marcel.vaz

Código da função

```
###################################
### Cálculo do poder da anova.MC ###
###################################

# ARGUMENTOS

# N - número de amostras (testes anova.MC a serem feitos)
# n - número de permutações a serem feitos pelo anova.MC
# p - valor crítico de alfa
# dados - data.frame com 4 colunas e b linhas (b = número de blocos)


anova.power=function(dados,N,n,pc=.05){

#########################################################################
#########################################################################
####
            anova.MC=function(dados,n){
                 aleat=function(dados){
# função que faz as permutações
                       mc=dados
# crio o objeto que receberá as permutações, idêntico aos dados originais
                       mc$resp=rep(NA,length(dados$resp))
# retiro os valores originais
                       for(j in 1:length(unique(dados$bloco))){
# número de blocos
mc$resp[mc$bloco==j]=sample(dados$resp[dados$bloco==j])   # permutação dos
valores encontrados
                                                                  }
return(summary(aov(resp~A*B+Error(bloco/(A*B)),mc))[5][[1]][[1]][[4]][1:3])
# extração e cálculo de F
                                      }
                 result=data.frame(A=rep(NA,n),B=rep(NA,n),A.B=rep(NA,n))
# crio o objeto que receberá os valores de F

                 for(i in 1:n){                          # número de
permutações
                               result[i,]=aleat(dados)   # gera n valores de
F
                              }

real=summary(aov(resp~A*B+Error(bloco/(A*B)),data=dados))[5][[1]][[1]][[4]][
1:3] # extração e cálculo de F dos dados reais
                 p=data.frame(fatores="p",
A=(length(result[result[,1]>=real[1],1])+1)/(n+1),   # probabilidade de se
```

```
encontrar o efeito do fator A ao acaso
B=(length(result[result[,2]>=real[2],2])+1)/(n+1),   # o mesmo para o fator
B
A.B=(length(result[result[,3]>=real[3],3])+1)/(n+1)  # e idem para a
interação A:B
                              )
                return(p)

                                     }

##########################################################################
##########################################################################
####
           data.gen=function(nb,int,varb,vare,efA,efB,efAB){

                d=data.frame(
                             bloco=rep(1:nb,each=4),    ######### crio o
fator BLOCO
                             A=rep(0:1,each=2),           ####### idem para
o fator A
                             B=0:1,                         ##### idem para
o fator B
                             resp=NA                          ### crio
coluna vazia para a variável resposta
                             )

                efb=rnorm(nb,0,varb)                            # crio o
efeito randômico BLOCO com média zero; a variância diz quão diferentes serão
os blocos

                for(i in 1:nb){ # ciclo número de blocos
                  for(j in 0:1){ ######### ciclo fator A
                    for(k in 0:1){ ####### ciclo fator B
                      d$resp[d$bloco==i&d$A==j&d$B==k]=round((
#
                                                             int+
# crio cada valor resposta a partir de uma média
                                                             efA*j+
#################### adiciono o efeito do fator A
                                                             efB*k+
############################### idem para o fator B
                                                             efAB*j*k+
##########################idem para a interação A:B
                                                             efb[i]+
########################## idem para o fator BLOCO
rnorm(1,0,vare) ##### adiciono o erro associado a cada observação
                                                             ),2) }}}
############################## arredonda os dados
#
                return(d)} # retorna a tabela de dados gerados
```

```
##########################################################################
##########################################################################
####

            medias=aggregate(dados$resp,list(dados$A,dados$B),mean) #
calculo as médias por tratamento
            desvios=aggregate(dados$resp,list(dados$A,dados$B),sd) # idem
para os desvios
            medias.bloco=aggregate(dados$resp,list(dados$bloco),mean) #
calculo as médias por bloco

            nb=length(unique(dados$bloco)) # conto o número de blocos do
desenho
            int=medias[1,3]   # intercepto ou média dos controles
            varb=sd(medias.bloco[,2]) # desvios entre os blocos
            efA=medias[2,3]-medias[1,3]  # calculo o efeito do tratamento
para o fator A
            efB=medias[3,3]-medias[1,3]  # idem para o fator B
            efAB=medias[4,3]-medias[2,3]-medias[3,3]+medias[1,3] # idem para
a interação entre A e B

            efbloco=numeric()
            for(i in 1:nb){
            efbloco[i]=medias.bloco[i,2]-mean(dados$resp) # calculo o efeito
do bloco
            }

            residuos=numeric()
            for(i in 1:length(dados$resp)){
                residuos[i]=dados[i,4]-int-efA*dados[i,2]-efB*dados[i,3]-
efAB*dados[i,2]*dados[i,3]-efbloco[dados[i,1]] # cálculo para encontrar os
resíduos
                }

            vare=sd(residuos) # desvio padrão dos resíduos

            d=list()
            for(i in 1:N){
                d[[i]]=data.gen(nb,int,varb,vare,efA,efB,efAB)
                }

            result=data.frame(pA=rep(NA,N),pB=rep(NA,N),pAB=rep(NA,N))
            for(i in 1:N){
                result[i,]=anova.MC(d[[i]],n)[2:4]
                }

            power=data.frame(row.names="poder(%)",
A=length(result[result$pA<=pc,1])*100/length(result$pA),
B=length(result[result$pB<=pc,2])*100/length(result$pB),
AB=length(result[result$pAB<=pc,3])*100/length(result$pAB)
                             )
```

```
            return(power)
            }


# testando...

anova.power(dados,2,999)
```