MÉRIDA
PUBLISHERS

para Entomologistas: nível básico

José Bruno Malaquias
Jéssica Karina da Silva Pachú
Tatiane Caroline Grella

2020

R para Entomologistas:

nível básico

José Bruno Malaquias

Jéssica Karina da Silva Pachú

Tatiane Caroline Grella

(Eds)

Canoas

2020

MÉRIDA PUBLISHERS

R para Entomologistas: nível básico

https://doi.org/10.4322/mp.2020-12

Editores
José Bruno Malaquias
Jéssica Karina da Silva Pachú
Tatiane Caroline Grella

Adaptação da capa e desenho gráfico
Reynaldo Miquel

Canoas - RS - Brasil
contact@meridapublishers.com
www.meridapublishers.com

Dados Internacionais de Catalogação na Publicação (CIP)
(eDOC BRASIL, Belo Horizonte/MG)

R111 R para entomologistas [recurso eletrônico] : nível básico / Organizadores José Bruno Malaquias, Jéssica Karina da Silva Pachú, Tatiane Caroline Grella. – Canoas, RS: Mérida Publishers, 2020.

Formato: PDF
Requisitos de sistema: Adobe Acrobat Reader
Modo de acesso: World Wide Web
Inclui bibliografia
ISBN 978-65-991393-4-5

1. Estatística. 2. Entomologia. 3. R (Linguagem de programação). I. Malaquias, José Bruno. II. Pachú, Jéssica Karina da Silva. III. Grella, Tatiane Caroline.

CDD 595.7

Elaborado por Maurício Amormino Júnior – CRB6/2422

Apresentação

A presente obra tem caráter introdutório, que se destina aos entomologistas que se iniciam no uso do programa R. Poderá ser explorado nos primeiros passos em programa R, como leitura de arquivos, instalação e carregamento de pacotes e análise descritiva, testes de normalidade e homogeneidade de variâncias, análise de variância, testes de comparação de médias e introdução a modelos lineares generalizados para dados de contagem.

O objetivo do livro é apenas apresentar ideias iniciais para cada abordagem citada anteriormente. Através de alguns exemplos de bancos de dados que foram tanto gerados de forma simulada quanto obtidos de forma experimental, apresentamos de forma gradativa a programação necessária para rodar cada análise.

No início de cada capítulo, há um resumo e uma sequência das linhas de comando comentadas e discutidas de forma prática e simples, cujo objetivo é nortear o usuário para iniciar o trabalho estatístico e que ele possa associar com bancos de dados e situações similares.

Esta obra nasceu da vivência, no dia a dia em cursos de extensão sobre R para entomologistas. Dessa forma, o presente trabalho, apesar de ser introdutório, é fruto de uma grande medida da colaboração de vários colegas, docentes e parceiros, a quem sinceramente prestamos nossos agradecimentos.

Diante disso, ressaltamos que o objetivo não tão somente ensinar o programa R ou estatística, mas propor um norte para aqueles interessados em iniciar seus trabalhos com o R e suas ferramentas.

Dr. José Bruno Malaquias

Instituto de Biociências
Departamento de Bioestatística da UNESP
Botucatu - SP, Brasil.

Índice

CAPÍTULO 1

Introdução ao ambiente R

Jéssica Karina da Silva Pachú, José Bruno Malaquias, Tatiane Caroline Grella

https://doi.org/10.4322/mp.2020-12.c1

Resumen

O R apresenta uma gama de ferramentas como testes paramétricos e não paramétricos, análise de regressão linear e não linear, análises de sobrevivência, análises multivariadas, produção gráfica, dentre outros. É importante mencionar que o R não é simplesmente um programa estatístico. No presente capítulo, apresentamos aspectos introdutórios ao ambiente R, como instalação do programa, carregamento e instalação de pacotes. Os comandos a serem executados no R estão sendo apresentados com a cor azul.
.
Palabras clave: linguagem R; apresentação; instalação; carregamento; pacotes.

1. Introdução

A estatística pode ser definida como a matemática aplicada aos dados de observação [1]. Através da Estatística conseguimos descrições claras, sintéticas e objetivas [2].

Em uma perspectiva entomológica, a estatística é uma condição essencial pois trabalha com métodos científicos para coleta, organização, resumo e apresentação de dados e também para obtenção de conclusões e a tomada de decisões.

As dificuldades enfrentadas por estudantes e pesquisadores entomologistas que precisam fazer uso da inferência estatística parecem decorrer da falta de livros didáticos, escritos em linguagem a eles adequada, livre de algebrismos usualmente empregados pelos estatísticos matemáticos. Para sanar essa lacuna, o presente livro tem como objetivo divulgar comandos básicos para análise descritiva, análise de variância e análise de deviance com dados entomológicos em linguagem R. A estratégia utilizada para alcançar esse objetivo consistiu na preparação de scripts com linguagem simples e com exemplos entomológicos.

O programa estatístico que tem mais auxiliado em análises estatísticas de dados biológicos é aquela baseada no ambiente R, que é uma linguagem de programação estatística, de acesso aberto, e que vem se tornando cada vez mais popular.

Em virtude de suas características, o *R Development Core Team* classifica como ambiente R [3].

O R é um software livre para análise de dados. Isto significa que ele pode ser utilizado, copiado, distribuído, alterado e melhorado de forma livre. Foi desenvolvido em 1996, com os professores de estatística Ross Ihaka e Robert Gentleman, eles trabalhavam e

eram colegas no departamento de estatística na Universidade de Auckland, os dois compartilhavam o interesse por estatística computacional e viram a necessidade de um melhor ambiente de software para o laboratório da área. Então, eles desenvolveram uma nova linguagem computacional, similar a linguagem S [3].

O software é um ambiente colaborativo internacional de desenvolvimento e pesquisa mantido formalmente pela R Foundation, disponibiliza uma ampla variedade de técnicas estatísticas e gráficas, incluindo modelagem linear e não linear, testes estatísticos clássicos, análise de séries temporais, classificação, agrupamento e outras. Você pode “baixá-lo" da Internet, de forma gratuita (www.r-project.org). Um conjunto básico de pacotes vem embutido na instalação do R, com muitos outros disponíveis na rede de distribuição do R (CRAN). Pesquisas e levantamentos com profissionais da área mostram que a popularidade do R aumentou substancialmente nos últimos anos [4]. Em 2017, o R já possui mais de 10.000 pacotes disponíveis [5]. Um pacote é um conjunto de funções que você consegue acessar em seus códigos, bastando importa-lo. Existem pacotes para resolver vários problemas como construção de gráficos, análise descritiva, análise de variância etc.

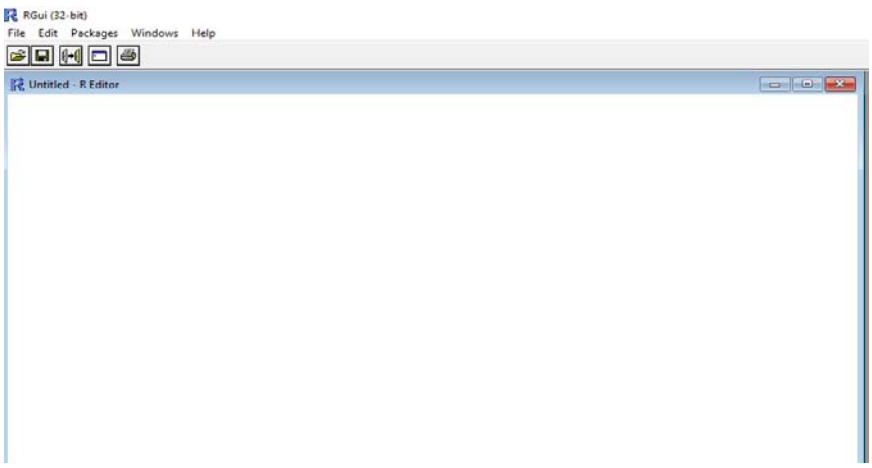
2. Motivos para utilizar o R

- Todos os códigos são abertos, reproduzíveis e adaptáveis;
- Compatível com Windows, Mac, Linux e etc;
- Utilizado para criar outros softwares;
- Empresas privadas e públicas estão se atentando ao R;
- Link com C, C++, Java entre outros;
- Ótima documentação (alterações de versões e help’s);
- Comunidade acadêmica disseminando conhecimento;
- Altamente extensível;
- É continuamente suportado pela comunidade, com rápida correção de *bugs* e adição de novas funcionalidades.

3. Instalação do R

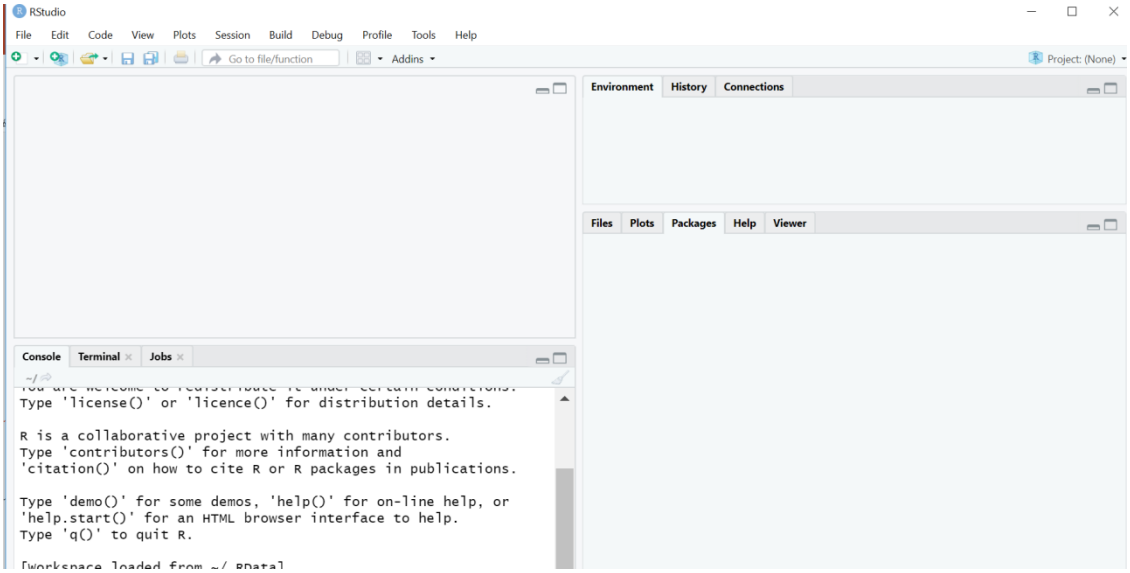
- Clique em CRAN;
- Escolha o espelho de sua preferência (CRAN mirrors);
- Clique em Windows;
- Clique em base e salve o arquivo do R para Windows;
- Depois é só executar o arquivo.

4. Console do software “R”



5. Usando o R como um ambiente de desenvolvimento integrado (IDEs)

RStudio é uma interface integrada para gráficos e cálculos estatísticos. Um acessório escrito na linguagem C++, está disponível em duas edições: RStudio Desktop, que roda localmente como um aplicativo desktop padrão; e RStudio Server, que permite acessar o RStudio usando um navegador web.



Para **baixar o RStudio** clique em: https://www.rstudio.com/products/rstudio/download/ e escolha opção *Rstudio desktop*.

6. Script

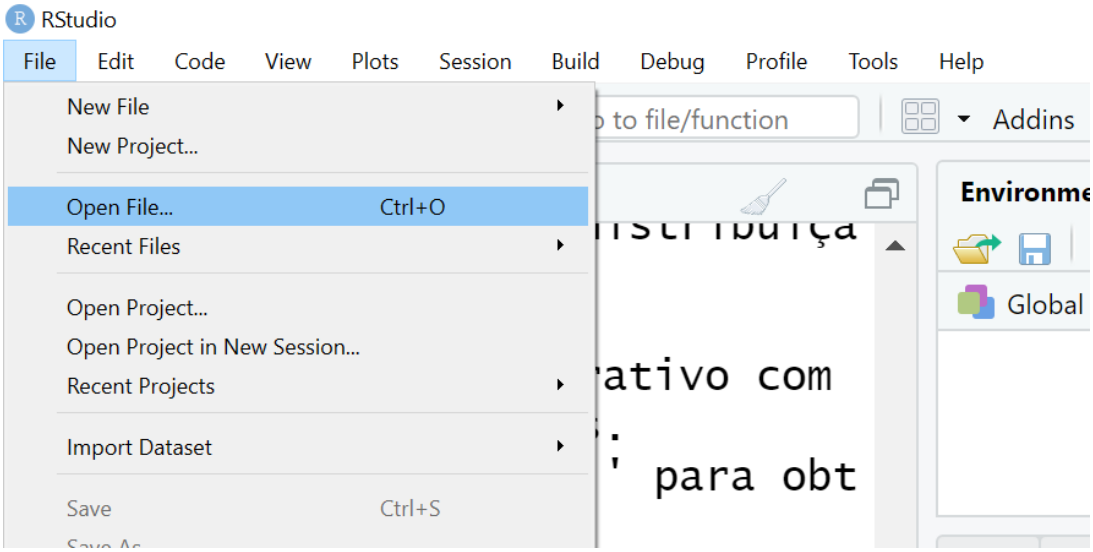
Para abrir e praticar no Rstudio com um novo script, vá em:

- File > New File >R Script

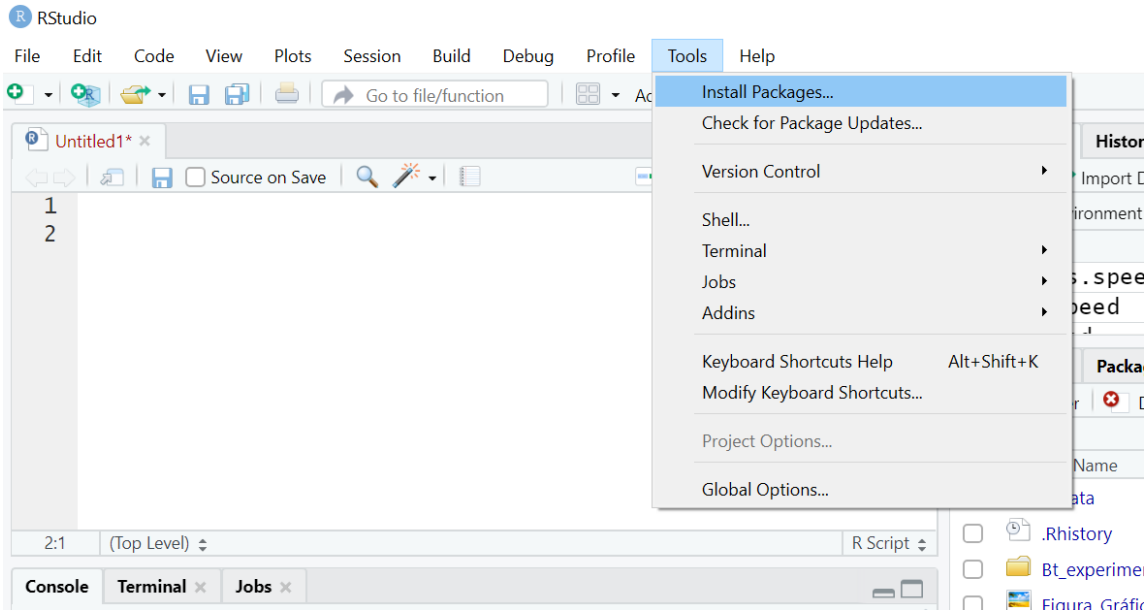
Se preferir abrir um script que já tenha salvo, vá em:

- Open File> selecione o arquivo > Abrir

No script você pode digitar comandos a serem executados e também comentários. Os comandos são escritos como no console e tudo que é escrito após # são considerados apenas como comentários.

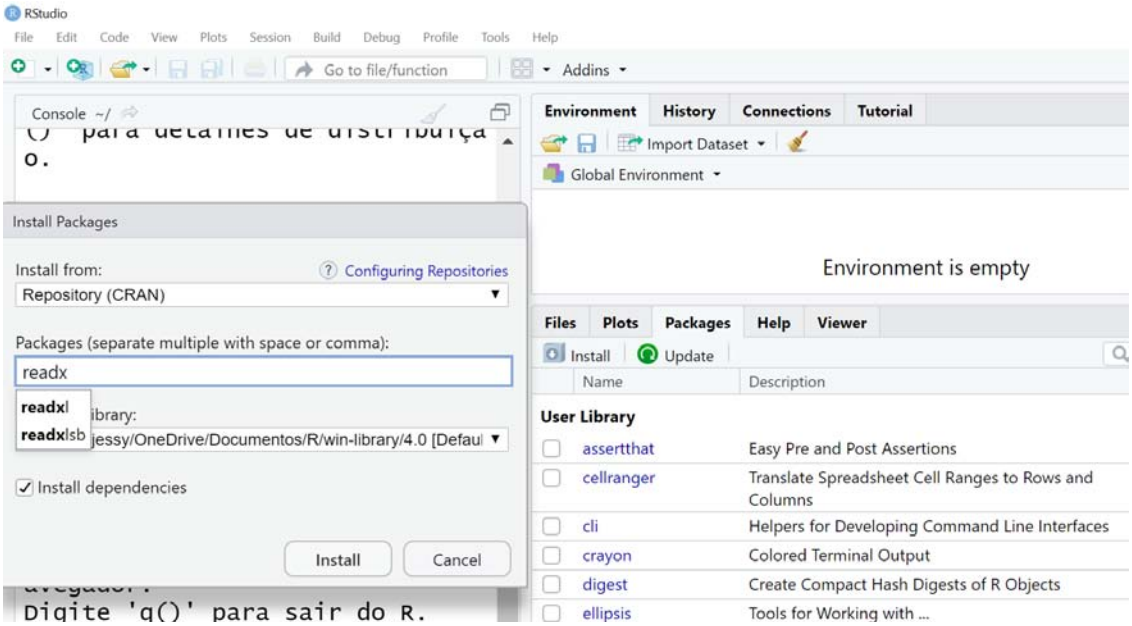


O R é um programa que ocupa pouco espaço e memória e geralmente roda rápido, isso porque os pacotes vêm na instalação base, ou seja, configurações mínimas para funcionar. Para realizar tarefas mais complexas pode ser necessário instalar pacotes adicionais (packages), esse é um dos grandes benefícios do R, seu grande acervo de pacotes. Qualquer pessoa pode enviar um pacote e todo código enviado está disponível na internet. Existe, porém, um processo de avaliação que o código passa e certas normas rígidas devem ser respeitadas sobre o formato do código, o manual do usuário e a forma de atualização do pacote.



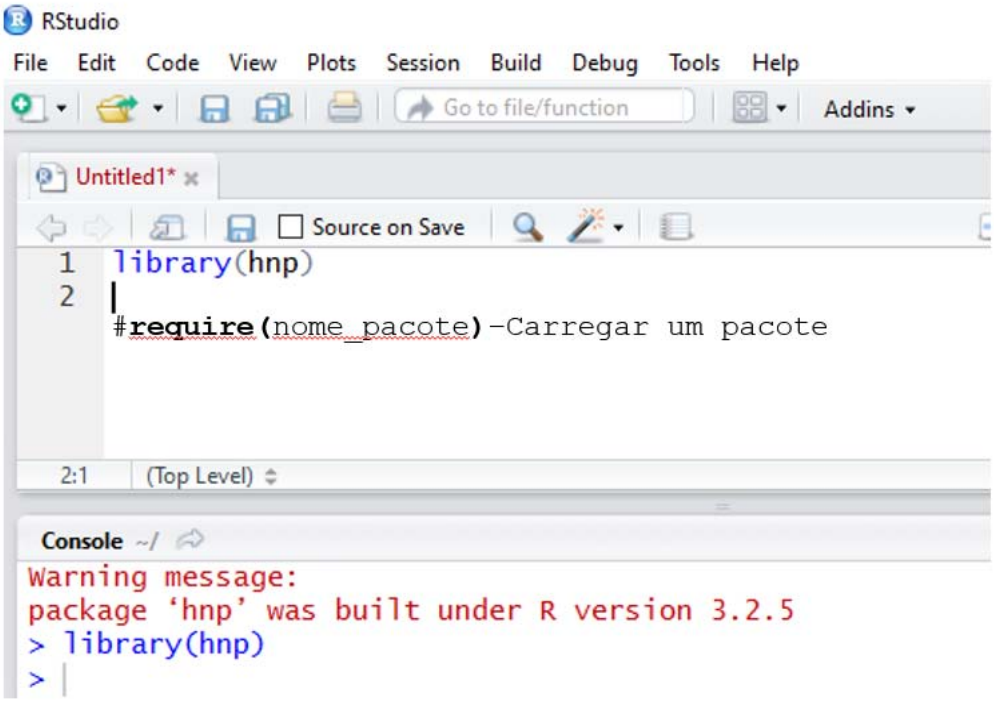
Outra forma é, no lado direito da tela, clicar em:

- Packages>Install> digitar o nome do pacote e instalar.



7. Usando o R como um ambiente de desenvolvimento integrado (IDEs)

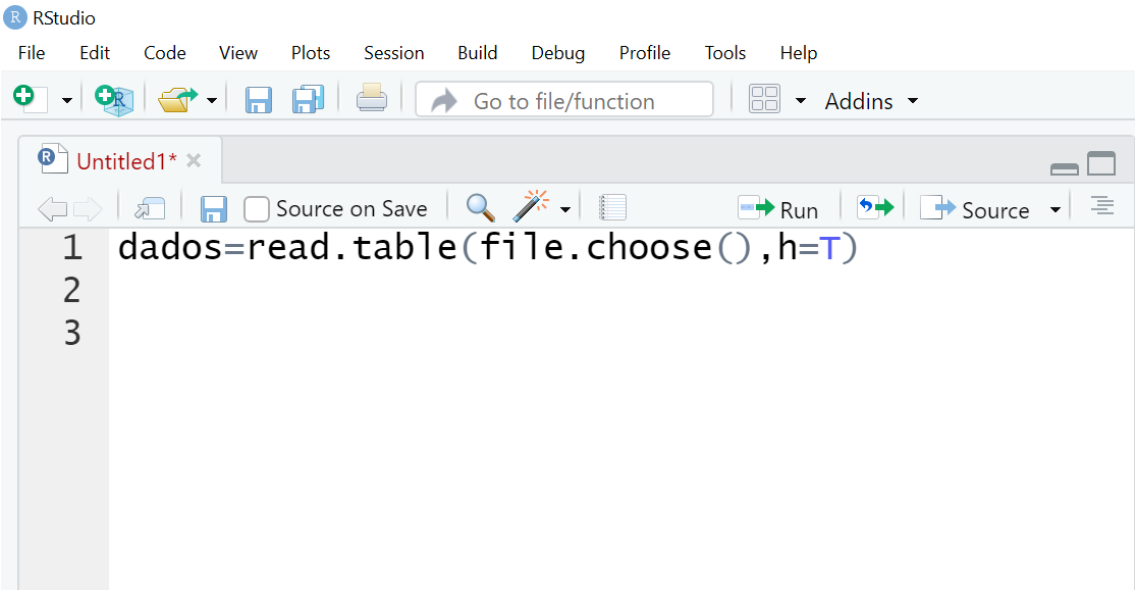
Para carregar um pacote dentro de um script, utilizamos a função `library()` ou `require()`, dentro do parêntese o nome do pacote que deverá ser carregado.



8. Lendo um arquivo

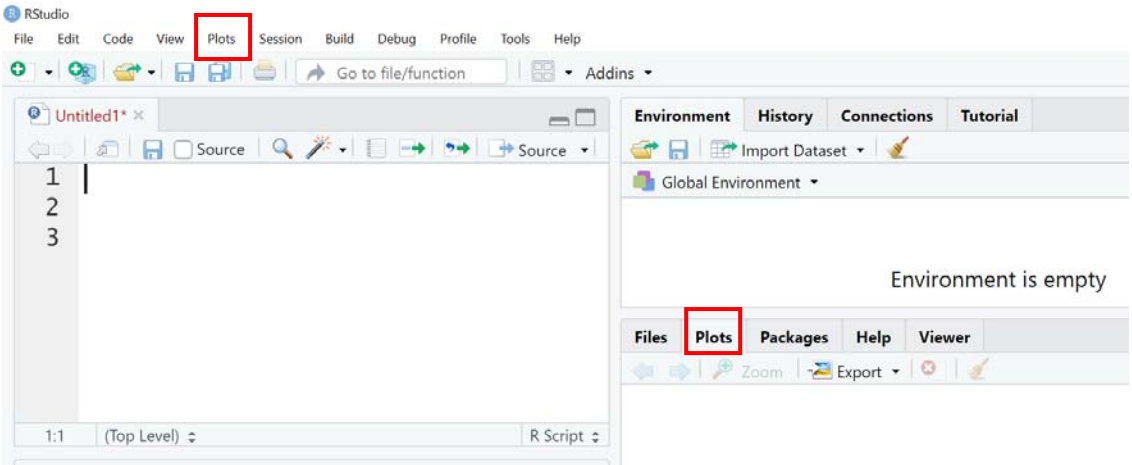
Para ler um banco de dados utilize a função **read.table** para fazer a importação dos dados.

A linha de comando exemplificada **dados=read.table(file.choose(),h=T** direciona você à pasta que contém o seu banco de dados no ambiente R (dados), a função para fazer a importação dos dados (read.table) e a função para escolher a pasta na qual seus dados estão (file.choose).



9. Visualizando um Plot

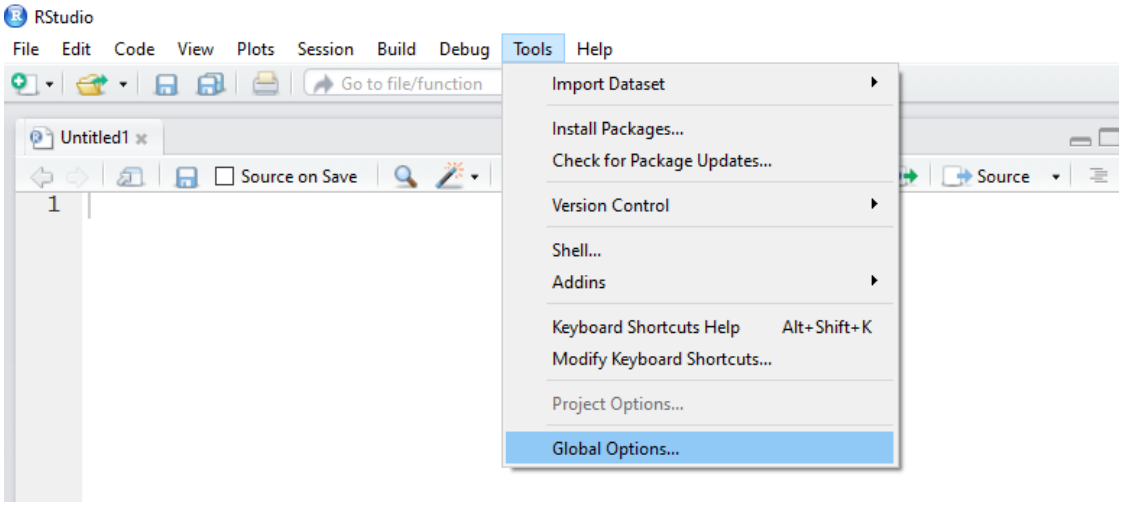
Para visualizar seus gráficos, clique na aba plots na linha superior do console ou na aba plots no lado direito, dessa maneira você pode usar a vassoura para limpar ou zoom para aumentar o seu plot.



10. Customize seu Rstudio

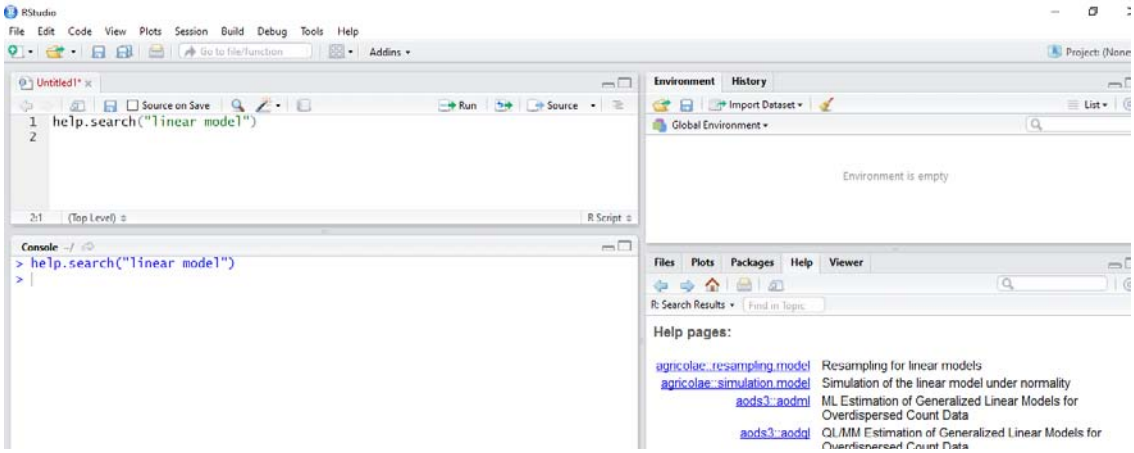
Nessa opção você pode alterar e personalizar o Rstudio de acordo com suas preferências, por exemplo, modificar a fonte, o tamanho, a cor etc. Clique em:

- Tools> Global Options>



11. Buscando ajuda

Existem algumas maneiras para obter ajuda sobre o funcionamento de comandos e pacotes. Por exemplo, clicando na aba Help ou digitando a linha de comando help.search ()



12. Algumas funções e comandos básicos

Para visualizar o banco de dados
View ()

Limpar o console
Ctrl+L

Para ler as linhas de comando
Ctrl+Enter ou Ctrl+R

Mudança de diretório
Ctrl+Shift+H:

getwd()

Para visualizar os nomes dos arquivos que tem dentro da pasta escolhida para trabalhar:
list.files()

13. Referências dos pacotes utilizados

[1] Pimentel-Gomes F. Curso de estatística experimental. 15. Ed. Piracicaba: FEALQ, 2009.

[2] Nazareth H. Curso básico de estatística. Ática, São Paulo, 2003.

[3] Peternelli L.A, Mello M.P. Conhecendo o R: uma visão estatística. Viçosa: UFV, v. 1, 2011.

[4] Muenchen R.A. The popularity of data analysis software. URL http://r4stats. com/popularity, 2012.

[5] Smith D. CRAN now has 10,000 R packages. Here's how to find the ones you need, 2017.

14. Referências recomendadas

Borcard D., Gillet F., Legendre P. Numerical ecology with R. Springer, 2018.

British Ecological Society. A guide to data management in ecology and evolution. 2014.

Crawley M.J. The R book. John Wiley & Sons, 2012.

Matloff Norman. The art of R programming: A tour of statistical software design. No Starch Press, 2011.

Peternelli L.A., Mello M.P. Conhecendo o R: uma visão estatística. Viçosa: UFV, v. 1, 2011.

Autores

Jéssica Karina da Silva Pachú*, Departamento de Entomologia e Acarologia - LEA Avenida Pádua Dias, 11 - CEP 13418-900 - Piracicaba/SP, Brasil

José Bruno Malaquias, Instituto de Biociências - Câmpus de Botucatu. R. Prof. Dr. Antônio Celso Wagner Zanin, 250 - Distrito de Rubião Junior - Botucatu/SP, Brasil - CEP 18618-689

Tatiane Caroline Grella, Biologia Celular e Molecular, Universidade Estadual Paulista "Júlio de Mesquita Filho" – Laboratório de Ecotoxicologia e Conservação de Abelhas (LECA) - Avenida 24 A,1515 - Bela Vista - CEP 13506-900 – Rio Claro/SP, Brasil

* Autor para correspondência: jessikapachu@gmail.com

CAPÍTULO 2

Análise descritiva e testes de pressuposições para análise de variância

Tatiane Caroline Grella, José Bruno Malaquias, Jéssica Karina da Silva Pachú

https://doi.org/10.4322/mp.2020-12.c2

Resumen

Neste capítulo, descrevemos de forma simples e objetiva elementos essenciais para análise descritiva no R por meio de boxplot e também exploramos algumas linhas de comando que são essenciais para testes de normalidade e homogeneidade de variâncias, para posterior análise de variância e testes de comparação de duas médias, no caso teste t, e comparação múltipla de médias – teste de Tukey. Todas as linhas de comando são apresentadas com um passo a passo de forma comentada e detalhada. Utilizaremos os seguintes pacotes: **readxl** e **ExpDes.pt**. Os comandos a serem executados no R estão sendo apresentados com a cor azul.
.
Palabras clave: boxplot; normalidade; homocedasticidade; variáveis contínuas; testes de comparação; ANOVA.

1. Leitura de arquivos no R

Para acessar o banco de dados e o script utilizados como exemplo neste capítulo **clique aqui**.

Para iniciar a análise é necessário fazer o download do banco de dados e script (disponibilizados a cima) e em seguida verificar em qual local do computador (diretório) o arquivo que será analisado se encontra.

Para isso, vamos usar o comando **getwd**, que irá verificar em qual diretório você está trabalhando: **getwd()**

Caso seja necessário alterar o diretório, basta seguir os passos:

Session -> Set working Directory -> Choose Directory

Após escolher um diretório é possivel ver quais os arquivos existem no mesmo, para isso, usamos a função: **list.files()** que mostrará a lista de arquivos dentro do seu diretório.

Após a escolha do diretório, vamos ler o arquivo que será analisado.

Para ler o arquivo em excel, nós iremos precisar do pacote: **readxl** [1].

Uma forma elegante de carregar o pacote é utilizar a seguinte linha de comando (essa linha também funciona caso o pacote não esteja instalado, pois automaticamente a instalação será realizada):

if(!require("readxl")) install.packages("readxl"); require(readxl)

O sinal de exclamação indica negação, então a linha de comando acima é traduzida como: *"caso o pacote necessário (readxl) nao esteja instalado, instale o pacote, e em seguida carregue-o"*.

2. Análise descritiva com box plot

Depois de carregar o pacote, precisamos ler o arquivo, para isso usaremos a linha de comando:
df<-read_excel("BD1.xls ", sheet = 1)

- df é o nome dado ao dataframe (você pode colocar o nome que desejar);
- read_excel é o comando para ler o arquivo do Excel
- **BD1** é o nome do seu arquivo dentro da pasta selecionada
- xls é a extensão do arquivo com o qual você está trabalhando
- sheet = 1 faz referência a qual aba do seu arquivo do Excel quer analisar

Vamos ler o Arquivo: **BD1.xls**.

Nesse exemplo didático foi analisado o efeito de dois tratamentos no peso dos insetos (Tabela 1). O delineamento experimental foi inteiramente casualizado e com apenas 3 repetições.

Tabela 1 – Peso (g) de uma espécie hipotética de inseto submetida a dois tratamentos

TRATAMENTO	REPETIÇÃO	PESO
A	1	0,0750
A	2	0,0810
A	3	0,0820
B	1	0,0780
B	2	0,0790
B	3	0,0800

Para ver o cabeçalho usamos a função: **head(df)**
Para ver o banco de dados, usamos a função: **View(df)**
Para expressar os dados em um boxplot e construi-lo, usamos a função:

boxplot(PESO~TRATAMENTO, data=df)

Após a construção é possivel alterar diversos itens, como:

- nome nos eixos, usando o comando:

```
boxplot(PESO~TRATAMENTO, data=df,
        xlab = "Tratamentos",
        ylab = "Matéria seca (kg)")
```

Neste exemplo os nomes do eixos são "Tratamentos" "Matéria seca (kg)"

- nome nos eixos e com limite inferior e superior.

```
boxplot(PESO~TRATAMENTO, data=df,
        xlab = "Tratamentos",
        ylab = "Matéria seca (kg)"
        , ylim = c(0.07,0.085))
```

- nome nos eixos e com limite inferior e superior e adicionando a média dentro do Boxplot.

```
boxplot(PESO~TRATAMENTO, data=df, col= "white",
      xlab= "Variedades", ylab= "Matéria Seca (Kg)", ylim = c(0.07,0.085))
points(1:nlevels(TRATAMENTO), tapply(PESO, TRATAMENTO, mean), data=df)
abline(mean(TRATAMENTO), data=df)
```

Para exportar a figura no formato desejado (uma sugestão é o formato tiff com 300 dpi, que geralmente é o formato solicitado pelos periódicos).

Para alterar os tamanhos, utilize: **cex.main** (título), **cex.lab** (rótulos) e **cex.axis** (tamanho dos eixos).

```
tiff("Figura_BoxPlot_.tiff", width=12, height=8, units="in", res=300)
boxplot(PESO~TRATAMENTO, data=df, col= "white",
      xlab= "Variedades", ylab= "Matéria Seca (Kg)",
      ylim = c(0.07,0.085),
      cex.main=1.5, cex.lab=1.5, cex.axis=1.5)
dev.off()
```

OBS: observe o seu diretório, pois a figura será diretamente exportada para essa pasta.
Para escolher o diretório utilize **Control + Shift + H** ou simplesmente use **getwd()**

A Figura 1 é o boxplot exportado. Cada seta expressa um parâmetro da análise descritiva, ou seja, os valores mínimo, máximo e quantis (1°, 2° e 3°).

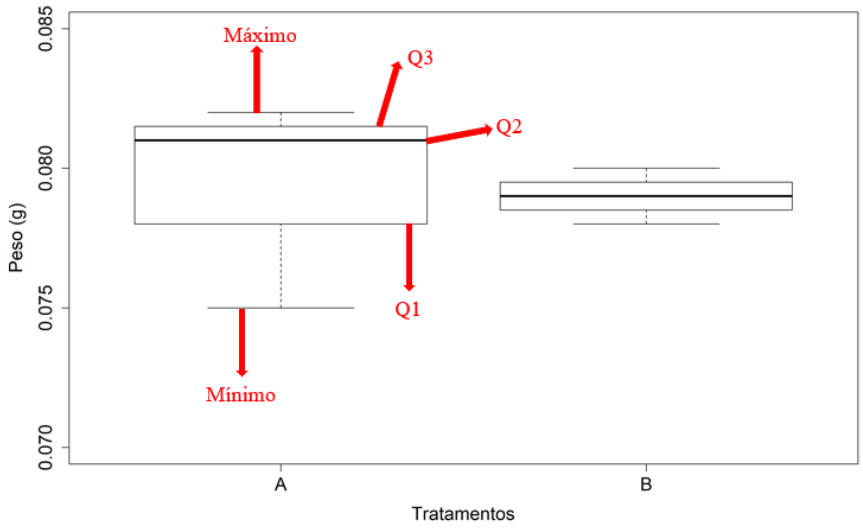


Figura 1 – Boxplot representando o efeito de dois tratamentos no peso (g) de uma espécie hipotética de inseto. **Q1**: primeiro quantil. **Q2**: segundo quantil ou mediana. **Q3**: terceiro quantil.

Com a utilização da linha de comanda abaixo, será possível visualizar os mesmos valores expressos no boxplot, ou seja, valor mínimo (Min), primeiro quantil (1st Qu.), mediana (Median), média (Mean), terceiro quantil (3rd Qu.) e máximo (Max), portanto, será demonstrado um resumo da análise descritiva do peso (g) de uma espécie hipotética de inseto submetida a cada um dos dois tratamentos.

tapply(df$PESO, df$TRATAMENTO, summary)

```
$A
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
0.07500 0.07800 0.08100 0.07933 0.08150 0.08200 

$B
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0780  0.0785  0.0790  0.0790  0.0795  0.0800 
```

3. Testando as pressuposições do modelo da ANOVA

Vamos utilizar o mesmo banco de dados que foi utilizado anteriormente, então leia o arquivo: **BD1.xls** (Tabela 1) utilizando o script que está disponível ao clicar aqui. Conforme exposto na Tabela 1 e comentado anteriormente, nesse exemplo didático foi analisado o efeito de dois tratamentos no peso dos insetos. O delineamento experimental foi inteiramente casualizado e com apenas 3 repetições.

Carregue o pacote **readxl**, para ler o arquivo do Excel:

require(readxl) caso tenha dúvidas em relação ao carregamento do pacote e instalação, veja o tópico “Leitura de arquivos no R”.

Leia o banco de dados: **df<- read_excel("BD1.xlsx", sheet = 1)**
Ver o cabeçalho: **head(df)**
Ver o banco de dados: **View(df)**

Antes de realizar a análise **ANOVA**, é necessário testar a normalidade e a homogeneidade das variâncias.

Para testar a normalidade, utilizamos o teste de Shapiro Wilk, usando:
shapiro.test(df$PESO)

Teste de Shapiro: Se o *p-value* for superior a 0,05 (5%), há normalidade dos dados.

Para testar a homogeneidade, em um estudo conduzido em DIC – com apenas um fator, utilizamos o teste de Bartlett, usando:
bartlett.test(df$PESO, df$TRATAMENTO)

Teste de Bartlett: Se o *p-value* for superior a 0.05, as variâncias são homogêneas.

OBS: caso seus dados sejam variáveis contínuas e não atendam à normalidade e/ou homogeneidade serão necessárias transformações.

4. Teste T

Vamos continuar utilizando o mesmo banco de dados que foi utilizado anteriormente, mas com alteração do script que podem ser acessados clicando aqui, então leia o Arquivo: **BD1.xls** (Tabela 1). Como são apenas dois tratamentos, iremos aplicar um teste t. Mas antes disso, precisaremos testar as pressuposições de normalidade e homogeneidade de variâncias, para maiores detalhes veja o tópico: “Testando as Pressuposições do Modelo da Anova”.

Carregue o pacote **readxl**, para ler o arquivo do Excel:

require(readxl) caso tenha dúvidas em relação ao carregamento do pacote e instalação, veja o tópico “Leitura de arquivos no R”.

Leia o banco de dados: **df<- read_excel("BD1.xlsx", sheet = 1)**

Ver o cabeçalho: **head(df)**

Ver o banco de dados: **View(df)**

Teste de Shapiro Wilk: **shapiro.test(df$PESO)**

Teste de Bartlett para um estudo conduzido em DIC – – com apenas um fator: **bartlett.test(df$PESO, df$TRATAMENTO)**

Para aplicar o teste t, utilize a função **t.test**.

- Peso é a variável resposta;
- Tratamento é a variável independente.

Pode-se utilizar as opções: "**two.sided**", "**less**" ou "**greater**".

t.test(PESO ~ TRATAMENTO, data = df, alternative = "two.sided")

5. Teste de Tukey

Vamos utilizar o banco de dados presente no Arquivo: **Anova1.0.xlsx** (Tabela 2). O delineamento experimental foi inteiramente casualizado e com 4 repetições. Como são mais de dois tratamentos, iremos aplicar um teste de comparações múltiplas, nesse caso o teste de Tukey. Lembre-se, de testar as pressuposições de normalidade e homogeneidade de variâncias, para maiores detalhes veja o tópico: Testando as Pressuposições do Modelo da Anova.

Tabela 2 – Diâmetro (mm) do *pronotum* de uma espécie hipotética de inseto submetida a três tratamentos

TRATAMENTO	REPETIÇÃO	DIÂMETRO
X	1	30
X	2	40
X	3	20
X	4	67
Y	1	20
Y	2	20
Y	3	35
Y	4	45
Z	1	60
Z	2	40
Z	3	50
Z	4	30

Carregue o pacote **readxl**, para ler o arquivo do Excel:

require(readxl) caso tenha dúvidas em relação ao carregamento do pacote e instalação, veja o tópico “Leitura de arquivos no R”.

Leia o banco de dados: **df<- read_excel("BD1.xlsx", sheet = 1)**

Ver o cabeçalho: **head(df)**

Ver o banco de dados: **View(df)**

Teste de Shapiro Wilk: **shapiro.test(df$Diametro)**

Teste de Bartlett para um estudo conduzido em DIC – com apenas um fator: **bartlett.test(df$Diametro, df$Tratamento)**

Para essa análise vamos utilizar o pacote ExpDes.pt [2]: **require(ExpDes.pt)**

Para enviar os comandos digitados (data frame) para memória: **attach(df)**

O ensaio foi conduzido no delineamento “dic”, por isso vamos utilizar a função **dic**

#em **mcomp**, escolha o método: "**tukey**"

dic(Tratamento, Diametro, quali = TRUE, mcomp = "tukey", nl=FALSE, sigT = 0.05, sigF = 0.05)

Esse é o resultado da análise:

```
De acordo com o teste F, as medias nao podem ser consideradas diferentes.
------------------------------------------------------------------------
  Niveis Medias
1      x  39.25
2      y  30.00
3      z  45.00
------------------------------------------------------------------------
```

A partir da análise, percebe-se que não existem evidências de diferenças entre os tratamentos. Caso queira apresentar os resultados usando letras, basta atribuir a mesma letra para as 3 médias – embora isto seja considerado redundante.

6. Referências dos pacotes utilizados

[1] WICKHAM H., BRYAN J. readxl: Read Excel Files. R package version 1.3.1. 2019. https://CRAN.R-project.org/package=readxl

[2] FERREIRA E.B., CAVALCANTI P.P., NOGUEIRA D.A. ExpDes.pt: Pacote Experimental Designs (Portuguese). R package version 1.2.0. 2018. https://CRAN.R-project.org/package=ExpDes.pt

7. Referências recomendadas

CRAWLEY, Michael J. The R book. John Wiley & Sons, 2012.

MATLOFF, Norman. The art of R programming: A tour of statistical software design. No Starch Press, 2011.

PETERNELLI, Luiz Alexandre; MELLO, MP de. Conhecendo o R: uma visão estatística. Viçosa: UFV, v. 1, 2011.

VENABLES W.N., RIPLEY B. D. Modern Applied Statistics with S. Fourth Edition. Springer, New York. 2002.

Autores

Tatiane Caroline Grella* - Biologia Celular e Molecular na Universidade Estadual Paulista "Júlio de Mesquita Filho" – Laboratório de Ecotoxicologia e Conservação de Abelhas (LECA) - Avenida 24 A,1515 - Bela Vista - CEP 13506-900 – Rio Claro/SP, Brasil

José Bruno Malaquias, Instituto de Biociências - Câmpus de Botucatu. R. Prof. Dr. Antônio Celso Wagner Zanin, 250 - Distrito de Rubião Junior - Botucatu/SP, Brasil - CEP 18618-689

Jéssica Karina da Silva Pachú, Departamento de Entomologia e Acarologia - LEA Avenida Pádua Dias, 11 - CEP 13418-900 - Piracicaba/SP, Brasil

* Autor para correspondência: tati_cg04@hotmail.com

CAPÍTULO 3

Análise de variáveis contínuas coletadas em experimentos com esquemas fatoriais

José Bruno Malaquias, Tatiane Caroline Grella, Jéssica Karina da Silva Pachú

https://doi.org/10.4322/mp.2020-12.c3

Resumen

Nesta seção, estamos expondo um exemplo de análise de banco de dados hipotético que foi delineado em blocos ao acaso e com estrutura de tratamento em esquema fatorial 2 x 2. Como a variável hipotética é contínua, então novamente exploramos algumas linhas de comando que são essenciais para testes de normalidade e homogeneidade de variâncias, para posterior análise de variância e testes de comparação de médias. Apresentamos todas as linhas de comando com um passo a passo de forma comentada e detalhada. Utilizaremos os seguintes pacotes: readxl, MASS e ExpDes.pt. Os comandos a serem executados no R estão sendo apresentados com a cor azul.

Palabras clave: análise fatorial; ANOVA; dois fatores; variável contínua; desdobramento; interação.

1. Importação do banco de dados

Após fazer o download dos arquivos utilizados neste capítulo ao **clicar aqui**, é necessário verificar em qual local do computador (diretório) o arquivo que será analisado se encontra.

Para isso, vamos usar o comando **getwd()**, que irá verificar em qual diretório você está trabalhando: **getwd()**

Caso seja necessário alterar o diretório, basta seguir os passos:

Session -> Set working Directory -> Choose Directory

Após escolher um diretório é possivel ver quais os arquivos existem no mesmo, para isso, usamos a função: **list.files()** que mostrará a lista de arquivos dentro do seu diretório.

Após a escolha do diretório, vamos ler o arquivo que será analisado.

Para ler o arquivo em excel, nós iremos precisar do pacote **readxl** [1].

Uma forma elegante de carregar o pacote é utilizar a linha de comando (essa linha também funciona caso o pacote não esteja instalado, pois automaticamente a instalação será realizada):

```
if (!require("readxl")) install.packages("readxl"); require(readxl)
```

O sinal de exclamação indica negação, então a linha de comando acima é traduzida como: *"caso o pacote necessário (readxl) nao esteja instalado, instale o pacote, e em seguida carregue-o"*

Vamos agora explorar uma outra forma de ler o banco de dados, mas utilizando o mesmo pacote. Primeiro vamos colocar o nosso **path** para leitura do arquivo, veja que atribui o meu **path** com o nome *MinhaPastaEArquivo* (cada um pode atribuir o nome que desejar):

MinhaPastaEArquivo <- "C:/Users/Bruno/Google Drive/Grupo de Discussão/Grupo - Linguagem R/Material-01082020/BD1-Anovatwoway.xlsx"

Veja que parte do **path** é pessoal: **"C:/Users/Bruno/Google Drive/Grupo de Discussão/Grupo - Linguagem R/Material-01082020**

É possível extrair essa parte, apenas rodando o comando **getwd(),** copiando e colando o que aparecer.

A outra parte extraímos do resultado de **list.files()**

Para mostrar os nomes das planilhas disponíveis, é necessário executar a seguinte linha de comando:
excel_sheets(path=MinhaPastaEArquivo)

Vamos utilizar agora a função **read_excel** para ler o arquivo e especificar a planilha que temos interesse.

df <-read_excel(path=MinhaPastaEArquivo, sheet = "Planilha1")

Veja que estamos atribuindo o nome do banco de dados agora de *df*
head(df): para ler o cabeçalho do dataframe.
View(df): para ver o banco de dados em outra janela
attach(df): para enviar o banco de dados para a memória

2. Testes de pressuposição da ANOVA

Para testar a homogeneidade de variâncias, iremos utilizar o teste de Bartlett, com a função **bartlett.test**

Vresp: é a variável resposta de interesse, ou variável dependente
Fator1: é o fator 1, ou a variável independente 1
Fator2: é o fator 1, ou a variável independente 2
Bloco: é o fator 1, ou a variável independente 2

bartlett.test(df$Vresp, df$Fator1)	teste de Bartlett para um estudo conduzido em DIC – – com apenas um fator.
bartlett.test(df$Vresp, df$Fator1, df$Bloco)	teste de Bartlett para um estudo conduzido em DBC – com apenas um fator.

bartlett.test(df$Vresp, df$Fator1, df$Fator2)	teste de Bartlett para um estudo conduzido em DIC com esquema fatorial (2 fatores).
bartlett.test(df$Vresp, df$Fator1, df$Fator2, df$Bloco)	teste de Bartlett para um estudo conduzido em DBC com esquema fatorial (2 fatores).

Teste de Bartlett: Se o *p-value* for **superior** a **0.05**, as variâncias são homogêneas.

Teste de Shapiro: **shapiro.test**
Se o *p-value* for **superior** a **0.05**, há normalidade dos dados.

3. ANOVA com dois fatores (two-way ANOVA)

Esse é o modelo que iremos trabalhar para two way-ANOVA:
Modelofatorial<-aov(ALT~FAT1*FAT2+BLOC, data=df)

Veja que usamos a função **aov** para rodar a ANOVA.
ALT: é a variável resposta.
FAT1: é o fator 1.
FAT2: é o fator 2.
BLOC: é o fator bloco.

Essas designações **ALT**, **FAT1**, **FAT2** e **BLOC** são provenientes do seguinte banco de dados:

FAT1	FAT2	BLOC	ALT
A	1	1	80
A	1	2	60
A	1	3	69
A	1	4	87
A	2	1	94
A	2	2	83
A	2	3	81
A	2	4	80
B	1	1	91
B	1	2	103

B	1	3	98
B	1	4	107
B	2	1	95
B	2	2	94
B	2	3	96
B	2	4	85

summary(Modelofatorial): mostra o resumo da ANOVA

```
            Df Sum Sq Mean Sq F value  Pr(>F)
FAT1         1 1139.1  1139.1  16.257 0.00198 **
FAT2         1   10.6    10.6   0.151 0.70523
BLOC         1    0.0     0.0   0.000 0.98958
FAT1:FAT2    1  315.1   315.1   4.497 0.05752 .
Residuals   11  770.7    70.1
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Veja que essa análise contém um *erro*. O erro desse *output* está no número de graus de liberdade para o fator **BLOC** (que corresponde a blocos). O número correto seria **3**. A fórmula de graus de liberdade é *n*-1, nesse caso temos 4 blocos, que perfazeria BLOC= 3.

Antes de seguir com a análise precisamos perguntar se é fator, porque temos variáveis dependentes e independentes, quando elas são dependentes precisa ser fator, para isso usamos o comando:

is.factor(df$FAT1)
is.factor(df$FAT2)
is.factor(df$BLOC)

Quando as variáveis são independentes é necessário converter, para isso precisamos utilizar a função **as.factor** e atualizar cada fator convertido

df$FAT1<-as.factor(df$FAT1)
df$FAT2<-as.factor(df$FAT2)
df$BLOC<-as.factor(df$BLOC)

ATENÇÃO: não faça esse procedimento de conversão em fator para as suas variáveis dependentes (variáveis respostas).

Após a conversão das variáveis independentes em fator, nós precisamos testar novamente se as variáveis são fatores:

is.factor(df$FAT1)
is.factor(df$FAT2)
is.factor(df$BLOC)

o resultado deverá ser **TRUE**

Agora que as suas variáveis independentes são fatores, você pode rodar a two-way ANOVA.

```
Modelofatorial<-aov(ALT~FAT1*FAT2+BLOC, data=df)
summary(Modelofatorial)
```

Perceba que o número de graus de liberdade do fator BLOC agora está correto!!

```
            Df Sum Sq Mean Sq F value  Pr(>F)
FAT1         1 1139.1  1139.1  14.813 0.00391 **
FAT2         1   10.6    10.6   0.137 0.71949
BLOC         3   78.7    26.2   0.341 0.79637
FAT1:FAT2    1  315.1   315.1   4.097 0.07362 .
Residuals    9  692.1    76.9
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Agora poderemos aplicar os testes de comparações de médias. Para isto, vamos utilizar a função **fat2.dbc**, vamos carregar o pacote com a nossa forma elegante:

```
if(!require("ExpDes.pt")) install.packages("ExpDes.pt"); require(ExpDes.pt)

fat2.dbc(FAT1, FAT2, BLOC, ALT, quali = c(TRUE, TRUE), mcomp = "tukey",
      fac.names = c("FATOR 1", "FATOR 2"), sigT = 0.05, sigF = 0.05)
```

Como a interação não foi significativa (podemos observar isso na linha “FAT1:FAT2”) serão analisados apenas os efeitos simples.

4. Outros exemplos

Para acessar o script e banco de dados, **clique aqui**. Vamos escolher o diretório no qual queremos trabalhar:
Session -> Set working Directory -> Choose Directory

Abra a pasta que contém o seu banco de dados.

Vamos usar o comando **list.files()** para visualizarmos os nomes de todos os arquivos que contém na pasta selecionada.

Quando seu banco de dados está no formato do Excel é necessário rodar o comando **df<-read_excel("BD.xlsx", sheet = 1)**, onde:

- df é o nome dado ao dataframe (você pode colocar o nome que quiser);
- read_excel é o comando para ler o arquivo do Excel
- BD. é o nome do seu arquivo dentro da pasta selecionada
- xlsx é a extensão do arquivo com o qual você está trabalhando
- sheet = 1 faz referência a qual aba do seu arquivo do Excel quer analisar

Para ver o cabeçalho do banco de dados a função utilizada é: **head(df, n=2)** neste caso, em específico, será visualizada apenas 2 linhas do seu banco de dados.

Antes de realizarmos os testes de normalidade e homogeneidade é necessário verificar se as variáveis independentes são fatores.
Neste caso, as variáveis são: Ins, Fung e Bloco

Ins	Fung	Bloco
A	X	1
A	X	2
A	X	3
A	X	4
B	X	1
B	X	2
B	X	3

Para realizar a verificação é necessário utilizar os comandos:
is.factor(df$Ins)
is.factor(df$Fung)
is.factor(df$Bloco)

A resposta para a verificação precisa ser: **TRUE**
Caso a resposta de: **FALSE**, é necessário converter as variáveis independentes em fatores, para isso utilizamos a função “as.factor”:
df$Ins<-as.factor(df$Ins)
df$Fung<-as.factor(df$Fung)
df$Bloco<-as.factor(df$Bloco)

em seguida é necessário verificar se a conversão foi bem sucedida:
is.factor(df$Ins)
is.factor(df$Fung)
is.factor(df$Bloco)

Em seguida é necessário selecionar qual variável deseja analisar

df$VRESP<-df$D

Neste caso, trabalharemos com a variável “D”, proveniente do banco de dado.

Atenção: o nome da variável deve ser idêntico ao escrito no banco de dado.

Depois de selecionarmos a variável que queremos analisar, podemos realizar os testes de normalidade e homogeneidade:

Para normalidade usamos o comando:

shapiro.test(df$VRESP)

Shapiro-Wilk normality test

data: df$VRESP
W = 0.95084, **p-value = 0.5031**

Para que os dados sejam considerados normais, o valor de "p" tem que ser maior que 0.05. Dessa forma podemos verificar que a análise em questão passou no teste de normalidade, pois p=0.5031.

Para homogeneidade usamos o comando:

bartlett.test(df$VRESP, df$Ins, df$Fung, df$Bloco)

```
Bartlett test of homogeneity of variances
data:  df$VRESP and df$Ins
Bartlett's K-squared = 2.3704, df = 1, p-value = 0.1237
```

Para que os dados sejam considerados homogêneos, o valor de "p" tem que ser maior que 0.05. Dessa forma podemos verificar que a análise em questão passou no teste de homogeneidade, pois p=0.1237.

OBS: caso os dados não atendam à normalidade e homogeneidade é necessário transforma-los, verifique como fazer isso mais a baixo.
Em seguida podemos rodar a ANOVA, para isso utilizaremos a função "aov":

Modelofatorial<-aov(VRESP~Ins*Fung+Bloco, data=df)
summary(Modelofatorial)

Em seguida fazemos um Summary para verificar o resultado do teste a cima:

```
            Df Sum Sq Mean Sq F value   Pr(>F)
Ins          1   2233    2233   41.25 0.000122 ***
Fung         1   1828    1828   33.77 0.000256 ***
Bloco        3   2841     947   17.50 0.000426 ***
Ins:Fung     1   7183    7183  132.72 1.09e-06 ***
Residuals    9    487      54
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Através do teste realizado podemos observar que existe diferença significativa nos grupos quando analisados separadamente e também na interação inseticida *versus* fungicida.

OBS: quando a interação não for significativa, não é necessário fazer desdobramento da interação inseticida *versus* fungicida.

Quando ocorre diferença significativa podemos rodar o teste de Tukey, o qual faz comparações múltiplas, para isso é necessário instalar o pacote "ExpDes" [2], utilizando o comando:

if(!require("ExpDes.pt")) install.packages("ExpDes.pt"); require(ExpDes.pt)

Esse comando verifica se você possui o pacote necessário e caso não possua, ele baixa e instala esse pacote.

Após a instalação do pacote, podemos rodar o teste de Tukey, com o seguinte comando:

```
fat2.dbc(Ins, Fung, Bloco, VRESP, quali = c(TRUE, TRUE), mcomp = "tukey",
    fac.names = c("Inseticida", "Fungicida"), sigT = 0.05, sigF = 0.05)
```

Esse comando já especifica que a significância dos resultados é de 95%
Como resultado do teste de Tukey, obtemos os desdobramentos:

Inseticida dentro do nível X de Fungicida

Grupos	Tratamentos	Médias
a	1	52.5
b	2	33.75

Inseticida dentro do nível Y de Fungicida

Grupos	Tratamentos	Médias
a	2	97.5
b	1	31.5

Fungicida dentro do nível A de Inseticida

Grupos	Tratamentos	Médias
a	1	52.5
b	2	31.5

Fungicida dentro do nível B de Inseticida

Grupos	Tratamentos	Médias
a	2	97.5
b	1	33.75

Letras diferentes indicam que houve diferença significativa entre os tratamentos.Para apresentarmos os resultados finais nos artigos, utilizamos essas letras, mas paraobtermos a média juntamente com uma medida de variabilidade desvio ou erro padrão que devemos apresentar é necessário instalar o pacote “Rmisc” e rodar um último co-mando:

```
if (!require("Rmisc")) install.packages("Rmisc"); require(Rmisc)

summarySE(df, measurevar = "VRESP", groupvars = c("Ins", "Fung"))
```

Obtemos como resultado:

Ins	Fung	N	VRESP	sd	se	ci
1	A	X 4	**52.50**	17.07825	**8.539126**	27.17531

2 A Y 4 **31.50** 18.33939 **9.169696** 29.18207
3 B X 4 **33.75** 13.76893 **6.884463** 21.90943
4 B Y 4 **97.50** 17.07825 **8.539126** 27.17531

Essa é a média — Esse é o erro padrão

Um exemplo de tabela que pode ser construída a partir dos resultados obtidos é:

	X	**Y**
A	52.50±8,53Aa	31.50±9,16Bb
B	33.75±6,88Bb	97.50±8,53Aa

Letras maiúsculas comparam colunas.
Letras minúsculas comparam linhas.

5. Caso que necessita de transformação

Clique aqui para acessar o banco de dados e script.

Vamos ler o banco de dados:

```
df<-read_excel("BD (1).xlsx", sheet = 1)
head(df, n=2)
```

Vamos testar as pressuposições:

```
shapiro.test(df$VRESP)
```

```
        Shapiro-Wilk normality test

data:  df$VRESP
W = 0.86057, p-value = 0.01957
```

```
bartlett.test(df$VRESP, df$Ins, df$Fung, df$Bloco)
```

```
        Bartlett test of homogeneity of variances

data:  df$VRESP and df$Ins
Bartlett's K-squared = 4.3204, df = 1, p-value = 0.03766
```

Perceba que os dados não atendarem às pressuposições de normalidade e homogeneidade de variâncias ($p < 0.05$). Portanto, é necessário transformá-los, para isso, precisaremos do pacote MASS [3] e usaremos o comando:

```
library(MASS)

Box = boxcox(VRESP ~ Ins*Fung+Bloco,
             data = df,
             lambda = seq(-6,6,0.1))

Cox = data.frame(Box$x, Box$y)
(Cox2 = Cox[with(Cox, order(-Cox$Box.y)),])
Cox2[1,]
(lambda = Cox2[1, "Box.x"]+0.0000001)
```

Para outros bancos de dados é necessário alterar:

Nomes das: variáveis dependentes e independentes.

Nome do dataframe.

(lambda = Cox2[1, "Box.x"]+0.0000001): essa linha de comando mostra o valor de lambda.

(df$VRESP_box = (df$VRESP ^ lambda - 1)/lambda): aqui inserimos a fórmula de Box-Cox [5].

Após a transformação devemos repetir os testes “Shapiro” e “Bartlett” para verificar se os dados atendem as pressuposições de normalidade e homogeneidade (SE ATENTAR A FORMA DE ESCREVER O COMANDO, POIS AGORA OS DADOS ESTÃO TRANSFORMADOS, ENTÀO É NECESSÁRIO ACRESCENTAR “box”).

```
shapiro.test(df$VRESP_box)
bartlett.test(df$VRESP_box, df$Ins, df$Fung, df$Bloco)
```

Após isso vamos rodar a análise de variância:

```
if(!require("ExpDes.pt")) install.packages("ExpDes.pt"); require(ExpDes.pt)

fat2.dbc(Ins, Fung, Bloco, VRESP_box, quali = c(TRUE, TRUE), mcomp = "tukey",
        fac.names = c("Inseticida", "Fungicida"), sigT = 0.05, sigF = 0.05)


if (!require("Rmisc")) install.packages("Rmisc"); require(Rmisc)
summarySE(df, measurevar = "VRESP", groupvars = c("Ins", "Fung"))
```

Quando é necessário fazer a transformação dos dados, devemos nos atentar para os valores de médias que colocamos na tabela dos resultados finais, pois elas não devem ser dos dados transformados e sim aqueles originais, os quais são obtidos na análise descritiva acima com a função summarySE do pacote Rmisc [4].

6. Referências dos pacotes utilizados

[1] Wicham H., Bryan J. readxl: Read Excel Files. R package version 1.3.1. 2019. https://CRAN.R-project.org/package=readxl

[2] Ferreira E.B., Cavalcanti P.P., Nogueira D.A. ExpDes.pt: Pacote Experimental Designs (Portuguese). R package version 1.2.0. 2018. https://CRAN.R-project.org/package=ExpDes.pt

[3] Venables W.N., Ripley B.D. Modern Applied Statistics with S. Fourth Edition. Springer, New York. 2002.

[4] Hope R.M. Rmisc: Rmisc: Ryan Miscellaneous. R package version 1.5. https://CRAN.R-project.org/package=Rmisc. 2013.

[5] Box G., Cox DR. An analysis of transformations. Journal of the Royal Society, 26: 211-252. 1964.

7. Referências recomendadas

Crawley, M.J. The R book. John Wiley & Sons, 2012.

Matloff Norman. The art of R programming: A tour of statistical software design. No Starch Press, 2011.

Peternelli L.A., Mello M.P. Conhecendo o R: uma visão estatística. Viçosa: UFV, v. 1, 2011.

Autores

José Bruno Malaquias*, Instituto de Biociências - Câmpus de Botucatu. R. Prof. Dr. Antônio Celso Wagner Zanin, 250 - Distrito de Rubião Junior - Botucatu/SP, Brasil - CEP 18618-689

Tatiane Caroline Grella - Biologia Celular e Molecular na Universidade Estadual Paulista "Júlio de Mesquita Filho" – Laboratório de Ecotoxicologia e Conservação de Abelhas (LECA) - Avenida 24 A,1515 - Bela Vista - CEP 13506-900 – Rio Claro/SP, Brasil

Jéssica Karina da Silva Pachú, Departamento de Entomologia e Acarologia - LEA
Avenida Pádua Dias, 11 - CEP 13418-900 - Piracicaba/SP, Brasil

* Autor para correspondência: malaquias.josebruno@gmail.com

CAPÍTULO 4

Aplicações de MLG para dados entomológicos de contagem

José Bruno Malaquias, Jéssica Karina da Silva Pachú, Filipe Lemos Jacques, Paulo Eduardo Degrande

https://doi.org/10.4322/mp.2020-12.c4

Resumen

Apresentamos nesta última seção, dois exemplos de dados que são de natureza discreta. Para isto, lançamos mão do teste de ajuste de Modelos Lineares Generalizados (MLG) com as seguintes distribuições: Poisson; quase-Poisson e Binomial Negativo. Um dos bancos de dados apresenta apenas um fator, enquanto o segundo banco de dados é constituído de um fatorial. Utilizaremos os seguintes pacotes do R: **readxl**, **hnp**, **MASS**, **multcomp** e **Rmisc**. Os comandos a serem executados no R estão sendo apresentados com a cor azul.

Palabras clave: deviance; variável discreta; número de insetos; contrastes; glm.

1. Importação do banco de dados – exemplo 1

O banco de dados a ser trabalhado, trata-se da ocorrência de *Scaptocoris castanea* Perty, 1830 (Hemiptera: Cydnidae) em diferentes culturas. Para acessar o banco de dados clique aqui, o banco de dados também está exposto no anexo desse capítulo.

- *Host* é a planta hospedeira;
- *Rep* corresponde a bloco (repetição) e
- *y* é a variável resposta que foi estudada como sendo a ocorrência dos insetos.

Antes de iniciarmos a análise devemos rodar o comando: **rm(list=ls(all=TRUE))** que serve para limpar a memória do programa e evitar erros.

Depois de alterar o nosso diretório (selecionar a pasta que contém os arquivos que utilizaremos): **Session -> Set working Directory -> Choose Directory**

Podemos visualizar os nomes de todos os arquivos que contén na pasta selecionada: **list.files()**

Em seguida devemos usar o comando para carregar o pacote que lê a planilha do Excel, que é o pacote readxl [1]: **require(readxl)**

Depois de carregar o pacote, precisamos ler o arquivo, para isso usaremos a linha de comando:

data<-read_excel("Dados-Abundancia.xlsx", sheet = 1)

- data é o nome dado ao dataframe (você pode colocar o nome que quiser);
- read_excel é o comando para ler o arquivo do Excel;
- Dados-Abundancia é o nome do seu arquivo dentro da pasta selecionada;
- xlsx é a extensão do arquivo com o qual você está trabalhando;
- sheet = 1 faz referência a qual aba do seu arquivo do Excel quer analisar.

2. Análise descritiva

O primeiro passo é a produção de uma análise descritiva, para isto utilizaremos o pacote **Rmisc** [2], para carregar o pacote usaremos o comando: **require(Rmisc).**

Em seguida com a função **summarySE**, iremos obter os valores médios, desvio padrão, erro padrão e intervalos de confiança associados à média, para isso devemos rodar a linha de comando:

(descritiva<- summarySE(data, measurevar="y", groupvars=c("Host")))

O resultado da análise descritiva segue abaixo:

	Host	N	y	sd	se	ci
1	Algodão	10	194.8	53.1722568	16.8145440	38.0371411
2	Brachiaria	10	17.5	6.7535999	2.1356758	4.8312343
3	Crotalaria	10	25.5	19.9568980	6.3109253	14.2763048
4	Girassol	10	0.6	0.6992059	0.2211083	0.5001818
5	Guandu	10	1.9	1.7288403	0.5467073	1.2367379
6	Milho	10	366.2	98.5391067	31.1608016	70.4906305
7	Mucuna Preta	10	1.6	1.8378732	0.5811865	1.3147353
8	Pousio	10	0.2	0.4216370	0.1333333	0.3016210
9	Soja	10	58.3	29.4016250	9.2976102	21.0326555
10	Sorgo	10	1.6	2.0110804	0.6359595	1.4386403

O próximo passo é a escolha do modelo que melhor se ajusta a esse conjunto de dados.

3. Teste de ajuste dos modelos

Quando a variável é independente precisamos verificar se as mesmas são fatores, para isso utilizamos o comando:

is.factor(data$Host)

A resposta para a verificação precisa ser: **TRUE**

Caso a resposta de: FALSE, é necessário converter as variáveis independentes em fatores, para isso utilizamos a função "as.factor": **data$Host<-as.factor(data$Host)**

Em seguida é necessário verificar se a conversão foi bem sucedida: **is.factor(data$Host)**

Com a conversão bem sucedida, podemos analisar qual dos 4 modelos melhor se ajusta aos dados, para isso utilizaremos os modelos:

model1<-glm(y~Host+Rep, data=data): Modelo gaussiano (sem necessidade de testar esse modelo, pois ele é aplicado à variáveis contínuas.
model2<-glm(y~Host+Rep, family="poisson", data=data): Modelo Poisson.
model3<-glm(y~Host+Rep, family="quasipoisson", data=data): Modelo quase-Poisson.

Alguns modelos precisam de pacotes específicos (como o model4, modelo binomial negativo). Para isso é necessário carregar o pacote **MASS** [3] para trabalharmos com a função **glm.nb**

Require(MASS)
model4<-glm.nb(y~Host+Rep, data=data): Modelo Binomial Negativo.

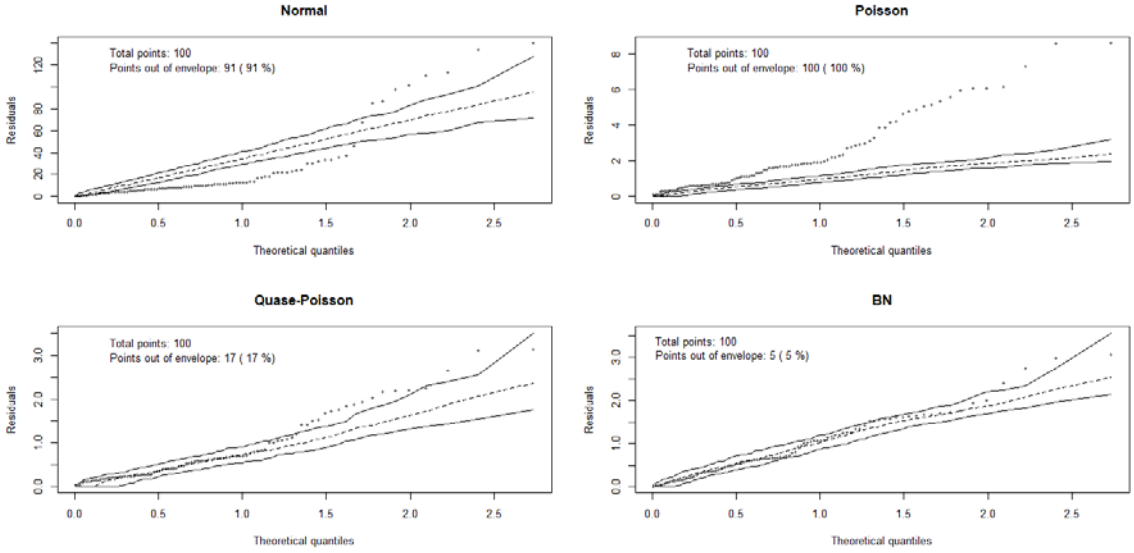
Em seguida é necessário carregar o pacote "**hnp**" para analisarmos a qualidade de ajuste dos modelos aos dados, usamos: **require(hnp)**, o pacote hnp foi desenvolvido por Moral et al., [4], para maiores detalhes sobre envelope simulado meio normal, consulte Demétrio [5] e Demétrio et al., [6].

Para visualizarmos os gráficos em outra janela usamos o comando: **dev.new()**
Para agrupar os 4 gráficos em um grid 2 x 2: **par(mfrow=c(2,2))**

Para testar o ajuste dos diferentes modelos utilizaremos diferentes linhas de comando:

hnp(model1, print.on="T", main="Normal"): Teste da qualidade de ajuste do modelo normal
hnp(model2, print.on="T", main="Poisson"): Teste da qualidade de ajuste do modelo Poisson
hnp(model3, print.on="T", main="QuasePoisson"): Teste da qualidade de ajuste do modelo quase-Poisson
hnp(model4, print.on="T", main="BN"): Teste da qualidade de ajuste do modelo Binomial Negativo

Esse é o resultado do ajuste dos modelos aos dados de contagem:



Percebe-se que o modelo **Binomial Negativo (BN)** foi o modelo que melhor se ajustou aos dados de contagem de *S. castanea,* pois ficaram apenas 5% dos pontos fora do envelope. Portanto, esse será o modelo selecionado.

O próximo passo é rodar o comando anova com teste F para mostrar a análise de *deviance* do modelo selecionado. Para isso, utilizaremos o comando:

anova(model4, test="F")

```
Analysis of Deviance Table

Model: Negative Binomial(3.3323), link: log

Response: y

Terms added sequentially (first to last)


     Df Deviance Resid. Df Resid. Dev        F Pr(>F)
NULL                    99    1189.58
Host  9  1071.87        90     117.71 119.0965 <2e-16 ***
Rep   9     3.54        81     114.17   0.3935 0.9389
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Conforme análise de *deviance*, é possível verificar que existem evidências de diferença significativa entre as plantas hospedeiras (fator *host*), pois Pr (> F) é inferior a 0,05 (5%).

Agora vamos comparar os tratamentos utilizando o pacote “**multcomp**” [7], para isso vamos rodar a linha de comando: **require(multcomp)**

Para visualizar as diferenças nós utilizaremos a **função glht** e o método Tukey, para isso é necessário rodar o comando:

(Comparações <- summary(glht(model4, linfct = mcp(Host = "Tukey"))))
Comparações <- summary(Comparações, test = univariate())

Em seguida vamos imprimir os resultados expresando as letras para cada tratamento:

(Letras.mP<-cld(Comparações, level=0.05, Letters= c(LETTERS, letters), decreasing=TRUE))

A partir do teste a cima obteremos como resultado as comparações, as que possuírem as memas letras são consideradas significativamente iguais:

```
     Algodão    Brachiaria    Crotalaria      Girassol       Guandu        Milho
         "B"           "D"           "D"          "FG"          "E"          "A"
Mucuna Preta       Pousio          Soja         Sorgo
        "EF"          "G"           "C"          "EF"
```

4. Importação do banco de dados – exemplo 2

Exemplo 2 – Dados de contagem em um ensaio com estrutura de fatorial que tratou da contagem de *Spodoptera frugiperda* (Lepidoptera: Noctuidae) em diferentes plantas hospedeiras (fator 1) em dois estádios vegetativos (fator 2), para obter o banco de dados e script, **clique aqui**

Para ler os arquivos do Excel, precisamos do pacote "readxl" [1], para isso usamos o comando: **library(readxl)**

Para ler o banco de dados, usar o comando:

df <- read_excel("DadosContagem.xlsx", sheet = 1)

Para visualizar o cabeçalho: **head(df, n=2)**

Antes de seguir com as analises é necesario verificar se as variáveis sao fatores, para isso usamos o comando:

is.factor(df$Estadio): Perguntar se Estádio é fator
is.factor(df$Bloco): Perguntar se Bloco é fator
is.factor(df$Cultura): Perguntar se Estádio é fator

Caso a resposta seja "FALSE", é necessário converter as variáveis.

df$Estadio<-as.factor(df$Estadio): Conversão da variável Estadio a um fator
df$Bloco<-as.factor(df$Bloco): Conversão da variável Estadio a um fator
df$Cultura<-as.factor(df$Cultura): Conversão da variável Estadio a um fator

após a conversão é necessário verificar se deu certo.

is.factor(df$Estadio): Perguntar novamente se Estádio é fator
is.factor(df$Bloco): Perguntar novamente se Bloco é fator
is.factor(df$Cultura): Perguntar novamente se Estádio é fator

Se a resposta for "TRUE", podemos seguir com as análises

OBS: caso tenha dúvida sobre os passos descritos a cima, consultar os capítulos anteriores.

5. Teste de ajuste

Vamos testar o ajuste dos modelos Poisson, quase-Poisson e Binomial Negativo:

Modelo.poisson<-glm(SfrugTot~Estadio*Cultura+Bloco, family= "poisson", data=df)
Modelo.qpoisson<-glm(SfrugTot~Estadio*Cultura+Bloco, family= "quasipoisson", data=df)

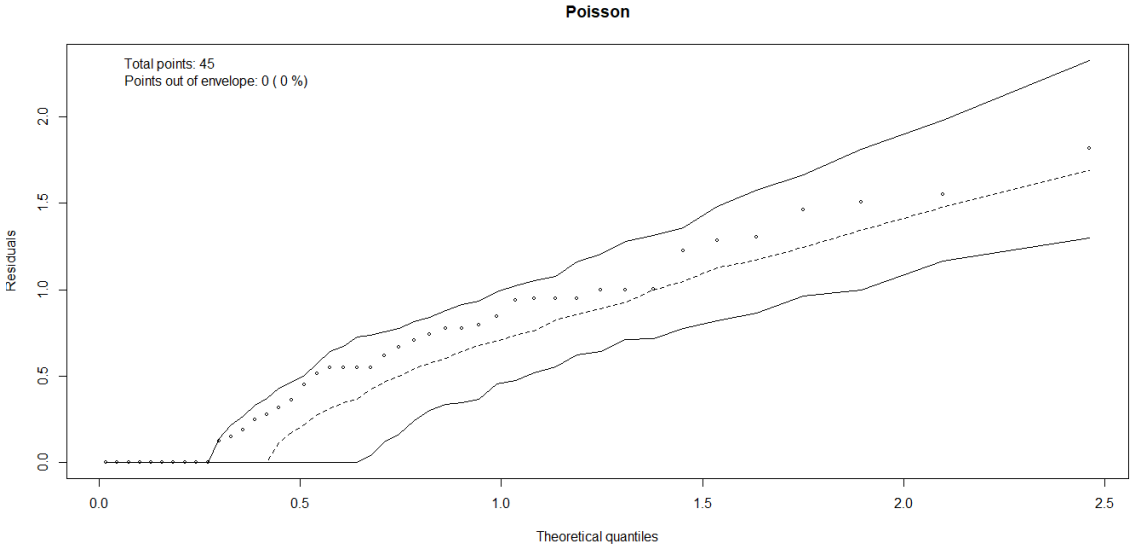
library(MASS) – lembre-se que é necessário carregar esse pacote MASS [2] para testar o modelo binomial negativo.
Modelo.BN<-glm.nb(SfrugTot~Estadio*Cultura+Bloco, data=df)

Agora vamos testar se o modelo se ajustou aos dados, para isso vamos rodar o pacote hnp [3] com o seguinte comando:

```
require(hnp)
```

```
hnp(Modelo.poisson, print.on="T", main="Poisson")
```

É possível observar que o modelo Poisson se ajustou aos dados, pois nenhum dos pontos ficou fora das linhas, portanto, utilizaremos esse modelo.



Após escolher o modelo, podemos rodar o comando ANOVA. Como o modelo de poisson se ajustou os dados, vamos usar o teste de qui-quadrado (Chisq):

```
anova(Modelo.poisson, test="Chisq")
```

Esse é o resumo da análise de deviance. É possível observar que houve interação significativa envolvendo os fatores Estádio e Cultura.

Analysis of Deviance Table - Model: poisson, link: log

Response: SfrugTot

```
Analysis of Deviance Table

Model: poisson, link: log

Response: SfrugTot

Terms added sequentially (first to last)


               Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                           44     89.170
Estadio         2   4.9117        42     84.259  0.085792 .
Cultura         2  18.6056        40     65.653 9.117e-05 ***
Bloco           4  13.4147        36     52.238  0.009418 **
Estadio:Cultura 4  24.2078        32     28.031 7.257e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

6. Desdobramento da interação

Como a interação foi significativa, vamos realizar o desdobramento da interação, precisamos conhecer os níveis de cada fator. Para isto vamos utilizar a função **sqldf** do pacote **sqldf** [8]

sqldf::sqldf("select distinct Estadio from df"): Verificar os níveis do fator Estadio
sqldf::sqldf("select distinct Cultura from df"): Verificar os níveis do fator Cultura

- Para o fator Estádio nós temos: V5, V7 e V8.
- Enquanto para Cultura nós temos: Milhoconv; MilhoBt e Urochloa

Vamos comparar primeiro os estádios dentro de cada cultura.

Primeiro, faremos um subset dentro de cada cultura:

```
EstadiosMC<-subset(df, df$Cultura=="Milhoconv")
EstadiosMBt<-subset(df, df$Cultura=="MilhoBt")
EstadiosU<-subset(df, df$Cultura=="Urochloa")
```

Depois, precisaremos criar um modelo para cultura

```
Modelo.EstadiosMC<-glm(SfrugTot~Estadio+Bloco, family= "poisson", data=EstadiosMC)

Modelo.EstadiosMBt<-glm(SfrugTot~Estadio+Bloco, family= "poisson", data=EstadiosMBt)

Modelo.EstadiosU<-glm(SfrugTot~Estadio+Bloco, family= "poisson", data=EstadiosU)
```

Para realizar as comparações, nós precisaremos do pacote "**multcomp**" [7], para isso vamos rodar a linha de comando (ela irá carregar ou instalar o pacote, caso necessário):

```
if(!require(multcomp)){install.packages("multcomp")}
```

Modelo para os dados de milho convencional:

```
Comp.EstadiosMC<-
summary(glht(Modelo.EstadiosMC,linfct=mcp(Estadio="Tukey")))
```

Modelo para os dados de milho Bt:

```
Comp.EstadiosMBt<-summary(glht(Modelo.EstadiosMBt,
infct=mcp(Estadio="Tukey")))
```

Modelo para os dados de Urochloa:

```
Comp.EstadiosU<-summary(glht(Modelo.EstadiosU,
linfct=mcp(Estadio="Tukey")))
```

Resultados das comparações na forma de probabilidade:

```
(Comp.EstadiosMC<-summary(Comp.EstadiosMC, test = univariate()))
```

```
Linear Hypotheses:
             Estimate Std. Error z value Pr(>|z|)
V7 - V5 == 0  -1.3863     0.7906  -1.754   0.0795 .
V8 - V5 == 0  -0.9808     0.6770  -1.449   0.1474
V8 - V7 == 0   0.4055     0.9129   0.444   0.6569
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Univariate p values reported)
```

(Comp.EstadiosMBt<-summary(Comp.EstadiosMBt, test = univariate()))

```
Linear Hypotheses:
                Estimate Std. Error z value Pr(>|z|)
V7 - V5 == 0  2.472e+01  8.153e+04       0        1
V8 - V5 == 0  3.203e-09  1.153e+05       0        1
V8 - V7 == 0 -2.472e+01  8.153e+04       0        1
(Univariate p values reported)
```

(Comp.EstadiosU<-summary(Comp.EstadiosU, test = univariate()))

```
Linear Hypotheses:
             Estimate Std. Error z value Pr(>|z|)
V7 - V5 == 0   1.7918     1.0801   1.659   0.0971 .
V8 - V5 == 0   2.8332     1.0290   2.753   0.0059 **
V8 - V7 == 0   1.0415     0.4749   2.193   0.0283 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Univariate p values reported)
```

A partir dos resultados das comparações na forma de probabilidade podemos notar que houve diferença significativa apenas na terceira comparação em “V8 - V5” e “V8 - V7”.

Para visualizar de forma mais clara, é possível atribuir letras para as diferenças significativas, com os comando a baixo:

(Letras.EstadiosMC<-cld(Comp.EstadiosMC, level=0.05, Letters= c(LETTERS, letters), decreasing=TRUE))

V5	V7	V8
"A"	"A"	"A"

(Letras.EstadiosMBt<-cld(Comp.EstadiosMBt, level=0.05, Letters= c(LETTERS, letters), decreasing=TRUE))

V5	V7	V8
"A"	"A"	"A"

(Letras.EstadiosU<-cld(Comp.EstadiosU, level=0.05, Letters= c(LETTERS, letters), decreasing=TRUE))

V5	V7	V8
"B"	"B"	"A"

Com a atribuição de letras podemos confirmar a diferença estatística que ocorre na última comparação.

Agora vamos comparar as culturas dentro de cada estádio.

Faremos um subset dentro de cada estádio:

```
CulturasV5<-subset(df, df$Estadio=="V5")
CulturasV7<-subset(df, df$Estadio=="V7")
CulturasV8<-subset(df, df$Estadio=="V8")
```

Posteriormente, necessitaremos criar um modelo para cada estádio.

```
Modelo.V5<-glm(SfrugTot~Cultura+Bloco, family= "poisson", data=CulturasV5)
Modelo.V7<-glm(SfrugTot~Cultura+Bloco, family= "poisson", data=CulturasV7)
Modelo.V8<-glm(SfrugTot~Cultura+Bloco, family= "poisson", data=CulturasV8)
```

Vamos realizar as comparações utilizando o mesmo método anterior, só contrastando o fator “Cultura”.

```
Comp.CulturasV5<-summary(glht(Modelo.V5, linfct=mcp(Cultura="Tukey")))
Comp.CulturasV7<-summary(glht(Modelo.V7, linfct=mcp(Cultura="Tukey")))
Comp.CulturasV8<-summary(glht(Modelo.V8, linfct=mcp(Cultura="Tukey")))
```

Resultados das comparações na forma de probabilidade:

```
(Comp.CulturasV5<-summary(Comp.CulturasV5, test = univariate()))
```

```
Linear Hypotheses:
                          Estimate Std. Error z value Pr(>|z|)
Milhoconv - MilhoBt == 0    22.425  15878.588   0.001   0.9989
Urochloa - MilhoBt == 0     20.345  15878.588   0.001   0.9990
Urochloa - Milhoconv == 0   -2.079      1.061  -1.961   0.0499 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Univariate p values reported)
```

```
(Comp.CulturasV7<-summary(Comp.CulturasV7, test = univariate()))
```

```
Linear Hypotheses:
                          Estimate Std. Error z value Pr(>|z|)
Milhoconv - MilhoBt == 0   -0.4055     0.9129  -0.444    0.657
Urochloa - MilhoBt == 0     0.6931     0.7071   0.980    0.327
Urochloa - Milhoconv == 0   1.0986     0.8165   1.346    0.178
(Univariate p values reported)
```

```
(Comp.CulturasV8<-summary(Comp.CulturasV8, test = univariate()))
```

```
Linear Hypotheses:
                          Estimate Std. Error z value Pr(>|z|)
Milhoconv - MilhoBt == 0  2.234e+01  2.485e+04   0.001  0.99928
Urochloa - MilhoBt == 0   2.407e+01  2.485e+04   0.001  0.99923
Urochloa - Milhoconv == 0 1.735e+00  6.262e-01   2.770  0.00561 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Univariate p values reported)
```

A partir dos resultados das comparações na forma de probabilidade podemos notar que houve diferença significativa apenas na primeira comparação em “Urochloa - Milhoconv” e na ultima comparação em “Urochloa - Milhoconv”.

Para visualizar de forma mais clara, é possível atribuir letras para as diferenças significativas, com os comando a baixo:

```
(Letras.CulturasV5<-cld(Comp.CulturasV5, level=0.05, Letters= c(letters, letters), decreasing=TRUE))
```

```
MilhoBt Milhoconv Urochloa
   "ab"       "a"       "b"
```

```
(Letras.CulturasV7<-cld(Comp.CulturasV5, level=0.05, Letters= c(letters, letters), decreasing=TRUE))
```

```
MilhoBt Milhoconv Urochloa
   "a"       "a"       "a"
```

```
(Letras.CulturasV8<-cld(Comp.CulturasV5, level=0.05, Letters= c(letters, letters), decreasing=TRUE))
```

```
MilhoBt Milhoconv Urochloa
   "ab"       "a"       "b"
```

Com a atribuição de letras podemos confirmar a diferença estatistica que ocorre na ultima comparação.

Agora vamos realizar uma análise descritiva, para isso precisamos do pacote “Rmisc” [2]. Vamos rodar a linha de comando a baixo, ela instalará o pacote, caso necessário.

`if(!require(Rmisc)){install.packages("Rmisc")}`: vamos utilizar o pacote “Rmisc”

Em seguida, fazemos um “summary”, para visualizar um resumo da análise:

```
summarySE(df, measurevar = "SfrugTot", groupvars = c("Estadio", "Cultura"))
```

```
Estadio   Cultura N SfrugTot        sd        se        ci
     V5   MilhoBt 5      0.0 0.0000000 0.0000000 0.0000000
     V5 Milhoconv 5      1.6 1.1401754 0.5099020 1.4157148
     V5  Urochloa 5      0.2 0.4472136 0.2000000 0.5552890
     V7   MilhoBt 5      0.6 1.3416408 0.6000000 1.6658671
     V7 Milhoconv 5      0.4 0.8944272 0.4000000 1.1105780
     V7  Urochloa 5      1.2 0.8366600 0.3741657 1.0388506
     V8   MilhoBt 5      0.0 0.0000000 0.0000000 0.0000000
     V8 Milhoconv 5      0.6 0.5477226 0.2449490 0.6800874
     V8  Urochloa 5      3.4 2.7018512 1.2083046 3.3547914
```

Os valores que não possuem variabilidade não podem ser considerados nas comparações. Os valores obtidos a partir da análise descritiva pode ser colocada em uma tabela da seguinte forma:

Cultura	Estádios		
	V5	V7	V8
Milho Bt	0.00±0.00 *	0.60±0.60 a A	0.00±0.00 *
Milho convencional	1.60±0.50 a A	0.40±0.40 a A	0.60±0.24 a A
Urochloa	0.20±0.20 b B	1.20±0.37 a B	3.40±1.20 b A

Letras minúsculas comparam culturas dentro do fator estádio (comparação dentro das colunas), enquanto letras maiúsculas comparam estádios dentro de cada cultura (comparação dentro das linhas). *Como não houve variabilidade nesse tratamento, o mesmo não será considerado nas comparações.

Anexo – Banco de Dados 1

Host	Rep	y
Brachiaria	1	13
Brachiaria	2	20
Brachiaria	3	24
Brachiaria	4	17
Brachiaria	5	22
Brachiaria	6	2
Brachiaria	7	17
Brachiaria	8	15
Brachiaria	9	26
Brachiaria	10	19
Crotalaria	1	36
Crotalaria	2	50
Crotalaria	3	1
Crotalaria	4	34
Crotalaria	5	2
Crotalaria	6	48
Crotalaria	7	6
Crotalaria	8	45
Crotalaria	9	27
Crotalaria	10	6
Algodão	1	216
Algodão	2	180
Algodão	3	197
Algodão	4	124
Algodão	5	180
Algodão	6	281
Algodão	7	203
Algodão	8	100
Algodão	9	228
Algodão	10	239
Soja	1	29
Soja	2	4

Soja	3	61
Soja	4	50
Soja	5	34
Soja	6	77
Soja	7	84
Soja	8	70
Soja	9	103
Soja	10	71
Milho	1	222
Milho	2	218
Milho	3	373
Milho	4	400
Milho	5	478
Milho	6	454
Milho	7	400
Milho	8	263
Milho	9	472
Milho	10	382
Guandu	1	2
Guandu	2	2
Guandu	3	4
Guandu	4	0
Guandu	5	2
Guandu	6	0
Guandu	7	5
Guandu	8	3
Guandu	9	0
Guandu	10	1
Mucuna Preta	1	0
Mucuna Preta	2	0
Mucuna Preta	3	0
Mucuna Preta	4	2
Mucuna Preta	5	4
Mucuna Preta	6	0
Mucuna Preta	7	1
Mucuna Preta	8	1
Mucuna Preta	9	5
Mucuna Preta	10	3
Sorgo	1	2
Sorgo	2	0
Sorgo	3	5
Sorgo	4	4
Sorgo	5	0
Sorgo	6	4
Sorgo	7	0

Sorgo	8	0
Sorgo	9	1
Sorgo	10	0
Girassol	1	2
Girassol	2	0
Girassol	3	1
Girassol	4	0
Girassol	5	1
Girassol	6	0
Girassol	7	0
Girassol	8	0
Girassol	9	1
Girassol	10	1
Pousio	1	0
Pousio	2	0
Pousio	3	1
Pousio	4	0
Pousio	5	1
Pousio	6	0
Pousio	7	0
Pousio	8	0
Pousio	9	0
Pousio	10	0

Anexo – Banco de Dados 2

Estadio	Cultura	Bloco	SfrugTot
V5	Milhoconv	1	2
V5	Milhoconv	2	0
V5	Milhoconv	3	1
V5	Milhoconv	4	2
V5	Milhoconv	5	3
V5	MilhoBt	1	0
V5	MilhoBt	2	0
V5	MilhoBt	3	0
V5	MilhoBt	4	0
V5	MilhoBt	5	0
V5	Urochloa	1	1
V5	Urochloa	2	0
V5	Urochloa	3	0
V5	Urochloa	4	0
V5	Urochloa	5	0
V7	Milhoconv	1	0
V7	Milhoconv	2	0
V7	Milhoconv	3	0

V7	Milhoconv	4	0
V7	Milhoconv	5	2
V7	MilhoBt	1	3
V7	MilhoBt	2	0
V7	MilhoBt	3	0
V7	MilhoBt	4	0
V7	MilhoBt	5	0
V7	Urochloa	1	1
V7	Urochloa	2	2
V7	Urochloa	3	1
V7	Urochloa	4	2
V7	Urochloa	5	0
V8	Milhoconv	1	1
V8	Milhoconv	2	1
V8	Milhoconv	3	0
V8	Milhoconv	4	0
V8	Milhoconv	5	1
V8	MilhoBt	1	0
V8	MilhoBt	2	0
V8	MilhoBt	3	0
V8	MilhoBt	4	0
V8	MilhoBt	5	0
V8	Urochloa	1	7
V8	Urochloa	2	3
V8	Urochloa	3	0
V8	Urochloa	4	2
V8	Urochloa	5	5

6. Referências dos pacotes utilizados

[1] Wickham H., Bryan J. readxl: Read Excel Files. R package version 1.3.1. 2019. https://CRAN.R-project.org/package=readxl

[2] Hope, R.M. Rmisc: Rmisc: Ryan Miscellaneous. R package version 1.5. https://CRAN.R-project.org/package=Rmisc. 2013.

[3] Venables W.N., Ripley B. D. Modern Applied Statistics with S. Fourth Edition. Springer, New York. 2002.

[4] Moral, R.A., Hinde, J., Demétrio, C.G.B. Half-normal plots and overdispersed models in R: The hnp package. Journal of Statistical Software, v. 81, n. 10, 2017.

[5] Demétrio, C.G.B. Modelos lineares generalizados em experimentação agronômica. USP/ESALQ, 2001.

[6] Demétrio, C.G., Hinde, J., Moral, R.A. Models for overdispersed data in entomology. In: Ecological modelling applied to entomology. Springer, Cham, 2014. p. 219-259.

[7] Hothorn, T., Bretz F., Westfall P. Simultaneous inference in general parametric models. Biometrical Journal 50(3), 346--363. 2008.

[8] Grothendieck, G. sqldf: Manipulate R Data Frames Using SQL. R package version 0.4-11. https://CRAN.R-project.org/package=sqldf. 2017.

6. Referências recomendadas

Demétrio, C.G.B. Modelos lineares generalizados em experimentação agronômica. USP/ESALQ, 2001.

Demétrio, C.G., Hinde, J., Moral, R.A. Models for overdispersed data in entomology. In: Ecological modelling applied to entomology. Springer, Cham, 2014. p. 219-259.

Autores

José Bruno Malaquias*, Instituto de Biociências - Campus de Botucatu. R. Prof. Dr. Antônio Celso Wagner Zanin, 250 - Distrito de Rubião Junior - Botucatu/SP, Brasil - CEP 18618-689

Jéssica Karina da Silva Pachú, Departamento de Entomologia e Acarologia - LEA Avenida Pádua Dias, 11 - CEP 13418-900 - Piracicaba/SP, Brasil

Filipe Lemos Jacques, Universidade Federal da Grande Dourados - Unidade I: Rua João Rosa Góes, nº 1761, Vila Progresso, Dourados/ MS, Brasil, CEP: 79825-070

Paulo Eduardo Degrande, Universidade Federal da Grande Dourados - Unidade I: Rua João Rosa Góes, nº 1761, Vila Progresso, Dourados/ MS, Brasil, CEP: 79825-070

* Autor para correspondência: malaquias.josebruno@gmail.com


MÉRIDA
PUBLISHERS