```r
knitr::opts_chunk$set(echo = TRUE)
nsims <- 1000000 #set number of simulations
library(mvtnorm)
library(afex)
library(emmeans)
library(ggplot2)
library(gridExtra)
library(reshape2)
```

Validation of Power in Mixed ANOVA

We install the functions:

```r
# Install the two functions from GitHub by running the code below:

source("https://raw.githubusercontent.com/Lakens/ANOVA_power_simulation/master/ANOVA_design.R")
source("https://raw.githubusercontent.com/Lakens/ANOVA_power_simulation/master/ANOVA_power.R")
```

Two by two ANOVA, within-between design

We can simulate a 2x2 ANOVA, both factors manipulated within participants, with a specific sample size and effect size, to achieve a desired statistical power.

As Potvin & Schutz (2000) explain, analytic procedures for a two-factor repeated measures ANOVA do not seem to exist. The main problem is quantifying the error variance (the denominator when calculating lambda or Cohen's f). Simulation based aproaches provide a solution.

We can reproduce the simulation coded by Ben Amsel

```r
knitr::opts_chunk$set(echo=TRUE, warning=FALSE, message=FALSE)

# define the parameters
mu = c(700, 670, 670, 700) # true effects (in this case, a double dissociation)
sigma = 150  # population standard deviation
rho = 0.75 # correlation between repeated measures
nsubs = 25 # how many subjects?
nsims = 5000 # how many simulation replicates?

# create 2 factors representing the 2 independent variables
cond = data.frame(
  X1 = rep(factor(letters[1:2]), nsubs * 2),
  X2 = rep(factor(letters[1:2]), nsubs, each=2))

# create a subjects factor
subject = factor(sort(rep(1:nsubs, 4)))

# combine above into the design matrix
dm = data.frame(subject, cond)
```

Build Sigma: the population variance-covariance matrix

```r
# create k x k matrix populated with sigma
sigma.mat <- rep(sigma, 4)
S <- matrix(sigma.mat, ncol=length(sigma.mat), nrow=length(sigma.mat))

# compute covariance between measures
```

```r
Sigma <- t(S) * S * rho

# put the variances on the diagonal
diag(Sigma) <- sigma^2
```

Run the simulation

```r
# stack 'nsims' individual data frames into one large data frame
df = dm[rep(seq_len(nrow(dm)), nsims), ]

# add an index column to track the simulation run
df$simID = sort(rep(seq_len(nsims), nrow(dm)))

# sample the observed data from a multivariate normal distribution
# using MASS::mvrnorm with the parameters mu and Sigma created earlier
# and bind to the existing df

require(MASS)
make.y = expression(as.vector(t(mvrnorm(nsubs, mu, Sigma))))
df$y = as.vector(replicate(nsims, eval(make.y)))

# use do(), the general purpose complement to the specialized data
# manipulation functions available in dplyr, to run the ANOVA on
# each section of the grouped data frame created by group_by

require(dplyr)
require(car)
require(broom)

mods <- df %>%
  group_by(simID) %>%
    do(model = aov(y ~ X1 * X2 + Error(subject / (X1*X2)), qr=FALSE, data = .))

# extract p-values for each effect and store in a data frame
p = data.frame(
  mods %>% do(as.data.frame(tidy(.$model[[3]])$p.value[1])),
  mods %>% do(as.data.frame(tidy(.$model[[4]])$p.value[1])),
  mods %>% do(as.data.frame(tidy(.$model[[5]])$p.value[1])))
colnames(p) = c('X1','X2','Interaction')
```

The empirical power is easy to compute, it's just the proportion of simulation runs where $p < .05$.

```r
power.res = apply(as.matrix(p), 2,
  function(x) round(mean(ifelse(x < .05, 1, 0) * 100),2))
power.res
```

```
##          X1          X2 Interaction
##        4.58        4.44       47.72
```

Visualize the distributions of p-values

```r
# plot the known effects
require(ggplot2)
require(gridExtra)

means = data.frame(cond[1:4, ], mu, SE = sigma / sqrt(nsubs))
```

```
plt1 = ggplot(means, aes(y = mu, x = X1, fill=X2)) +
  geom_bar(position = position_dodge(), stat="identity") +
  geom_errorbar(aes(ymin = mu-SE, ymax = mu+SE),
    position = position_dodge(width=0.9), size=.6, width=.3) +
  coord_cartesian(ylim=c((.7*min(mu)), 1.2*max(mu))) +
  theme_bw()

# melt the data into a ggplot friendly 'long' format
require(reshape2)
plotData <- melt(p, value.name = 'p')

# plot each of the p-value distributions on a log scale
options(scipen = 999) # 'turn off' scientific notation
plt2 = ggplot(plotData, aes(x = p)) +
    scale_x_log10(breaks=c(1, 0.05, 0.001),
                  labels=c(1, 0.05, 0.001)) +
    geom_histogram(colour = "darkblue", fill = "white") +
    geom_vline(xintercept = 0.05, colour='red') +
    facet_grid(variable ~ .) +
    labs(x = expression(Log[10]~P)) +
    theme(axis.text.x = element_text(color='black', size=7))

# arrange plots side by side and print
grid.arrange(plt1, plt2, nrow=1)
```

We can reproduce this simulation:

```
mu = c(700, 670, 670, 700) # true effects (in this case, a double dissociation)
sigma = 150  # population standard deviation
rho = 0.75 # correlation between repeated measures
n <- 25
sd <- 150
r <- 0.75
string = "2w*2w"
alpha_level <- 0.05
p_adjust = "none"
labelnames = c("age", "old", "young", "color", "blue", "red")
design_result <- ANOVA_design(string = string,
                              n = n,
                              mu = mu,
                              sd = sd,
                              r = r,
                              p_adjust = p_adjust,

                                                           labelnames = labelnames)
```

```
simulation_result <- ANOVA_power(design_result, alpha = 0.05, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##                   power effect size
## anova_age          4.56      0.0198
## anova_color        4.90      0.0190
## anova_age:color 48.24      0.1444
##
## Power and Effect sizes for contrasts
```

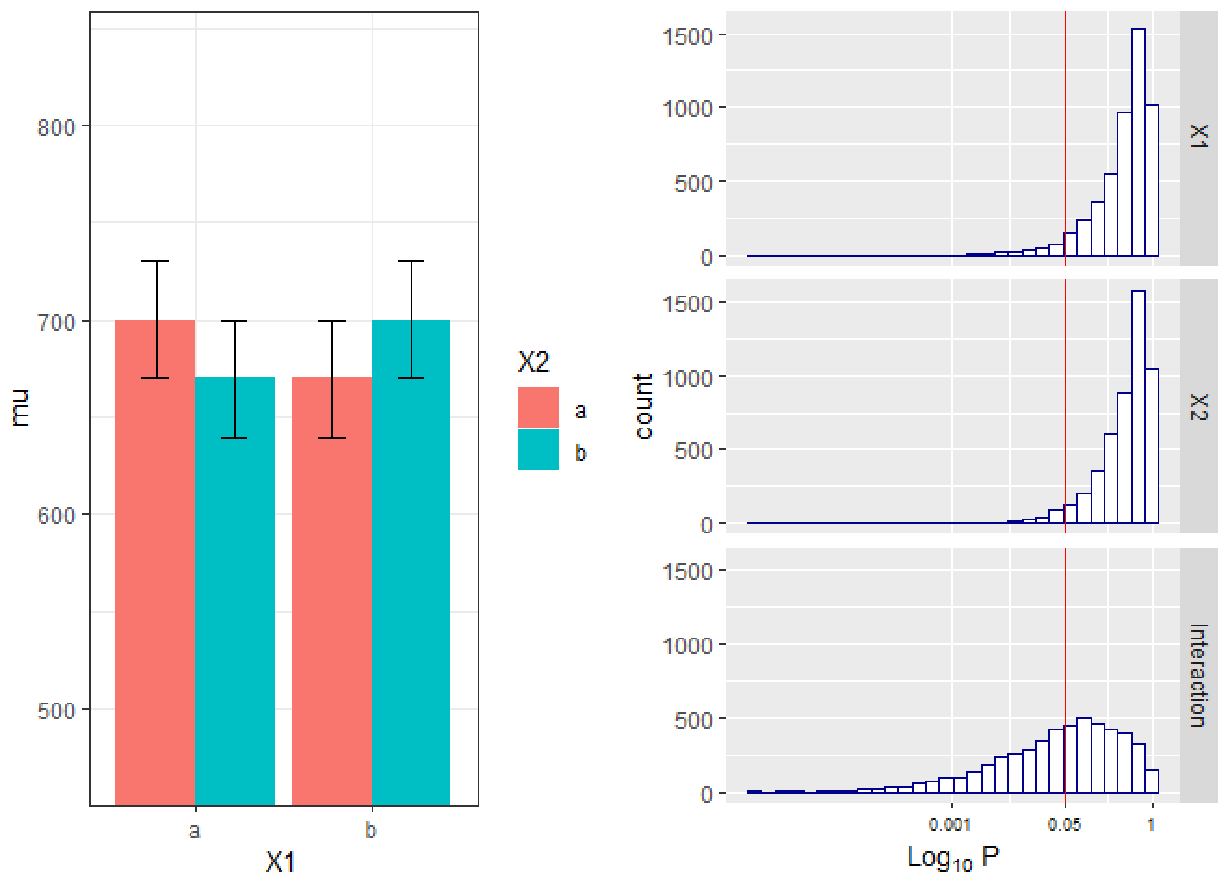
mu
800
700
600
500
a
b
X1
X2
a
b
count
1500
1000
500
0
X1
X2
Interaction
0.001
0.05
1
Log10 P

Figure 1:

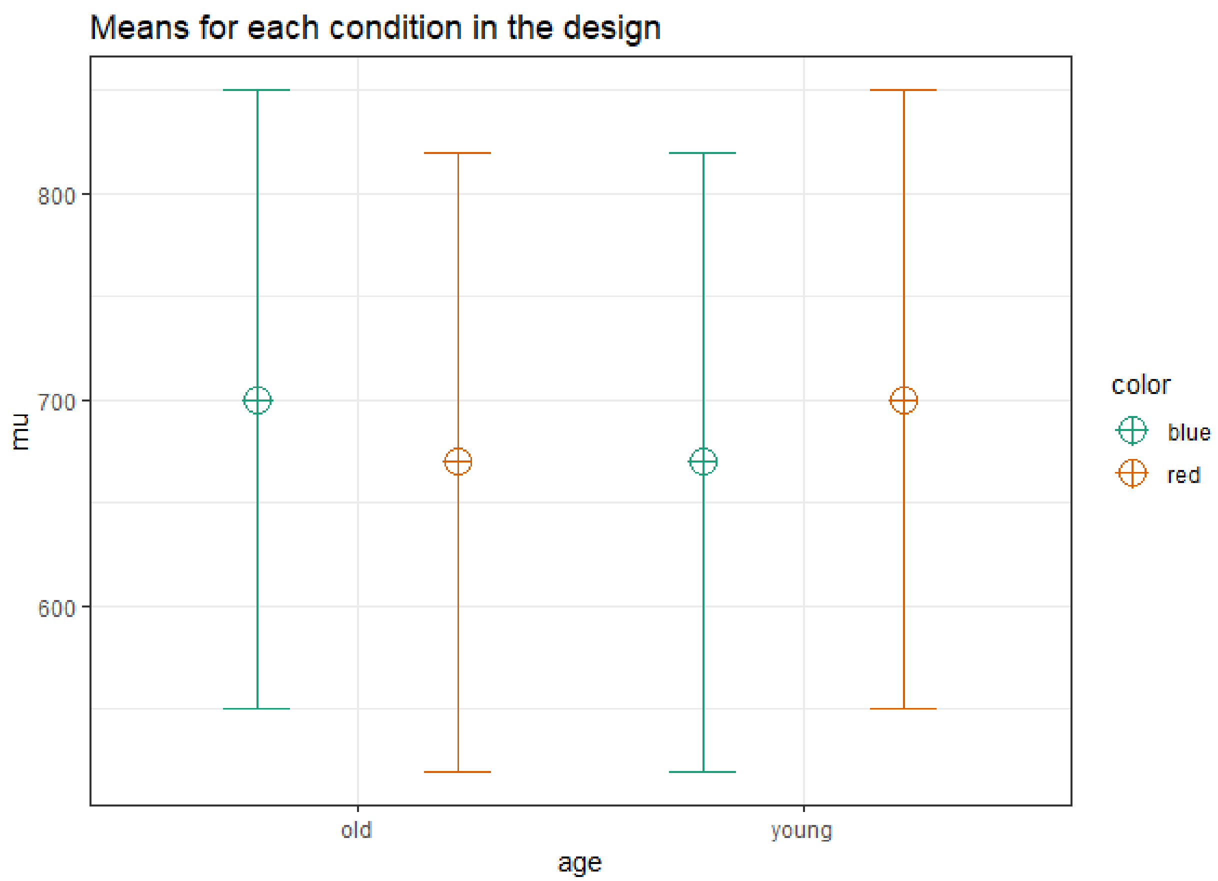
Means for each condition in the design
800
700
600
mu
old
young
age
color
blue
red

Figure 2:

```
##                                           power effect size
## p_age_old_color_blue_age_old_color_red     26.36     -0.2881
## p_age_old_color_blue_age_young_color_blue  27.40     -0.2934
## p_age_old_color_blue_age_young_color_red    5.48     -0.0008
## p_age_old_color_red_age_young_color_blue    4.70     -0.0038
## p_age_old_color_red_age_young_color_red    27.00      0.2900
## p_age_young_color_blue_age_young_color_red 26.90      0.2921
```

We see the results of the two simulations correspond.