Thiago Mitonori Pereira

Sou mestrando em Ecologia, Instituto de Biociências, USP. Graduado em ciências biológicas pela UNIFESP

A minha dissertação proposta estava relacionada com plasticidade fenotípica e adaptações locais para sementes e plântulas de espécies ocorrentes em três fisionomias vegetais de restinga, porém o projeto foi abortado por inexequibilidade. Agora estou trabalhando em um projeto com chuva de sementes de 4 áreas de restinga com estágios de regeneração diferentes. Sou orientado pela Dra. Adriana M. Z. Martini.

Meus Exercícios

thiago_m._p._exercicio1_.r

thiago.m.p.exercicio2_.r

exercicio3fthi.r

exec4tfthi_.r

exec5.thi.r

exec.6.thi.r

exec.7fthi_.r

exec.9_thi.r

Proposta de Trabalho Final

Principal

Dentro do ciclo de vida das plantas a germinação é um período crítico e essencial e o conhecimento sobre este tema é importante para a compreensão dos processos que ocorrem dentro de comunidades, como o estabelecimento, sucessão e regeneração natural, além da distribuição e da abundância das espécies (Raich & Khoon, 1990; Vázquez-Yanes & Orozco-Segovia, 1993)

Neste contexto durante o meu TCC trabalhei com germinação de sementes de mirtáceas em diferentes qualidades de luz (diferentes razões Vermelho/Vermelho Extremo, ausência de luz e luz branca). A germinação foi avaliada por 2 parâmetros: a germinabilidade, que é representada pela porcentagem final de germinação, e o tempo médio de germinação, dado em dias, que representa a média ponderada do tempo necessário para a germinação, tendo como fator de ponderação, o número de sementes germinadas a cada intervalo de tempo (Ranal & Santana, 2006). Porém durante

às analises dos dados em quase todos os momentos (exceto durante os testes estatísticos) utilizei o “contestado” Excel.

Assim percebendo que com uma única função do R eu poderia fazer os cálculos de germinabilidade e de tempo médio de germinação, que são parâmetros medidos na maioria dos trabalhos que envolvem germinação, além dos testes e produção dos gráficos a partir de um banco de dados organizado em numero sementes germinadas por dia nas réplicas dos diferentes tratamentos, e que eu poderia generalizar a entrada de dados para qualquer espécie ou tratamento criei minha proposta.

Pretendo criar uma função que calcule a porcentagem final de germinação e o tempo médio de germinação para diferentes tratamentos (de qualquer gênero), a partir de um banco de dados organizado em numero sementes germinadas por dia nas réplicas dos diferentes tratamentos, fazendo também análise da distribuição dos dados, testes de normalidade, análise comparativa de médias através de gráficos de médias, e análise comparativa através de ANOVA para um fator(tratamento), caso os pressupostos sejam atendidos.

Esta função teria uma grande utilidade para reanalisar os resultados das espécies que eu analisei e para realizar continuidade para as espécies que ainda faltam antes de escrever o artigo para publicação e poderia ser útil para quem quiser analisar germinação de forma comparativa entre tratamentos diferentes.

REFERÊNCIAS:

RAICH, JW.; KHOON, GW. Effect of canopy openings on tree seed germination in a Malayan dipterocarp forest. Journal of Tropical Ecology, v. 6, no. 2, p. 203-17, Mai. 1990.

RANAL, M. A.; SANTANA, D. G. How and why to measure the germination process? Revista Brasileira de Botânica, São Paulo, v.29, n.1, p.1-11, Jan. 2006.

VÁZQUEZ-YANES, C.; OROZCO-SEGOVIA, A. Patterns of seed longevity and germination in the tropical rainforest. Annual Review of Ecology and Systematics, v.24, n.1, p.69-87, 1993.

Plano B

Ainda no contexto germinação Criar uma função que relacione características numéricas da sementes como massa e tamanho das sementes com o tempo médio de germinação e a porcentagem final de sementes germinadas, através de regressões lineares.

Comentários das propostas (Leo)

A proposta A parece factível, mas sugiro ir por partes. Um primeiro passo seria calcular taxas e tempo médio de germinação por tratamento (que poderiam ser também espécies diferentes, ou seja, tu pode ter qualquer indexador que pode ser um argumento da função). Depois de resolvido o primeiro problema, invista em complicações como a análise das distribuições (normalidade) e testes de hipótese. Não ficou claro como a taxas de germinação e tempos médios serão obtidos a partir dos dados de entrada. O gráfico de médias que tu se refere é um box-plot? A proposta B parece muito simples, haja vista que pode ser resolvida com poucas linhas de código em um script, depois que você obteve o tempo médio de germinação e a porcentagem final de sementes germinadas.

Página de Ajuda

```
measure.germ        package:       R Documentation

     Análise de dados de germinação para tratamentos independentes


Description:

    Através de uma tabela organizada em numero sementes germinadas
    por contagem nas réplicas dos diferentes tratamentos, a função
    calcula o tempo médio de germinação e a porcentagem final de
    sementes germinadas para cada réplica fazendo também a análise
    da distribuição dos dados, testes de normalidade, análise
    gráfica através de Boxplots, e análise comparativa das médias
    através de ANOVA para um fator(tratamento), podendo ainda fazer
    comparações multiplas através do teste de Tukey e analise gráfica
    dessas comparações através de gráficos do intervaldo de confiança
    das diferenças das médias.




Usage:

     measure.germ<-function(x, medida="TMG",
teste="param",comp.mult=FALSE,mediaplot= TRUE)


Arguments:

    x: Tem que ser uma tabela organizada onde a primeira coluna deve conter
    no primeiro elemento a palavra Tratamento com T maiúsculo, no segundo
    elemento deve conter -, e nos demais cada réplica entitulada de acordo
    com o seu tratamento. A primeira linha, exceto à primeira coluna, deve
    conter ts que vão de t1 até ti onde i é igual ao número de contagens,
    a segunda linha deve conter o dia após o início do experimeto relativo
    à contagem, as demais linhas devem conter no numero de sementes germi-
    nadas por réplica em cada contagem (ver nas tabelas exemplos).
    medida: Define a medida a ser calculada pela função. Se "TMG" calcula
    o tempo médio de germinação,sendo plotados histograma dos dados do tempo
    médio, e boxplots dos tempos médios por tratamento. Se "porcentagem"
    calcula a porcentagem final de sementes germinadas sendo plotados os
    mesmos gráficos da anterior mas para a porcentagem. Por Default calcula
    o tempo médio de germinação
    teste: define se o teste a ser realizado é paramétrico ou não-
paramétrico.
    Se "param" roda teste paramétrico ANOVA, plotará gráficos e realizará
    testes para veriifcar se os pressupostos do teste são atendidos. Se
    "nparam" roda teste  não-paramétrico  kruskal-wallis.
```

```
    comp.mult: Se TRUE a função realizará teste de comparações múltiplas de
    Tukey entre os tratamentos, e plotará gráfico de intervalos de confiança
    das diferenças utilizadas nos teste de comparação multipla.

    mediaplot: Se TRUE a função plotará gráfico de médias da medida
calculada.
    Tempo médio de germinação e ou porcentagem final de sementes germinadas.



Details:

    A tabela de entrada deve sempre seguir um padrão formatação
    onde a primeira coluna deve conter no primeiro elemento a palavra
    Tratamento com T maiúsculo, no segundo elemento deve conter -, e nos
    demais cada réplica entitulada de acordo com o seu tratamento. A
primeira
    linha, exceto à primeira coluna, deve conter ts que vão de t1 até ti
    onde i é igual ao número de contagens, a segunda linha deve conter o
    dia após o início do experimeto relativo à contagem, as demais linhas
    devem conter no numero de sementes germinadas por réplica em cada
    contagem (ver nas tabelas exemplos). Os tipos de tratamentos,o numero
    de réplicas e o numero de contagens por tratamento ficam à critério do
    usuário, numero total de sementes por réplica será perguntado
    interativamente.



Value:

    A função calcula o tempo médio de germinação ou a porcentagem de
sementes
    germinadas para cada réplica retornando:
    - A média dos parametros calculados por tratamento.
    - Os desvios padrões de cada parametro calculado por tratamento
    - Teste de normalidade dos dados de Shapiro- Wilk
    - Teste de homocedastidade de variancia de Bartlett
    Dependendo dos parametros condicionantes
    - Teste ANOVA ou Kruskall-Wallis
    - Teste de normalidade e homocedasticidade relacionados aos resíduos
    - Teste de comparações múltiplas de Tukey, os intervalos de confiança
    e resumo do teste em letras que indentificam quais tratamentos são tem
    respostas iguais (mesmas letras) ou diferentes (letras diferentes).


Warning:


Author
     Thiago Mitonori Pereira
```

```
References:

        RAICH, JW.; KHOON, GW. Effect of canopy openings on tree seed
germination
    in a Malayan dipterocarp forest. Journal of Tropical Ecology, v. 6,
    no. 2, p. 203-17, Mai. 1990.

    RANAL, M. A.; SANTANA, D. G. How and why to measure the germination
    process? Revista Brasileira de Botânica, São Paulo, v.29, n.1,
    p.1-11, Jan. 2006.

    VÁZQUEZ-YANES, C.; OROZCO-SEGOVIA, A. Patterns of seed longevity and
    germination in the tropical rainforest. Annual Review of Ecology and
    Systematics, v.24, n.1, p.69-87, 1993.
Examples:

e.umb= read.table("e.umbF.csv", sep=";",header=TRUE, as.is=T)
measure.germ(e.umb, medida="TMG", teste="param",comp.mult=T,mediaplot= T)
```

Código da Função

```
measure.germ<-function(x, medida="TMG", teste="param",comp.mult=FALSE,
mediaplot= TRUE)
{
  nt <-as.numeric(readline("Qual é numero total de sementes por réplica?"))
  x[is.na(x)]<-0
  x1=x[-1,]
  dias=as.numeric(x[1,-1])
  x1[,-1]=dias*x[-1,-1]
  x1
  x1[,"Sum(ni*ti)"]=rowSums(x1[,-1])
  x1
  x1[,"Sum(ni)"]=rowSums(x[-1,-1])
  x1
  x1[,"TMG"]=x1[,"Sum(ni*ti)"]/x1[,"Sum(ni)"]
  x1
  if (medida=="TMG" )
  {
    X11()
    media.TMG= tapply(x1[,"TMG"],x1[,1],mean)
    dp.TMG= tapply(x1[,"TMG"],x1[,1],sd)
    hist(x1$TMG, main= "Histograma do Tempo Médio de Germinação",
xlab="Tempo Médio de Germinação",ylab= "Frequência" )
    shapiro.test(x1$TMG)
    bartlett.test(x1$TMG ~ Tratamento, data=x1)
    X11()
    boxplot(TMG~Tratamento, ylab="Tempo Médio de Germinação (dias)",
cex.axis=0.8, xlab="Tratamentos", data=x1)
    if (teste=="param")
```

```
    {
      X11()
      m.1= aov(x1$TMG~x1$Tratamento)
      summary(m.1)
      par(mfrow=c(2,2))
      plot(m.1)
      x1$m.1.res <- residuals(m.1)
      X11()
      hist(x1$m.1.res, main= "Histograma dos Resíduos",
xlab="Resíduos",ylab= "Frequência")
      shapiro.test(x1$m.1.res)
      bartlett.test(residuals(m.1) ~ Tratamento, data=x1)
      resultados <-list("Média do Tempo Médio de Germinação por Tratamento"
= media.TMG,
                        "Desvio Padrão do Tempo Médio de Germinação por
Tratamento" = dp.TMG,
                        "Teste de Normalidade do Tempo Médio de Germinação"
= shapiro.test(x1$TMG),
                        "Teste de Homocedasticidade do Tempo Médio de
Germinação por Tratamento"=bartlett.test(x1$TMG ~ Tratamento, data=x1),
                        "ANOVA" = summary(m.1), "Teste de Normalidade dos
Resíduos" =shapiro.test(x1$m.1.res),
                        "Teste de Homocedasticidade dos
Resíduos"=bartlett.test(residuals(m.1) ~ Tratamento, data=x1) )
      if (comp.mult==TRUE)
      {
        require(multcomp)
        x2=data.frame(x1$TMG,x1$Tratamento)
        m.2=aov(x1.TMG~x1.Tratamento, data=x2)
        tukey <-  glht(m.2, linfct = mcp(x1.Tratamento = "Tukey"))
        summary(tukey)
        confint(tukey)
        cld(tukey)
        X11()
        old.oma <- par(oma=c(0,5,0,0))
        plot(confint(tukey),main= "Intervalos de Confiança das Diferenças
(95%)", xlab="Valores das Diferenças",cex.axis=0.8)
        par(old.oma)
        resultados <-list("Média do Tempo Médio de Germinação por
Tratamento" = media.TMG,
                          "Desvio Padrão do Tempo Médio de Germinação por
Tratamento" = dp.TMG,
                          "Teste de Normalidade do Tempo Médio de
Germinação" = shapiro.test(x1$TMG),
                          "Teste de Homocedasticidade do Tempo Médio de
Germinação por Tratamento"=bartlett.test(x1$TMG ~ Tratamento, data=x1),
                          "ANOVA" = summary(m.1), "Teste de Normalidade dos
Resíduos" =shapiro.test(x1$m.1.res),
                          "Teste de Homocedasticidade dos
Resíduos"=bartlett.test(residuals(m.1) ~ Tratamento, data=x1),
```

```
                           " Teste de Multiplas Comparações "=
summary(tukey),"Intervalos de Confiança das Comparações Simultâneas" =
confint(tukey), "Resumo das Comparações"=cld(tukey) )
      }
    }
    if (teste=="nparam")
    {
      kruskal.test(x1$TMG~as.factor(x1$Tratamento))
      resultados <-list("Média do Tempo Médio de Germinação por Tratamento"
= media.TMG,
                        "Desvio Padrão do Tempo Médio de Germinação por
Tratamento" = dp.TMG,
                        "Teste de Normalidade do Tempo Médio de Germinação"
= shapiro.test(x1$TMG),
                        "Teste de Homocedasticidade do Tempo Médio de
Germinação por Tratamento"=bartlett.test(x1$TMG ~ Tratamento, data=x1),
                        "Teste Não Paramétrico Kruskal-Wallis"=
kruskal.test(x1$TMG~as.factor(x1$Tratamento)) )
      if (comp.mult==TRUE)
      {
        cat("\n\t
Erro!! Não é possível fazer comparações multiplas sem o modelo aov. Para
habilitar utilizar teste paramétrico.
            \n\t\n\t
            ")
      }
    }
    if (mediaplot==TRUE)
    {
    y= readline("\n Qual é o tipo de barra de erro desejado para o grafico
de médias?\n\t Se erro padrão responda: se \n\t Se desvio padrão responda:
sd \n\t Se intervalos de confiança responda: conf.int \n\t Se nenhum
responda: none \n Resposta: " )
    bis<- function (response, factor1, factor2, error.bars = c("se", "sd",
                                                                "conf.int",
"none"), level = 0.95, xlab = deparse(substitute(factor1)),
                    ylab = paste("mean of", deparse(substitute(response))),
legend.lab = deparse(substitute(factor2)),
                    main = "Plot of Means", pch = 1:n.levs.2, lty =
1:n.levs.2,
                    col = palette(), ylim = NULL)
    {
      if (!is.numeric(response))
        stop(cat("Argument response must be numeric."))
      xlab
      ylab
      legend.lab
      error.bars <- match.arg(error.bars)
      if (missing(factor2)) {
        if (!is.factor(factor1))
          stop(cat("Argument factor1 must be a factor."))
```

```
        valid <- complete.cases(factor1, response)
        factor1 <- factor1[valid]
        response <- response[valid]
        means <- tapply(response, factor1, mean)
        sds <- tapply(response, factor1, sd)
        ns <- tapply(response, factor1, length)
        if (error.bars == "se")
          sds <- sds/sqrt(ns)
        if (error.bars == "conf.int")
          sds <- qt((1 - level)/2, df = ns - 1, lower.tail = FALSE) *
            sds/sqrt(ns)
        sds[is.na(sds)] <- 0
        yrange <- if (error.bars != "none")
          c(min(means - sds, na.rm = TRUE), max(means + sds,
                                                na.rm = TRUE))
        else range(means, na.rm = TRUE)
        levs <- levels(factor1)
        n.levs <- length(levs)
        if (is.null(ylim)) {
          plot(c(1, n.levs), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main)
        }
        else {
          plot(c(1, n.levs), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main, ylim = ylim)
        }
        points( means, pch = 16, cex = 2)
        box()
        axis(2)
        axis(1, at = 1:n.levs, labels = levs)
        if (error.bars != "none")
          arrows(1:n.levs, means - sds, 1:n.levs, means + sds,
                 angle = 90, lty = 2, code = 3, length = 0.125)
      }
      else {
        if (!(is.factor(factor1) | is.factor(factor2)))
          stop(cat("Arguments factor1 and factor2 must be factors."))
        valid <- complete.cases(factor1, factor2, response)
        factor1 <- factor1[valid]
        factor2 <- factor2[valid]
        response <- response[valid]
        means <- tapply(response, list(factor1, factor2), mean)
        sds <- tapply(response, list(factor1, factor2), sd)
        ns <- tapply(response, list(factor1, factor2), length)
        if (error.bars == "se")
          sds <- sds/sqrt(ns)
        if (error.bars == "conf.int")
          sds <- qt((1 - level)/2, df = ns - 1, lower.tail = FALSE) *
            sds/sqrt(ns)
        sds[is.na(sds)] <- 0
```

```
        yrange <- if (error.bars != "none")
          c(min(means - sds, na.rm = TRUE), max(means + sds,
                                                na.rm = TRUE))
        else range(means, na.rm = TRUE)
        levs.1 <- levels(factor1)
        levs.2 <- levels(factor2)
        n.levs.1 <- length(levs.1)
        n.levs.2 <- length(levs.2)
        if (length(pch) == 1)
          pch <- rep(pch, n.levs.2)
        if (length(col) == 1)
          col <- rep(col, n.levs.2)
        if (length(lty) == 1)
          lty <- rep(lty, n.levs.2)
        if (n.levs.2 > length(col))
          stop(sprintf(cat("Number of groups for factor2, %d, \nexceeds
number of distinct colours, %d."),
                       n.levs.2, length(col)))
        if (is.null(ylim)) {
          plot(c(1, n.levs.1 * 1.4), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main)
        }
        else {
          plot(c(1, n.levs.1 * 1.4), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main, ylim = ylim)
        }
        box()
        axis(2)
        axis(1, at = 1:n.levs.1, labels = levs.1)
        for (i in 1:n.levs.2) {
          points(1:n.levs.1, means[, i], type = "b", pch = pch[i],
                 cex = 2, col = col[i], lty = lty[i])
          if (error.bars != "none")
            arrows(1:n.levs.1, means[, i] - sds[, i], 1:n.levs.1,
                   means[, i] + sds[, i], angle = 90, code = 3,
                   col = col[i], lty = lty[i], length = 0.125)
        }
        x.posn <- n.levs.1 * 1.1
        y.posn <- sum(c(0.1, 0.9) * par("usr")[c(3, 4)])
        text(x.posn, y.posn, legend.lab, adj = c(0, -0.5))
        legend(x.posn, y.posn, levs.2, pch = pch, col = col,
               lty = lty)
      }
      invisible(NULL)
    }
    X11()
    bis(x1$TMG, as.factor(x1$Tratamento), error.bars=y,  main = "Grafico de
Médias: Tempo Médio de Germinação", ylab="Tempo Médio de Germinação (dias)",
xlab="Tratamentos")
    }
    }
```

```
  if (medida=="porcentagem" )
  {
    x1[,"porc.de.germinadas"]=x1[,"Sum(ni)"]*100/nt
    media.porc= tapply(x1[,"porc.de.germinadas"],x1[,1],mean)
    dp.porc= tapply(x1[,"porc.de.germinadas"],x1[,1],sd)
    X11()
    hist(x1$porc.de.germinadas, main ="Histograma da Porcentagem de Sementes
Germinadas", ylab="Frequência", xlab="Porcentagem de Sementes Germinadas")
    X11()
    boxplot(porc.de.germinadas~Tratamento,main = "Boxplots: Porcentagem por
Tratamento", cex.axis=0.8, ylab="Porcentagem de Sementes Germinadas (%)",
xlab="Tratamentos", data=x1)
    shapiro.test(x1$porc.de.germinadas)
    bartlett.test(x1$porc.de.germinadas ~ Tratamento, data=x1)
    if (teste=="param")
    {
      X11()
      m.1= aov(x1$porc.de.germinadas~x1$Tratamento)
      summary(m.1)
      par(mfrow=c(2,2))
      plot(m.1)
      x1$m.1.res <- residuals(m.1)
      X11()
      hist(x1$m.1.res, main= "Histograma dos Resíduos",
xlab="Resíduos",ylab= "Frequência")
      shapiro.test(x1$m.1.res)
      bartlett.test(residuals(m.1) ~ Tratamento, data=x1)
      resultados <-list("Média da Porcentagem  Final de Sementes Germinadas
por Tratamento" = media.porc,
                        "Desvio Padrão da Porcentagem  Final de Sementes
Germinadas por Tratamento" = dp.porc,
                        "Teste de Normalidade da Porcentagem  Final de
Sementes Germinadas" = shapiro.test(x1$porc.de.germinadas),
                        "Teste de Homocedasticidade da Porcentagem  Final de
Sementes Germinadas por Tratamento"=bartlett.test(x1$porc.de.germinadas ~
Tratamento, data=x1),
                        "ANOVA" = summary(m.1), "Teste de Normalidade dos
Resíduos" =shapiro.test(x1$m.1.res),
                        "Teste de Homocedasticidade dos
Resíduos"=bartlett.test(residuals(m.1) ~ Tratamento, data=x1) )
      if (comp.mult==TRUE)
      {
        require(multcomp)
        x2=data.frame(x1$porc.de.germinadas,x1$Tratamento)
        m.2=aov(x1.porc.de.germinadas~x1.Tratamento, data=x2)
        tukey <-  glht(m.2, linfct = mcp(x1.Tratamento = "Tukey"))
        summary(tukey)
        confint(tukey)
        cld(tukey)
        X11()
```

```
        old.oma <- par(oma=c(0,5,0,0))
        plot(confint(tukey),main= "Intervalos de Confiança das Diferenças
(95%)", xlab="Valores das Diferenças",cex.axis=0.8)
        par(old.oma)
        resultados <-list("Média da Porcentagem  Final de Sementes
Germinadas por Tratamento" = media.porc,
                          "Desvio Padrão da Porcentagem  Final de Sementes
Germinadas por Tratamento" = dp.porc,
                          "Teste de Normalidade da Porcentagem  Final de
Sementes Germinadas" = shapiro.test(x1$porc.de.germinadas),
                          "Teste de Homocedasticidade da Porcentagem  Final
de Sementes Germinadas por Tratamento"=bartlett.test(x1$porc.de.germinadas ~
Tratamento, data=x1),
                          "ANOVA" = summary(m.1), "Teste de Normalidade dos
Resíduos" =shapiro.test(x1$m.1.res),
                          "Teste de Homocedasticidade dos
Resíduos"=bartlett.test(residuals(m.1) ~ Tratamento, data=x1),
                          " Teste de Multiplas Comparações "=
summary(tukey),"Intervalos de Confiança das Comparações Simultâneas" =
confint(tukey) , "Resumo das Comparações"=cld(tukey) )
      }
    }
    if (teste=="nparam")
    {
      kruskal.test(x1$porc.de.germinadas~as.factor(x1$Tratamento))
      resultados <-list("Média da Porcentagem  Final de Sementes Germinadas
por Tratamento" = media.porc,
                        "Desvio Padrão da Porcentagem  Final de Sementes
Germinadas por Tratamento" = dp.porc,
                        "Teste de Normalidade da Porcentagem  Final de
Sementes Germinadas" = shapiro.test(x1$porc.de.germinadas),
                        "Teste de Homocedasticidade da Porcentagem  Final de
Sementes Germinadas por Tratamento"=bartlett.test(x1$porc.de.germinadas ~
Tratamento, data=x1),
                        "Teste Não Paramétrico Kruskal-Wallis"=
kruskal.test(x1$TMG~as.factor(x1$Tratamento)) )
      if (comp.mult==TRUE)
      {
        cat("\n\t
Erro!! Não é possível fazer comparações multiplas sem o modelo aov. Para
habilitar utilizar teste paramétrico.
            \n\t\n\t
            ")
      }
    }
    if (mediaplot==TRUE)
    {
      y= readline("\n Qual é o tipo de barra de erro desejado para o grafico
de médias?\n\t Se erro padrão responda: se \n\t Se desvio padrão responda:
sd \n\t Se intervalos de confiança responda: conf.int \n\t Se nenhum
responda: none \n Resposta: " )
```

```
    bis<- function (response, factor1, factor2, error.bars = c("se", "sd",
                                                               "conf.int",
"none"), level = 0.95, xlab = deparse(substitute(factor1)),
                    ylab = paste("mean of", deparse(substitute(response))),
legend.lab = deparse(substitute(factor2)),
                    main = "Plot of Means", pch = 1:n.levs.2, lty =
1:n.levs.2,
                    col = palette(), ylim = NULL)
    {
      if (!is.numeric(response))
        stop(cat("Argument response must be numeric."))
      xlab
      ylab
      legend.lab
      error.bars <- match.arg(error.bars)
      if (missing(factor2)) {
        if (!is.factor(factor1))
          stop(cat("Argument factor1 must be a factor."))
        valid <- complete.cases(factor1, response)
        factor1 <- factor1[valid]
        response <- response[valid]
        means <- tapply(response, factor1, mean)
        sds <- tapply(response, factor1, sd)
        ns <- tapply(response, factor1, length)
        if (error.bars == "se")
          sds <- sds/sqrt(ns)
        if (error.bars == "conf.int")
          sds <- qt((1 - level)/2, df = ns - 1, lower.tail = FALSE) *
            sds/sqrt(ns)
        sds[is.na(sds)] <- 0
        yrange <- if (error.bars != "none")
          c(min(means - sds, na.rm = TRUE), max(means + sds,
                                                na.rm = TRUE))
        else range(means, na.rm = TRUE)
        levs <- levels(factor1)
        n.levs <- length(levs)
        if (is.null(ylim)) {
          plot(c(1, n.levs), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main)
        }
        else {
          plot(c(1, n.levs), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main, ylim = ylim)
        }
        points( means, pch = 16, cex = 2)
        box()
        axis(2)
        axis(1, at = 1:n.levs, labels = levs)
        if (error.bars != "none")
          arrows(1:n.levs, means - sds, 1:n.levs, means + sds,
```

```
                  angle = 90, lty = 2, code = 3, length = 0.125)
      }
      else {
        if (!(is.factor(factor1) | is.factor(factor2)))
          stop(cat("Arguments factor1 and factor2 must be factors."))
        valid <- complete.cases(factor1, factor2, response)
        factor1 <- factor1[valid]
        factor2 <- factor2[valid]
        response <- response[valid]
        means <- tapply(response, list(factor1, factor2), mean)
        sds <- tapply(response, list(factor1, factor2), sd)
        ns <- tapply(response, list(factor1, factor2), length)
        if (error.bars == "se")
          sds <- sds/sqrt(ns)
        if (error.bars == "conf.int")
          sds <- qt((1 - level)/2, df = ns - 1, lower.tail = FALSE) *
            sds/sqrt(ns)
        sds[is.na(sds)] <- 0
        yrange <- if (error.bars != "none")
          c(min(means - sds, na.rm = TRUE), max(means + sds,
                                                na.rm = TRUE))
        else range(means, na.rm = TRUE)
        levs.1 <- levels(factor1)
        levs.2 <- levels(factor2)
        n.levs.1 <- length(levs.1)
        n.levs.2 <- length(levs.2)
        if (length(pch) == 1)
          pch <- rep(pch, n.levs.2)
        if (length(col) == 1)
          col <- rep(col, n.levs.2)
        if (length(lty) == 1)
          lty <- rep(lty, n.levs.2)
        if (n.levs.2 > length(col))
          stop(sprintf(cat("Number of groups for factor2, %d, \nexceeds
number of distinct colours, %d."),
                       n.levs.2, length(col)))
        if (is.null(ylim)) {
          plot(c(1, n.levs.1 * 1.4), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main)
        }
        else {
          plot(c(1, n.levs.1 * 1.4), yrange, type = "n", xlab = xlab,
               ylab = ylab, axes = FALSE, main = main, ylim = ylim)
        }
        box()
        axis(2)
        axis(1, at = 1:n.levs.1, labels = levs.1)
        for (i in 1:n.levs.2) {
          points(1:n.levs.1, means[, i], type = "b", pch = pch[i],
                 cex = 2, col = col[i], lty = lty[i])
          if (error.bars != "none")
```

```
            arrows(1:n.levs.1, means[, i] - sds[, i], 1:n.levs.1,
                   means[, i] + sds[, i], angle = 90, code = 3,
                   col = col[i], lty = lty[i], length = 0.125)
        }
        x.posn <- n.levs.1 * 1.1
        y.posn <- sum(c(0.1, 0.9) * par("usr")[c(3, 4)])
        text(x.posn, y.posn, legend.lab, adj = c(0, -0.5))
        legend(x.posn, y.posn, levs.2, pch = pch, col = col,
               lty = lty)
      }
      invisible(NULL)
    }
    X11()
    bis(x1$porc.de.germinadas, as.factor(x1$Tratamento), error.bars=y, main
= "Grafico de Médias: Porcentagem", ylab="Porcentagem de Sementes Germinadas
(%)", xlab="Tratamentos")
    }
  }
  else
  {
    cat("Caso ocorra um erro ou objeto resultado não seja encontrado\n\t
          Verifique:\n 1)se a tabela esta no formato dos exemplos \n 2)se os
argumentos estão digitados de forma correta de acordo com o help \n 3) se
você tem instalado o pacote multcomp  \n\t\n\t
        " )}
  return(resultados)
}
```

Arquivos de Teste para Função

Tabela de germinação em diferentes qualidades de luz para *Eugenia umbelliflora*. Número de sementes por réplica =20

e.umbf.csv

Tabela de germinação em diferentes qualidades de luz para espécie com tendência fotoblástica positiva. Número de sementes por réplica= 20

f.pos.csv

Tabela de teste para verificar a se a função pode ser generalizada Tratamentos = A, B e C

testinho.csv

Arquivos da Função

funcao_measure.germ_.r

From:
http://ecor.ib.usp.br/ - **ecoR**

Permanent link:
http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2013:alunos:trabalho_final:thiagomitonori:start

Last update: **2020/08/12 06:04**