Familiarity Facilitates Feature-based Face Processing Figures and Effect Sizes for Reaction Times

Matteo Visconti di Oleggio Castello, Kelsey G. Wheeler, Carlo Cipolli, M. Ida Gobbini

Contents

Setup

This R markdown file will produce the figures, descriptive statistics, and effect sizes for reaction times.

Load some useful libraries, return version information, and load the data.

```
# return version information
version
```

```
##                _
## platform       x86_64-apple-darwin13.4.0
## arch           x86_64
## os             darwin13.4.0
## system         x86_64, darwin13.4.0
## status
## major          3
## minor          2.3
## year           2015
## month          12
## day            10
## svn rev        69752
## language       R
## version.string R version 3.2.3 (2015-12-10)
## nickname       Wooden Christmas-Tree
```

```
packages <- c('dplyr',
              'ggplot2',
              'doParallel',
              'foreach',
              'assertthat',
              'knitr')
```

```
for (package in packages) {
  require(package, character.only=T)
  cat(paste(package, packageVersion(package), '\n'))
}
```

```
## dplyr 0.4.3
## ggplot2 2.1.0
## doParallel 1.0.10
## foreach 1.4.3
## assertthat 0.1
## knitr 1.12.3
```

```
data <- read.csv('../data/data.csv')
# set order of levels for plotting
data$orientation <- factor(data$orientation,
                           levels=c('Upright', 'Inverted'))
data$target_presence <- factor(data$target_presence,
                               levels=c('Target Present', 'Target Absent'))
# get correct trials
data_correct <- data %>% filter(correct == 1)
```

Set up some variables that will be used later.

```
nproc <- 4  # change this to use more/less processors for parallel use
seed <- 42  # seed for rng to obtain reproducible results in different runs
alpha <- .05  # significance level for confidence intervals
nbs <- 10000  # number of bootstrapping repetitions
```

Bootstrapping of the data

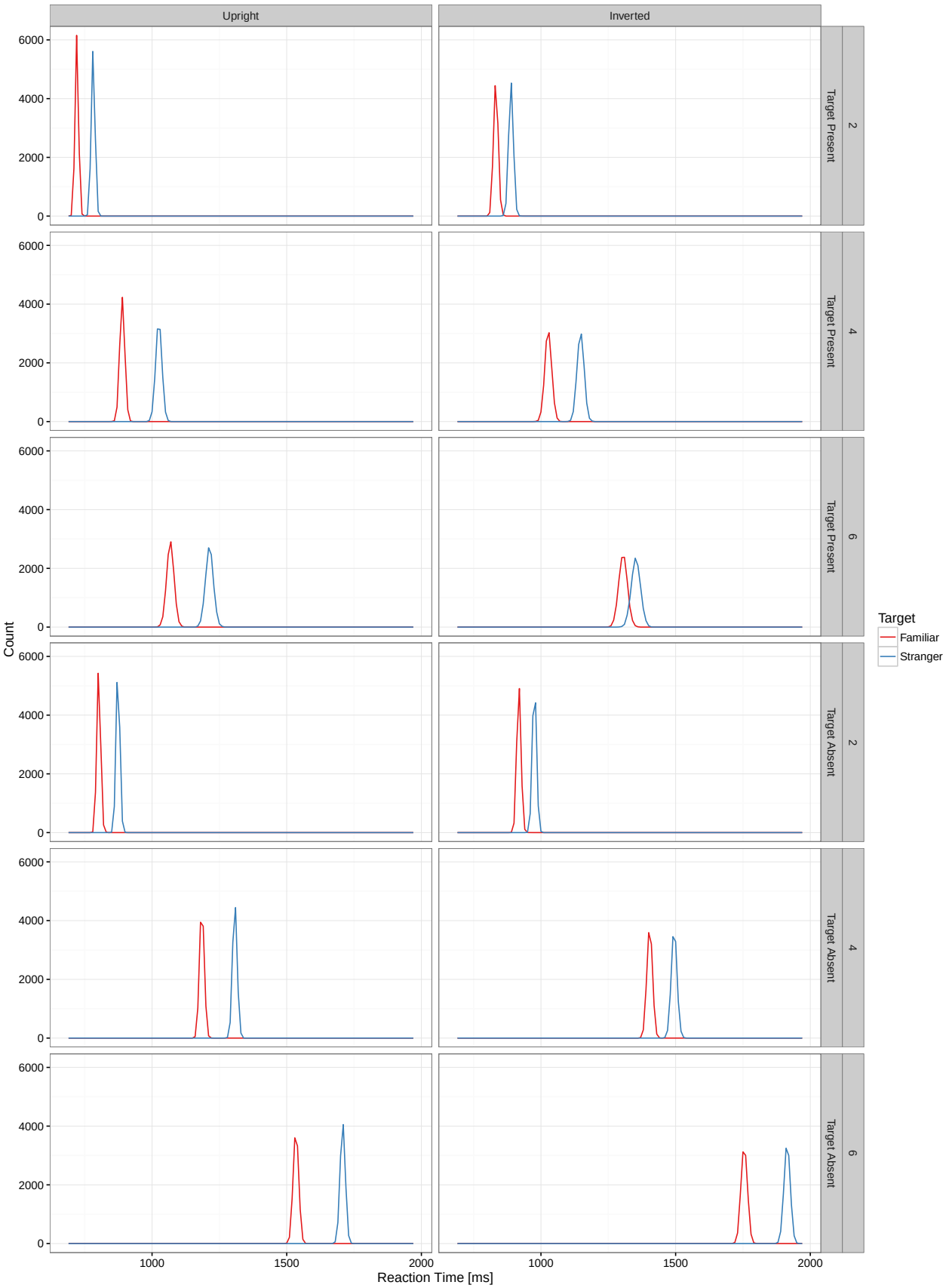
Bootstrap data resampling within each condition. It could take a while depending on the number of processors used.

```
cl <- makeCluster(nproc)
registerDoParallel(cl)
# push required packages to each worker
clusterCall(cl, function() library(magrittr))
clusterCall(cl, function() library(plyr))
clusterCall(cl, function() library(dplyr))
bstrap <- data.frame()
set.seed(seed)
bstrap <- foreach(i = 1:nbs, .combine=rbind) %dopar% {
    tmp <-
        data_correct %>%
        group_by(orientation, target_presence, familiarity,
                 set_size, target_sex, subid) %>%
        sample_frac(1, replace=T) %>%
        group_by(orientation, target_presence, familiarity, set_size) %>%
        summarise(avg=mean(RT))
```

```
    tmp$index <- i
    tmp
}
stopCluster(cl)
```

Visualize bootstrapping distribution

```
ggplot(bstrap, aes(avg, color=familiarity)) +
    geom_freqpoly(binwidth=10) +
    facet_grid(target_presence*set_size ~ orientation) +
    theme_bw(base_size=12) +
    labs(color='Target', x='Reaction Time [ms]', y='Count') +
    scale_color_brewer(palette='Set1')
```


Upright
Inverted
Target Present
2
4
6
Target Absent
2
4
6
Count
0
2000
4000
6000
1000
1500
2000
Reaction Time [ms]
Target
Familiar
Stranger

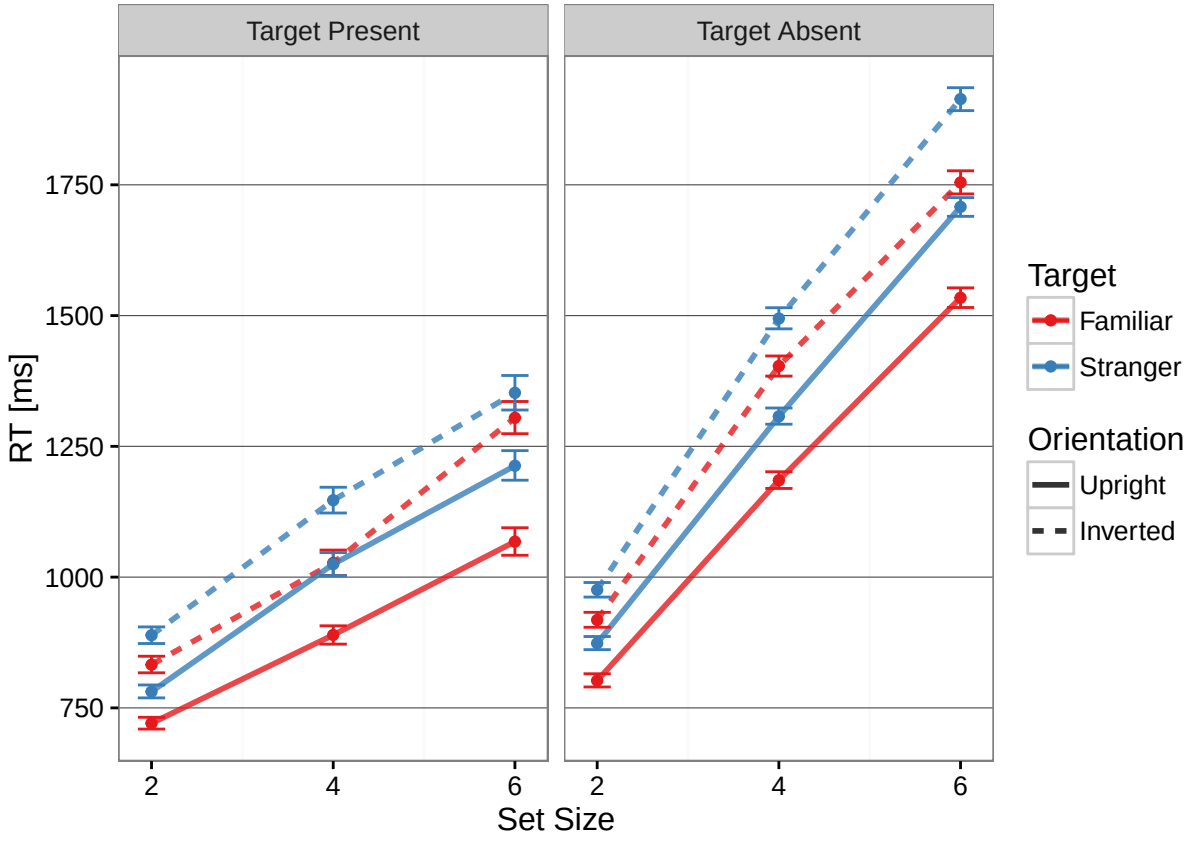
Plot of Average Reaction Times

Now compute confidence intervals and averages from the bootstrapped samples.

```
cis <-
    bstrap %>%
    group_by(orientation, target_presence, familiarity, set_size) %>%
    summarise(low=quantile(avg, alpha/2), high=quantile(avg, 1-alpha/2))

# compute average from original data
avgs <- data_correct %>%
    group_by(orientation, target_presence, familiarity, set_size) %>%
    summarise(avg=mean(RT))

cis <- merge(cis, avgs)
```

Now plot the figure

```
# first reorder the levels of target_presence and orientation
cis$orientation <- factor(cis$orientation, levels=c('Upright', 'Inverted'))
cis$target_presence <- factor(cis$target_presence,
                              levels=c('Target Present', 'Target Absent'))

ggplot(cis, aes(set_size, avg, ymin=low, ymax=high,
                color=familiarity, linetype=orientation)) +
    geom_line(alpha=.8, size=1) +
    geom_errorbar(width=0.3, linetype='solid')  +
    geom_point() +
    facet_grid(~ target_presence) +
    theme_bw(base_size=12) +
    theme(panel.grid.major.y = element_line(colour = "gray30"),
          panel.grid.major.x = element_blank(),
          panel.grid.minor.y = element_blank()) +
    scale_y_continuous(breaks=c(750, 1000, 1250, 1500, 1750)) +
    scale_x_continuous(breaks=c(2, 4, 6)) +
    labs(x='Set Size', y='RT [ms]', color='Target', linetype='Orientation') +
    scale_color_brewer(palette='Set1')
```



Plot of Unstandardized Effect Sizes for each Set Size and Condition

Compute difference Stranger - Familiar for each bootstrapped sample and obtain their confidence intervals.

```
bstrap_fam <- filter(bstrap, familiarity == 'Familiar') %>%
    select(-familiarity)
bstrap_str <- filter(bstrap, familiarity == 'Stranger') %>%
    select(-familiarity)
# assert that the order is the same
tocheck <- c('orientation', 'target_presence', 'set_size', 'index')
for (check in tocheck) {
    assert_that(all(bstrap_fam[, check] == bstrap_str[, check]))
}

bstrap_es <- bstrap_str
bstrap_es$avg <- bstrap_es$avg - bstrap_fam$avg

# compute cis for effect size
cis_es <-
    bstrap_es %>%
    group_by(orientation, target_presence, familiarity, set_size) %>%
    summarise(low=quantile(avg, alpha/2), high=quantile(avg, 1-alpha/2))
```

Now we do the same in the original data.

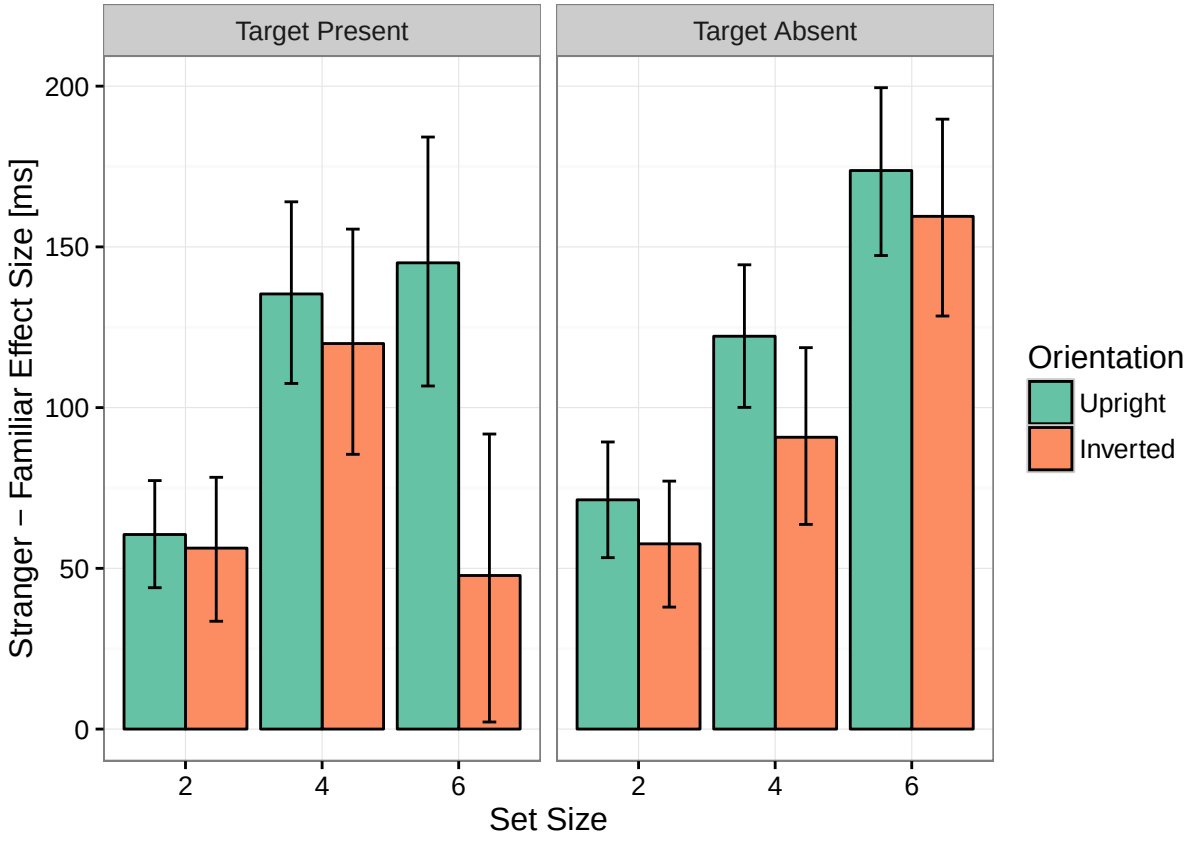
```
avg_fam <- avgs %>% filter(familiarity == 'Familiar') %>% select(-familiarity)
avg_str <- avgs %>% filter(familiarity == 'Stranger') %>% select(-familiarity)
# assert that the order is the same
tocheck <- c('orientation', 'target_presence', 'set_size')
for (check in tocheck) {
    assert_that(all(avg_fam[, check] == avg_str[, check]))
}

# get avg effect size
avg_es <- avg_str
avg_es$avg <- avg_es$avg - avg_fam$avg
```

Merge datasets and obtain data frame for plotting.

```
cis_es <- merge(avg_es, cis_es)
cis_es$orientation <- factor(cis_es$orientation,
                             levels=c('Upright', 'Inverted'))
cis_es$target_presence <- factor(cis_es$target_presence,
                                 levels=c('Target Present', 'Target Absent'))
```

Plot the data.

```
pd <- position_dodge(w=0.9)
ggplot(cis_es, aes(as.factor(set_size), avg, ymin=low, ymax=high,
                   fill=orientation, group=orientation)) +
    geom_bar(position=pd, stat='identity', color='black') +
    geom_errorbar(width=0.2, position=pd) +
    facet_grid(~target_presence) +
    scale_fill_brewer(palette='Set2') +
    theme_bw() +
    labs(x='Set Size', y='Stranger - Familiar Effect Size [ms]',
         fill='Orientation')
```



And these are the plotted data:

```
kable(select(cis_es, -familiarity), digits=0)
```

orientation	target_presence	set_size	avg	low	high
Inverted	Target Absent	2	58	38	77
Inverted	Target Absent	4	91	64	119
Inverted	Target Absent	6	160	128	190
Inverted	Target Present	2	56	34	78
Inverted	Target Present	4	120	85	156
Inverted	Target Present	6	48	2	92
Upright	Target Absent	2	71	53	89
Upright	Target Absent	4	122	100	144
Upright	Target Absent	6	174	147	200
Upright	Target Present	2	61	44	77
Upright	Target Present	4	135	108	164
Upright	Target Present	6	145	107	184

Unstandardized Effect Sizes across Set Size

Now we can compute the average effect size across set size.

```
cis_es_setsize <-
    bstrap_es %>%
```

```
    group_by(index, familiarity, orientation, target_presence) %>%
    summarise(avg=mean(avg)) %>%
    group_by(orientation, target_presence) %>%
    summarise(low=quantile(avg, alpha/2), high=quantile(avg, 1-alpha/2))

avg_es_setsize <-
    avg_es %>%
    group_by(orientation, target_presence) %>%
    summarise(avg=mean(avg))

avg_es_setsize <- merge(avg_es_setsize, cis_es_setsize)

kable(avg_es_setsize, digits=0)
```

orientation	target_presence	avg	low	high
Inverted	Target Absent	103	87	118
Inverted	Target Present	75	55	95
Upright	Target Absent	122	110	135
Upright	Target Present	114	97	131

And also the average difference between Upright faces and Inverted faces.

```
# for each bootstrap, obtain averages across set size
bstrap_es_setsize <-
    bstrap_es %>%
    group_by(index, orientation, target_presence) %>%
    summarise(avg=mean(avg)) %>%
    ungroup() %>%
    arrange(index, target_presence, orientation)

# subtract Inverted from Upright for every bootstrap and obtain CIs
cis_es_setsize_uprinv <-
    bstrap_es_setsize %>%
    ungroup() %>%
    arrange(index, target_presence, orientation) %>%
    group_by(index, target_presence) %>%
    mutate(avg = lag(avg) - avg) %>%
    na.omit() %>%
    group_by(target_presence) %>%
    summarise(low=quantile(avg, alpha/2), high=quantile(avg, 1-alpha/2))

# same for the original dataset
avg_es_setsize_uprinv <-
    avg_es_setsize %>%
    select(orientation, target_presence, avg) %>%
    ungroup() %>%
    arrange(target_presence, orientation) %>%
    group_by(target_presence) %>%
    mutate(avg = lag(avg) - avg) %>%
    select(-orientation) %>%
    na.omit()
```

```
avg_es_setsize_uprinv <-
    avg_es_setsize_uprinv %>% merge(cis_es_setsize_uprinv)

kable(avg_es_setsize_uprinv, digits=0)
```

target_presence	avg	low	high
Target Absent	20	0	40
Target Present	39	13	65