

Validation of Superpower

Aaron Caldwell & Daniël Lakens

2019-09-12

Contents

1	Introduction	5
2	One-way ANOVA Part 1	7
2.1	Validation of Power in One-Way ANOVA	7
2.2	Two conditions	7
3	One-way ANOVA Part 2	17
3.1	Validation of Power in One-Way ANOVA with Brysbaert example	17
3.2	Power for simple effects	19
4	One-way ANOVA Part 3	25
4.1	Three conditions replication	26
4.2	Variation 1	27
4.3	Three conditions replication	29
4.4	Variation 2	30
4.5	Three conditions replication	31
5	Repeated Measures-ANOVA Part 1	33
5.1	Two conditions, medium effect size	33
6	Repeated Measures-ANOVA Part 2	43
6.1	The relation between Cohen's f and Cohen's d	43
6.2	Three within conditions, medium effect size	44
7	Repeated Measures-ANOVA Part 3	51
7.1	Reproducing Brysbaert	52
7.2	Reproducing Brysbaert Variation 1 Changing Correlation	55
8	Mixed ANOVA Part 1	59
8.1	Two by two ANOVA, within-between design	59
8.2	Two by two ANOVA, within-between design Variation 1	62

9 Mixed ANOVA Part 2	65
9.1 Two by two ANOVA, within-within design	65
9.2 Examine variation of means and correlation	70
10 Mixed ANOVA Part 3	75
10.1 Two by two ANOVA, within design	75
11 Simple example: 2x2 within design	77
12 Error Control in Exploratory ANOVA	83
13 Power in Interactions	91
14 Analytic Power Functions	103
14.1 One-Way Between Subject ANOVA	103
14.2 Two-way Between Subject Interaction	106
14.3 3x3 Between Subject ANOVA	108
14.4 Two by two ANOVA, within design	110
15 Power curves	115
16 Explore increase in effect size for moderated interactions.	119
16.1 Explore increase in effect size for cross-over interactions.	143
16.2 Explore increase in correlation in moderated interactions.	167
16.3 Increasing correlation in on factor decreases power in second factor	191

This is a compilation of validation documents for **Superpower** written in **Markdown** and compiled by **Bookdown**.

Chapter 1

Introduction

Below we have included various examples of the performance of **Superpower** against other R packages and statistical programs (such as GPower).

Chapter 2

One-way ANOVA Part 1

2.1 Validation of Power in One-Way ANOVA

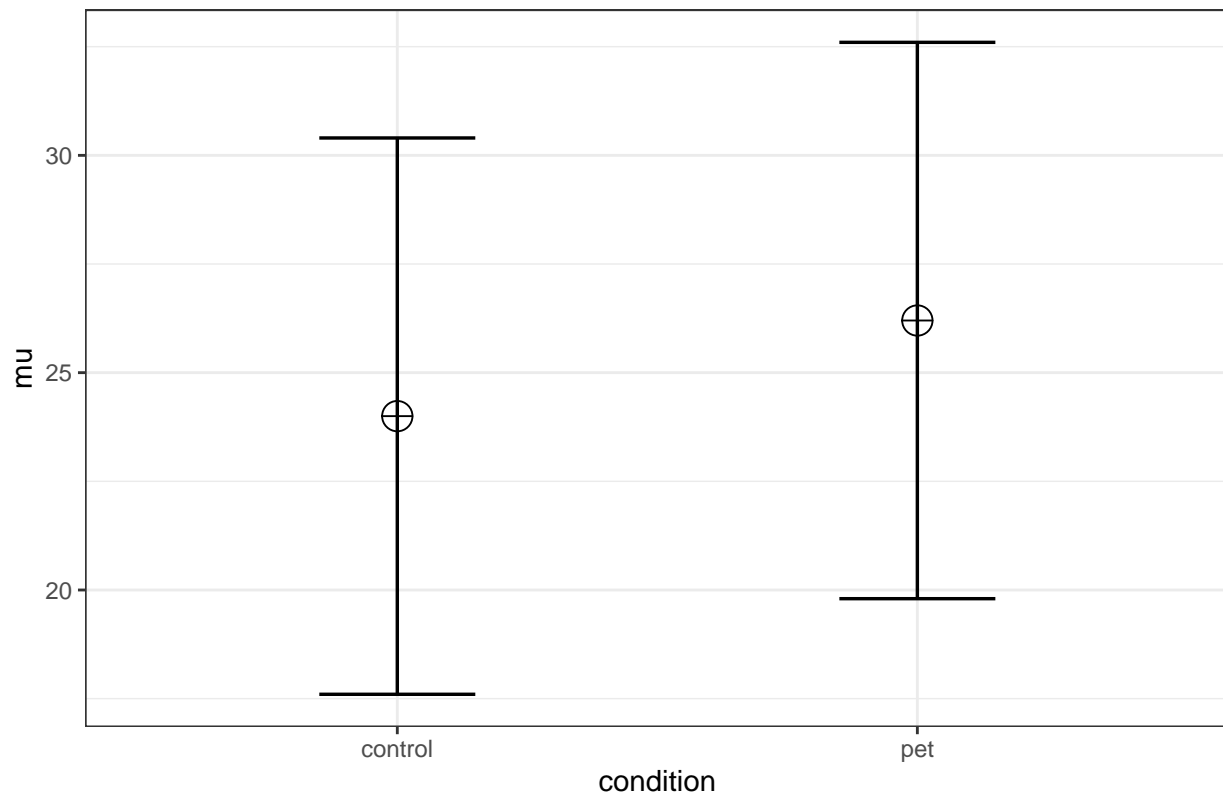
Using the formula also used in Albers & Lakens (2018), we can determine the means that should yield a specified effect sizes (expressed in Cohen's f). Eta-squared (identical to partial eta-squared for One-Way ANOVA's) has benchmarks of .0099, .0588, and .1379 for small, medium, and large effect sizes (Cohen, 1988). Although these benchmarks are quite random, and researchers should only use such benchmarks for power analyses as a last resort, we will demonstrate a-priori power analysis for these values.

2.2 Two conditions

Imagine we aim to design a study to test the hypothesis that giving people a pet to take care of will increase their life satisfaction. We have a control condition, and a condition where people get a pet, and randomly assign participants to either condition. We can simulate a One-Way ANOVA with a specified alpha, sample size, and effect size, on see the statistical power we would have for the ANOVA and the follow-up comparisons. We expect pets to increase life-satisfaction compared to the control condition. Based on work by Pavot and Diener (1993) we believe that we can expect responses on the life-satisfaction scale to have a mean of approximately 24 in our population, with a standard deviation of 6.4. We expect having a pet increases life satisfaction with approximately 2.2 scale points for participants who get a pet. 200 participants in total, with 100 participants in each condition. But before we proceed with the data collection, we examine the statistical power our design would have to detect the differences we predict.

```
string <- "2b"
n <- 100
# We are thinking of running 50 people in each condition
mu <- c(24, 26.2)
# Enter means in the order that matches the labels below.
# In this case, control, cat, dog.
sd <- 6.4
labelnames <- c("condition", "control", "pet") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```

Means for each condition in the design



```
alpha_level <- 0.05
# You should think carefully about how to justify your alpha level.
# We will give some examples later, but for now, use 0.05.
ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_condition 66      0.03461
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_condition_control_condition_pet 66      0.3465
```

The result shows that we have exactly the same power for the ANOVA, as we have for the t -test. This is because when there are only two groups, these tests are mathematically identical. In a study with 100 participants, we would have quite low power (around 67.7%). An ANOVA with 2 groups is identical to a t -test. For our example, Cohen's d (the standardized mean difference) is $2.2/6.4$, or $d = 0.34375$ for the difference between the control condition and pets, which we can use to easily compute the expected power for these simple comparisons using the `pwr` package.

```
pwr.t.test(d = 2.2/6.4,
           n = 100,
           sig.level = 0.05,
           type="two.sample",
           alternative="two.sided")$power
```



```
## [1] 0.6768572
```

We can also directly compute Cohen's f from Cohen's d for two groups, as Cohen (1988) describes, because $f = 1/2d$. So $f = 0.5 * 0.34375 = 0.171875$. And indeed, power analysis using the `pwr` package yields the same result using the `pwr.anova.test` as the `power.t.test`.

```
K <- 2
n <- 100
f <- 0.171875
pwr.anova.test(n = n,
               k = K,
               f = f,
               sig.level = alpha_level)$power
```

```
## [1] 0.6768572
```

This analysis tells us that running the study with 100 participants in each condition is too likely to *not* yield a significant test result, even if our expected pattern of differences is true. This is not optimal.

Let's mathematically explore which pattern of means we would need to expect to have 90% power for the ANOVA with 50 participants in each group. We can use the `pwr` package in R to compute a sensitivity analysis that tells us the effect size, in Cohen's f , that we are able to detect with 3 groups and 50 participants in each group, in order to achieve 90% power with an alpha level of 5%.

```
K <- 2
n <- 100
sd <- 6.4
r <- 0
#Calculate f when running simulation
f <- pwr.anova.test(n = n,
                   k = K,
                   power = 0.9,
                   sig.level = alpha_level)$f
f
```

```
## [1] 0.2303587
```

This sensitivity analysis shows we have 90% power in our planned design to detect effects of Cohen's f of 0.2303587. Benchmarks by Cohen (1988) for small, medium, and large Cohen's f values are 0.1, 0.25, and 0.4, which correspond to eta-squared values of small (.0099), medium (.0588), and large (.1379), in line with $d = .2$, $.5$, or $.8$. So, at least based on these benchmarks, we have 90% power to detect effects that are slightly below a medium effect benchmark.

```
f2 <- f^2
ES <- f2/(f2+1)
ES
```

```
## [1] 0.0503911
```

Expressed in eta-squared, we can detect values of eta-squared = 0.05 or larger.

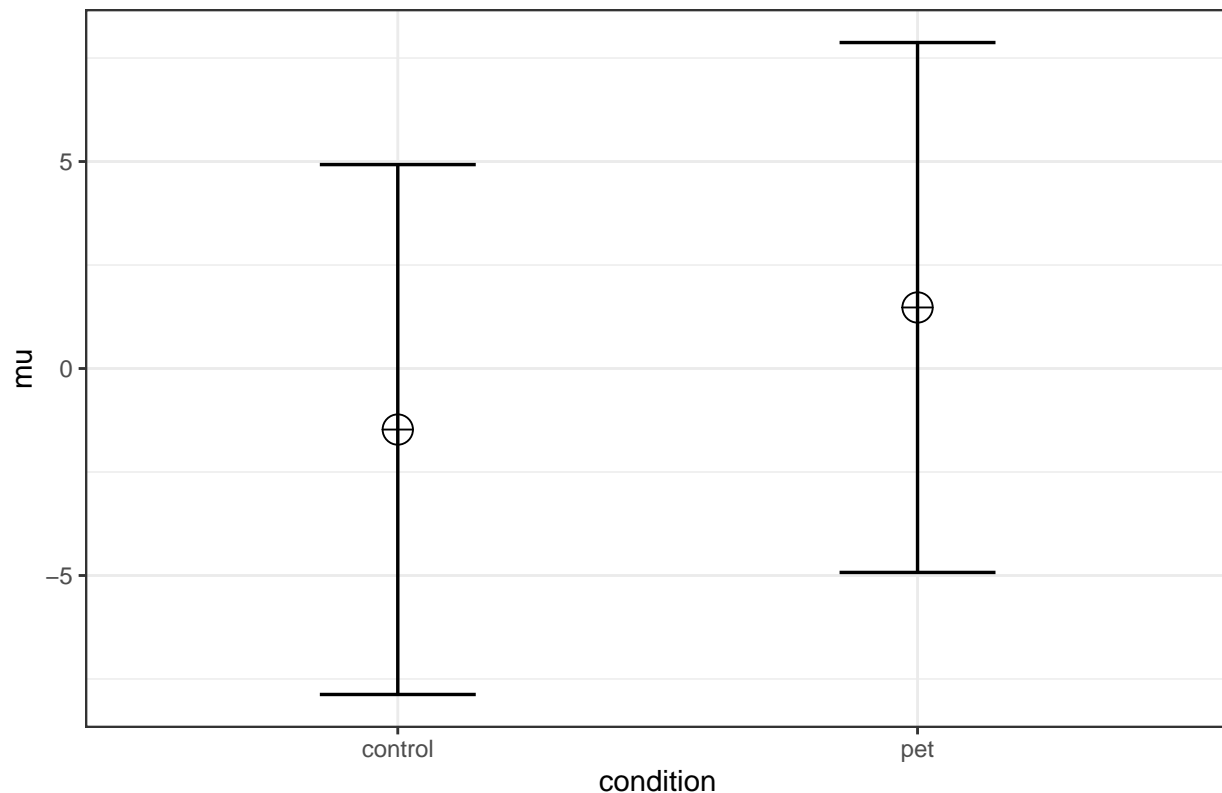
```
mu <- mu_from_ES(K = K, ES = ES)
mu <- mu * sd
mu
```

```
## [1] -1.474295  1.474295
```

We can compute a pattern of means, given a standard deviation of 6.4, that would give us an effect size of $f = 0.23$, or eta-squared of 0.05. We should be able to accomplish this if the means are -1.474295 and 1.474295. We can use these values to confirm the ANOVA has 90% power.

```
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```

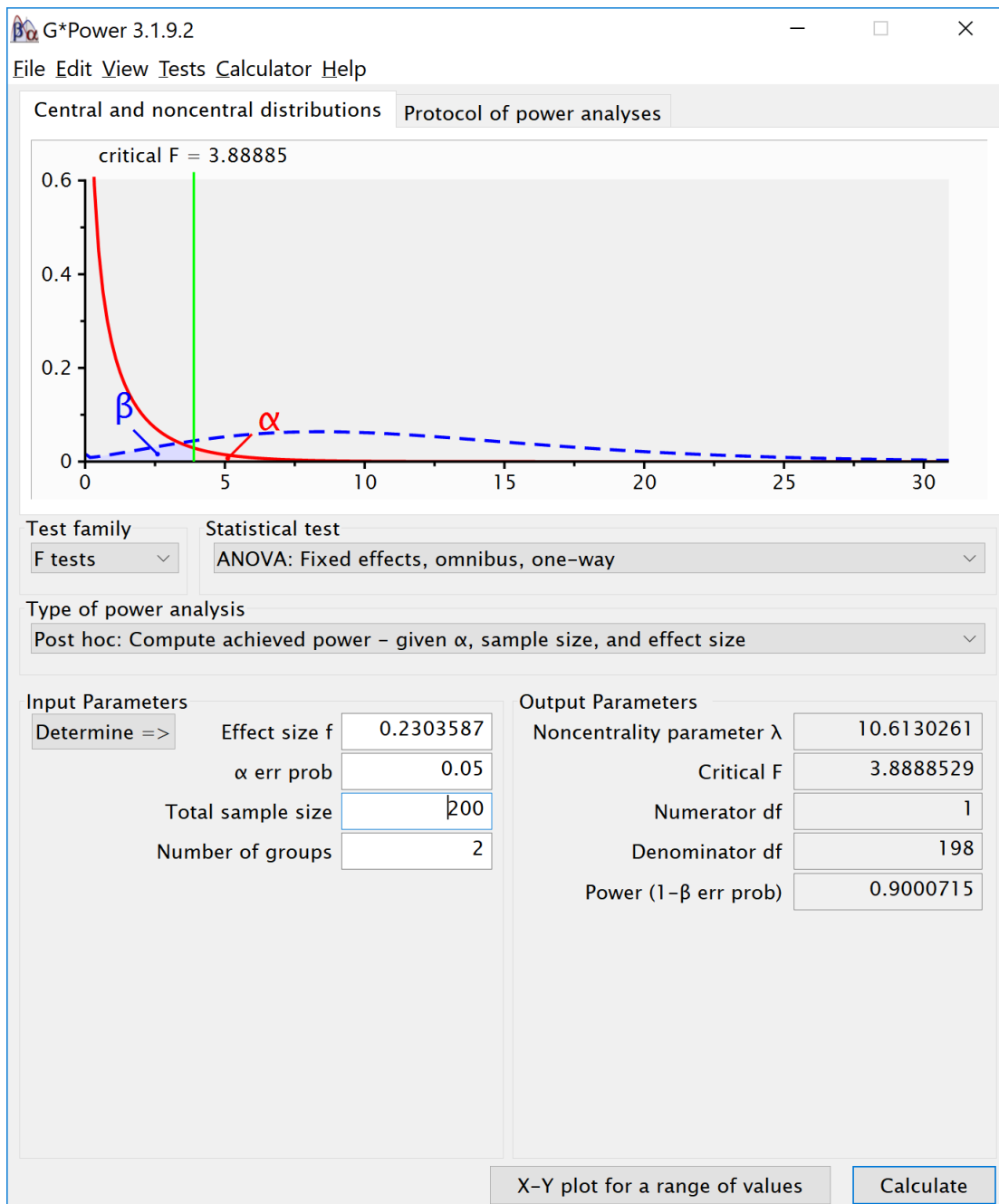
Means for each condition in the design



```
ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_condition    88    0.0511
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_condition_control_condition_pet    88    0.4444
```

The simulation confirms that for the F -test for the ANOVA we have 90% power. This is also what g*power tells us what would happen based on a post-hoc power analysis with an f of 0.2303587, 2 groups, 200 participants in total (100 in each between subject condition), and an alpha of 5%.



If we return to our expected means, how many participants do we need for sufficient power? Given the expected difference and standard deviation, $d = 0.34375$, and $f = 0.171875$. We can perform an a-priori power analysis for this simple case, which tells us we need 179 participants in each group (we can't split people in parts, and thus always round a power analysis upward), or 358 in total.

```
K <- 2
power <- 0.9
f <- 0.171875
```

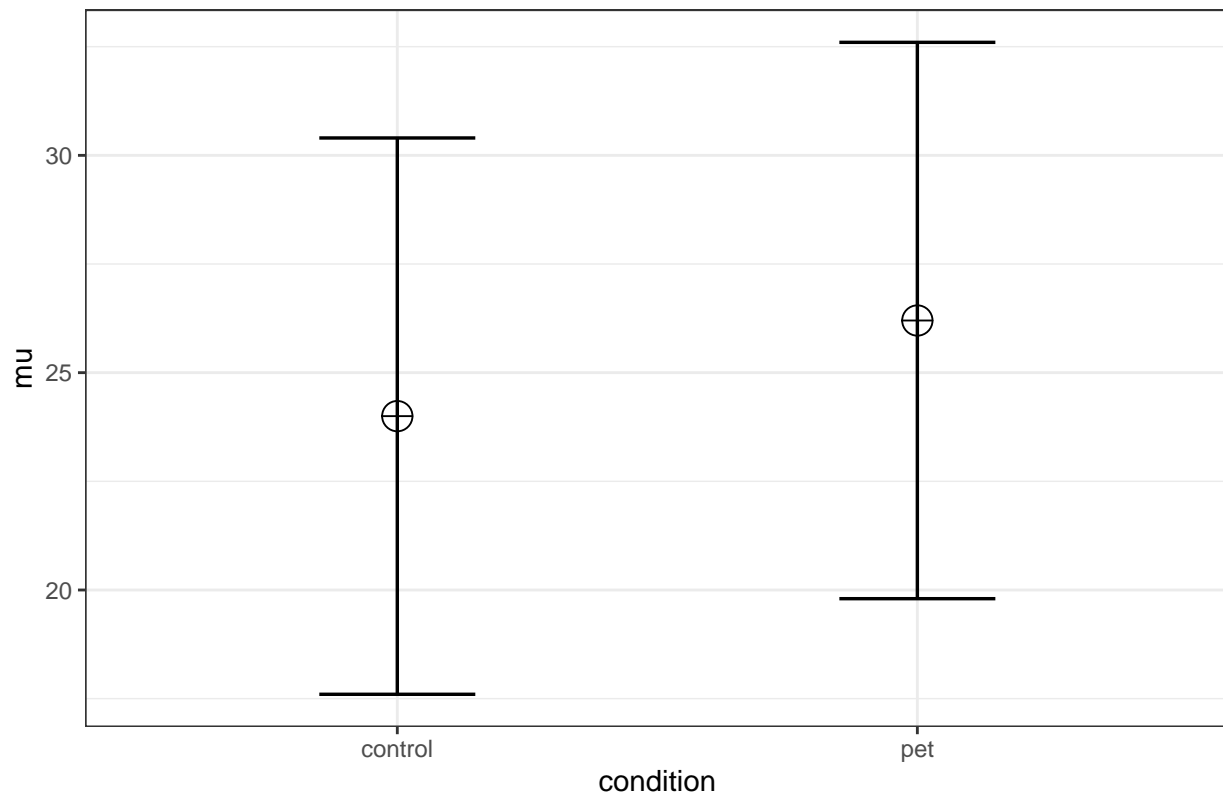
```
pwr.anova.test(power = power,
               k = K,
               f = f,
               sig.level = alpha_level)
```

```
##
##      Balanced one-way analysis of variance power calculation
##
##           k = 2
##           n = 178.8104
##           f = 0.171875
##      sig.level = 0.05
##           power = 0.9
##
## NOTE: n is number in each group
```

If we re-run the simulation with this sample size, we indeed have 90% power.

```
string <- "2b"
n <- 179
mu <- c(24, 26.2)
# Enter means in the order that matches the labels below.
# In this case, control, pet.
sd <- 6.4
labelnames <- c("condition", "control", "pet") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```

Means for each condition in the design

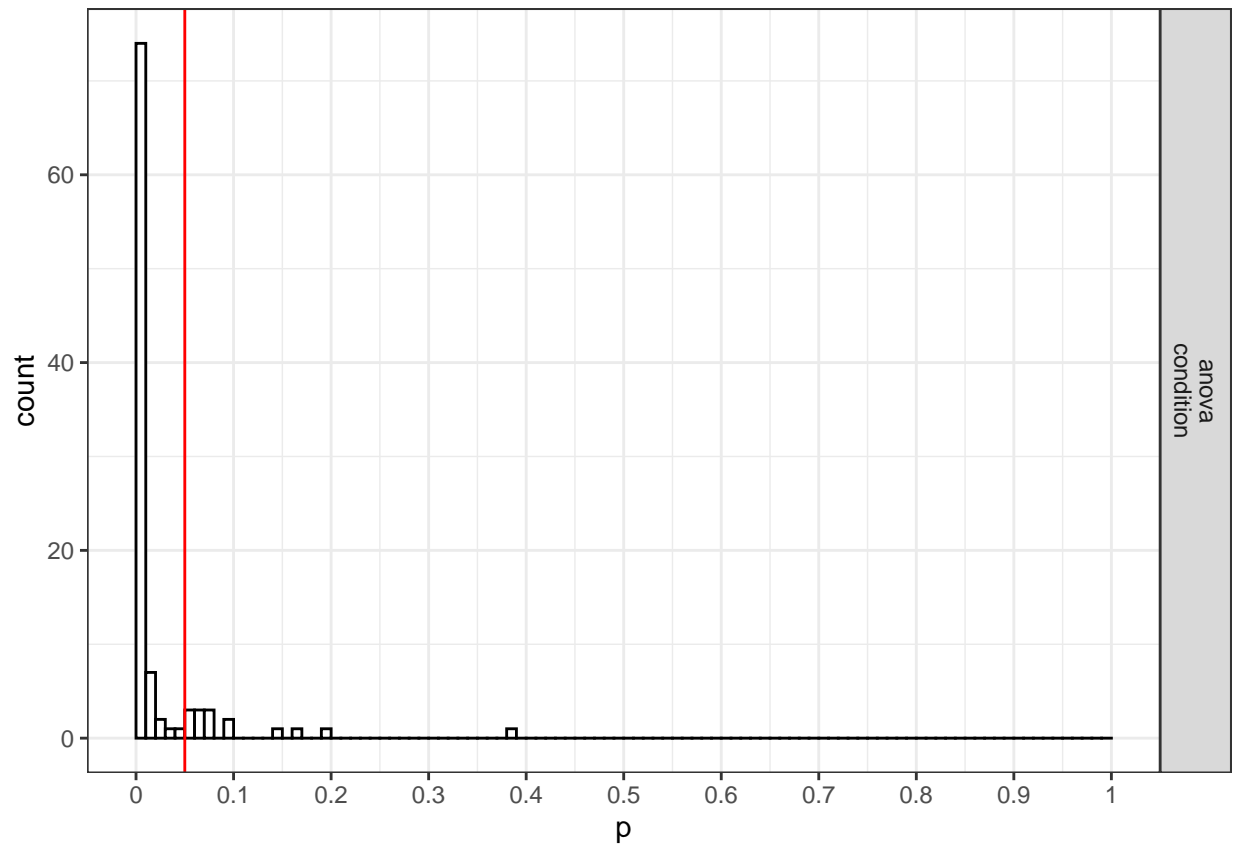


```
alpha_level <- 0.05
power_result <- ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_condition    85    0.03191
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_condition_control_condition_pet    85    0.3474
```

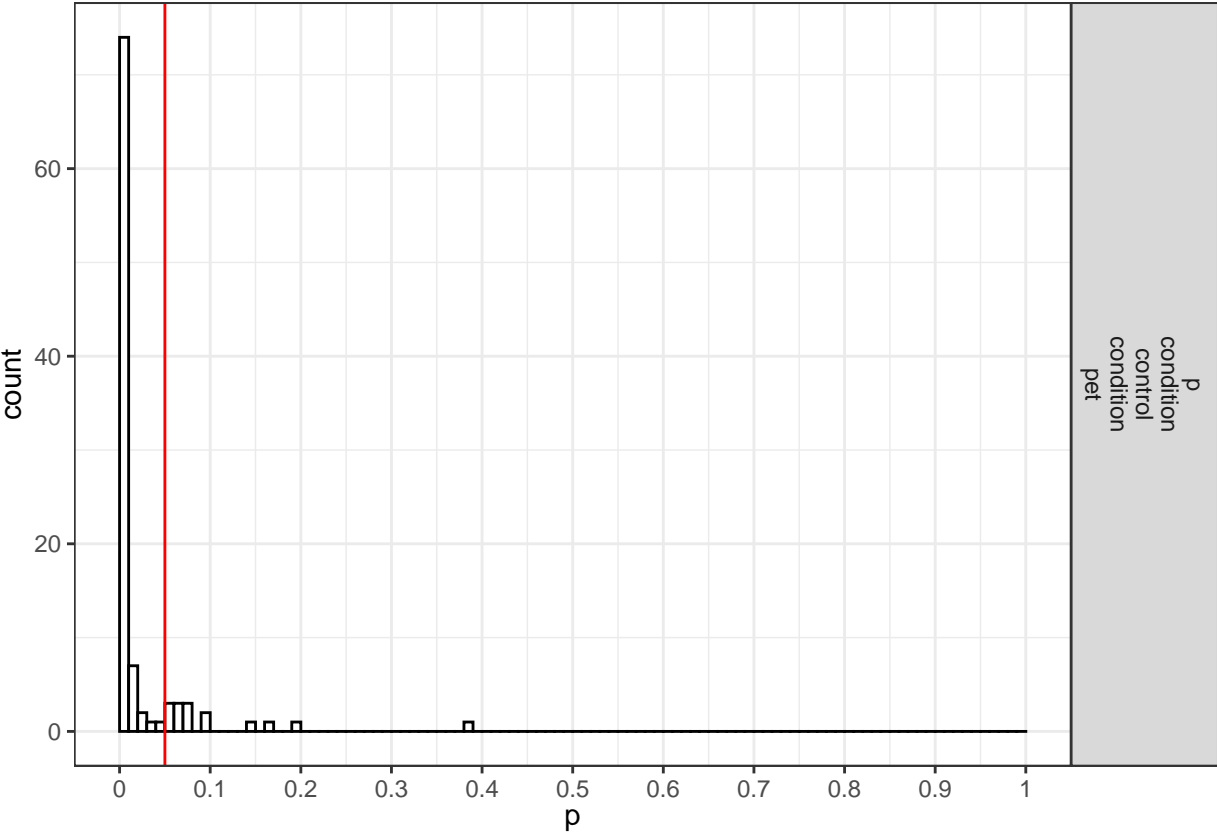
We stored the result from the power analysis in an object. This allows us to request plots (which are not printed automatically) showing the p -value distribution. If we request `power_result$plot1` we get the p -value distribution for the ANOVA:

```
power_result$plot1
```



If we request `power_result$plot2` we get the p-value distribution for the paired comparisons (in this case only one):

```
power_result$plot2
```



Chapter 3

One-way ANOVA Part 2

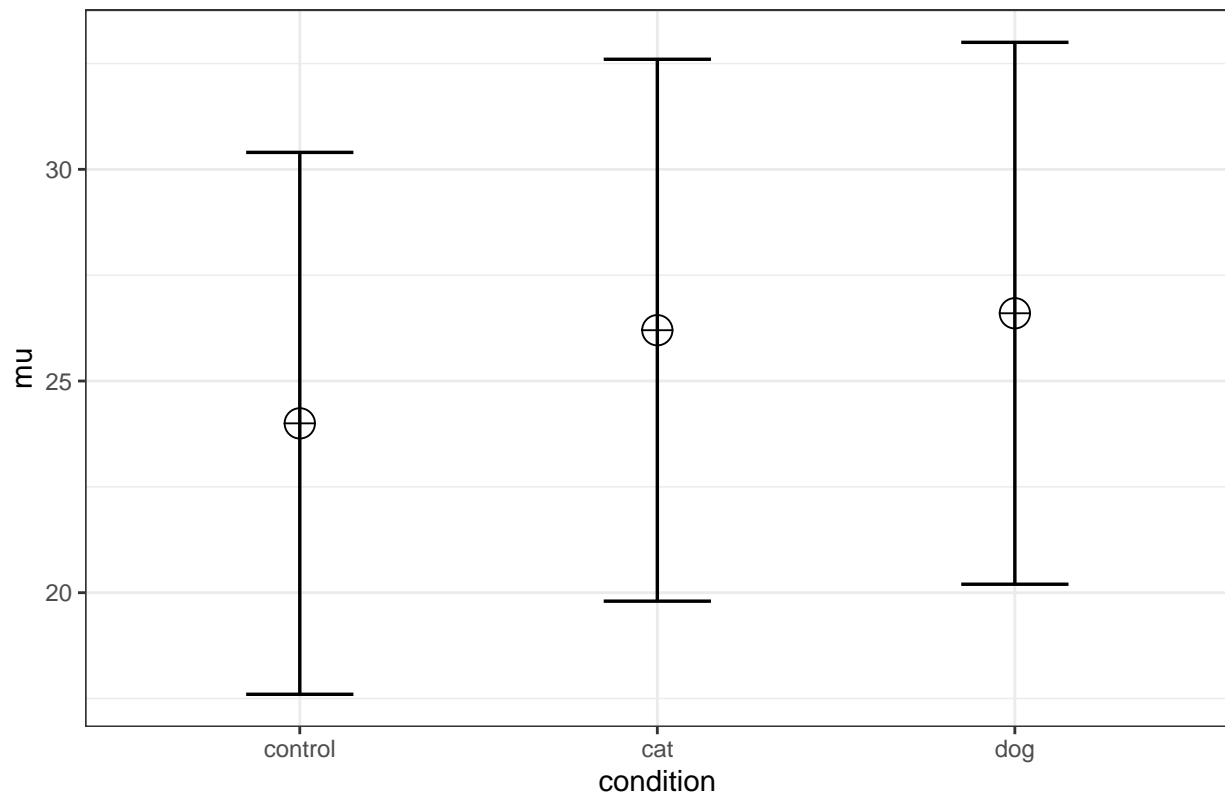
3.1 Validation of Power in One-Way ANOVA with Brysbaert example

3.1.1 Three conditions

Imagine we aim to design a study to test the hypothesis that giving people a pet to take care of will increase their life satisfaction. We have a control condition, a ‘cat’ pet condition, and a ‘dog’ pet condition. We can simulate a One-Way ANOVA with a specified alpha, sample size, and effect size, on see the statistical power we would have for the ANOVA and the follow-up comparisons. We expect all pets to increase life-satisfaction compared to the control condition. Obviously, we also expect the people who are in the ‘dog’ pet condition to have even greater life-satisfaction than people in the ‘cat’ pet condition. Based on work by Pavot and Diener (1993) we believe that we can expect responses on the life-satisfaction scale to have a mean of approximately 24 in our population, with a standard deviation of 6.4. We expect having a pet increases life satisfaction with approximately 2.2 scale points for participants who get a cat, and 2.6 scale points for participants who get a dog. We initially consider collecting data from 150 participants in total, with 50 participants in each condition. But before we proceed with the data collection, we examine the statistical power our design would have to detect the differences we predict.

```
string <- "3b"
n <- 50
# We are thinking of running 50 people in each condition
mu <- c(24, 26.2, 26.6)
# Enter means in the order that matches the labels below.
# In this case, control, cat, dog.
sd <- 6.4
labelnames <- c("condition", "control", "cat", "dog") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```

Means for each condition in the design



```
alpha_level <- 0.05
# You should think carefully about how to justify your alpha level.
# We will give some examples later, but for now, use 0.05.
ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_condition    53    0.05078
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_condition_control_condition_cat    44    0.3614
## p_condition_control_condition_dog    61    0.4644
## p_condition_cat_condition_dog        5    0.1008
```

```
#should yield
#0.3983064
#0.5205162
#0.06104044
```

The result shows that you would have quite low power with 50 participants, both for the overall ANOVA (just around 50% power), as for the follow up comparisons (approximately 40% power for the control vs cat condition, around 50% for the control vs dogs condition, and a really low power (around 6%, just above the Type 1 error rate of 5%) for the expected difference between cats and dogs.

3.2 Power for simple effects

We are typically not just interested in the ANOVA, but also in follow up comparisons. In this case, we would perform a *t*-test comparing the control condition against the cat and dog condition, and we would compare the cat and dog conditions against each other, in independent *t*-tests.

For our example, Cohen's *d* (the standardized mean difference) is $2.2/6.4$, or $d = 0.34375$ for the difference between the control condition and cats, $2.6/6.4$ or $d = 0.40625$ for the difference between the control condition and dogs, and $0.4/6.4$ or $d = 0.0625$ for the difference between cats and dogs as pets.

We can easily compute the expected power for these simple comparisons using the *pwr* package.

```
pwr.t.test(d = 2.2/6.4,
           n = 50,
           sig.level = 0.05,
           type="two.sample",
           alternative="two.sided")$power
```

```
## [1] 0.3983064
```

```
pwr.t.test(d = 2.6/6.4,
           n = 50,
           sig.level = 0.05,
           type="two.sample",
           alternative="two.sided")$power
```

```
## [1] 0.5205162
```

```
pwr.t.test(d = 0.4/6.4,
           n = 50,
           sig.level = 0.05,
           type="two.sample",
           alternative="two.sided")$power
```

```
## [1] 0.06104044
```

This analysis tells us that running the study with 50 participants in each condition is more likely to *not* yield a significant test result, even if our expected pattern of differences is true, than that we will observe a *p*-value smaller than our alpha level. This is not optimal.

Let's mathematically explore which pattern of means we would need to expect to have 90% power for the ANOVA with 50 participants in each group. We can use the *pwr* package in R to compute a sensitivity analysis that tells us the effect size, in Cohen's *f*, that we are able to detect with 3 groups and 50 participants in each group, in order to achieve 90% power with an alpha level of 5%.

```
K <- 3
n <- 50
sd <- 6.4
r <- 0
#Calculate f when running simulation
f <- pwr.anova.test(n = n,
                   k = K,
                   power = 0.9,
                   sig.level = alpha_level)$f
f
```

```
## [1] 0.2934417
```

This sensitivity analysis shows we have 90% power in our planned design to detect effects of Cohen's f of 0.2934417. Benchmarks by Cohen (1988) for small, medium, and large Cohen's f values are 0.1, 0.25, and 0.4, which correspond to eta-squared values of small (.0099), medium (.0588), and large (.1379), in line with $d = .2$, $.5$, or $.8$. So, at least based on these benchmarks, we have 90% power to detect effects that are somewhat sizeable.

```
f2 <- f^2
ES <- f2/(f2+1)
ES
```

```
## [1] 0.07928127
```

Expressed in eta-squared, we can detect values of eta-squared = 0.0793 or larger.

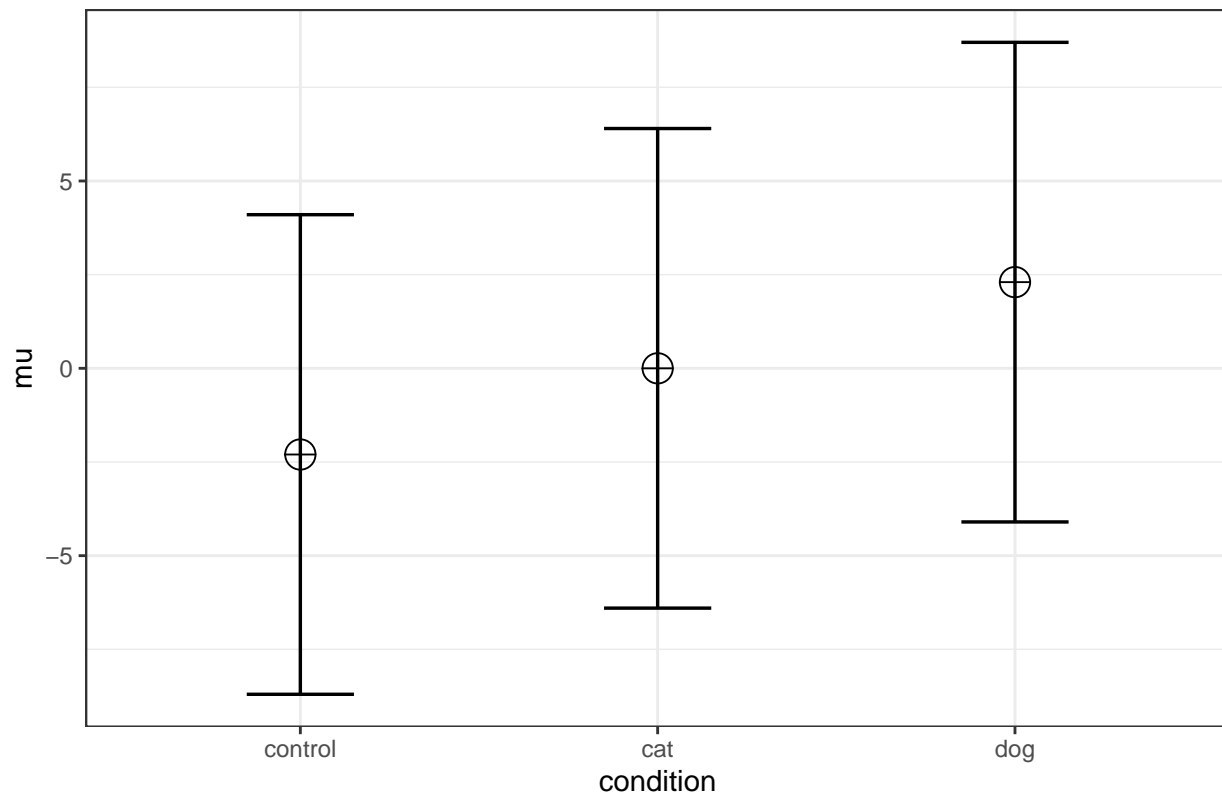
```
mu <- mu_from_ES(K = K, ES = ES)
mu <- mu * sd
mu
```

```
## [1] -2.300104 0.000000 2.300104
```

We can compute a pattern of means, given a standard deviation of 6.4, that would give us an effect size of $f = 0.2934$, or eta-squared of 0.0793. We should be able to accomplish this if the means are -2.300104, 0.000000, and 2.300104. We can use these values to confirm the ANOVA has 90% power.

```
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```

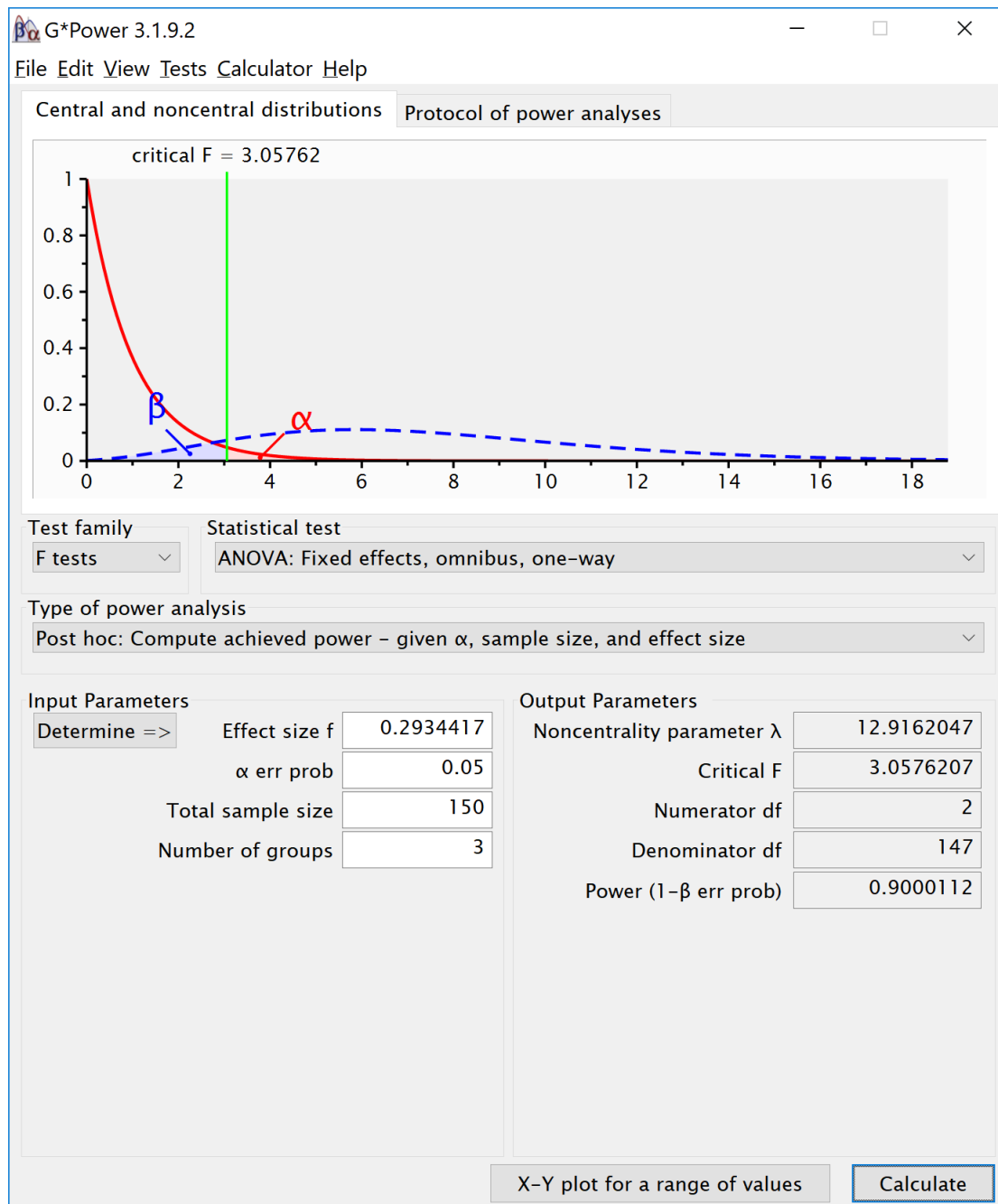
Means for each condition in the design



```
ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_condition    95    0.09547
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_condition_control_condition_cat    46    0.4072
## p_condition_control_condition_dog    98    0.7647
## p_condition_cat_condition_dog    39    0.3475
```

The simulation confirms that for the F -test for the ANOVA we have 90% power. This is also what g^* power tells us what would happen based on a post-hoc power analysis with an f of 0.2934417, 3 groups, 150 participants in total (50 in each between subject condition), and an alpha of 5%.



We can also compute the power for the ANOVA and simple effects in R with the `pwr` package. The calculated effect sizes and power match those from the simulation.

```
K <- 3
n <- 50
sd <- 6.4
f <- 0.2934417
pwr.anova.test(n = n,
               k = K,
               f = f,
               sig.level = alpha_level)$power
```

```
## [1] 0.9000112
```

```
d <- 2.300104/6.4
d
```

```
## [1] 0.3593912
```

```
pwr.t.test(d = 2.300104/6.4,
           n = 50,
           sig.level = 0.05,
           type="two.sample",
           alternative="two.sided")$power
```

```
## [1] 0.4284243
```

```
d <- 2*2.300104/6.4
d
```

```
## [1] 0.7187825
```

```
pwr.t.test(d = d,
           n = 50,
           sig.level = 0.05,
           type="two.sample",
           alternative="two.sided")$power
```

```
## [1] 0.9450353
```

We can also compare the results against the analytic solution by Aberson (2019).

First, load the function for a 3-way ANOVA.

```
anova1f_3<-function(m1=NULL,m2=NULL,m3=NULL,s1=NULL,s2=NULL,s3=NULL,n1=NULL,n2=NULL,n3=NULL,alpha=.05){
  x<-stats::rnorm(n1,m1,s1)
  X<-x
  MEAN<-m1
  SD<-s1
  Z <- (((X - mean(X, na.rm = TRUE))/stats::sd(X, na.rm = TRUE))) * SD
  y<-MEAN + Z
  group<-rep("A1",n1)
  l1<-data.frame(y, group)
  x<-stats::rnorm(n2,m2,s2)
  X<-x
  MEAN<-m2
  SD<-s2
  Z <- (((X - mean(X, na.rm = TRUE))/stats::sd(X, na.rm = TRUE))) * SD
  y<-MEAN + Z
  group<-rep("A2",n2)
  l2<-data.frame(y, group)
  x<-stats::rnorm(n3,m3,s3)
  X<-x
```

```

MEAN<-m3
SD<-s3
Z <- (((X - mean(X, na.rm = TRUE))/stats::sd(X, na.rm = TRUE))) * SD
y<-MEAN + Z
group<-rep("A3",n3)
l3<-data.frame(y, group)
simdat<-rbind(l1,l2,l3)
anova<-stats::aov(y~group, data=simdat)
anova<-car::Anova(anova, type="III")
SSA<-anova[2,1] #column, row
SSwin<-anova[3,1]
dfwin<-anova[3,2]
dfbg<-anova[2,2]
eta2<-SSA/(SSA+SSwin)
f2<-eta2/(1-eta2)
lambda<-f2*dfwin
minusalpha<-1-alpha
Ft<-stats::qf(minusalpha, dfbg, dfwin)
power<-1-stats::pf(Ft, dfbg,dfwin,lambda)
list(Power = power)}

```

Then we use the function to calculate power.

```

#Initial example, low power
anova1f_3(m1=24, m2=26.2, m3=26.6, s1=6.4, s2=6.4, s3=6.4, n1=50, n2=50, n3=50, alpha=.05)

## $Power
## [1] 0.4769468

#From: Aberson, Christopher L. Applied Power Analysis for the Behavioral Sciences, 2nd Edition.
# $Power [1] 0.4769468
#Later example, based on larger mean difference
anova1f_3(m1=-2.300104, m2=0, m3=2.300104, s1=6.4, s2=6.4, s3=6.4, n1=50, n2=50, n3=50, alpha=.05)

## $Power
## [1] 0.9000112

# $Power [1] 0.9000112

```


Chapter 4

One-way ANOVA Part 3

We first repeat the simulation by Brysbaert:

```
# Simulations to estimate the power of an ANOVA with three unrelated groups
# the effect between the two extreme groups is set to d = .4, the effect for the third group is d = .4
# we use the built-in aov-test command
# give sample sizes (all samples sizes are equal)
N = 90
# give effect size d
d1 = .4 #difference between the extremes
d2 = .4 #third condition goes with the highest extreme
# give number of simulations
nSim = nsims
# give alpha levels
alpha1 = .05 #alpha level for the omnibus ANOVA
alpha2 = .05 #alpha level for three post hoc one-tailed t-tests Bonferroni correction
# create progress bar in case it takes a while
#pb <- winProgressBar(title = "progress bar", min = 0, max = nSim, width = 300)
# create vectors to store p-values
p1 <- numeric(nSim) #p-value omnibus ANOVA
p2 <- numeric(nSim) #p-value first post hoc test
p3 <- numeric(nSim) #p-value second post hoc test
p4 <- numeric(nSim) #p-value third post hoc test
pes1 <- numeric(nSim) #partial eta-squared
pes2 <- numeric(nSim) #partial eta-squared two extreme conditions
library(lsr)
for(i in 1:nSim){ #for each simulated experiment
  # setWinProgressBar(pb, i, title=paste(round(i/nSim*100, 1), "% done"))
  x<-rnorm(n = N, mean = 0, sd = 1)
  y<-rnorm(n = N, mean = d1, sd = 1)
  z<-rnorm(n = N, mean = d2, sd = 1)
  data = c(x,y,z)
  groups= factor(rep(letters[24:26], each = N))
  test <- aov(data~groups)
  pes1[i] <- etaSquared(test)[1,2]
  p1[i] <- summary(test)[[1]][["Pr(>F)"]][[1]]
  p2[i] <- t.test(x,y)$p.value
  p3[i] <- t.test(x,z)$p.value
  p4[i] <- t.test(y,z)$p.value
```

```

data = c(x,y)
groups= factor(rep(letters[24:25], each = N))
test <- aov(data~groups)
pes2[i] <- etaSquared(test)[1,2]
}
#close(pb)#close progress bar
# results are as predicted when omnibus ANOVA is significant, t-tests are significant between x and y p
#printing all unique tests (adjusted code by DL)
sum(p1<alpha1)/nSim

```

```
## [1] 0.81
```

```
sum(p2<alpha2)/nSim
```

```
## [1] 0.79
```

```
sum(p3<alpha2)/nSim
```

```
## [1] 0.78
```

```
sum(p4<alpha2)/nSim
```

```
## [1] 0.03
```

```
mean(pes1)
```

```
## [1] 0.04011467
```

```
mean(pes2)
```

```
## [1] 0.04241091
```

4.1 Three conditions replication

```

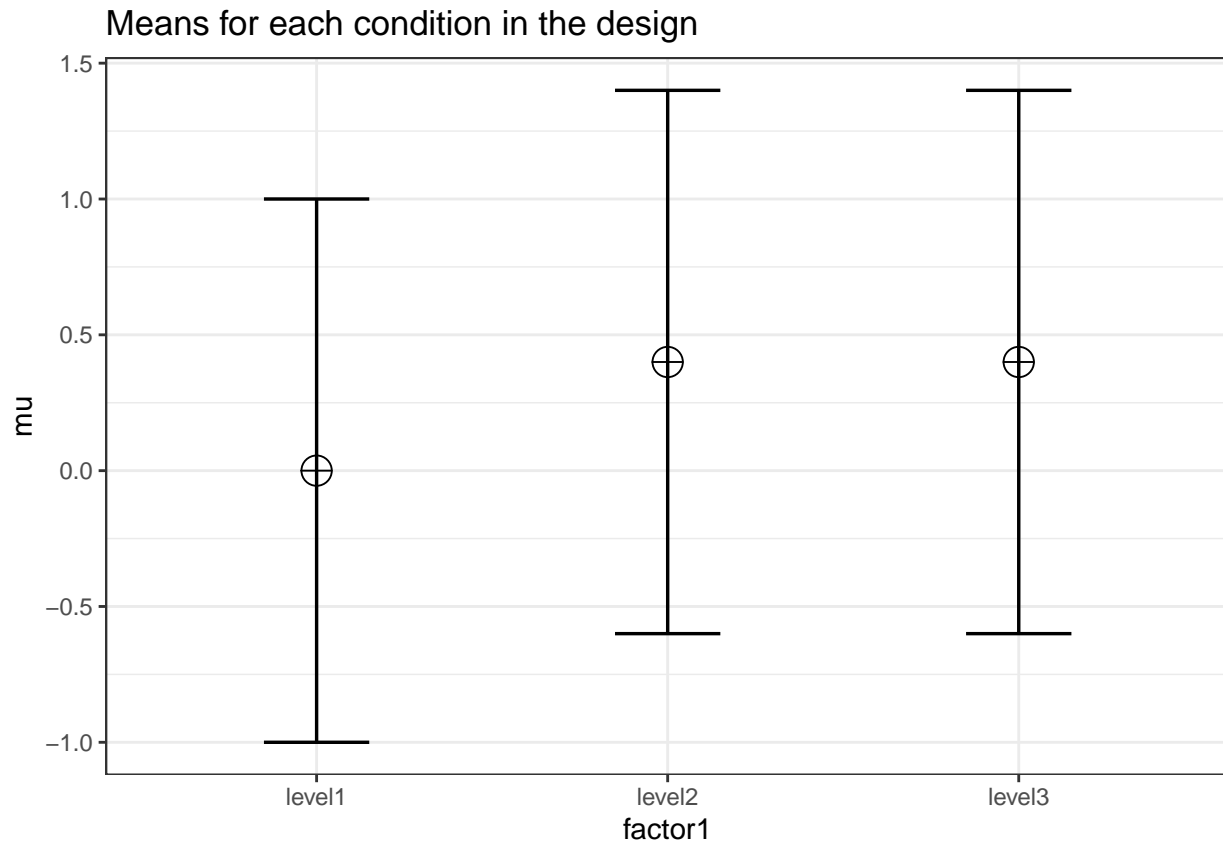
K <- 3
mu <- c(0, 0.4, 0.4)
n <- 90
sd <- 1
r <- 0
design =paste(K,"b",sep="")

```

```

design_result <- ANOVA_design(design =string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = c("factor1", "level1", "level2", "level3"))

```



```
ANOVA_power(design_result, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_factor1    82    0.04214
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_factor1_level1_factor1_level2    74    0.401812
## p_factor1_level1_factor1_level3    72    0.398766
## p_factor1_level2_factor1_level3     7   -0.002054
```

4.2 Variation 1

```
# give sample sizes (all samples sizes are equal)
N = 145
# give effect size d
d1 = .4 #difference between the extremes
d2 = .0 #third condition goes with the highest extreme
# give number of simulations
nSim = nsims
# give alpha levels
```

```

alpha1 = .05 #alpha level for the omnibus ANOVA
alpha2 = .05 #alpha level for three post hoc one-tailed t-tests Bonferroni correction
# create progress bar in case it takes a while
#pb <- winProgressBar(title = "progress bar", min = 0, max = nSim, width = 300)
# create vectors to store p-values
p1 <-numeric(nSim) #p-value omnibus ANOVA
p2 <-numeric(nSim) #p-value first post hoc test
p3 <-numeric(nSim) #p-value second post hoc test
p4 <-numeric(nSim) #p-value third post hoc test
pes1 <-numeric(nSim) #partial eta-squared
pes2 <-numeric(nSim) #partial eta-squared two extreme conditions
library(lsr)
for(i in 1:nSim){ #for each simulated experiment
  # setWinProgressBar(pb, i, title=paste(round(i/nSim*100, 1), "% done"))
  x<-rnorm(n = N, mean = 0, sd = 1)
  y<-rnorm(n = N, mean = d1, sd = 1)
  z<-rnorm(n = N, mean = d2, sd = 1)
  data = c(x,y,z)
  groups= factor(rep(letters[24:26], each = N))
  test <- aov(data~groups)
  pes1[i] <- etaSquared(test)[1,2]
  p1[i] <- summary(test)[[1]][["Pr(>F)"]][[1]]
  p2[i] <- t.test(x,y)$p.value
  p3[i] <- t.test(x,z)$p.value
  p4[i] <- t.test(y,z)$p.value
  data = c(x,y)
  groups= factor(rep(letters[24:25], each = N))
  test <- aov(data~groups)
  pes2[i] <- etaSquared(test)[1,2]
}
#close(pb)#close progress bar
# results are as predicted when omnibus ANOVA is significant, t-tests are significant between x and y p
#printing all unique tests (adjusted code by DL)
sum(p1<alpha1)/nSim

```

```
## [1] 0.94
```

```
sum(p2<alpha2)/nSim
```

```
## [1] 0.89
```

```
sum(p3<alpha2)/nSim
```

```
## [1] 0.05
```

```
sum(p4<alpha2)/nSim
```

```
## [1] 0.94
```

```
mean(pes1)
```

```
## [1] 0.04057115
```

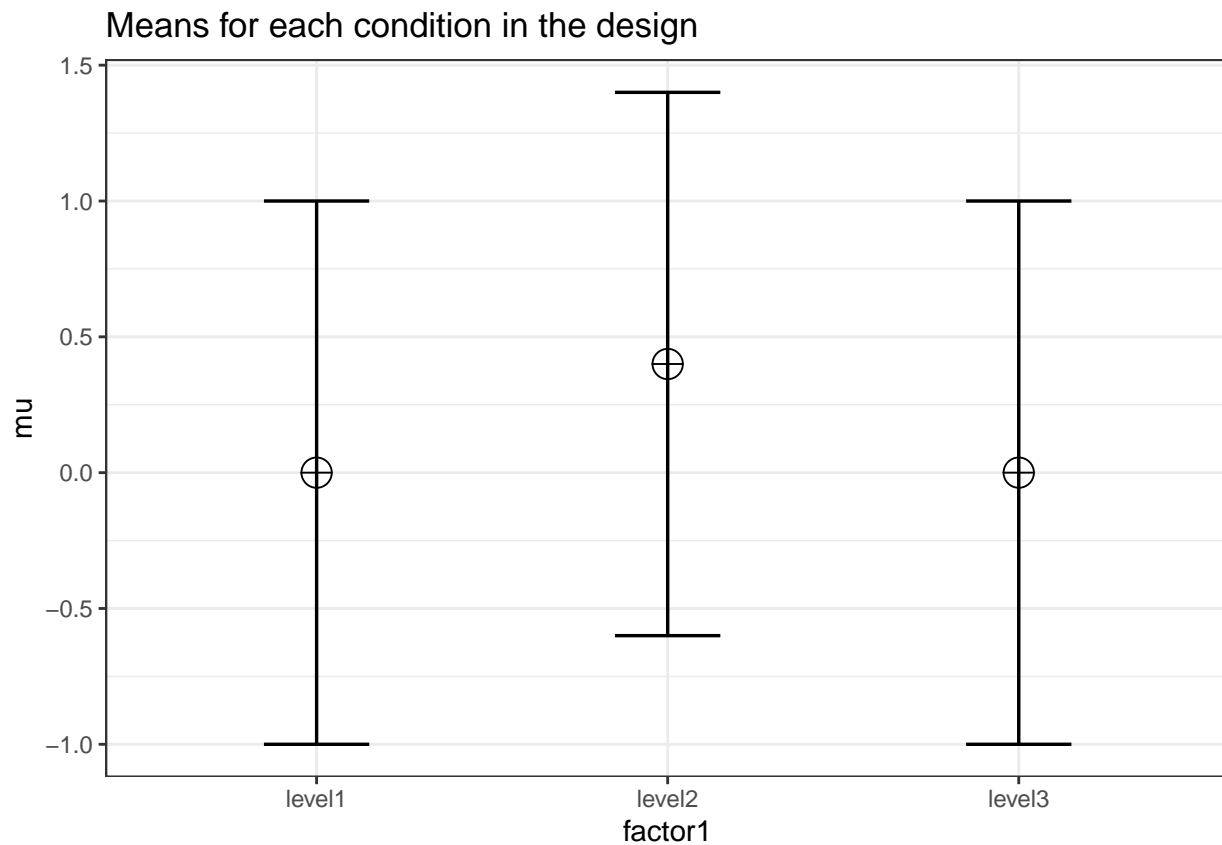
```
mean(pes2)
```

```
## [1] 0.04531236
```

4.3 Three conditions replication

```
K <- 3  
mu <- c(0, 0.4, 0.0)  
n <- 145  
sd <- 1  
r <- 0  
design = paste(K, "b", sep="")
```

```
design_result <- ANOVA_design(design = string,  
                             n = n,  
                             mu = mu,  
                             sd = sd,  
                             labelnames = c("factor1", "level1", "level2", "level3"))
```



```
ANOVA_power(design_result, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_factor1    92    0.03798
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_factor1_level1_factor1_level2    93    0.394404
## p_factor1_level1_factor1_level3     6    0.005287
## p_factor1_level2_factor1_level3    88   -0.391447
```

4.4 Variation 2

```
# give sample sizes (all samples sizes are equal)
N = 82
# give effect size d
d1 = .4 #difference between the extremes
d2 = .2 #third condition goes with the highest extreme
# give number of simulations
nSim = nsims
# give alpha levels
alpha1 = .05 #alpha level for the omnibus ANOVA
alpha2 = .05 #alpha level for three post hoc one-tailed t-tests Bonferroni correction
# create progress bar in case it takes a while
#pb <- winProgressBar(title = "progress bar", min = 0, max = nSim, width = 300)
# create vectors to store p-values
p1 <- numeric(nSim) #p-value omnibus ANOVA
p2 <- numeric(nSim) #p-value first post hoc test
p3 <- numeric(nSim) #p-value second post hoc test
p4 <- numeric(nSim) #p-value third post hoc test
pes1 <- numeric(nSim) #partial eta-squared
pes2 <- numeric(nSim) #partial eta-squared two extreme conditions
library(lsr)
for(i in 1:nSim){ #for each simulated experiment
  # setWinProgressBar(pb, i, title=paste(round(i/nSim*100, 1), "% done"))
  x<-rnorm(n = N, mean = 0, sd = 1)
  y<-rnorm(n = N, mean = d1, sd = 1)
  z<-rnorm(n = N, mean = d2, sd = 1)
  data = c(x,y,z)
  groups= factor(rep(letters[24:26], each = N))
  test <- aov(data~groups)
  pes1[i] <- etaSquared(test)[1,2]
  p1[i] <- summary(test)[[1]][["Pr(>F)"]][[1]]
  p2[i] <- t.test(x,y)$p.value
  p3[i] <- t.test(x,z)$p.value
  p4[i] <- t.test(y,z)$p.value
  data = c(x,y)
  groups= factor(rep(letters[24:25], each = N))
  test <- aov(data~groups)
```

```

    pes2[i] <- etaSquared(test)[1,2]
  }
#close(pb)#close progress bar
# results are as predicted when omnibus ANOVA is significant, t-tests are significant between x and y p
#printing all unique tests (adjusted code by DL)
sum(p1<alpha1)/nSim

## [1] 0.63

sum(p2<alpha2)/nSim

## [1] 0.73

sum(p3<alpha2)/nSim

## [1] 0.18

sum(p4<alpha2)/nSim

## [1] 0.3

mean(pes1)

## [1] 0.03383246

mean(pes2)

## [1] 0.0425566

```

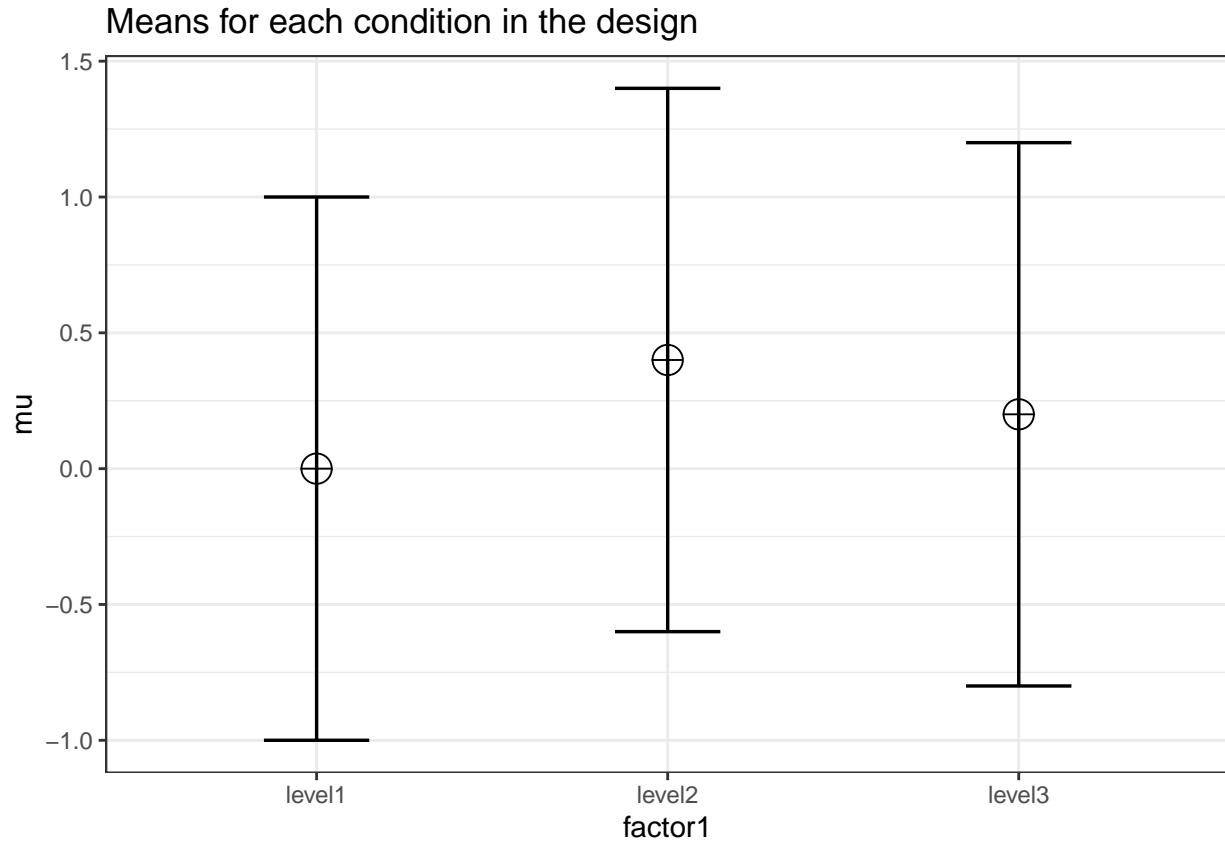
4.5 Three conditions replication

```

K <- 3
mu <- c(0, 0.4, 0.2)
n <- 82
sd <- 1
design =paste(K,"b",sep="")

design_result <- ANOVA_design(design =string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = c("factor1", "level1", "level2", "level3"))

```



```
ANOVA_power(design_result, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_factor1    64    0.0368
##
## Power and Effect sizes for contrasts
##                               power effect_size
## p_factor1_level1_factor1_level2    77    0.4242
## p_factor1_level1_factor1_level3    21    0.2039
## p_factor1_level2_factor1_level3    26   -0.2198
```

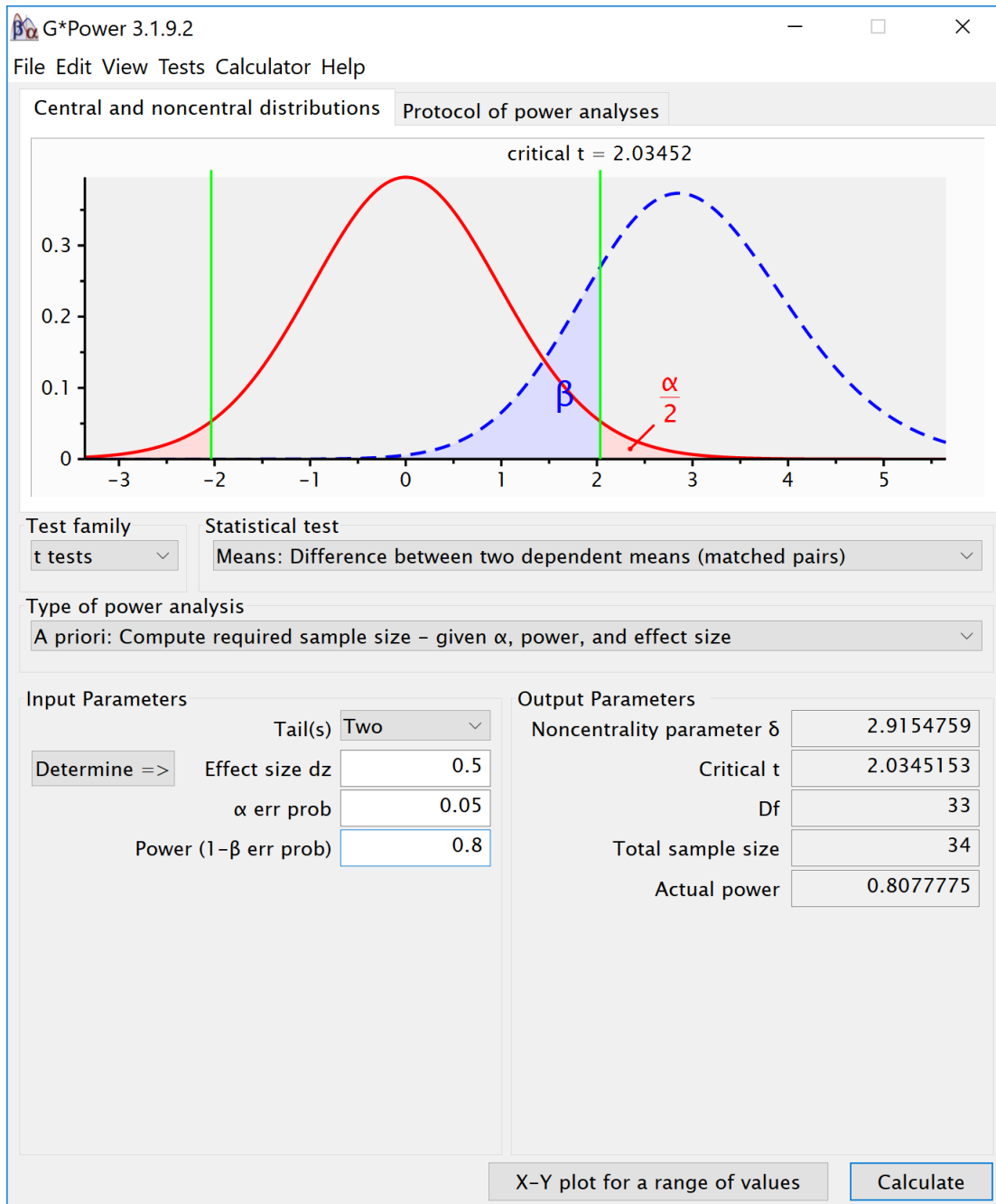

Chapter 5

Repeated Measures-ANOVA Part 1

In a repeated measures design multiple observations are collected from the same participants. In the simplest case, where there are two repeated observations, a repeated measures ANOVA equals a dependent or paired t -test. The difference compared to a between subject design is that repeated measures can be correlated, and in psychology, they often are. Let's first explore the impact of this correlation on the power of a repeated measures ANOVA.

5.1 Two conditions, medium effect size

To illustrate the effect of correlated observations, we start by simulating data for a medium effect size for a dependent (or paired, or within-subject) t -test. Let's first look at G*power. If we want to perform an a-priori power analysis, we are asked to fill in the effect size d_z . As Cohen (1988) writes, "The Z subscript is used to emphasize the fact that our raw score unit is no longer X or Y, but Z", where Z are the difference scores of X-Y.



Within designs can have greater power to detect differences than between designs because the values are correlated, and a within design requires less participants because each participant provides multiple observations. One difference between an independent t -test and a dependent t -test is that an independent t -test has $2(n-1)$ degrees of freedom, while a dependent t -test has $(n-1)$ degrees of freedom. The sample size needed in a two-group within-design (NW) relative to the sample needed in two-group between-designs (NB), assuming normal distributions, and ignoring the difference in degrees of freedom between the two types of tests, is (from Maxwell & Delaney, 2004, p. 561, formula 45):

$$N_W = \frac{N_B(1-\rho)}{2}$$

The division by 2 in the equation is due to the fact that in a two-condition within design every participant provides two data-points. The extent to which this reduces the sample size compared to a between-subject design depends on the correlation (r) between the two dependent variables, as indicated by the $1-r$ part of

the equation. If the correlation is 0, a within-subject design needs half as many participants as a between-subject design (e.g., 64 instead 128 participants), simply because every participants provides 2 datapoints. The higher the correlation, the larger the relative benefit of within designs, and whenever the correlation is negative (up to -1) the relative benefit disappears.

Whereas in an independent t -test the two observations are uncorrelated, in a within design the observations are correlated. This has an effect on the standard deviation of the difference scores. In turn, because the standardized effect size is the mean difference divided by the standard deviation of the difference scores, the correlation has an effect on the standardized mean difference in a within design, Cohen's d_z . The relation, as Cohen (1988, formula 2.3.7) explains, is:

$$\sigma_z = \sigma \sqrt{2(1 - \rho)}$$

Therefore, the relation between d_z and d is $\sqrt{2(1 - \rho)}$. As Cohen (1988) writes: "In other words, a given difference between population means for matched (dependent) samples is standardized by a value which is $\sqrt{2(1 - \rho)}$ as large as would be the case were they independent. If we enter a correlation of 0.5 in the formula, we get $\sqrt{2(0.5)} = 1$. In other words, when the correlation is 0.5, $d = d_z$. When there is a strong correlation between dependent variables, for example $r = 0.9$, we get $d = d_z \sqrt{2(1 - 0.9)}$, and a d_z of 1 would be a $d = 0.45$. Reversely, $d_z = \frac{d}{\sqrt{2(1-r)}}$, so with a $r = 0.9$, a d of 1 would be a $d_z = 2.24$. Some consider this increase in d_z compared to d when observations are strongly correlated an 'inflation' when estimating effect sizes, but since the reduction in the standard deviation of the difference scores due to the correlation makes it easier to distinguish signal from noise in a hypothesis test, it leads to a clear power benefit.

```
# Check sample size formula Maxwell
# Power is pretty similar with n/2, same d (assuming r = 0.5).
# Small differences due to df = 2(n-1) vs df = n-1
pwr.t.test(d = 0.05,
           n = c(2000, 4000, 8000),
           sig.level = 0.05,
           type = "two.sample",
           alternative = "two.sided")
```

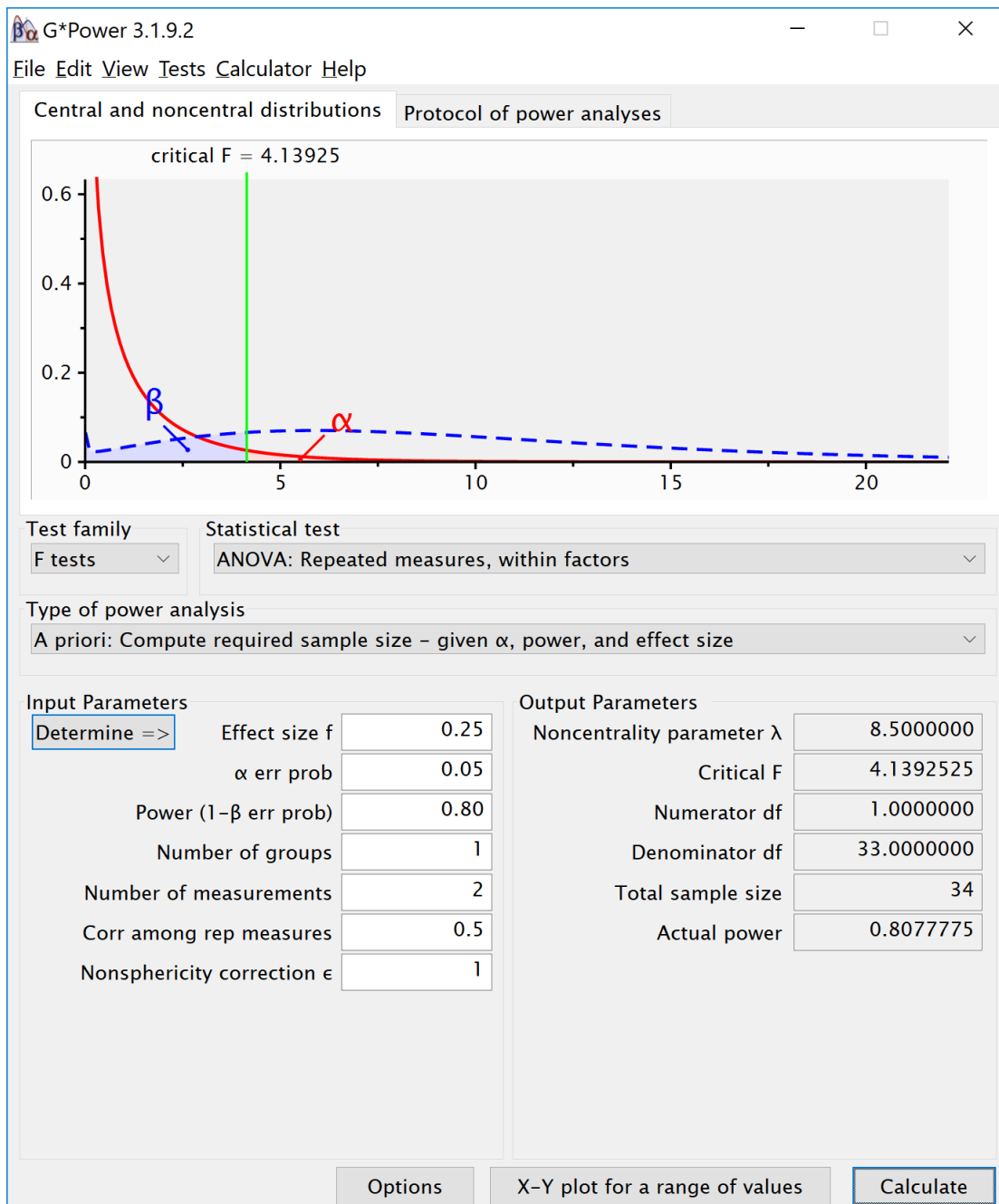
```
##
##      Two-sample t test power calculation
##
##              n = 2000, 4000, 8000
##              d = 0.05
##      sig.level = 0.05
##      power = 0.3524674, 0.6086764, 0.8853424
##      alternative = two.sided
##
## NOTE: n is number in *each* group
```

```
pwr.t.test(d = 0.05,
           n = c(1000, 2000, 4000),
           sig.level = 0.05,
           type = "paired",
           alternative = "two.sided")
```

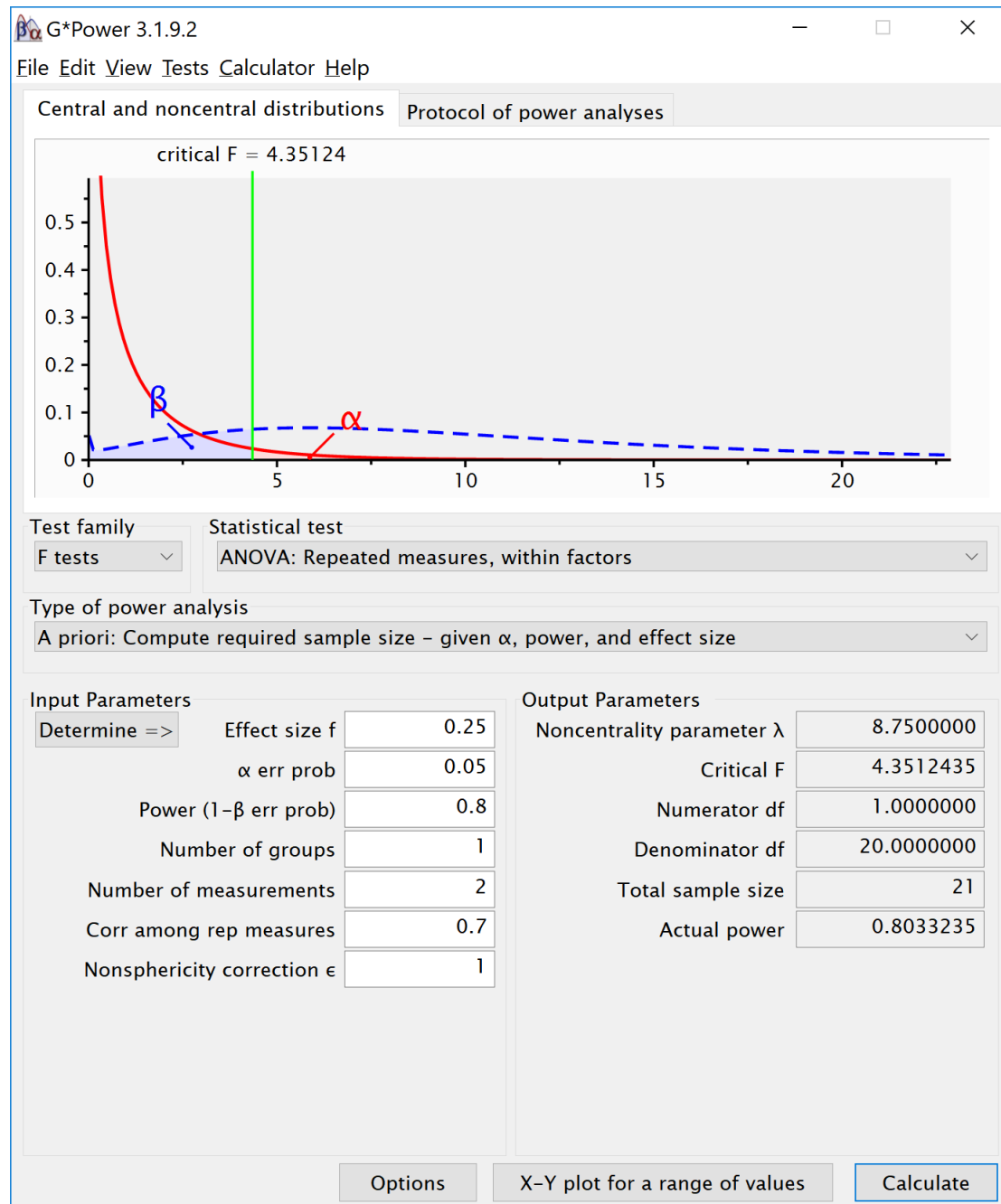
```
##
##      Paired t test power calculation
##
##              n = 1000, 2000, 4000
##              d = 0.05
```

```
##      sig.level = 0.05
##      power = 0.3520450, 0.6083669, 0.8852320
##      alternative = two.sided
##
## NOTE: n is number of *pairs*
```

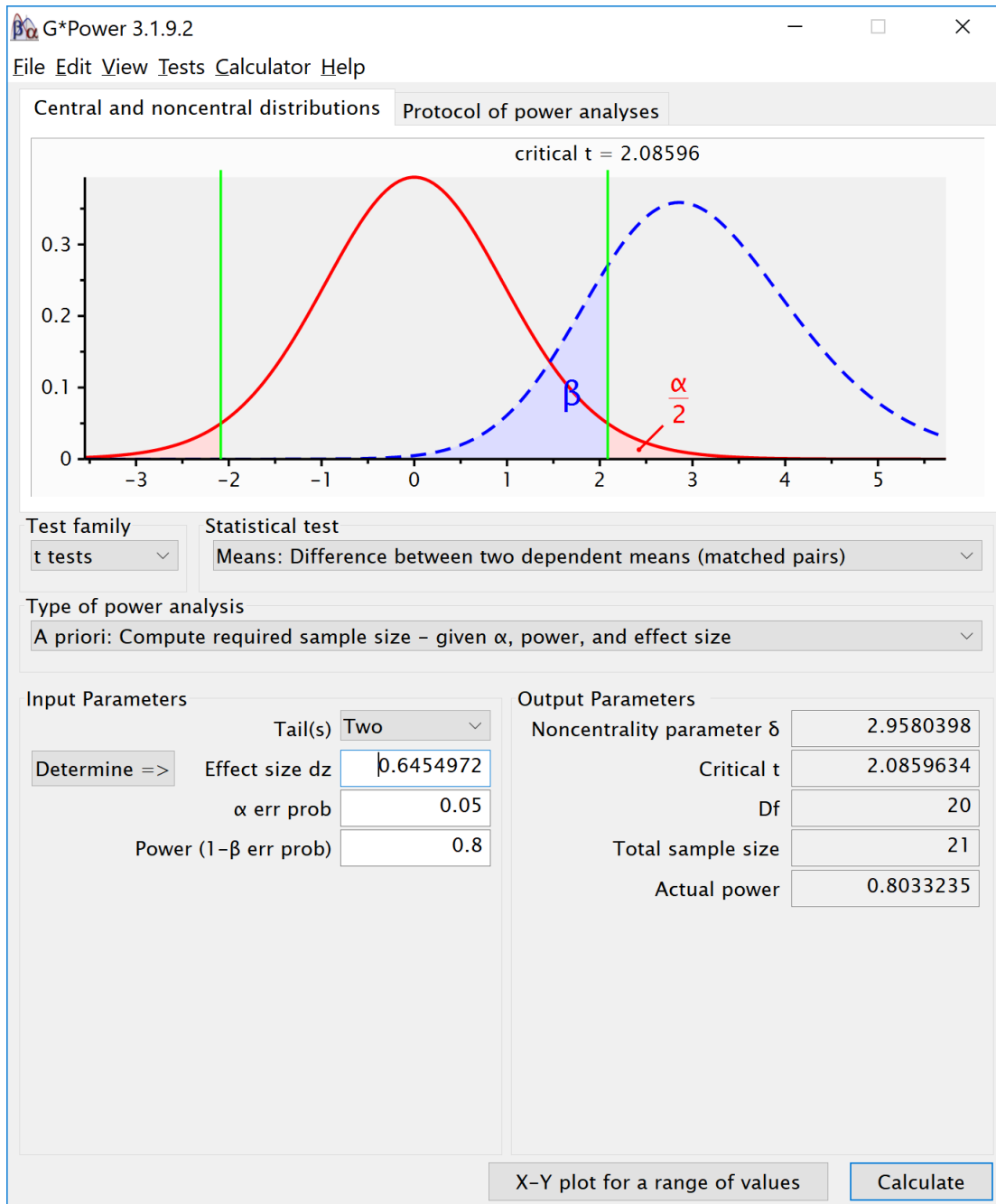
There is no equivalent f_z for Cohen's f for a within subject ANOVA. For two groups, we can directly compute Cohen's f from Cohen's d for two groups, as Cohen (1988) describes, because $f = 1/2d$. For a $d = 0.5$, $f = 0.25$. In Gpower we can run a 2 group within-subject power analysis for ANOVA. We plan for 80% power, and reproduce the analysis above for the dependent t -test. This works because the correlation is set to 0.5, when $d = dz$, and thus the transformation of $f=1/2d$ works.



If we change the correlation to 0.7 and keep all other settings the same, the repeated measure a-priori power analysis yields a sample of 21. The correlation increases the power for the test.



To reproduce this analysis in Gpower with a dependent t -test we need to change d_z following the formula above, $d_z = \frac{0.5}{\sqrt{2(1-0.7)}}$, which yields $d_z = 0.6454972$. If we enter this value in Gpower for an a-priori power analysis, we get the exact same results (as we should, since an repeated measures ANOVA with 2 groups equals a dependent t -test). This example illustrates that the correlation between dependent variables always factors into a power analysis, both for a dependent t -test, and for a repeated measures ANOVA. Because a dependent t -test uses d_z the correlation might be less visible, but given the relation between d and d_z , the correlation is always taken into account and can greatly improve power for within designs compared to between designs.



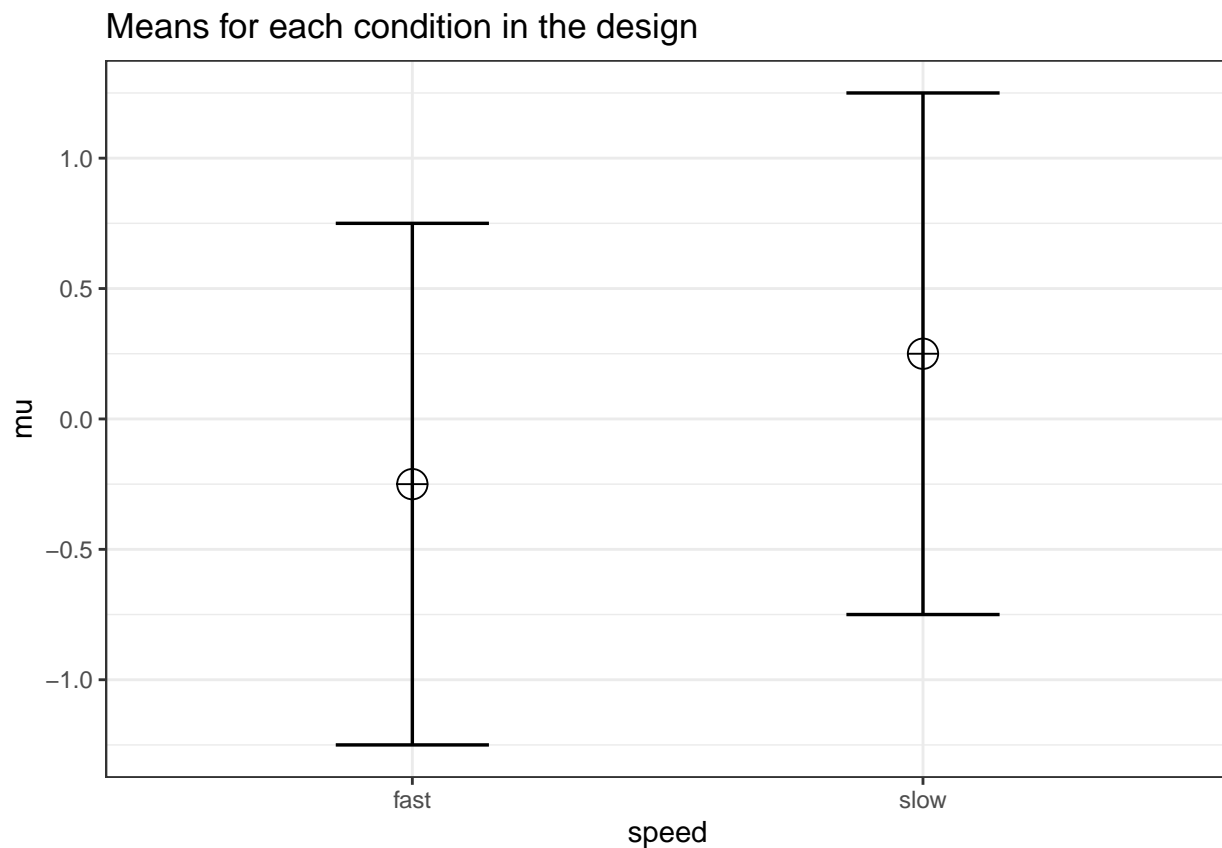
We can perform both these power analyses using simulations as well. We set groups to 2 for the simulation, $n = 34$ (which should give 80.777 power, according to the g*power program), a correlation among repeated measures of 0.5, and an alpha of 0.05. In this case, we simulate data with means -0.25 and 0.25, and set the sd to 1. This means we have a mean difference of 0.5, and a Cohen's d of $0.5/1 = 0.5$. In the first example, we set the correlation to 0.5, and the result should be 80.77% power, and an effect size estimate of 0.5 for the simple effect. We also calculate partial eta-squared for the ANOVA, which equals $\frac{f^2}{f^2+1}$, or 0.05882353.

```
K <- 2
n <- 34
sd <- 1
r <- 0.5
```

```
alpha = 0.05
f <- 0.25
f2 <- f^2
ES <- f2/(f2+1)
ES
```

```
## [1] 0.05882353
```

```
mu <- mu_from_ES(K = K, ES = ES)
design = paste(K, "w", sep = "")
labelnames <- c("speed", "fast", "slow")
design_result <- ANOVA_design(design = design,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             labelnames = labelnames)
```



```
alpha_level <- 0.05
ANOVA_power(design_result, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_speed 80      0.2138
```

```
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_speed_fast_speed_slow    80    0.5077
##
## Within-Subject Factors Included: Check MANOVA Results
```

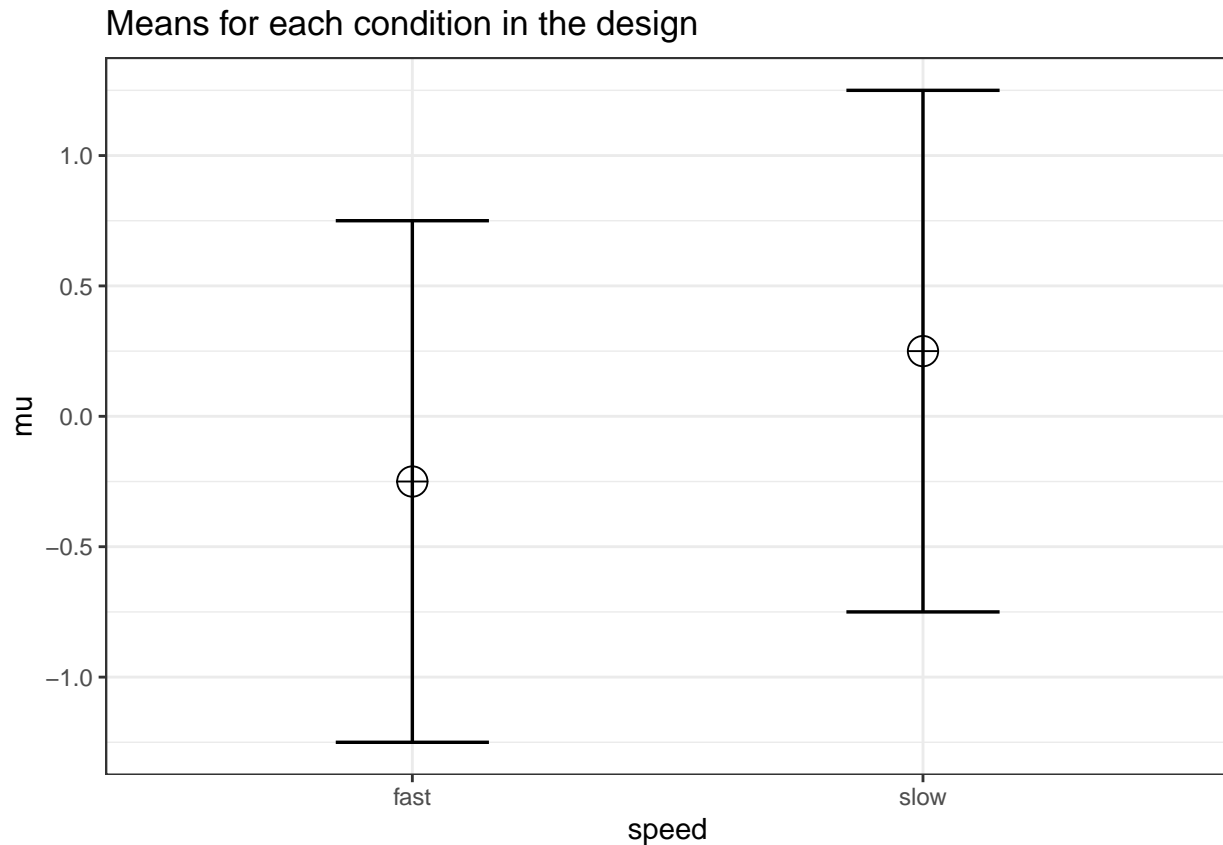
The results of the simulation are indeed very close to 80.777%. Note that the simulation calculates Cohen's d_z effect sizes for paired comparisons - which here given the correlation of 0.5 is also 0.5 for a medium effect size.

We should see a larger d_z if we increase the correlation, keeping the sample size the same, following the example in Gpower above. We repeat the simulation, and the only difference is a correlation between dependent variables of 0.7. This should yield an effect size $d_z = 0.6454972$.

```
K <- 2
n <- 21
sd <- 1
r <- 0.7
alpha = 0.05
f <- 0.25
f2 <- f^2
ES <- f2/(f2+1)
ES
```

```
## [1] 0.05882353
```

```
mu <- mu_from_ES(K = K, ES = ES)
design = paste(K,"w",sep="")
labelnames <- c("speed", "fast", "slow")
design_result <- ANOVA_design(design = design,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             labelnames = labelnames)
```

```
alpha_level <- 0.05
design_result$sigmatrix
```

```
##      fast slow
## fast  1.0  0.7
## slow  0.7  1.0
```

```
ANOVA_power(design_result, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_speed    75      0.2973
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_speed_fast_speed_slow    75      0.6434
##
## Within-Subject Factors Included: Check MANOVA Results
```

```
#relation dz and f for within designs
```

```
f <- 0.5*0.6454972
```

```
# Entering this f in G*power, with a correlation of 0.5, yields the same as entering f = 0.25 and corre
```


Chapter 6

Repeated Measures-ANOVA Part 2

In a repeated measures design multiple observations are collected from the same participants. Here, we will examine a repeated measures experiment with 3 within-subject conditions, to illustrate how a repeated measures ANOVA extends a dependent t -test with 3 groups.

In the example for a two-group within design we provided a specific formula for the sample size benefit for two groups. The sample size needed in within-designs (NW) with more than 2 conditions, relative to the sample needed in between-designs (NB), assuming normal distributions and compound symmetry, and ignoring the difference in degrees of freedom between the two types of tests, is (from Maxwell & Delaney, 2004, p. 562, formula 47):

$$N_W = \frac{N_B(1-\rho)}{a}$$

Where a is the number of within-subject levels.

6.1 The relation between Cohen's f and Cohen's d

Whereas in the case of a repeated measures ANOVA with 2 groups we could explain the principles of a power analysis by comparing our test against a t -test and Cohen's d , this becomes more difficult when we have more than 2 groups. It is more useful to explain how to directly calculate Cohen's f , the effect size used in power analyses for ANOVA. Cohen's f is calculated following Cohen, 1988, formula 8.2.1 and 8.2.2:

$$f = \sqrt{\frac{\sum \frac{(\mu - \bar{\mu})^2}{N}}{\sigma^2}}$$

Imagine we have a within-subject experiment with 3 conditions. We ask people what they mood is when their alarm clock wakes them up, when they wake up naturally on a week day, and when they wake up naturally on a weekend day. Based on pilot data, we expect the means (on a 7 point validated mood scale) are 3.8, 4.2, and 4.3. The standard deviation is 0.9, and the correlation between the dependent measurements is 0.7. We can calculate Cohen's f for the ANOVA, and Cohen's d_z for the contrasts:

```
mu <- c(3.8, 4.2, 4.3)
sd <- 0.9
f <- sqrt(sum((mu-mean(mu))^2)/length(mu))/sd #Cohen, 1988, formula 8.2.1 and 8.2.2
f
```

```
## [1] 0.2400274
```

```
r <- 0.7
(4.2-3.8)/0.9/sqrt(2*(1-r))
```

```
## [1] 0.5737753
```

```
(4.3-3.8)/0.9/sqrt(2*(1-r))
```

```
## [1] 0.7172191
```

```
(4.3-4.2)/0.9/sqrt(2*(1-r))
```

```
## [1] 0.1434438
```

The relation between Cohen's d or d_z and Cohen's f becomes more difficult when there are multiple groups, because the relationship depends on the pattern of the means. Cohen (1988) presents calculations for three patterns, minimal variability (for example, for 5 means: -0.25, 0, 0, 0, 0.25), medium variability (for example, for 5 means: -0.25, -0.25, 0.25, 0.25, 0.25 or -0.25, -0.25, -0.25, 0.25, 0.25). For these three patterns, formula's are available that compute Cohen's f from Cohen's d , where d is the effect size calculated for the difference between the largest and smallest mean (if the largest mean is 0.25 and the smallest mean is -0.25, $0.25 - -0.25 = 0.5$, so d is 0.5 divided by the standard deviation of 0.9). In our example, d would be $(4.3-3.8)/0.9 = 0.5555556$. If we divide this value by $\sqrt{2*(1-r)}$ we have $d_z = 0.5555556/0.7745967 = 0.7172191$.

I have created a custom function that will calculate f from d , based on a specification of one of the three patterns of means. Our pattern is most similar (but not identical) to a maximum variability pattern (two means are high, one is lower). So we could attempt to calculate f from d (0.5555556), by calculating d from the largest and smallest mean:

```
source("https://raw.githubusercontent.com/Lakens/ANOVA_power_simulation/master/calc_f_d_eta.R")
res <- calc_f_d_eta(mu = mu, sd = sd, variability = "maximum")
res$f
```

```
## [1] 0.2618914
```

```
res$d
```

```
## [1] 0.5555556
```

We see the Cohen's f value is 0.2618914 and $d = 0.5555556$. The Cohen's f is not perfectly accurate - it is assuming the pattern of means is 3.8, 4.3, 4.3, and not 3.8, 4.2, 4.3. If the means and sd is known, it is best to calculate Cohen's f directly from these values.

6.2 Three within conditions, medium effect size

We can perform power analyses for within designs using simulations. We set groups to 3 for the simulation, $n = 20$, and the correlation between dependent variables to 0.8. If the true effect size is $f = 0.25$, and the alpha level is 0.05, the power is 96.6%.

In this case, we simulate data with means -0.3061862, 0.0000000, and 0.3061862, and set the sd to 1.

```

K <- 3
n <- 20
sd <- 1
r <- 0.8
alpha = 0.05
f <- 0.25
f2 <- f^2
ES <- f2/(f2+1)
ES

```

```
## [1] 0.05882353
```

```

mu <- mu_from_ES(K = K, ES = ES)
sqrt(sum((mu-mean(mu))^2)/length(mu))/sd #Cohen, 1988, formula 8.2.1 and 8.2.2

```

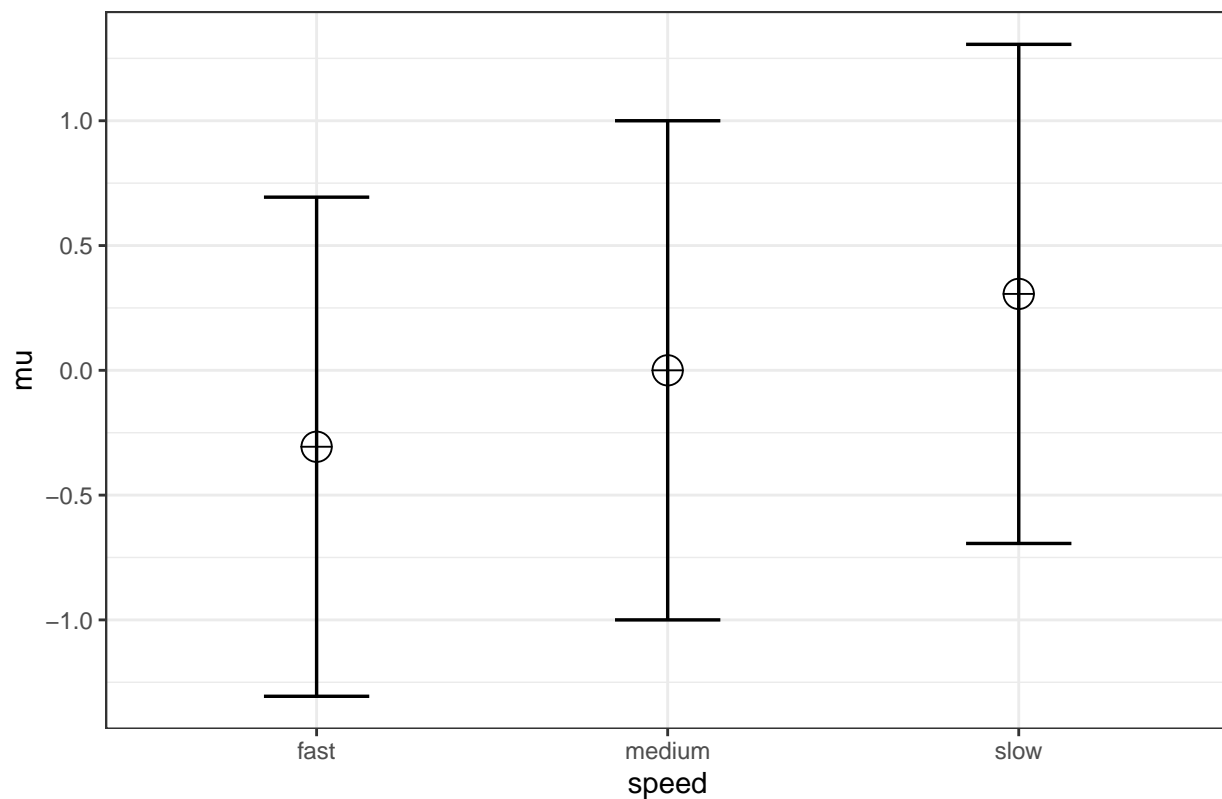
```
## [1] 0.25
```

```

design = paste(K, "w", sep="")
labelnames <- c("speed", "fast", "medium", "slow")
design_result <- ANOVA_design(design = design,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             labelnames = labelnames)

```

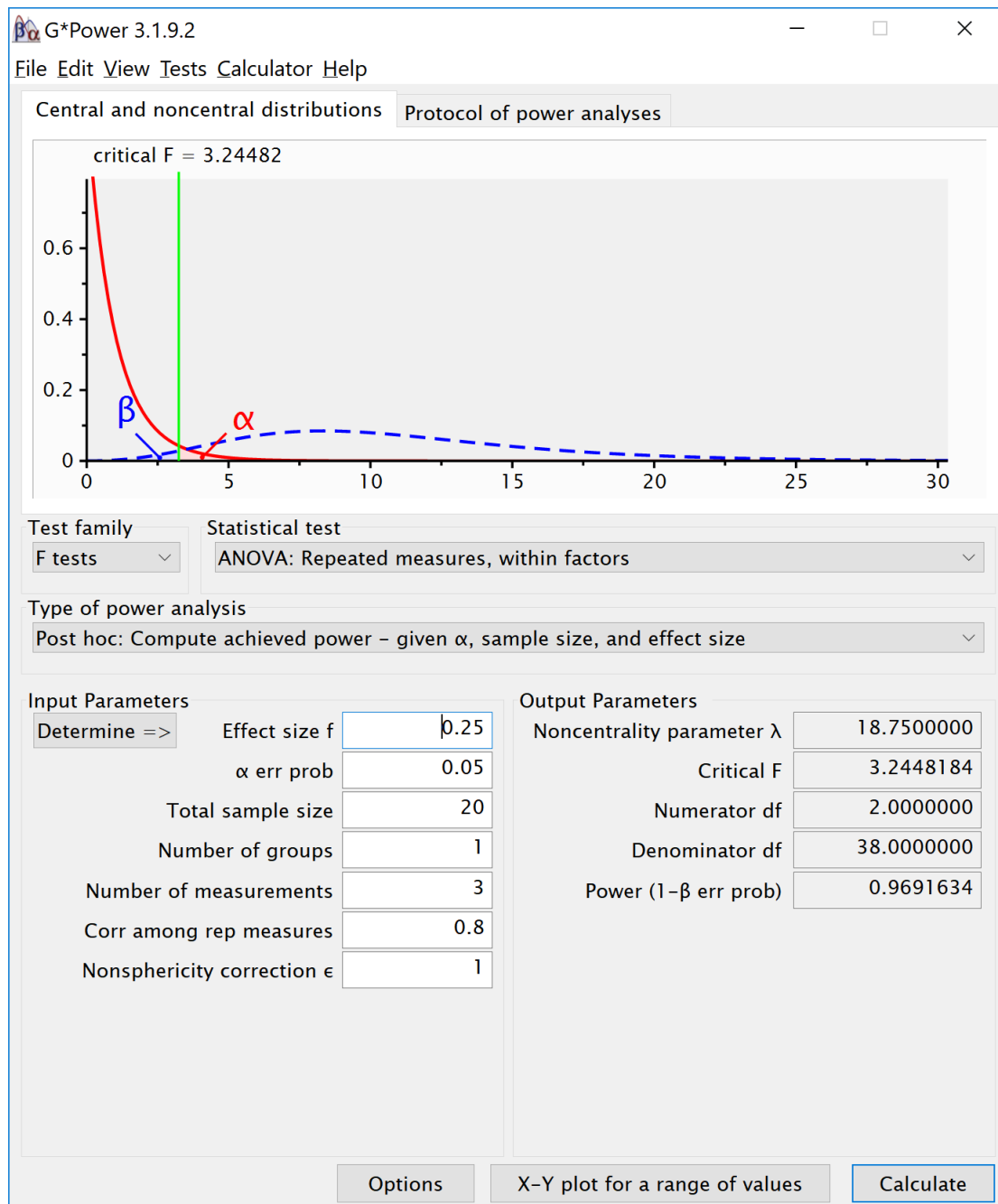
Means for each condition in the design



```
alpha_level <- 0.05
ANOVA_power(design_result, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_speed    95      0.3583
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_speed_fast_speed_medium    60      0.5350
## p_speed_fast_speed_slow      97      1.0342
## p_speed_medium_speed_slow    53      0.5059
##
## Within-Subject Factors Included: Check MANOVA Results
```

The results of the simulation are indeed very close to 96.9%. We can see this is in line with the power estimate from Gpower:

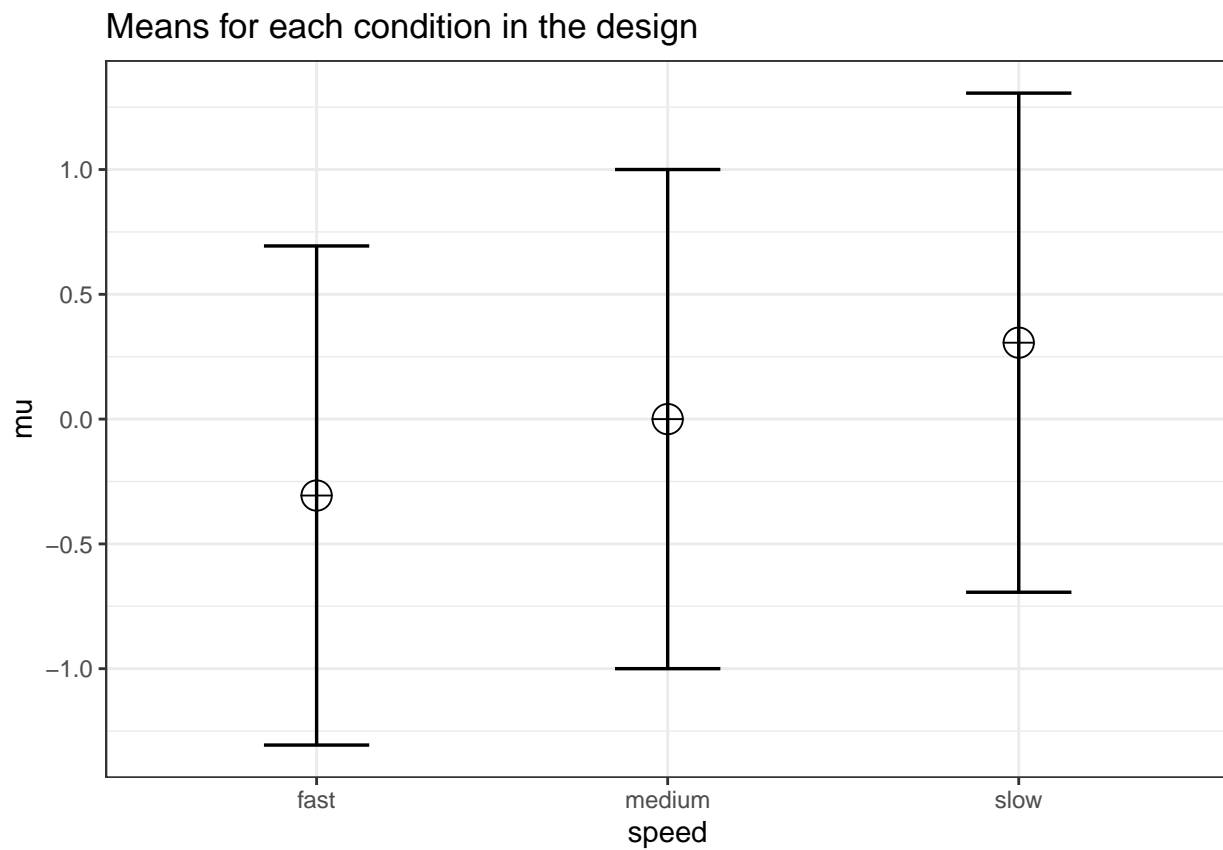


We can also validate this by creating the code to do a power analysis in R from scratch:

```
K <- 3 #three groups
n <- 20
sd <- 1
r <- 0.8
alpha = 0.05
f <- 0.25
f2 <- f^2
ES <- f2/(f2+1)
ES
```

```
## [1] 0.05882353
```

```
mu <- mu_from_ES(K = K, ES = ES)
design = paste(K, "w", sep = "")
labelnames <- c("speed", "fast", "medium", "slow")
design_result <- ANOVA_design(design = design,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             labelnames = labelnames)
```



```
power_oneway_within(design_result)$power
```

```
## [1] 0.9691634
```

```
power_oneway_within(design_result)$eta_p_2
```

```
## [1] 0.05882353
```

```
power_oneway_within(design_result)$eta_p_2_SPSS
```

```
## [1] 0.3303965
```



```
power_oneway_within(design_result)$Cohen_f
```

```
## [1] 0.25
```

```
power_oneway_within(design_result)$Cohen_f_SPSS
```

```
## [1] 0.7024394
```

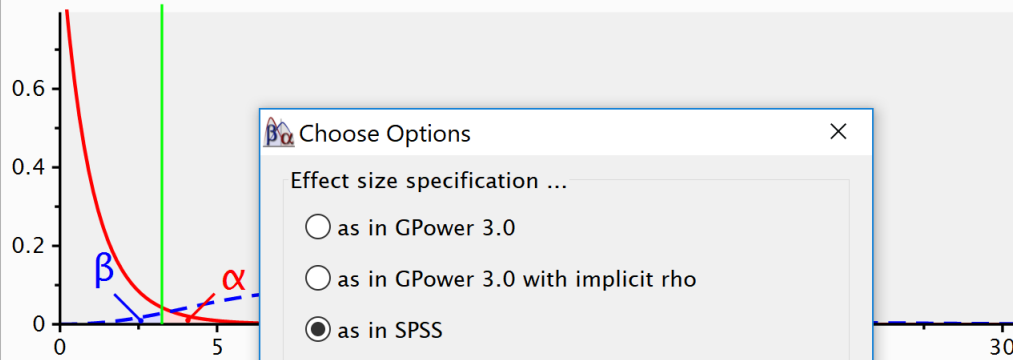
We can even check the calculation of Cohen's f SPSS style in GPower. We take the GPower settings as illustrated above. We click the 'Options' button, and check the radiobutton next to 'As in SPSS'. Click ok, and you will notice that the 'Corr among rep measures' field has disappeared. The correlation does not need to be entered separately, but is incorporated in Cohen's f . The value of Cohen's f , which was 0.25, has changed into 0.7024394. This is the SPSS equivalent. The value is much larger. This value, and its corresponding partial eta-squared, incorporate the correlation between observations.

G*Power 3.1.9.2

File Edit View Tests Calculator Help

Central and noncentral distributions Protocol of power analyses

critical F = 3.24482



Choose Options

Effect size specification ...

- ☐ as in GPower 3.0
- ☐ as in GPower 3.0 with implicit rho
- ☒ as in SPSS
- ☐ as in Cohen (1988) – recommended

Cancel OK

Test family: F tests

Statistical test: ANOVA

Type of power analysis: Post hoc: Compute achieved power – given α , sample size, and effect size

Input Parameters

Determine => Effect size f(U): 0.7024394

α err prob: 0.05

Total sample size: 20

Number of groups: 1

Number of measurements: 3

Nonsphericity correction ϵ : 1

Output Parameters

Noncentrality parameter λ : ?

Critical F: ?

Numerator df: ?

Denominator df: ?

Power ($1 - \beta$ err prob): ?

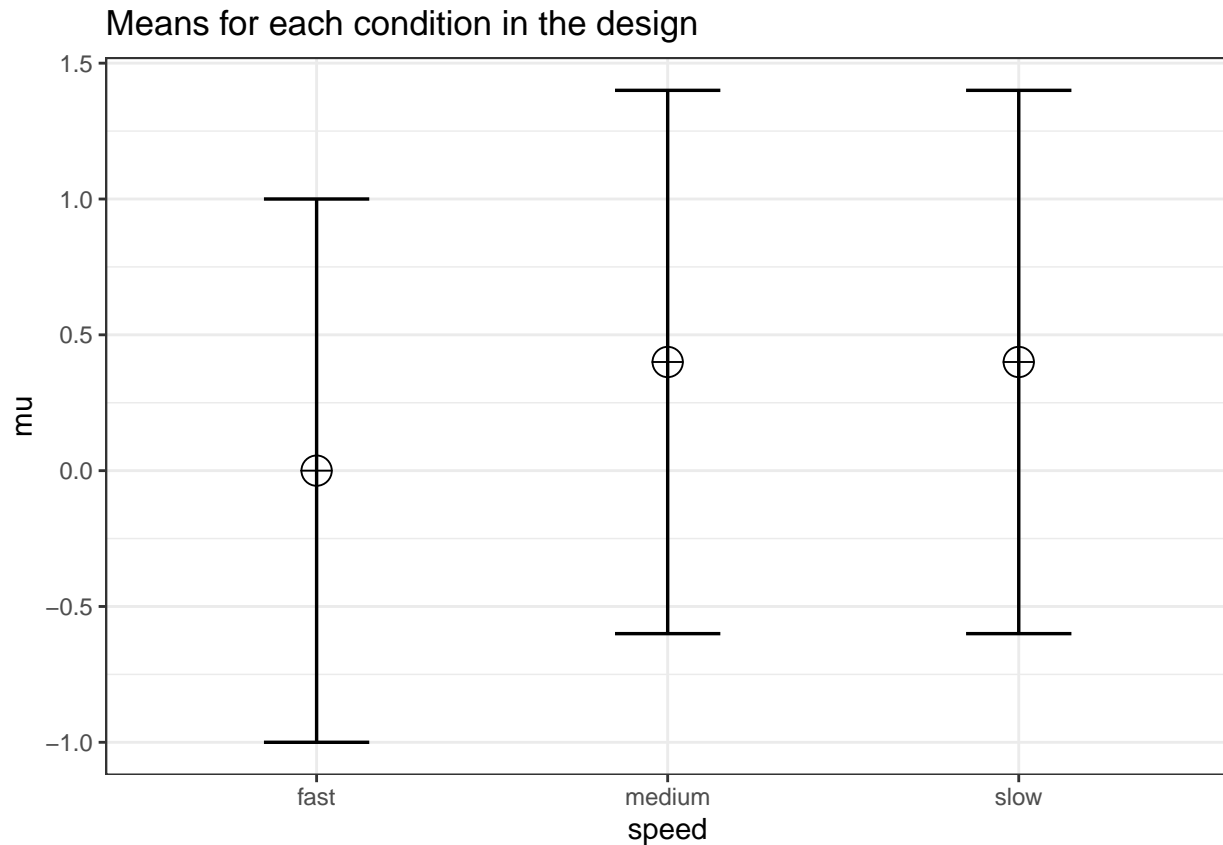
Options X-Y plot for a range of values Calculate

Chapter 7

Repeated Measures-ANOVA Part 3

We first repeat the simulation by Brysbaert:

```
# give sample size
N = 75
# give effect size d
d1 = .4 #difference between the extremes
d2 = .4 #third condition goes with the highest extreme
# give the correlation between the conditions
r = .5
# give number of simulations
nSim = nsims
# give alpha levels
alpha1 = .05 #alpha level for the omnibus ANOVA
alpha2 = .05 #also adjusted from original by DL
# create progress bar in case it takes a while
#pb <- winProgressBar(title = "progress bar", min = 0, max = nSim, width = 300)
# create vectors to store p-values
p1 <- numeric(nSim) #p-value omnibus ANOVA
p2 <- numeric(nSim) #p-value first post hoc test
p3 <- numeric(nSim) #p-value second post hoc test
p4 <- numeric(nSim) #p-value third post hoc test
# open library MASS
library('MASS')
# define correlation matrix
rho <- cbind(c(1, r, r), c(r, 1, r), c(r, r, 1))
# define participant codes
part <- paste("part", seq(1:N))
for(i in 1:nSim){ #for each simulated experiment
  # setWinProgressBar(pb, i, title=paste(round(i/nSim*100, 1), "% done"))
  data = mvrnorm(n=N, mu=c(0, 0, 0), Sigma=rho)
  data[,2] = data[,2]+d1
  data[,3] = data[,3]+d2
  datalong = c(data[,1], data[,2], data[,3])
  conds= factor(rep(letters[24:26], each = N))
  partID = factor(rep(part, times = 3))
  output <- data.frame(partID, conds, datalong)
  test <- aov(datalong~conds + Error(partID/conds), data=output)
  tests <- (summary(test))
```

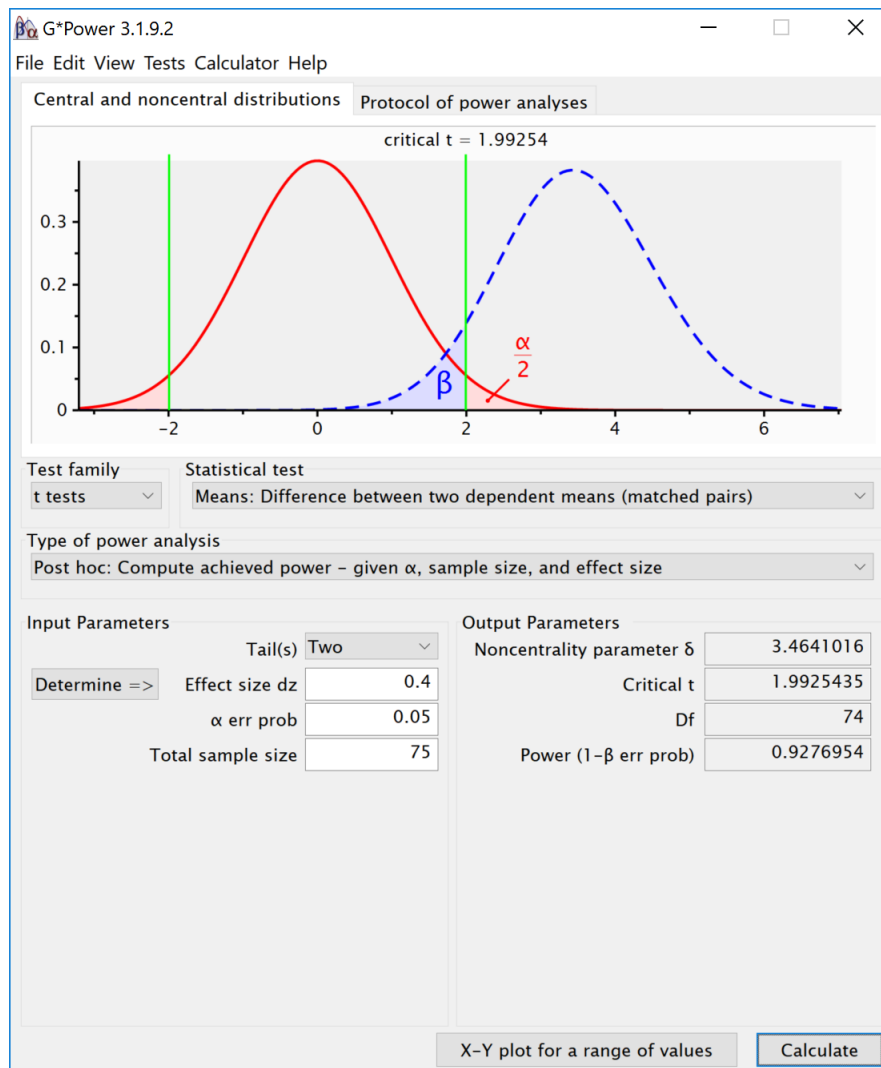



```
ANOVA_power(design_result, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_speed    94    0.09853
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_speed_fast_speed_medium    91    0.3842
## p_speed_fast_speed_slow      93    0.3887
## p_speed_medium_speed_slow     6    0.0078
##
## Within-Subject Factors Included: Check MANOVA Results
```

```
#Results
```

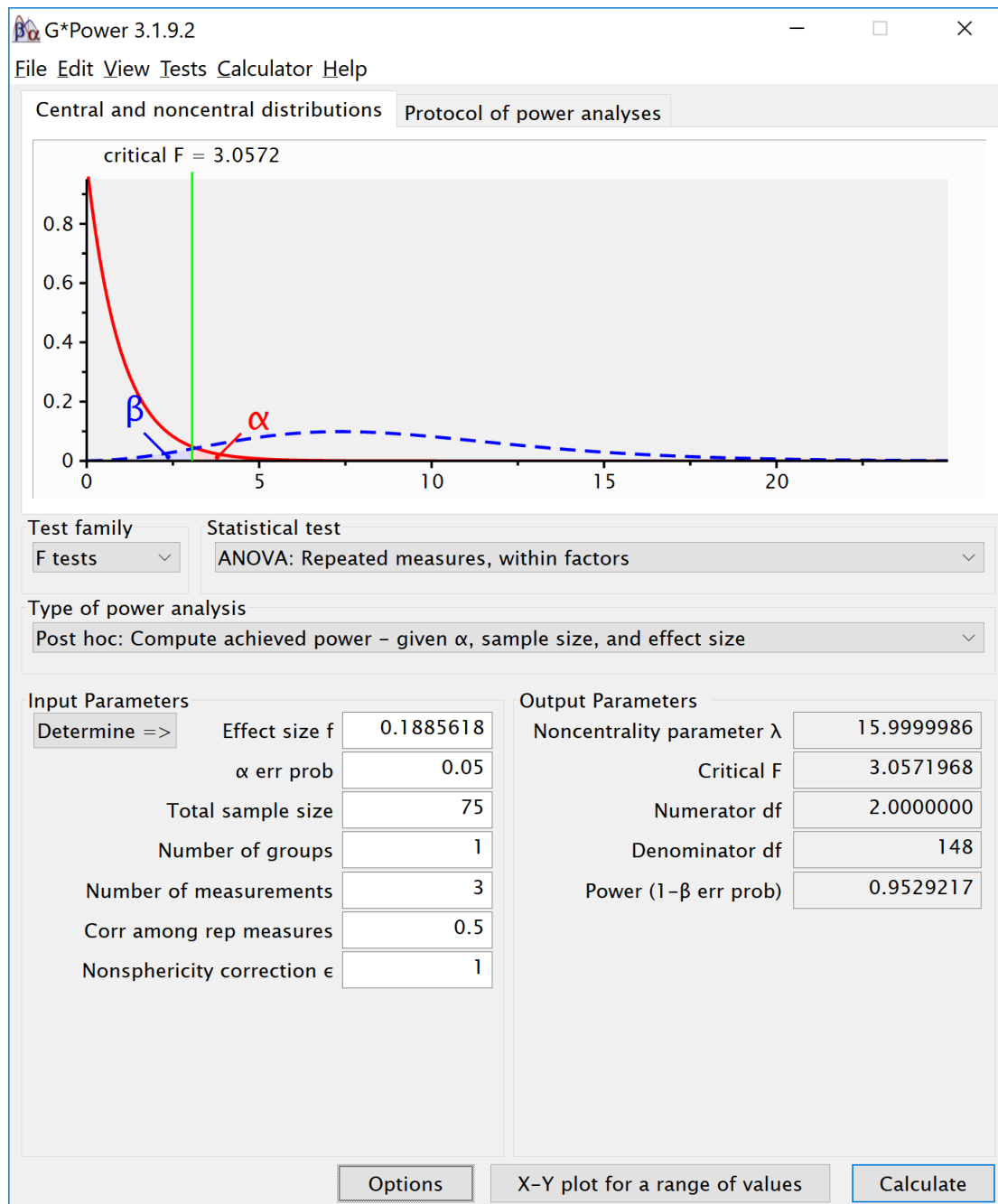
The results of the simulation are very similar. Power for the ANOVA F -test is around 95.2%. For the three paired t -tests, power is around 92.7. This is in line with the a-priori power analysis when using g^* power:



We can perform an post-hoc power analysis in G*power. We can calculate Cohen's f based on the means and sd, using our own custom formula.

```
# Our simulation is based on the following means and sd:
# mu <- c(0, 0.4, 0.4)
# sd <- 1
f <- sqrt(sum((mu-mean(mu))^2)/length(mu))/sd #Cohen, 1988, formula 8.2.1 and 8.2.2
# We can see why f = 0.5*d.
# Imagine 2 group, mu = 1 and 2
# Grand mean is 1.5, we have sqrt(sum(0.5^2 + 0.5^2)/2), or sqrt(0.5/2), = 0.5.
# For Cohen's d we use the difference, 2-1 = 1.
```

The Cohen's f is 0.1885618. We can enter the f (using the default 'as in G*Power 3.0' in the option window) and enter a sample size of 75, number of groups as 1, number of measurements as 3, correlation as 0.5. This yields:



7.2 Reproducing Brysbaert Variation 1 Changing Correlation

```
# give sample size
N = 75
# give effect size d
d1 = .4 #difference between the extremes
d2 = .4 #third condition goes with the highest extreme
# give the correlation between the conditions
r = .6 #increased correlation
```

```

# give number of simulations
nSim = nsims
# give alpha levels
alpha1 = .05 #alpha level for the omnibus ANOVA
alpha2 = .05 #also adjusted from original by DL
# create progress bar in case it takes a while
#pb <- winProgressBar(title = "progress bar", min = 0, max = nSim, width = 300)
# create vectors to store p-values
p1 <- numeric(nSim) #p-value omnibus ANOVA
p2 <- numeric(nSim) #p-value first post hoc test
p3 <- numeric(nSim) #p-value second post hoc test
p4 <- numeric(nSim) #p-value third post hoc test
# open library MASS
library('MASS')
# define correlation matrix
rho <- cbind(c(1, r, r), c(r, 1, r), c(r, r, 1))
# define participant codes
part <- paste("part", seq(1:N))
for(i in 1:nSim){ #for each simulated experiment
  # setWinProgressBar(pb, i, title=paste(round(i/nSim*100, 1), "% done"))
  data = mvrnorm(n=N, mu=c(0, 0, 0), Sigma=rho)
  data[,2] = data[,2]+d1
  data[,3] = data[,3]+d2
  datalong = c(data[,1],data[,2],data[,3])
  conds= factor(rep(letters[24:26], each = N))
  partID = factor(rep(part, times = 3))
  output <- data.frame(partID,conds,dalong)
  test <- aov(dalong~conds + Error(partID/conds), data=output)
  tests <- (summary(test))
  p1[i] <- tests$'Error: partID:conds'[[1]]$'Pr(>F)'[[1]]
  p2[i] <- t.test(data[,1],data[,2], paired=TRUE)$p.value
  p3[i] <- t.test(data[,1],data[,3], paired=TRUE)$p.value
  p4[i] <- t.test(data[,2],data[,3], paired=TRUE)$p.value
}
#close(pb)#close progress bar
#printing all unique tests (adjusted code by DL)
sum(p1<alpha1)/nSim

```

```
## [1] 0.94
```

```
sum(p2<alpha2)/nSim
```

```
## [1] 0.93
```

```
sum(p3<alpha2)/nSim
```

```
## [1] 0.92
```

```
sum(p4<alpha2)/nSim
```

```
## [1] 0.01
```



```

design <- "3w"
n <- 75
mu <- c(0, 0.4, 0.4)
sd <- 1
r <- 0.6
labelnames <- c("speed", "fast", "medium", "slow")

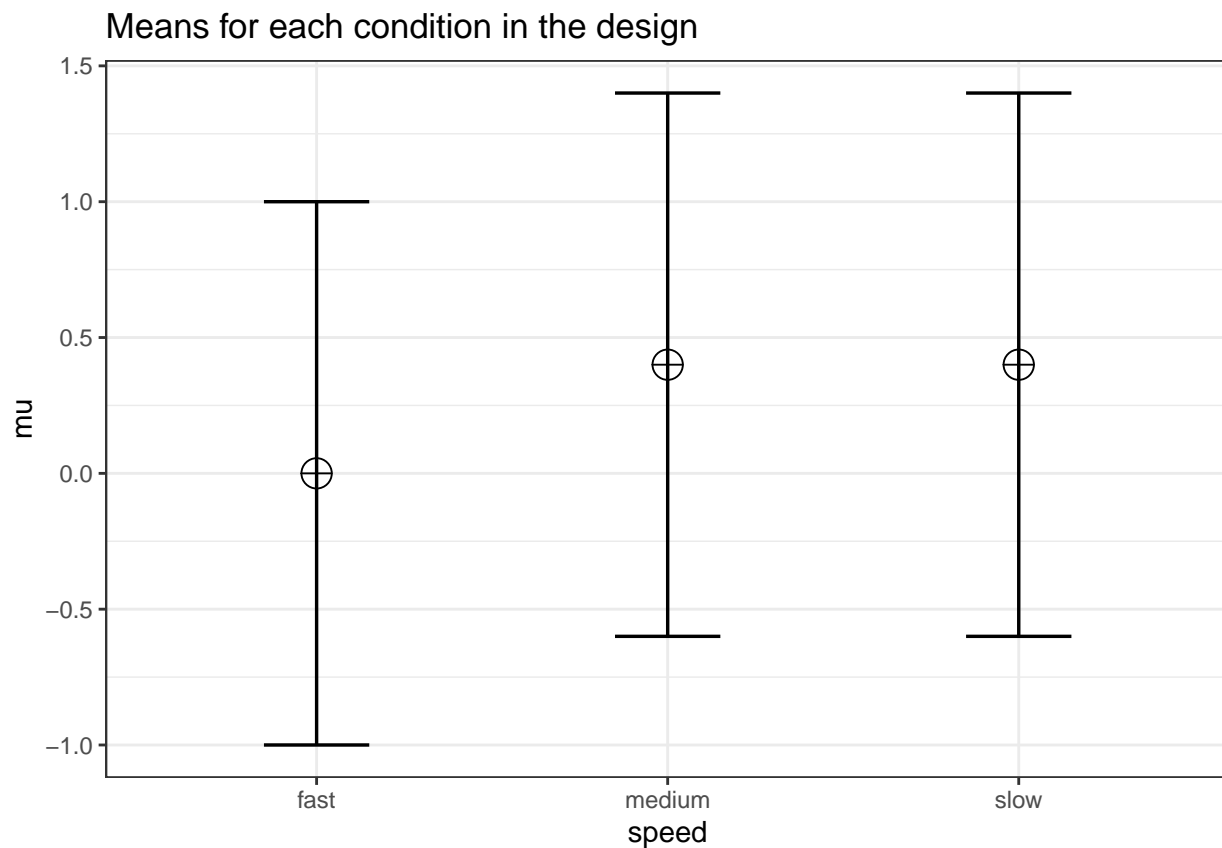
```

We create the within design, and run the simulation

```

design_result <- ANOVA_design(design = design,
  n = n,
  mu = mu,
  sd = sd,
  r = r,
  labelnames = labelnames)

```



```
ANOVA_power(design_result, nsims = nsims)
```

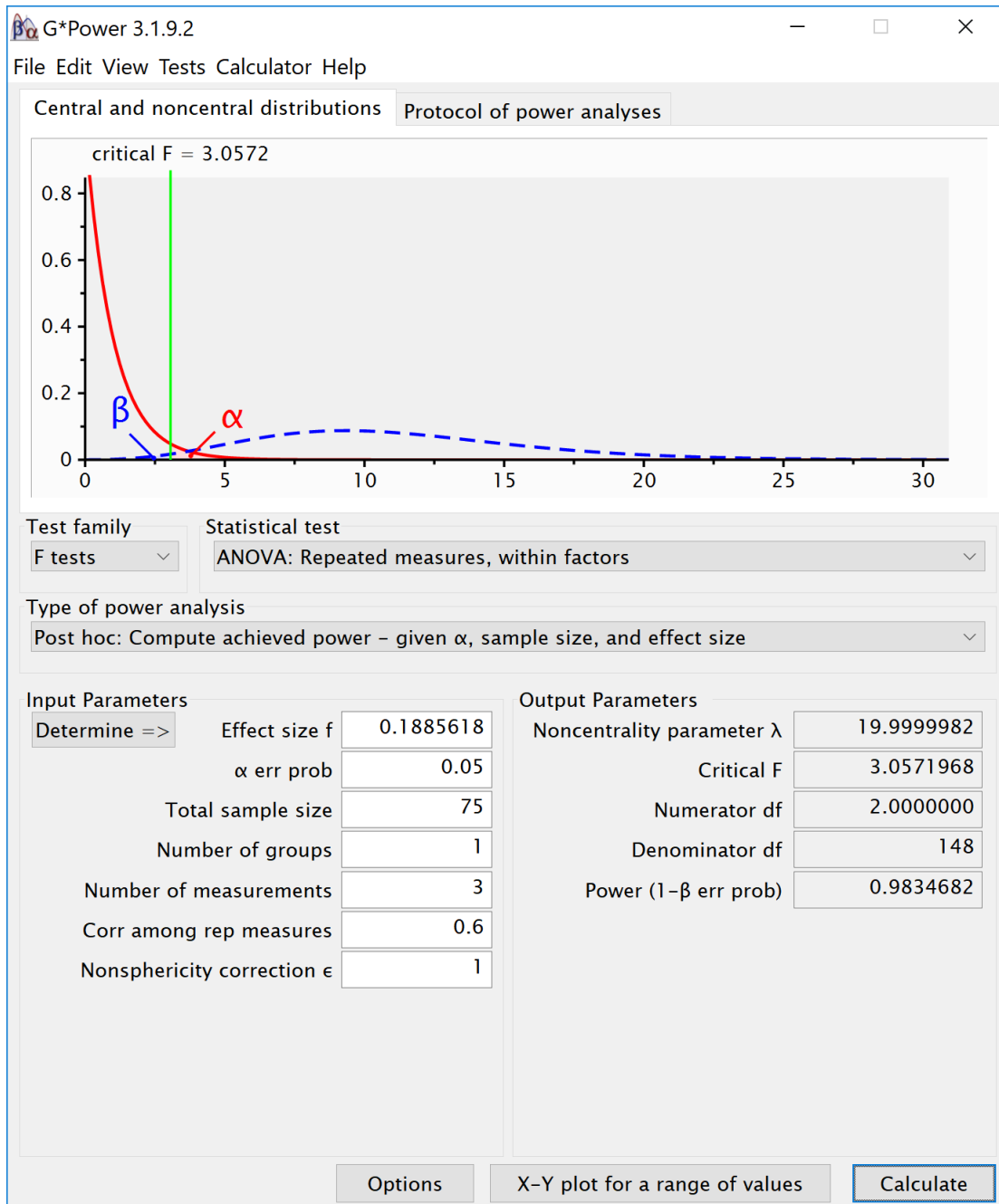
```

## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_speed    99    0.1223
##
## Power and Effect sizes for contrasts
##           power effect_size

```

```
## p_speed_fast_speed_medium    98    0.436167
## p_speed_fast_speed_slow      97    0.443268
## p_speed_medium_speed_slow     8   -0.001819
##
## Within-Subject Factors Included: Check MANOVA Results
```

Again, this is similar to g*power for the ANOVA:

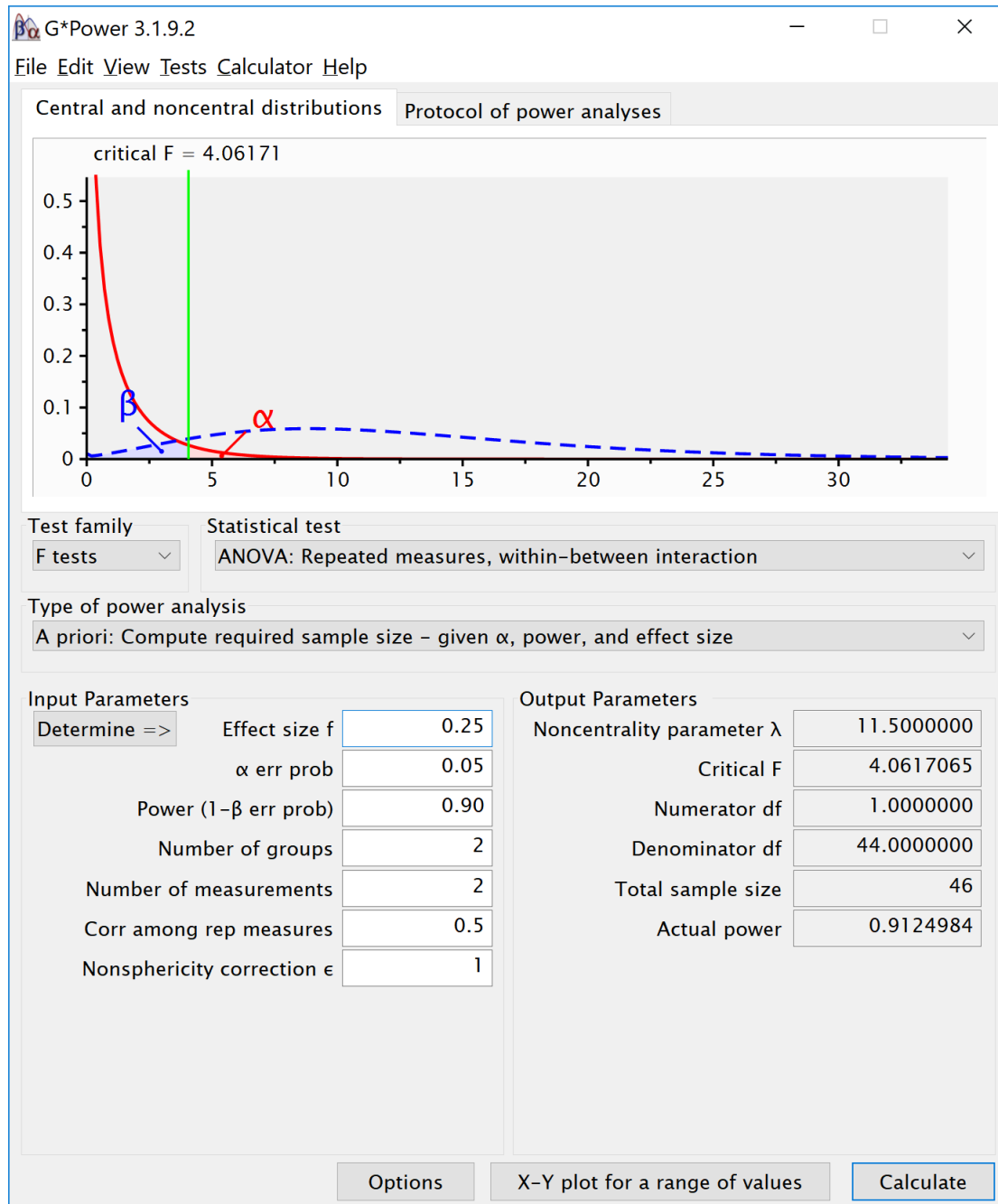


Chapter 8

Mixed ANOVA Part 1

8.1 Two by two ANOVA, within-between design

We can simulate a Two-Way ANOVA with a specific alpha, sample size and effect size, to achieve a specified statistical power. We will try to reproduce the power analysis by g*power for an F-test, ANOVA: Repeated measures, within-between interaction.



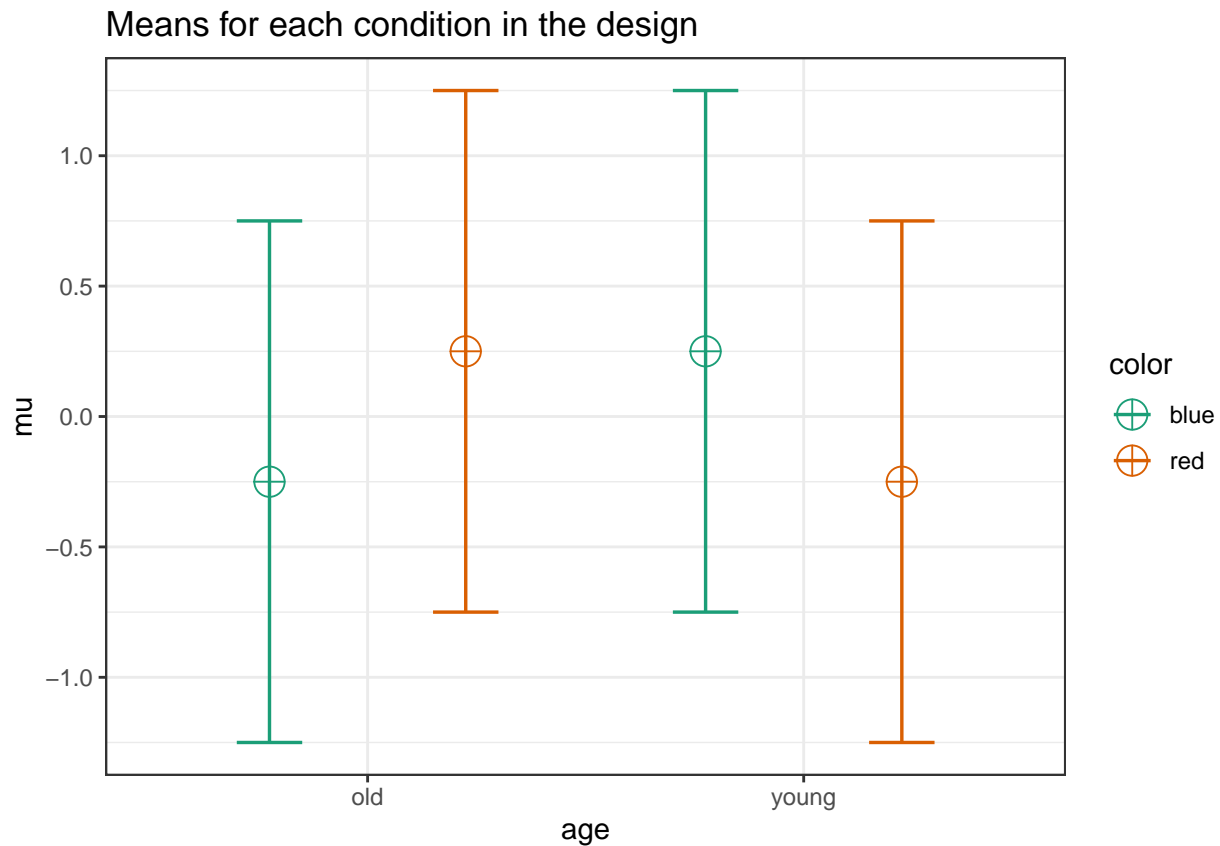
For the 2-way interaction, the result should be a power of 91.25% if we have a total sample size of 46. Since we have 2 groups in the between factor that means the sample size per group is 2 (and both these groups collect 2 repeated measures).

```
mu <- c(-0.25, 0.25, 0.25, -0.25)
n <- 23
sd <- 1
r <- 0.5
string = "2w*2b"
alpha_level <- 0.05
labelnames = c("age", "old", "young", "color", "blue", "red")
design_result <- ANOVA_design(design = string,
```

```

n = n,
mu = mu,
sd = sd,
r = r,
labelnames = labelnames)

```



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

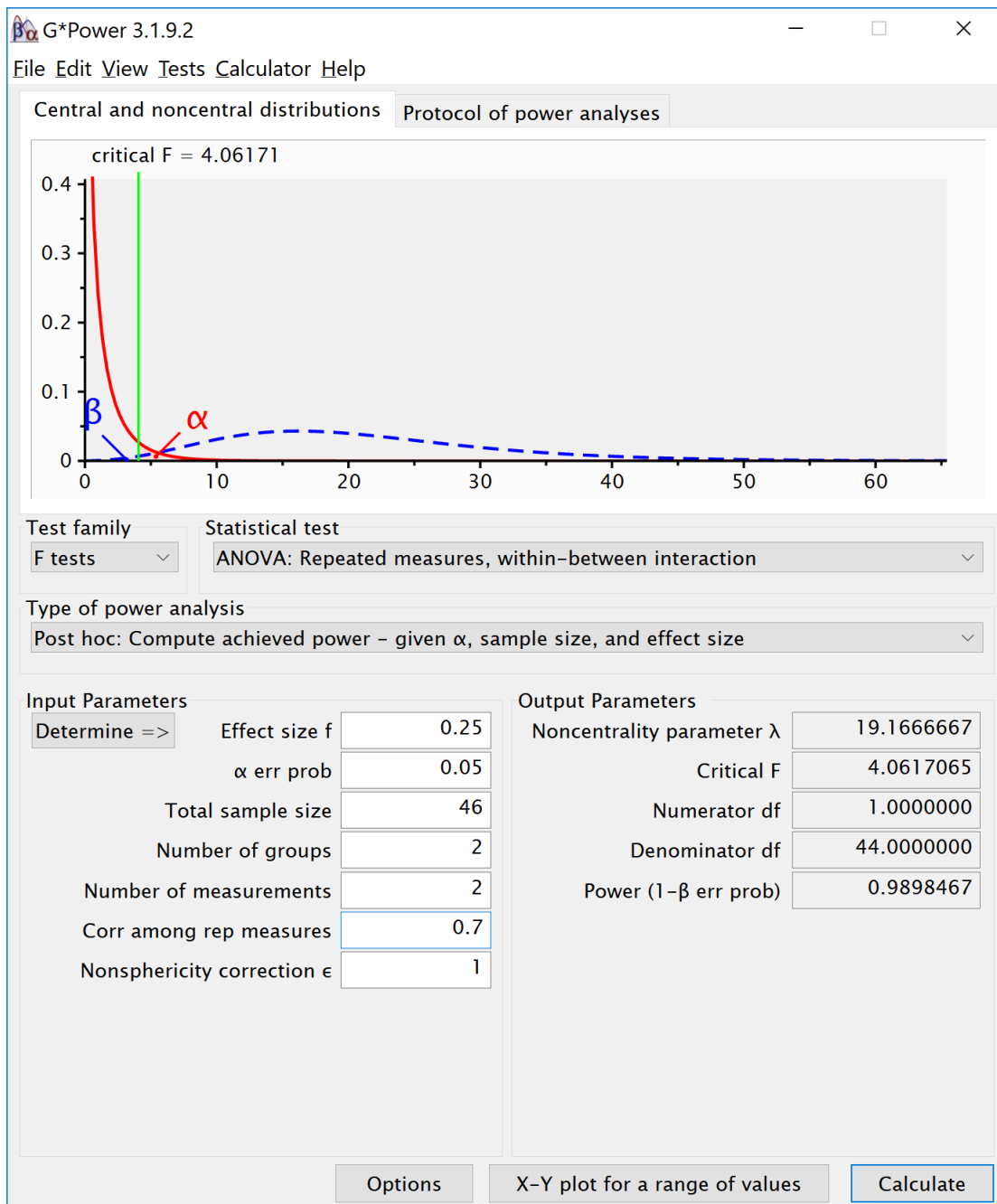
```

## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_color      5    0.02118
## anova_age         6    0.02362
## anova_color:age   91    0.20318
##
## Power and Effect sizes for contrasts
##                                     power effect_size
## p_age_old_color_blue_age_old_color_red    28    0.46087
## p_age_old_color_blue_age_young_color_blue  56    0.49235
## p_age_old_color_blue_age_young_color_red    9   -0.02785
## p_age_old_color_red_age_young_color_blue    5    0.01655
## p_age_old_color_red_age_young_color_red    59   -0.50668
## p_age_young_color_blue_age_young_color_red  37   -0.50244
##
## Within-Subject Factors Included: Check MANOVA Results

```

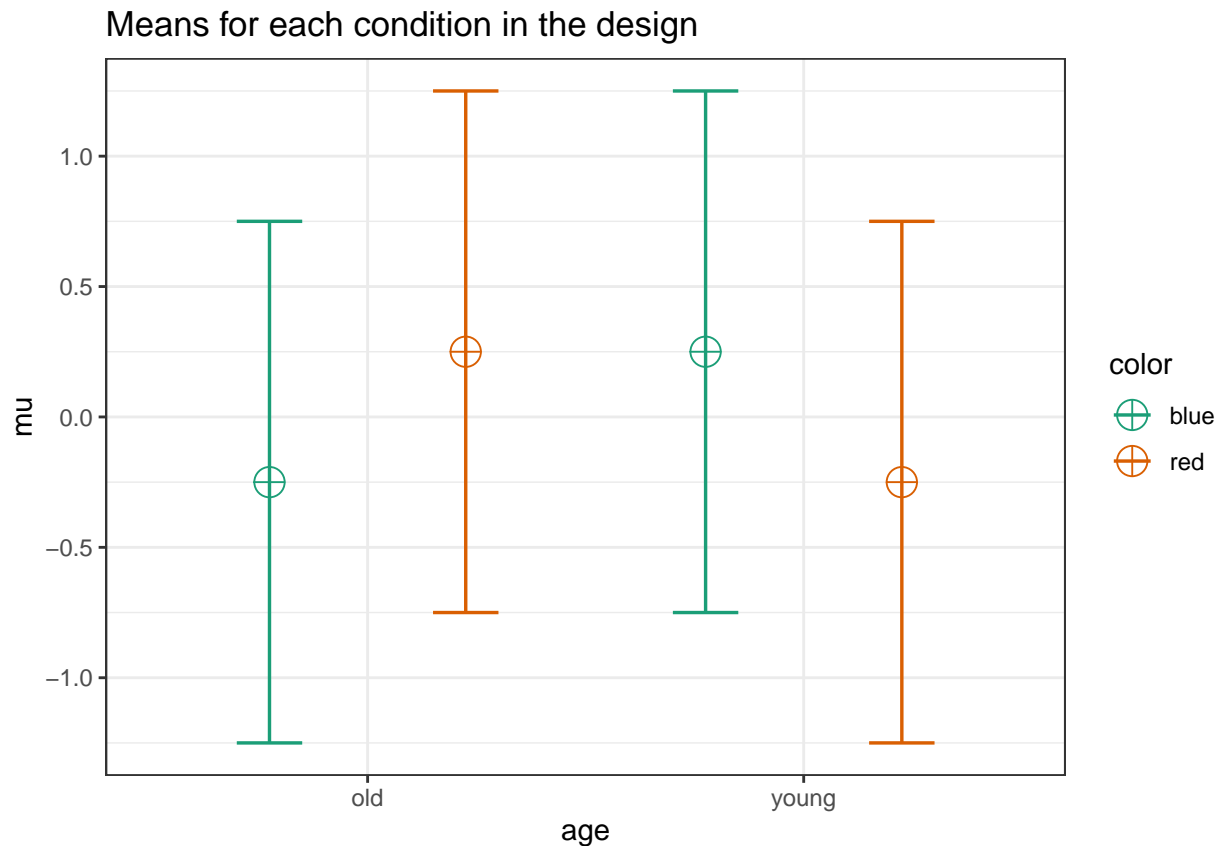
8.2 Two by two ANOVA, within-between design Variation 1

We can simulate the same Two-Way ANOVA increasing the correlation to 0.7.



```
mu <- c(-0.25, 0.25, 0.25, -0.25)
n <- 23
sd <- 1
r <- 0.7
string = "2w*2b"
alpha_level <- 0.05
labelnames = c("age", "old", "young", "color", "blue", "red")
```

```
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             labelnames = labelnames)
```



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

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_color      2    0.01759
## anova_age        7    0.02260
## anova_color:age  98    0.29823
##
## Power and Effect sizes for contrasts
##                                     power effect_size
## p_age_old_color_blue_age_old_color_red      45    0.54169
## p_age_old_color_blue_age_young_color_blue    87    0.68826
## p_age_old_color_blue_age_young_color_red      5    0.04402
## p_age_old_color_red_age_young_color_blue      2   -0.01272
## p_age_old_color_red_age_young_color_red      77   -0.62245
## p_age_young_color_blue_age_young_color_red    34   -0.47829
##
## Within-Subject Factors Included: Check MANOVA Results
```


Chapter 9

Mixed ANOVA Part 2

9.1 Two by two ANOVA, within-within 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 approaches provide a solution.

We can reproduce the simulation coded by Ben Amsel

```
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 = nsims # 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

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

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

Run the simulation

```
# 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 $p < .05$.

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

```
##          X1          X2 Interaction
##          3          6          47
```

Visualize the distributions of p-values

```

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

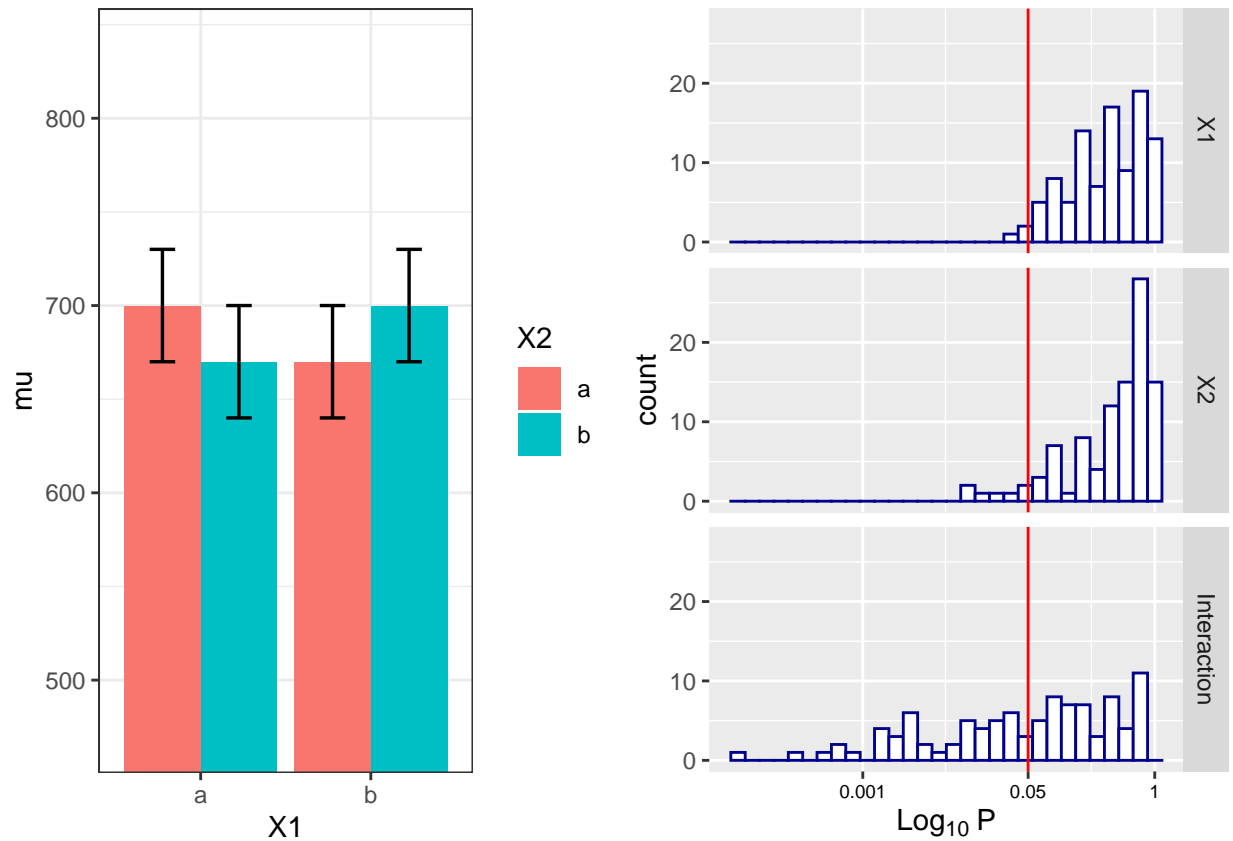
means = data.frame(cond[1:4, ], mu, SE = sigma / sqrt(nsubs))
plt1 = ggplot(means, aes(y = mu, x = X1, fill=X2)) +
  geom_bar(position = position_dodge(), stat="identity") +
  geom_errorbar(aes(ymin = mu-SE, ymax = mu+SE),
    position = position_dodge(width=0.9), size=.6, width=.3) +
  coord_cartesian(ylim=c(.7*min(mu), 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))

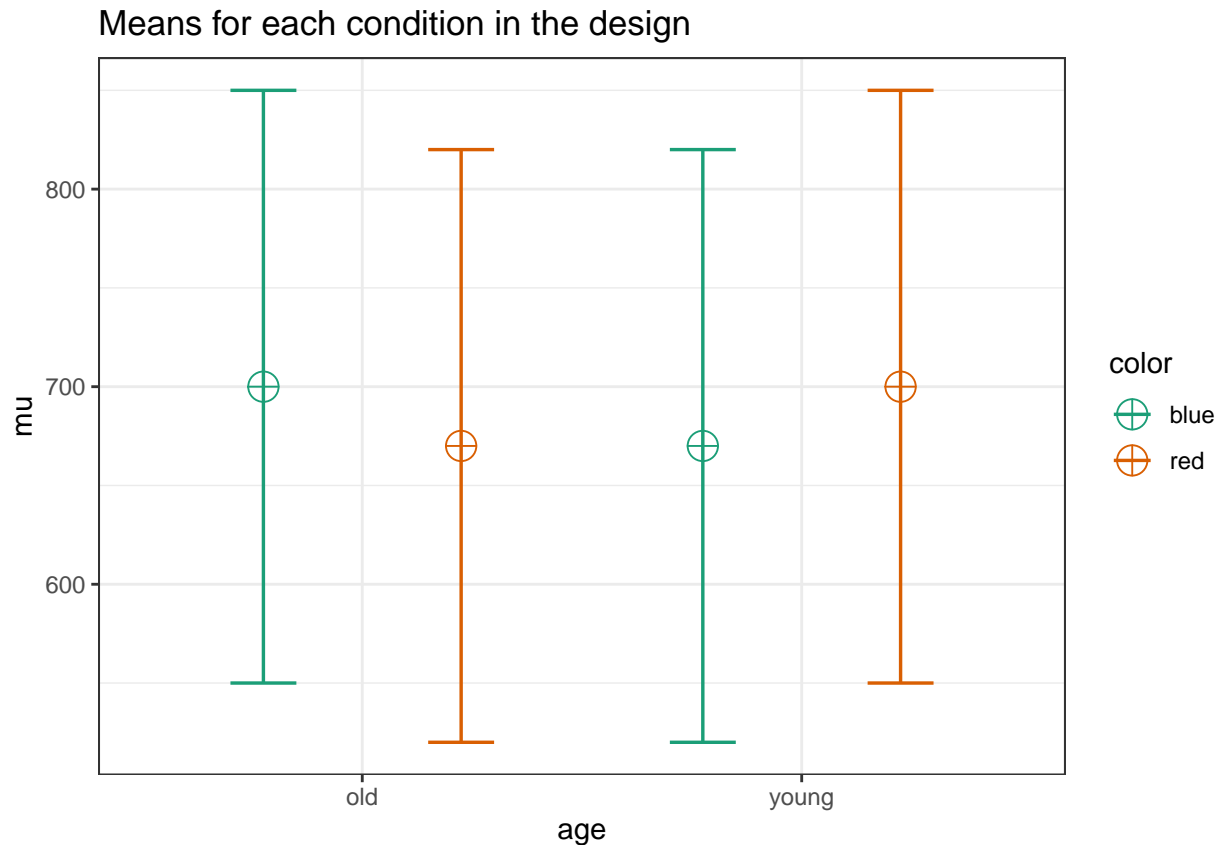
# arrange plots side by side and print
grid.arrange(plt1, plt2, nrow=1)

```



We can reproduce this simulation:

```
mu = c(700, 670, 670, 700) # true effects (in this case, a double dissociation)
sigma = 150 # population standard deviation
n <- 25
sd <- 150
r <- 0.75
string = "2w*2w"
alpha_level <- 0.05
labelnames = c("age", "old", "young", "color", "blue", "red")
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             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      5    0.03539
## anova_color     7    0.04235
## anova_age:color 43    0.16063
##
## Power and Effect sizes for contrasts
##                                     power effect_size
## p_age_old_color_blue_age_old_color_red      19  -0.277372
## p_age_old_color_blue_age_young_color_blue    23  -0.275603
## p_age_old_color_blue_age_young_color_red      6   0.024031
## p_age_old_color_red_age_young_color_blue      4  -0.009016
## p_age_old_color_red_age_young_color_red       28   0.310872
## p_age_young_color_blue_age_young_color_red    28   0.309777
##
## Within-Subject Factors Included: Check MANOVA Results
```

The simulations yield closely matching results.

9.2 Examine variation of means and correlation

```
# define the parameters
mu = c(700, 670, 690, 750) # true effects (in this case, a double dissociation)
sigma = 150 # population standard deviation
rho = 0.4 # correlation between repeated measures
nsubs = 25 # how many subjects?
nsims = nsims # 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

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

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

Run the simulation

```
# 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 $p < .05$.

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

```
##           X1           X2 Interaction
##           7           35           50
```

Visualize the distributions of p-values

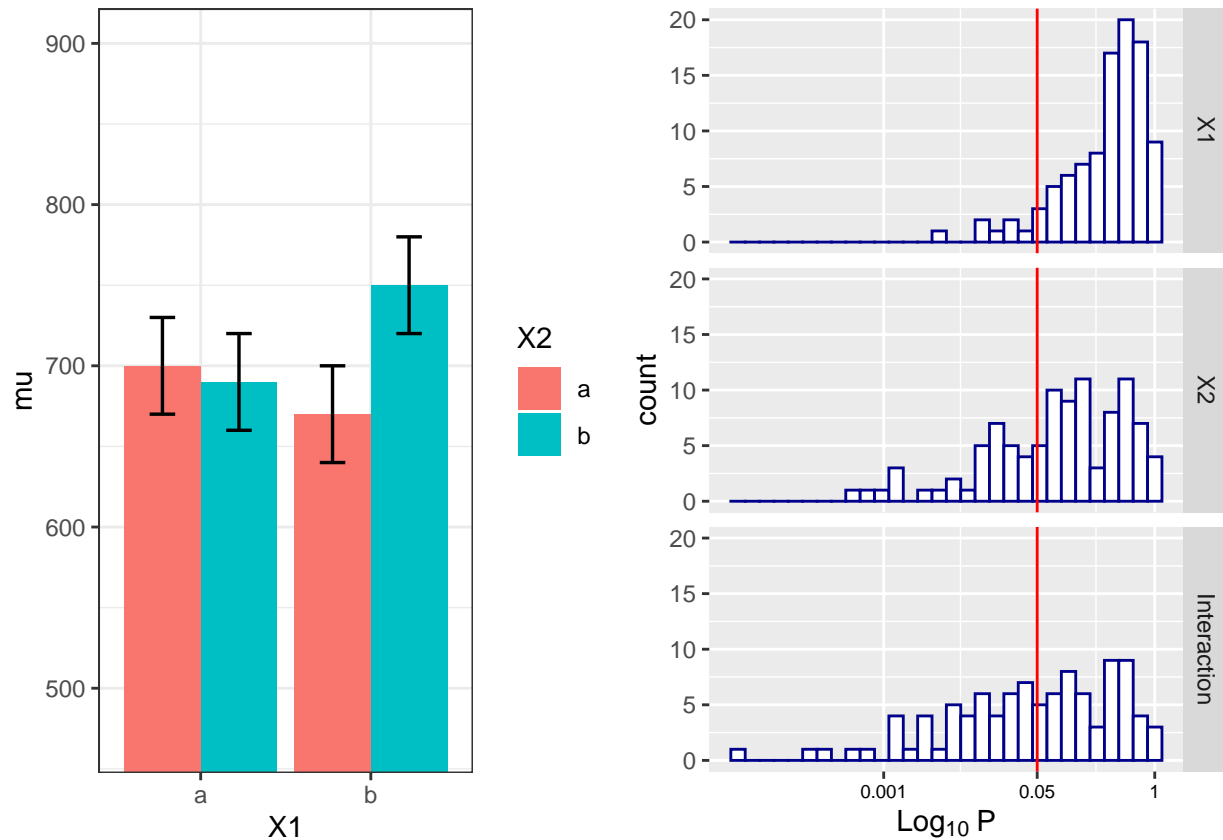
```
# plot the known effects
require(ggplot2)
require(gridExtra)

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')

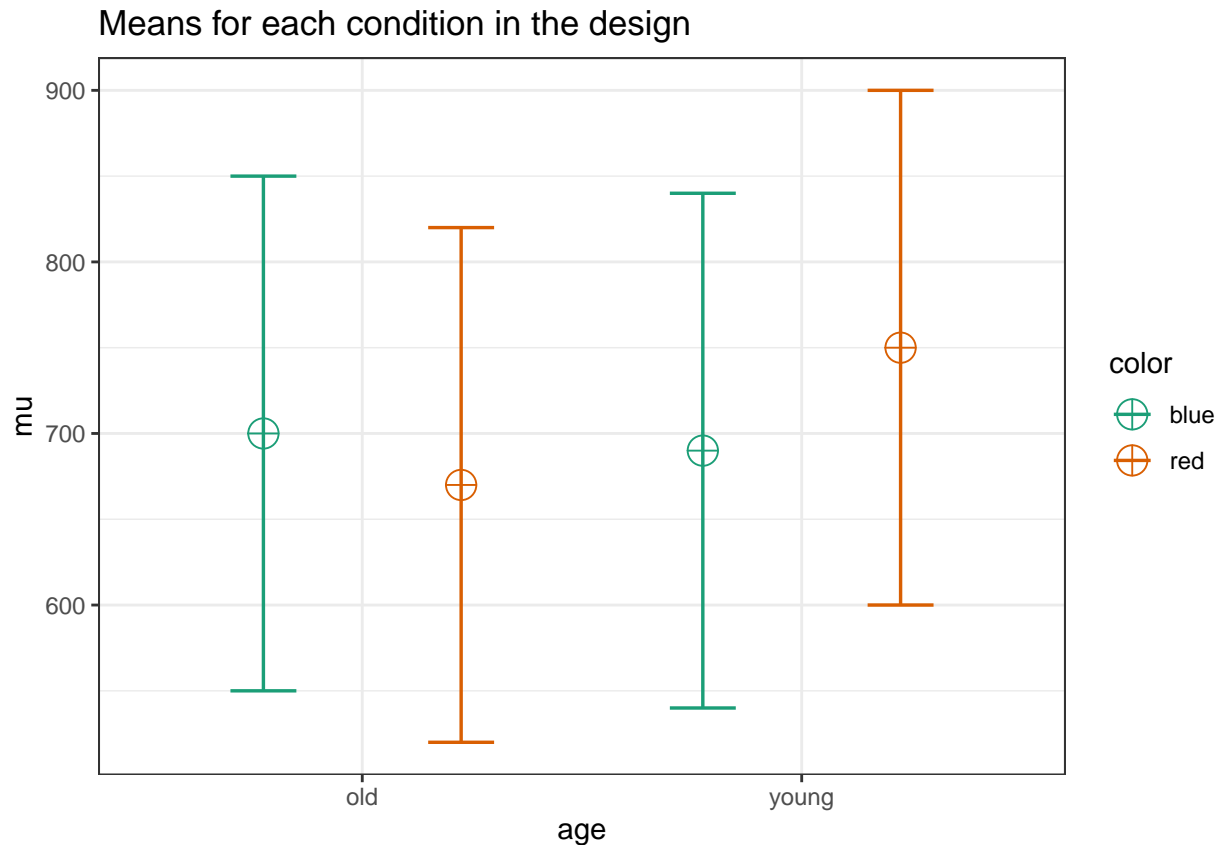
# plot each of the p-value distributions on a log scale
options(scipen = 999) # 'turn off' scientific notation
plt2 = ggplot(plotData, aes(x = p)) +
  scale_x_log10(breaks=c(1, 0.05, 0.001),
    labels=c(1, 0.05, 0.001)) +
  geom_histogram(colour = "darkblue", fill = "white") +
  geom_vline(xintercept = 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, 690, 750) # true effects (in this case, a double dissociation)
sigma = 150 # population standard deviation
n <- 25
sd <- 150
r <- 0.4
string = "2w*2w"
alpha_level <- 0.05
labelnames = c("age", "old", "young", "color", "blue", "red")
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             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      29    0.10813
## anova_color     10    0.05224
## anova_age:color  43    0.14890
##
## Power and Effect sizes for contrasts
##                                     power effect_size
## p_age_old_color_blue_age_old_color_red      16   -0.19631
## p_age_old_color_blue_age_young_color_blue     6   -0.05495
## p_age_old_color_blue_age_young_color_red     25    0.27436
## p_age_old_color_red_age_young_color_blue     11    0.13305
## p_age_old_color_red_age_young_color_red      57    0.46801
## p_age_young_color_blue_age_young_color_red   32    0.32829
##
## Within-Subject Factors Included: Check MANOVA Results
```


Chapter 10

Mixed ANOVA Part 3

10.1 Two by two ANOVA, within design

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

Example

$\rho_A = 0.4$ $\rho_B = 0.8$ $\rho_{AB} = 0.4$

		$\rho_A = 0.4$			$\rho_B = 0.8$			$\rho_{AB} = 0.4$		
		B ₁	A ₁ B ₂	B ₃	B ₁	A ₂ B ₂	B ₃	B ₁	A ₃ B ₂	B ₃
A ₁	B ₁	1.0	0.8	0.8	0.4	0.4	0.4	0.4	0.4	0.4
	B ₂		1.0	0.8	0.4	0.4	0.4	0.4	0.4	0.4
	B ₃			1.0	0.4	0.4	0.4	0.4	0.4	0.4
A ₂	B ₁				1.0	0.8	0.8	0.4	0.4	0.4
	B ₂					1.0	0.8	0.4	0.4	0.4
	B ₃						1.0	0.4	0.4	0.4
A ₃	B ₁							1.0	0.8	0.8
	B ₂								1.0	0.8
	B ₃									1.0

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.

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: $\sigma_e^2 = \sigma^2(1 - \bar{\rho}_A) + \sigma^2(q - 1)(\bar{\rho}_B - \bar{\rho}_{AB})$

For the main effect of B: $\sigma_e^2 = \sigma^2(1 - \bar{\rho}_B) + \sigma^2(p - 1)(\bar{\rho}_A - \bar{\rho}_{AB})$

For the interaction between A and B: $\sigma_e^2 = \sigma^2(1 - \rho_{\max}) - \sigma^2(\bar{\rho}_{\min} - \bar{\rho}_{AB})$

Chapter 11

Simple example: 2x2 within design

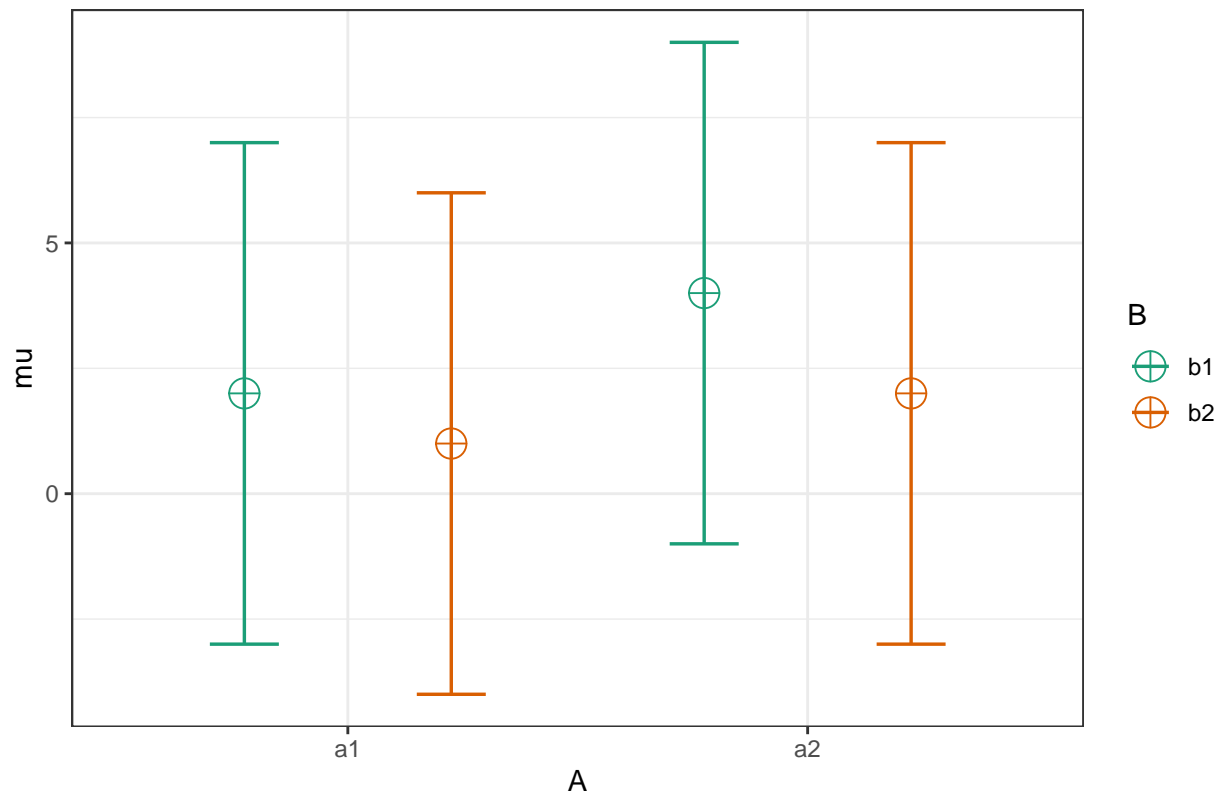
It is difficult to just come up with a positive definite covariance matrix. The best way to achieve this is to get the correlations from a pilot study. Indeed, it should be rather difficult to know which correlations to fill in without some pilot data.

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, 0.4, 0.4,
    0.4, 0.4,
      0.8
)
string = "2w*2w"
alpha_level <- 0.05

labelnames = c("A", "a1", "a2", "B", "b1", "b2")
design_result <- ANOVA_design(design = string,
  n = n,
  mu = mu,
  sd = sd,
  r = r,
  labelnames = labelnames)
```

Means for each condition in the design



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

```
## Power and Effect sizes for ANOVA tests
```

```
##           power effect_size
## anova_A    25.2    0.12701
## anova_B    80.9    0.33263
## anova_A:B   15.7    0.09032
```

```
##
```

```
## Power and Effect sizes for contrasts
```

```
##           power effect_size
## p_A_a1_B_b1_A_a1_B_b2  28.8   -0.337604
## p_A_a1_B_b1_A_a2_B_b1  34.6    0.384171
## p_A_a1_B_b1_A_a2_B_b2   6.1    0.003857
## p_A_a1_B_b2_A_a2_B_b1  65.2    0.577616
## p_A_a1_B_b2_A_a2_B_b2  13.2    0.197427
## p_A_a2_B_b1_A_a2_B_b2  75.9   -0.652759
```

```
##
```

```
## Within-Subject Factors Included: Check MANOVA Results
```

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
```

```
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
```

```
# Potvin & 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-value
                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-value
               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-value
               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
```

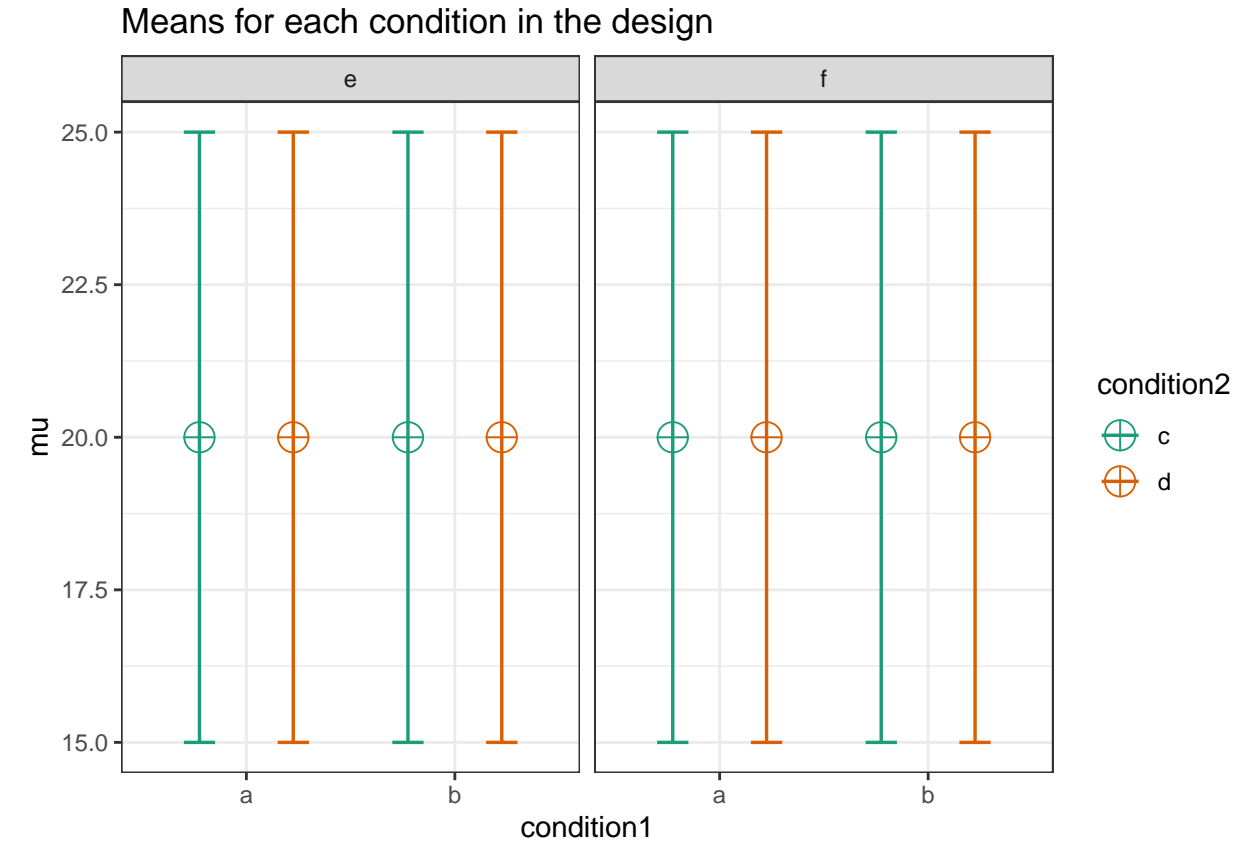
Maybe the simulation is not correct for the interaction, or the formula is not correctly programmed.

Chapter 12

Error Control in Exploratory ANOVA

In a 2X2X2 design, an ANOVA will give the test results for three main effects, three two-way interactions, and one three-way interaction. That's 7 statistical tests. The probability of making at least one Type 1 error in a single 2x2x2 ANOVA is $1-(0.95)^7 = 30\%$.

```
string <- "2b*2b*2b"
n <- 50
mu <- c(20, 20, 20, 20, 20, 20, 20, 20, 20) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 5
p_adjust = "none"
# "none" means we do not correct for multiple comparisons
labelnames <- c("condition1", "a", "b", "condition2", "c", "d", "condition3", "e", "f") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```



```
alpha_level <- 0.05
#We set the alpha level at 0.05.
power_result <- ANOVA_power(design_result,
                             alpha_level = alpha_level,
                             p_adjust = p_adjust,
                             nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
```

```
##
## anova_condition1          power effect_size
## anova_condition2          6    0.002890
## anova_condition3          4    0.002388
## anova_condition1:condition2 5    0.002922
## anova_condition1:condition3 8    0.003331
## anova_condition2:condition3 5    0.002616
## anova_condition1:condition2:condition3 4    0.002362
##
```

```
## Power and Effect sizes for contrasts
```

```
##
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_c_condition3_f 7
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_e 5
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_f 9
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_e 9
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f 9
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e 6
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f 7
```

```

## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_e      4
## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_f      2
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_e      5
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_f      2
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e      3
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f      8
## p_condition1_a_condition2_d_condition3_e_condition1_a_condition2_d_condition3_f      3
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_e      5
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_f      3
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_e      5
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f      5
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_e      4
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_f      5
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_e      4
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_f      3
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f      6
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e      5
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f      3
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e      5
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f      6
## p_condition1_b_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f      6
##                                                                                      effect_size
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_c_condition3_f    -0.033968
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_e    -0.028220
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_f    -0.029225
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_e    -0.020848
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f    -0.005659
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e    -0.022539
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f    -0.021272
## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_e      0.002424
## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_f      0.001919
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_e      0.010053
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_f      0.024860
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e      0.010224
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f      0.010262
## p_condition1_a_condition2_d_condition3_e_condition1_a_condition2_d_condition3_f      0.001681
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_e      0.008714
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_f      0.021857
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_e      0.007698
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f      0.008300
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_e      0.009875
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_f      0.022620
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_e      0.003900
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_f      0.005537
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f      0.012153
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e     -0.003082
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f     -0.003265
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e     -0.015912
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f     -0.014475
## p_condition1_b_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f      0.000289

```

When there is no true effect, we formally do not have ‘power’ (which is defined as the probability of finding $p < \alpha$ if there is a true effect to be found) so the power column should be read as the ‘Type 1 error rate’. Because we have saved the power simulation in the ‘power_result’ object, we can perform calculations on

the ‘sim_data’ dataframe that is stored. This dataframe contains the results for the nsims simulations (e.g., 10000 rows if you ran 10000 simulations) and stores the p-values and effect size estimates for each ANOVA. The first 7 columns are the p-values for the ANOVA, first the main effects of condition 1, 2, and 3, then three two-way interactions, and finally the threeway interaction.

We can calculate the number of significant results for each test (which should be 5%) by counting the number of significant p-values in each of the 7 rows:

```
apply(as.matrix(power_result$sim_data[(1:7)]), 2,
      function(x) round(mean(ifelse(x < alpha_level, 1, 0) * 100),4))
```

```
##          anova_condition1
##                      7
##          anova_condition2
##                      6
##          anova_condition3
##                      4
##    anova_condition1:condition2
##                      5
##    anova_condition1:condition3
##                      8
##    anova_condition2:condition3
##                      5
## anova_condition1:condition2:condition3
##                      4
```

This is the Type 1 error rate for each test. When we talk about error rate inflation due to multiple comparisons, we are talking about the probability that you conclude there is an effect, when there is actually no effect, when there is a significant effect for the main effect of condition 1, or condition 2, or condition 3, or for the two-way interaction between condition 1 and 2, or condition 1 and 3, or condition 2 and 3, or in the threeway interaction.

To calculate this error rate we do not just add the 7 error rates (so $7 * 5\% = 35\%$). Instead, we calculate the probability that there will be at least one significant result in an ANOVA we perform. Some ANOVA results will have multiple significant results, just due to the Type 1 error rate (e.g., a significant result for the threeway interaction, and for the main effect of condition 1) but such an ANOVA is counted only once. Iwe calculate this percentage from our simulations, we see the number is indeed very close to $1 - (0.95)^7 = 30\%$.

```
sum(apply(as.matrix(power_result$sim_data[(1:7)]), 1,
          function(x) round(mean(ifelse(x < alpha_level, 1, 0) * 100),4)) > 0)/nsims*100
```

```
## [1] 33
```

The question is what we should do about this alpha inflation. It is undesirable if you perform exploratory ANOVA’s and are fooled too often by Type 1 errors, which will not replicate if you try to build on them. Therefore, you need to control the Type 1 error rate.

In the simulation code, which relies on the afex package, there is the option to set p_adjust. In the simulation above, p_adjust was set to “none”. This means no adjustment is made to which p-values are considered to be significant, and the alpha level is used as it is set in the simulation (above this was 0.05).

Afex relies on the p.adjust function in the stats package in R (more information is available here). From the package details:

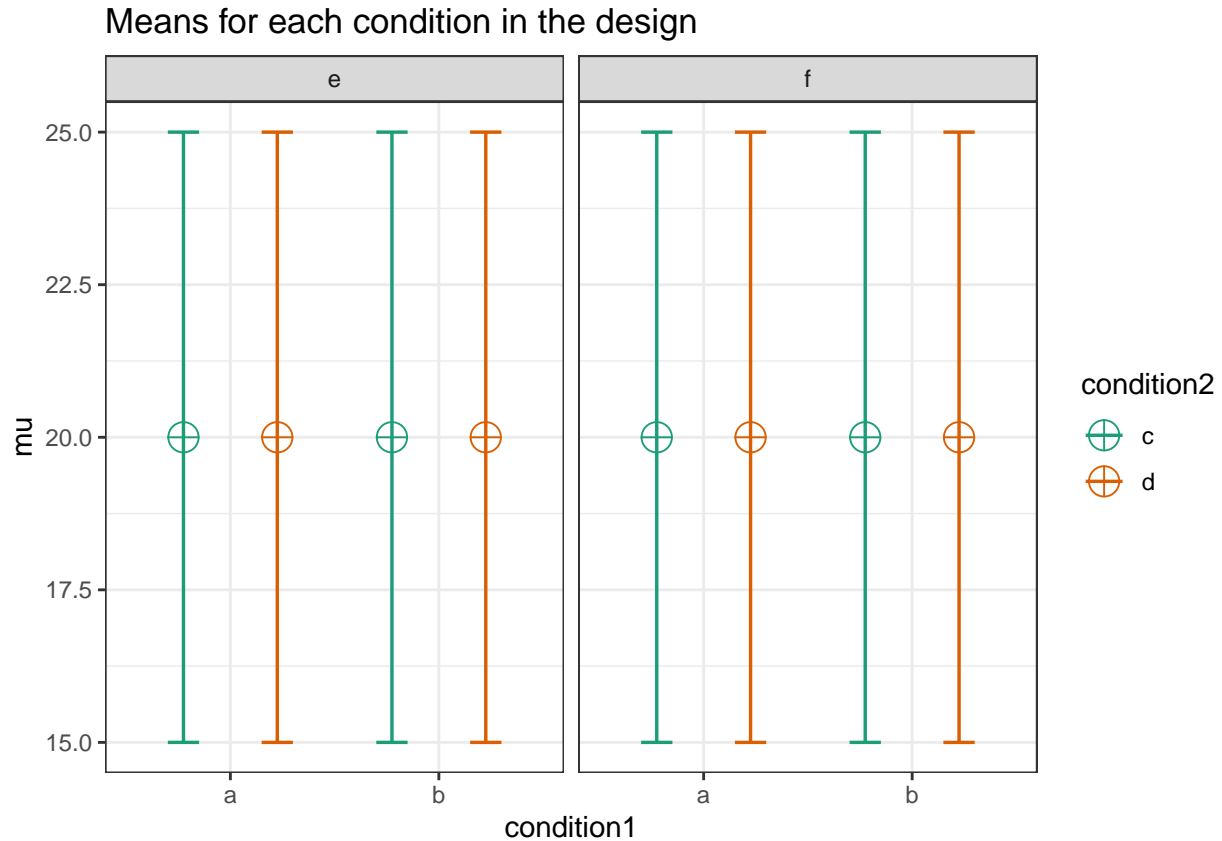
The adjustment methods include the Bonferroni correction (“bonferroni”) in which the p -values are multiplied by the number of comparisons. Less conservative corrections are also included by Holm (1979) (“holm”), Hochberg (1988) (“hochberg”), Hommel (1988) (“hommel”), Benjamini & Hochberg (1995) (“BH” or its alias “fdr”), and Benjamini & Yekutieli (2001) (“BY”), respectively. A pass-through option (“none”) is also included. The first four methods are designed to give strong control of the family-wise error rate. There seems no reason to use the unmodified Bonferroni correction because it is dominated by Holm’s method, which is also valid under arbitrary assumptions.

Hochberg’s and Hommel’s methods are valid when the hypothesis tests are independent or when they are non-negatively associated (Sarkar, 1998; Sarkar and Chang, 1997). Hommel’s method is more powerful than Hochberg’s, but the difference is usually small and the Hochberg p -values are faster to compute.

The “BH” (aka “fdr”) and “BY” method of Benjamini, Hochberg, and Yekutieli control the false discovery rate, the expected proportion of false discoveries amongst the rejected hypotheses. The false discovery rate is a less stringent condition than the family-wise error rate, so these methods are more powerful than the others.

Let’s re-run the simulation with the Holm-Bonferroni correction, which is simple and require no assumptions.

```
string <- "2b*2b*2b"
n <- 50
mu <- c(20, 20, 20, 20, 20, 20, 20, 20, 20) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 5
p_adjust = "holm"
# Changed to Holm-Bonferroni
labelnames <- c("condition1", "a", "b", "condition2", "c", "d", "condition3", "e", "f") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```



```
alpha_level <- 0.05
#We set the alpha level at 0.05.
power_result <- ANOVA_power(design_result,
                             alpha_level = alpha_level,
                             p_adjust = p_adjust,
                             nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
```

```
##
##          power effect_size
## anova_condition1      0  0.002118
## anova_condition2      0  0.002285
## anova_condition3      0  0.002092
## anova_condition1:condition2    1  0.002154
## anova_condition1:condition3    0  0.002392
## anova_condition2:condition3    1  0.002676
## anova_condition1:condition2:condition3  0  0.002047
##
```

```
## Power and Effect sizes for contrasts
```

```
##
##                                     power
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_c_condition3_f    0
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_e    0
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_f    0
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_e    0
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f    0
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e    0
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f    0
```



```

## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_e 0
## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_f 0
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_e 0
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_f 0
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e 0
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f 1
## p_condition1_a_condition2_d_condition3_e_condition1_a_condition2_d_condition3_f 0
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_e 0
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_f 0
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_e 0
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f 0
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_e 0
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_f 0
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_e 0
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_f 0
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f 1
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e 0
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f 0
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e 0
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f 1
## p_condition1_b_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f 0
##
## effect_size
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_c_condition3_f 0.0154433
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_e 0.0205541
## p_condition1_a_condition2_c_condition3_e_condition1_a_condition2_d_condition3_f 0.0422268
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_e 0.0035021
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f 0.0044807
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e 0.0138438
## p_condition1_a_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f -0.0094933
## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_e 0.0053333
## p_condition1_a_condition2_c_condition3_f_condition1_a_condition2_d_condition3_f 0.0265343
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_e -0.0091160
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_c_condition3_f -0.0120365
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e -0.0026837
## p_condition1_a_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f -0.0230180
## p_condition1_a_condition2_d_condition3_e_condition1_a_condition2_d_condition3_f 0.0204428
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_e -0.0169158
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_c_condition3_f -0.0190223
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_e -0.0070959
## p_condition1_a_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f -0.0292813
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_e -0.0365485
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_c_condition3_f -0.0382325
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_e -0.0288324
## p_condition1_a_condition2_d_condition3_f_condition1_b_condition2_d_condition3_f -0.0504461
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_c_condition3_f 0.0007649
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_e 0.0091853
## p_condition1_b_condition2_c_condition3_e_condition1_b_condition2_d_condition3_f -0.0144361
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_e 0.0104412
## p_condition1_b_condition2_c_condition3_f_condition1_b_condition2_d_condition3_f -0.0120434
## p_condition1_b_condition2_d_condition3_e_condition1_b_condition2_d_condition3_f -0.0225975

```

If we now calculate the overall Type 1 error rate:

```
sum(apply(as.matrix(power_result$sim_data[(1:7)]), 1,  
  function(x) round(mean(ifelse(x < alpha_level, 1, 0) * 100),4)) > 0)/nsims*100
```

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

We see it is close to 5%. Note that error rates have variation, and even in a few thousand simulations, the error rate in the sample of studies can easily be half a percentage point higher or lower. But *in the long run* the error rate should equal the alpha level. Furthermore, note that the Holm-bonferroni method is slightly more powerful than the Bonferroni procedure (which is simply α divided by the number of tests). There are more powerful procedures to control the Type 1 error rate, which require more assumptions. For a small number of tests, the Holm-Bonferroni procedure works well. Alternative procedure to control error rates can be found in the multcomp R package.

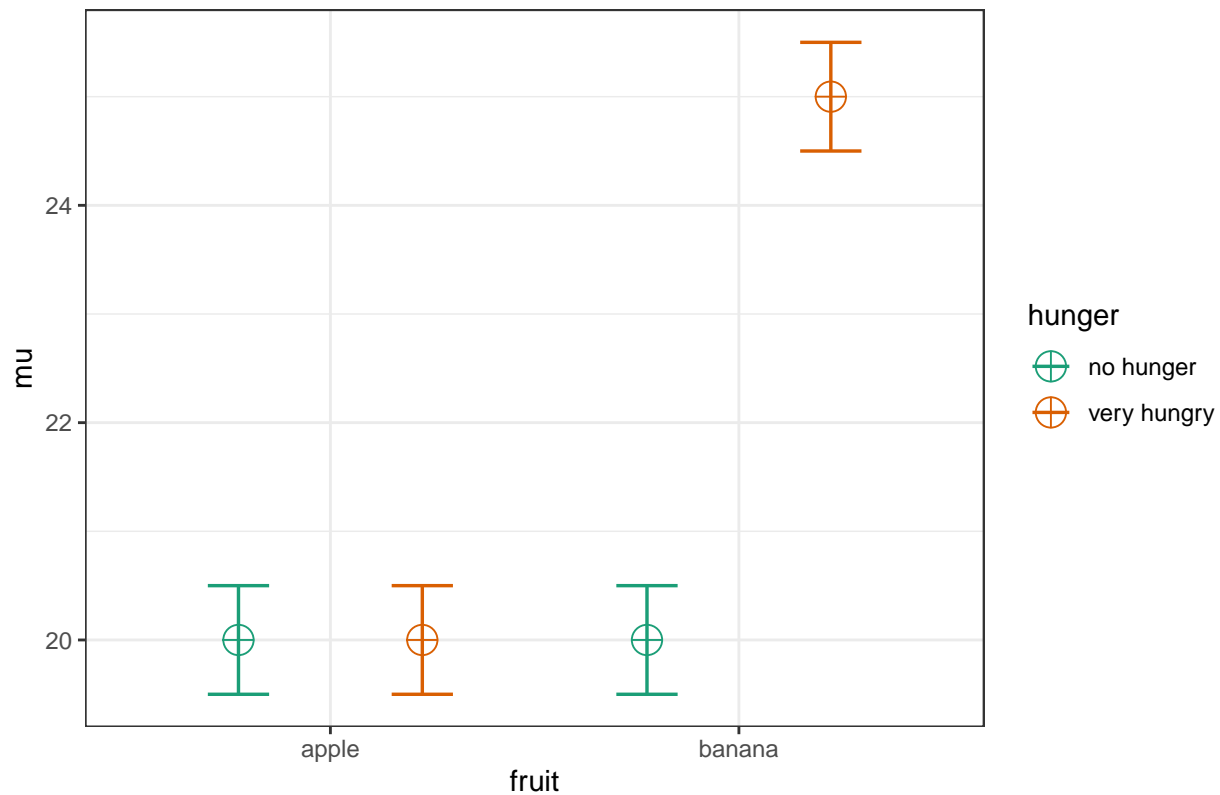
Chapter 13

Power in Interactions

In the 17th Data Colada blog post titled No-way Interactions Uri Simonsohn discusses how a moderated interaction (the effect is there in one condition, but disappears in another condition) requires at least twice as many subjects per cell as a study that simply aims to show the simple effect. For example, see the plot below. Assume the score on the vertical axis is desire for fruit, as a function of the fruit that is available (an apple or a banana) and how hungry people are (not, or very). We see there is a difference between the participants desire for a banana compared to an apple, but only for participants who are very hungry. The point that is made is that you need twice as many participants in each cell to have power for the interaction, as you need for the simple effect.

```
string <- "2b*2b"
n <- 20
mu <- c(20, 20, 20, 25) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 0.5
labelnames <- c("fruit", "apple", "banana", "hunger", "no hunger", "very hungry") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```

Means for each condition in the design



We can reproduce the simulations in the Data Colada blog post, using the original code.

```
#R-Code
#
#Written by Uri Simonsohn, March 2014
#
#
#In DataColada[17] I propose that 2x2 interaction studies need 2x the sample size
#http://datacolada.org/2014/03/10/17-no-way-interactions
#In a companion ,pdf I show the simple math behind it
#
#
#Simulations are often more persuasive than math, so here it goes.
#I run simulations that compute power for 2 and 4 cell design, the latter testing the interaction
#####
#Create function that computes power of Studies 1 and 2, where Study 1 has 2 cells and tests a simple
#and Study 2 has 4 cells and tests the interaction
colada17=function(d1,d2,n1,n2,simtot)
{
  #n1: sample size, per cell, study 1
  #n2: sample size, per cell, study 2
  #d1: simple effect M1-M2
  #d2: moderated effect M3-M4, full elimination of effect implies d2=0
  #simtot: how many simulations to run
  #Here we will store results
  p1=c()    #p-values for Study 1
```

```

    p2=c()      #p-values for Study 2
for(i in 1:simt) {
  #draw data 4 samples
  y1=rnorm(n=max(n1,n2),mean=d1)
  y2=rnorm(n=max(n1,n2))
  y3=rnorm(n=max(n1,n2),mean=d2)
  y4=rnorm(n=max(n1,n2))

  #GET DATA READY FOR ANOVA
  y=c(y1,y2,y3,y4)      #the d.v.
  nrep=rep(n2,4)
  A=rep(c(1,1,0,0),times=nrep)
  B=rep(c(1,0,1,0),times=nrep)

  #STUDY 1
  p1.k=t.test(y1[1:n1],y2[1:n1],var.equal=TRUE)$p.value  #Do a t-test on the first n1 observations

  #STUDY 2
  p2.k=anova(lm(y ~ A * B))["A:B", "Pr(>F)"]              #Do anova, keep p-value of the interaction

  #Store the results
  p1=c(p1,p1.k)
  p2=c(p2,p2.k)

}

#What share off comparisons are significant
power1=sum(p1<=.05)/simt      #Simple test using estimate of variance from 2 cells only
power2=sum(p2<=.05)/simt      #Interaction

cat("\nStudy 1 is powered to:",round(power1,2))
cat("\nStudy 2 is powered to:",round(power2,2))

}

#Same power for 2n regardless of n and d
colada17(simt=2000, n1=20,n2=40,d1=1,d2=0)

##
## Study 1 is powered to: 0.87
## Study 2 is powered to: 0.9

colada17(simt=2000, n1=50,n2=100,d1=.3,d2=0)

##
## Study 1 is powered to: 0.33
## Study 2 is powered to: 0.3

colada17(simt=2000, n1=150,n2=300,d1=.25,d2=0)

##

```

```
## Study 1 is powered to: 0.58
## Study 2 is powered to: 0.59
```

```
#Need 4n if effect is 70% attenuated
colada17(simtot=2000, n1=25,n2=100,d1=.5, d2=.3*.5)
```

```
##
## Study 1 is powered to: 0.39
## Study 2 is powered to: 0.41
```

```
colada17(simtot=2000, n1=50,n2=200,d1=.5, d2=.3*.5)
```

```
##
## Study 1 is powered to: 0.7
## Study 2 is powered to: 0.68
```

```
colada17(simtot=2000, n1=22,n2=88,d1=.41, d2=.3*.41)
```

```
##
## Study 1 is powered to: 0.26
## Study 2 is powered to: 0.27
```

```
#underpowered if run with the same n
colada17(simtot=nsims, n1=20,n2=20,d1=1,d2=0)
```

```
##
## Study 1 is powered to: 0.87
## Study 2 is powered to: 0.53
```

And we can reproduce the results using the ANOVA_power function.

```
alpha_level <- 0.05 #We set the alpha level at 0.05.
power_result <- ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
```

```
##           power effect_size
## anova_fruit      100      0.8631
## anova_hunger      100      0.8637
## anova_fruit:hunger 100      0.8622
##
```

```
## Power and Effect sizes for contrasts
```

```
##                                     power
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry      5
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger        3
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry     100
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger      5
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry    100
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry     100
##                                     effect_size
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry      0.033070
```

```
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger      0.023914
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry    10.036278
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger    -0.009534
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry   9.872160
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry    10.099645
```

We see we get the same power for the `anova_fruit:hunger` interaction and for the simple effect `p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry` as the simulations by Uri Simonsohn in his blog post.

```
#Same power for 2n regardless of n and d
colada17(simtot = 10000, n1 = 20, n2 = 40, d1 = 1, d2 = 0)
```

```
##
## Study 1 is powered to: 0.87
## Study 2 is powered to: 0.88
```

```
colada17(simtot = 10000, n1 = 50, n2 = 100, d1 = .3, d2 = 0)
```

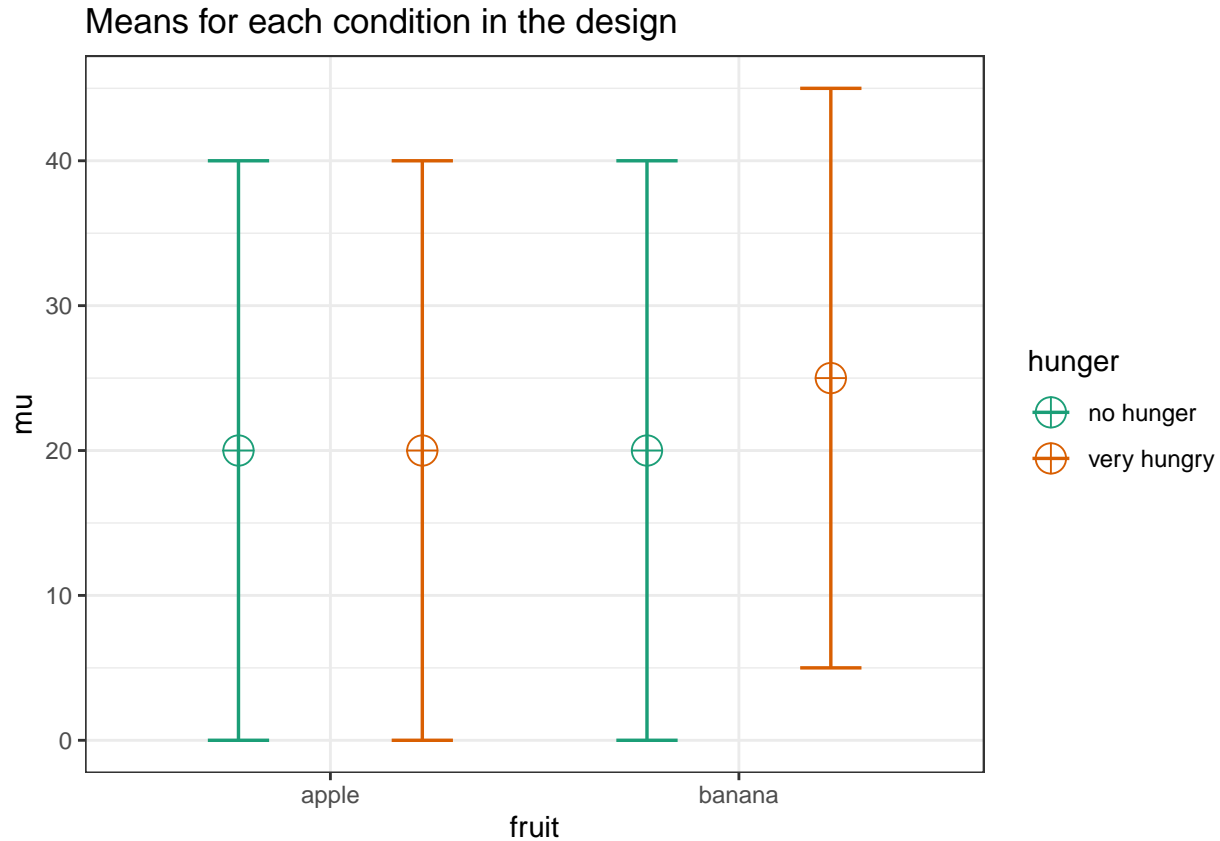
```
##
## Study 1 is powered to: 0.32
## Study 2 is powered to: 0.31
```

```
colada17(simtot = 10000, n1 = 150, n2 = 300, d1 = .25, d2 = 0)
```

```
##
## Study 1 is powered to: 0.58
## Study 2 is powered to: 0.59
```

We can also reproduce the last example by adjusting the means and standard deviation. With 150 people, and a Cohen's *d* of 0.25 (the difference is 5, the sd 20, so $5/20 = 0.25$) we should reproduce the power for the simple effect.

```
string <- "2b*2b"
n <- 150
mu <- c(20, 20, 20, 25) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 20
labelnames <- c("fruit", "apple", "banana", "hunger", "no hunger", "very hungry") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```



```
alpha_level <- 0.05 #We set the alpha level at 0.05.
power_result <- ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
```

```
##           power effect_size
## anova_fruit      33    0.005426
## anova_hunger     38    0.006345
## anova_fruit:hunger 30    0.005728
##
```

```
## Power and Effect sizes for contrasts
```

```
##                                     power
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry      5
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger        6
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry     53
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger       5
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry     63
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry     69
##                                     effect_size
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry    0.007274
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger    -0.009399
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry    0.256463
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger    -0.016857
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry    0.249103
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry    0.266276
```

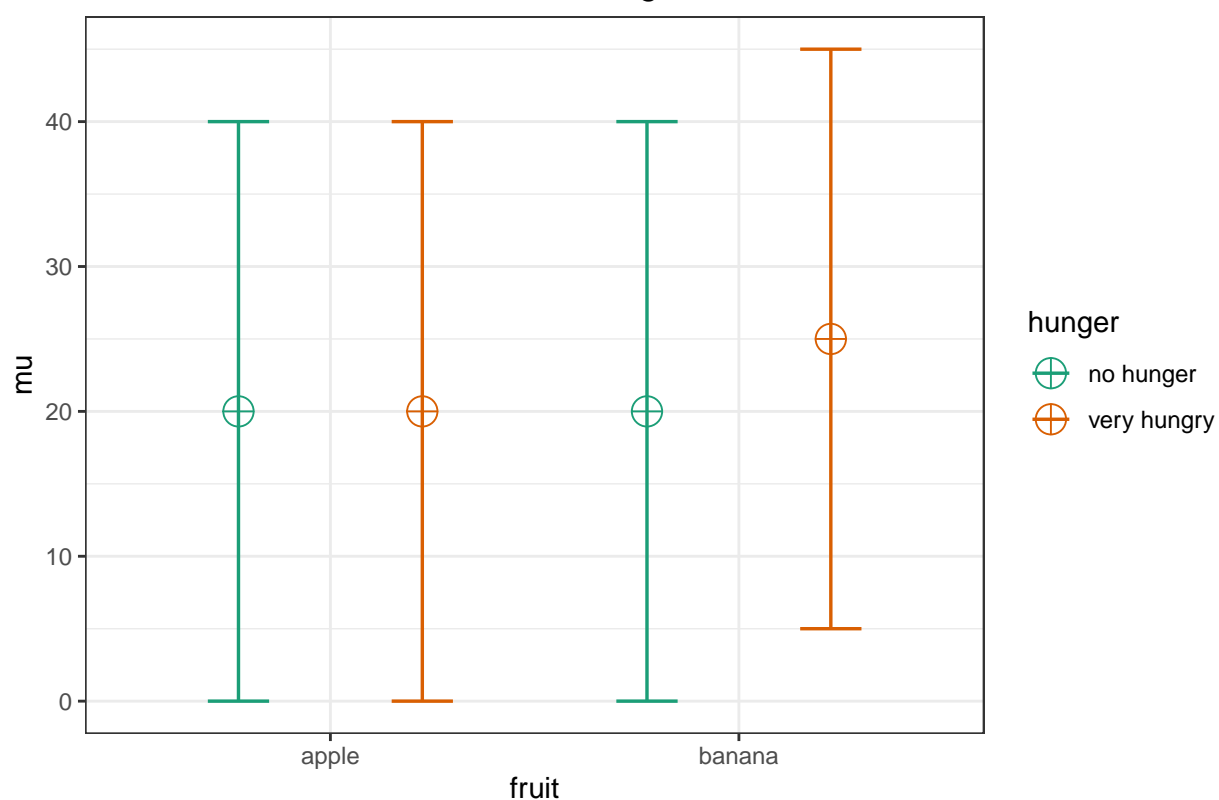
And changing the sample size to 300 should reproduce the power for the interaction in the ANOVA.


```

string <- "2b*2b"
n <- 300
mu <- c(20, 20, 20, 25) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 20
labelnames <- c("fruit", "apple", "banana", "hunger", "no hunger", "very hungry") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)

```

Means for each condition in the design



```

alpha_level <- 0.05 #We set the alpha level at 0.05.
power_result <- ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)

```

```

## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_fruit      63    0.004449
## anova_hunger      64    0.005619
## anova_fruit:hunger 53    0.004759
##

```

```

## Power and Effect sizes for contrasts
##

```

```

##           power
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry      8

```

```
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger      4
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry    86
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger    7
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry   86
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry   89
##                                                                    effect_size
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry    0.010926
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger     -0.002831
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry    0.258490
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger    -0.013783
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry   0.246630
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry   0.261495
```

Now if we look at the power analysis table for the last simulation, we see that the power for the ANOVA is the same for the main effect of fruit, the main effect of hunger, and the main effect of the interaction. All the effect sizes are equal as well. We can understand why if we look at the means in a 2x2 table:

```
mean_mat <- t(matrix(mu,
                     nrow = 2,
                     ncol = 2)) #Create a mean matrix
rownames(mean_mat) <- c("apple", "banana")
colnames(mean_mat) <- c("no hunger", "very hungry")
mean_mat
```

```
##      no hunger very hungry
## apple      20      20
## banana     20      25
```

The first main effect tests the marginal means if we sum over rows, 22.5 vs 20.

```
rowMeans(mean_mat)
```

```
##  apple banana
##  20.0   22.5
```

The second main effect tests the marginal means over the rows, which is also 22.5 vs 20.

```
colMeans(mean_mat)
```

```
##  no hunger very hungry
##    20.0      22.5
```

The interaction tests whether the average effect of hunger on liking fruit differs in the presence of bananas. In the presence of bananas the effect of hunger on the desirability of fruit is 5 scalepoints. The average effect (that we get from the marginal means) of hunger on fruit desirability is 2.5 (22.5-20). In other words, the interaction tests whether the difference effect between hunger and no hunger is different in the presence of an apple versus in the presence of a banana.

Mathematically the interaction effect is computed as the difference between a cell mean and the grand mean, the marginal mean in row i and the grand mean, and the marginal mean in column j and grand mean. For example, for the very hungry-banana condition this is 25 (the value in the cell) - (21.25 [the grand mean])

+ 1.25 [the marginal mean in row 2, 22.5, minus the grand mean of 21.25] + 1.25 [the marginal mean in column 2, 22.5, minus the grand mean of 21.25]). $25 - (21.25 + (22.5 - 21.25) + (22.5 - 21.25)) = 1.25$.

We can repeat this for every cell, and get for no hunger-apple: $20 - (21.25 + (20 - 21.25) + (20 - 21.25)) = 1.25$, for very hungry apple: $20 - (21.25 + (22.5 - 21.25) + (20 - 21.25)) = 1.25$, and no hunger-banana: $20 - (21.25 + (20 - 21.25) + (22.5 - 21.25)) = 1.25$. These values are used to calculate the sum of squares.

```
a1 <- mean_mat[1,1] - (mean(mean_mat) + (mean(mean_mat[1,]) - mean(mean_mat)) + (mean(mean_mat[,1]) - mean(mean_mat)))
a2 <- mean_mat[1,2] - (mean(mean_mat) + (mean(mean_mat[1,]) - mean(mean_mat)) + (mean(mean_mat[,2]) - mean(mean_mat)))
b1 <- mean_mat[2,1] - (mean(mean_mat) + (mean(mean_mat[2,]) - mean(mean_mat)) + (mean(mean_mat[,1]) - mean(mean_mat)))
b2 <- mean_mat[2,2] - (mean(mean_mat) + (mean(mean_mat[2,]) - mean(mean_mat)) + (mean(mean_mat[,2]) - mean(mean_mat)))
SS_ab <- n * sum(c(a1, a2, b1, b2)^2)
```

The sum of squares is dependent on the sample size, as can be seen in the code above. The larger the sample size, the larger the sum of squares, and therefore (all else equal) the larger the F -statistic, and the smaller the p -value. We see from the simulations that all three tests have the same effect size, and therefore the same power.

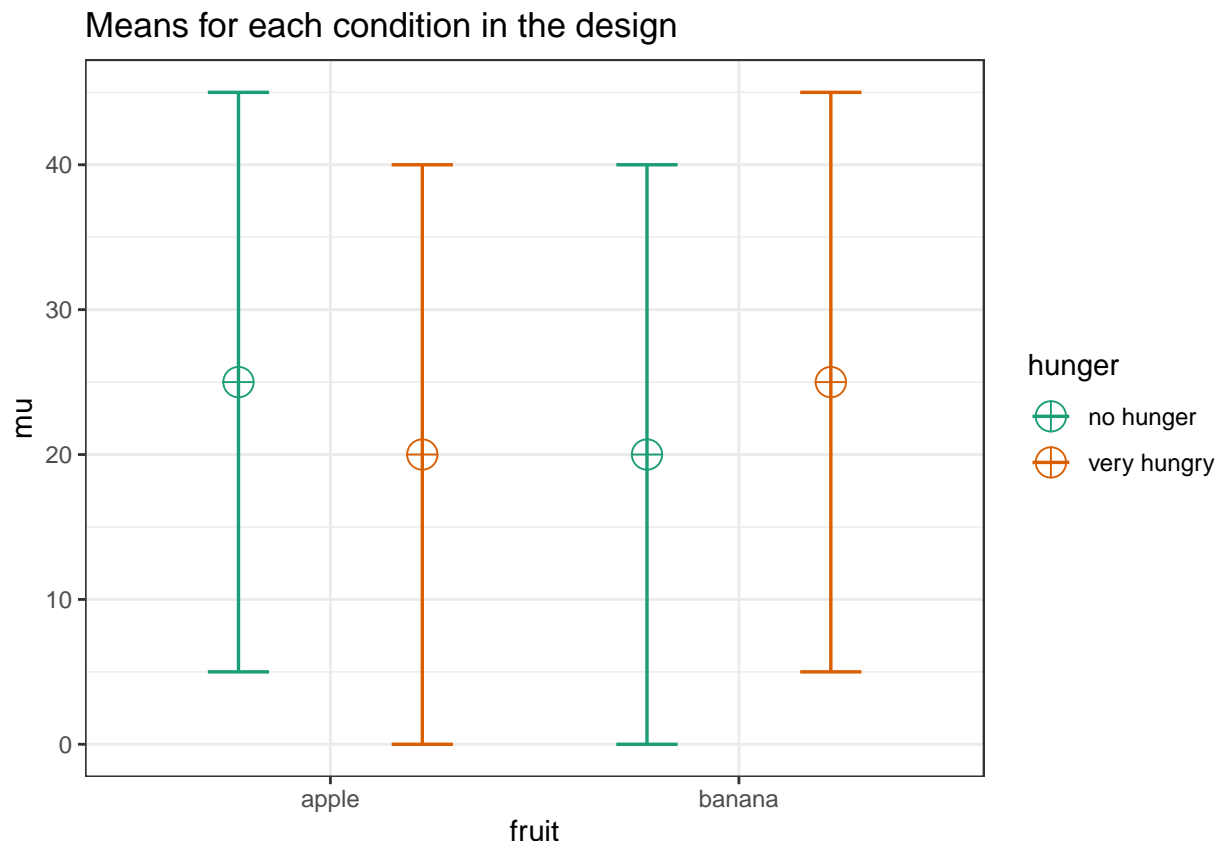
Interactions can have more power than main effects if the effect size of the interaction is larger than the effect size of the main effects. An example of this is a cross-over interaction. For example, let's take a 2x2 matrix of means with a crossover interaction:

```
mu <- c(25, 20, 20, 25)
mean_mat <- t(matrix(mu,
                     nrow = 2,
                     ncol = 2)) #Create a mean matrix
rownames(mean_mat) <- c("apple", "banana")
colnames(mean_mat) <- c("no hunger", "very hungry")
mean_mat
```

```
##      no hunger very hungry
## apple      25         20
## banana     20         25
```

Neither of the main effects is now significant, as the marginal means are 22.5 vs 22.5 for both main effects. The interaction is much stronger, however. We are testing whether the average effect of hunger on the desirability of fruit is different in the presence of bananas. Since the average effect is 0, and the effect of hunger on the desirability of bananas is 5, so the effect size is now twice as large.

```
string <- "2b*2b"
n <- 300
mu <- c(25, 20, 20, 25) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 20
labelnames <- c("fruit", "apple", "banana", "hunger", "no hunger", "very hungry") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```



```
alpha_level <- 0.05 #We set the alpha level at 0.05.
power_result <- ANOVA_power(design_result, alpha_level = alpha_level, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_fruit      5  0.0009895
## anova_hunger      6  0.0006749
## anova_fruit:hunger 99  0.0166921
##
## Power and Effect sizes for contrasts
##
##           power
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry 90
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger 87
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry 8
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger 5
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry 84
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry 90
##           effect_size
## p_fruit_apple_hunger_no hunger_fruit_apple_hunger_very hungry -0.2516739
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_no hunger -0.2552283
## p_fruit_apple_hunger_no hunger_fruit_banana_hunger_very hungry 0.0008431
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_no hunger -0.0041340
## p_fruit_apple_hunger_very hungry_fruit_banana_hunger_very hungry 0.2532954
## p_fruit_banana_hunger_no hunger_fruit_banana_hunger_very hungry 0.2570198
```

We can also reproduce the power analysis using the analytic function:

```
power_analytic <- power_twoway_between(design_result)
power_analytic$power_A
```

```
## [1] 0.05
```

```
power_analytic$power_B
```

```
## [1] 0.05
```


Chapter 14

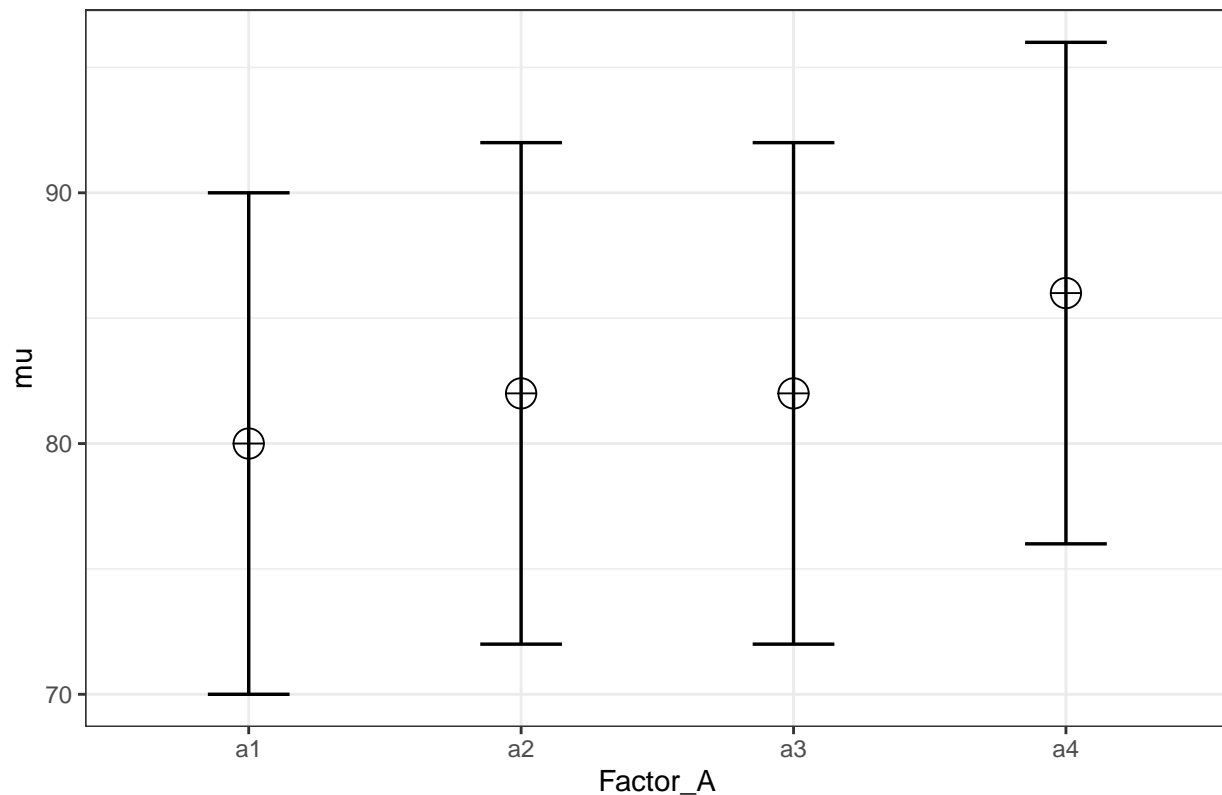
Analytic Power Functions

For some designs it is possible to calculate power analytically, using closed functions.

14.1 One-Way Between Subject ANOVA

```
string <- "4b"
n <- 60
mu <- c(80, 82, 82, 86) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 10
labelnames <- c("Factor_A", "a1", "a2", "a3", "a4") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```

Means for each condition in the design



```
power_result <- ANOVA_power(design_result, alpha_level = 0.05, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_Factor_A    84    0.05683
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_Factor_A_a1_Factor_A_a2    21    0.218427
## p_Factor_A_a1_Factor_A_a3    20    0.227226
## p_Factor_A_a1_Factor_A_a4    92    0.608481
## p_Factor_A_a2_Factor_A_a3     6    0.007324
## p_Factor_A_a2_Factor_A_a4    52    0.391843
## p_Factor_A_a3_Factor_A_a4    51    0.382210
```

We can also calculate power analytically with our own function.

```
power_oneway_between(design_result)$power #using default alpha level of .05
```

```
## [1] 0.8121291
```

This is a generalized function for One-Way ANOVA's for any number of groups. It is in part based on code provided with the excellent book by Aberson (2019) Applied Power Analysis for the Behavioral Sciences (but Aberson's code allows for different n per condition, and different sd per condition).

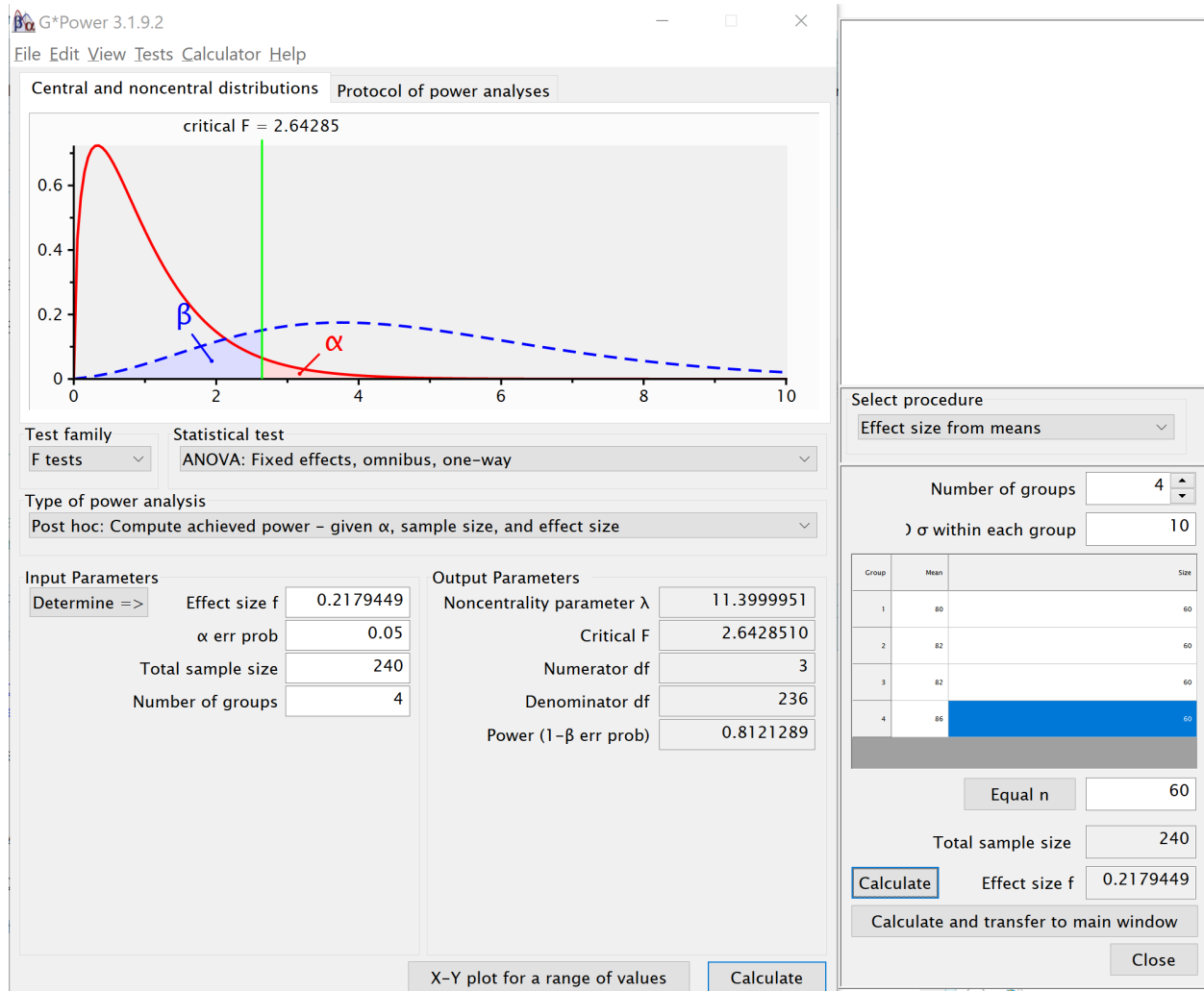

```
anova1f_4(m1 = 80, m2 = 82, m3 = 82, m4 = 86,
          s1 = 10, s2 = 10, s3 = 10, s4 = 10,
          n1 = 60, n2 = 60, n3 = 60, n4 = 60,
          alpha = .05)
```

We can also use the function in the pwr package. Note that we need to calculate f to use this function, which is based on the means and sd, as illustrated in the formulas above.

```
pwr.anova.test(n = 60,
               k = 4,
               f = 0.2179449,
               sig.level = 0.05)
```

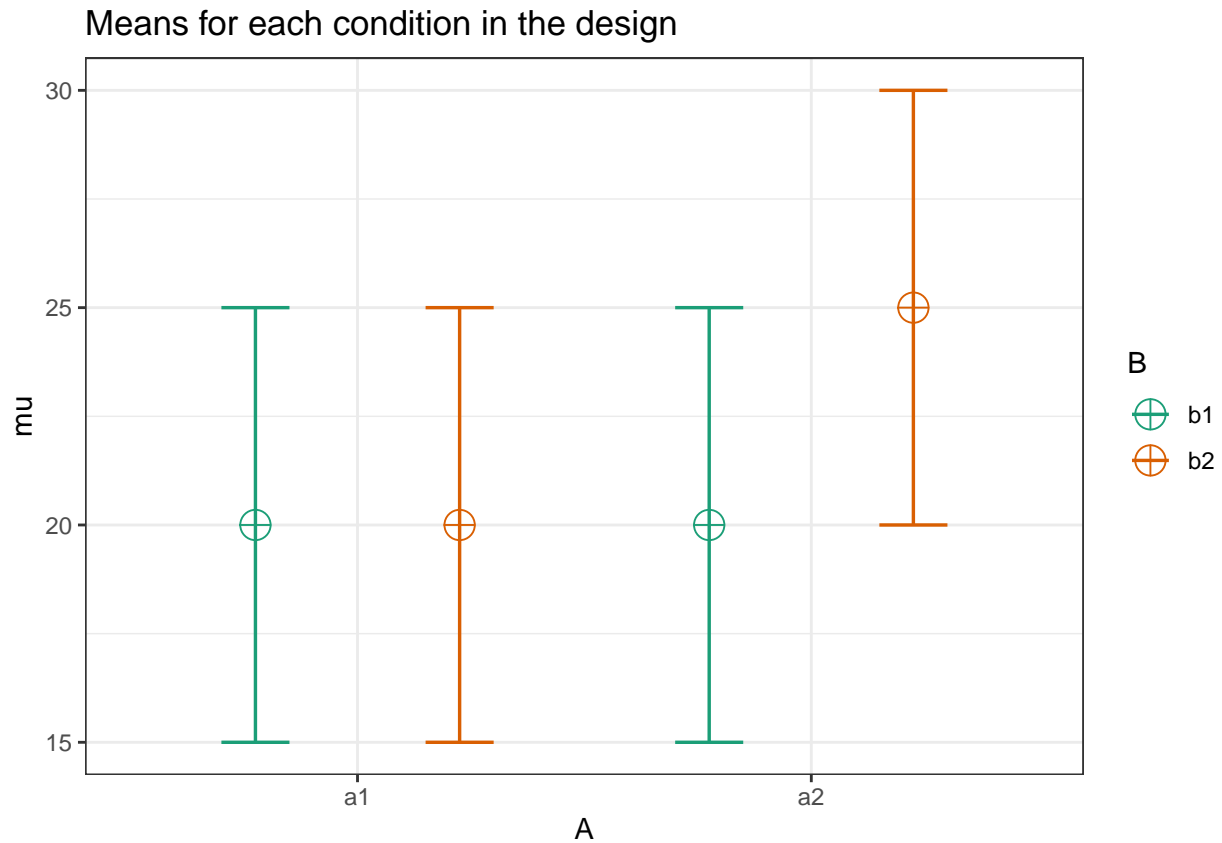
```
##
##      Balanced one-way analysis of variance power calculation
##
##              k = 4
##              n = 60
##              f = 0.2179449
##      sig.level = 0.05
##      power = 0.8121289
##
## NOTE: n is number in each group
```

Finally, G*Power provides the option to calculate f from the means, sd and n for the cells. It can then be used to calculate power.



14.2 Two-way Between Subject Interaction

```
string <- "2b*2b"
n <- 20
mu <- c(20, 20, 20, 25)
# Enter means in the order that matches the labels below.
sd <- 5
labelnames <- c("A", "a1", "a2", "B", "b1", "b2") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```



```
power_result <- ANOVA_power(design_result, alpha_level = 0.05, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
##           power effect_size
## anova_A      65    0.08011
## anova_B      61    0.07314
## anova_A:B     60    0.06963
##
## Power and Effect sizes for contrasts
##           power effect_size
## p_A_a1_B_b1_A_a1_B_b2      6    0.01091
## p_A_a1_B_b1_A_a2_B_b1      4    0.04024
## p_A_a1_B_b1_A_a2_B_b2     88    1.06041
## p_A_a1_B_b2_A_a2_B_b1      4    0.02876
## p_A_a1_B_b2_A_a2_B_b2     88    1.03465
## p_A_a2_B_b1_A_a2_B_b2     87    1.00829
```

```
power_res <- power_twoway_between(design_result) #using default alpha level of .05
power_res$power_A
```

```
## [1] 0.5978655
```

```
power_res$power_B
```

```
## [1] 0.5978655
```

```
power_res$power_AB
```

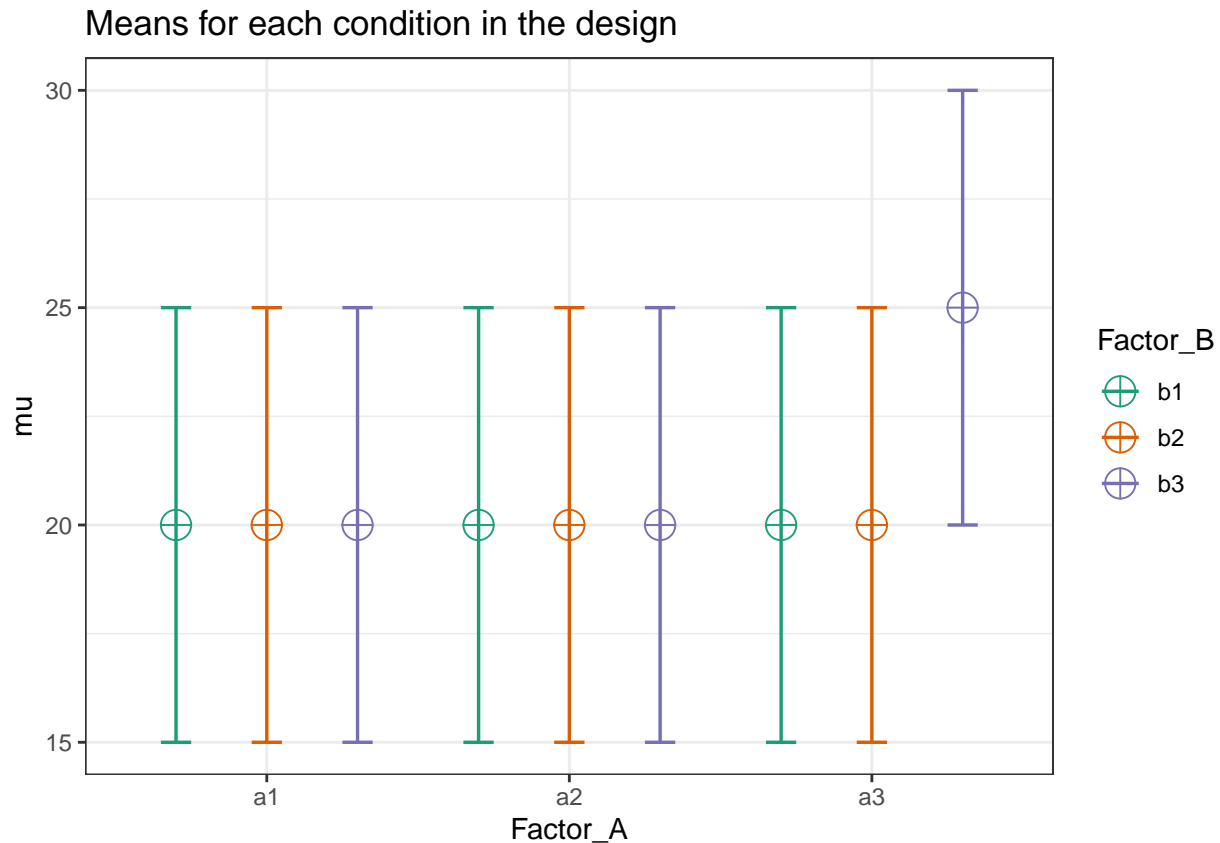
```
## [1] 0.5978655
```

We can use the function by Aberson, 2019, as well.

```
anova2x2(m1.1=20,
          m1.2=20,
          m2.1=20,
          m2.2=25,
          s1.1=5,
          s1.2=5,
          s2.1=5,
          s2.2=5,
          n1.1=20,
          n1.2=20,
          n2.1=20,
          n2.2=20,
          alpha=.05,
          all="OFF")
```

14.3 3x3 Between Subject ANOVA

```
string <- "3b*3b"
n <- 20
mu <- c(20, 20, 20, 20, 20, 20, 20, 20, 25) #All means are equal - so there is no real difference.
# Enter means in the order that matches the labels below.
sd <- 5
labelnames <- c("Factor_A", "a1", "a2", "a3", "Factor_B", "b1", "b2", "b3") #
# the label names should be in the order of the means specified above.
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             labelnames = labelnames)
```



```
power_result <- ANOVA_power(design_result, alpha_level = 0.05, nsims = nsims)
```

```
## Power and Effect sizes for ANOVA tests
```

```
##               power effect_size
## anova_Factor_A      40    0.03563
## anova_Factor_B      43    0.03753
## anova_Factor_A:Factor_B 61    0.06852
##
```

```
## Power and Effect sizes for contrasts
```

```
##               power effect_size
## p_Factor_A_a1_Factor_B_b1_Factor_A_a1_Factor_B_b2      6  -0.0024935
## p_Factor_A_a1_Factor_B_b1_Factor_A_a1_Factor_B_b3      3   0.0484236
## p_Factor_A_a1_Factor_B_b1_Factor_A_a2_Factor_B_b1      5   0.0123203
## p_Factor_A_a1_Factor_B_b1_Factor_A_a2_Factor_B_b2      3   0.0033535
## p_Factor_A_a1_Factor_B_b1_Factor_A_a2_Factor_B_b3      6  -0.0097273
## p_Factor_A_a1_Factor_B_b1_Factor_A_a3_Factor_B_b1      1   0.0183040
## p_Factor_A_a1_Factor_B_b1_Factor_A_a3_Factor_B_b2      5   0.0085122
## p_Factor_A_a1_Factor_B_b1_Factor_A_a3_Factor_B_b3     87   1.0117743
## p_Factor_A_a1_Factor_B_b2_Factor_A_a1_Factor_B_b3      5   0.0501841
## p_Factor_A_a1_Factor_B_b2_Factor_A_a2_Factor_B_b1      7   0.0106398
## p_Factor_A_a1_Factor_B_b2_Factor_A_a2_Factor_B_b2      9   0.0038071
## p_Factor_A_a1_Factor_B_b2_Factor_A_a2_Factor_B_b3      5  -0.0116012
## p_Factor_A_a1_Factor_B_b2_Factor_A_a3_Factor_B_b1      2   0.0195116
## p_Factor_A_a1_Factor_B_b2_Factor_A_a3_Factor_B_b2      5   0.0007596
## p_Factor_A_a1_Factor_B_b2_Factor_A_a3_Factor_B_b3     82   1.0134672
```

```
## p_Factor_A_a1_Factor_B_b3_Factor_A_a2_Factor_B_b1    4 -0.0379532
## p_Factor_A_a1_Factor_B_b3_Factor_A_a2_Factor_B_b2    5 -0.0490203
## p_Factor_A_a1_Factor_B_b3_Factor_A_a2_Factor_B_b3    6 -0.0591611
## p_Factor_A_a1_Factor_B_b3_Factor_A_a3_Factor_B_b1    5 -0.0286974
## p_Factor_A_a1_Factor_B_b3_Factor_A_a3_Factor_B_b2    7 -0.0425004
## p_Factor_A_a1_Factor_B_b3_Factor_A_a3_Factor_B_b3   86  0.9577216
## p_Factor_A_a2_Factor_B_b1_Factor_A_a2_Factor_B_b2    2 -0.0074014
## p_Factor_A_a2_Factor_B_b1_Factor_A_a2_Factor_B_b3    7 -0.0216996
## p_Factor_A_a2_Factor_B_b1_Factor_A_a3_Factor_B_b1    7  0.0105310
## p_Factor_A_a2_Factor_B_b1_Factor_A_a3_Factor_B_b2    6  0.0045423
## p_Factor_A_a2_Factor_B_b1_Factor_A_a3_Factor_B_b3   83  1.0019800
## p_Factor_A_a2_Factor_B_b2_Factor_A_a2_Factor_B_b3    9 -0.0133881
## p_Factor_A_a2_Factor_B_b2_Factor_A_a3_Factor_B_b1    7  0.0229581
## p_Factor_A_a2_Factor_B_b2_Factor_A_a3_Factor_B_b2    4  0.0101205
## p_Factor_A_a2_Factor_B_b2_Factor_A_a3_Factor_B_b3   86  1.0049495
## p_Factor_A_a2_Factor_B_b3_Factor_A_a3_Factor_B_b1    9  0.0375630
## p_Factor_A_a2_Factor_B_b3_Factor_A_a3_Factor_B_b2    4  0.0203393
## p_Factor_A_a2_Factor_B_b3_Factor_A_a3_Factor_B_b3   84  1.0411899
## p_Factor_A_a3_Factor_B_b1_Factor_A_a3_Factor_B_b2    2 -0.0170923
## p_Factor_A_a3_Factor_B_b1_Factor_A_a3_Factor_B_b3   82  0.9962469
## p_Factor_A_a3_Factor_B_b2_Factor_A_a3_Factor_B_b3   81  1.0041388
```

```
power_res <- power_twoway_between(design_result) #using default alpha level of .05
power_res$power_A
```

```
## [1] 0.4486306
```

```
power_res$power_B
```

```
## [1] 0.4486306
```

```
power_res$power_AB
```

```
## [1] 0.6434127
```

14.4 Two by two ANOVA, within design

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

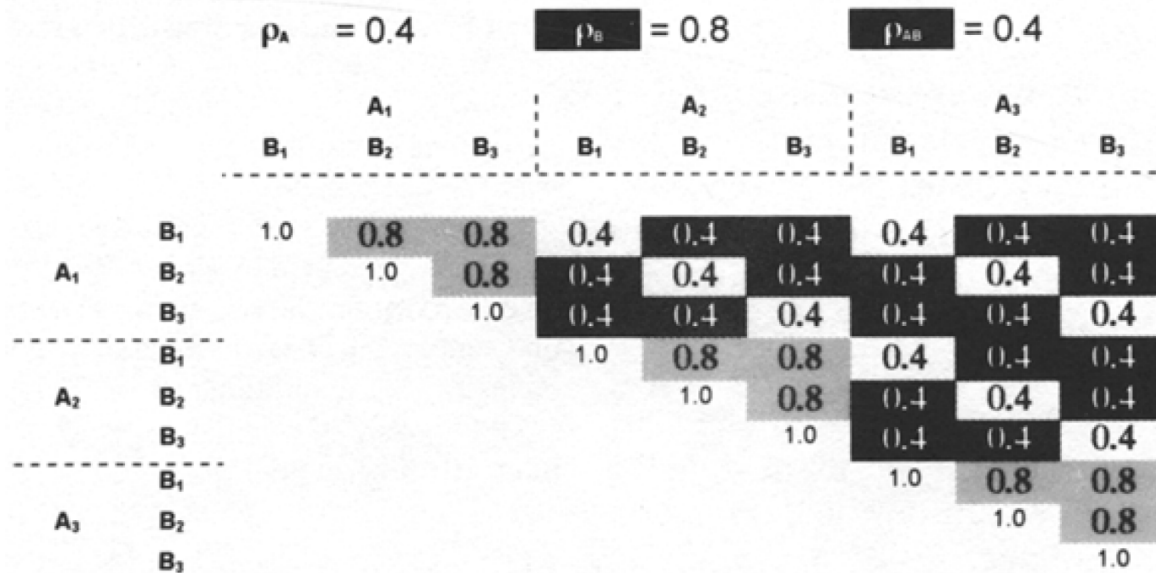
Example

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.

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: $\sigma_e^2 = \sigma^2(1 - \bar{\rho}_A) + \sigma^2(q - 1)(\bar{\rho}_B - \bar{\rho}_{AB})$

For the main effect of B: $\sigma_e^2 = \sigma^2(1 - \bar{\rho}_B) + \sigma^2(p - 1)(\bar{\rho}_A - \bar{\rho}_{AB})$

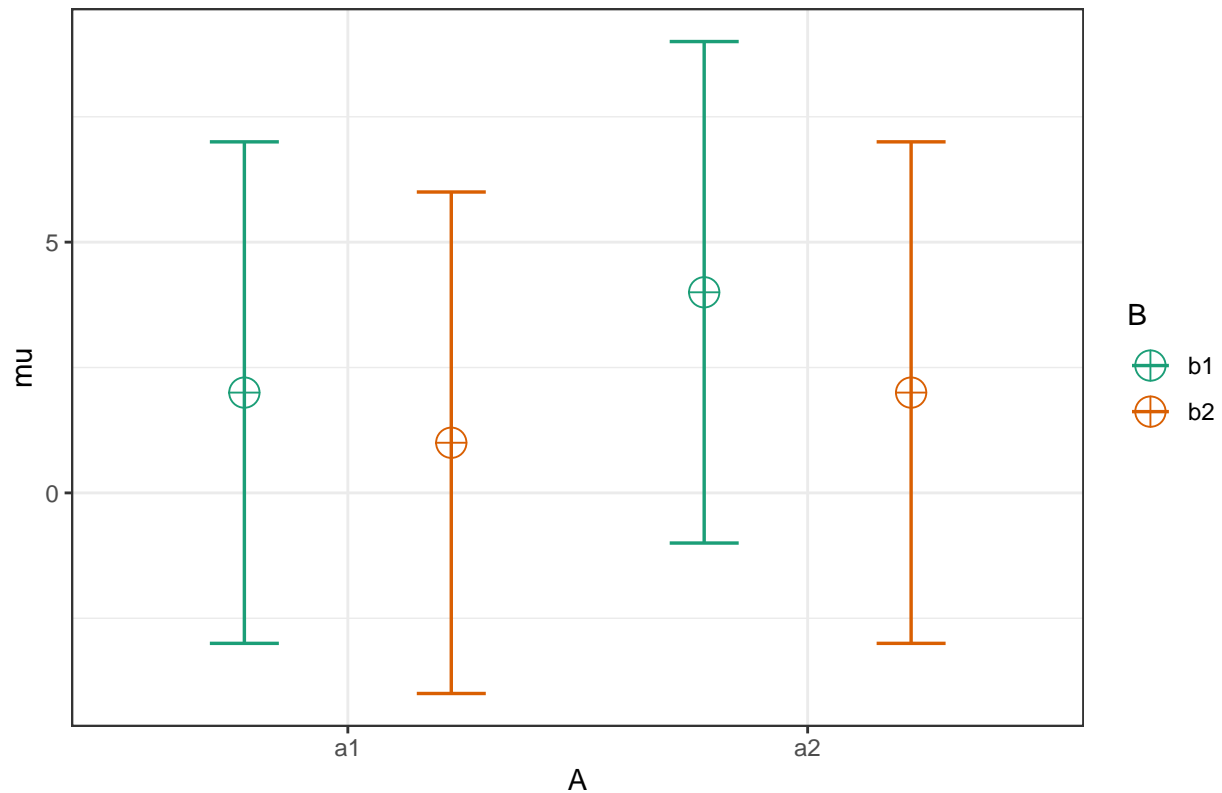
For the interaction between A and B: $\sigma_e^2 = \sigma^2(1 - \rho_{\max}) - \sigma^2(\bar{\rho}_{\min} - \bar{\rho}_{AB})$

We first simulate a within subjects 2x2 ANOVA design.

```
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"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
```

```
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             labelnames = labelnames)
```

Means for each condition in the design



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

```
## Power and Effect sizes for ANOVA tests
```

```
##           power effect_size
## anova_A      31      0.1433
## anova_B      65      0.2631
## anova_A:B     24      0.1165
##
```

```
## Power and Effect sizes for contrasts
```

```
##           power effect_size
## p_A_a1_B_b1_A_a1_B_b2    23    -0.34478
## p_A_a1_B_b1_A_a2_B_b1    42     0.41916
## p_A_a1_B_b1_A_a2_B_b2     7     0.01247
## p_A_a1_B_b2_A_a2_B_b1    65     0.58137
## p_A_a1_B_b2_A_a2_B_b2    15     0.22130
## p_A_a2_B_b1_A_a2_B_b2    80    -0.64982
##
```

```
## Within-Subject Factors Included: Check MANOVA Results
```


We can use the `ANOVA_exact` function to evaluate this design.

```
power_res <- ANOVA_exact(design_result = design_result)
```

```
## Power and Effect sizes for ANOVA tests
##      power partial_eta_squared cohen_f non centrality
## A   26.92                0.0952  0.3244             2
## B   64.23                0.2400  0.5620             6
## A:B 26.92                0.0952  0.3244             2
##
## Power and Effect sizes for contrasts
##              power effect_size
## p_A_a1_B_b1_A_a1_B_b2 26.92    -0.3162
## p_A_a1_B_b1_A_a2_B_b1 39.70     0.4000
## p_A_a1_B_b1_A_a2_B_b2  5.00     0.0000
## p_A_a1_B_b2_A_a2_B_b1 64.23     0.5477
## p_A_a1_B_b2_A_a2_B_b2 13.60     0.2000
## p_A_a2_B_b1_A_a2_B_b2 76.52    -0.6325
```

```
power_res$power_A
```

```
## NULL
```

```
power_res$power_B
```

```
## NULL
```

```
power_res$power_AB
```

```
## NULL
```

We can use the code by Abelson (2019) to produce the same results.

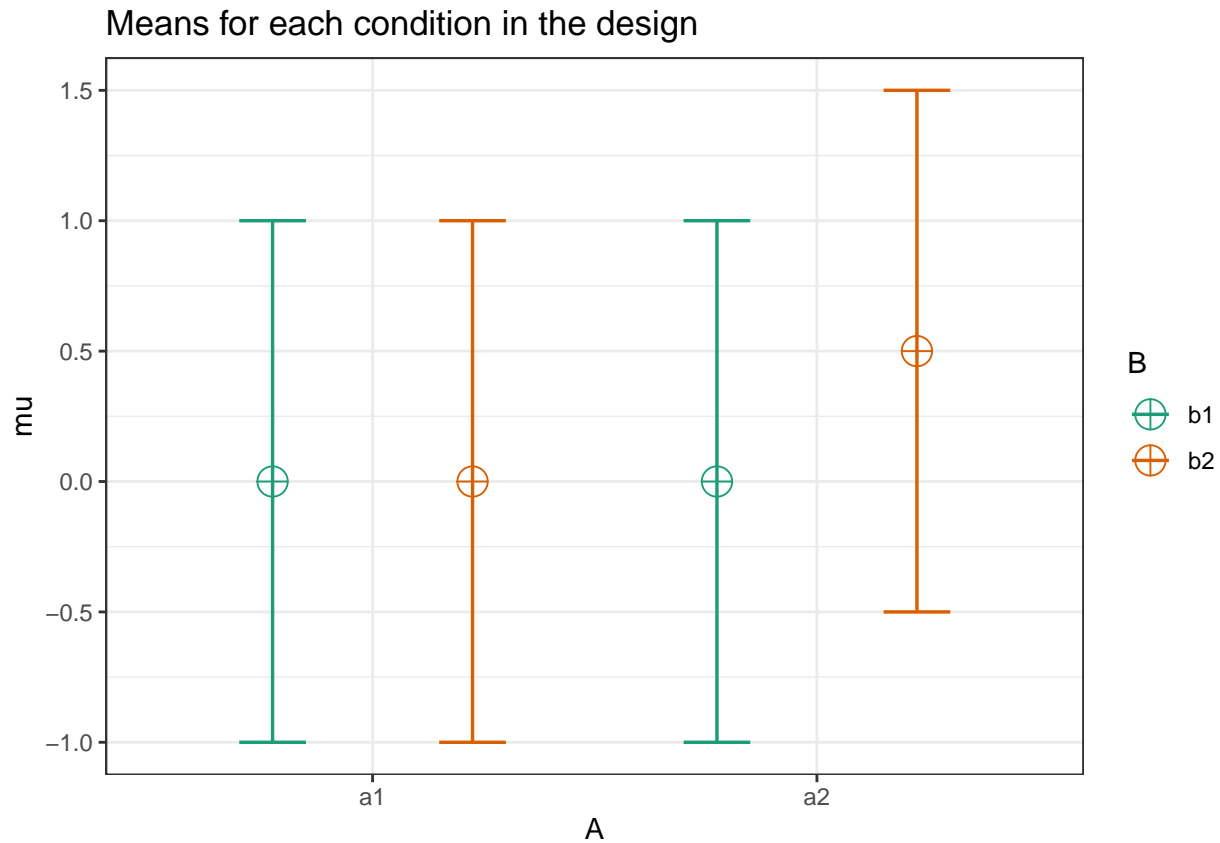
```
win2F(m1.1=2, m2.1=1, m1.2=4, m2.2=2,
      s1.1=5, s2.1=5, s1.2=5, s2.2=5,
      r12=0.8, r13=0.5, r14=0.4,
          r23=0.4, r24=0.5,
              r34=0.8,
      n=20)
```


Chapter 15

Power curves

Power is calculated for a specific value of an effect size, alpha level, and sample size. Because you often do not know the true effect size, it often makes more sense to think of the power curve as a function of the size of the effect. Although power curves can be calculated based on simulations for any design, we will use the analytic solution to calculate the power of ANOVA designs because these calculations are much faster. The basic approach is to calculate power for a specific pattern of means, a specific effect size, a given alpha level, and a specific pattern of correlations. This is one example:

```
#2x2 design
string = "2w*2w"
mu = c(0,0,0,0.5)
n <- 20
sd <- 1
r <- 0.5
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
design_result <- ANOVA_design(design = string,
                             n = n,
                             mu = mu,
                             sd = sd,
                             r = r,
                             labelnames = labelnames)
```



```
power_res <- ANOVA_exact(design_result)
```

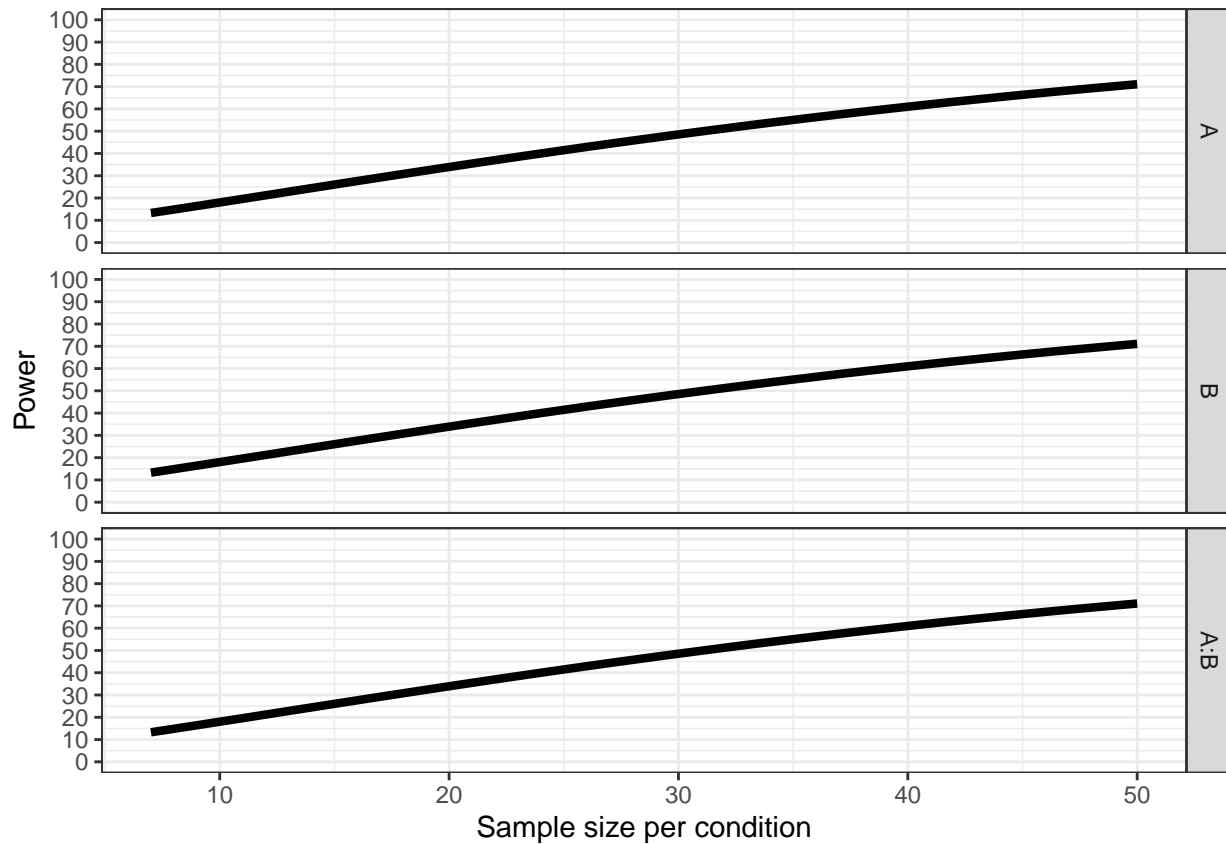
```
## Power and Effect sizes for ANOVA tests
##      power partial_eta_squared cohen_f non centrality
## A    32.36                0.1163 0.3627          2.5
## B    32.36                0.1163 0.3627          2.5
## A:B  32.36                0.1163 0.3627          2.5
##
## Power and Effect sizes for contrasts
##              power effect_size
## p_A_a1_B_b1_A_a1_B_b2  5.00      0.0
## p_A_a1_B_b1_A_a2_B_b1  5.00      0.0
## p_A_a1_B_b1_A_a2_B_b2 56.45      0.5
## p_A_a1_B_b2_A_a2_B_b1  5.00      0.0
## p_A_a1_B_b2_A_a2_B_b2 56.45      0.5
## p_A_a2_B_b1_A_a2_B_b2 56.45      0.5
```

```
power_res$main_results
```

```
##      power partial_eta_squared cohen_f non centrality
## A    32.36                0.1163 0.3627          2.5
## B    32.36                0.1163 0.3627          2.5
## A:B  32.36                0.1163 0.3627          2.5
```

We can make these calculations for a range of sample sizes, to get a power curve. We created a simple function that performs these calculations across a range of sample sizes (from $n = 2$ to `max_`, a variable you can specify in the function).

```
p_a <- plot_power(design_result,  
                  max_n = 50)
```

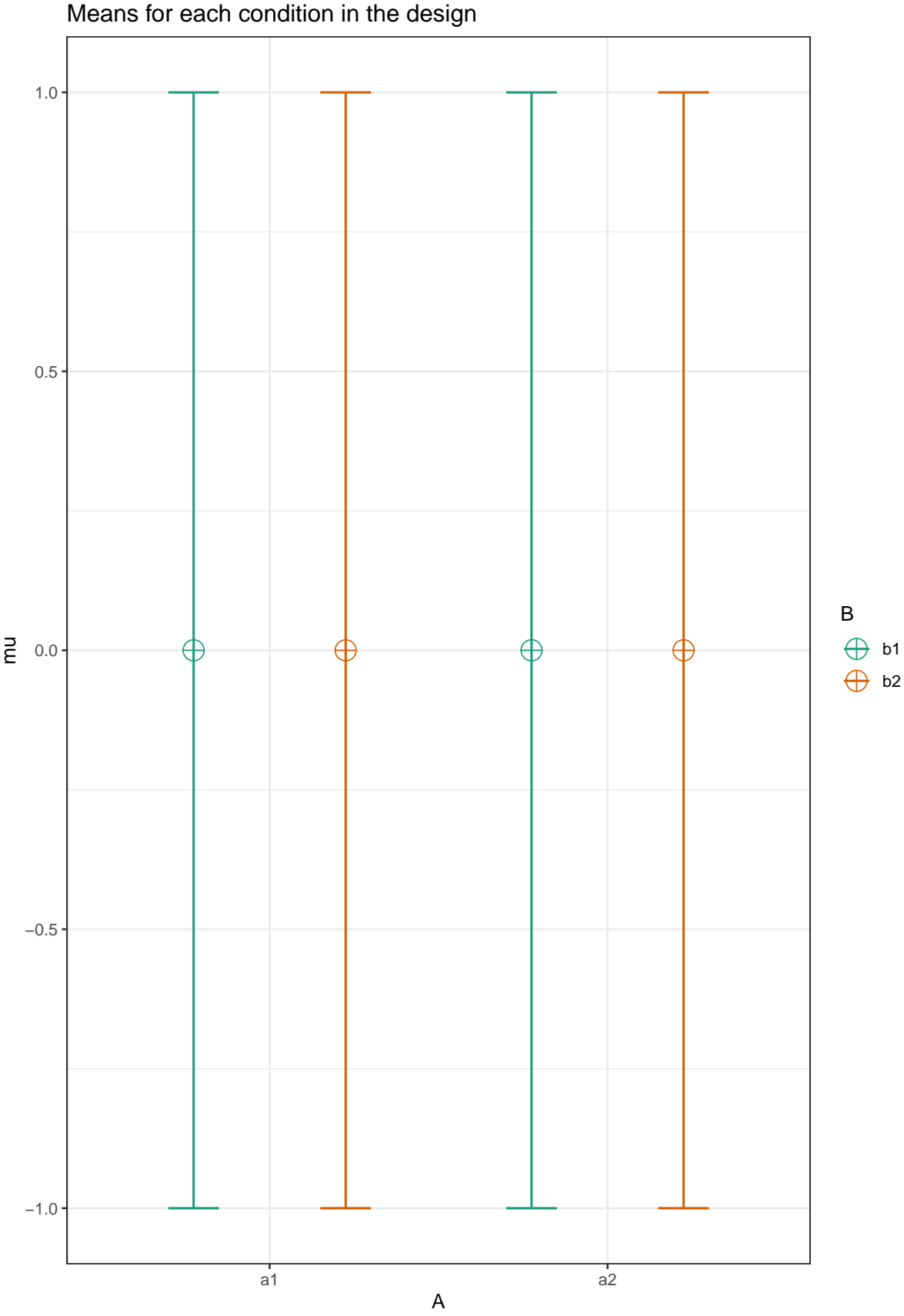


Chapter 16

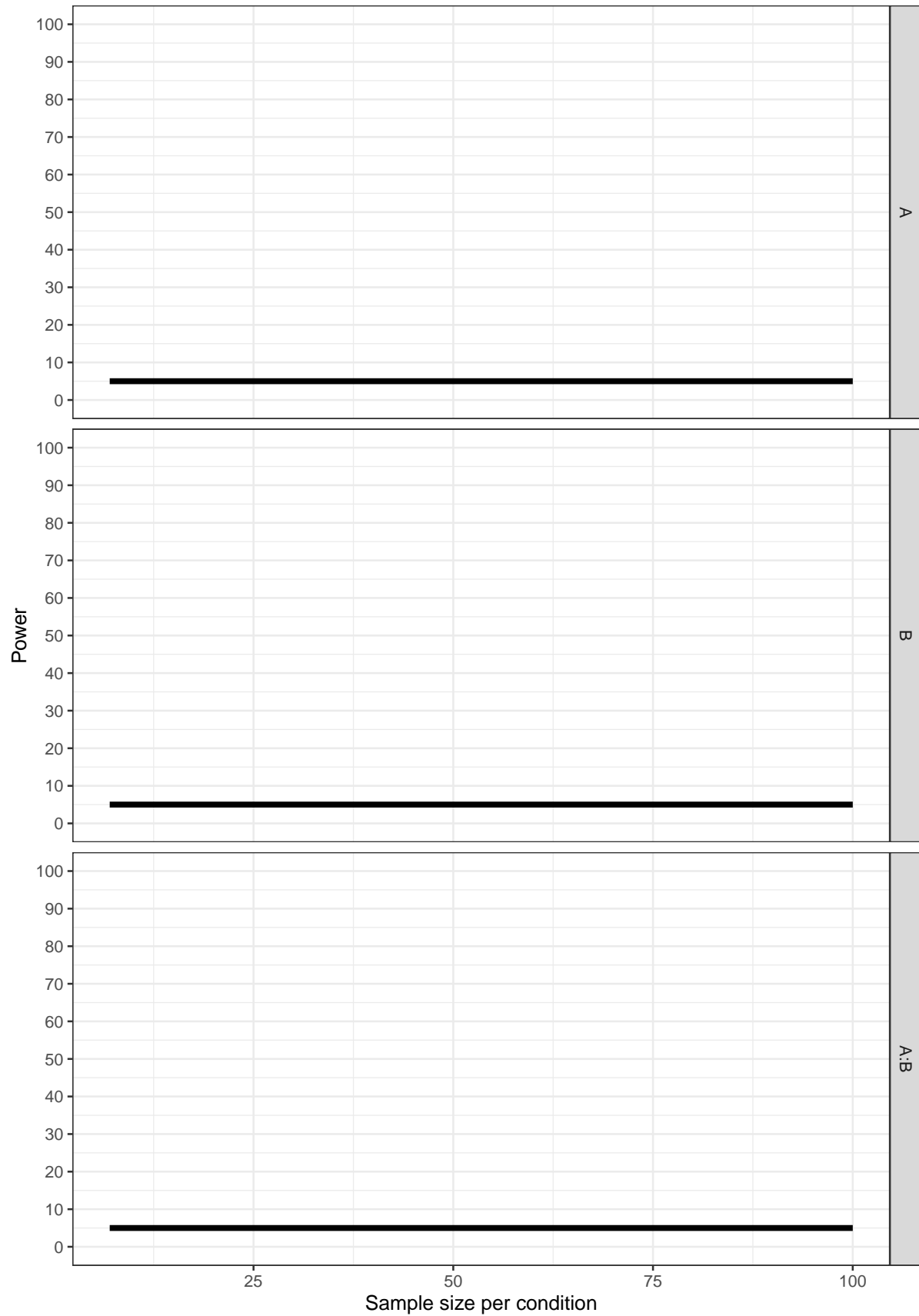
Explore increase in effect size for moderated interactions.

The design has means 0, 0, 0, 0, with one cell increasing by 0.1, up to 0, 0, 0, 0.5. The standard deviation is set to 1. The correlation between all variables is 0.5.

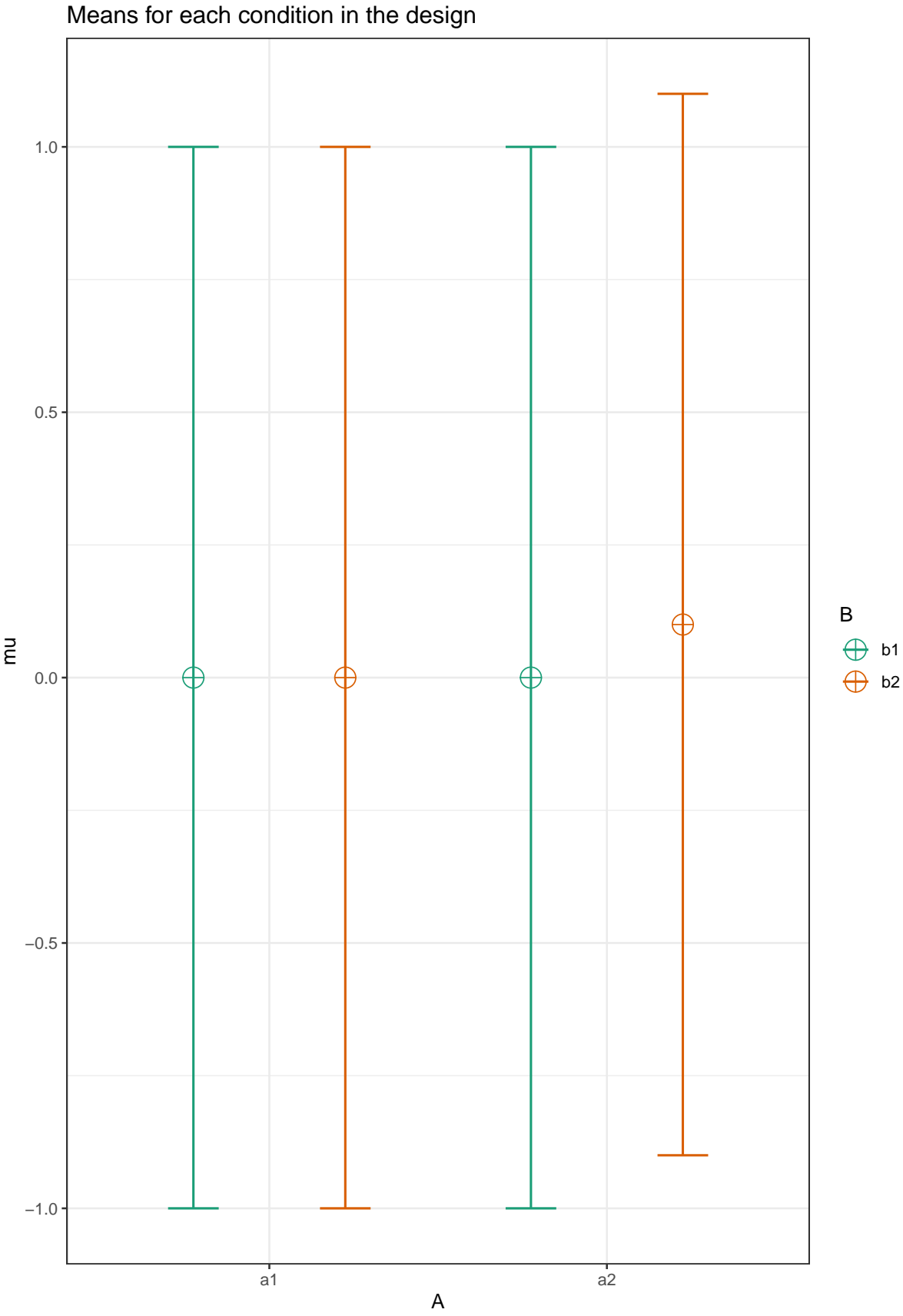
```
string <- "2w*2w"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
design_result <- ANOVA_design(design = string,
                             n = 20,
                             mu = c(0,0,0,0.0),
                             sd = 1,
                             r = 0.5,
                             labelnames = labelnames)
```



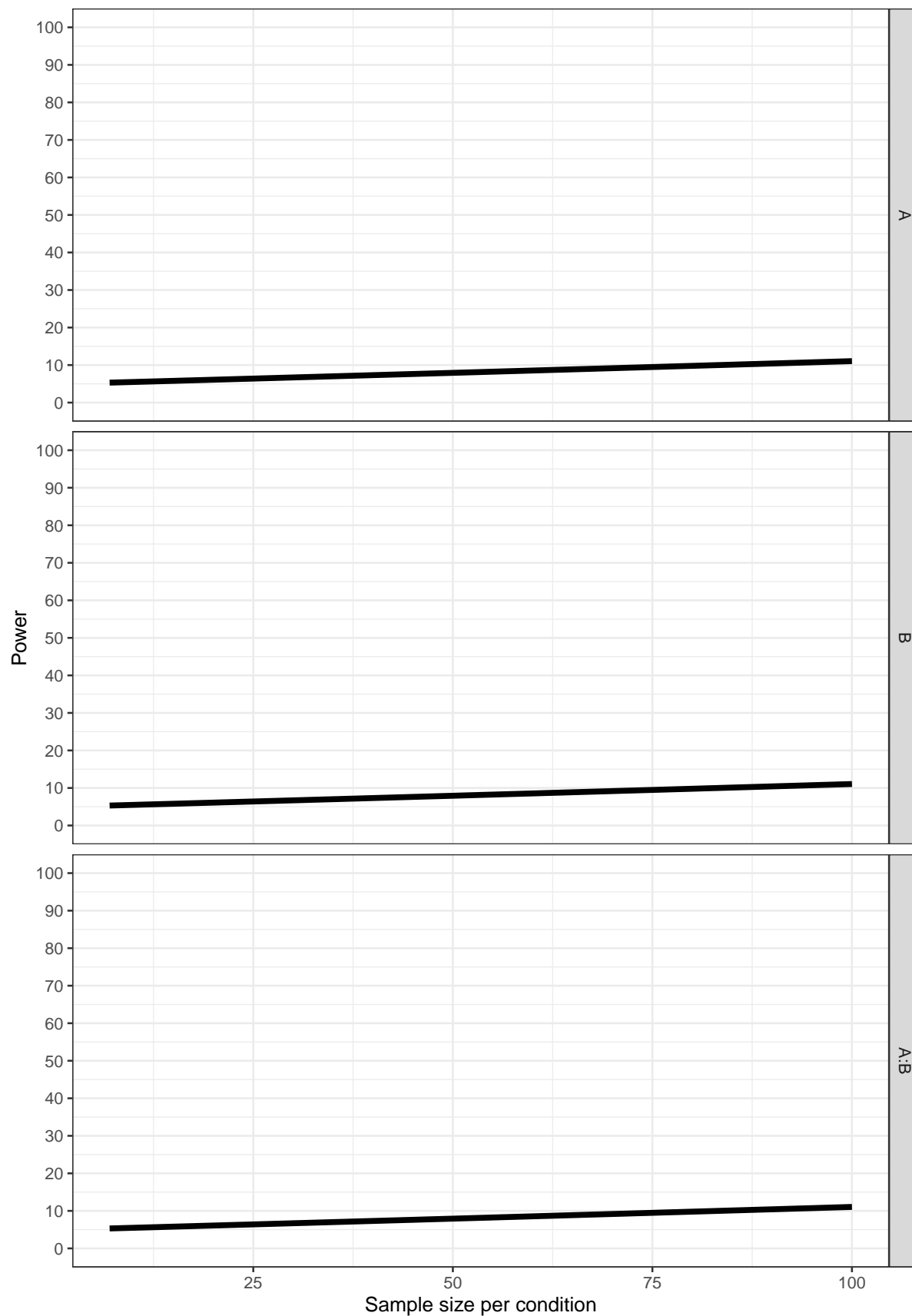

```
p_a <- plot_power(design_result,  
                  max_n = 100)
```



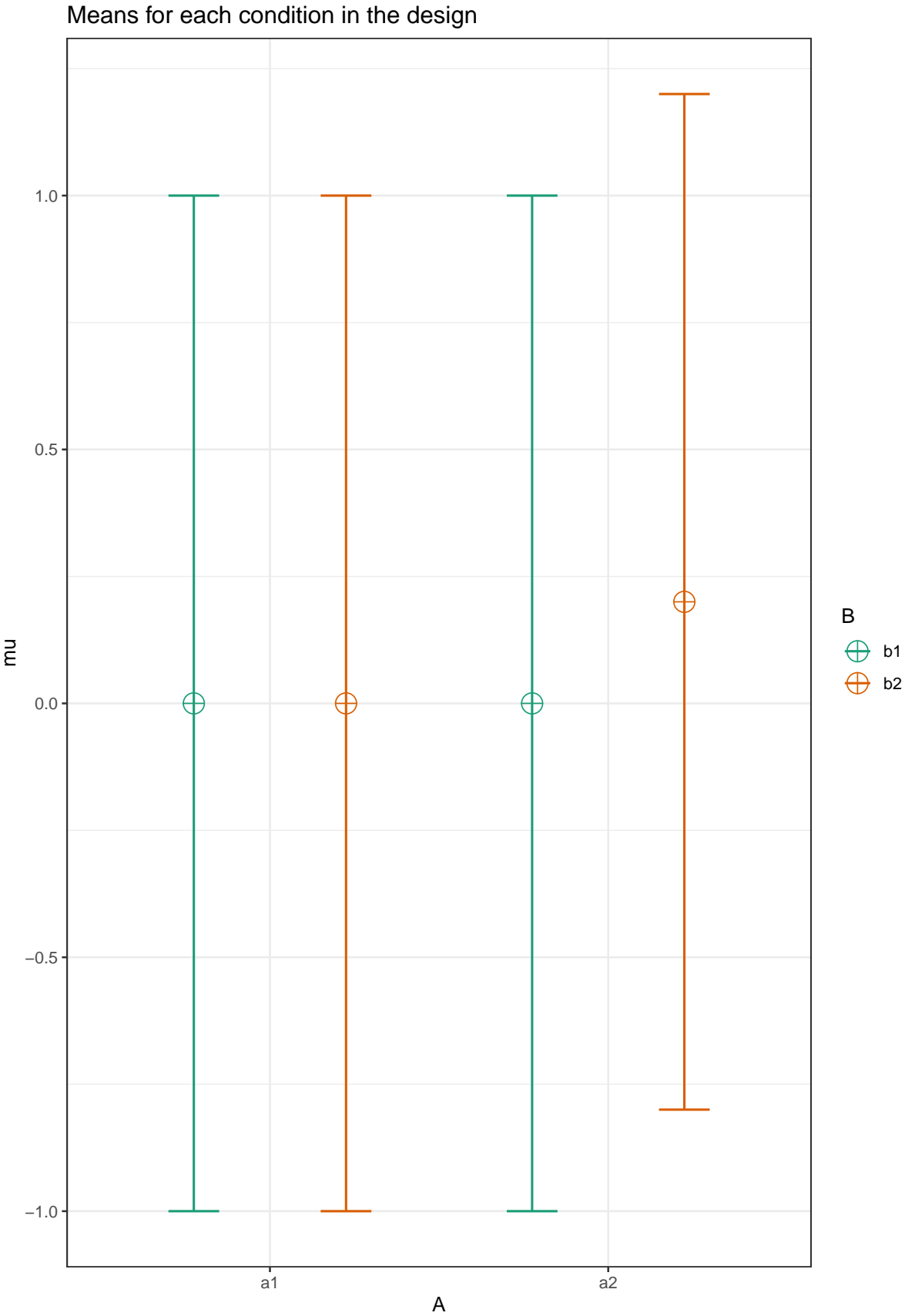
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.1),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



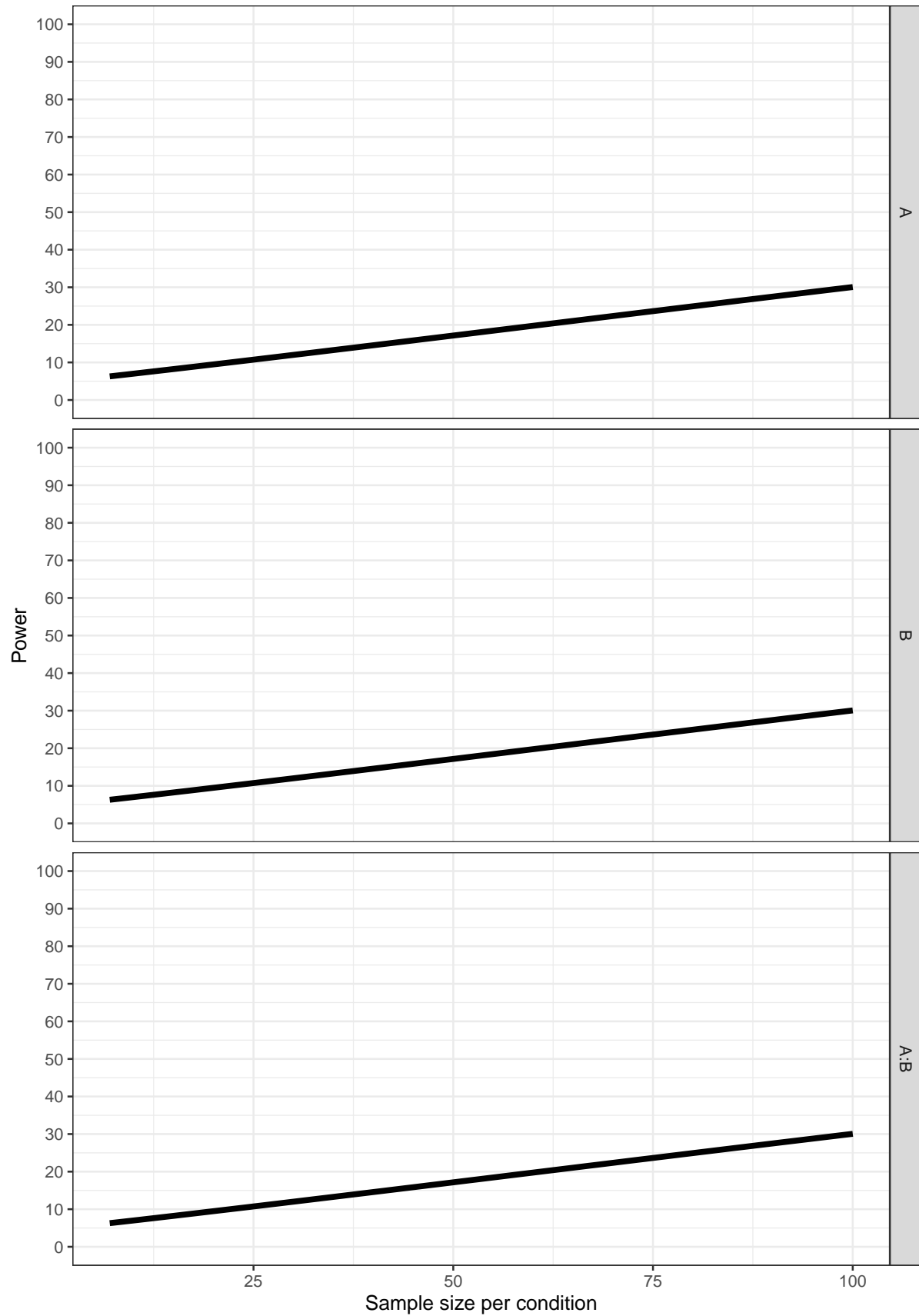
```
p_b <- plot_power(design_result,  
                  max_n = 100)
```



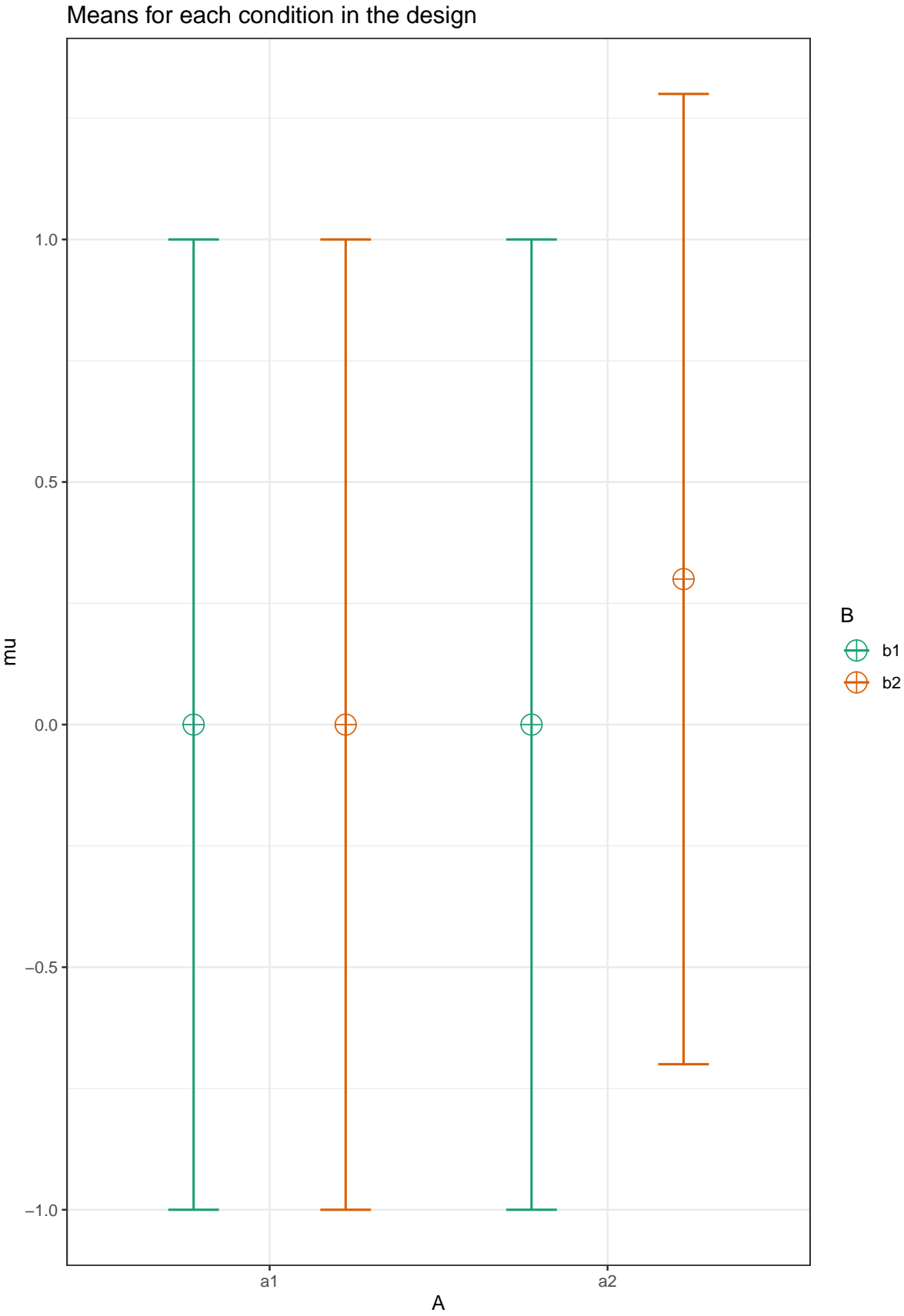
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.2),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



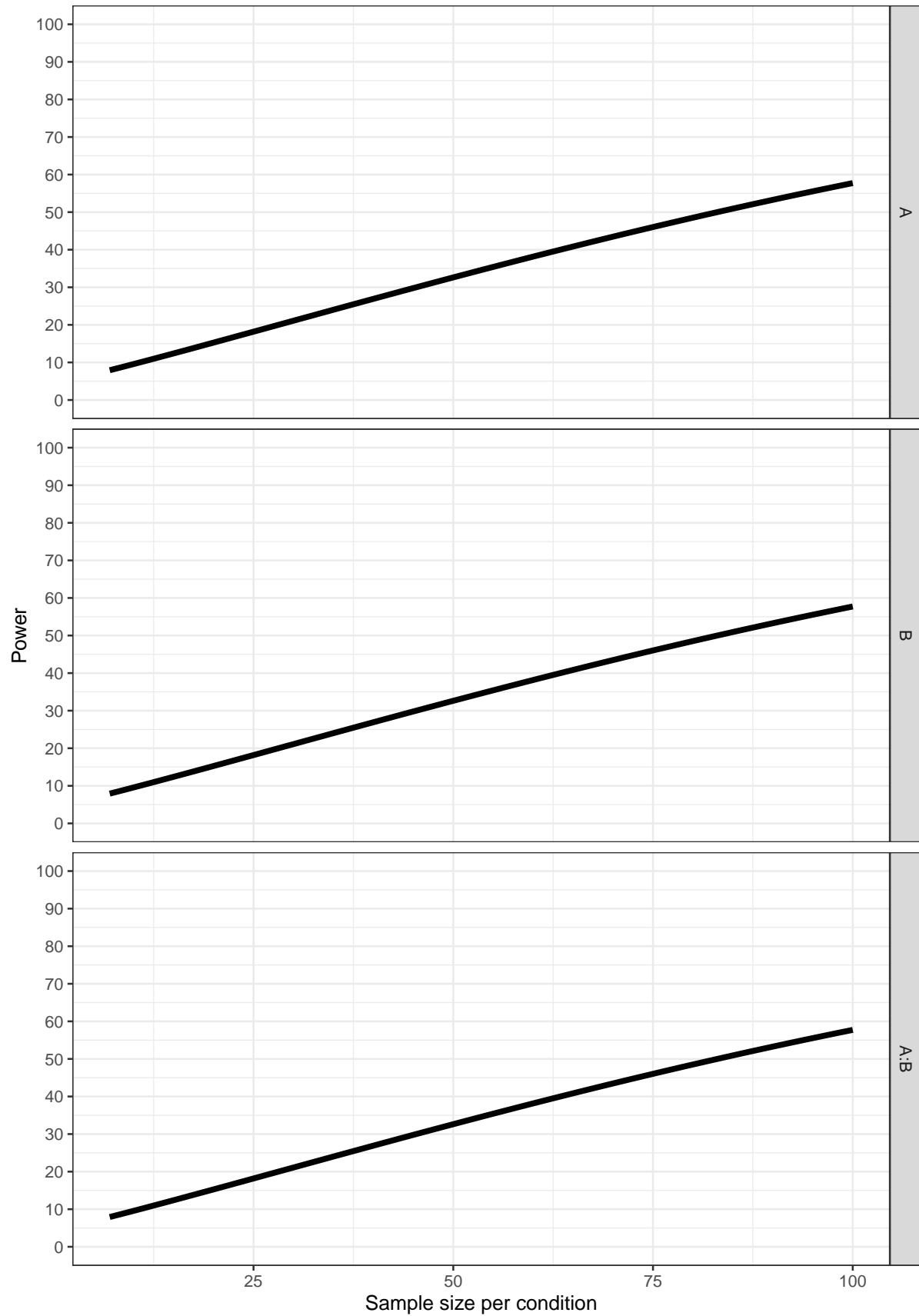

```
p_c <- plot_power(design_result,  
                  max_n = 100)
```



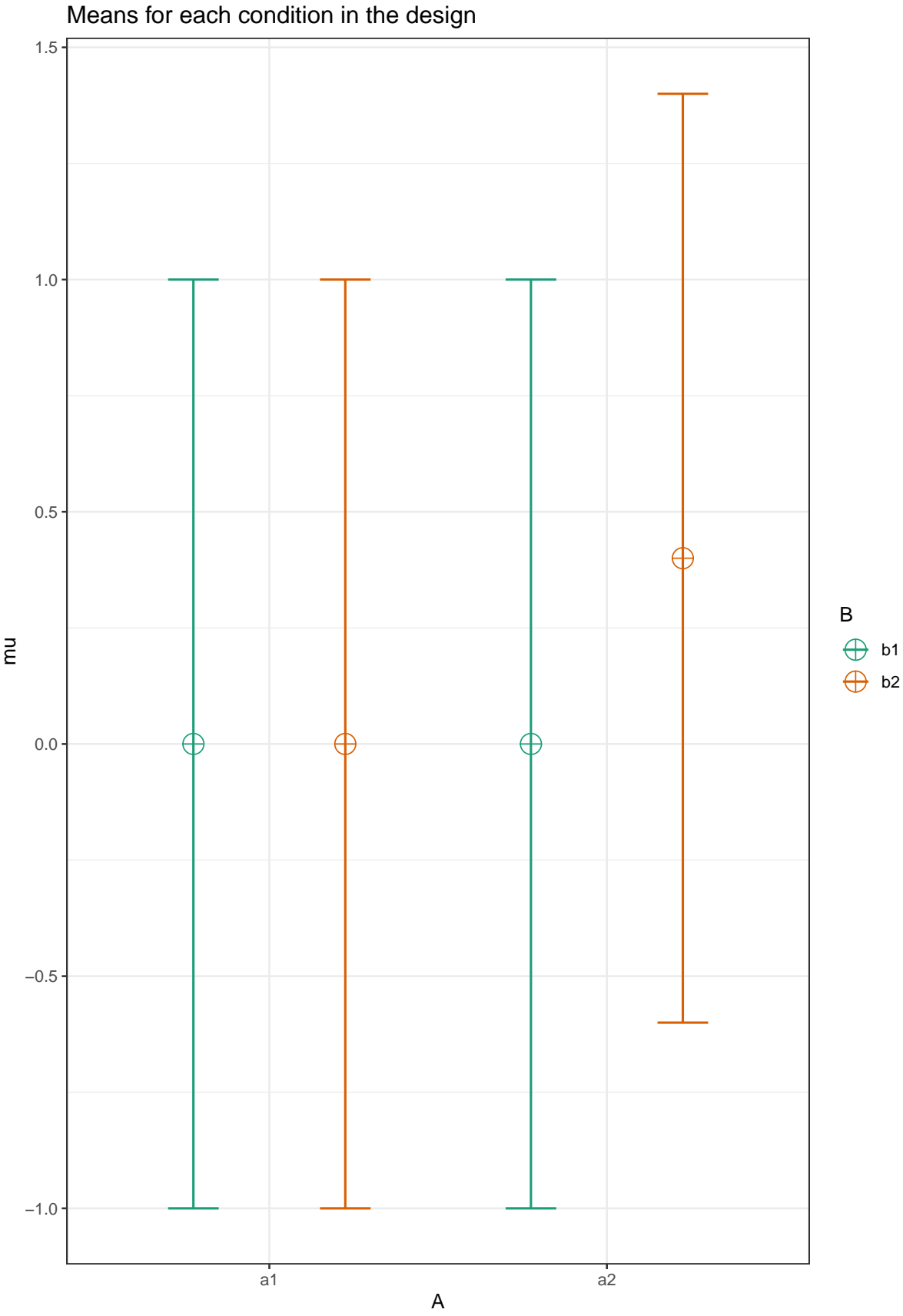
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



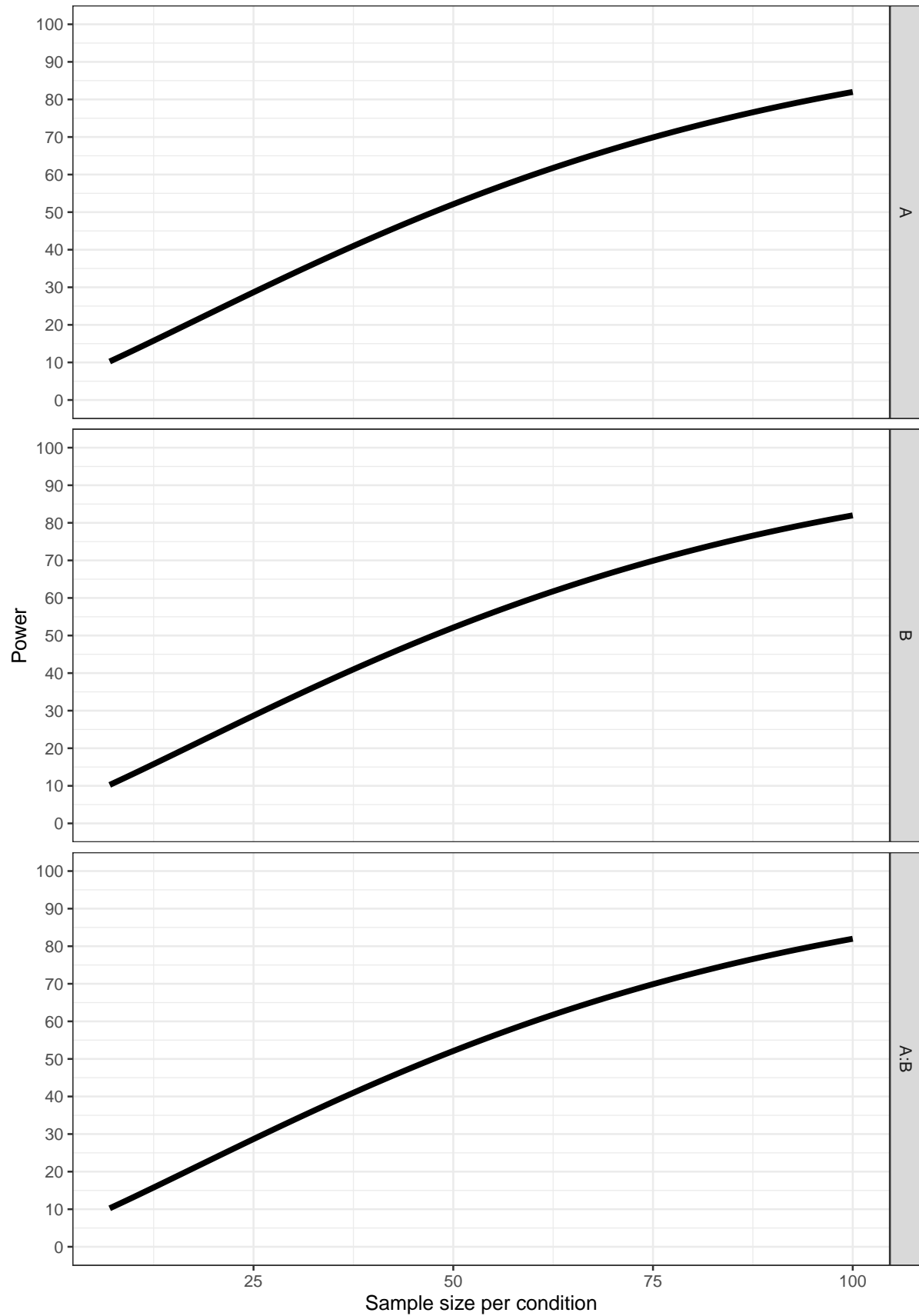
```
p_d <- plot_power(design_result,  
                  max_n = 100)
```



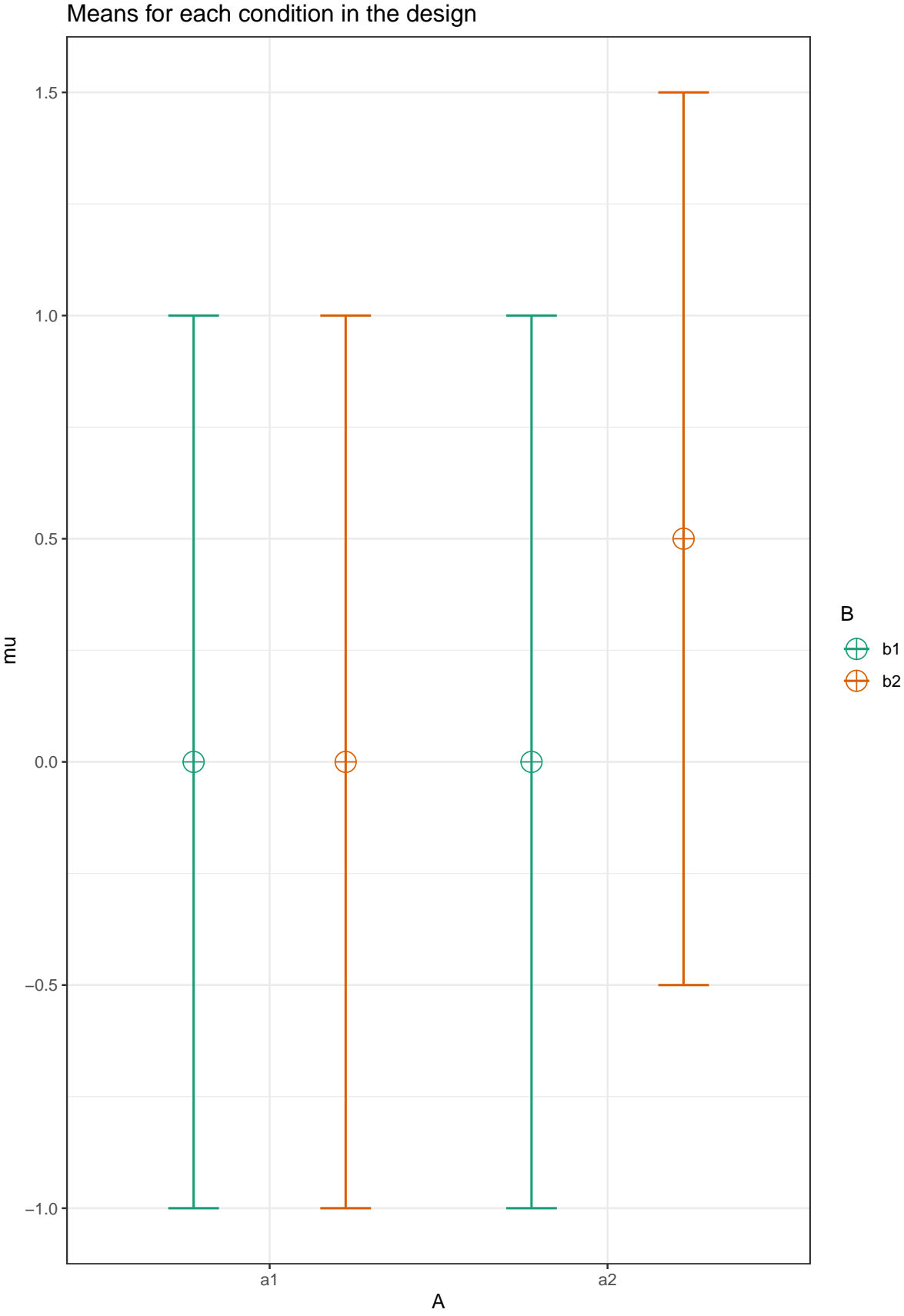
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.4),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



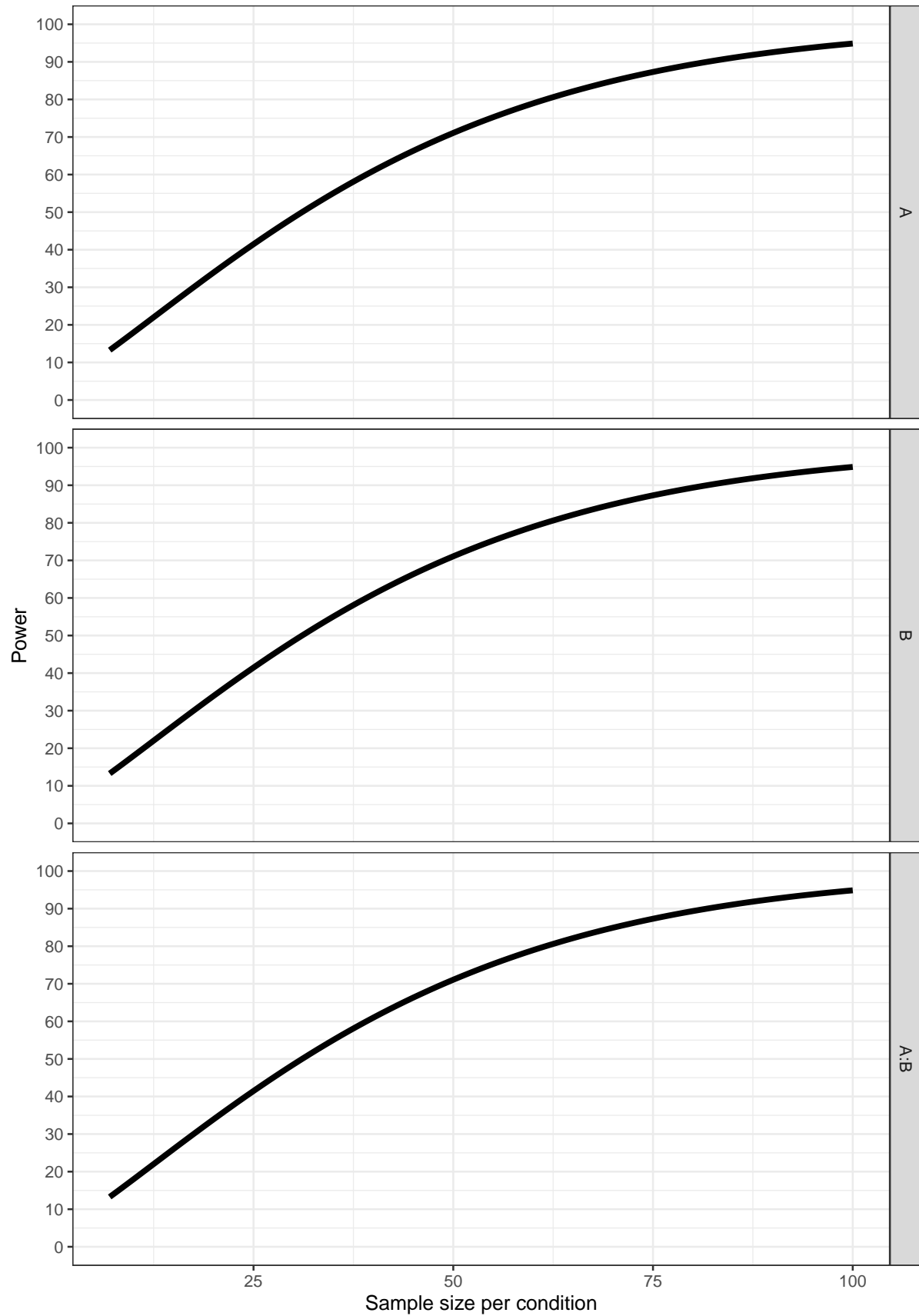

```
p_e <- plot_power(design_result,  
                  max_n = 100)
```



```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.5),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



```
p_f <- plot_power(design_result,  
                  max_n = 100)
```



```

# Create long format dataframe
#zzz <- rbind(p_a$power_df, p_b$power_df, p_c$power_df, p_d$power_df, p_e$power_df, p_f$power_df)
#zzz <- cbind(zzz, seq(1, length(zzz$design)))
#colnames(zzz)[1] <- "design"
#colnames(zzz)[6] <- "ID"
#zzz <- melt(zzz, id.vars = c("ID", "design", "n_vec"), measure.vars = c("power_A", "power_B", "power_A"))
# Plot data using facets, split by factors and interaction, and design
#ggplot(data=zzz, aes(x = n_vec, y = value)) +
#  geom_line( size=1.5) +
#  scale_x_continuous(limits = c(0, max(zzz$n_vec))) +
#  scale_y_continuous(limits = c(0, 100)) +
#  theme_bw() +
#  labs(x="Sample size", y = "Power") +
#  facet_grid(design~variable)

```

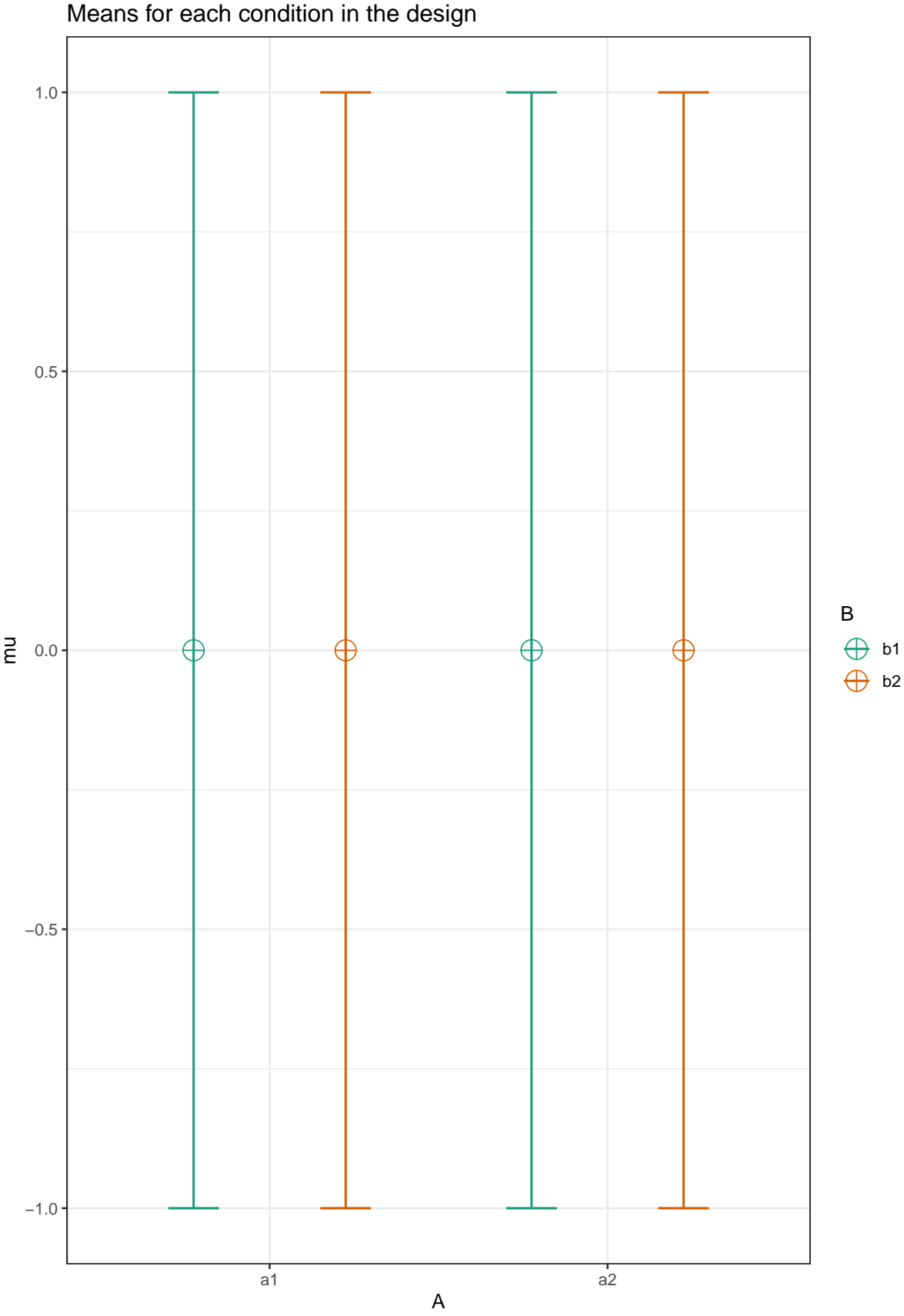
16.1 Explore increase in effect size for cross-over interactions.

The design has means 0, 0, 0, 0, with two cells increasing by 0.1, up to 0.5, 0, 0, 0.5. The standard deviation is set to 1. The correlation between all variables is 0.5.

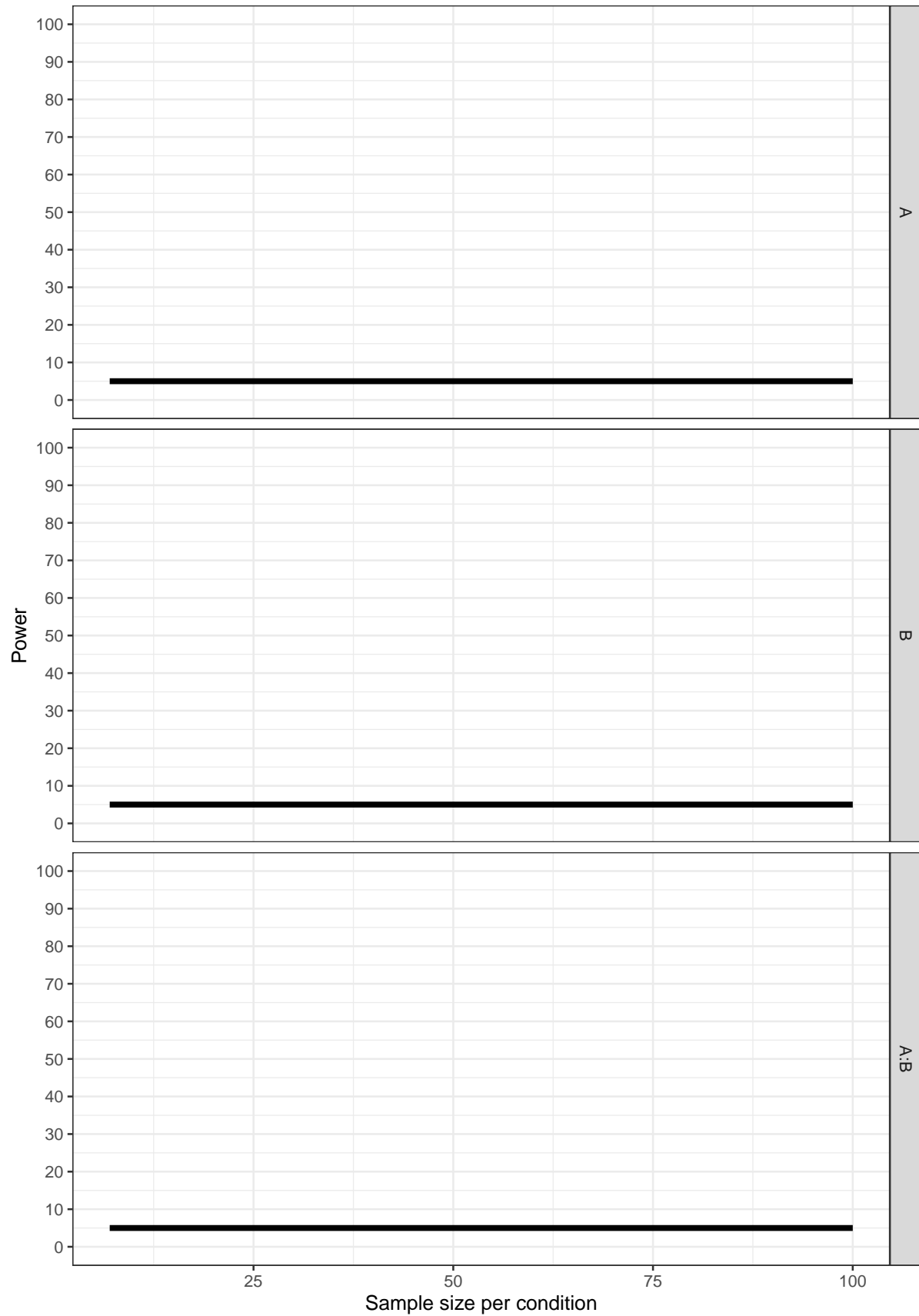
```

design_result <- ANOVA_design(design = string,
  n = 20,
  mu = c(0,0,0,0.0),
  sd = 1,
  r = 0.5,
  labelnames = labelnames)

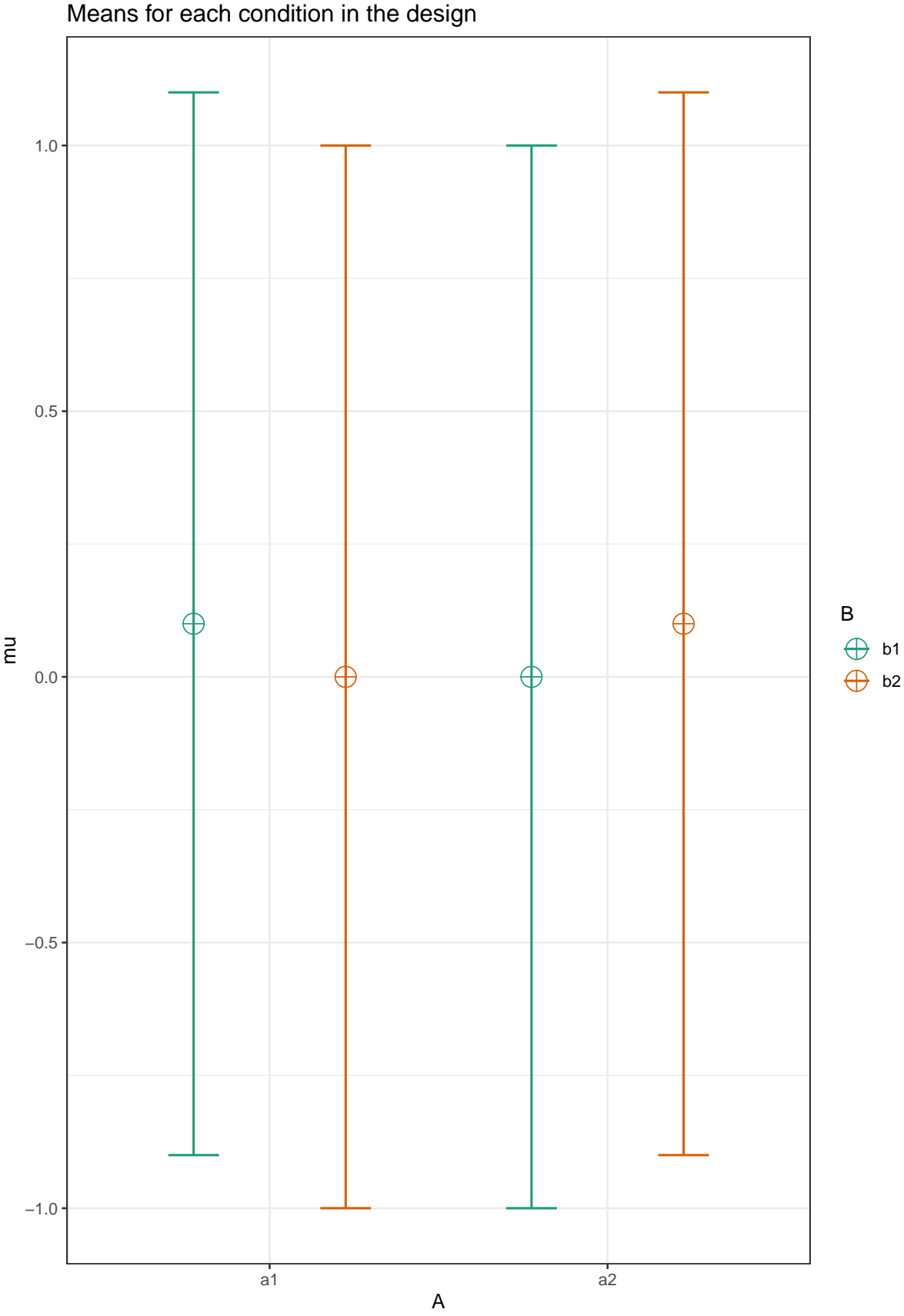
```



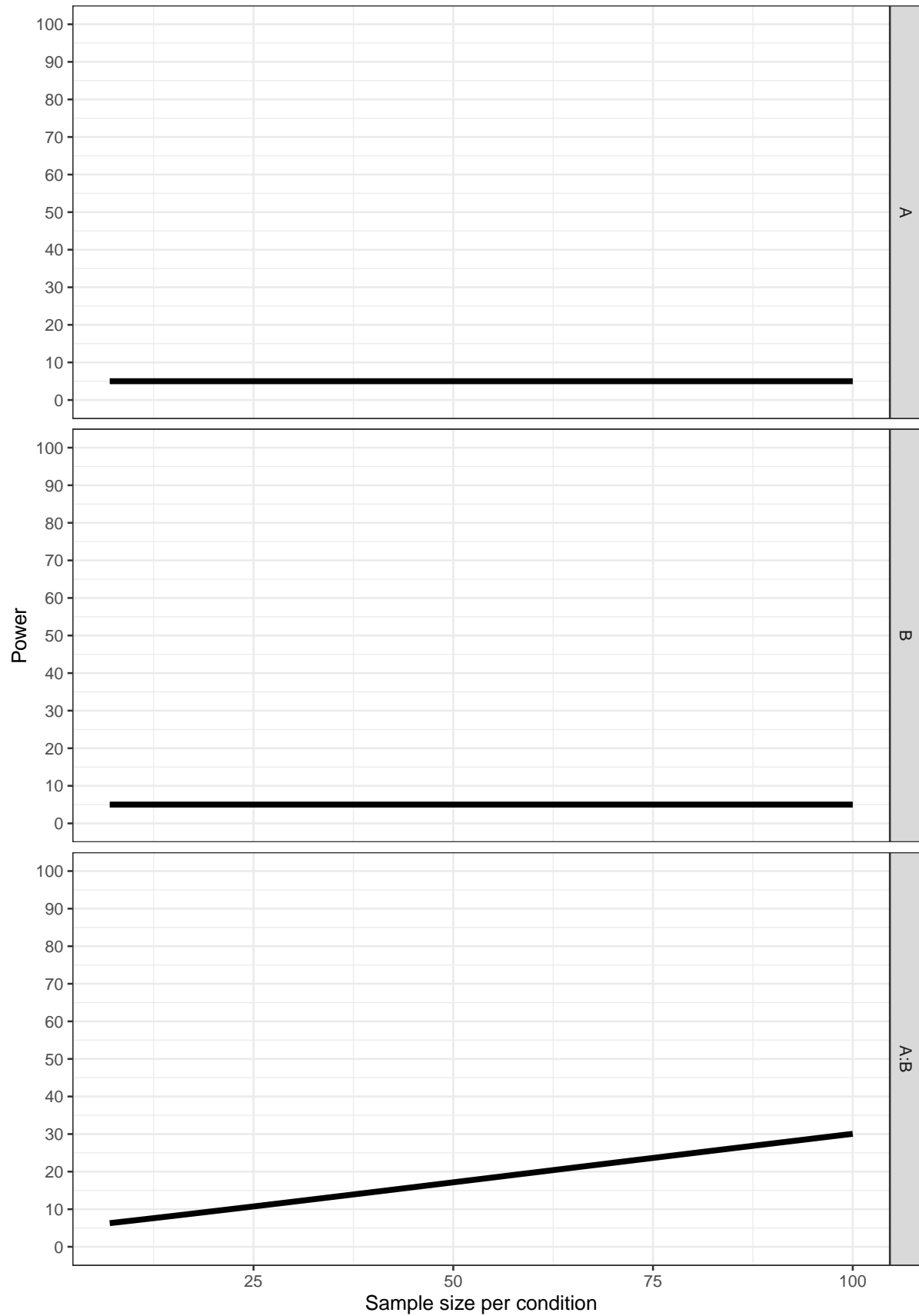

```
p_a <- plot_power(design_result,  
                  max_n = 100)
```



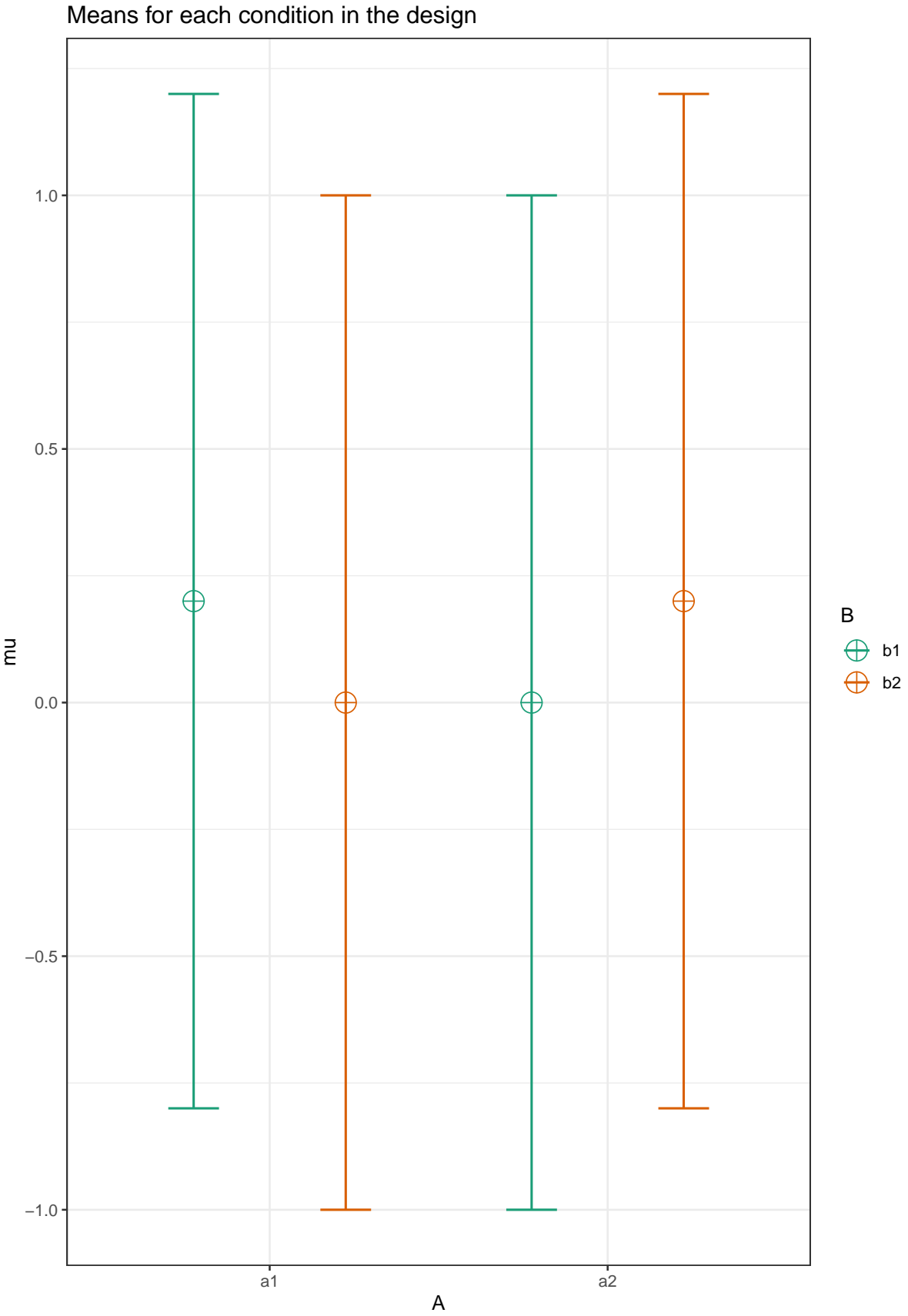
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0.1,0,0,0.1),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



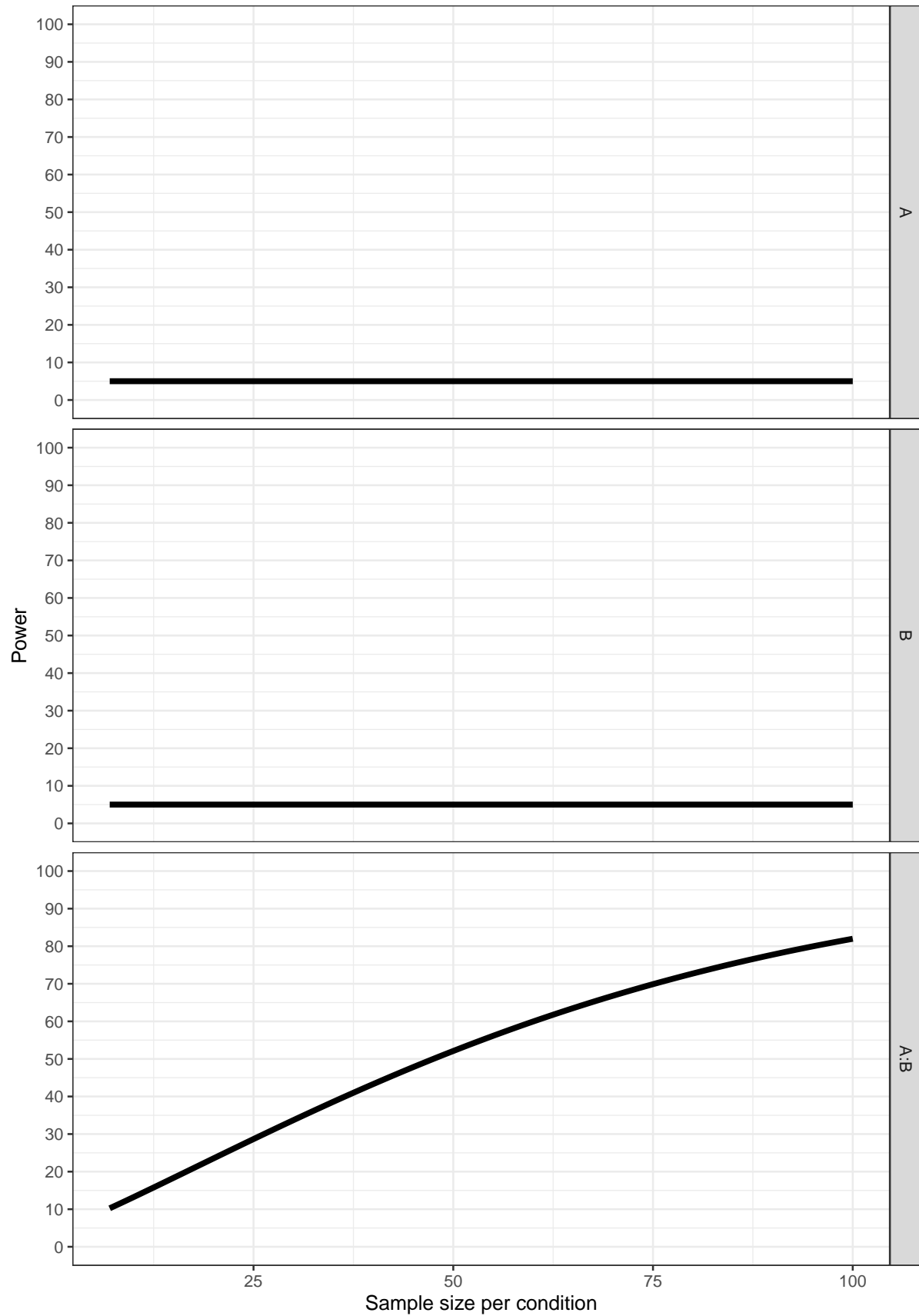
```
p_b <- plot_power(design_result,  
                  max_n = 100)
```



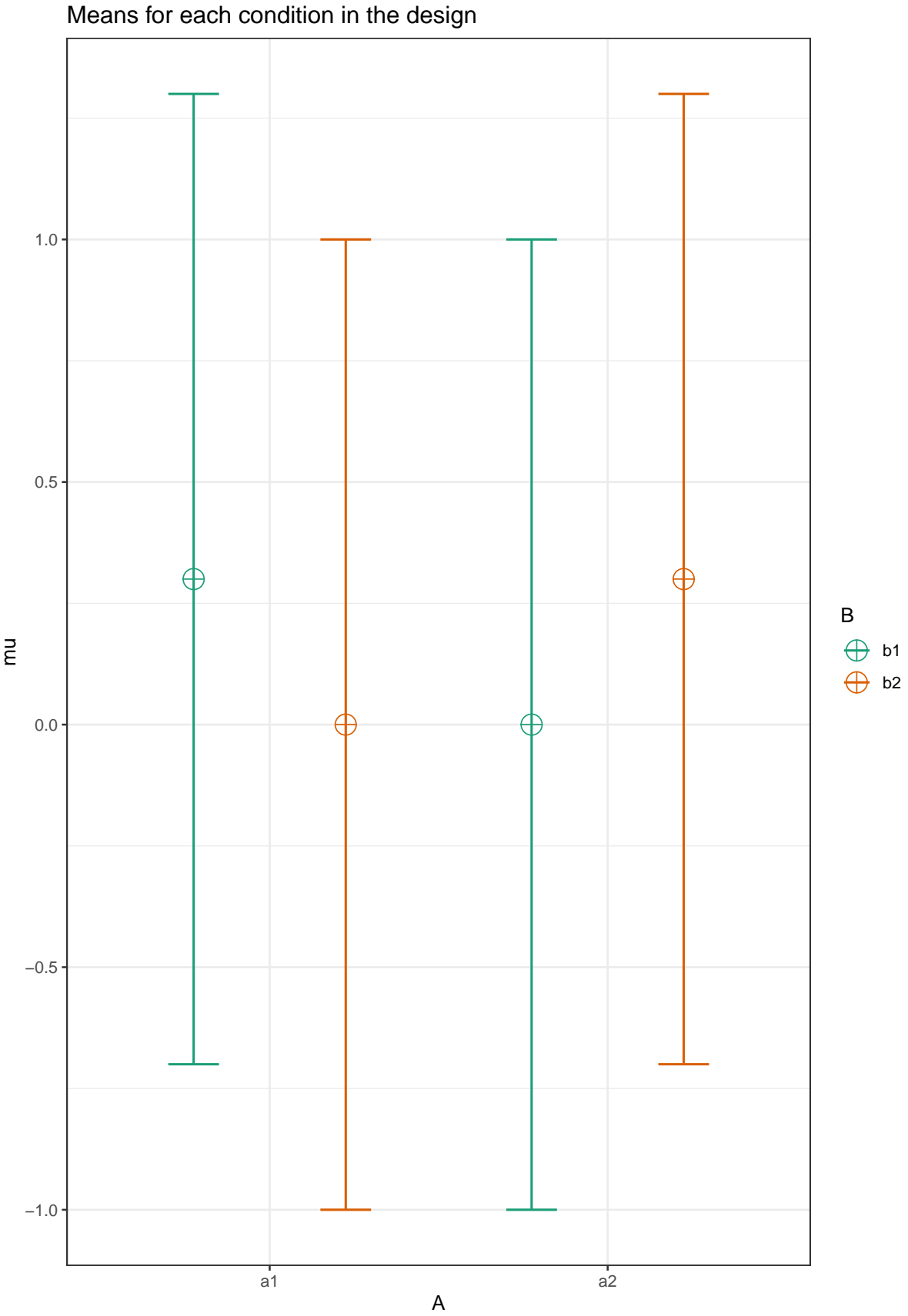
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0.2,0,0,0.2),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



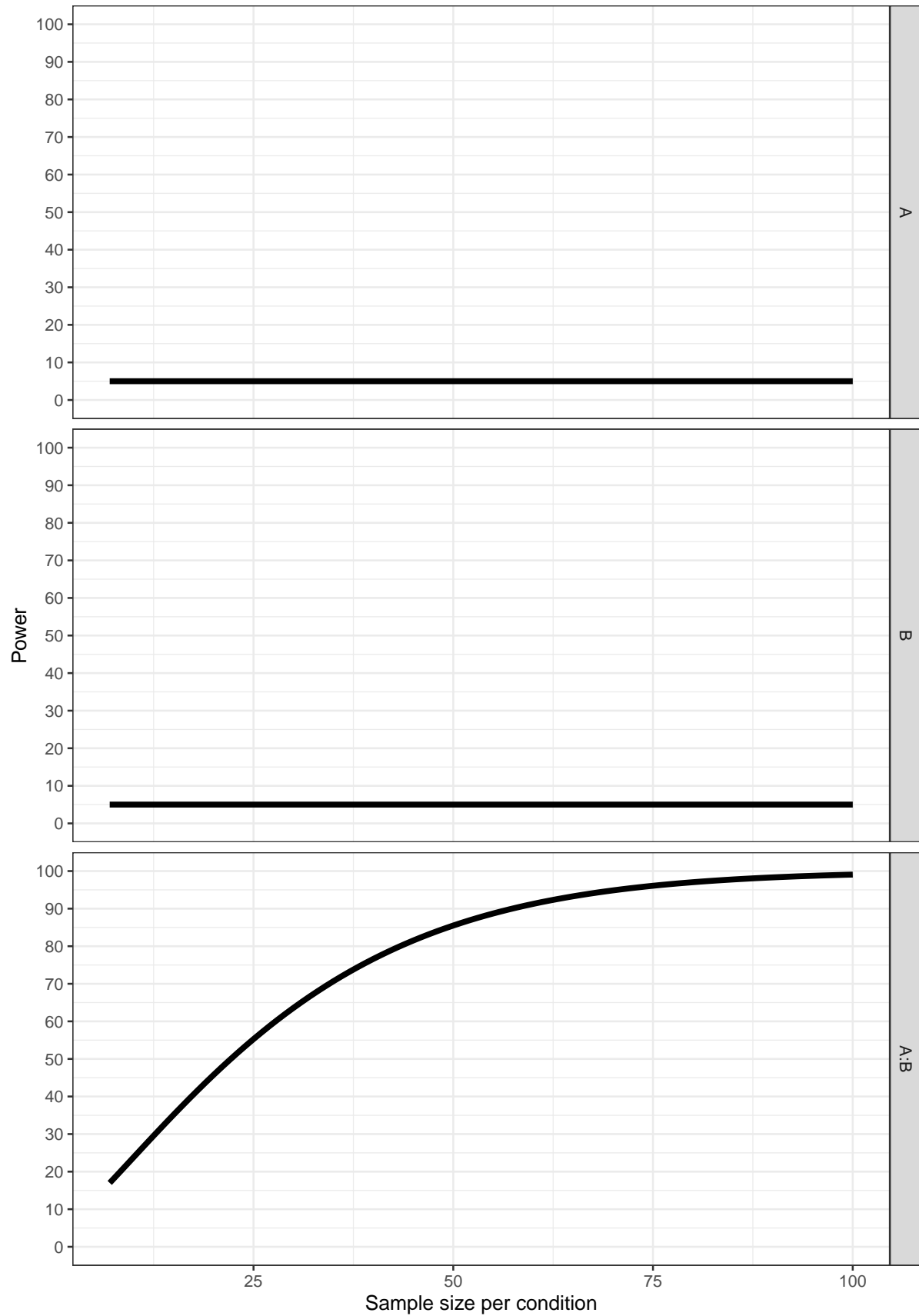

```
p_c <- plot_power(design_result,  
                  max_n = 100)
```



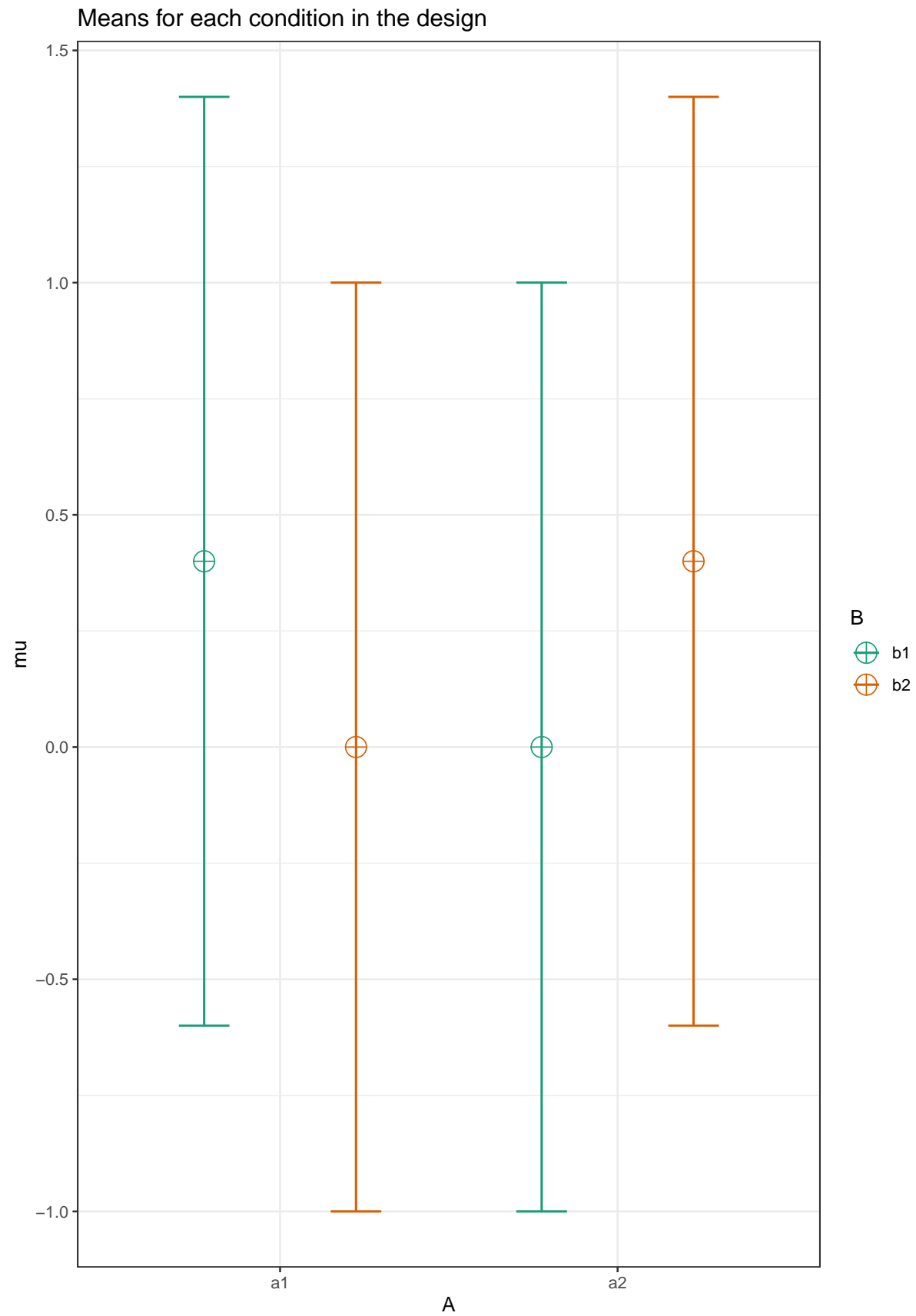
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0.3,0,0,0.3),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



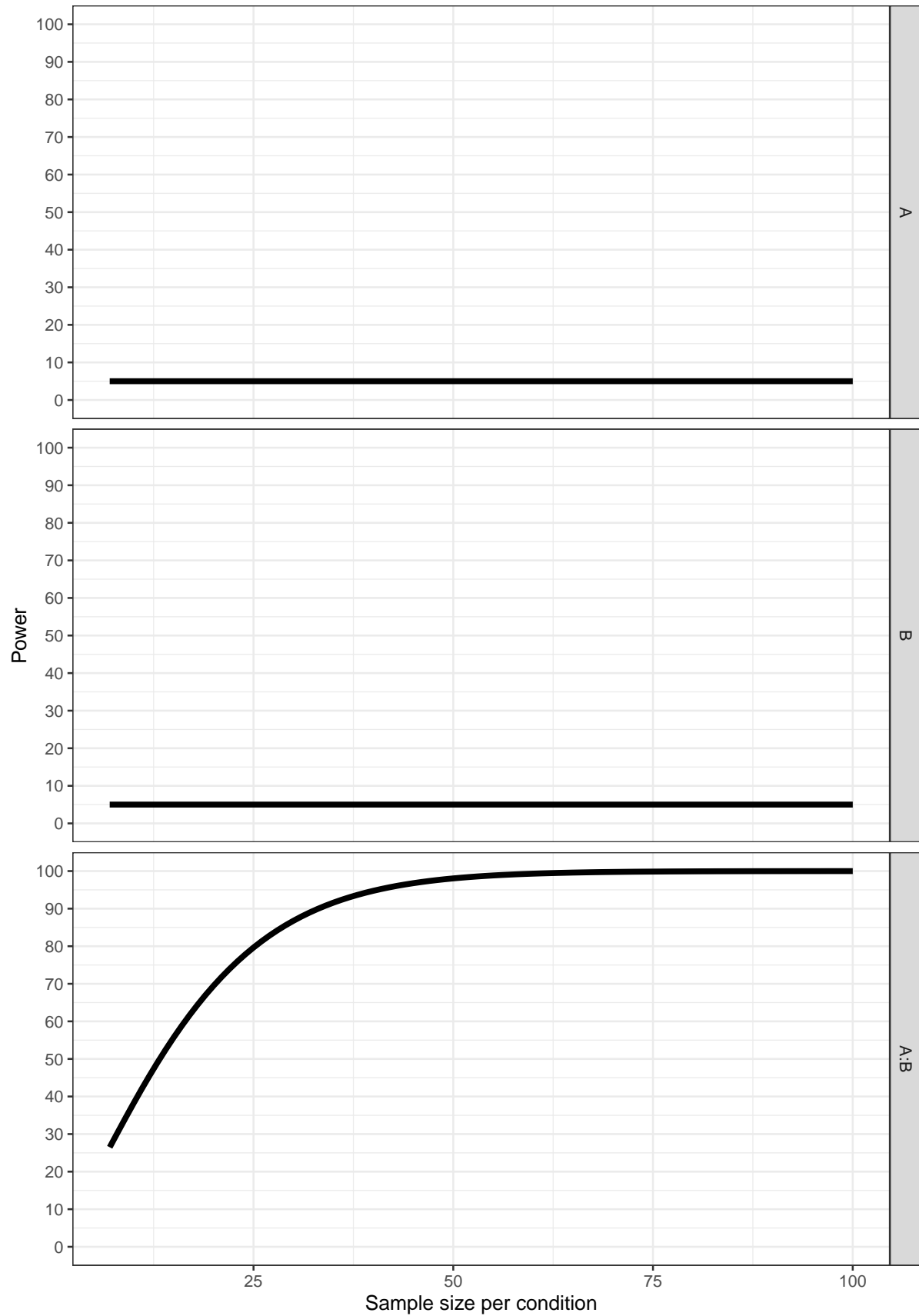
```
p_d <- plot_power(design_result,  
                  max_n = 100)
```



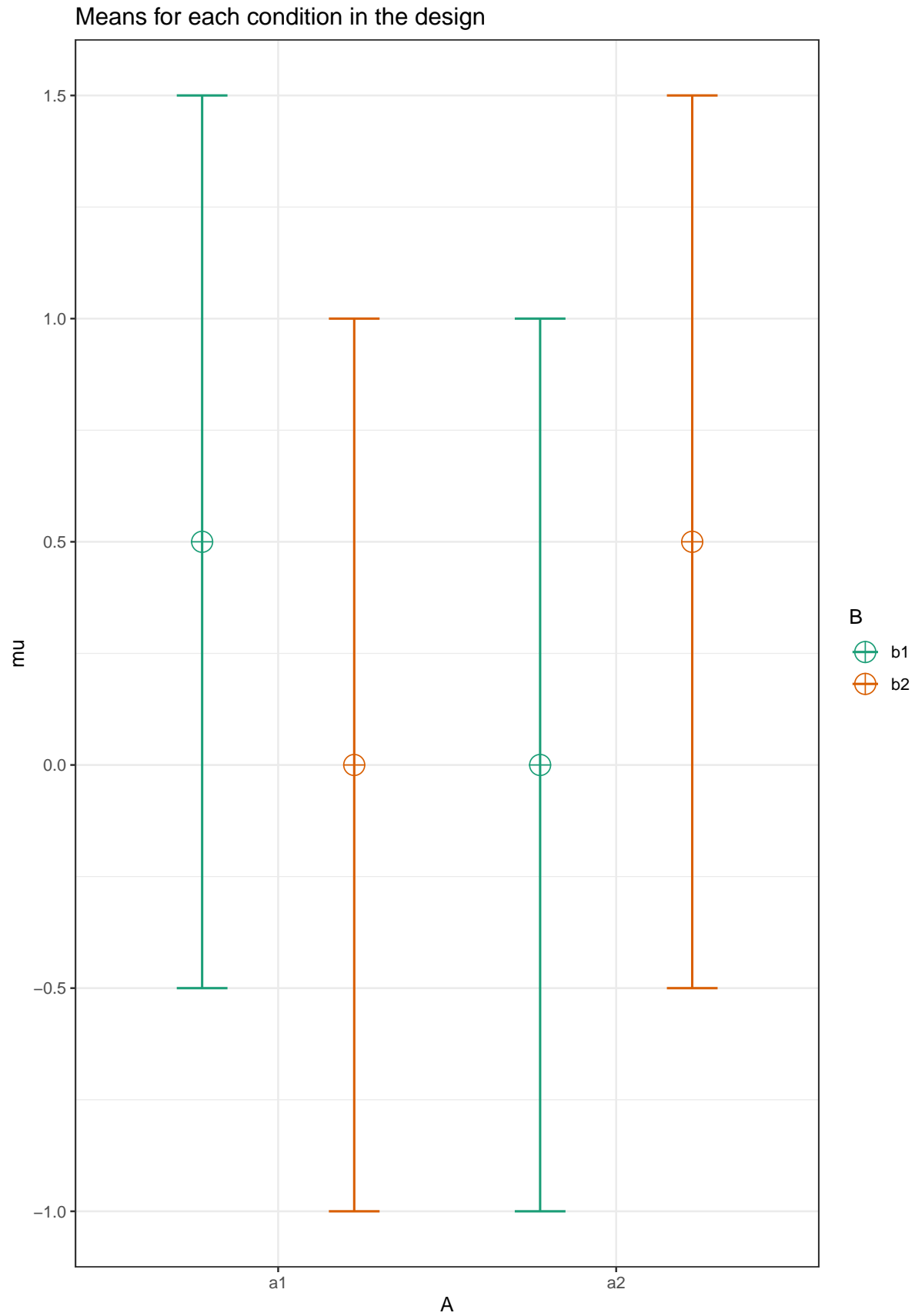
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0.4,0,0,0.4),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



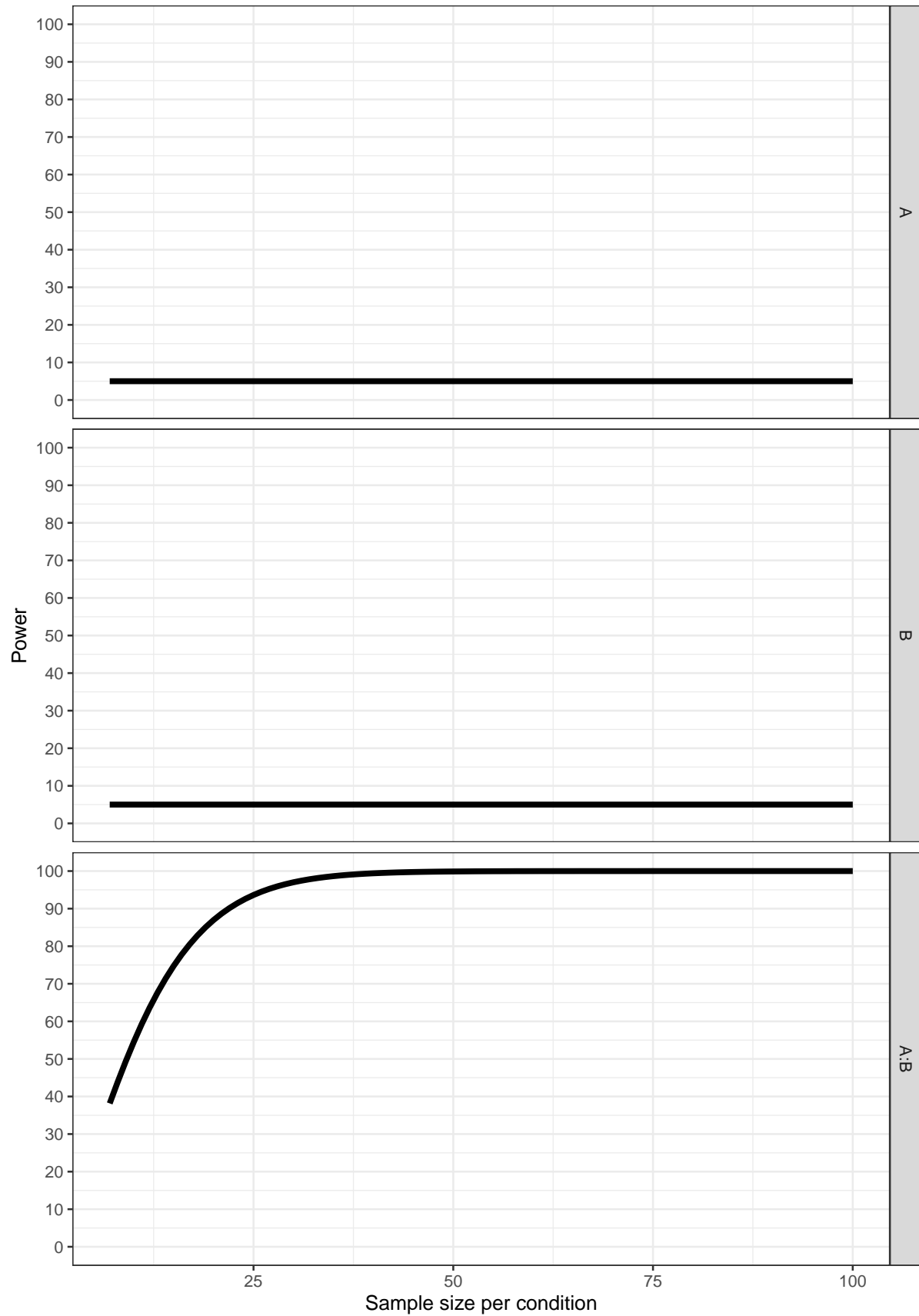

```
p_e <- plot_power(design_result,  
                  max_n = 100)
```



```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0.5,0,0,0.5),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



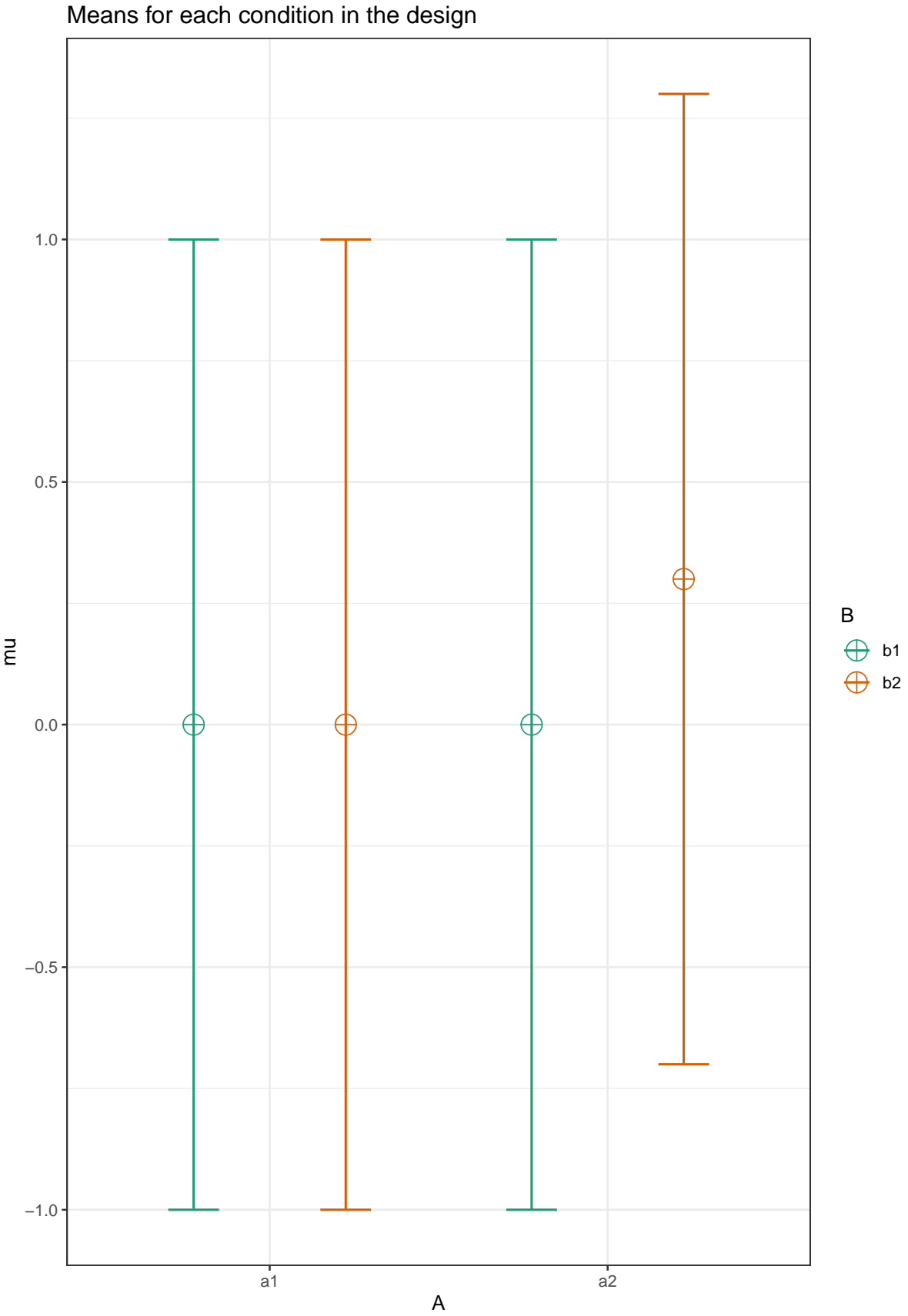
```
p_f <- plot_power(design_result,  
                  max_n = 100)
```



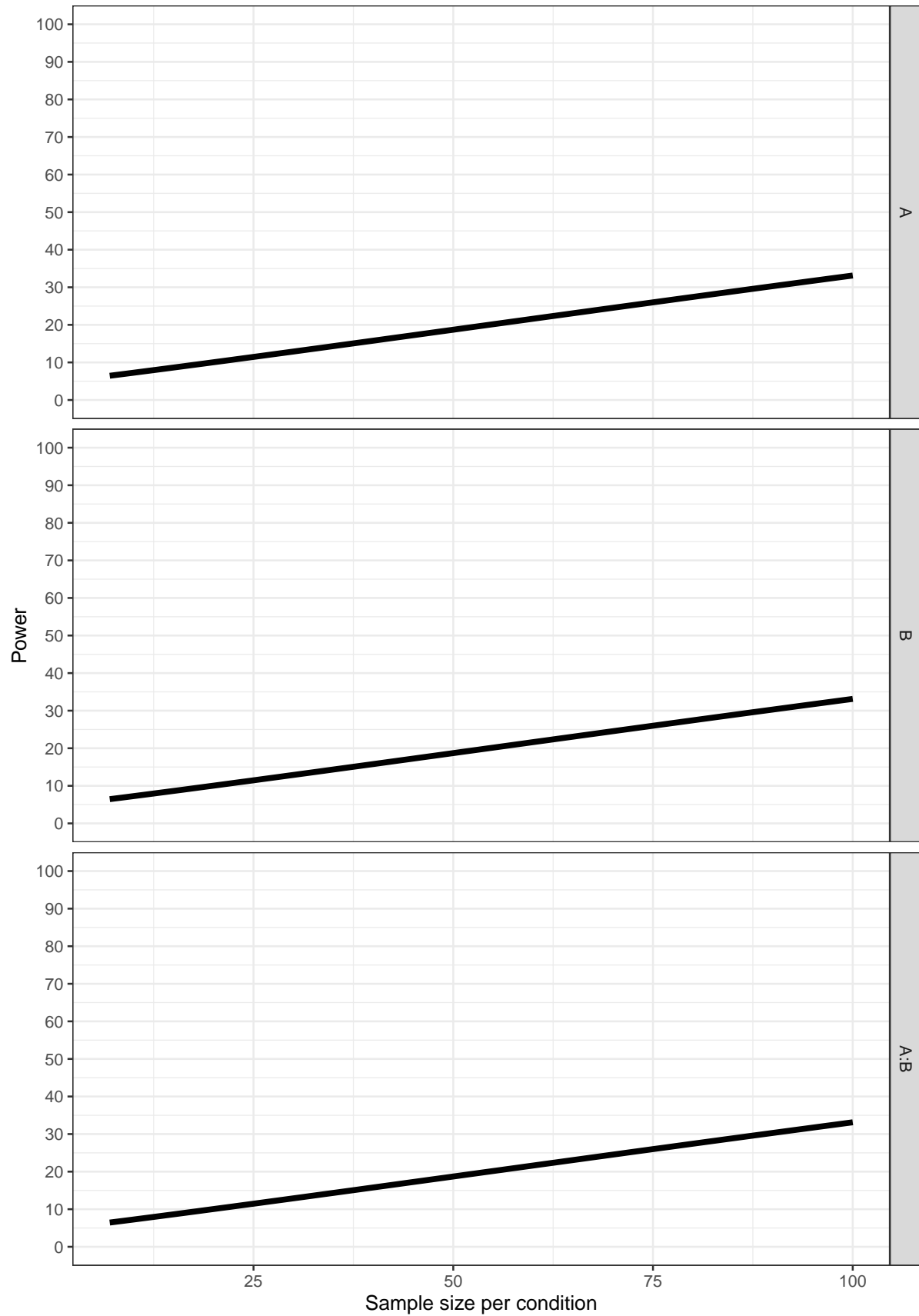
16.2 Explore increase in correlation in moderated interactions.

The design has means 0, 0, 0, 0.3. The standard deviation is set to 1. The correlation between all variables increases from 0.5 to 0.9.

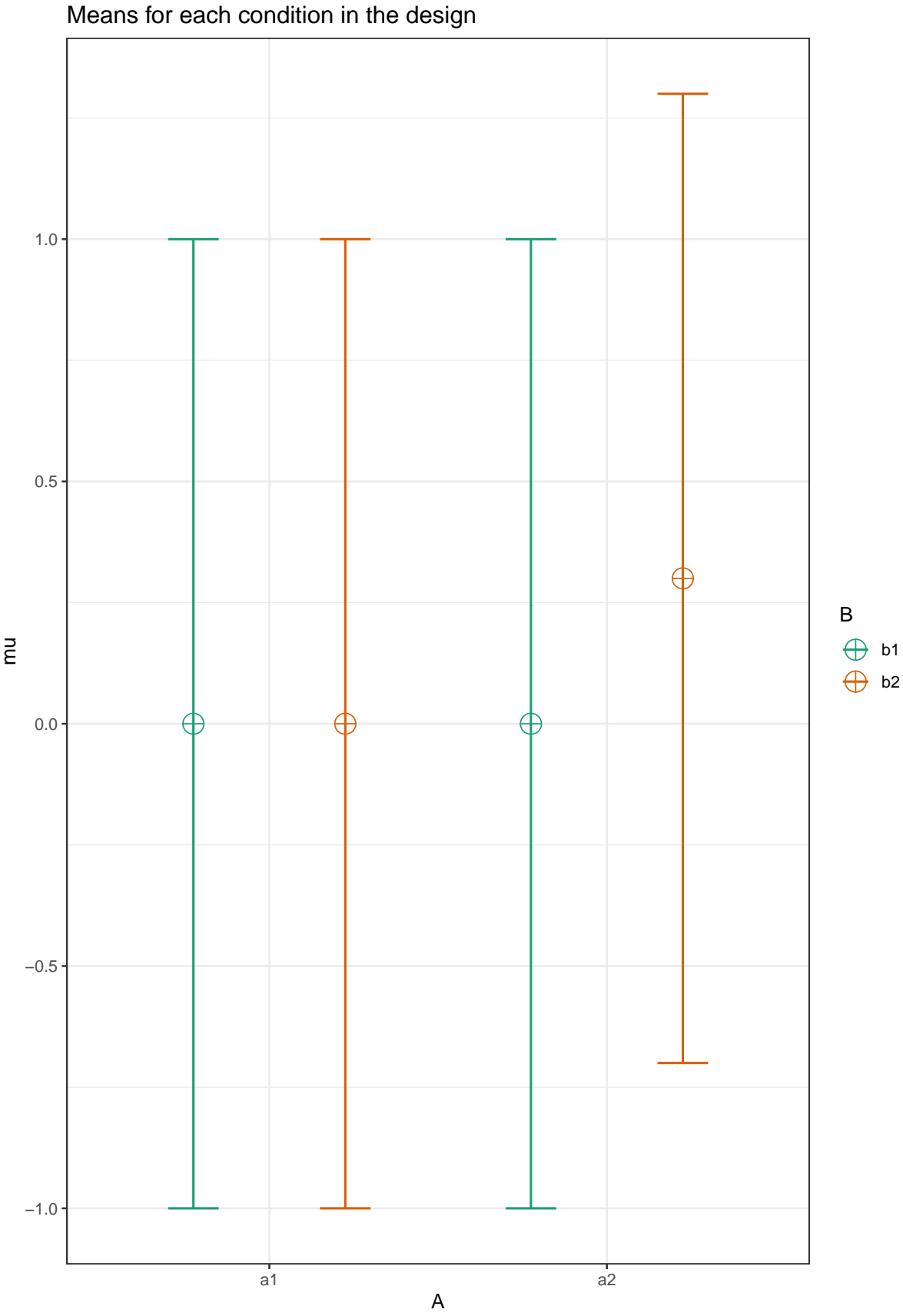
```
string <- "2w*2w"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
design_result <- ANOVA_design(design = string,
                             n = 20,
                             mu = c(0,0,0,0.3),
                             sd = 1,
                             r = 0.0,
                             labelnames = labelnames)
```



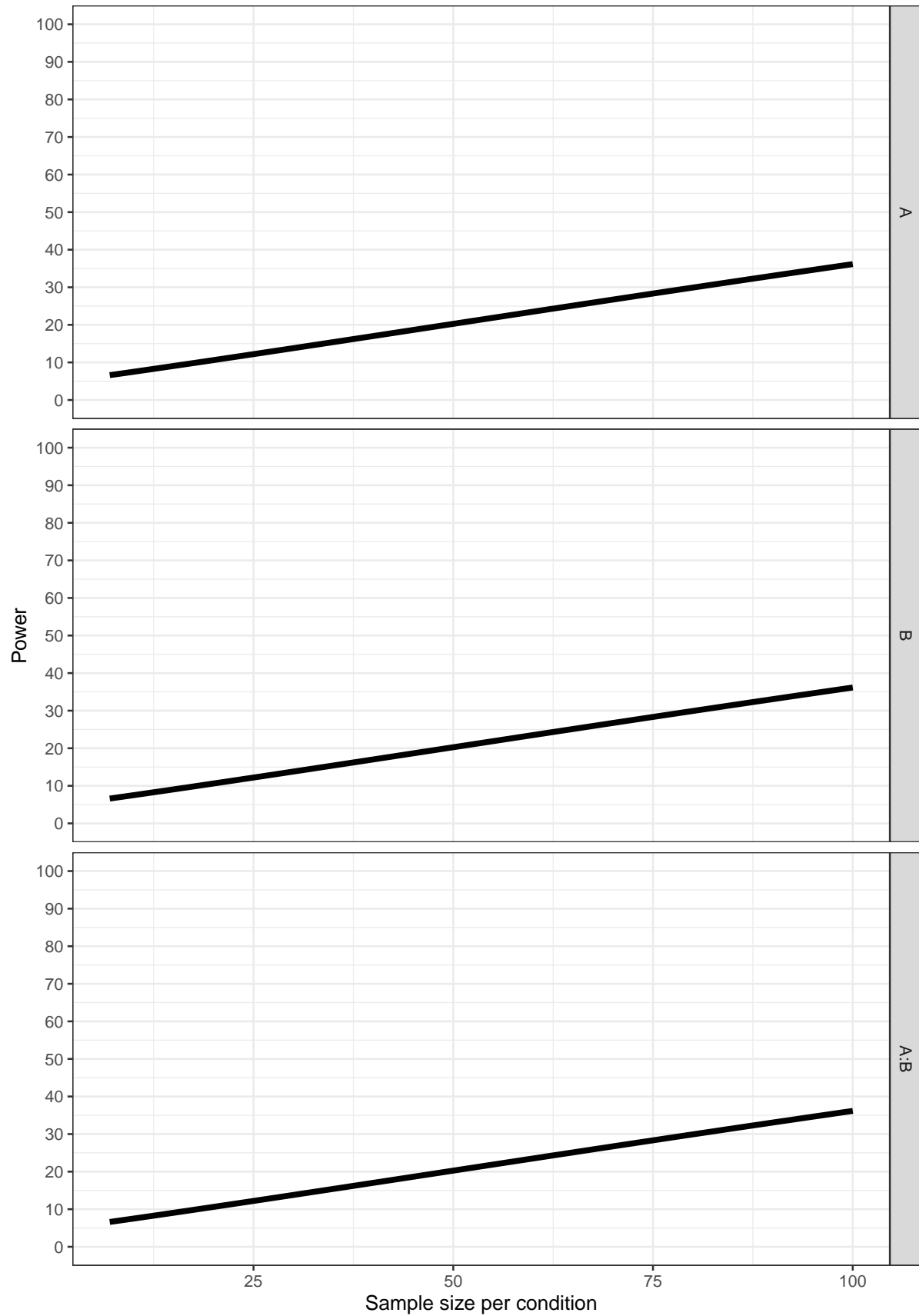

```
p_a <- plot_power(design_result,  
                  max_n = 100)
```



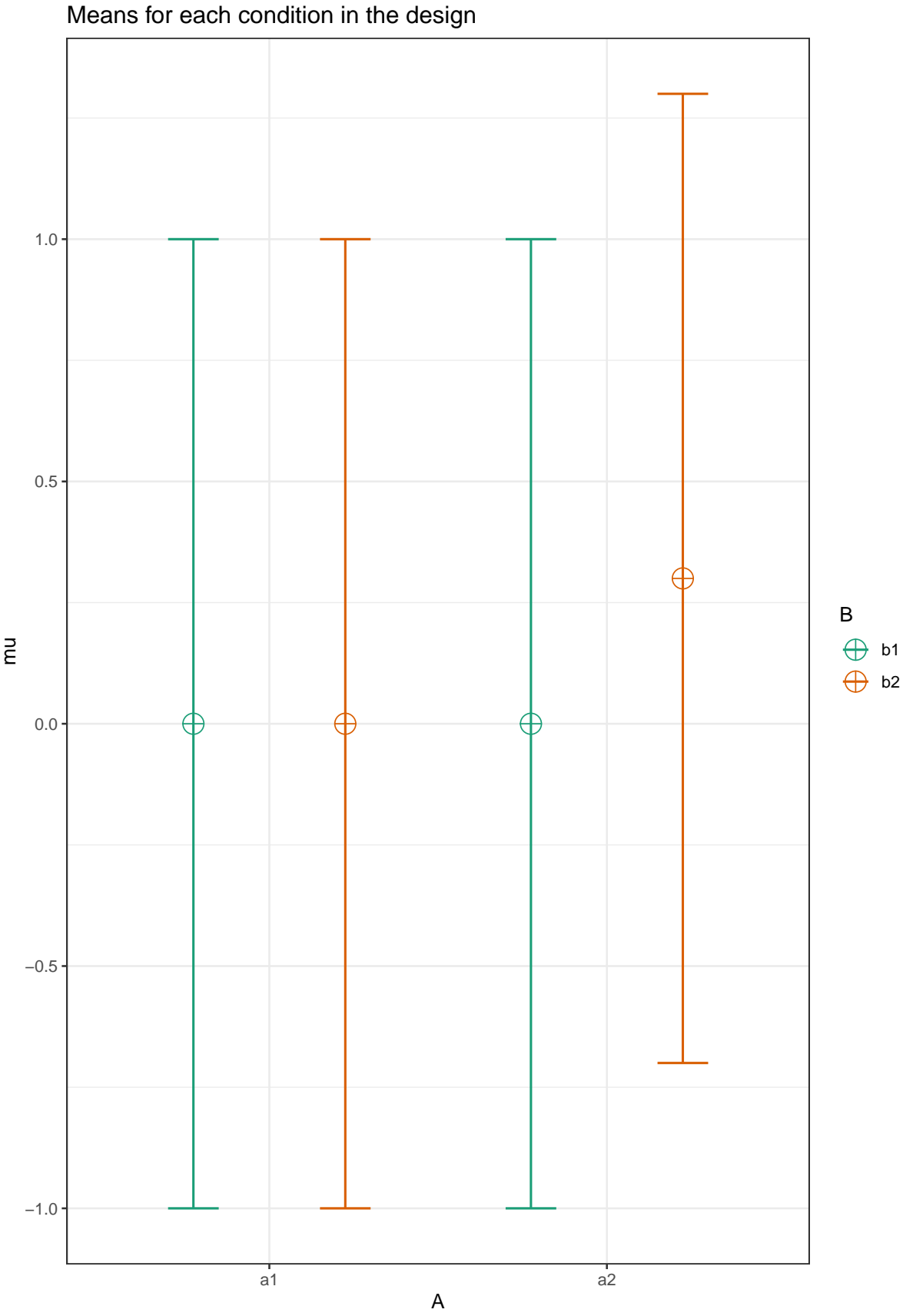
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r = 0.1,  
                             labelnames = labelnames)
```



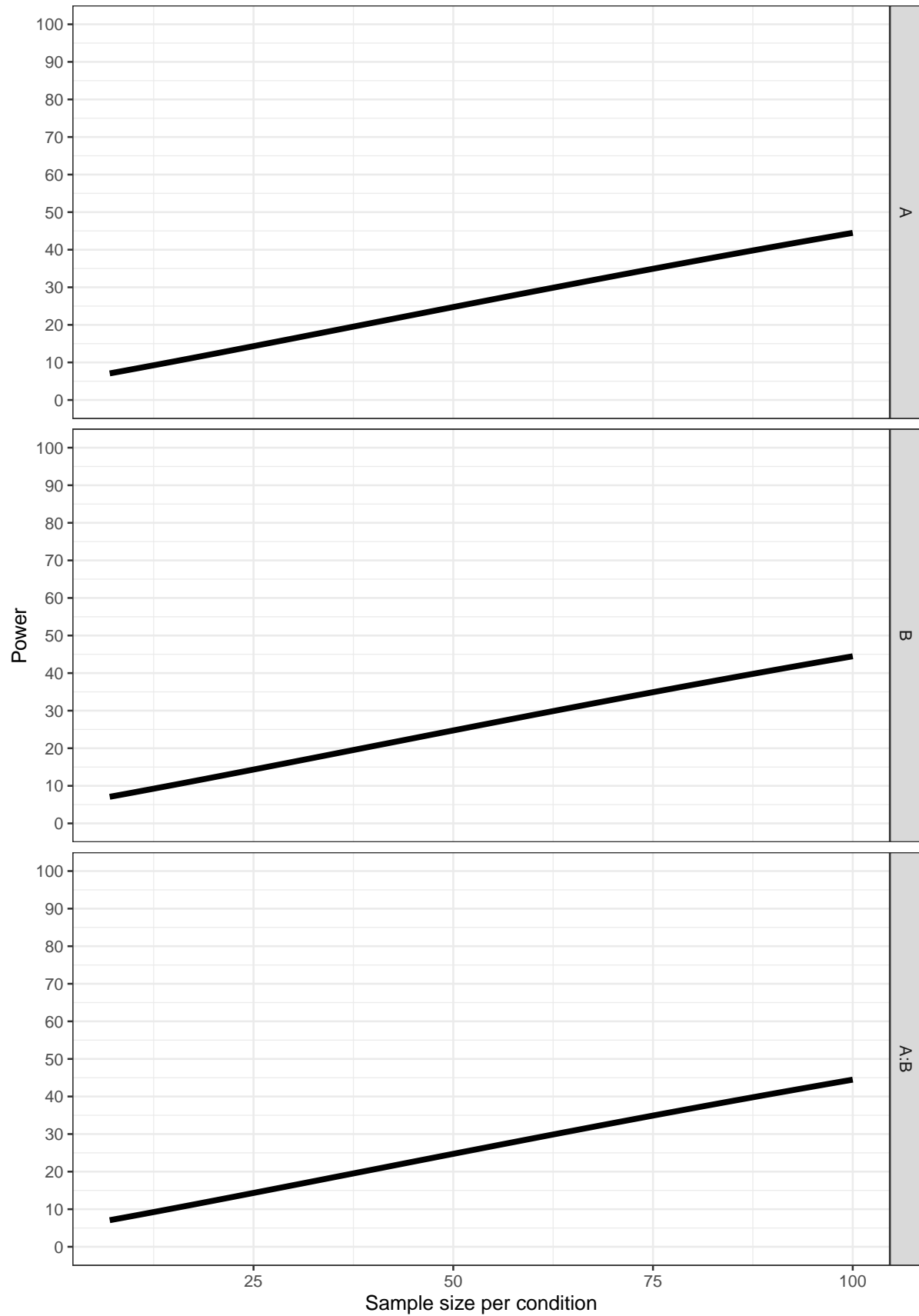
```
p_b <- plot_power(design_result,  
                  max_n = 100)
```



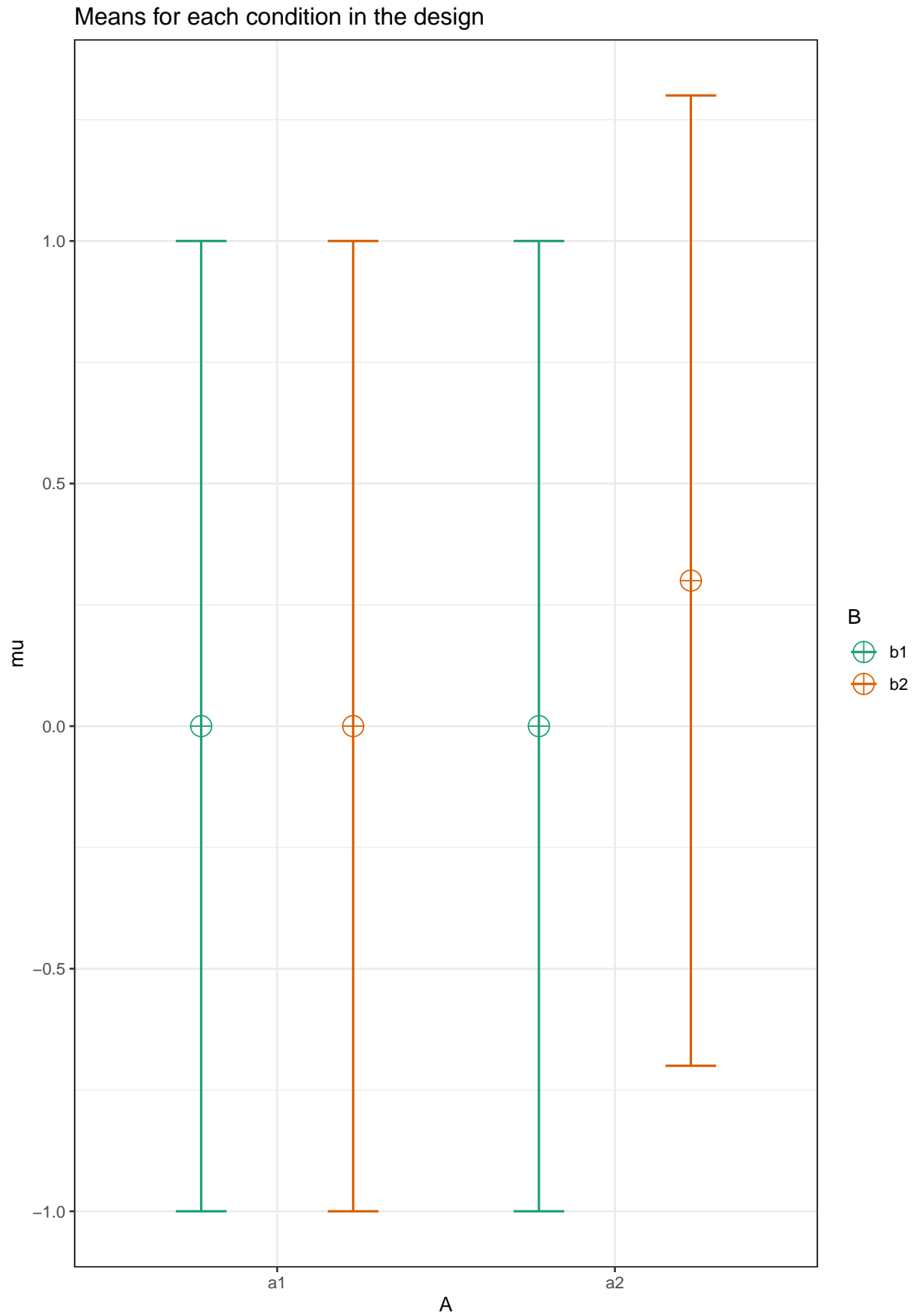
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r = 0.3,  
                             labelnames = labelnames)
```



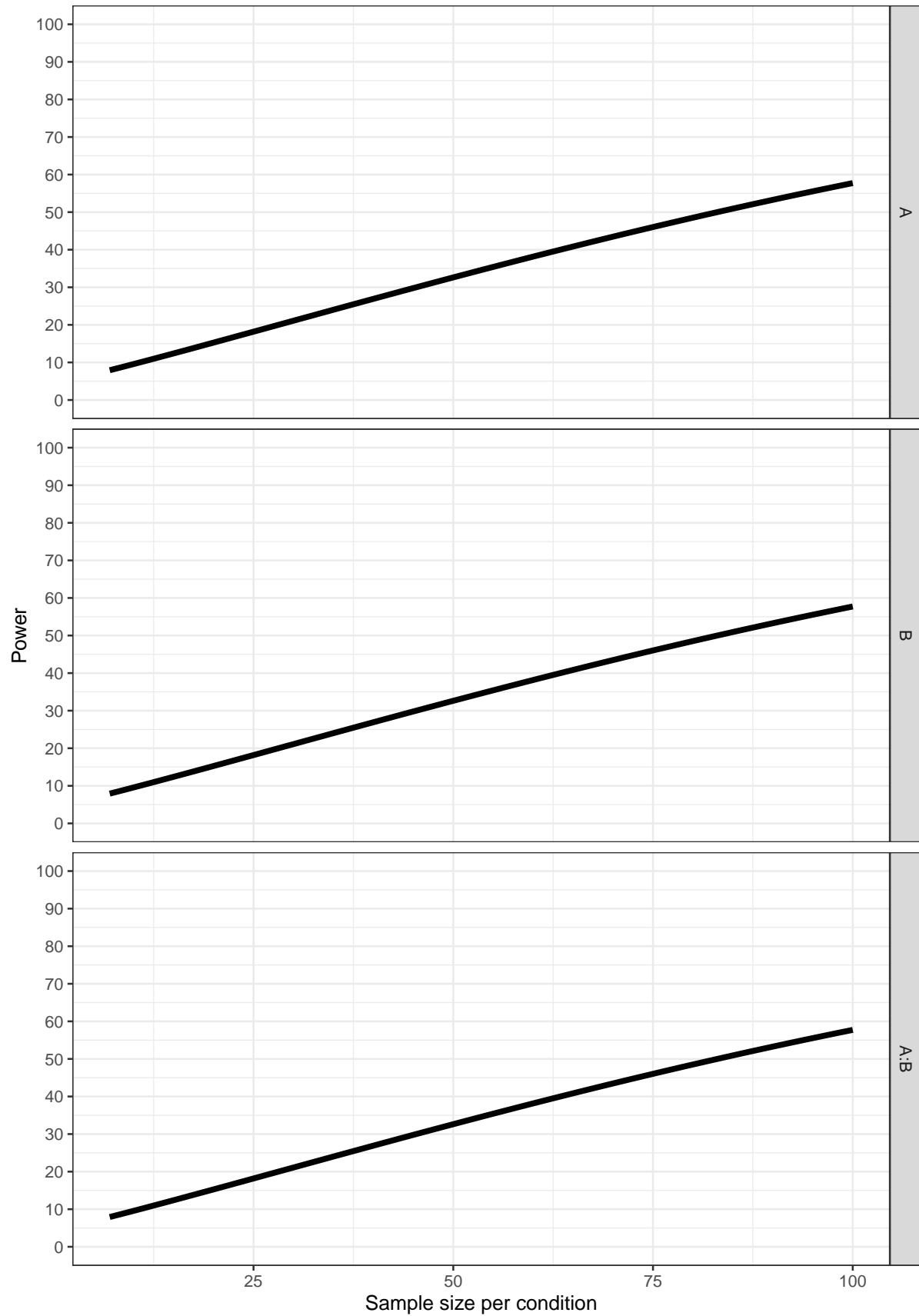

```
p_c <- plot_power(design_result,  
                  max_n = 100)
```



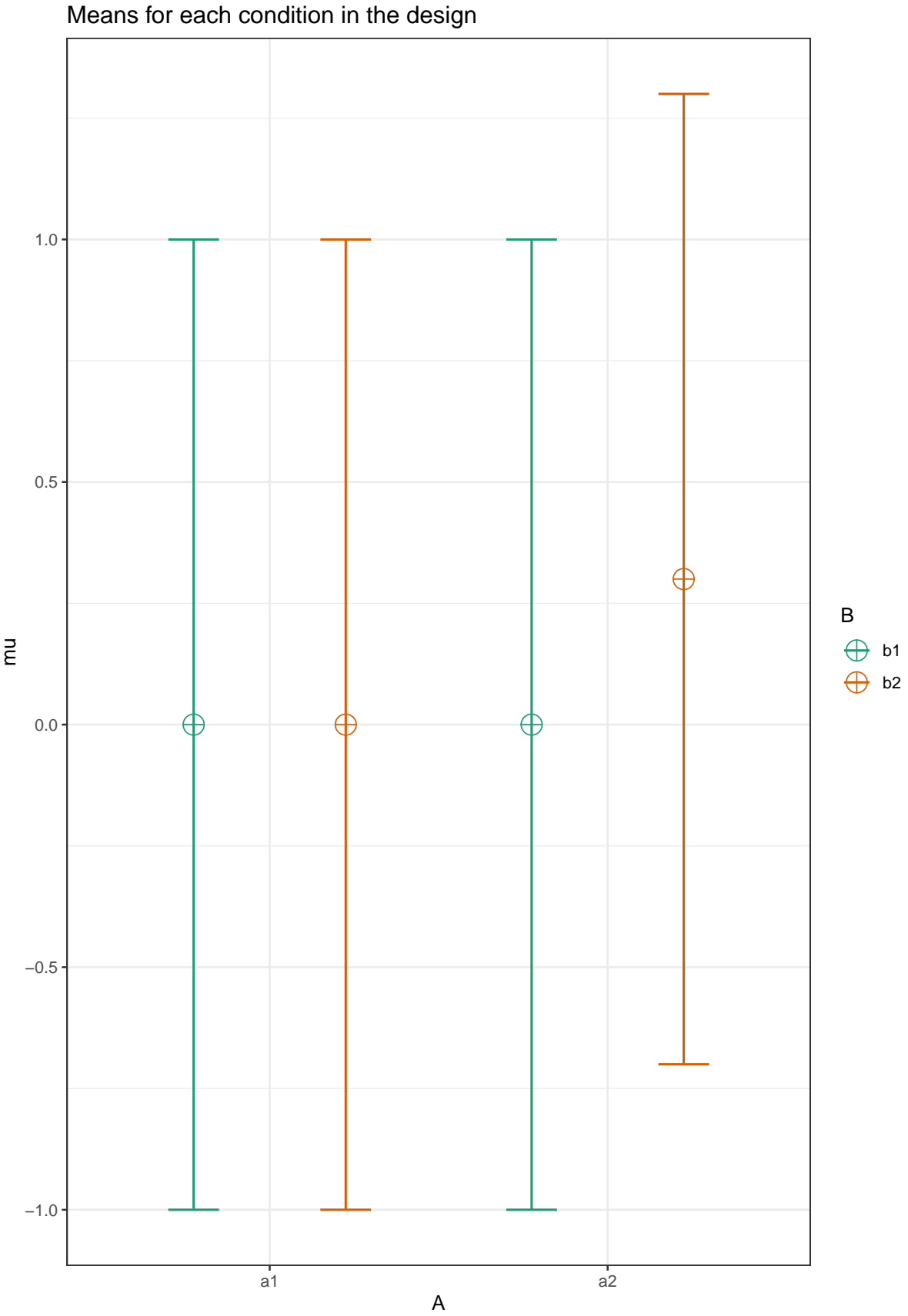
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r = 0.5,  
                             labelnames = labelnames)
```



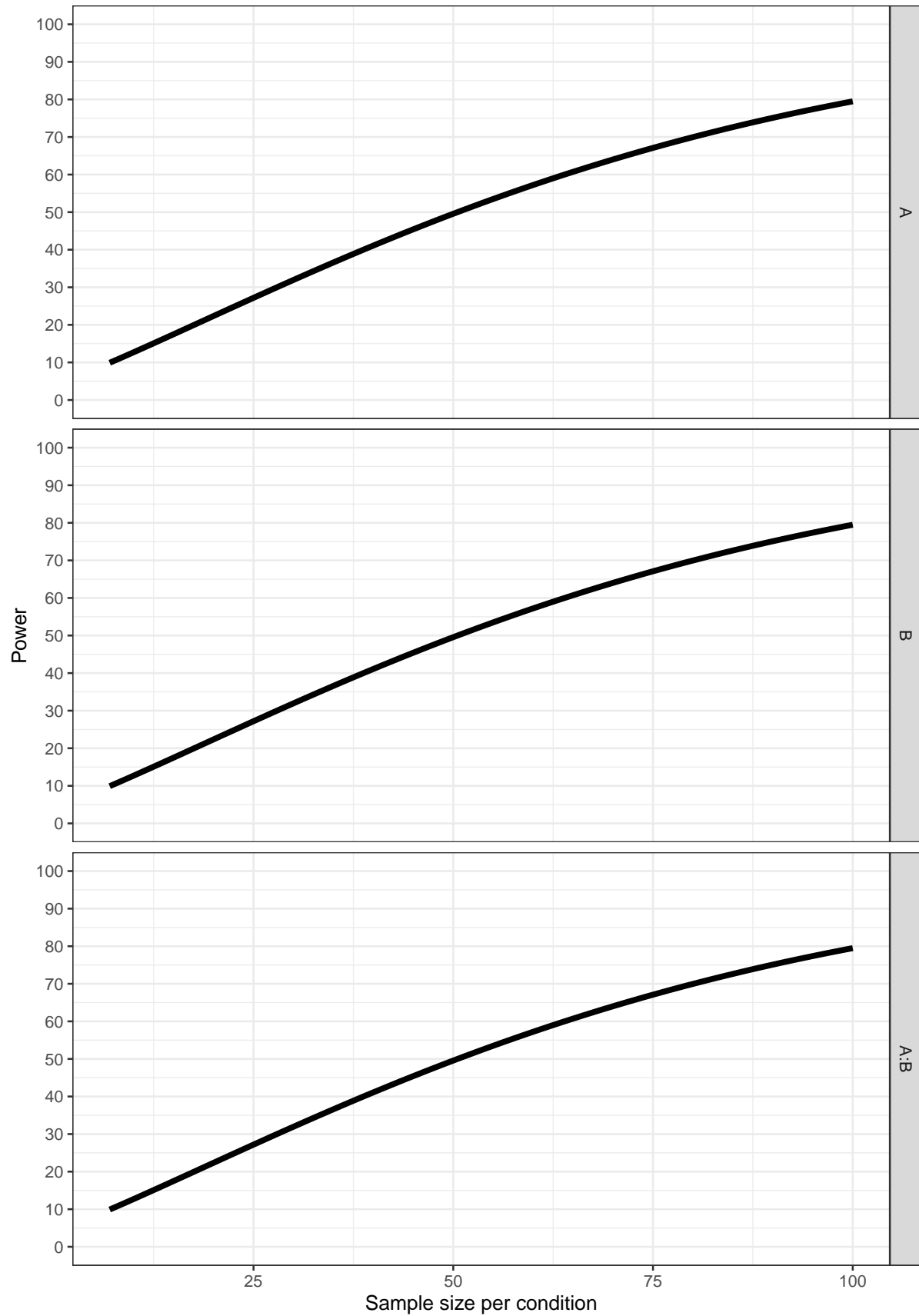
```
p_d <- plot_power(design_result,  
                  max_n = 100)
```



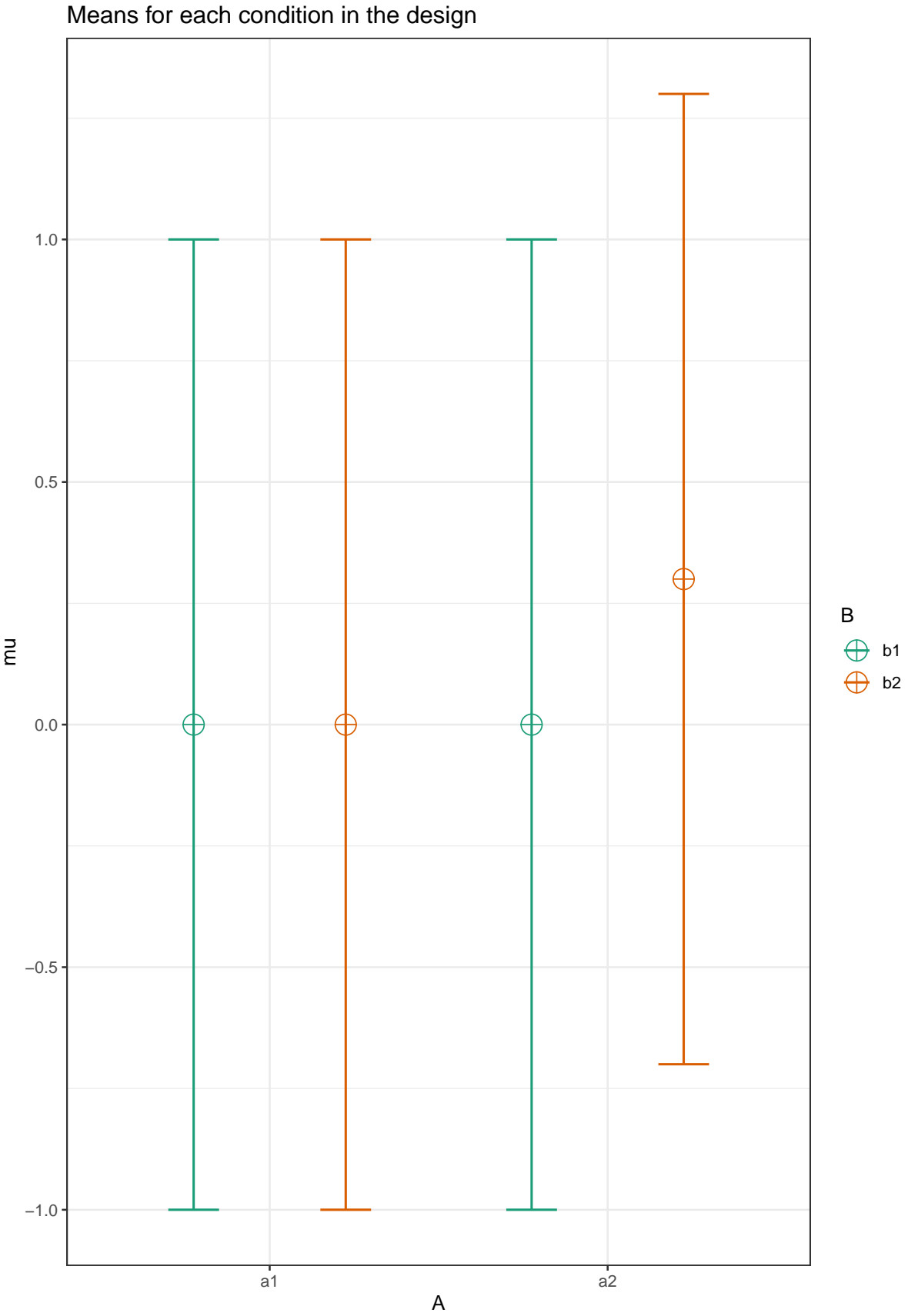
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r = 0.7,  
                             labelnames = labelnames)
```



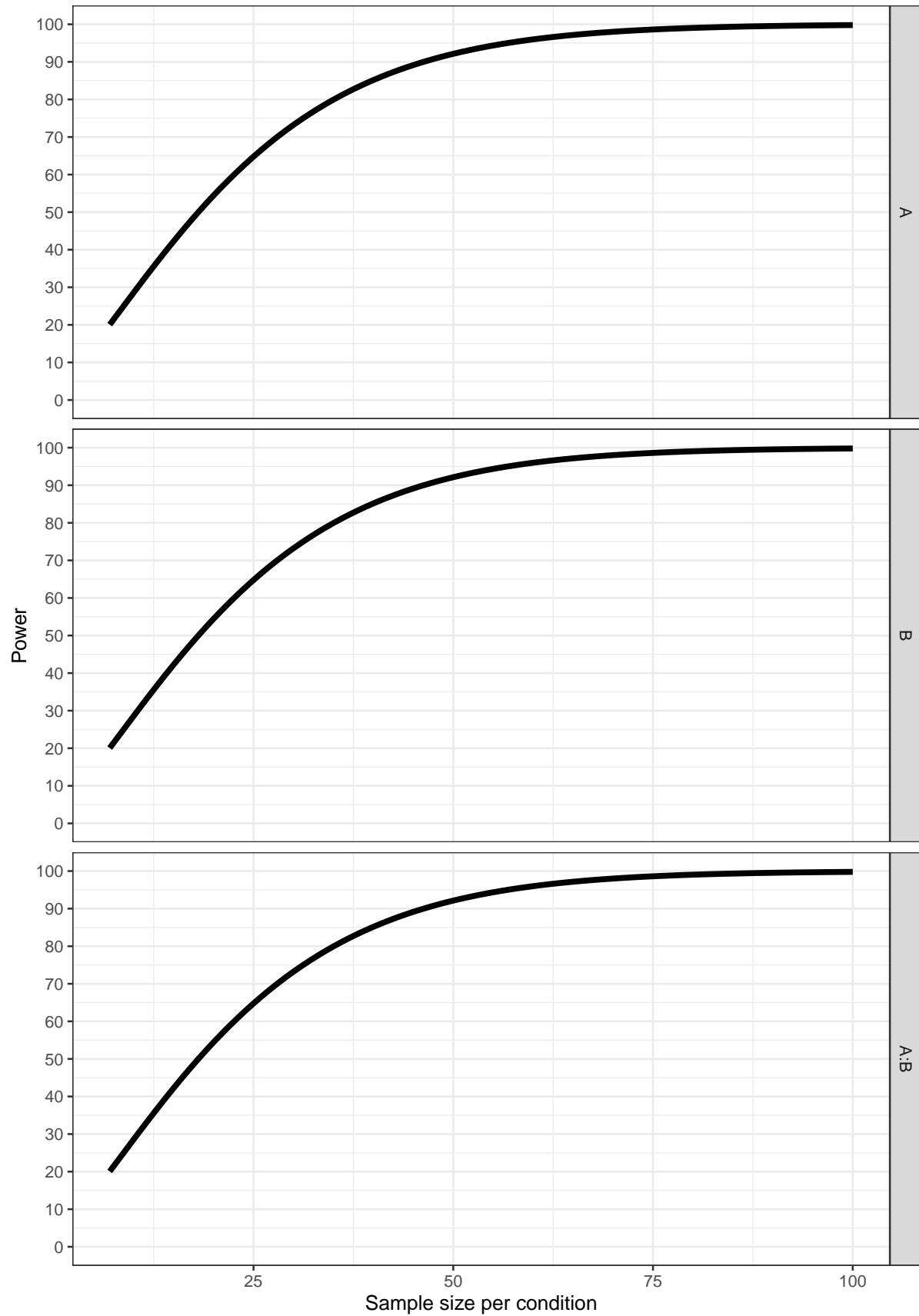

```
p_e <- plot_power(design_result,  
                  max_n = 100)
```



```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r = 0.9,  
                             labelnames = labelnames)
```



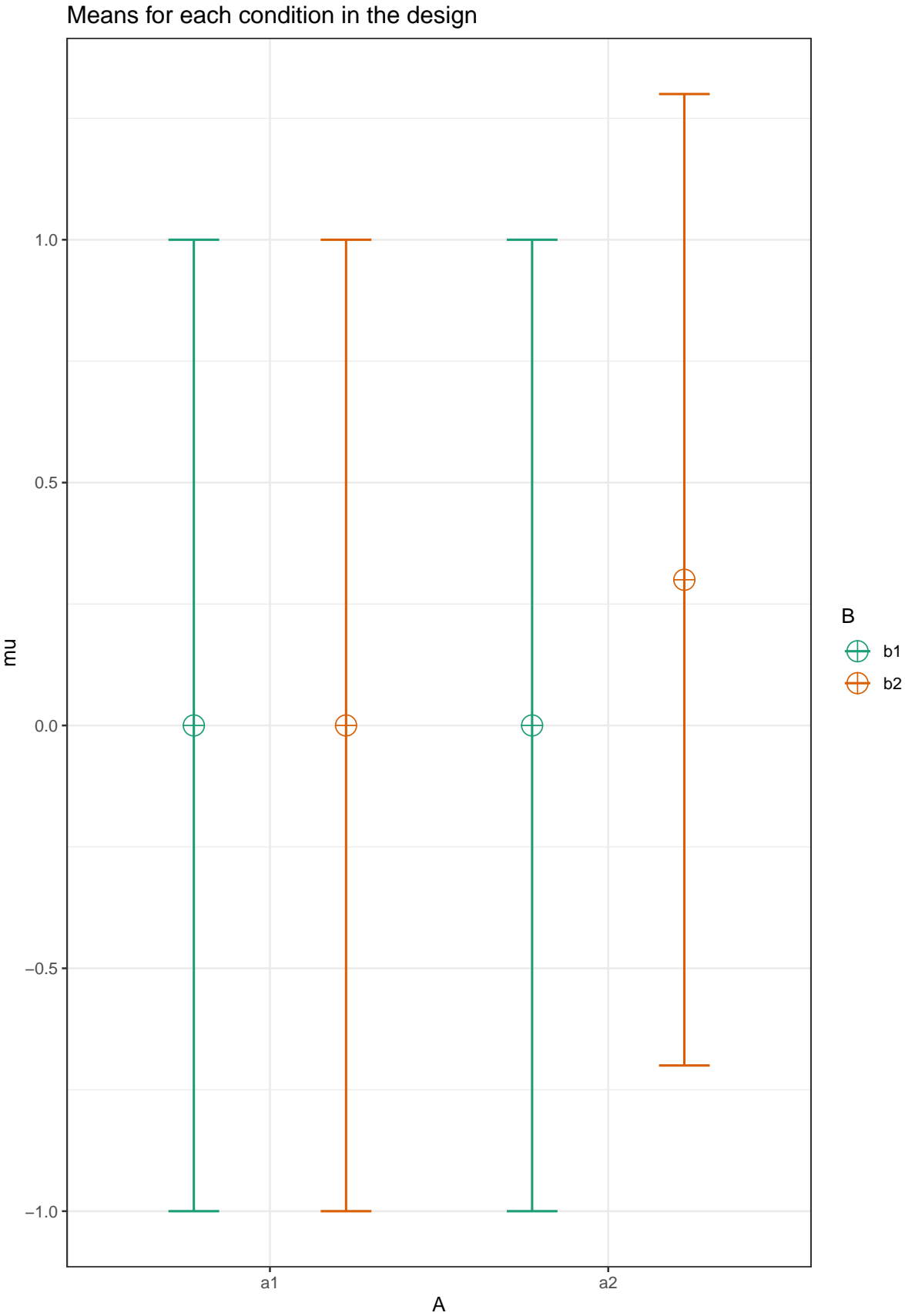
```
p_f <- plot_power(design_result,  
                  max_n = 100)
```



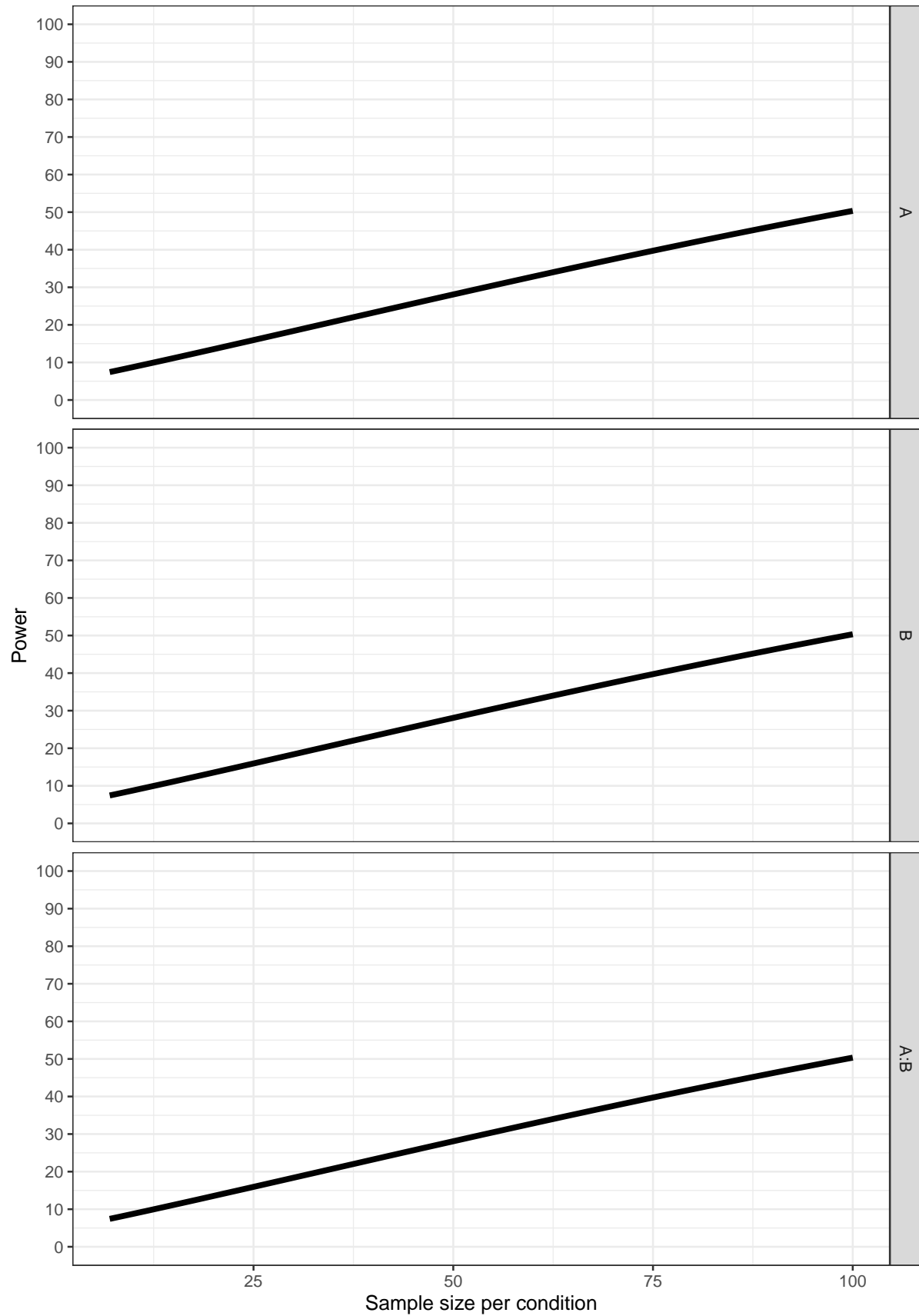
16.3 Increasing correlation in on factor decreases power in second factor

As Potvin and Schutz (2000) write: “The more important finding with respect to the effect of r on power relates to the effect of the correlations associated with one factor on the power of the test of the main effect of the other factor. Specifically, if the correlations among the levels of B are larger than those within the AB matrix (i.e., $r_B - r_{AB} > 0.0$), there is a reduction in the power for the test of the A effect (and the test on B is similarly affected by the A correlations).” We see this in the plots below. As the correlation of the A factor increases from 0.4 to 0.9, we see the power for the main effect of factor B decreases.

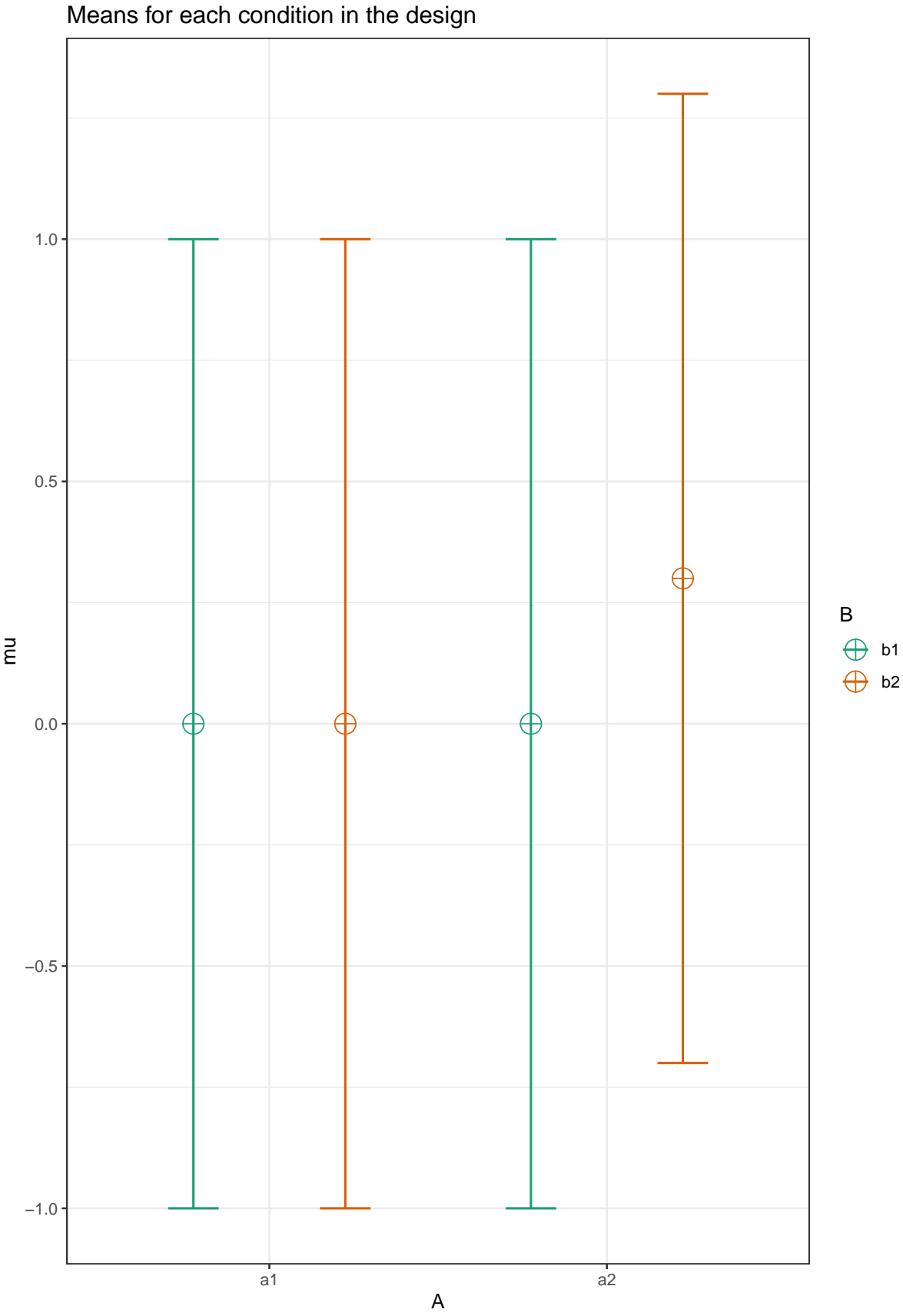
```
string <- "2w*2w"
labelnames = c("A", "a1", "a2", "B", "b1", "b2")
design_result <- ANOVA_design(design = string,
                             n = 20,
                             mu = c(0,0,0,0.3),
                             sd = 1,
                             r <- c(
                               0.4, 0.4, 0.4,
                               0.4, 0.4,
                               0.4),
                             labelnames = labelnames)
```



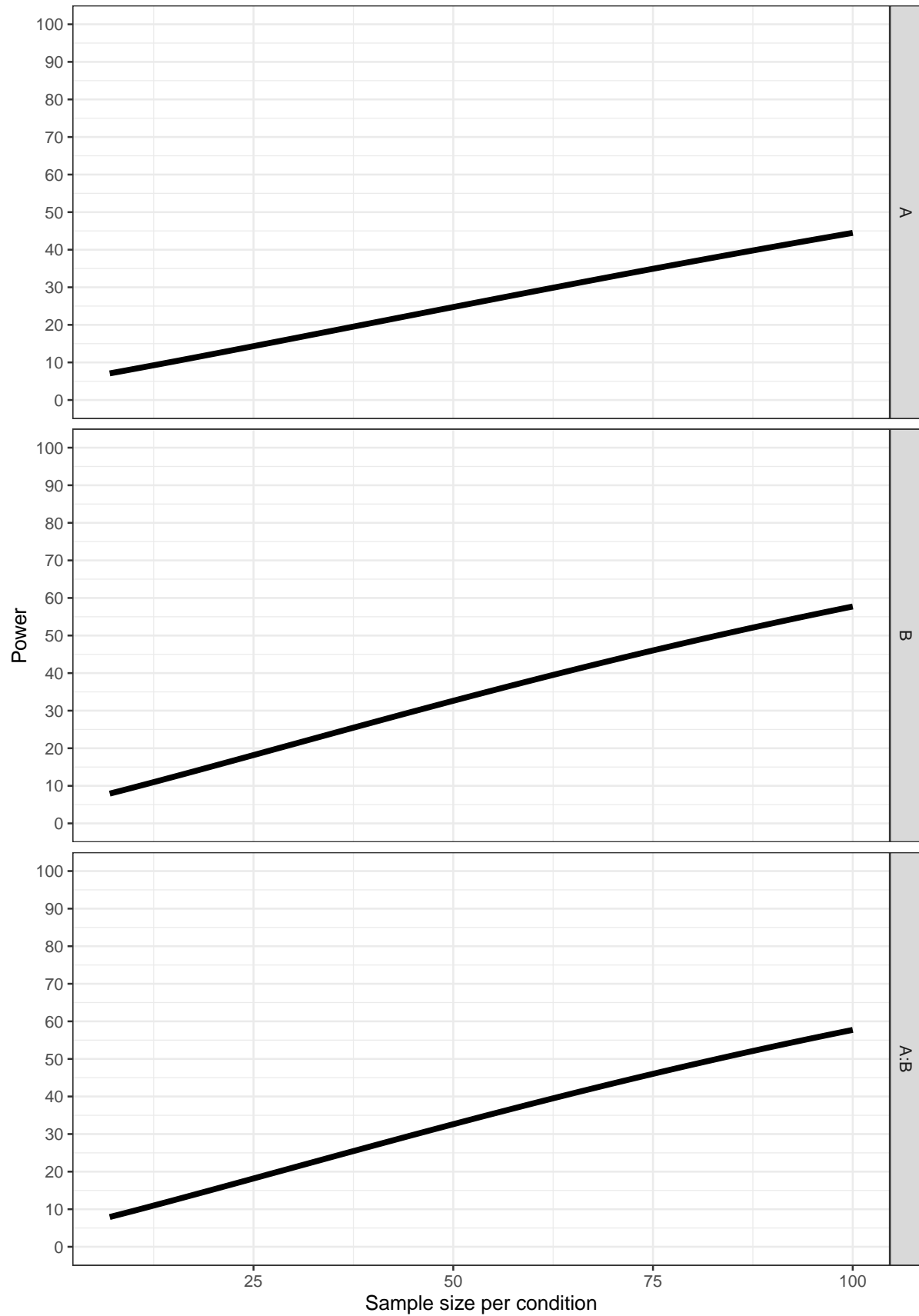

```
p_a <- plot_power(design_result,  
                  max_n = 100)
```



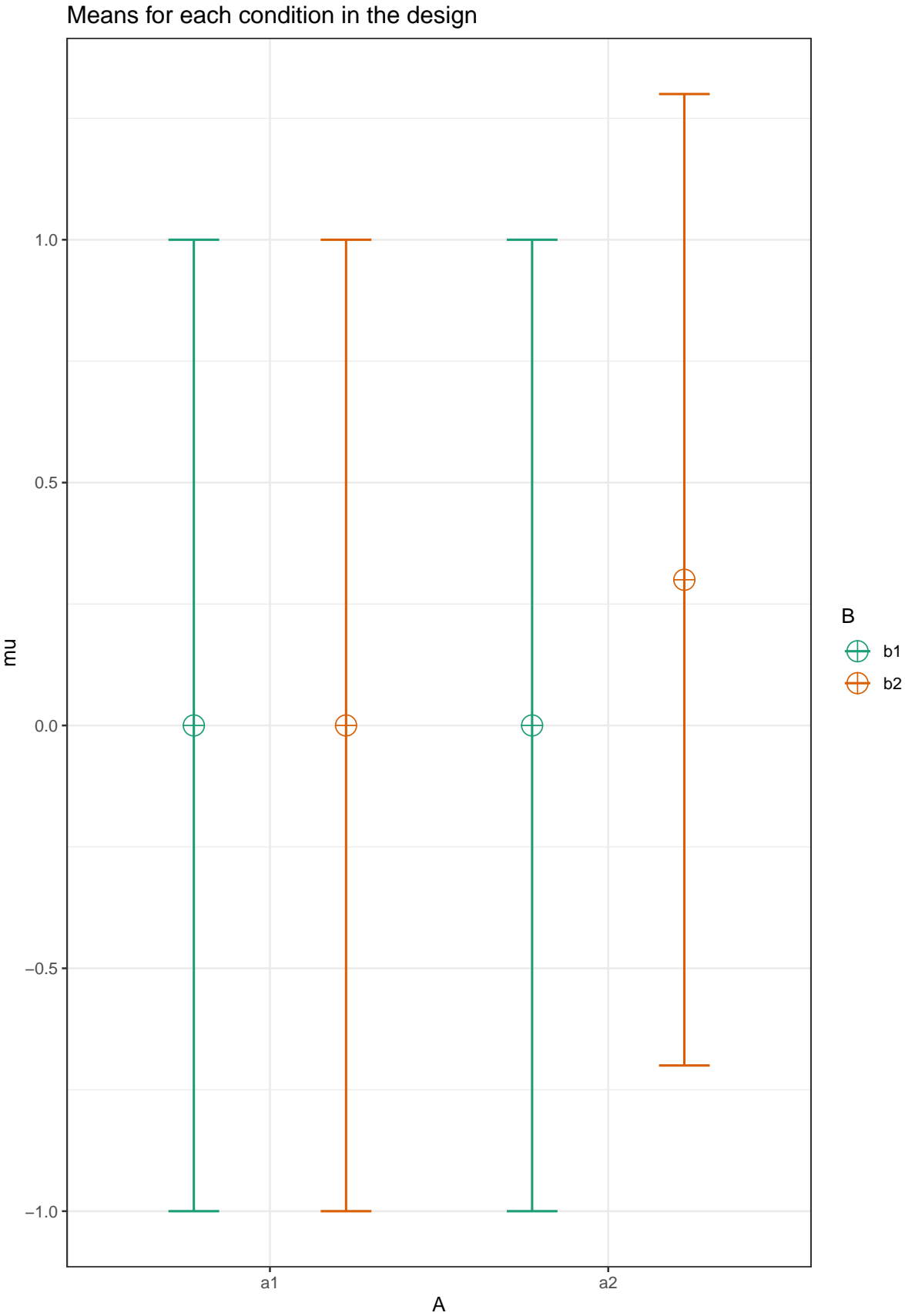
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r <- c(  
                               0.5, 0.4, 0.4,  
                               0.4, 0.4,  
                               0.5),  
                             labelnames = labelnames)
```



```
p_b <- plot_power(design_result,  
                  max_n = 100)
```

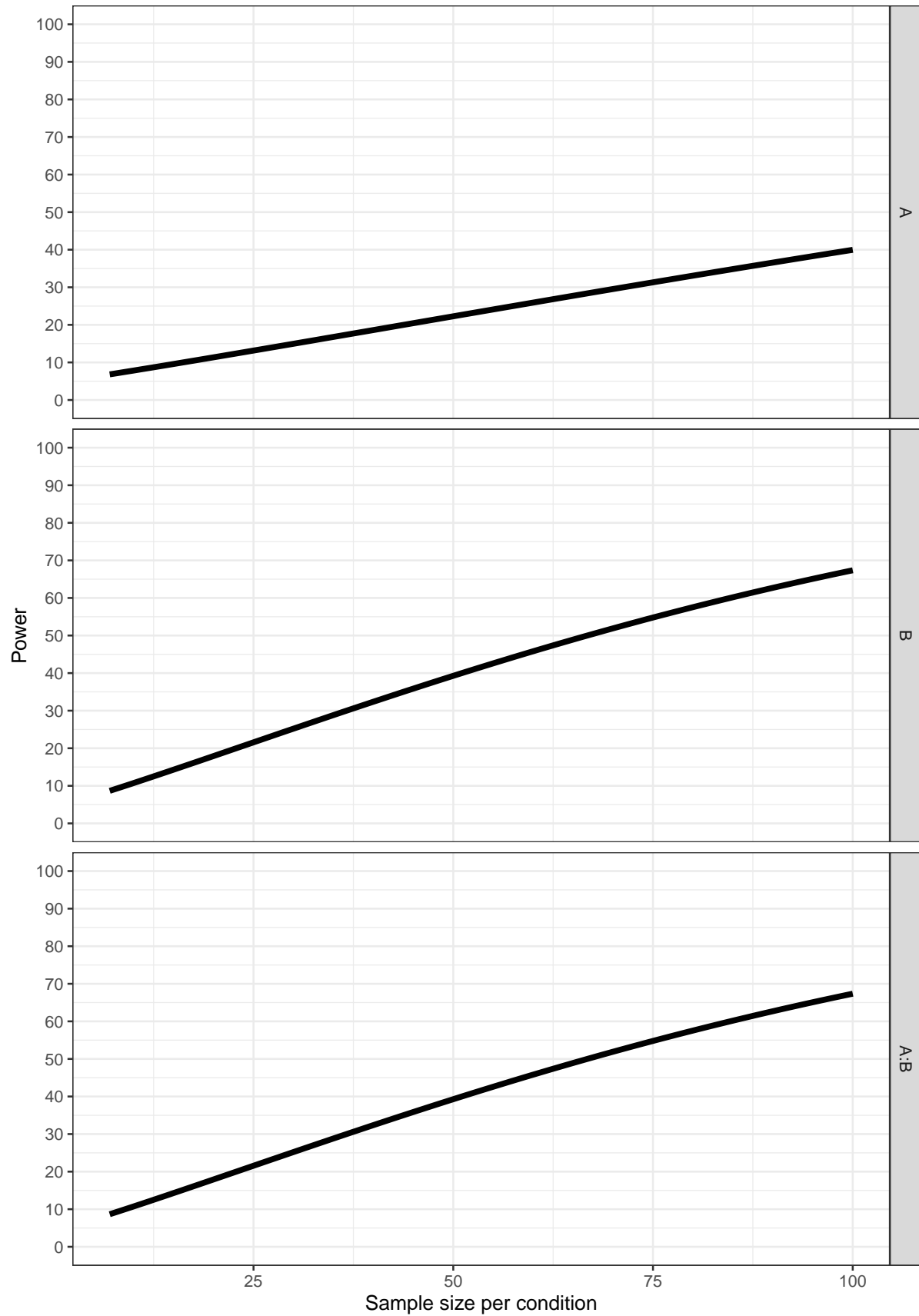


```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r <- c(  
                               0.6, 0.4, 0.4,  
                               0.4, 0.4,  
                               0.6),  
                             labelnames = labelnames)
```

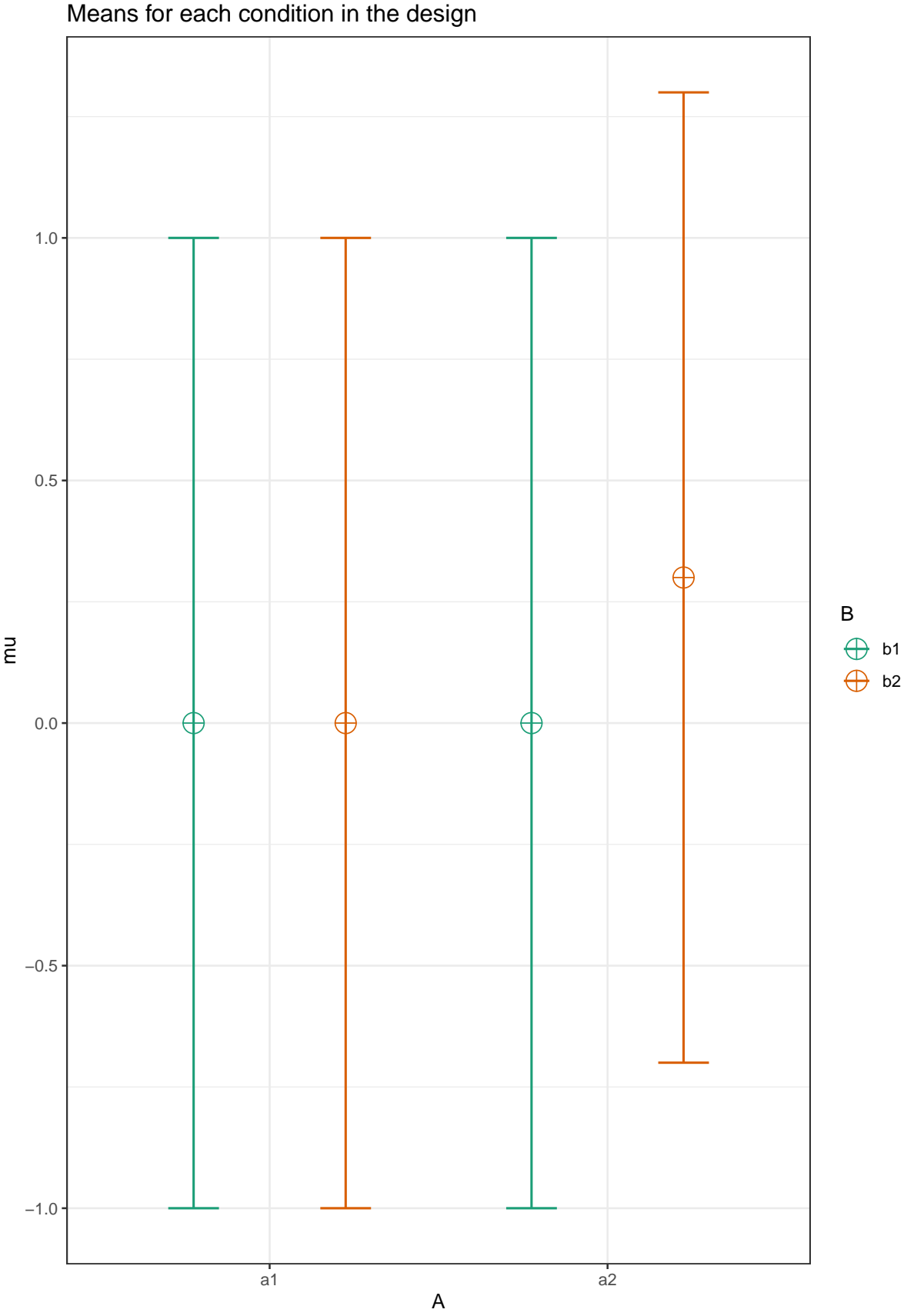


16.3. INCREASING CORRELATION IN ON FACTOR DECREASES POWER IN SECOND FACTOR²⁰¹

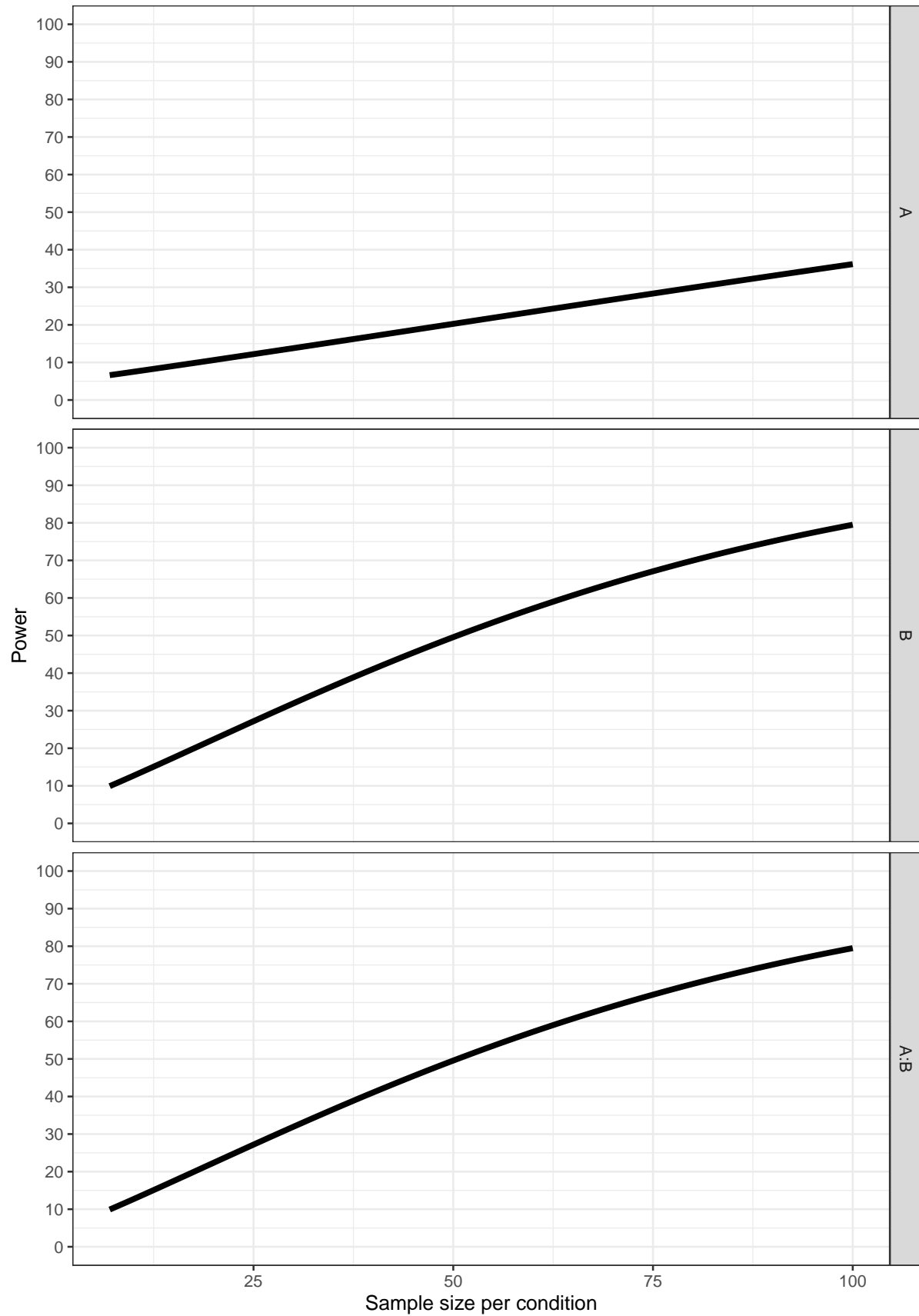
```
p_c <- plot_power(design_result,  
                  max_n = 100)
```



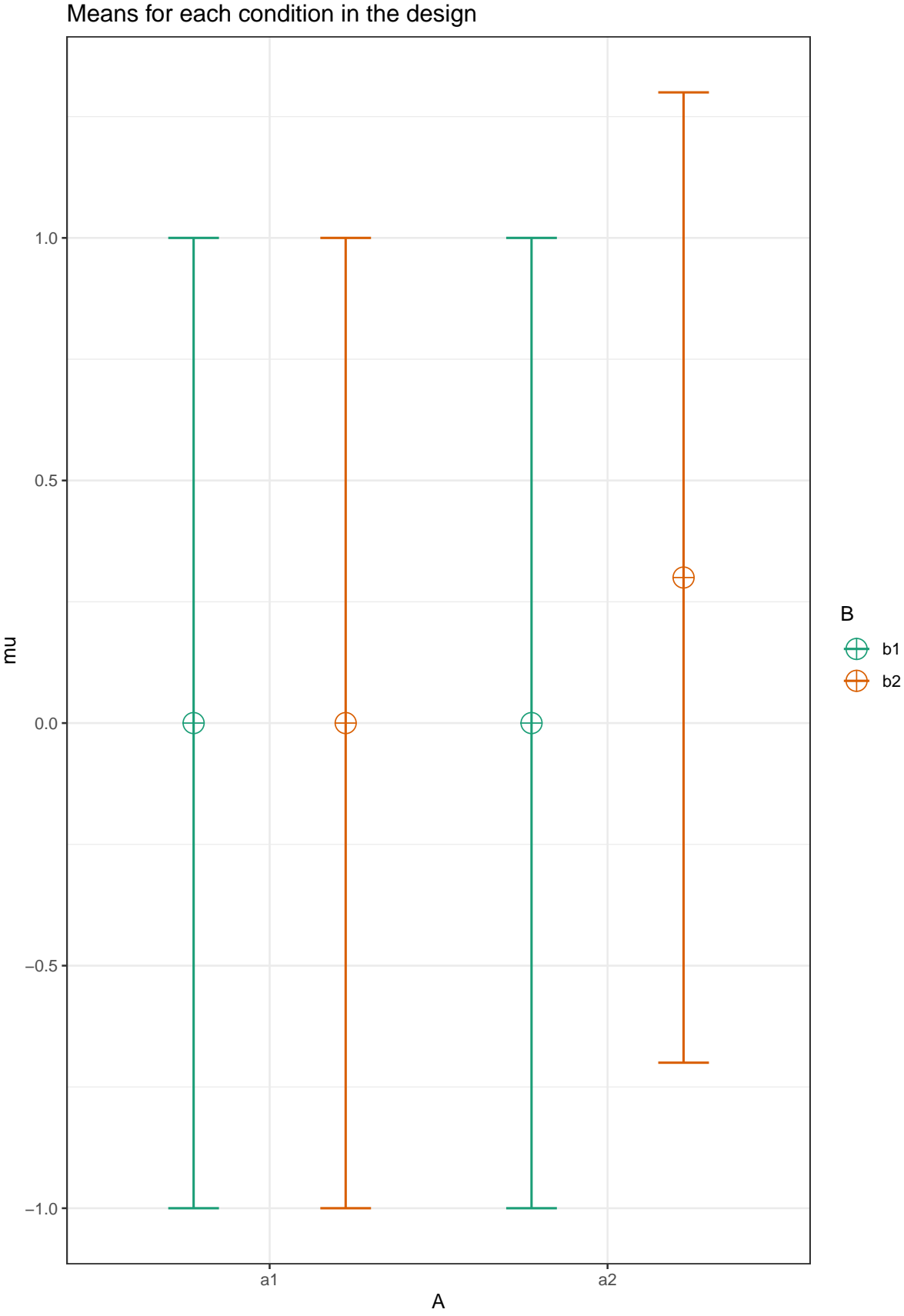
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r <- c(  
                               0.7, 0.4, 0.4,  
                               0.4, 0.4,  
                               0.7),  
                             labelnames = labelnames)
```



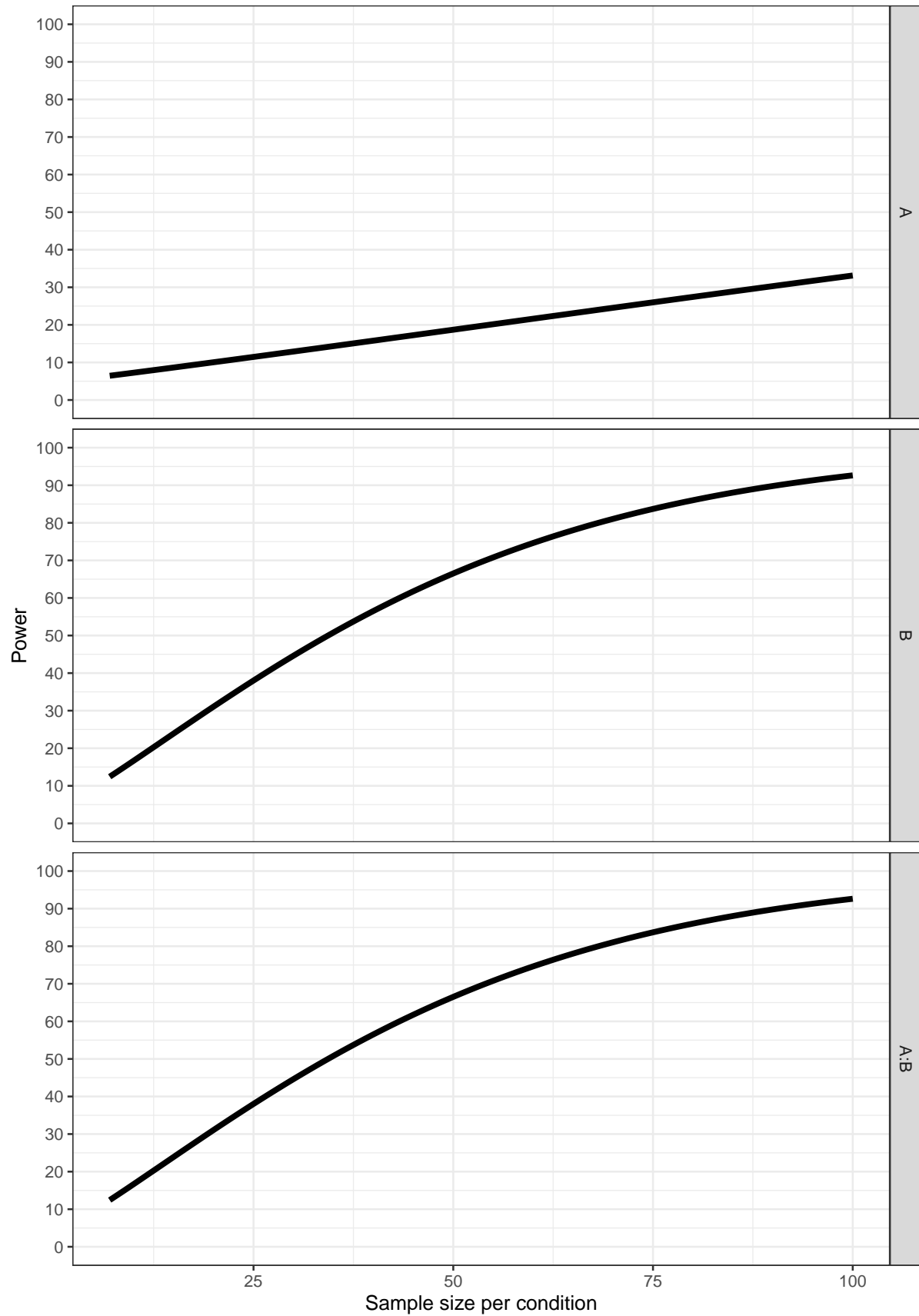
```
p_d <- plot_power(design_result,  
                  max_n = 100)
```



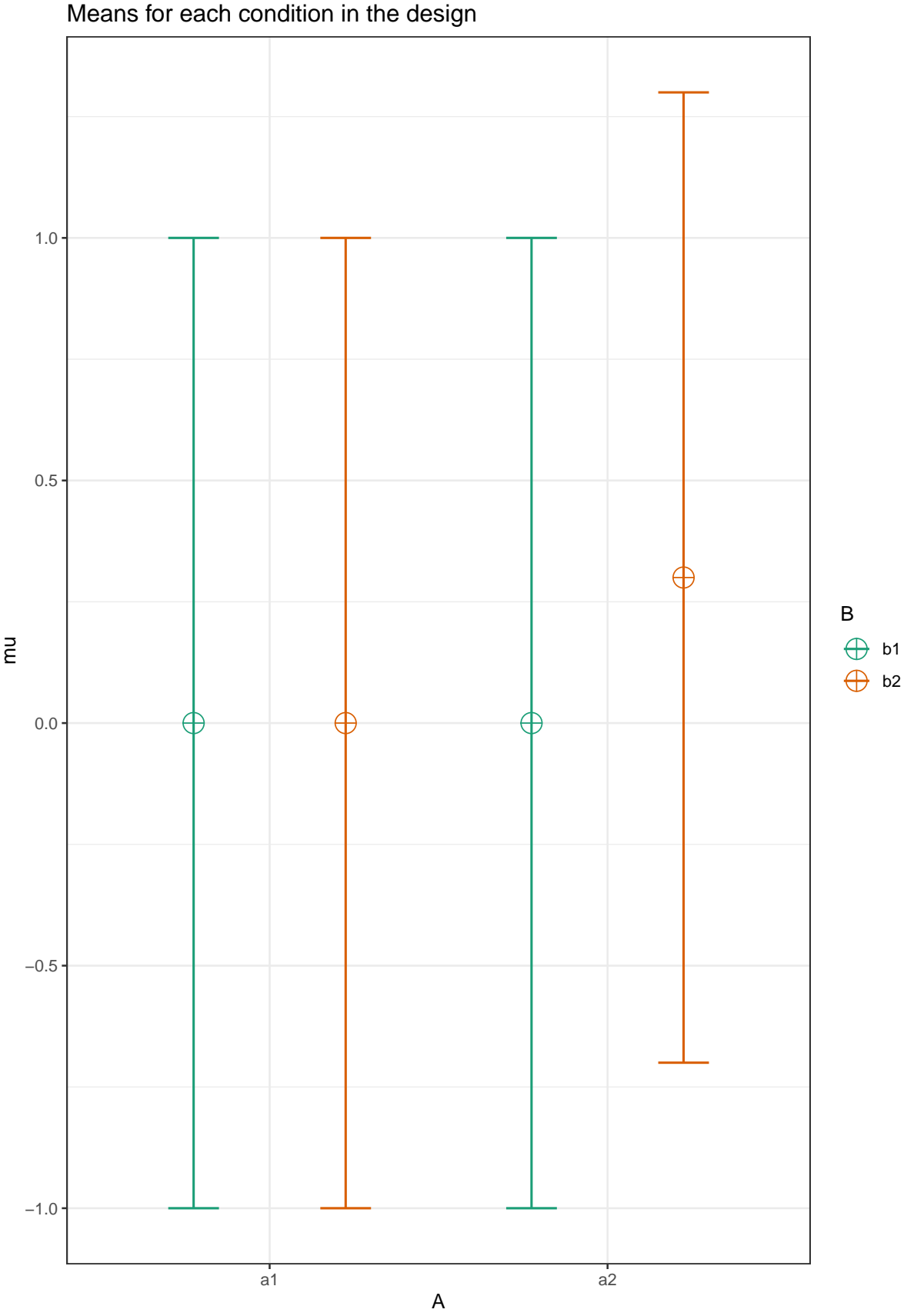
```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r <- c(  
                               0.8, 0.4, 0.4,  
                               0.4, 0.4,  
                               0.8),  
                             labelnames = labelnames)
```




```
p_e <- plot_power(design_result,  
                  max_n = 100)
```



```
design_result <- ANOVA_design(design = string,  
                             n = 20,  
                             mu = c(0,0,0,0.3),  
                             sd = 1,  
                             r <- c(  
                               0.9, 0.4, 0.4,  
                               0.4, 0.4,  
                               0.9),  
                             labelnames = labelnames)
```



16.3. INCREASING CORRELATION IN ON FACTOR DECREASES POWER IN SECOND FACTOR²¹³

```
p_f <- plot_power(design_result,  
                  max_n = 100)
```

