                            conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  SyncTheta_NoFlickerHalf
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.00156
```

```
## bootstrap mean of x (SE) = -5.941366 (1.652283)
## 95 percent bootstrap percentile confidence interval:
##  -9.166667 -2.617187
##
## Results without bootstrap:
## t = -3.4974, df = 31, p-value = 0.001443
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -9.399921 -2.475079
## sample estimates:
## mean of x
##   -5.9375
```

```
abs(mean(SyncTheta_NoFlickerHalf)/sd(SyncTheta_NoFlickerHalf)) # Cohen's d
```

```
## [1] 0.6182659
```

```
(SyncTheta_NoFlickerFull_out <- boot.t.test(SyncTheta_NoFlickerFull,
                                         alternative = c("two.sided", "less", "greater"),
                                         mu = 0, paired = FALSE, var.equal = FALSE,
                                         conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  SyncTheta_NoFlickerFull
## number of bootstrap samples:  1e+05
## bootstrap p-value < 1e-05
## bootstrap mean of x (SE) = -8.203337 (1.390857)
## 95 percent bootstrap percentile confidence interval:
##  -10.937500  -5.403646
##
## Results without bootstrap:
## t = -5.7581, df = 31, p-value = 2.446e-06
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -11.108686  -5.297564
## sample estimates:
## mean of x
## -8.203125
```

```
abs(mean(SyncTheta_NoFlickerFull)/sd(SyncTheta_NoFlickerFull)) # Cohen's d
```

```
## [1] 1.01789
```

```
(SyncTheta_AsyncTheta_out <- boot.t.test(SyncTheta_AsyncTheta,
                                          alternative = c("two.sided", "less", "greater"),
                                          mu = 0, paired = FALSE, var.equal = FALSE,
                                          conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
```

```
##  Bootstrap One Sample t-test
##
## data:  SyncTheta_AsyncTheta
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.1827
## bootstrap mean of x (SE) = 1.947514 (1.44862)
## 95 percent bootstrap percentile confidence interval:
##  -0.9114583  4.8177083
##
## Results without bootstrap:
## t = 1.318, df = 31, p-value = 0.1971
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -1.069103  4.975353
## sample estimates:
## mean of x
##  1.953125
```

```
abs(mean(SyncTheta_AsyncTheta)/sd(SyncTheta_AsyncTheta)) # Cohen's d
```

```
## [1] 0.2329992
```

```
(AsyncTheta_NoFlickerHalf_out <- boot.t.test(AsyncTheta_NoFlickerHalf,
                                       alternative = c("two.sided", "less", "greater"),
                                       mu = 0, paired = FALSE, var.equal = FALSE,
                                       conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  AsyncTheta_NoFlickerHalf
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.03588
## bootstrap mean of x (SE) = -3.997006 (1.708695)
## 95 percent bootstrap percentile confidence interval:
##  -7.3307292 -0.5859375
##
## Results without bootstrap:
## t = -2.2761, df = 31, p-value = 0.0299
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -7.5546446 -0.4141054
## sample estimates:
## mean of x
## -3.984375
```

```
abs(mean(AsyncTheta_NoFlickerHalf)/sd(AsyncTheta_NoFlickerHalf)) # Cohen's d
```

```
## [1] 0.4023563
```

```r
(NoFlickerFull_NoFlickerHalf_out <- boot.t.test(NoFlickerFull_NoFlickerHalf,
                                              alternative = c("two.sided", "less", "greater"),
                                              mu = 0, paired = FALSE, var.equal = FALSE,
                                              conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  NoFlickerFull_NoFlickerHalf
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.03588
## bootstrap mean of x (SE) = 4.22217 (1.895222)
## 95 percent bootstrap percentile confidence interval:
##  0.4947917 7.9817708
##
## Results without bootstrap:
## t = 2.1776, df = 31, p-value = 0.03717
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.267563 8.169937
## sample estimates:
## mean of x
##   4.21875
```

```r
abs(mean(NoFlickerFull_NoFlickerHalf)/sd(NoFlickerFull_NoFlickerHalf)) # Cohen's d
```

```
## [1] 0.3849531
```

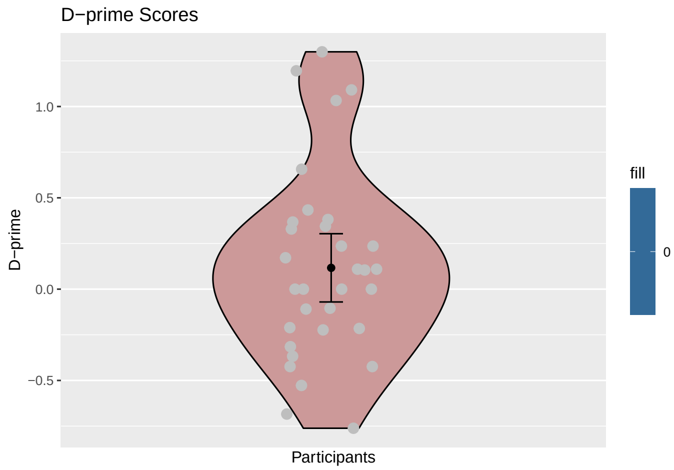
```r
# Set the seed for reproducibility
set.seed(13021996)

# Arrange long-format data

# List of subject names
subjects <- c("Sub_01", "Sub_03", "Sub_04", "Sub_06", "Sub_07",
              "Sub_08", "Sub_09", "Sub_10", "Sub_12", "Sub_13", "Sub_14",
              "Sub_16", "Sub_17", "Sub_18", "Sub_19", "Sub_20", "Sub_21",
              "Sub_23", "Sub_24", "Sub_25", "Sub_26", "Sub_27",
              "Sub_28", "Sub_29", "Sub_30", "Sub_31", "Sub_32", "Sub_33",
              "Sub_35", "Sub_36", "Sub_37", "Sub_38")
nSub <- length(subjects)

# Initialize vectors to store results
nHits_pSub <- rep(NA, nSub)
HitRate_pSub <- rep(NA, nSub)
nFAs_pSub <- rep(NA, nSub)
FARate_pSub <- rep(NA, nSub)
Exp3Dprime_pSub <- rep(NA, nSub)
Exp3Sync_subnums <- rep(NA, nSub)

# Loop over subjects
for (iSub in 1:nSub) {
```

```r
  # List all files in the current directory that match the pattern
  file_name <- list.files(path = Exp3SyncDataFolder, pattern = subjects[iSub])

  Exp3SyncData <- read.csv(file.path(Exp3SyncDataFolder, file_name), header = TRUE)

  # Calculate Hit Rate
  nHits_pSub[iSub] <- sum(Exp3SyncData$SyncRating == 1 & Exp3SyncData$Accuracy == 1)
  HitRate_pSub[iSub] <- nHits_pSub[iSub] / sum(Exp3SyncData$SyncRating == 1)

  # Calculate False Alarm Rate
  nFAs_pSub[iSub] <- sum(Exp3SyncData$SyncRating == 2 & Exp3SyncData$Accuracy == 0)
  FARate_pSub[iSub] <- nFAs_pSub[iSub] / sum(Exp3SyncData$SyncRating == 2)

  # Calculate d-prime
  Exp3Dprime_pSub[iSub] <- qnorm(HitRate_pSub[iSub]) - qnorm(FARate_pSub[iSub])
  Exp3Sync_subnums[iSub] <- Exp3SyncData$ParticipantID[iSub]
}

# Create a data frame
Exp3DprimeData <- data.frame(ParticipantID = as.factor(Exp3Sync_subnums), Dprime = Exp3Dprime_pSub)

# significance test
(DPrime_out <- boot.t.test(Exp3DprimeData$Dprime, alternative = c("two.sided", "less", "greater"),
                           mu = 0, paired = FALSE, var.equal = FALSE, conf.level = 0.95,
                           R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  Exp3DprimeData$Dprime
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.1808
## bootstrap mean of x (SE) = 0.1165539 (0.08941703)
## 95 percent bootstrap percentile confidence interval:
##  -0.05540737  0.29742508
##
## Results without bootstrap:
## t = 1.2728, df = 31, p-value = 0.2126
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.07024092  0.30344193
## sample estimates:
## mean of x
## 0.1166005
```

```r
# Create a violin plot

ggplot(data = Exp3DprimeData, aes(x = 0, y = Dprime, fill = 0)) +
  geom_violin(width=1, fill = SyncDiscColour, color = "black") +
  scale_x_discrete( ) +
  geom_point(size = 3, color = "gray",position = position_jitterdodge()) +
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black") +
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black") +
```

```r
  labs(title = "D-prime Scores",
       x = "Participants",
       y = "D-prime")
```




```r
# Set the seed for reproducibility
set.seed(13021996)

# Calculate accuracy for each subject and condition
Exp2pSub_pCondition_acc <- aggregate(Accuracy ~ ParticipantID + Condition, data = Exp2Data, mean)
Exp2pSub_pCondition_acc$Condition = factor(Exp2pSub_pCondition_acc$Condition, levels = c( "NoFlicker",

Exp2_SyncTheta <- Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition == "SyncTheta"]*10
Exp2_AsyncTheta <- Exp2pSub_pCondition_acc$Accuracy[Exp2pSub_pCondition_acc$Condition == "AsyncTheta"]*
Exp2_Sync_Async <- Exp2_SyncTheta - Exp2_AsyncTheta

Exp3SyncTheta <-  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "SyncTheta"]*10
Exp3AsyncTheta <-  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "AsyncTheta"]*
Exp3Sync_Async <- Exp3SyncTheta -Exp3AsyncTheta

(MixedVsBlocked <- boot.t.test(Exp3Sync_Async, Exp2_Sync_Async,
                                        alternative = c("two.sided", "less", "greater"),
                                        mu = 0, paired = FALSE, var.equal = FALSE,
                                        conf.level = 0.95, R = 100000, symmetric = FALSE))
```

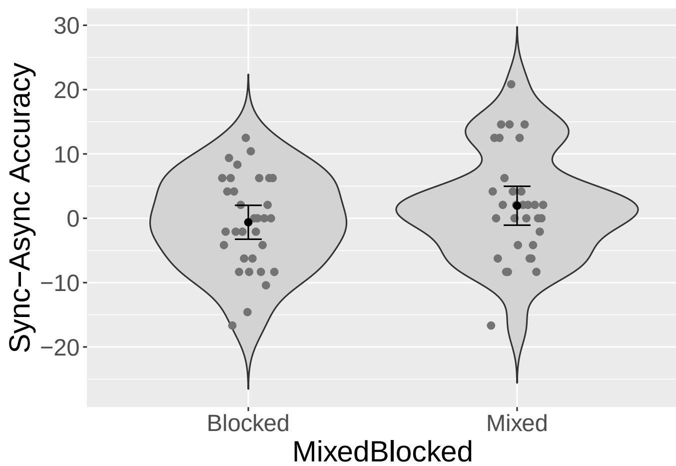
```
##
##  Bootstrap Welch Two Sample t-test
##
## data:  Exp3Sync_Async and Exp2_Sync_Async
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.1906
## bootstrap difference of means (SE) = 2.572741 (1.927639)
## 95 percent bootstrap percentile confidence interval:
##  -1.204427  6.412760
##
## Results without bootstrap:
## t = 1.3081, df = 60.868, p-value = 0.1958
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
##  -1.359588  6.502817
## sample estimates:
##  mean of x  mean of y
##  1.9531250 -0.6184896
```

```r
MixedandBlocked <- data.frame(c(Exp3Sync_Async,Exp2_Sync_Async),c(rep("Mixed",32), rep("Blocked",32)))
colnames(MixedandBlocked) <- c("Accuracy", "MixedBlocked")

bf = ttestBF(formula = Accuracy ~ MixedBlocked, data = MixedandBlocked, paired = FALSE)

1/bf
```

```
## Bayes factor analysis
## --------------
## [1] Null, mu1-mu2=0 : 1.902953 ±0.01%
##
## Against denominator:
##   Alternative, r = 0.707106781186548, mu =/= 0
## ---
## Bayes factor type: BFindepSample, JZS
```

```r
ggplot(MixedandBlocked, aes(x = MixedBlocked, y = Accuracy)) +
  geom_violin(trim=FALSE, fill = "lightgray")+
  geom_point(size = 2, color = "gray45", position = position_dodge2(width = .2)) +
  labs(y = "Sync-Async Accuracy")+
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")+
  theme(text = element_text(size=20))+
  scale_fill_grey()
```




```r
# Set the seed for reproducibility
set.seed(13021996)


Exp23_sync_async <- rbind(Exp2pSub_pCondition_acc, Exp3pSub_pCondition_acc)
Exp23_sync_async <- subset(Exp23_sync_async, select = c(Condition, Accuracy))
```

```r
Exp23_sync_async$Accuracy <- Exp23_sync_async$Accuracy*100

SyncAsyncDiff <- Exp23_sync_async[Exp23_sync_async$Condition == "SyncTheta",]$Accuracy -
           Exp23_sync_async[Exp23_sync_async$Condition == "AsyncTheta",]$Accuracy

(Exp23_syncVSasync <- boot.t.test(SyncAsyncDiff,
                                  alternative = c("two.sided", "less", "greater"),
                                  mu = 0, paired = FALSE, var.equal = FALSE,
                                  conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  SyncAsyncDiff
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.5031
## bootstrap mean of x (SE) = 0.667764 (0.9768732)
## 95 percent bootstrap percentile confidence interval:
##  -1.269531  2.604167
##
## Results without bootstrap:
## t = 0.6751, df = 63, p-value = 0.5021
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -1.307992  2.642628
## sample estimates:
## mean of x
## 0.6673177
```

```r
abs(mean(SyncAsyncDiff)/sd(SyncAsyncDiff)) # Cohen's d
```

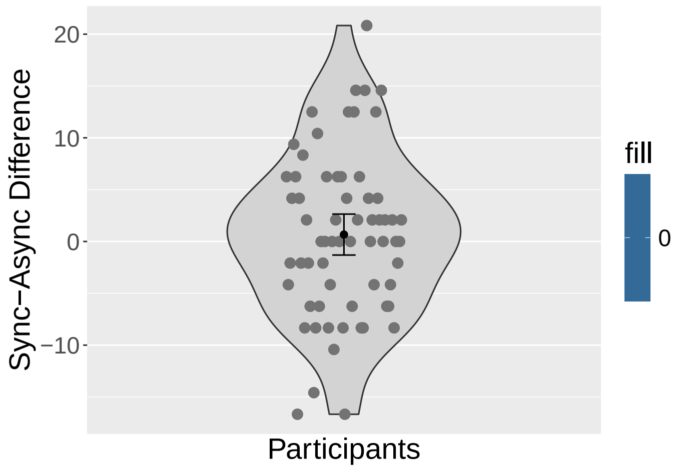
```
## [1] 0.08438727
```

```r
bf = ttestBF(SyncAsyncDiff, paired = FALSE)

1/bf
```

```
## Bayes factor analysis
## --------------
## [1] Null, mu=0 : 5.868865 ±0.08%
##
## Against denominator:
##   Alternative, r = 0.707106781186548, mu =/= 0
## ---
## Bayes factor type: BFoneSample, JZS
```

```r
Exp23_sync_asyncDiff <- data.frame(SyncAsyncDiff)

ggplot(data = Exp23_sync_asyncDiff, aes(x = 0, y = SyncAsyncDiff, fill = 0)) +
  geom_violin(width=1, fill = "lightgray") +
  scale_x_discrete( ) +
```

```r
   geom_point(size = 3, color = "gray45", position = position_dodge2(width = .5)) +
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black") +
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black") +
  labs(x = "Participants",
       y = "Sync-Async Difference")+
  theme(text = element_text(size=20))
```




```r
# Set the seed for reproducibility
set.seed(13021996)

# select the participants with Dprime
ClouterBaseline <- read.csv("Clouter_baseline.csv")


Clouter_NoFlickerHalf <- ClouterBaseline$AvgCorrect[ClouterBaseline$x6 == "half-length"]

NewNoFlickerHalf <-  Exp3pSub_pCondition_acc$Accuracy[Exp3pSub_pCondition_acc$Condition == "NoFlickerHal

(ClouterVsExp3 <- boot.t.test(Clouter_NoFlickerHalf, NewNoFlickerHalf,
                                        alternative = c("two.sided", "less", "greater"),
                                        mu = 0, paired = FALSE, var.equal = FALSE,
                                        conf.level = 0.95, R = 100000, symmetric = FALSE))
```

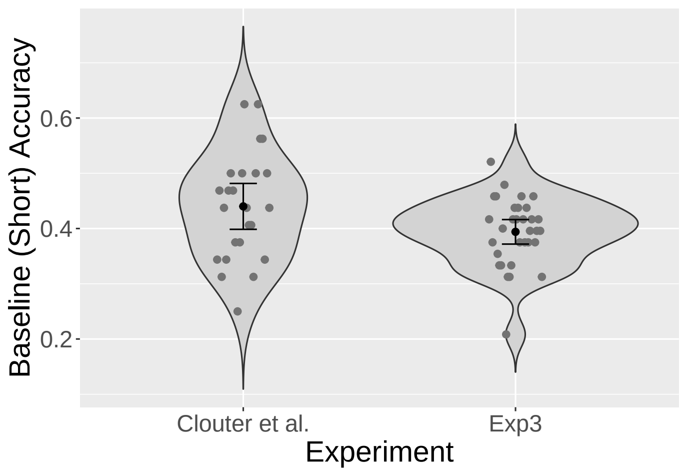
```
##
##  Bootstrap Welch Two Sample t-test
##
## data:  Clouter_NoFlickerHalf and NewNoFlickerHalf
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.04554
## bootstrap difference of means (SE) = 0.04609135 (0.02226702)
## 95 percent bootstrap percentile confidence interval:
##  0.002473958 0.090234375
##
## Results without bootstrap:
## t = 2.0183, df = 36.189, p-value = 0.05101
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
```

```
## -0.0002143163 0.0924018163
## sample estimates:
## mean of x mean of y
## 0.4401042 0.3940104
```

```
# Cohen's d
((mean(Clouter_NoFlickerHalf)-mean(NewNoFlickerHalf))/sqrt((var(Clouter_NoFlickerHalf)+var(NewNoFlicker
```

```
## [1] 0.5617639
```

```
ClouterandExp3 <- data.frame(c(Clouter_NoFlickerHalf, NewNoFlickerHalf), c(rep("Clouter et al.",length(

colnames(ClouterandExp3) <- c("Accuracy", "Experiment")

ggplot(ClouterandExp3, aes(x = Experiment, y = Accuracy)) +
  geom_violin(trim=FALSE, fill = "lightgray")+
  geom_point(size = 2, color = "gray45", position = position_dodge2(width = .2)) +
  labs(x = "Experiment", y = "Baseline (Short) Accuracy")+
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")+
  theme(text = element_text(size=20))+
  scale_fill_grey()
```




```
# Set the seed for reproducibility
set.seed(13021996)

# Average long and short no-flicker accuracies for each participant in Exp3 and combine with Exp2 parti
Exp1_2NoFlicker <- c(Exp2pSub_pCondition_acc[Exp2pSub_pCondition_acc$Condition == "NoFlicker",]$Accurac

# Get the median value but by through sorting as there are repeating scores near median
#
Exp1_2_Sync <- c(Exp2pSub_pCondition_acc[Exp2pSub_pCondition_acc$Condition == "SyncTheta",]$Accuracy, E
Exp1_2_Async <- c(Exp2pSub_pCondition_acc[Exp2pSub_pCondition_acc$Condition == "AsyncTheta",]$Accuracy,

# Get theta conditions' accuracies for participants above median
HighAcc_Sync <- Exp1_2_Sync[order(Exp1_2NoFlicker)[33:64]]*100
```

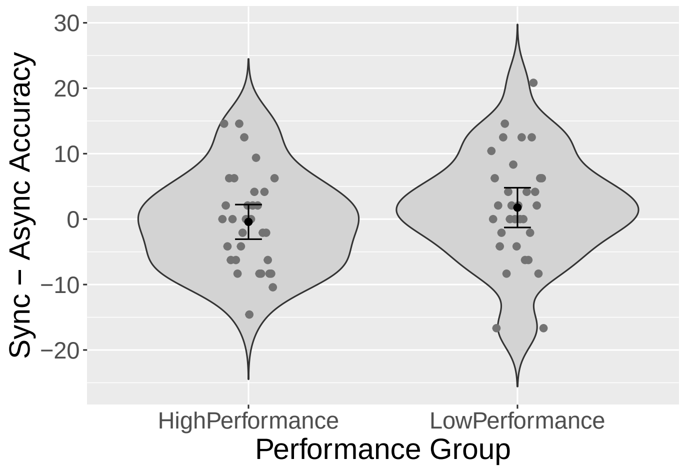
```r
HighAcc_Async <- Exp1_2_Async[order(Exp1_2NoFlicker)[33:64]]*100

# Get theta conditions' accuracies for participants below median
LowAcc_Sync <- Exp1_2_Sync[order(Exp1_2NoFlicker)[1:32]]*100

LowAcc_Async <- Exp1_2_Async[order(Exp1_2NoFlicker)[1:32]]*100

# Take the difference of Sync and Async theta
HighAcc_Sync_Async <- HighAcc_Sync - HighAcc_Async
LowAcc_Sync_Async <- LowAcc_Sync - LowAcc_Async

(HighvsLowPerf <- boot.t.test(HighAcc_Sync_Async, LowAcc_Sync_Async,
                              alternative = c("two.sided", "less", "greater"),
                              mu = 0, paired = FALSE, var.equal = FALSE,
                              conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap Welch Two Sample t-test
##
## data:  HighAcc_Sync_Async and LowAcc_Sync_Async
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.2765
## bootstrap difference of means (SE) = -2.174704 (1.935171)
## 95 percent bootstrap percentile confidence interval:
##  -5.957031  1.660156
##
## Results without bootstrap:
## t = -1.1051, df = 60.852, p-value = 0.2734
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -6.127414  1.765435
## sample estimates:
##  mean of x  mean of y
## -0.4231771  1.7578125
```

```r
HighLowPerf_SyncAsync <- data.frame(c(HighAcc_Sync_Async,LowAcc_Sync_Async),c(rep("High",32), rep("Low"
colnames(HighLowPerf_SyncAsync) <- c("Accuracy", "Performance")

bf = ttestBF(formula = Accuracy ~ Performance, data = HighLowPerf_SyncAsync, paired = FALSE)

1/bf
```

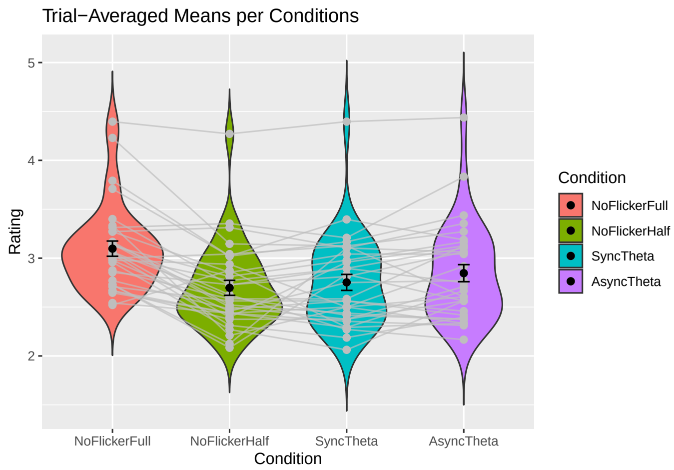
```
## Bayes factor analysis
## --------------
## [1] Null, mu1-mu2=0 : 2.33686 ±0.01%
##
## Against denominator:
##   Alternative, r = 0.707106781186548, mu =/= 0
## ---
## Bayes factor type: BFindepSample, JZS
```

```r
# plot
HighandLowPerf <- data.frame(c(HighAcc_Sync_Async,LowAcc_Sync_Async),c(rep("HighPerformance",length(Hig
colnames(HighandLowPerf) <- c("SyncAsync", "HighLowMedian")

ggplot(HighandLowPerf, aes(x = HighLowMedian, y = SyncAsync)) +
  geom_violin(trim=FALSE, fill = "lightgray")+
  geom_point(size = 2, color = "gray45", position = position_dodge2(width = .2)) +
  labs(x = paste("Performance Group"), y = "Sync - Async Accuracy")+
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")+
  theme(text = element_text(size=20))+
  scale_fill_grey()
```




```r
# Rating Analysis
# Set the seed for reproducibility
set.seed(13021996)

# Rating as random intercept----------------------

# Calculate the mean accuracy for each subject
sub_Ratingmeans <- tapply(Exp3Data$Rating, Exp3Data$ParticipantID , mean)
# Calculate accuracy for each subject and condition
pSub_pCondition_rating <- aggregate(Rating ~ ParticipantID + Condition, data = Exp3Data, mean)
pSub_pCondition_rating$Condition = factor(pSub_pCondition_rating$Condition, levels = c( "NoFlickerFull"

ggplot(pSub_pCondition_rating, aes(x = Condition, y = Rating, fill = Condition)) +
  geom_violin(trim=FALSE)+
  geom_point(size = 2, color = "gray") +
  geom_line(aes(group = ParticipantID), color = "gray", alpha = .7) +
  labs(title="Trial-Averaged Means per Conditions", x = "Condition", y = "Rating") +
  stat_summary(fun.pSub_pCondition_rating = mean_cl_normal, geom = "errorbar", width = 0.1, color = "bla
  stat_summary(fun.pSub_pCondition_rating = mean, geom = "point", size = 2, color = "black")
```

```
## No summary function supplied, defaulting to ‘mean_se()‘
## No summary function supplied, defaulting to ‘mean_se()‘
```




```r
# cor(Exp3Data$Rating, Exp3Data$Accuracy)
#
# ggplot(Exp3Data, aes(x=Rating, y=Accuracy)) +
#   geom_point()+
#   geom_smooth(method=lm)

# Model with Rating as random effect
TIME.Rating = glmer(Rating ~ Condition + (1|ClipID) + (1|ParticipantID),
                   data=Exp3Data)
Anova(TIME.Rating)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Rating
##            Chisq Df Pr(>Chisq)
## Condition 51.051  3   4.77e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(TIME.Rating)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Rating ~ Condition + (1 | ClipID) + (1 | ParticipantID)
##    Data: Exp3Data
##
## REML criterion at convergence: 18792.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3105 -0.7223 -0.0209  0.7272  3.4788
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.3932   0.6270
##  ParticipantID (Intercept) 0.1597   0.3996
##  Residual                  1.1412   1.0683
## Number of obs: 6128, groups:  ClipID, 192; ParticipantID, 32
```

```
##
## Fixed effects:
##                        Estimate Std. Error t value
## (Intercept)             2.84635    0.09914  28.709
## ConditionNoFlickerFull  0.25088    0.03860   6.499
## ConditionNoFlickerHalf -0.14895    0.09839  -1.514
## ConditionSyncTheta     -0.09440    0.09837  -0.960
##
## Correlation of Fixed Effects:
##             (Intr) CndNFF CndNFH
## CndtnNFlckF -0.194
## CndtnNFlckH -0.496  0.196
## CndtnSyncTh -0.496  0.196  0.923
```

```
TIME.emm.Rating <- emmeans(TIME.Rating, "Condition")
```

```
## Note: D.f. calculations have been disabled because the number of observations exceeds 3000.
## To enable adjustments, add the argument 'pbkrtest.limit = 6128' (or larger)
## [or, globally, 'set emm_options(pbkrtest.limit = 6128)' or larger];
## but be warned that this may result in large computation time and memory use.
## Note: D.f. calculations have been disabled because the number of observations exceeds 3000.
## To enable adjustments, add the argument 'lmerTest.limit = 6128' (or larger)
## [or, globally, 'set emm_options(lmerTest.limit = 6128)' or larger];
## but be warned that this may result in large computation time and memory use.
```

```
pairs(TIME.emm.Rating)
```

```
##  contrast                     estimate     SE  df z.ratio p.value
##  AsyncTheta - NoFlickerFull    -0.2509 0.0386 Inf  -6.499  <.0001
##  AsyncTheta - NoFlickerHalf     0.1489 0.0984 Inf   1.514  0.4291
##  AsyncTheta - SyncTheta         0.0944 0.0984 Inf   0.960  0.7724
##  NoFlickerFull - NoFlickerHalf  0.3998 0.0984 Inf   4.063  0.0003
##  NoFlickerFull - SyncTheta      0.3453 0.0984 Inf   3.509  0.0025
##  NoFlickerHalf - SyncTheta     -0.0545 0.0386 Inf  -1.413  0.4911
##
## Degrees-of-freedom method: asymptotic
## P value adjustment: tukey method for comparing a family of 4 estimates
```

```
# Conditions within TIME.emm.s ordered: AsyncTheta, NoFlicker, SyncDelta, SyncTheta
# Planned comparisons for correction .05/3
(out1 <- contrast(TIME.emm.Rating, list(SyncTheta.vs.NoFlicker  = c(0, -1, 0, 1))))
```

```
##  contrast               estimate     SE  df z.ratio p.value
##  SyncTheta.vs.NoFlicker   -0.345 0.0984 Inf  -3.509  0.0004
##
## Degrees-of-freedom method: asymptotic
```

```
(out2 <- contrast(TIME.emm.Rating, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

```
##  contrast                estimate     SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta  -0.0944 0.0984 Inf  -0.960  0.3372
##
## Degrees-of-freedom method: asymptotic
```

```r
(out3 <- contrast(TIME.emm.Rating, list(SyncTheta.vs.SyncDelta  = c(0, 0, -1, 1))))
```

```
##  contrast               estimate     SE  df z.ratio p.value
##  SyncTheta.vs.SyncDelta   0.0545 0.0386 Inf   1.413  0.1576
##
## Degrees-of-freedom method: asymptotic
```

```r
(EffectSizes <- eff_size(TIME.emm.Rating, sigma = sigma(TIME.Rating),
                         edf = df.residual(TIME.Rating), method = "pairwise"))
```

```
##  contrast                      effect.size     SE  df asymp.LCL asymp.UCL
##  AsyncTheta - NoFlickerFull        -0.2349 0.0362 Inf   -0.3058   -0.1639
##  AsyncTheta - NoFlickerHalf         0.1394 0.0921 Inf   -0.0411    0.3200
##  AsyncTheta - SyncTheta             0.0884 0.0921 Inf   -0.0921    0.2689
##  NoFlickerFull - NoFlickerHalf      0.3743 0.0922 Inf    0.1936    0.5550
##  NoFlickerFull - SyncTheta          0.3232 0.0922 Inf    0.1426    0.5038
##  NoFlickerHalf - SyncTheta         -0.0511 0.0361 Inf   -0.1219    0.0198
##
## sigma used for effect sizes: 1.068
## Degrees-of-freedom method: inherited from asymptotic when re-gridding
## Confidence level used: 0.95
```

```r
Exp3Data$RatingFactor = as.factor(Exp3Data$Rating)

# Model with Rating as random effect
TIME.modelwRating = glmer(Accuracy ~ Condition*poly(RatingFactor,2) + (1|ClipID) + (1|ParticipantID),
                   data=Exp3Data, family = binomial)

Anova(TIME.modelwRating)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: Accuracy
##                                 Chisq Df Pr(>Chisq)
## Condition                      36.691  3  5.348e-08 ***
## poly(RatingFactor, 2)          15.005  2  0.0005517 ***
## Condition:poly(RatingFactor, 2) 14.126  6  0.0282647 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(TIME.modelwRating)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Accuracy ~ Condition * poly(RatingFactor, 2) + (1 | ClipID) +
##     (1 | ParticipantID)
##    Data: Exp3Data
##
##      AIC      BIC   logLik deviance df.resid
##   7995.0   8089.0  -3983.5   7967.0     6114
```

```
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.5199 -0.7664 -0.6211  1.1015  2.1224
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  ClipID        (Intercept) 0.18584  0.4311
##  ParticipantID (Intercept) 0.03145  0.1773
## Number of obs: 6128, groups:  ClipID, 192; ParticipantID, 32
##
## Fixed effects:
##                                               Estimate Std. Error z value
## (Intercept)                                   -0.71965    0.07744  -9.293
## ConditionNoFlickerFull                         0.42656    0.07730   5.518
## ConditionNoFlickerHalf                         0.28398    0.09931   2.860
## ConditionSyncTheta                             0.09736    0.09982   0.975
## poly(RatingFactor, 2)1                        -1.99850    4.27687  -0.467
## poly(RatingFactor, 2)2                         3.67635    4.46363   0.824
## ConditionNoFlickerFull:poly(RatingFactor, 2)1 10.50215    5.68842   1.846
## ConditionNoFlickerHalf:poly(RatingFactor, 2)1 13.84837    5.90544   2.345
## ConditionSyncTheta:poly(RatingFactor, 2)1     15.80455    5.95909   2.652
## ConditionNoFlickerFull:poly(RatingFactor, 2)2  0.45487    6.09241   0.075
## ConditionNoFlickerHalf:poly(RatingFactor, 2)2 -6.78320    6.08137  -1.115
## ConditionSyncTheta:poly(RatingFactor, 2)2      7.51970    6.08954   1.235
##                                               Pr(>|z|)
## (Intercept)                                    < 2e-16 ***
## ConditionNoFlickerFull                        3.43e-08 ***
## ConditionNoFlickerHalf                         0.00424 **
## ConditionSyncTheta                             0.32938
## poly(RatingFactor, 2)1                         0.64030
## poly(RatingFactor, 2)2                         0.41015
## ConditionNoFlickerFull:poly(RatingFactor, 2)1  0.06486 .
## ConditionNoFlickerHalf:poly(RatingFactor, 2)1  0.01903 *
## ConditionSyncTheta:poly(RatingFactor, 2)1      0.00800 **
## ConditionNoFlickerFull:poly(RatingFactor, 2)2  0.94048
## ConditionNoFlickerHalf:poly(RatingFactor, 2)2  0.26468
## ConditionSyncTheta:poly(RatingFactor, 2)2      0.21688
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr) CndNFF CndNFH CndtST p(RF,2)1 p(RF,2)2 CNFF:(RF,2)1
## CndtnNFlckF  -0.512
## CndtnNFlckH  -0.651  0.399
## CndtnSyncTh  -0.647  0.396  0.701
## ply(RtF,2)1   0.008 -0.027 -0.003 -0.006
## ply(RtF,2)2   0.013 -0.015 -0.010 -0.010  0.061
## CNFF:(RF,2)1 -0.008 -0.083  0.004  0.006 -0.654   -0.038
## CNFH:(RF,2)1 -0.006  0.017  0.040  0.007 -0.668   -0.045    0.443
## CST:(RF,2)1  -0.007  0.018  0.006  0.023 -0.672   -0.048    0.445
## CNFF:(RF,2)2 -0.009  0.008  0.008  0.008 -0.056   -0.720   -0.057
## CNFH:(RF,2)2 -0.009  0.011  0.014  0.008 -0.033   -0.712    0.016
## CST:(RF,2)2  -0.009  0.011  0.007  0.002 -0.050   -0.700    0.030
```

```
##               CNFH:(RF,2)1 CST:(RF,2)1 CNFF:(RF,2)2 CNFH:(RF,2)2
## CndtnNFlckF
## CndtnNFlckH
## CndtnSyncTh
## ply(RtF,2)1
## ply(RtF,2)2
## CNFF:(RF,2)1
## CNFH:(RF,2)1
## CST:(RF,2)1   0.492
## CNFF:(RF,2)2  0.048        0.050
## CNFH:(RF,2)2  0.047        0.031       0.523
## CST:(RF,2)2   0.039        0.068       0.510        0.522
```

```r
TIME.emm.swRating <- emmeans(TIME.modelwRating,"Condition")
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```r
pairs(TIME.emm.swRating)
```

```
##  contrast                       estimate    SE  df z.ratio p.value
##  AsyncTheta - NoFlickerFull    -0.42028 0.114 Inf  -3.693  0.0013
##  AsyncTheta - NoFlickerHalf    -0.37762 0.129 Inf  -2.924  0.0182
##  AsyncTheta - SyncTheta         0.00645 0.130 Inf   0.049  1.0000
##  NoFlickerFull - NoFlickerHalf  0.04265 0.128 Inf   0.333  0.9873
##  NoFlickerFull - SyncTheta      0.42673 0.130 Inf   3.283  0.0057
##  NoFlickerHalf - SyncTheta      0.38407 0.113 Inf   3.410  0.0036
##
## Results are given on the log odds ratio (not the response) scale.
## P value adjustment: tukey method for comparing a family of 4 estimates
```

```r
# Conditions within TIME.emm.s ordered: AsyncTheta, NoFlicker, SyncDelta, SyncTheta
# Planned comparisons for correction .05/3
(out1 <- contrast(TIME.emm.swRating, list(SyncTheta.vs.NoFlicker  = c(0, -1, 0, 1))))
```

```
##  contrast               estimate   SE  df z.ratio p.value
##  SyncTheta.vs.NoFlicker   -0.427 0.13 Inf  -3.283  0.0010
##
## Results are given on the log odds ratio (not the response) scale.
```

```r
(out2 <- contrast(TIME.emm.swRating, list(SyncTheta.vs.AsyncTheta = c(-1, 0, 0, 1))))
```

```
##  contrast                estimate   SE  df z.ratio p.value
##  SyncTheta.vs.AsyncTheta -0.00645 0.13 Inf  -0.049  0.9605
##
## Results are given on the log odds ratio (not the response) scale.
```

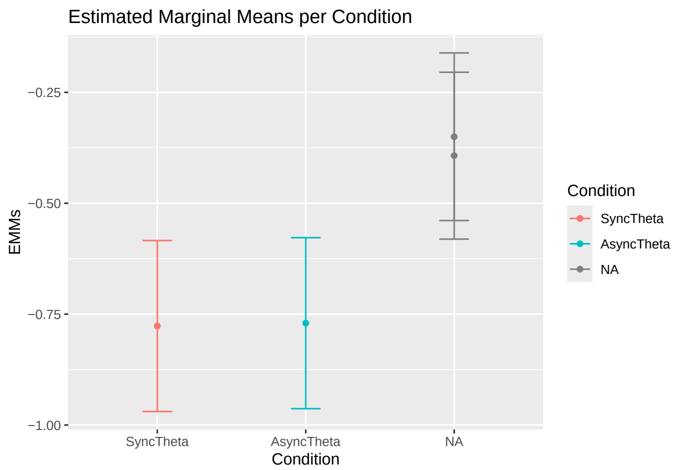
```r
(out3 <- contrast(TIME.emm.swRating, list(SyncTheta.vs.SyncDelta  = c(0, 0, -1, 1))))
```

```
##  contrast               estimate    SE  df z.ratio p.value
##  SyncTheta.vs.SyncDelta   -0.384 0.113 Inf  -3.410  0.0007
##
## Results are given on the log odds ratio (not the response) scale.
```

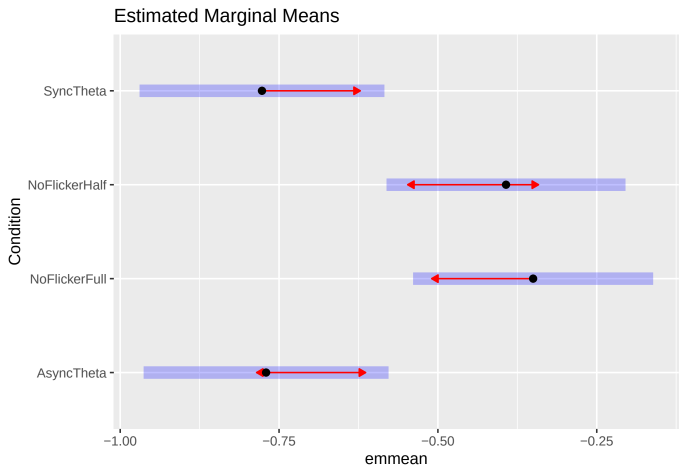
```r
(EffectSizes <- eff_size(TIME.emm.swRating, sigma = sigma(TIME.modelwRating),
                        edf = df.residual(TIME.modelwRating), method = "pairwise"))
```

```
##  contrast                      effect.size    SE  df asymp.LCL asymp.UCL
##  AsyncTheta - NoFlickerFull       -0.42028 0.114 Inf    -0.643    -0.197
##  AsyncTheta - NoFlickerHalf       -0.37762 0.129 Inf    -0.631    -0.124
##  AsyncTheta - SyncTheta            0.00645 0.130 Inf    -0.249     0.262
##  NoFlickerFull - NoFlickerHalf     0.04265 0.128 Inf    -0.208     0.294
##  NoFlickerFull - SyncTheta         0.42673 0.130 Inf     0.172     0.682
##  NoFlickerHalf - SyncTheta         0.38407 0.113 Inf     0.163     0.605
##
## sigma used for effect sizes: 1
## Confidence level used: 0.95
```

```r
RatingEMFrame <- data.frame(TIME.emm.swRating)
RatingEMFrame$Condition = factor(RatingEMFrame$Condition, levels = c("SyncTheta", "NoFlicker", "AsyncThe

ggplot(RatingEMFrame, aes(x = Condition, y = emmean, color = Condition)) +
  geom_point() +
  geom_errorbar(aes(ymin = asymp.LCL, ymax = asymp.UCL), width = 0.2) +
  labs(title="Estimated Marginal Means per Condition", x = "Condition", y = "EMMs")
```




```r
plot(TIME.emm.swRating, comparisons = TRUE) +
  ggtitle("Estimated Marginal Means")
```




```
# Set the seed for reproducibility
set.seed(13021996)

# select the participants with Dprime
Exp3DatawDprime <- Exp3pSub_pCondition_acc[Exp3pSub_pCondition_acc$ParticipantID %in% c(Exp3Sync_subnums

Theta_means <- aggregate(Accuracy ~ Condition*ParticipantID, data = Exp3DatawDprime, FUN = mean)
STheta_means <- Theta_means[Theta_means == "SyncTheta",]
ATheta_means <- Theta_means[Theta_means == "AsyncTheta",]
Exp3TIMEEffect <- STheta_means$Accuracy - ATheta_means$Accuracy

cor(Exp3DprimeData $Dprime, Exp3TIMEEffect, method = "pearson")
```

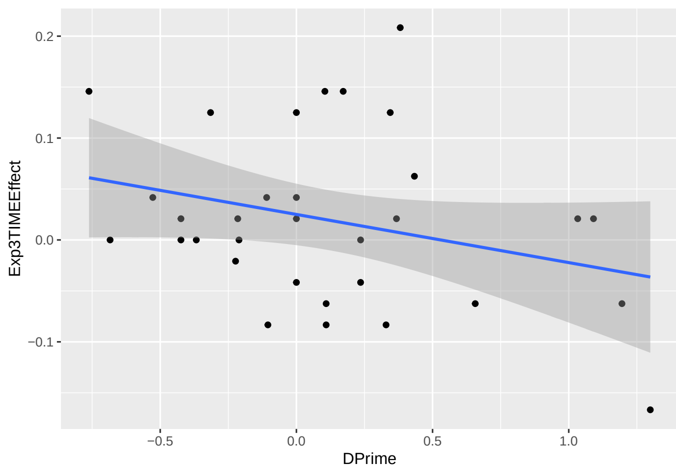
```
## [1] -0.2921964
```

```
cor.test(Exp3DprimeData $Dprime, Exp3TIMEEffect, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  Exp3DprimeData$Dprime and Exp3TIMEEffect
## t = -1.6735, df = 30, p-value = 0.1046
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.58162978  0.06290698
## sample estimates:
##        cor
## -0.2921964
```

```
DprimewTIME <- data.frame(DPrime = Exp3DprimeData $Dprime, Exp3TIMEEffect)

ggplot(DprimewTIME, aes(x=DPrime, y=Exp3TIMEEffect)) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



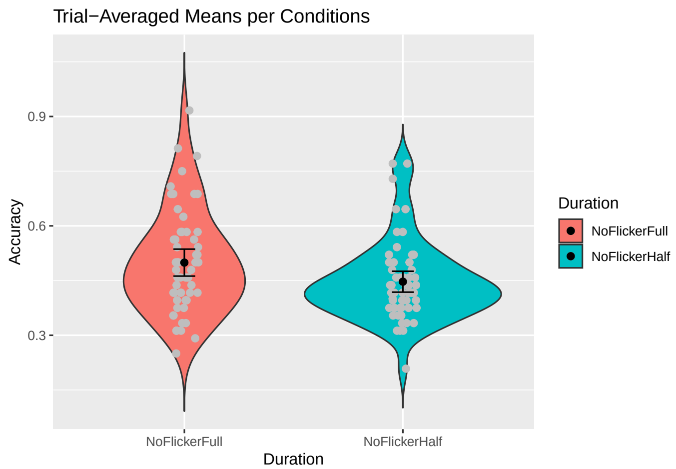

```r
# Set the seed for reproducibility
set.seed(13021996)

NoFlickerFull <- c(NoFlicker_pSub[NoFlicker_pSub$Condition == "NoFlickerFull",]$Accuracy,
Exp3pSub_pCondition_acc[Exp3pSub_pCondition_acc$Condition == "NoFlickerFull",]$Accuracy)

NoFlickerHalf <- c(NoFlicker_pSub[NoFlicker_pSub$Condition == "NoFlickerHalf",]$Accuracy,
Exp3pSub_pCondition_acc[Exp3pSub_pCondition_acc$Condition == "NoFlickerHalf",]$Accuracy)

(FullvsHalfDuration <- boot.t.test(NoFlickerFull, NoFlickerHalf,
                                   alternative = c("two.sided", "less", "greater"),
                                   mu = 0, paired = TRUE, var.equal = FALSE,
                                   conf.level = 0.95, R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap Paired t-test
##
## data:  NoFlickerFull and NoFlickerHalf
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.00078
## bootstrap mean of the differences (SE) = 0.0524216 (0.01451297)
## 95 percent bootstrap percentile confidence interval:
##  0.02399425 0.08081897
##
## Results without bootstrap:
## t = 3.5653, df = 57, p-value = 0.000744
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.02295622 0.08178516
## sample estimates:
## mean of the differences
##              0.05237069
```

```r
(mean(NoFlickerFull-NoFlickerHalf))/sd(NoFlickerFull-NoFlickerHalf) # Cohen's d
```

```
## [1] 0.4681427
```

```r
FullandHalfDuration <- data.frame(c(NoFlickerFull,NoFlickerHalf),c(rep("NoFlickerFull",length(NoFlickerF
colnames(FullandHalfDuration) <- c("Accuracy", "Duration")

ggplot(FullandHalfDuration, aes(x = Duration, y = Accuracy, fill = Duration)) +
  geom_violin(trim=FALSE)+
  geom_point(size = 2, color = "gray", position = position_jitterdodge(jitter.width = .13)) +
  labs(title="Trial-Averaged Means per Conditions", x = "Duration", y = "Accuracy")+
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black")+
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black")
```



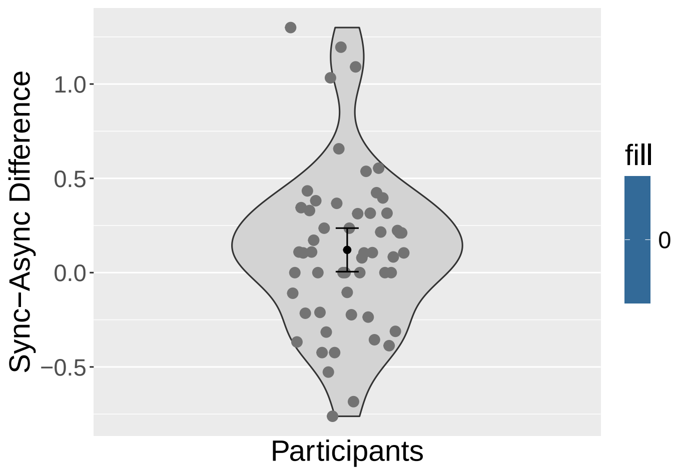

```r
# Set the seed for reproducibility
set.seed(13021996)

# significance test
(DPrime_out <- boot.t.test(c(Exp3DprimeData$Dprime, Exp2DprimeData$Dprime), alternative = c("two.sided"
                           mu = 0, paired = FALSE, var.equal = FALSE, conf.level = 0.95,
                           R = 100000, symmetric = FALSE))
```

```
##
##  Bootstrap One Sample t-test
##
## data:  c(Exp3DprimeData$Dprime, Exp2DprimeData$Dprime)
## number of bootstrap samples:  1e+05
## bootstrap p-value = 0.02906
## bootstrap mean of x (SE) = 0.1204927 (0.05662605)
## 95 percent bootstrap percentile confidence interval:
##  0.01035482 0.23392240
##
## Results without bootstrap:
## t = 2.094, df = 54, p-value = 0.04097
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.005129427 0.236039347
## sample estimates:
## mean of x
## 0.1205844
```

```r
(mean(c(Exp3DprimeData$Dprime, Exp2DprimeData$Dprime)))/sd(c(Exp3DprimeData$Dprime, Exp2DprimeData$Dpri
```

```
## [1] 0.2823484
```

```r
Exp23DprimeData <- rbind(Exp3DprimeData, Exp2DprimeData)

ggplot(data = Exp23DprimeData, aes(x = 0, y = Dprime, fill = 0)) +
  geom_violin(width=1, fill = "lightgray") +
  scale_x_discrete( ) +
    geom_point(size = 3, color = "gray45", position = position_dodge2(width = .5)) +
  stat_summary(fun.data = "mean_cl_normal", geom = "errorbar", width = 0.1, color = "black") +
  stat_summary(fun = "mean", geom = "point", size = 2, color = "black") +
  labs(x = "Participants",
       y = "Sync-Async Difference")+
  theme(text = element_text(size=20))
```




```r
cor(c(Exp3DprimeData$Dprime, Exp2DprimeData$Dprime), c(Exp3TIMEEffect, Exp2TIMEEffect), method = "pears
```

```
## [1] -0.1555573
```

```r
cor.test(c(Exp3DprimeData$Dprime, Exp2DprimeData$Dprime), c(Exp3TIMEEffect, Exp2TIMEEffect), method = "
```

```
##
##  Pearson's product-moment correlation
##
## data:  c(Exp3DprimeData$Dprime, Exp2DprimeData$Dprime) and c(Exp3TIMEEffect, Exp2TIMEEffect)
## t = -1.1464, df = 53, p-value = 0.2568
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.4041746  0.1144637
## sample estimates:
##        cor
## -0.1555573
```

```
DprimewTIME <- data.frame(DPrime = c(Exp3DprimeData$Dprime, Exp2DprimeData$Dprime), TIMEEffect = c(Exp3T

ggplot(DprimewTIME, aes(x=DPrime, y=TIMEEffect)) +
  geom_point()+
  geom_smooth(method=lm)+
theme(text = element_text(size=20))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```