pwr_calc

```
library(ggplot2)
library(lme4)
```

```
## Indlæser krævet pakke: Matrix
```

```
#Load sample data
d <- read.table('pilot_means.csv', header = T, sep = ',')
d$rhythm <- factor(d$rhythm, levels = c('iso','low','med','high'))
d$harmony <- factor(d$harmony, levels = c('H','M'))
```

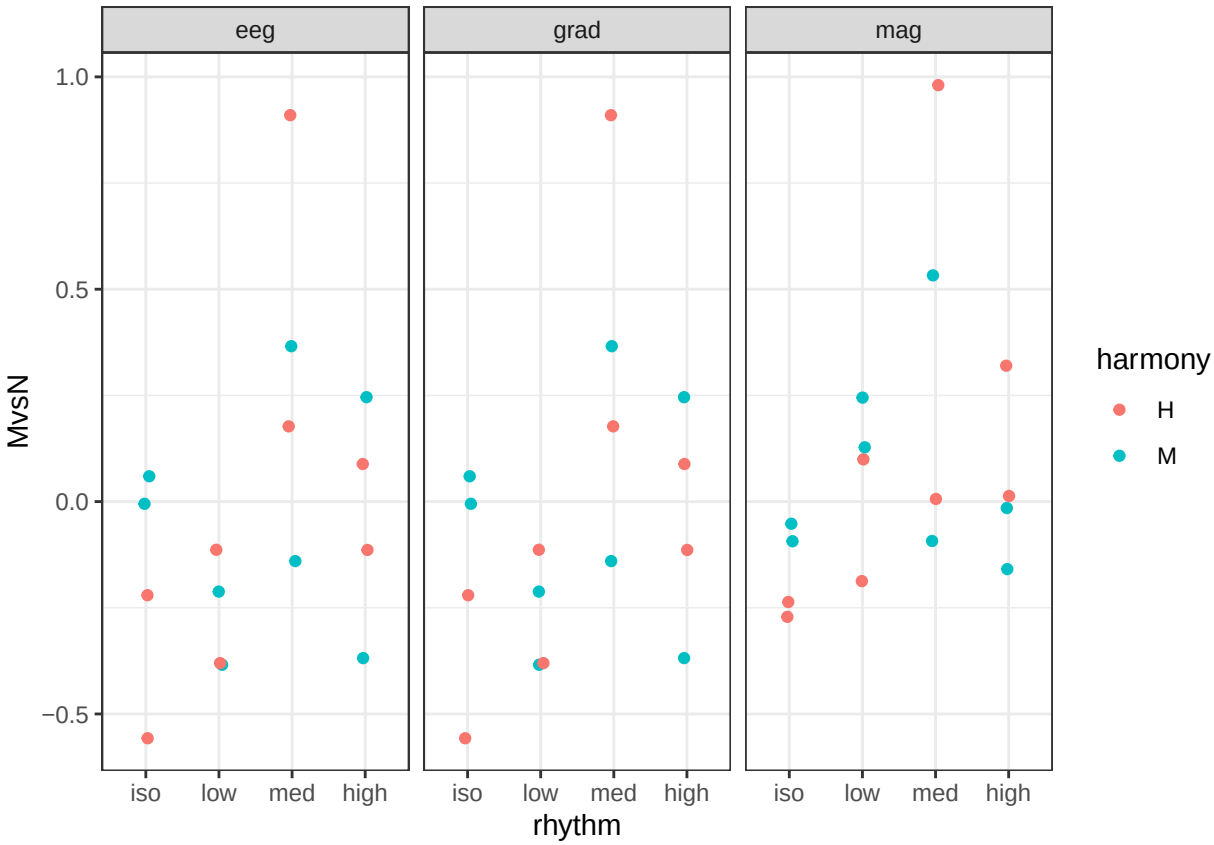
```
d$Mentrain <- d$MeterAmp - d$soundMeterAmp
d$Nentrain <- d$noMeterAmp - d$soundNoMeterAmp
d$MvsN <- d$Mentrain - d$Nentrain
sd(d$Mentrain)
```

```
## [1] 0.2182459
```

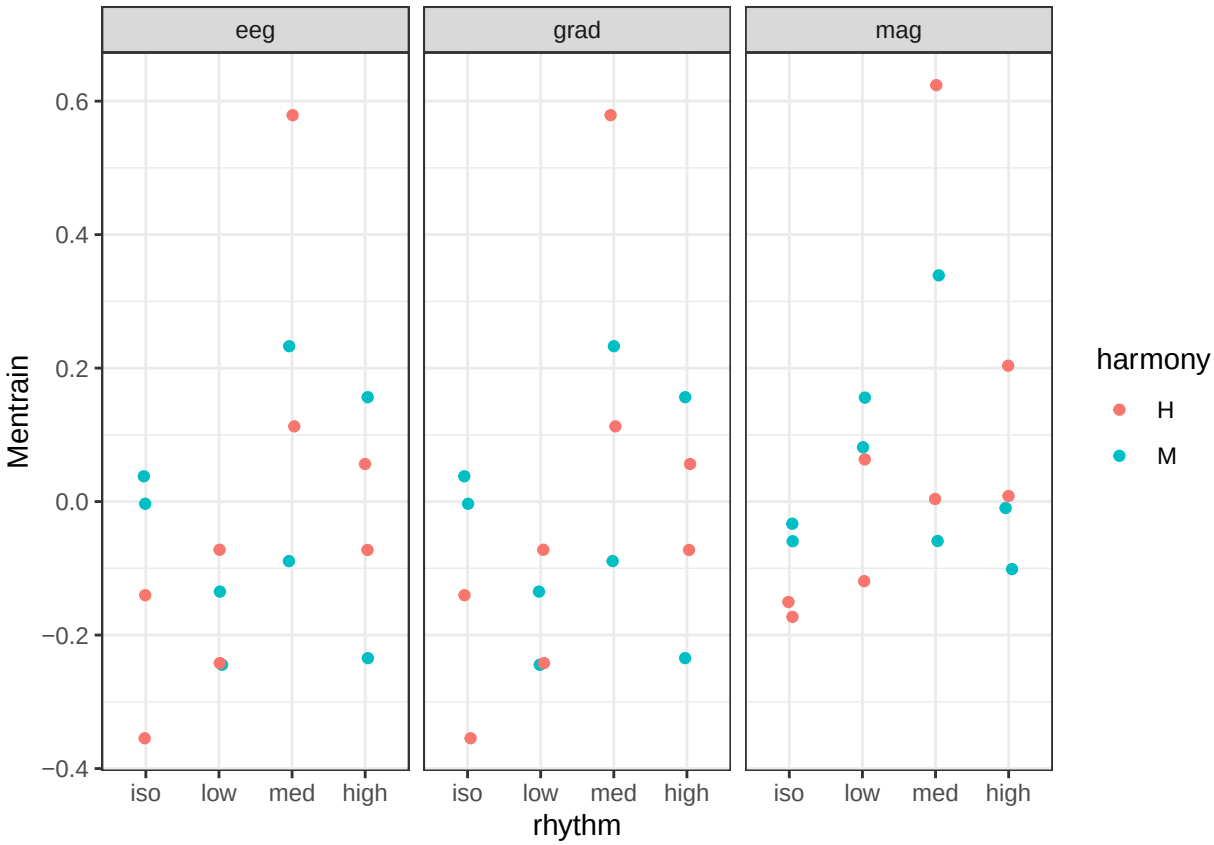
```
sd(d$MvsN)
```

```
## [1] 0.3429578
```

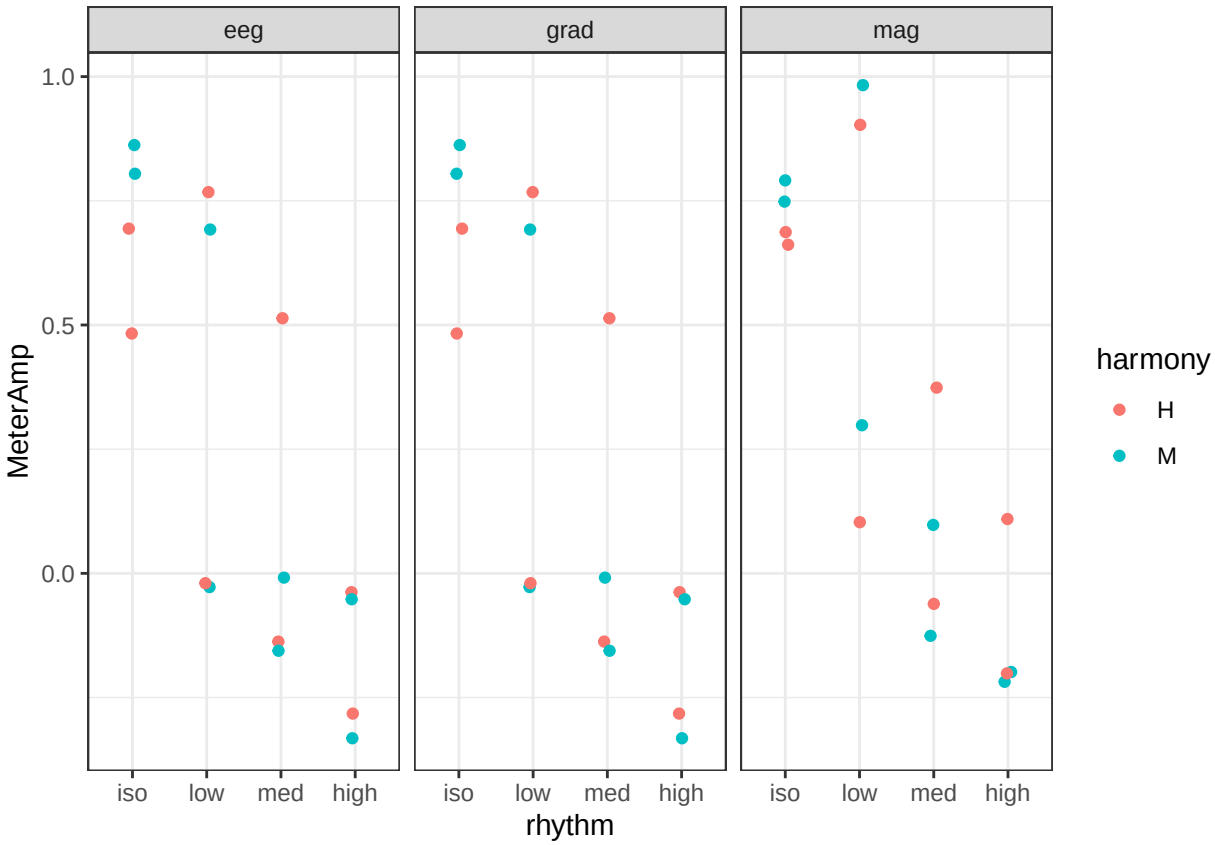
```
ggplot(d,aes(x = rhythm, y = MvsN, color = harmony)) +
  geom_jitter(width = 0.05, height = 0) +
  theme_bw() +
  facet_wrap(~channel)
```



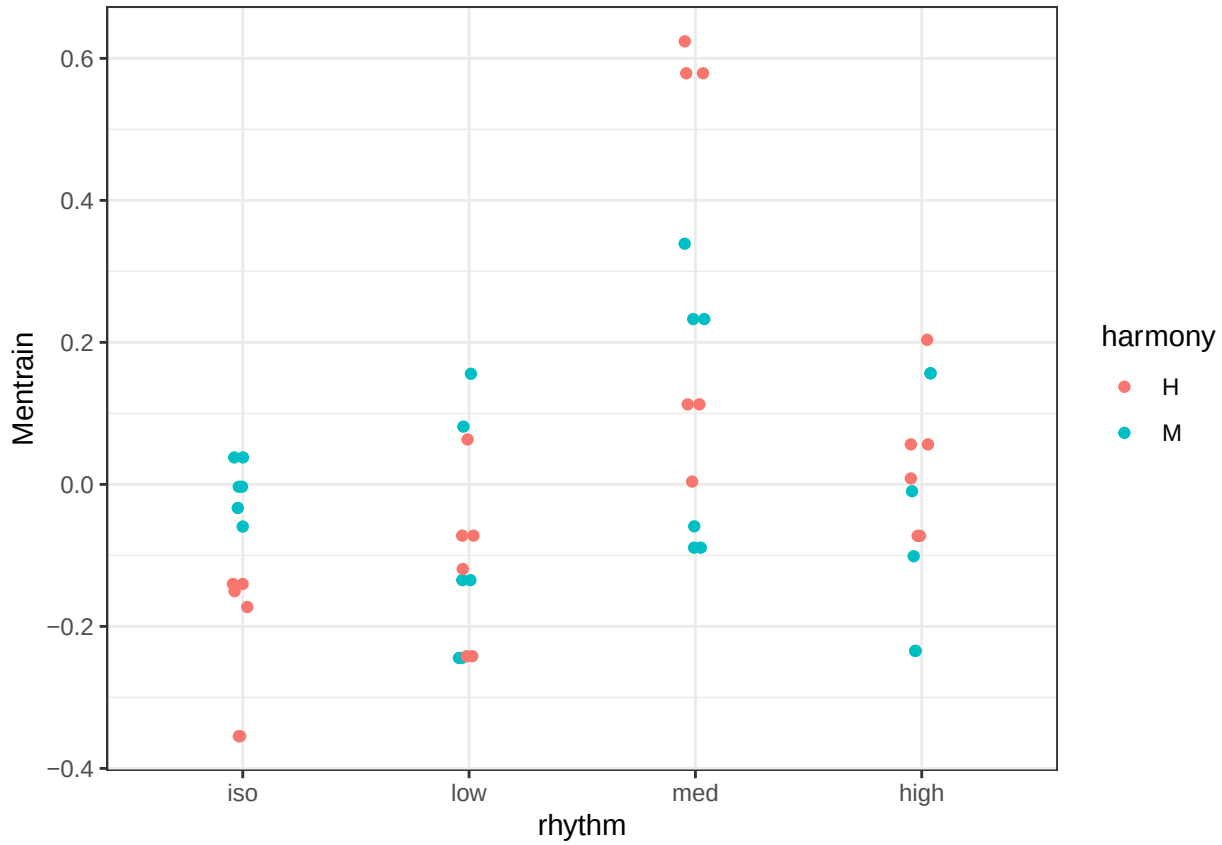

```
ggplot(d,aes(x = rhythm, y = Mentrain, color = harmony)) +
  geom_jitter(width = 0.05, height = 0) +
  theme_bw() +
  facet_wrap(~channel)
```




```
ggplot(d,aes(x = rhythm, y = MeterAmp, color = harmony)) +
  geom_jitter(width = 0.05, height = 0) +
  theme_bw() +
  facet_wrap(~channel)
```




```
ggplot(d,aes(x = rhythm, y = Mentrain, color = harmony)) +
  geom_jitter(width = 0.05, height = 0) +
  theme_bw()
```




```
m1 <- lm(d$Mentrain~rhythm*harmony*channel,data = d); summary(m1)
```

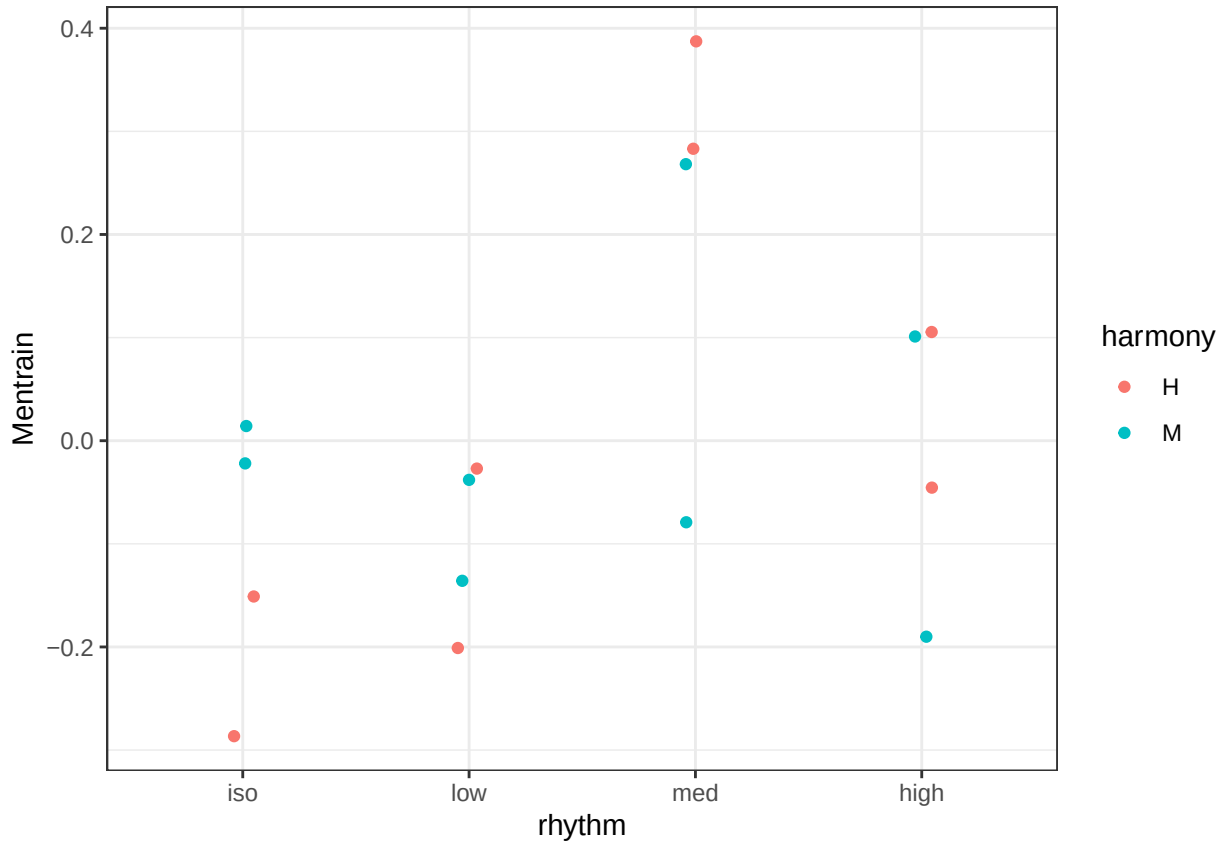
```
##
## Call:
## lm(formula = d$Mentrain ~ rhythm * harmony * channel, data = d)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.31000 -0.08648  0.00000  0.08648  0.31000
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -2.474e-01  1.364e-01  -1.813  0.08230 .
## rhythmlow                      9.034e-02  1.930e-01   0.468  0.64386
## rhythmmed                      5.933e-01  1.930e-01   3.075  0.00519 **
## rhythmhigh                     2.393e-01  1.930e-01   1.240  0.22682
## harmonyM                       2.647e-01  1.930e-01   1.372  0.18276
## channelgrad                    1.275e-16  1.930e-01   0.000  1.00000
## channelmag                     8.587e-02  1.930e-01   0.445  0.66028
## rhythmlow:harmonyM            -2.973e-01  2.729e-01  -1.090  0.28674
## rhythmmed:harmonyM            -5.387e-01  2.729e-01  -1.974  0.05997 .
## rhythmhigh:harmonyM           -2.957e-01  2.729e-01  -1.084  0.28933
## rhythmlow:channelgrad         -3.849e-16  2.729e-01   0.000  1.00000
## rhythmmed:channelgrad         -4.654e-17  2.729e-01   0.000  1.00000
## rhythmhigh:channelgrad         9.146e-17  2.729e-01   0.000  1.00000
```

```
## rhythmlow:channelmag             4.325e-02  2.729e-01   0.159  0.87538
## rhythmmed:channelmag            -1.178e-01  2.729e-01  -0.432  0.66991
## rhythmhigh:channelmag            2.822e-02  2.729e-01   0.103  0.91849
## harmonyM:channelgrad             2.464e-16  2.729e-01   0.000  1.00000
## harmonyM:channelmag             -1.495e-01  2.729e-01  -0.548  0.58893
## rhythmlow:harmonyM:channelgrad   8.513e-17  3.859e-01   0.000  1.00000
## rhythmmed:harmonyM:channelgrad  -1.426e-16  3.859e-01   0.000  1.00000
## rhythmhigh:harmonyM:channelgrad -6.029e-16  3.859e-01   0.000  1.00000
## rhythmlow:harmonyM:channelmag    3.286e-01  3.859e-01   0.851  0.40295
## rhythmmed:harmonyM:channelmag    2.495e-01  3.859e-01   0.646  0.52414
## rhythmhigh:harmonyM:channelmag   1.899e-02  3.859e-01   0.049  0.96116
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.193 on 24 degrees of freedom
## Multiple R-squared:  0.6008, Adjusted R-squared:  0.2183
## F-statistic: 1.571 on 23 and 24 DF,  p-value: 0.1395
```

```
d2 <- aggregate(d[,colnames(d)[c(1:4,9:11)]], by = list(d$rhythm,d$harmony,d$version), mean)
colnames(d2)[1:3] <- c('rhythm','harmony','version')
d2$rhythm <- factor(d2$rhythm, levels = c('iso','low','med','high'))
d2$harmony <- factor(d2$harmony, levels = c('H','M'))

cnames <- c('iso','low','med','high')
for (r in 1:nrow(d2)){
 d2[r,'rhythm2'] <- which(cnames %in% d2$rhythm[r])
}
```

```
ggplot(d2,aes(x = rhythm, y = Mentrain, color = harmony)) +
  geom_jitter(width = 0.05, height = 0) +
  theme_bw()
```




```
m2 <- lm(Mentrain~rhythm*harmony,data = d2); summary(m2)
```

```
##
## Call:
## lm(formula = Mentrain ~ rhythm * harmony, data = d2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.17365 -0.06966  0.00000  0.06966  0.17365
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.21879    0.09659  -2.265  0.05328 .
## rhythmlow            0.10476    0.13659   0.767  0.46513
## rhythmmed            0.55401    0.13659   4.056  0.00365 **
## rhythmhigh           0.24875    0.13659   1.821  0.10607
## harmonyM             0.21491    0.13659   1.573  0.15428
## rhythmlow:harmonyM  -0.18779    0.19317  -0.972  0.35946
## rhythmmed:harmonyM  -0.45557    0.19317  -2.358  0.04607 *
## rhythmhigh:harmonyM -0.28935    0.19317  -1.498  0.17253
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1366 on 8 degrees of freedom
## Multiple R-squared:  0.7207, Adjusted R-squared:  0.4764
## F-statistic:  2.95 on 7 and 8 DF,  p-value: 0.07633
```

```r
m3 <- lm(Mentrain~(rhythm2^2)*harmony,data = d2); summary(m3)
```

```
##
## Call:
## lm(formula = Mentrain ~ (rhythm2^2) * harmony, data = d2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23290 -0.11713 -0.01383  0.05060  0.31944
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -0.29079    0.15610  -1.863   0.0871 .
## rhythm2           0.11955    0.05700   2.097   0.0578 .
## harmonyM          0.26569    0.22076   1.204   0.2520
## rhythm2:harmonyM -0.11358    0.08061  -1.409   0.1842
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1803 on 12 degrees of freedom
## Multiple R-squared:  0.2706, Adjusted R-squared:  0.0882
## F-statistic: 1.484 on 3 and 12 DF,  p-value: 0.2687
```

```r
anova(m2,m3)
```

```
## Analysis of Variance Table
##
## Model 1: Mentrain ~ rhythm * harmony
## Model 2: Mentrain ~ (rhythm2^2) * harmony
##   Res.Df     RSS Df Sum of Sq      F  Pr(>F)
## 1      8 0.14926
## 2     12 0.38988 -4  -0.24062 3.2241 0.07451 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
pwr_calc <- function(b0,b1,b2,b3,b0_sd,res_sd,nsims,ssizes){
  pwr <- data.frame() # intialize data frame to store the output
  for (s in 1:length(ssizes)){
    for (n in 1:nsims){
      idx <- (s-1)*nsims + n
      ssize <- ssizes[s]
     print(sprintf('sample size: %d / sim %d of %d',ssize,n,nsims))

      #print(ssize)
      #print(n)
      conds1 <- rep(rep(1:4, each = 4), ssize) #add condition factor
      conds2 <- rep(0:1, ssize * 8) #add condition factor
      subs <- rep(1:ssize, each = 16) # add subject codes
      intercept <- rep(rnorm(ssize,b0,b0_sd), each = 16) # add intercept
      beta1 <- rep(rep(b1, each = 4), ssize) # add condition effect
      beta2 <- rep(b2, ssize * 8) #add condition factor
      beta3 <- c()
```

```
      for (c1 in 1:length(conds1)){
          beta3[c1] <- b3[conds1[c1]]*conds2[c1]
      }
      residuals <- rnorm(length(subs),0,res_sd) # add residual noise

      # collect in a dataframe and calculate simulated measured outcome (y)
      d <- data.frame('sub' = subs,
                      'cond1' = as.character(conds1),
                      'cond2' = as.character(conds2),
                      'b0' = intercept,
                      'b1' = beta1,
                      'b2' = beta2,
                      'b3' = beta3,
                      'res' = residuals,
                      'y' = intercept + beta1 + beta2 + beta3 + residuals)

      # fit models
      m0 <- lmer(y~1 + (1|sub), data = d, REML = FALSE)
      m1 <- lmer(y~cond1 + (1|sub), data = d, REML = FALSE)
      m2 <- lmer(y~cond1 + cond2 + (1|sub), data = d, REML = FALSE)
      m3 <- lmer(y~cond1*cond2 + (1|sub), data = d, REML = FALSE)
      # perform likelihood ratio test
      test1 <- anova(m0,m1)
      test2 <- anova(m1,m2)
      test3 <- anova(m2,m3)

      #store output of simulation
      pwr[idx,'sim'] <- n
      pwr[idx, 'ssize'] <- ssize
      #pwr[idx, 'b0'] <- summary(m1)$coefficients[1]
      #pwr[idx, 'b1'] <- summary(m1)$coefficients[2:(2+length(b1))]
      pwr[idx, 'sd_int'] <- attr(summary(m1)$varcor$sub,"stddev")
      pwr[idx, 'sd_res'] <- summary(m1)$sigma
      pwr[idx, 'x2_1'] <- test1$Chisq[2]
      pwr[idx, 'x2_2'] <- test2$Chisq[2]
      pwr[idx, 'x2_3'] <- test3$Chisq[2]
      pwr[idx, 'p1'] <- test1$`Pr(>Chisq)`[2]
      pwr[idx, 'p2'] <- test2$`Pr(>Chisq)`[2]
      pwr[idx, 'p3'] <- test3$`Pr(>Chisq)`[2]
    }
  }
  return(pwr)
}
```

```
b0 <- c(-0.2)
b1 <- c(0,0,0.1,0) # Z # fT # uV # minimum difference between conditions
b2 <- c(0,0.1)
b3 <- c(0,0,0.3,0)

b0_sd <- 1 # fT # standard deviation of the intercept
res_sd <- 0.5 # fT # residual standard deviation
nsims <- 200 # number of simulations per sample size
ssizes <- c(20,30,40,50) # sample sizes
```

Make a report on power, as a function of sample size:

```
summary1 <- aggregate(pwr1$p1,by = list(pwr1$ssize), FUN = function(x) sum(x < 0.05)/length(x))
colnames(summary1) <- c('sample.size','power')
print(summary1)
```

```
##   sample.size power
## 1          20 0.905
## 2          30 0.995
## 3          40 1.000
## 4          50 1.000
```

```
summary2 <- aggregate(pwr1$p2,by = list(pwr1$ssize), FUN = function(x) sum(x < 0.05)/length(x))
colnames(summary2) <- c('sample.size','power')
print(summary2)
```

```
##   sample.size power
## 1          20 0.880
## 2          30 0.985
## 3          40 1.000
## 4          50 1.000
```

```
summary3 <- aggregate(pwr1$p3,by = list(pwr1$ssize), FUN = function(x) sum(x < 0.05)/length(x))
colnames(summary3) <- c('sample.size','power')
print(summary3)
```

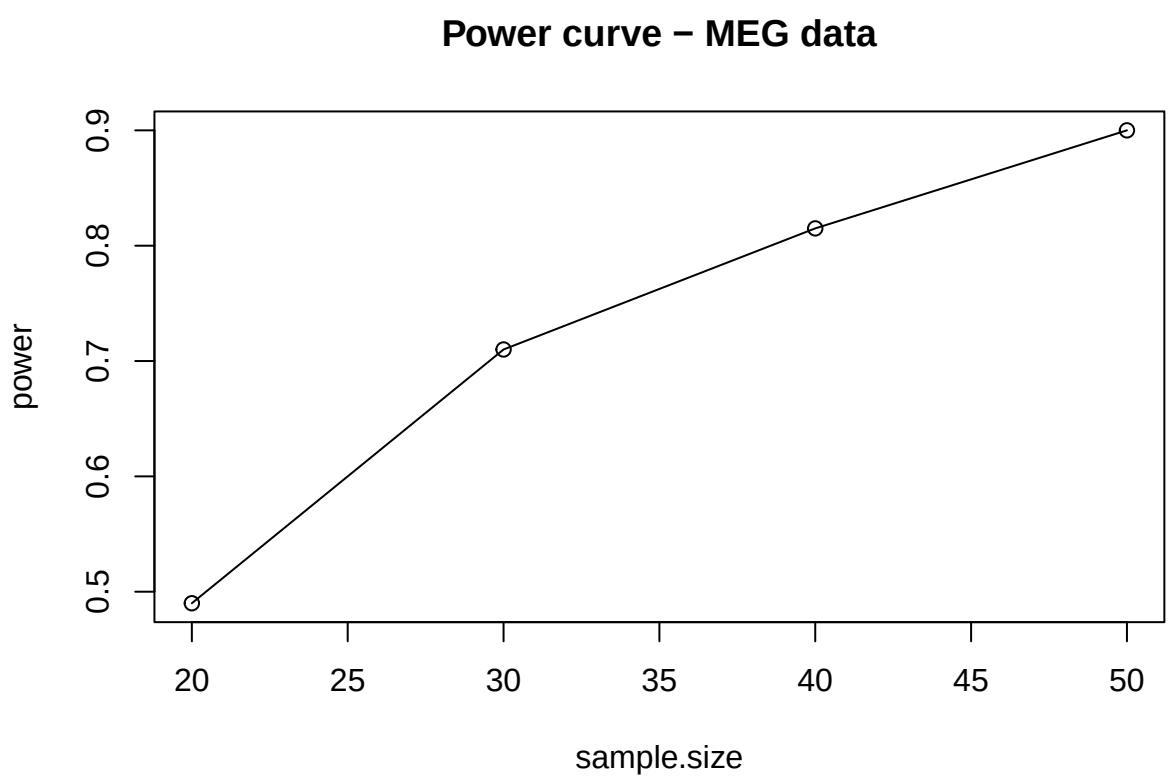
```
##   sample.size power
## 1          20 0.490
## 2          30 0.710
## 3          40 0.815
## 4          50 0.900
```

And make a plot:

```
with(summary3, plot(sample.size, power, type = 'ol'))
```

```
## Warning in plot.xy(xy, type, ...): plot type 'ol' will be truncated to first
## character
```

```
title('Power curve - MEG data')
```


Power curve − MEG data
power
0.5
0.6
0.7
0.8
0.9
20
25
30
35
40
45
50
sample.size