```r
knitr::opts_chunk$set(echo = TRUE)
nsims <- 100000 #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 design

Potvin & Schutz (2000) simulate a wide range of repeated measure designs. The give an example of a 3x3 design, with the following correlation matrix:

Variances were set to 1 (so all covariance matrices in their simulations were identical). In this specific example, the white fields are related to the correlation for the A main effect (these cells have the same level for B, but different levels of A). The grey cells are related to the main effect of B (the cells have the same level of A, but different levels of B). Finally, the black cells are related to the AxB interaction (they have different levels of A and B). The diagonal (all 1) relate to cells with the same levels of A and B.

Potvin & Schulz (2000) examine power for 2x2 within ANOVA designs and develop approximations of the error variance. For a design with 2 within factors (A and B) these are:

For the main effect of A:
$sigma\_e^2 =$
$sigma^2(1 -$
$overlinerho\_A) +$
$sigma^2(q - 1)($
$overlinerho\_B -$
$overlinerho\_AB)$

For the main effectof B:
$sigma\_e^2 =$
$sigma^2(1 -$
$overlinerho\_B) +$
$sigma^2(p - 1)($
$overlinerho\_A -$
$overlinerho\_AB)$

For the interaction between A and B:
$sigma\_e^2 =$
$sigma^2(1 -$
$rho\_max) -$
$sigma^2($

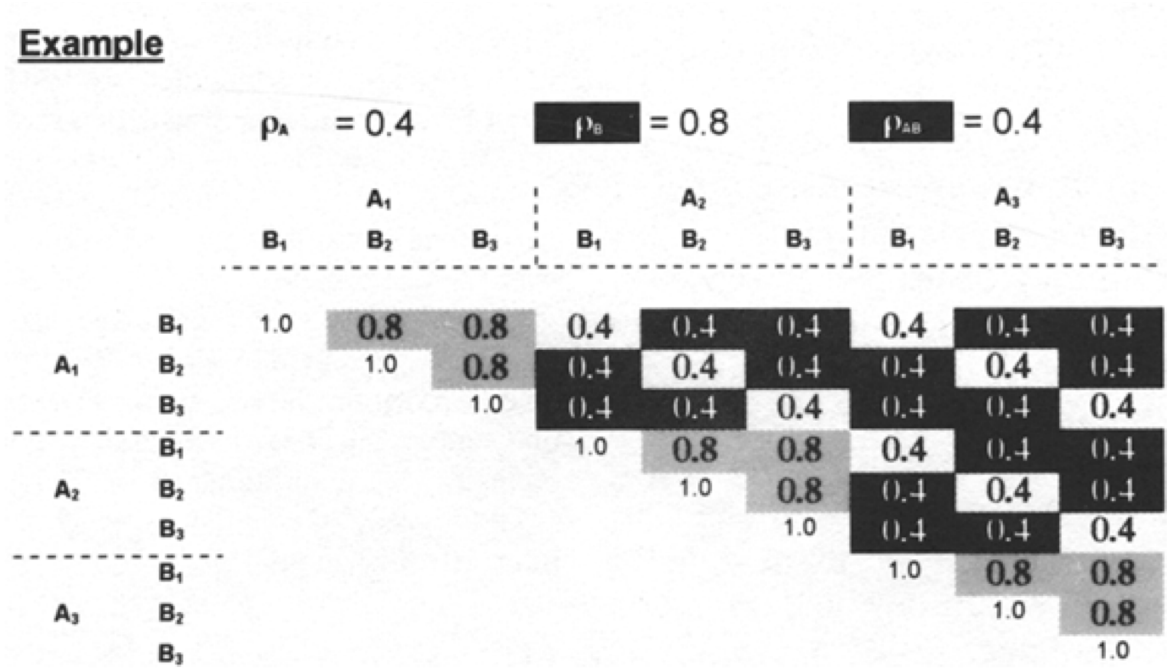

Figure 1. Representation of a correlation matrix for a 3 (A) × 3 (B) RM ANOVA: General form and numeric example. ρ_A and ρ_B represent the average correlation among the A and B (pooled) trials, respectively, and ρ_{AB} represents the average correlation among the AB coefficients having dissimilar levels.

Figure 1:

overlinerho_min −
overlinerho_AB)

Simple example: 2x2 within design

It is difficult to just come up with a positive definite covariance matrix. The best way to achieve this is to get the correlations from a pilot study. Indeed, it should be rather difficult to know which correlations to fill in without some pilot data.

We try to get the formulas in Potvin and Schutz (2000) working. **Below, I manage for the main effects, but not for the interaction**.

```
mu = c(2,1,4,2)
n <- 20
sd <- 5
r <- c(
  0.8, 0.5, 0.4,
       0.4, 0.5,
            0.8
  )

string = "2w*2w"
alpha_level <- 0.05
p_adjust = "none"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
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 = 1000)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect size
## anova_A    26.9      0.0959
## anova_B    58.7      0.2257
## anova_A:B  26.2      0.1017
##
## Power and Effect sizes for contrasts
##                      power effect size
## p_A_a1_B_b1_A_a1_B_b2  23.6     -0.3182
## p_A_a1_B_b1_A_a2_B_b1  38.9      0.4115
## p_A_a1_B_b1_A_a2_B_b2   5.8      0.0028
## p_A_a1_B_b2_A_a2_B_b1  61.9      0.5631
## p_A_a1_B_b2_A_a2_B_b2  13.8      0.2072
## p_A_a2_B_b1_A_a2_B_b2  72.7     -0.6406
```

Result simulation after 100000 simulations

simulation_result <- ANOVA_power(design_result, alpha = 0.05, nsims = 100000) Power and Effect sizes for ANOVA tests power effect size anova_A 26.849 0.0984 anova_B 64.091 0.2452 anova_A:B 26.875 0.0983

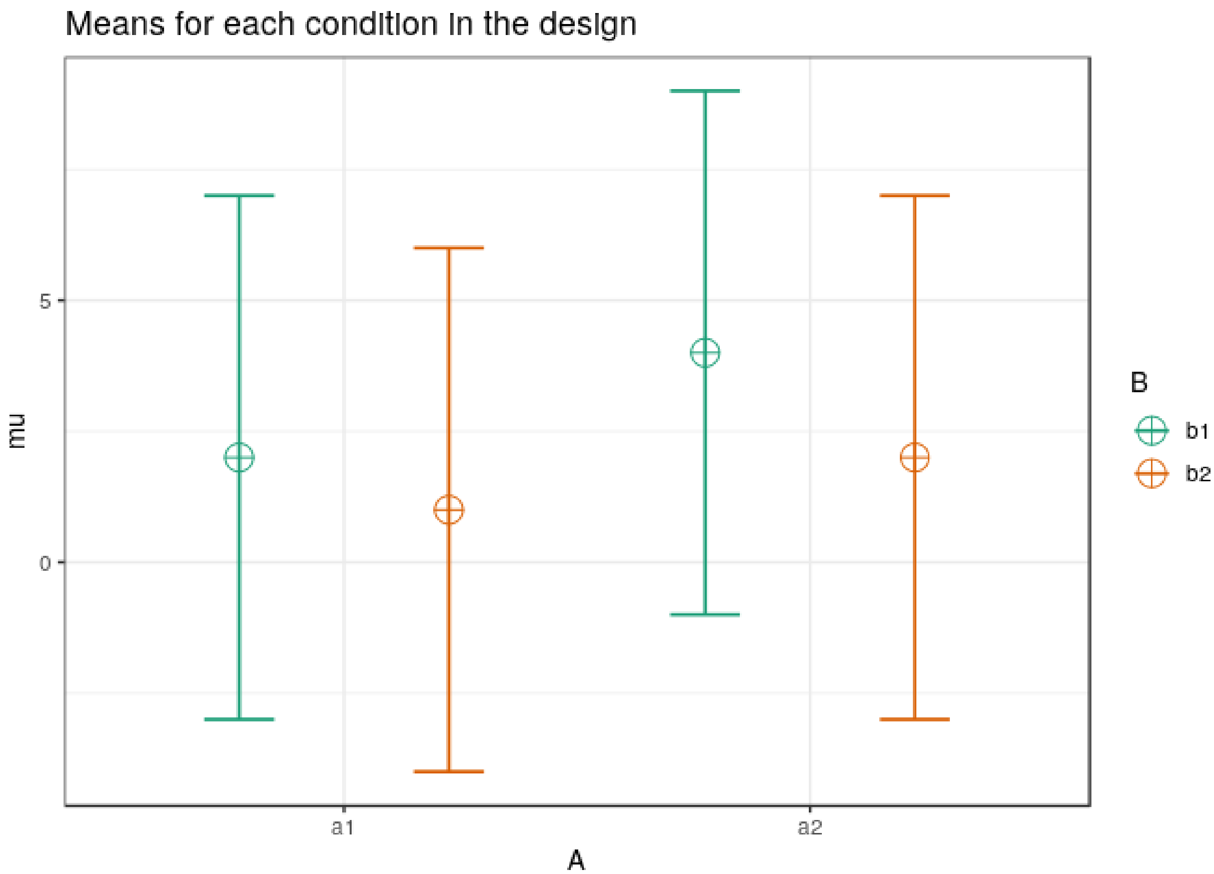

Figure 2:

Power and Effect sizes for contrasts power effect size p_A_a1_B_b1_A_a1_B_b2 27.052 -0.3298 p_A_a1_B_b1_A_a2_B_b1 39.637 0.4162 p_A_a1_B_b1_A_a2_B_b2 4.983 -0.0005 p_A_a1_B_b2_A_a2_B_b1 64.252 0.5699 p_A_a1_B_b2_A_a2_B_b2 13.479 0.2077 p_A_a2_B_b1_A_a2_B_b2 76.622 -0.6597

We can try to use the formula in Potvin & Schutz (2000).

```
k <- 1 #one group (because all factors are within)
rho_A <- 0.5 #mean r for factor A
rho_B <- 0.8 #mean r for factor B
rho_AB <- 0.4 #mean r for factor AB
alpha <- 0.05
sigma <- sd

m_A <- 2 #levels factor A
variance_e_A <- sigma^2 * (1 - rho_A) + sigma^2 * (m_A - 1) * (rho_B - rho_AB) #Variance A
variance_e_A
```

```
## [1] 22.5
```

```
m_B <- 2 #levels factor B
variance_e_B <- sigma^2 * (1 - rho_B) + sigma^2 * (m_B - 1) * (rho_A - rho_AB) #Variance B
variance_e_B
```

```
## [1] 7.5
```

```
variance_e_AB <- sigma^2 * (1 - max(rho_A, rho_B)) - sigma^2 * (min(rho_A, rho_B) - rho_AB) #Variance A
variance_e_AB
```

```
## [1] 2.5
```

```
mean_mat <- t(matrix(mu, nrow = m_B,ncol = m_A)) #Create a mean matrix
mean_mat
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    4    2
```

```
# Potving & Schutz, 2000, formula 2, p. 348
# For main effect A
lambda_A <- n * m_A * sum((rowMeans(mean_mat)-mean(rowMeans(mean_mat)))^2)/variance_e_A
lambda_A
```

```
## [1] 2
```

```
df1 <- (m_A - 1) #calculate degrees of freedom 1 - ignoring the * e sphericity correction
df2 <- (n - k) * (m_A - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_A <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_A,
```

```
         lower.tail = FALSE)

lambda_B <- n * m_B * sum((colMeans(mean_mat)-mean(colMeans(mean_mat)))^2)/variance_e_B
lambda_B
```

```
## [1] 6
```

```
df1 <- (m_B - 1) #calculate degrees of freedom 1
df2 <- (n - k) * (m_B - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_B <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_B,
          lower.tail = FALSE)

pow_A
```

```
## [1] 0.2691752
```

```
pow_B
```

```
## [1] 0.6422587
```

We see the 26.9 and 64.2 correspond to the results of the simulation quite closely.

```
#This (or the variance calculation above) does not work.
lambda_AB <- n * sum((mean_mat-rowMeans(mean_mat)-colMeans(mean_mat)+mean(mean_mat))^2) / variance_e_AB
lambda_AB
```

```
## [1] 38
```

```
df1 <- (m_A - 1)*(m_B - 1)  #calculate degrees of freedom 1
df2 <- (n - k) * (m_A - 1) * (m_B - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_AB,
          lower.tail = FALSE)
pow
```

```
## [1] 0.9999458
```

Maybe the simulation is not correct for the interaction, or the formula is not correctly programmed.

Testing a variation

Let's see if changing the means changes the patterns as we would expect.

```
mu = c(2,1,4,2)
n <- 20
sd <- 5
r <- c(
  0.8, 0.8, 0.8,
       0.8, 0.8,
            0.8
  )

string = "2w*2w"
alpha_level <- 0.05
p_adjust = "none"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
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 = 1000)
```

```
## Power and Effect sizes for ANOVA tests
##             power effect size
## anova_A      82.9      0.3240
## anova_B      80.8      0.3284
## anova_A:B    15.0      0.0539
##
## Power and Effect sizes for contrasts
##                         power effect size
## p_A_a1_B_b1_A_a1_B_b2   25.7     -0.3224
## p_A_a1_B_b1_A_a2_B_b1   75.6      0.6565
## p_A_a1_B_b1_A_a2_B_b2    3.4      0.0048
## p_A_a1_B_b2_A_a2_B_b1   98.0      0.9828
## p_A_a1_B_b2_A_a2_B_b2   27.7      0.3272
## p_A_a2_B_b1_A_a2_B_b2   76.3     -0.6556
```

Check against the formulas

We again use the formula in Potvin & Schutz (2000).

```
k <- 1 #one group (because all factors are within)
rho_A <- 0.8 #mean r for factor A
rho_B <- 0.8 #mean r for factor B
rho_AB <- 0.8 #mean r for factor AB
alpha <- 0.05
```

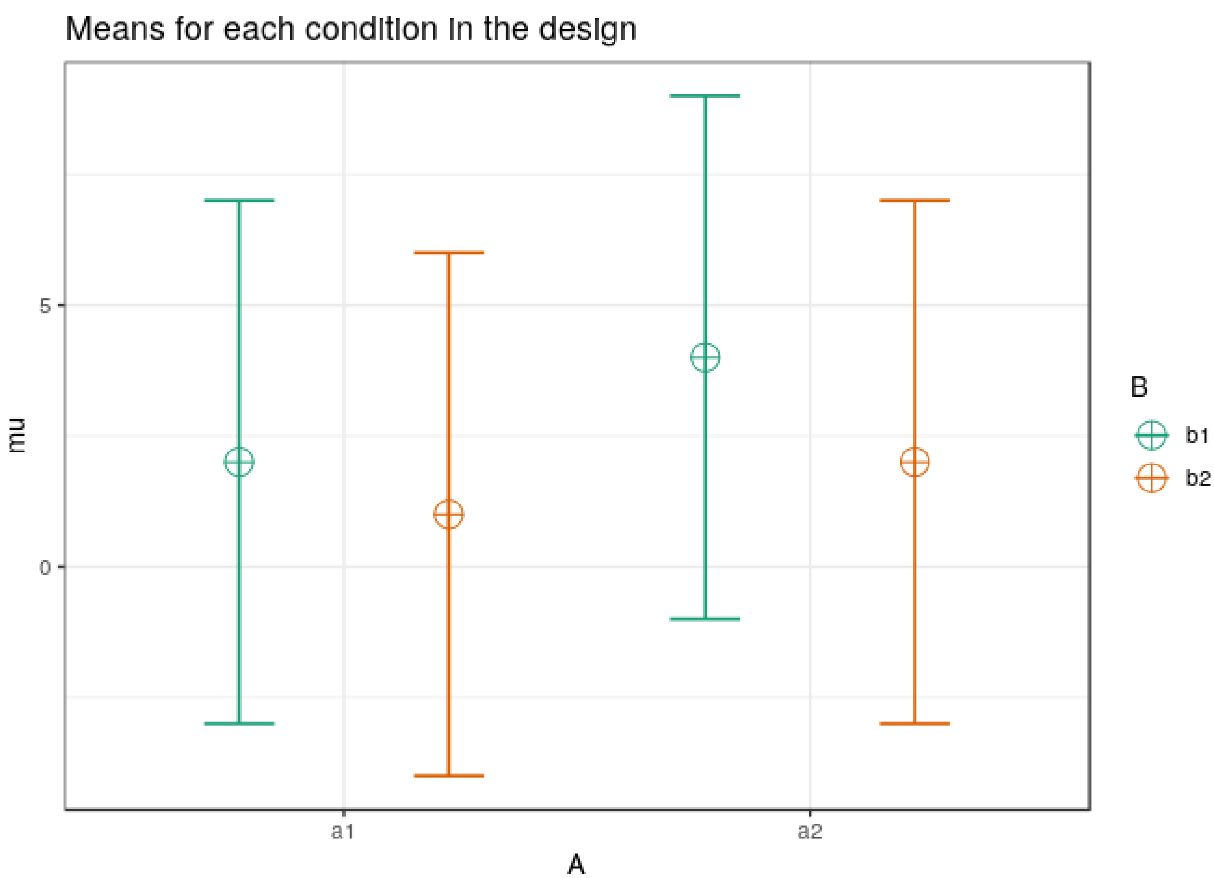

Figure 3:

```r
sigma <- sd

m_A <- 2 #levels factor A
variance_e_A <- sigma^2 * (1 - rho_A) + sigma^2 * (m_A - 1) * (rho_B - rho_AB) #Variance A
variance_e_A
```

```
## [1] 5
```

```r
m_B <- 2 #levels factor B
variance_e_B <- sigma^2 * (1 - rho_B) + sigma^2 * (m_B - 1) * (rho_A - rho_AB) #Variance B
variance_e_B
```

```
## [1] 5
```

```r
variance_e_AB <- sigma^2 * (1 - max(rho_A, rho_B)) - sigma^2 * (min(rho_A, rho_B) - rho_AB) #Variance A
variance_e_AB
```

```
## [1] 5
```

```r
mean_mat <- t(matrix(mu, nrow = m_B,ncol = m_A)) #Create a mean matrix
mean_mat
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    4    2
```

```r
# Potving & Schutz, 2000, formula 2, p. 348
# For main effect A
lambda_A <- n * m_A * sum((rowMeans(mean_mat)-mean(rowMeans(mean_mat)))^2)/variance_e_A
lambda_A
```

```
## [1] 9
```

```r
df1 <- (m_A - 1) #calculate degrees of freedom 1 - ignoring the * e sphericity correction
df2 <- (n - k) * (m_A - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_A <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_A,
          lower.tail = FALSE)

lambda_B <- n * m_B * sum((colMeans(mean_mat)-mean(colMeans(mean_mat)))^2)/variance_e_B
lambda_B
```

```
## [1] 9
```

```r
df1 <- (m_B - 1) #calculate degrees of freedom 1
df2 <- (n - k) * (m_B - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
```

```
                 df2,
                 lower.tail=FALSE)

pow_B <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_B,
          lower.tail = FALSE)

pow_A
```

```
## [1] 0.8120654
```

```
pow_B
```

```
## [1] 0.8120654
```

We see the simulated values are again close to the predicted 81.2%

```
mu = c(2,1,5,3)
n <- 20
sd <- 5
r <- c(
  0.8, 0.8, 0.8,
       0.8, 0.8,
            0.8
  )

string = "2w*2w"
alpha_level <- 0.05
p_adjust = "none"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")

k <- 1 #one group (because all factors are within)
rho_A <- 0.8 #mean r for factor A
rho_B <- 0.8 #mean r for factor B
rho_AB <- 0.8 #mean r for factor AB
alpha <- 0.05
sigma <- sd

m_A <- 2 #levels factor A
variance_e_A <- sigma^2 * (1 - rho_A) + sigma^2 * (m_A - 1) * (rho_B - rho_AB) #Variance A
variance_e_A
```

```
## [1] 5
```

```
m_B <- 2 #levels factor B
variance_e_B <- sigma^2 * (1 - rho_B) + sigma^2 * (m_B - 1) * (rho_A - rho_AB) #Variance B
variance_e_B
```

```
## [1] 5
```

```
variance_e_AB <- sigma^2 * (1 - max(rho_A, rho_B)) - sigma^2 * (min(rho_A, rho_B) - rho_AB) #Variance A
variance_e_AB
```

```
## [1] 5
```

```r
mean_mat <- t(matrix(mu, nrow = m_B,ncol = m_A)) #Create a mean matrix
mean_mat
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    5    3
```

```r
# Potving & Schutz, 2000, formula 2, p. 348
# For main effect A
lambda_A <- n * m_A * sum((rowMeans(mean_mat)-mean(rowMeans(mean_mat)))^2)/variance_e_A
lambda_A
```

```
## [1] 25
```

```r
df1 <- (m_A - 1) #calculate degrees of freedom 1 - ignoring the * e sphericity correction
df2 <- (n - k) * (m_A - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_A <- pf(qf(alpha, #power
              df1,
              df2,
              lower.tail = FALSE),
           df1,
           df2,
           lambda_A,
           lower.tail = FALSE)

lambda_B <- n * m_B * sum((colMeans(mean_mat)-mean(colMeans(mean_mat)))^2)/variance_e_B
lambda_B
```

```
## [1] 9
```

```r
df1 <- (m_B - 1) #calculate degrees of freedom 1
df2 <- (n - k) * (m_B - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_B <- pf(qf(alpha, #power
              df1,
              df2,
              lower.tail = FALSE),
           df1,
           df2,
           lambda_B,
           lower.tail = FALSE)

pow_A
```

```
## [1] 0.9972347
```

```r
pow_B
```

```
## [1] 0.8120654
```

Let's see if changing the means changes the patterns as we would expect.

```r
mu = c(2,1,4,2)
n <- 20
sd <- 5
r <- c(
  0.8, 0.8, 0.8,
       0.8, 0.8,
            0.8
  )

string = "2w*2w"
alpha_level <- 0.05
p_adjust = "none"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
design_result <- ANOVA_design(string = string,
                              n = n,
                              mu = mu,
                              sd = sd,
                              r = r,
                              p_adjust = p_adjust,
                              labelnames = labelnames)
```

```r
simulation_result <- ANOVA_power(design_result, alpha = 0.05, nsims = 1000)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect size
## anova_A    82.0      0.3280
## anova_B    82.6      0.3348
## anova_A:B  15.5      0.0625
##
## Power and Effect sizes for contrasts
##                         power effect size
## p_A_a1_B_b1_A_a1_B_b2  28.2     -0.3342
## p_A_a1_B_b1_A_a2_B_b1  78.2      0.6570
## p_A_a1_B_b1_A_a2_B_b2   4.5     -0.0049
## p_A_a1_B_b2_A_a2_B_b1  98.6      0.9966
## p_A_a1_B_b2_A_a2_B_b2  26.2      0.3259
## p_A_a2_B_b1_A_a2_B_b2  79.1     -0.6644
```

We can again check against the formula in Potvin & Schutz (2000).

```r
k <- 1 #one group (because all factors are within)
rho_A <- 0.8 #mean r for factor A
rho_B <- 0.8 #mean r for factor B
rho_AB <- 0.8 #mean r for factor AB
alpha <- 0.05
sigma <- sd

m_A <- 2 #levels factor A
variance_e_A <- sigma^2 * (1 - rho_A) + sigma^2 * (m_A - 1) * (rho_B - rho_AB) #Variance A
variance_e_A
```

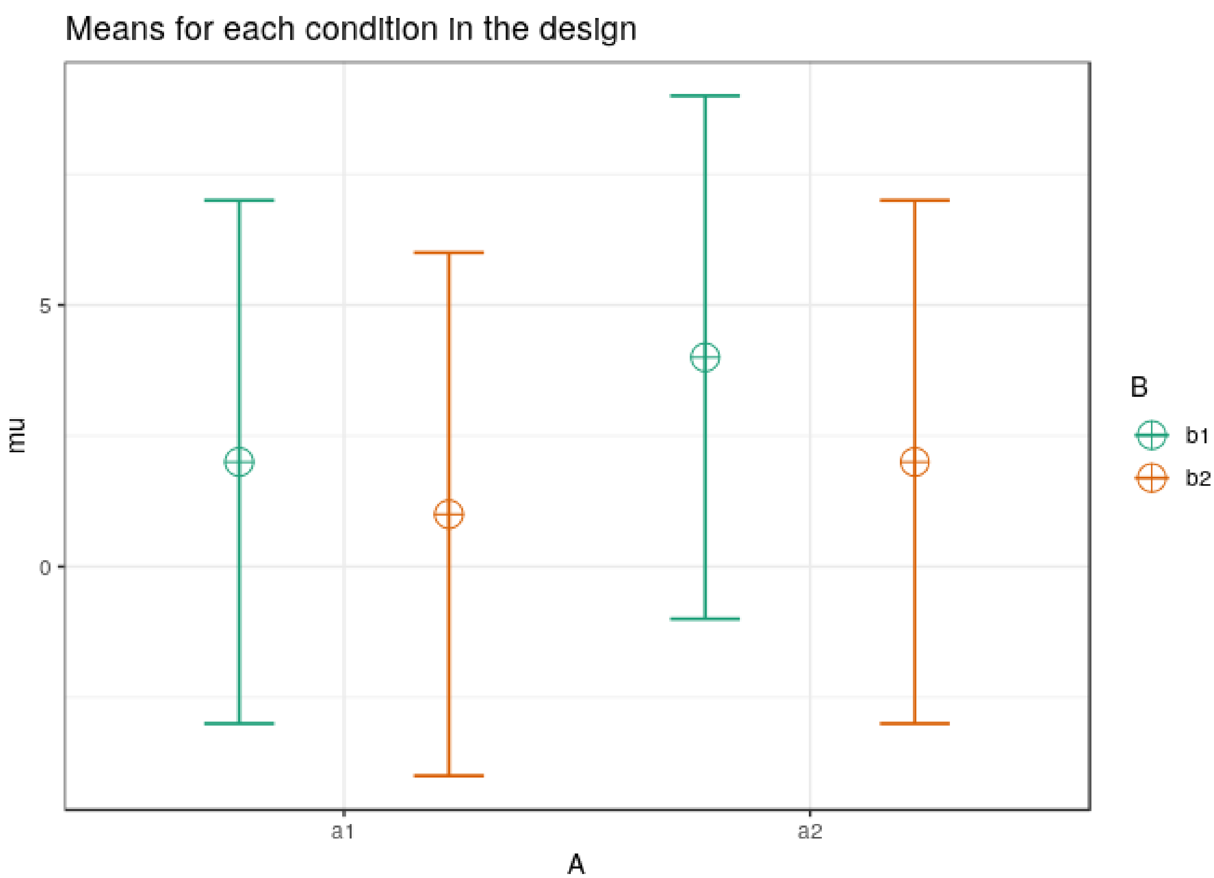

Figure 4:

```
## [1] 5
```

```
m_B <- 2 #levels factor B
variance_e_B <- sigma^2 * (1 - rho_B) + sigma^2 * (m_B - 1) * (rho_A - rho_AB) #Variance B
variance_e_B
```

```
## [1] 5
```

```
variance_e_AB <- sigma^2 * (1 - max(rho_A, rho_B)) - sigma^2 * (min(rho_A, rho_B) - rho_AB) #Variance A
variance_e_AB
```

```
## [1] 5
```

```
mean_mat <- t(matrix(mu, nrow = m_B,ncol = m_A)) #Create a mean matrix
mean_mat
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    4    2
```

```
# Potving & Schutz, 2000, formula 2, p. 348
# For main effect A
lambda_A <- n * m_A * sum((rowMeans(mean_mat)-mean(rowMeans(mean_mat)))^2)/variance_e_A
lambda_A
```

```
## [1] 9
```

```
df1 <- (m_A - 1) #calculate degrees of freedom 1 - ignoring the * e sphericity correction
df2 <- (n - k) * (m_A - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_A <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_A,
          lower.tail = FALSE)

lambda_B <- n * m_B * sum((colMeans(mean_mat)-mean(colMeans(mean_mat)))^2)/variance_e_B
lambda_B
```

```
## [1] 9
```

```
df1 <- (m_B - 1) #calculate degrees of freedom 1
df2 <- (n - k) * (m_B - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_B <- pf(qf(alpha, #power
             df1,
             df2,
```

```
              lower.tail = FALSE),
          df1,
          df2,
          lambda_B,
          lower.tail = FALSE)

pow_A
```

```
## [1] 0.8120654
```

```
pow_B
```

```
## [1] 0.8120654
```

We see the simulated values are again close to the predicted 81.2%

Let's see if changing the correlations changes the patterns as we would expect.

```
mu = c(2,1,4,2)
n <- 20
sd <- 5
r <- c(
  0.5, 0.5, 0.5,
       0.5, 0.5,
            0.5
  )

string = "2w*2w"
alpha_level <- 0.05
p_adjust = "none"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
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 = 1000)
```

```
## Power and Effect sizes for ANOVA tests
##             power effect size
## anova_A      44.3      0.1608
## anova_B      41.9      0.1560
## anova_A:B    10.3      0.0360
##
## Power and Effect sizes for contrasts
##                          power effect size
## p_A_a1_B_b1_A_a1_B_b2    13.1     -0.1998
## p_A_a1_B_b1_A_a2_B_b1    40.4      0.4225
## p_A_a1_B_b1_A_a2_B_b2     6.4      0.0130
## p_A_a1_B_b2_A_a2_B_b1    72.8      0.6276
## p_A_a1_B_b2_A_a2_B_b2    14.5      0.2150
## p_A_a2_B_b1_A_a2_B_b2    37.9     -0.4121
```

We again check against the formula in Potvin & Schutz (2000).

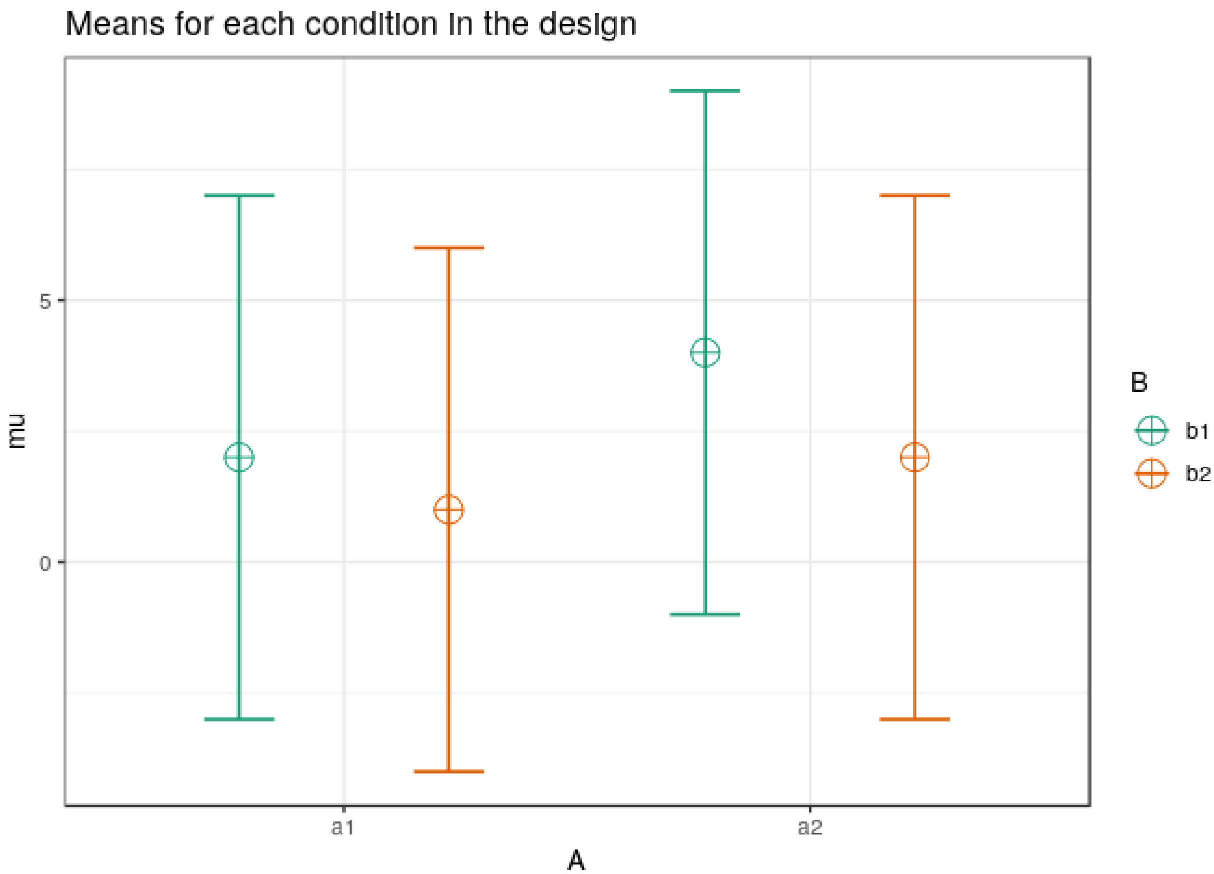

Figure 5:

```r
k <- 1 #one group (because all factors are within)
rho_A <- 0.5 #mean r for factor A
rho_B <- 0.5 #mean r for factor B
rho_AB <- 0.5 #mean r for factor AB
alpha <- 0.05
sigma <- sd

m_A <- 2 #levels factor A
variance_e_A <- sigma^2 * (1 - rho_A) + sigma^2 * (m_A - 1) * (rho_B - rho_AB) #Variance A
variance_e_A
```

```
## [1] 12.5
```

```r
m_B <- 2 #levels factor B
variance_e_B <- sigma^2 * (1 - rho_B) + sigma^2 * (m_B - 1) * (rho_A - rho_AB) #Variance B
variance_e_B
```

```
## [1] 12.5
```

```r
variance_e_AB <- sigma^2 * (1 - max(rho_A, rho_B)) - sigma^2 * (min(rho_A, rho_B) - rho_AB) #Variance A
variance_e_AB
```

```
## [1] 12.5
```

```r
mean_mat <- t(matrix(mu, nrow = m_B,ncol = m_A)) #Create a mean matrix
mean_mat
```

```
##      [,1] [,2]
## [1,]    2    1
## [2,]    4    2
```

```r
# Potving & Schutz, 2000, formula 2, p. 348
# For main effect A
lambda_A <- n * m_A * sum((rowMeans(mean_mat)-mean(rowMeans(mean_mat)))^2)/variance_e_A
lambda_A
```

```
## [1] 3.6
```

```r
df1 <- (m_A - 1) #calculate degrees of freedom 1 - ignoring the * e sphericity correction
df2 <- (n - k) * (m_A - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_A <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_A,
          lower.tail = FALSE)

lambda_B <- n * m_B * sum((colMeans(mean_mat)-mean(colMeans(mean_mat)))^2)/variance_e_B
lambda_B
```

```
## [1] 3.6
```

```r
df1 <- (m_B - 1) #calculate degrees of freedom 1
df2 <- (n - k) * (m_B - 1) #calculate degrees of freedom 2
F_critical <- qf(alpha, # critical F-vaue
                 df1,
                 df2,
                 lower.tail=FALSE)

pow_B <- pf(qf(alpha, #power
             df1,
             df2,
             lower.tail = FALSE),
          df1,
          df2,
          lambda_B,
          lower.tail = FALSE)

pow_A
```

```
## [1] 0.437076
```

```r
pow_B
```

```
## [1] 0.437076
```