

**Universidade Federal de Santa Maria**  
**Centro de Ciências Rurais**  
**Departamento de Fitotecnia**

**Software R para avaliação de dados  
experimentais:**

Congregando a prática, estatística e computação

Bruno Giacomini Sari

Tiago Olivoto

Santa Maria, RS

2018

# Prefácio

Atualmente na área das Ciências Agrárias identificam-se o uso de diversos software para a análise estatística de dados originados em coletas de experimentos. Esta miscelânea de software pode confundir o pesquisador no momento de escolher qual é o software que será adotado para suas análises estatísticas, já que existem aqueles que devem ser adquiridas licenças para uso e nem todos disponibilizam opções de todos os métodos de análise estatística de dados.

Dentre esses, o R (R Development Core Team, 2008) destaca-se por ser uma linguagem de programação de código aberto (open source) basicamente destinado para computação estatística e gráficos. Com a proposta de organização de um curso e capacitação de acadêmicos e professores envolvidos em Pós-Graduação na Área de Ciências Agrárias, os Drs. Bruno Giacomini Sari e Tiago Olivoto propuseram-se a elaborar um documento onde oferecem uma excelente apresentação e introdução ao ambiente R, bem como diversas aplicações de abordagens estatísticas em experimentos agrícolas.

Assim este documento é uma resenha da introdução ao uso do R, bem como de modelos de análise e interpretação de experimentos agrícolas. Apresentam-se variações nos tipos de tratamentos (qualitativos e quantitativos), variações nos desdobramentos das interações e variações nas formas da casualização de experimentos bifatoriais. Seu intuito não é ensinar estatística.

A expectativa é de que este documento seja útil para aqueles pesquisadores que desejam utilizar este ambiente de programação para a realização de suas análises estatísticas e que sirva de consulta para o planejamento de experimentos com diferentes casualizações dos tratamentos.

Parabenizamos os autores pela iniciativa e qualidade do material oferecido.

Alessandro Dal'Col Lúcio  
Professor Titular, Setor de Experimentação Vegetal  
Departamento de Fitotecnia  
Centro de Ciências Rurais, Universidade Federal de Santa Maria

## Comissão Organizadora

### Alessandro Dal'Col Lúcio

Possui graduação em Agronomia pela Universidade Federal do Espírito Santo (1994), mestrado em Agronomia pela Universidade Federal de Santa Maria (1997), doutorado em Agronomia (Produção Vegetal) [Jaboticabal] pela Universidade Estadual Paulista Júlio de Mesquita Filho (1999) e pós-doutorado no Instituto Politécnico de Bragança [Portugal] (2015). É professor titular do Departamento de Fitotecnia do Centro de Ciências Rurais da Universidade Federal de Santa Maria e líder do grupo de pesquisa Experimentação registrado no CNPq. Atualmente é associado e ocupa o cargo de Conselheiro da Região Brasileira da Sociedade Internacional de Biometria - RBRAS, é membro da The International Biometric Society, da Associação Brasileira de Horticultura - ABH e da Sociedade Brasileira para o Progresso da Ciência - SBPC. Tem experiência na área de Probabilidade e Estatística, com ênfase em Experimentação Agrícola, atuando principalmente nos seguintes temas: planejamento de experimentos, precisão experimental, ambiente protegido, amostragem e variabilidade.



### Bruno Giacomini Sari

Possui graduação (2012), mestrado (2015) e doutorado (2018) em Agronomia pela Universidade Federal de Santa Maria - UFSM. Atualmente realiza estágio pós doutoral junto ao Programa de Pós Graduação em Agronomia da UFSM. Tem experiência na área de estatística, com ênfase em experimentação agrícola, atuando nos seguintes temas: probabilidade, amostragem, planejamento experimental, análise de regressão linear e não linear.



### Tiago Olivoto

Engenheiro agrônomo pela Universidade do Oeste de Santa Catarina (2014). Mestre em Agronomia: Agricultura e Ambiente pela Universidade Federal de Santa Maria (2017). Doutorando do Programa de Pós Graduação em Agronomia da Universidade Federal de Santa Maria. Exerce atividades relacionadas estatística experimental, planejamento de experimentos, com ênfase no desenvolvimento de métodos para melhorar a acurácia experimental. Em seu Currículo, os termos mais frequentes na contextualização da produção científica são: Manejo cultural, Seleção Indireta, Interação Genótipo x Ambiente, Modelos Mistas e Parâmetros Genéticos.



## Sumário

<b>1 Introdução ao ambiente <i>R</i></b>	<b>6</b>
1.1 Tipos de objetos . . . . .	10
1.1.1 Vetores . . . . .	10
1.1.2 Matrizes . . . . .	11
1.1.3 Data Frame . . . . .	13
1.1.4 Lista . . . . .	15
1.1.5 Funções . . . . .	17
1.1.6 Identificando os tipos de objetos . . . . .	18
1.2 Operações matemáticas e álgebra de matrizes . . . . .	19
1.3 Loops . . . . .	22
1.4 Construindo uma tabela . . . . .	23
1.5 Entrada de dados . . . . .	25
1.6 Funções gráficas . . . . .	26
1.6.1 Gráficos utilizando a função <code>plot()</code> . . . . .	28
1.6.2 Histogramas utilizando a função <code>hist()</code> . . . . .	33
1.6.3 Gráficos quantil-quantil utilizando a função <code>qqnorm()</code> . . . . .	33
1.6.4 Gráfico de caixas usando a função <code>boxplot()</code> . . . . .	36
1.6.5 Gráficos usando a função <code>curve()</code> . . . . .	36
1.7 Exportando dados . . . . .	40
1.7.1 Exportando com diferentes extensões . . . . .	40
1.7.2 Exportando gráficos . . . . .	42
<b>2 Análise de dados experimentais</b>	<b>46</b>
2.1 Estatística básica . . . . .	46
2.2 Intervalos de confiança e teste de hipóteses . . . . .	49
2.2.1 Intervalo de confiança . . . . .	51
2.2.2 Teste de hipóteses . . . . .	55
2.3 Delineamentos básicos . . . . .	60
2.3.1 Delineamento inteiramente casualizado (DIC) . . . . .	63
2.3.2 Delineamento blocos ao acaso (DBC) . . . . .	79
2.3.3 Transformação de dados . . . . .	84
2.4 Experimentos bifatoriais . . . . .	89
2.5 Download dos dados . . . . .	89
2.6 Qualitativo vs quantitativo . . . . .	91
2.6.1 Com interação significativa . . . . .	91
2.6.2 Sem interação significativa . . . . .	108
2.7 Qualitativo vs qualitativo . . . . .	115
2.7.1 Com interação significativa . . . . .	117
2.7.2 Sem interação significativa . . . . .	122
2.8 Quantitativo vs quantitativo . . . . .	127
2.9 Experimentos em parcelas subdivididas . . . . .	140
<b>3 Análise de regressão</b>	<b>142</b>
3.1 Regressão Linear . . . . .	142
3.1.1 Estimação . . . . .	143

3.1.2	Ajustando regressões com a função <code>lm()</code>	146
3.1.3	Seleção de variáveis	149
3.1.4	Falta de ajuste	155
3.1.5	Análise dos resíduos	157
3.1.6	Pontos influentes	159
3.2	Regressão não linear	163
3.2.1	Estimação	163
3.2.2	Ajustando o modelo com a função <code>nls()</code>	163
3.2.3	Análise dos resíduos	165
3.2.4	Medidas de não linearidade	169
3.2.5	Comparação de parâmetros	169
3.2.6	Representação gráfica dos modelos	173
3.3	Regressão bisegmentada com platô	173
<b>4</b>	<b>Biometria aplicada ao melhoramento genético de plantas</b>	<b>177</b>
4.1	Associação entre variáveis	177
4.2	Download dos dados	179
4.3	Correlação linear simples	179
4.3.1	Visualização gráfica	179
4.3.2	Estimativa dos coeficientes de correlação	180
4.3.3	Combinando visualização gráfica e numérica (I)	181
4.3.4	Combinando visualização gráfica e numérica (II)	182
4.3.5	Combinando visualização gráfica e numérica (III)	184
4.3.6	Combinando visualização gráfica e numérica (IV)	184
4.3.7	Intervalo de confiança não paramétrico	187
4.3.8	Planejamento do tamanho de amostra	188
4.4	Correlação parcial	189
4.5	Análise de trilha	191
4.5.1	Introdução	191
4.5.2	Estimação	192
4.5.3	Multicolinearidade	192
4.5.4	Métodos para ajustar a multicolinearidade	193
4.5.5	Análise tradicional	193
4.5.6	Incluindo um fator de correção (k)	196
4.5.7	Excluindo variáveis	200
4.5.8	Utilizando regressões stepwise para seleção de variáveis	203
4.6	Análise multivariada	209
4.6.1	Componentes principais	209
4.6.2	Análise de agrupamento hierárquico	216
4.6.3	k-means	229
4.7	Interação genótipo vs ambiente	229
4.8	Download dos dados	230
4.9	Compreendendo a interação genótipo vs ambiente	233
4.9.1	Análise individual	233
4.9.2	Médias e efeitos da interação	234
4.9.3	ANOVA conjunta	235

4.9.4	medidas de estabilidade . . . . .	236
4.9.5	Regressão individual . . . . .	237
4.10	Interação genótipo vs ambiente (AMMI) . . . . .	237
4.10.1	Ajustando o modelo . . . . .	239
4.10.2	Escolha do número de termos multiplicativos . . . . .	240
4.10.3	Biplots . . . . .	244
4.10.4	Predição da variável resposta . . . . .	248
4.11	Modelos mistos na avaliação de MET . . . . .	249
4.11.1	Ajustando o modelo . . . . .	249
4.11.2	Componentes de variância e parâmetros genéticos . . . . .	250
4.11.3	Detalhes sobre a análise . . . . .	251
4.11.4	Médias preditas . . . . .	251
4.12	AMMI ou BLUP? Decisão em cada caso! . . . . .	253
4.13	Combinando as vantagens dos métodos AMMI e BLUP . . . . .	255
4.13.1	Decomposição por valores singulares da matriz BLUP . . . . .	256
4.13.2	Interpretação conjunta da estabilidade e produtividade. . . . .	258
4.13.3	Atribuindo pesos para a estabilidade e variável resposta . . . . .	259
<b>Referências</b>		<b>264</b>

## **Lista de figuras**

1	Interface do RStudio . . . . .	6
2	Selecionando um diretório . . . . .	7
3	Iniciando um novo R Script . . . . .	7
4	Instalando pacotes . . . . .	8
5	Ajuda para a função <code>nls()</code> . . . . .	8
6	Verificando se há atualizações disponíveis . . . . .	9
7	Demonstração do teorema central do limite . . . . .	24
8	Importando dados de planilhas eletrônicas do Excel - Passo 1 . . . . .	26
9	Importando dados de planilhas eletrônicas do Excel - Passo 2 . . . . .	27
10	Importando dados de planilhas eletrônicas do Excel - Passo 3 . . . . .	27
11	Dividindo a tela para plotagem de múltiplos gráficos . . . . .	29
12	Alterando a coloração dos pontos ou linhas através do argumento <code>color()</code> . . . . .	31
13	Modificando a posição da legenda em um gráfico . . . . .	32
14	Gráficos de diagnóstico de resíduos utilizando a função <code>plot()</code> . . . . .	34
15	Histograma de frequências . . . . .	35
16	Gráficos quantil-quantil utilizando a função <code>qqnorm()</code> . . . . .	37
17	Gráfico de caixas usando a função <code>boxplot()</code> . . . . .	38
18	Gráficos usando a função <code>curve()</code> . . . . .	39
19	Gráficos usando a função <code>curve()</code> . . . . .	41
20	Salvando figuras como imagens . . . . .	44
21	Salvando figuras em PDF . . . . .	44
22	Gráfico dos valores observados vs uma distribuição teórica . . . . .	50
23	Gráficos de intervalo de confiança (uma variável) . . . . .	53
24	Gráficos de intervalo de confiança (mais de uma variável) . . . . .	54
25	Gráficos de intervalo de confiança (mais de uma variável) . . . . .	55
26	Gráfico de resíduos gerado pela função <code>plotres()</code> . . . . .	67
27	Gráfico de barras gerado pela função <code>ggplot()</code> . . . . .	71
28	Regressão quadrática obtida no delineamento DIC . . . . .	77
29	Gráfico de linhas gerado pela função <code>ggplot()</code> . . . . .	78
30	Gráfico gerado pela função <code>boxcox()</code> para identificar o valor de lambda . . . . .	86
31	Rendimento observado de cada híbrido em cada dose de nitrogênio . . . . .	92
32	Gráfico das médias dos híbridos em cada dose de nitrogênio gerado pela função <code>plot_fatmeans()</code> . . . . .	105
33	Gráfico com uma regressão ajustada para cada híbrido gerado pela função <code>plot_fatcurves()</code> . . . . .	107
34	Característica de produção em um experimento bifatorial sem interação significativa . . . . .	109
35	Curvas ajustadas para híbrido utilizando a função <code>plot_fatcurves()</code> . . . . .	116
36	Característica da produção em um experimento bifatorial sem interação significativa (agrupado por fonte de N) . . . . .	122
37	Característica da produção em um experimento bifatorial sem interação significativa (agrupado por fonte híbrido) . . . . .	123
38	Gráfico de superfície de resposta utilizando a função <code>plot_supresp()</code> . . . . .	134
39	Gráfico de contornos utilizando a função <code>plot_supresp()</code> . . . . .	138

40	Distância de Cook representando a influencia dos pontos na predição das variáveis	160
41	Manipulando os valores dos parâmetros . . . . .	165
42	Comportamento da produção acumulada em olerícolas . . . . .	170
43	Representação gráfica em modelos não lineares . . . . .	174
44	Representação gráfica de regressão bisegmentada com platô . . . . .	178
45	Scatter plot de uma matriz de correlação de Pearson . . . . .	183
46	Gráfico de pizza de uma matriz de correlação de Pearson . . . . .	186
47	Valores de beta obtidos com 101 valores de k . . . . .	197
48	Diagrama de trilha utilizando a função path.diagram() . . . . .	205
49	Autovalores da matriz de correlação obtidos pela função pca() . . . . .	211
50	Escores dos genótipos nos dois primeiros PCA obtidos pela função pca() . . . . .	212
51	Escores das variáveis nos dois primeiros PCA obtidos pela função pca() . . . . .	213
52	Escores dos genótipos e das variáveis nos dois primeiros PCA obtidos pela função pca() . . . . .	214
53	Escores dos genótipos e das variáveis nos dois primeiros PCA obtidos pela função pca() considerando os efeitos de ambiente . . . . .	216
54	Dendrograma gerado pela função distdend() . . . . .	219
55	Coeficiente de correlação cofenética em matrizes de distância estimadas com diferentes números de variáveis . . . . .	222
56	Procedimento bootstrap indicando o número de clusters a serem formados no dendrograma . . . . .	225
57	Dendrograma personalizado com cores indicando os grupos . . . . .	226
58	Árvore filogenética representando a matriz de distâncias . . . . .	227
59	Dendrograma circular representando a matriz de distâncias . . . . .	228
60	Gráfico para agrupamento k-means gerado pela função kmeans() . . . . .	231
61	Gráfico para agrupamento k-means gerado pela função kmeans() . . . . .	232
62	Régressão individual da produtividade com o índice ambiental . . . . .	238
63	Barra de progresso. . . . .	241
64	Gráfico boxplot demonstrando o RMSE obtido na validação cruzada dos models AMMI . . . . .	243
65	Biplot da produtividade de grãos com o PC1 gerado pela função plot.scores() . . . . .	245
66	Biplot do PC1 com o PC2 gerado pela função plot.scores() . . . . .	246
67	Biplot da produtividade de grãos com a média ponderada dos escores absolutos gerado pela função plot.scores() . . . . .	247
68	Médias preditas para genótipos considerando um modelo misto . . . . .	252
69	Biplot WAAS x RG (a) e WAASB x RG (b) . . . . .	259
70	Índice WAASBY proposto para a classificação dos genótipos de acordo com pesos atribuídos para produtividade e estabilidade . . . . .	261
71	Rankeamento dos genótipos dependendo do número de componentes principais considerados no modelo . . . . .	262
72	Rankeamento dos genótipos dependendo do peso considerado para a produtividade e estabilidade. . . . .	263

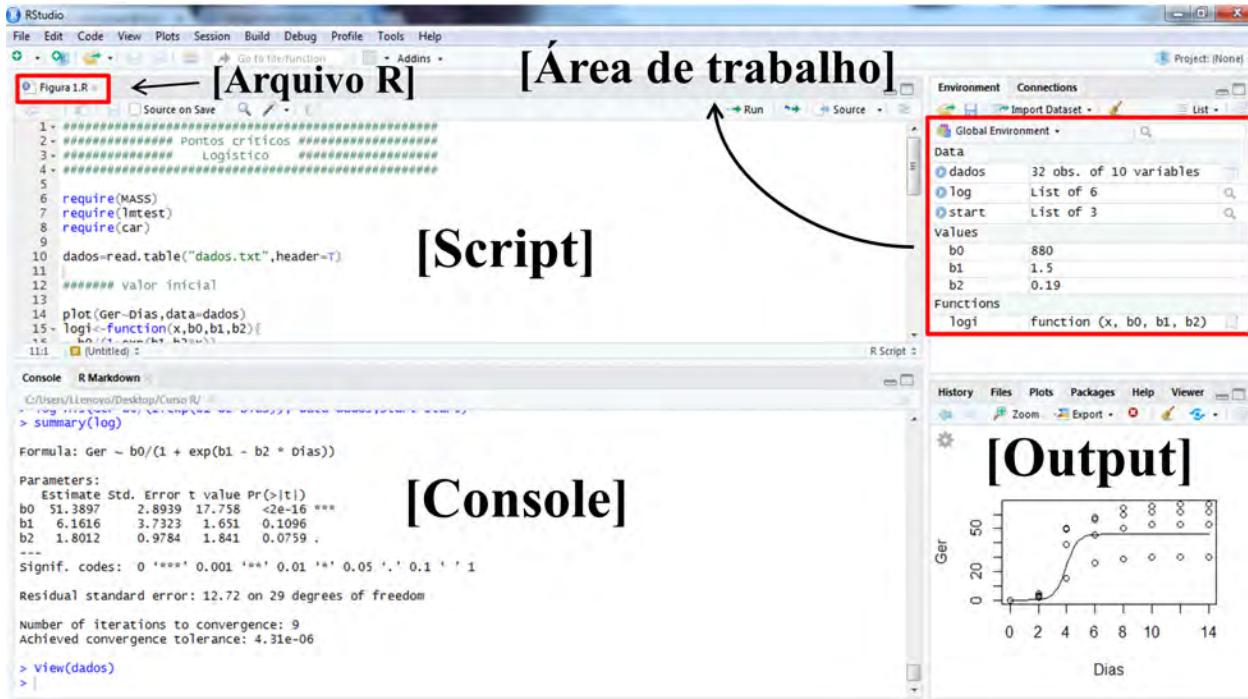


Figure 1: Interface do RStudio

## 1 Introdução ao ambiente *R*

Nesta seção serão abordados alguns aspectos básicos para que o usuário do *software R* possa desenvolver as suas análises. Será dado enfoque para áreas básicas da interface, cujo conhecimento é necessário para que um usuário iniciante possa fazer as suas análises. Em um primeiro momento, será explorada a interface do *software*, para que o usuário possa identificar o *script*, o *console*, a “área de trabalho” e o *output* para gráficos.

Antes de iniciar as análises, recomenda-se escolher um diretório onde devem estar *inputs* e para onde serão enviados os *outputs*. Para selecionar o diretório, basta seguir o caminho *Session > Set Working Directory > Choosing directory* ou utilizar as teclas de atalho *Ctrl+Shift+H*. Posteriormente, um *R Script* é iniciado conforme a figura abaixo ou pelas teclas de atalho *Ctrl+Shift+N*. No canto inferior direito, além de servir como *output* para gráficos (*Plots*), também é o local onde os pacotes utilizados nas análises (*Packages*) são instalados e maiores informações sobre as funções podem ser encontradas (*Help*). Para que certas funções possam ser utilizadas, os pacotes devem ser instalados previamente. Quando o pacote onde se encontra a função que o usuário deseja utilizar não está instalando, uma mensagem de erro (*Error in FUNÇÃO*) aparece no *console*, indicando que a função não foi encontrada (*could not find*). A instalação dos pacotes pode ser realizada conforme a figura abaixo, ou através da função *install.packages()*. Após a instalação do pacote, o usuário deve utilizar as funções *require()* ou *library()* para que o pacote seja carregado e as suas funções possam ser utilizadas.

Antes de inicial qualquer análise, os pacotes (com as funções que deseja-se utilizar) devem ser instalados e carregados. Em um primeiro momento, será mostrado como instalar e carregar

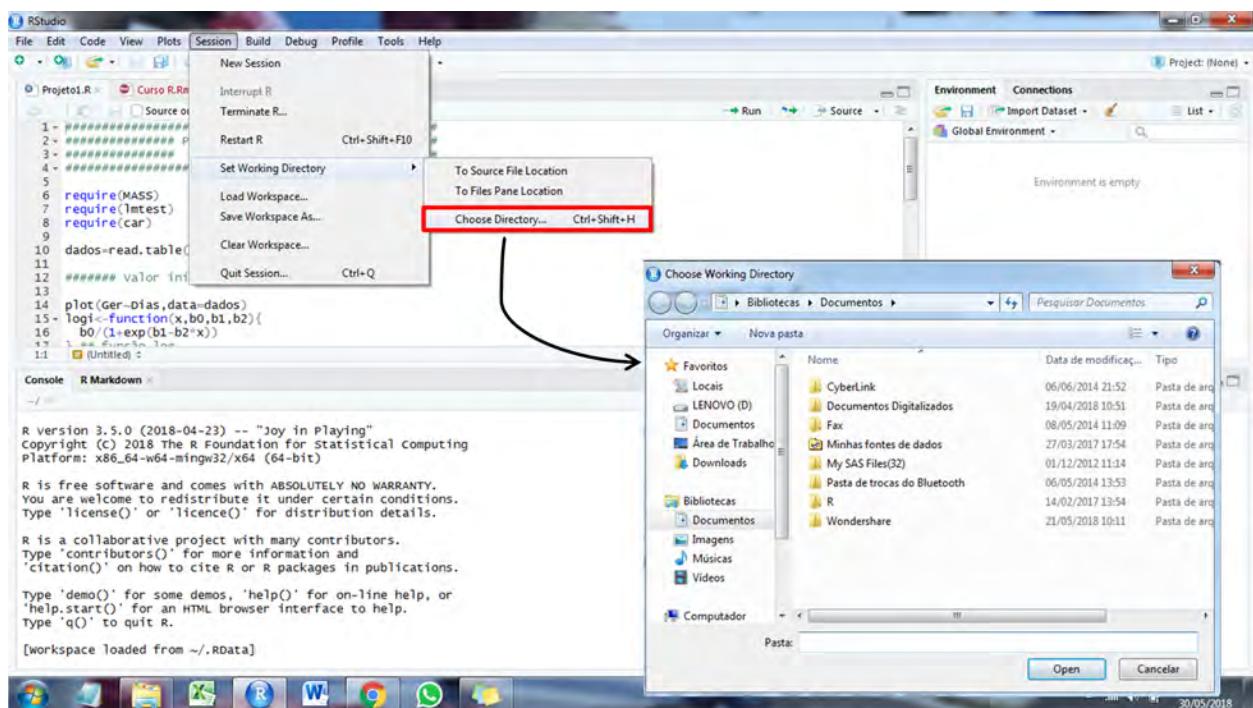


Figure 2: Selezionando um diretório

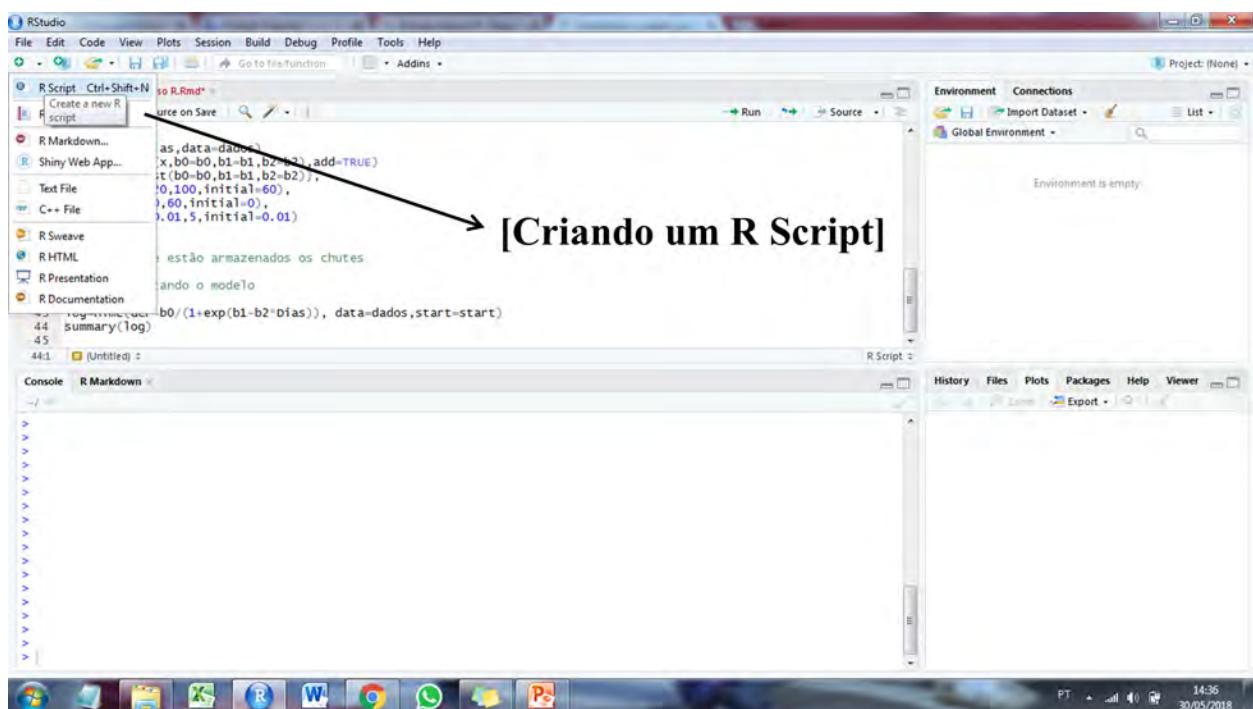


Figure 3: Iniciando um novo R Script

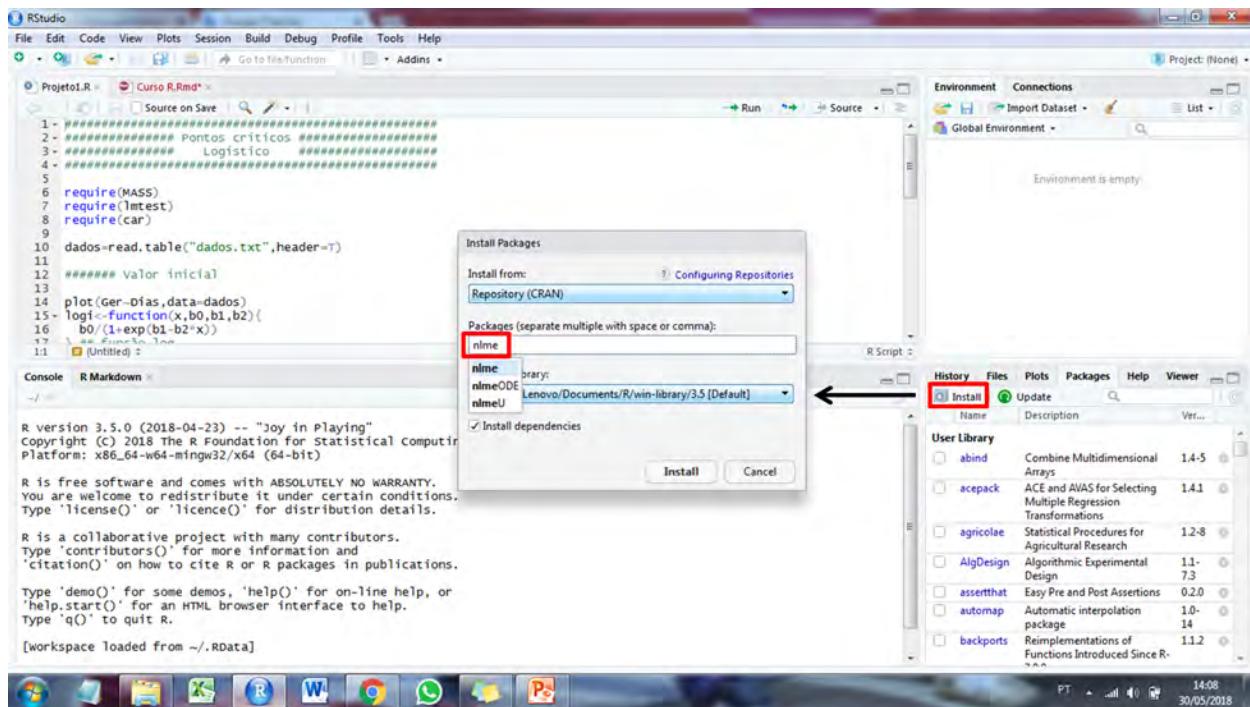


Figure 4: Instalando pacotes

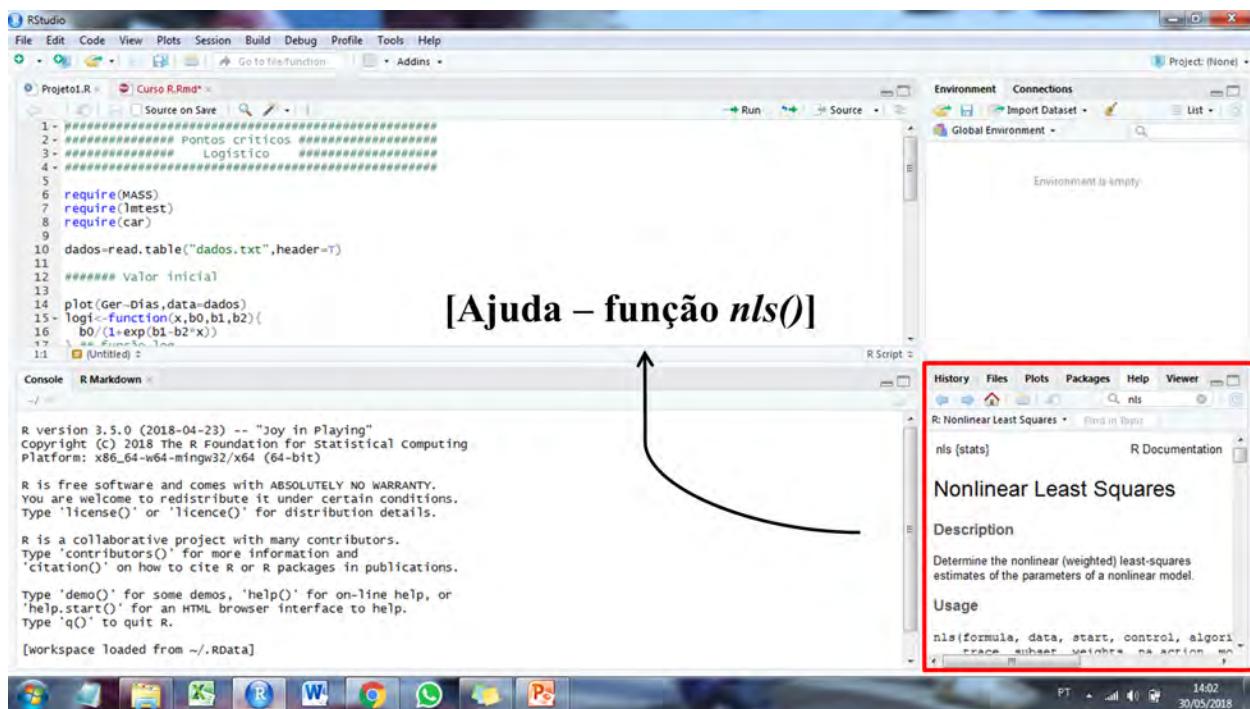


Figure 5: Ajuda para a função `nls()`

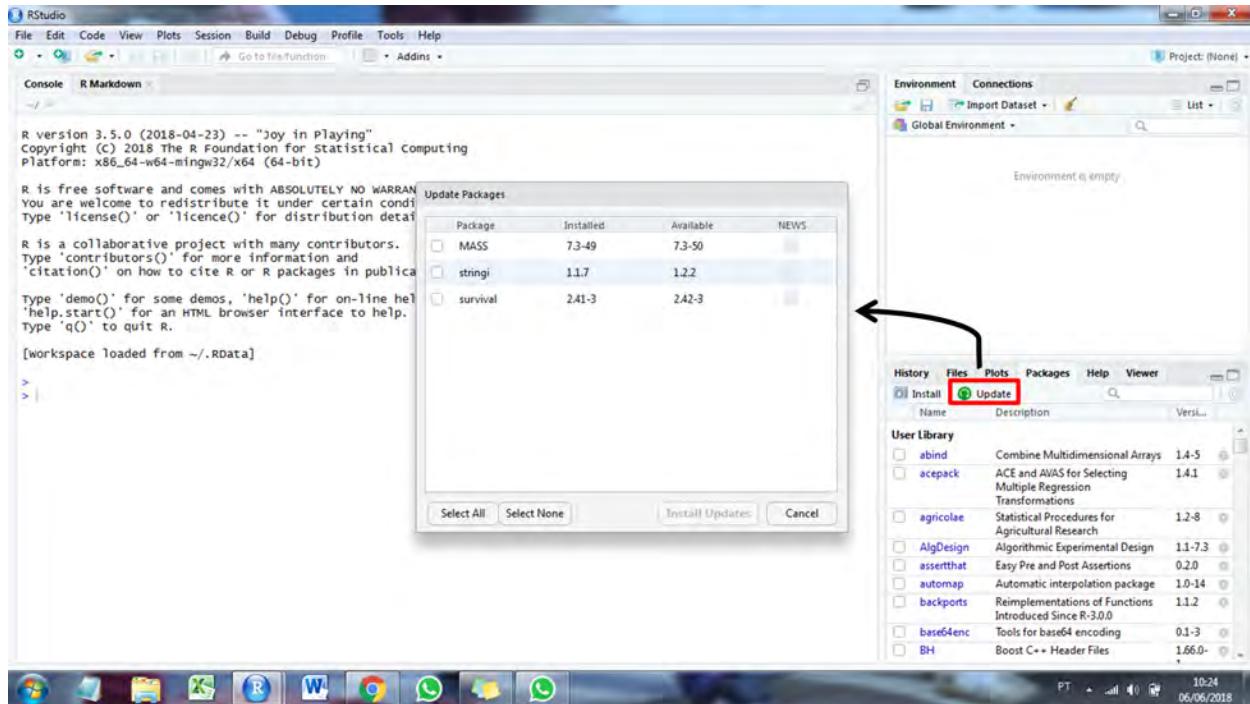


Figure 6: Verificando se há atualizações disponíveis

os pacotes individualmente e, posteriormente, em grupo.

```
install.packages("MASS")
require(MASS)
pacotes=c("MASS", "ExpDes", "car") # "armazenando" os pacotes
install.packages(pacotes)
require(pacotes)
```

Caso o pacote que contenha a função que se deseja utilizar não esteja instalado, uma mensagem de erro aparecerá no *console*. Para usar a função *nlme()* é necessário que o pacote *nlme* esteja instalado e carregado. O erro no *console* indica que a função *nlme()* não foi encontrada. O exemplo abaixo foi extraído do livro *Mixed-Effects models in S and S-PLUS* de Pinheiro and Bates (2000):

```
log=nlme(height ~ SSasymp(age, Asym, R0, lrc),
          data = Loblolly,
          fixed = Asym + R0 + lrc ~ 1,
          random = Asym ~ 1,
          start = c(Asym = 103, R0 = -8.5, lrc = -3.3))
```

```
## Error in nlme(height ~ SSasymp(age, Asym, R0, lrc), data = Loblolly, fixed = Asym + :
```

Os pacotes disponibilizados no software *R* estão em constante atualização. Utilizando a função *packageStatus()* e *summary(packageStatus())* é possível verificar se os pacotes estão atualizados. Outra forma é ir em *Packages > Update* que se encontra no canto inferior direito conforme demonstrado na figura abaixo:

```

## Verifica o status dos pacotes. Upgrade = 0 indica que não há pacotes a serem atualizados
packageStatus()
## Detalha os pacotes. $OK estão atualizados, $update precisam de atualização
summary(packageStatus())

```

Por fim, para citar os pacotes , recomenda-se verificar a referência através da função `citation()`.

```

citation("MASS")

##
## To cite the MASS package in publications use:
##
##   Venables, W. N. & Ripley, B. D. (2002) Modern Applied Statistics
##   with S. Fourth Edition. Springer, New York. ISBN 0-387-95457-0
##
## A BibTeX entry for LaTeX users is
##
## @Book{,
##   title = {Modern Applied Statistics with S},
##   author = {W. N. Venables and B. D. Ripley},
##   publisher = {Springer},
##   edition = {Fourth},
##   address = {New York},
##   year = {2002},
##   note = {ISBN 0-387-95457-0},
##   url = {http://www.stats.ox.ac.uk/pub/MASS4},
## }

```

## 1.1 Tipos de objetos

Os tipos de objeto mais utilizados na linguagem *R* são: a) vetores, b)matrizes, c) data frames, d)listas e e) funções. Um enfoque maior será dado aos vetores, matrizes e data frames, pois estes são amplamente utilizados, inclusive nas análises mais simples.

### 1.1.1 Vetores

A função `c()` combina valores que formam vetores. Como visto anteriormente, `c()` foi utilizado para carregar os pacotes, porém ao invés de valores foram utilizados caracteres (indicados por ""). Esse exemplo já demonstra a grande utilidade desta função. Abaixo, é demonstrado como vetores podem ser gerados com a função `c()`.

```

x1=c(1) #Escalar
x2=c(1,2) #Vetor
x3=c(1,2,3) #Vetor
x3.1=c("um","dois","três") #Vetor com caracteres

```

Os vetores foram armazenados em `x1`, `x2` e `x3` e ficaram armazenados como valores na área de trabalho como valores (*values*). Para que os valores sejam mostrados basta digitar no *console* onde os vetores foram armazenados.

```
x3
```

```
## [1] 1 2 3
```

A função `c()` pode ser utilizada com outras funções. Não é objetivo deste curso explorar a potencialidade desta função, mas algumas exemplos serão demonstrados para despertar o interesse dos usuários:

```
x4=c(rep(1:4,each=4))
x5=seq(1:5)
x6=c(rep(seq(1:5),each=2))
```

Utilizando colchetes [ ] é possível selecionar um (ou um conjunto) de elementos de um vetor. Por exemplo:

```
x7=x6[1] #Seleciona o primeiro elemento da matriz
x8=x6[4] #Seleciona o quarto elemento da matriz
x9=c(x6[1],x6[4],x6[8]) # Seleciona o primeiro, o quarto e o oitavo elemento
x10=x6[1:4] #armazena uma sequência de elementos (primeiro ao quarto)
```

```
x6
```

```
## [1] 1 1 2 2 3 3 4 4 5 5
```

```
x9
```

```
## [1] 1 2 4
```

### 1.1.2 Matrizes

As matrizes são um conjunto de valores (ou variáveis) dispostos em linhas e colunas, e que formam um corpo delimitado por [ ]. As matrizes são geralmente representadas genericamente por  $\mathbf{A}_{M \times N}$ , onde  $M$  e  $N$  representam os números de linhas e colunas da matriz, respectivamente. As matrizes podem ser facilmente construídas utilizando as funções `cbind()`, `rbind()`. A primeira função adiciona colunas as matrizes, enquanto que a segunda adiciona linhas. A função `matrix()` também pode ser utilizada.

```
### Usando cbind()
x10=cbind(1,2,3,4,5) # ou x10=cbind(1:5), 5 colunas com 1 elemento cada
x11=cbind(c(1,2,3,4,5)) # ou x11=cbind(c(1:5)), 1 coluna com 5 elementos cada
x12=cbind(c(1,2,3,4,5),c(6,7,8,9,10)) # 2 colunas de 5 elementos
x12.1=cbind(x11,c(6:10))
x13=cbind(c(1,2,3,4,5),c(6,7,8,9,10),c(11,12,13,14,15)) # 3 colunas de 5 elementos
x13.1=cbind(x12.1,c(11,12,13,14,15))

### Usando rbind()
x14=rbind(1,2,3,4,5) # x14 = x11, 5 linhas com 1 elemento cada
```

```

x15=rbind(c(1,2,3,4,5)) # x15 = x10, 1 linha com 5 elementos cada
x16=rbind(c(1,2,3,4,5),c(6,7,8,9,10)) # 2 linhas com 5 elementos cada
x16.1=rbind(x15,c(6,7,8,9,10))
x17=rbind(c(1,6),c(2,7),c(3,8),c(4,9),c(5,10)) # x16 = x12

```

As funções `cbind()` e `rbind()` podem ser utilizadas conjuntamente. Não queremos confundir a sua cabeça, mas se a lição anterior foi entendida, a próxima se torna fácil.

```

### Usando rbind() e cbind()
x18=cbind(c(1,2,3,4,5),c(6,7,8,9,10),rbind(c(11),c(12),c(13),c(14),c(15)))

```

x13

```

##      [,1] [,2] [,3]
## [1,]     1    6   11
## [2,]     2    7   12
## [3,]     3    8   13
## [4,]     4    9   14
## [5,]     5   10   15

```

x18

```

##      [,1] [,2] [,3]
## [1,]     1    6   11
## [2,]     2    7   12
## [3,]     3    8   13
## [4,]     4    9   14
## [5,]     5   10   15

```

Com a função `matrix()` podemos ter o mesmo resultado que o obtido com o uso das funções `cbind()` e `rbind()`. Porém, para utilizar a função `matrix()`, alguns *argumentos* devem ser declarados. Na função `matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)`, os argumentos que devemos inicialmente conhecer são o `nrow`, `ncol` e `byrow`. O primeiro indica o número de linhas da matriz, o segundo a número de colunas e o terceiro indica como a matriz é preenchida. Por *default*, `byrow` é `FALSE`, indicando que as matrizes são preenchidas por colunas. Se `TRUE`, o preenchimento ocorre por linhas.

```

### Usando matrix
x19=matrix(c(1:15),nrow=5,ncol=3)
x20=matrix(c(1:15), nrow=5, ncol=3, byrow=TRUE)
x21=matrix(c(1,6,11,2,7,12,3,8,13,4,9,14,5,10,15),nrow=5, ncol=3, byrow=TRUE)

```

x19

```

##      [,1] [,2] [,3]
## [1,]     1    6   11
## [2,]     2    7   12
## [3,]     3    8   13
## [4,]     4    9   14
## [5,]     5   10   15

```

```
x20
```

```
##      [,1] [,2] [,3]
## [1,]     1     2     3
## [2,]     4     5     6
## [3,]     7     8     9
## [4,]    10    11    12
## [5,]    13    14    15
```

```
x21
```

```
##      [,1] [,2] [,3]
## [1,]     1     6    11
## [2,]     2     7    12
## [3,]     3     8    13
## [4,]     4     9    14
## [5,]     5    10    15
```

Para selecionar elementos, linhas e colunas da matriz com [ ] utiliza-se um sistema de coordenadas:

```
x19[2,3] ## seleciona o elemento que está na linha 2 e coluna 3
```

```
## [1] 12
```

```
x19[,2] ## "," indica que todas as linhas serão selecionadas na coluna 2
```

```
## [1] 6 7 8 9 10
```

```
x19[1,] ## "," indica que todas as colunas serão selecionadas na linha 1
```

```
## [1] 1 6 11
```

### 1.1.3 Data Frame

A função `data.frame()` cria estruturas cujas colunas podem ser valores numéricos ou caracteres. É uma estrutura muito utilizada em funções do *software R*.

```
x22=data.frame(
  "Ambiente"=rep(c("Faxinal do Soturno","Santa Maria"),each=20),
  "Genotipo"=rep(c("G1","G2","G3","G4"),each=5),
  "Rep"=rep(c(1:5),each=4),
  "Y"=rep(c(11:15),each=4)
)
```

```
x22
```

```
##           Ambiente Genotipo Rep  Y
## 1  Faxinal do Soturno      G1   1 11
## 2  Faxinal do Soturno      G1   1 11
## 3  Faxinal do Soturno      G1   1 11
```

```

## 4 Faxinal do Soturno      G1  1 11
## 5 Faxinal do Soturno      G1  2 12
## 6 Faxinal do Soturno      G2  2 12
## 7 Faxinal do Soturno      G2  2 12
## 8 Faxinal do Soturno      G2  2 12
## 9 Faxinal do Soturno      G2  3 13
## 10 Faxinal do Soturno     G2  3 13
## 11 Faxinal do Soturno     G3  3 13
## 12 Faxinal do Soturno     G3  3 13
## 13 Faxinal do Soturno     G3  4 14
## 14 Faxinal do Soturno     G3  4 14
## 15 Faxinal do Soturno     G3  4 14
## 16 Faxinal do Soturno     G4  4 14
## 17 Faxinal do Soturno     G4  5 15
## 18 Faxinal do Soturno     G4  5 15
## 19 Faxinal do Soturno     G4  5 15
## 20 Faxinal do Soturno     G4  5 15
## 21      Santa Maria        G1  1 11
## 22      Santa Maria        G1  1 11
## 23      Santa Maria        G1  1 11
## 24      Santa Maria        G1  1 11
## 25      Santa Maria        G1  2 12
## 26      Santa Maria        G2  2 12
## 27      Santa Maria        G2  2 12
## 28      Santa Maria        G2  2 12
## 29      Santa Maria        G2  3 13
## 30      Santa Maria        G2  3 13
## 31      Santa Maria        G3  3 13
## 32      Santa Maria        G3  3 13
## 33      Santa Maria        G3  4 14
## 34      Santa Maria        G3  4 14
## 35      Santa Maria        G3  4 14
## 36      Santa Maria        G4  4 14
## 37      Santa Maria        G4  5 15
## 38      Santa Maria        G4  5 15
## 39      Santa Maria        G4  5 15
## 40      Santa Maria        G4  5 15

```

Em `x22` simulamos como muitos experimentos são organizados no momento de tabulação dos dados (fatores nas colunas e variáveis nas linhas). Caso o pesquisador opte por analisar os ambientes separadamente, ele pode lançar mão da função `subset()` para obter o banco de dados para um ambiente específico:

```
x23=subset(x22,Ambiente=="Faxinal do Soturno")
```

```
x23
```

```
##          Ambiente Genotipo Rep  Y
```

```

## 1 Faxinal do Soturno      G1  1 11
## 2 Faxinal do Soturno      G1  1 11
## 3 Faxinal do Soturno      G1  1 11
## 4 Faxinal do Soturno      G1  1 11
## 5 Faxinal do Soturno      G1  2 12
## 6 Faxinal do Soturno      G2  2 12
## 7 Faxinal do Soturno      G2  2 12
## 8 Faxinal do Soturno      G2  2 12
## 9 Faxinal do Soturno      G2  3 13
## 10 Faxinal do Soturno     G2  3 13
## 11 Faxinal do Soturno     G3  3 13
## 12 Faxinal do Soturno     G3  3 13
## 13 Faxinal do Soturno     G3  4 14
## 14 Faxinal do Soturno     G3  4 14
## 15 Faxinal do Soturno     G3  4 14
## 16 Faxinal do Soturno     G4  4 14
## 17 Faxinal do Soturno     G4  5 15
## 18 Faxinal do Soturno     G4  5 15
## 19 Faxinal do Soturno     G4  5 15
## 20 Faxinal do Soturno     G4  5 15

```

#### 1.1.4 Lista

No exemplo abaixo, será armazenado em uma lista dois data-frames e uma matriz. Posteriormente, será selecionado a matriz que está armazenada na lista:

```

x24=list(x22,x23,x19)
x25=x24[[3]]

```

```
x24
```

```

## [[1]]
##           Ambiente Genotipo Rep  Y
## 1  Faxinal do Soturno      G1  1 11
## 2  Faxinal do Soturno      G1  1 11
## 3  Faxinal do Soturno      G1  1 11
## 4  Faxinal do Soturno      G1  1 11
## 5  Faxinal do Soturno      G1  2 12
## 6  Faxinal do Soturno      G2  2 12
## 7  Faxinal do Soturno      G2  2 12
## 8  Faxinal do Soturno      G2  2 12
## 9  Faxinal do Soturno      G2  3 13
## 10 Faxinal do Soturno     G2  3 13
## 11 Faxinal do Soturno     G3  3 13
## 12 Faxinal do Soturno     G3  3 13
## 13 Faxinal do Soturno     G3  4 14
## 14 Faxinal do Soturno     G3  4 14

```

```

## 15 Faxinal do Soturno      G3   4 14
## 16 Faxinal do Soturno      G4   4 14
## 17 Faxinal do Soturno      G4   5 15
## 18 Faxinal do Soturno      G4   5 15
## 19 Faxinal do Soturno      G4   5 15
## 20 Faxinal do Soturno      G4   5 15
## 21      Santa Maria        G1   1 11
## 22      Santa Maria        G1   1 11
## 23      Santa Maria        G1   1 11
## 24      Santa Maria        G1   1 11
## 25      Santa Maria        G1   2 12
## 26      Santa Maria        G2   2 12
## 27      Santa Maria        G2   2 12
## 28      Santa Maria        G2   2 12
## 29      Santa Maria        G2   3 13
## 30      Santa Maria        G2   3 13
## 31      Santa Maria        G3   3 13
## 32      Santa Maria        G3   3 13
## 33      Santa Maria        G3   4 14
## 34      Santa Maria        G3   4 14
## 35      Santa Maria        G3   4 14
## 36      Santa Maria        G4   4 14
## 37      Santa Maria        G4   5 15
## 38      Santa Maria        G4   5 15
## 39      Santa Maria        G4   5 15
## 40      Santa Maria        G4   5 15
##
## [[2]]
##          Ambiente Genotipo Rep Y
## 1  Faxinal do Soturno      G1   1 11
## 2  Faxinal do Soturno      G1   1 11
## 3  Faxinal do Soturno      G1   1 11
## 4  Faxinal do Soturno      G1   1 11
## 5  Faxinal do Soturno      G1   2 12
## 6  Faxinal do Soturno      G2   2 12
## 7  Faxinal do Soturno      G2   2 12
## 8  Faxinal do Soturno      G2   2 12
## 9  Faxinal do Soturno      G2   3 13
## 10 Faxinal do Soturno      G2   3 13
## 11 Faxinal do Soturno      G3   3 13
## 12 Faxinal do Soturno      G3   3 13
## 13 Faxinal do Soturno      G3   4 14
## 14 Faxinal do Soturno      G3   4 14
## 15 Faxinal do Soturno      G3   4 14
## 16 Faxinal do Soturno      G4   4 14
## 17 Faxinal do Soturno      G4   5 15

```

```

## 18 Faxinal do Soturno      G4   5 15
## 19 Faxinal do Soturno      G4   5 15
## 20 Faxinal do Soturno      G4   5 15
##
## [[3]]
## [,1] [,2] [,3]
## [1,]    1    6   11
## [2,]    2    7   12
## [3,]    3    8   13
## [4,]    4    9   14
## [5,]    5   10   15

x25

## [,1] [,2] [,3]
## [1,]    1    6   11
## [2,]    2    7   12
## [3,]    3    8   13
## [4,]    4    9   14
## [5,]    5   10   15

```

### 1.1.5 Funções

As funções são a base da linguagem *R*. Através de *argumentos* que são indicados em `function()`, uma expressão (ou série de expressões) é resolvida e um valor (ou um conjunto de valores) é retornado.

```

F1=function(x){ # x é o argumento da função
  a=2*x+1
  return(a) ## retorna a
}

F2=function(x,y){ # dois argumentos na função
  a=2*x+1
  b=y
  c=a+b
  return(c) ## retorna a
}

F3=function(x){
  if(x<=5)a=2*x+1 # Se x for menor ou igual a 5, resolve essa expressão
  if(x>5&x<10)a=3*x+1 # Se x estiver entre 6 e 10, resolve essa expressão
  if(x>10)a="ERROU!" # Se x for maior que 10
  return(a)
}

```

```
F1(2)
```

```

## [1] 5
F2(2,2)

## [1] 7
F3(1)

## [1] 3
F3(6)

## [1] 19
F3(20)

## [1] "ERROU!"

```

### 1.1.6 Identificando os tipos de objetos

Conforme visto anteriormente, é possível construir vários tipos de objetos em linguagem R. Veremos mais adiante que muitas funções exigem objetos específicos como *argumento*, e por isso conhecê-los é muito importante. Funções genéricas como `class()` ou `is.tipo_de_objeto` são importantes para identificar o tipo de objeto gerado.

```

x26=c(3,7,0)
x19=matrix(c(1:15),nrow=5,ncol=3)
x22=data.frame(
  "Ambiente"=rep(c("Faxinal do Soturno","Santa Maria"),each=20),
  "Genotipo"=rep(c("G1","G2","G3","G4"),each=5),
  "Rep"=rep(c(1:5),each=4),
  "Y"=rep(c(11:15),each=4)
)

class(x26)

## [1] "numeric"

class(x19)

## [1] "matrix"

class(x22)

## [1] "data.frame"

is.matrix(x19)

## [1] TRUE

is.matrix(x22)

## [1] FALSE

```

Algumas funções permitem classificar os objetos criados conforme o desejado:

```

x22=as.matrix(x22)
x19=as.data.frame(x19)

class(x22)

## [1] "matrix"

class(x19)

## [1] "data.frame"

```

Podemos usar as funções vistas até agora para classificar como fatores e como vetores numéricos um conjunto de dados (data frame). Vamos usar como exemplo um conjunto de dados que simula um experimento com quatro genótipos conduzidos em dois ambientes. Devemos classificar o que é fator e o que é vetor numérico (variáveis) para executar a ANOVA:

```

x22=data.frame(
  "Ambiente"=rep(c("Faxinal do Soturno","Santa Maria"),each=20),
  "Genotipo"=rep(c("G1","G2","G3","G4"),each=5),
  "Rep"=rep(c(1:5),each=4),
  "Y"=rep(c(11:15),each=4)
)
x22$Ambiente=as.factor(x22$Ambiente)
# $ indica que estamos nos referindo apenas a ambiente
x22$Genotipo=as.factor(x22$Genotipo)
# $ indica que estamos nos referindo apenas a genótipo
x22$Rep=as.factor(x22$Rep)
# $ indica que estamos nos referindo apenas a repetição
x22$Y=as.numeric(x22$Y)
# $ indica que estamos nos referindo apenas a variável Y

```

## 1.2 Operações matemáticas e álgebra de matrizes

As operações matemáticas utilizam símbolos que são padrão em outros softwares estatísticos.

```

1+1 # Soma
2-1 # Subtração
2*2 # Multiplicação
sqrt(4) # Raiz quadrada
4^2 # Potência
log(10) # Por default, o logarítmico é de base e (logarítmico natural)
log(100,10) # Logarítmico de base 10
exp(100) # exponencial

```

No caso de matrizes, utiliza-se `%*%` para multiplicar matrizes, `t()` para transpor matrizes e `solve()` e `ginv()` para inverter matrizes. Para exemplificar como utiliza-los, vamos resolver o seguinte sistema de equações retirado do livro de Rencher and Schaalje (2008):

$$\begin{aligned}x_1 + 2x_2 &= 4 \\x_1 - x_2 &= 1 \\x_1 + x_2 &= 3\end{aligned}$$

Matricialmente o sistema acima é dado por:

$$\left[ \begin{array}{cc} 1 & 2 \\ 1 & -1 \\ 1 & 1 \end{array} \right] \left[ \begin{array}{c} x_1 \\ x_2 \end{array} \right] = \left[ \begin{array}{c} 4 \\ 1 \\ 3 \end{array} \right]$$

Esse sistema de equações é representado por  $\mathbf{AX} = \mathbf{c}$  e tem como solução  $\mathbf{X} = \mathbf{A}^{-1}\mathbf{c}$ :

X

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

Considere um equação linear multipla cuja variável dependente é  $\mathbf{Y}$ , e as variáveis independentes são  $\mathbf{X1}$  e  $\mathbf{X2}$  (dados obtidos em Kutner et al. (2005)). O sistema de equações é representado matricialmente por

$$(\mathbf{X}^T \mathbf{X}) \boldsymbol{\beta} = \mathbf{X}^T \mathbf{Y}$$

e pode ser resolvido por:

```
X0=rep(1,each=21)
X1=c(68.5,45.2,91.3,47.8,46.9,66.1,49.5,52,48.9,38.4,87.9,72.8,
     88.4,42.9,52.5,85.7,41.3,51.7,89.6,82.7,52.3)
X2=c(16.7,16.8,18.2,16.3,17.3,18.2,15.9,17.2,16.6,16,18.3,17.1,
     17.4,15.8,17.8,18.4,16.5,16.3,18.1,19.1,16)
Y=c(174.4,164.4,244.2,154.6,181.6,207.5,152.8,163.2,145.4,137.2,
    241.9,191.1,232,145.3,161.1,209.7,146.4,144,232.6,224.1,166.5)
X=matrix(c(X0,X1,X2),nrow=21,ncol=3)
X
```

```
##      [,1] [,2] [,3]
## [1,]    1 68.5 16.7
## [2,]    1 45.2 16.8
## [3,]    1 91.3 18.2
## [4,]    1 47.8 16.3
## [5,]    1 46.9 17.3
## [6,]    1 66.1 18.2
## [7,]    1 49.5 15.9
## [8,]    1 52.0 17.2
## [9,]    1 48.9 16.6
## [10,]   1 38.4 16.0
## [11,]   1 87.9 18.3
```

```

## [12,]    1 72.8 17.1
## [13,]    1 88.4 17.4
## [14,]    1 42.9 15.8
## [15,]    1 52.5 17.8
## [16,]    1 85.7 18.4
## [17,]    1 41.3 16.5
## [18,]    1 51.7 16.3
## [19,]    1 89.6 18.1
## [20,]    1 82.7 19.1
## [21,]    1 52.3 16.0

XX=t(X)%*%X # t() = transposta
B=(solve(XX))%*%(t(X)%*%Y) # solve() = inversa
B

```

```

##          [,1]
## [1,] -68.85707
## [2,]  1.45456
## [3,]  9.36550

```

O modelo ajustado é dado por  $\hat{Y} = -68,85707 + 1,45456\beta_1 + 9,36550\beta_2 + \varepsilon$ . Também podemos utilizar as funções `det()` para calcular o determinante de uma matriz. Já a função `eigen()` retorna uma lista com os autovalores e autovetores da matriz. A função `names()` indica o que contém no objeto `av`, e usando `$` é possível selecionar os autovalores ou os autovetores.

```

detXX=det(XX)
av=eigen(XX)

names(av)

## [1] "values"  "vectors"
av$values # Extrai os autovalores

## [1] 9.357825e+04 3.409163e+02 3.348663e-02
av$vectors # Extrai os autovetores

##          [,1]      [,2]      [,3]
## [1,] -0.01443945  0.06534768  0.997758079
## [2,] -0.96800779 -0.25090843  0.002424222
## [3,] -0.25050433  0.96580259 -0.066880041

```

A função `diag()` extrai a diagonal da matriz:

```

XX

##          [,1]      [,2]      [,3]
## [1,]    21.0   1302.40   360.00
## [2,]  1302.4  87707.94 22609.19
## [3,]   360.0  22609.19  6190.26

```

```

diag(XX)

## [1] 21.00 87707.94 6190.26

```

### 1.3 Loops

Muitas vezes queremos repetir muitas vezes (em sequência) nossa programação. Isso é muito comum e pode ser facilmente implementado pela função `for()`. Nesta função precisamos apenas indicar o número de vezes (ou de que forma) que ela será executada, e dentro dela indicar a programação que deve ser repetida  $n$  vezes.

```

dados_loop=data.frame(matrix(nrow=10,ncol = 1)) ### onde serão armazenados os dados
names(dados_loop)=c("Dados")

for(i in 1:10){
n=runif(20,0,1)
media=mean(n)
;
dados_loop$Dados[i]=media
}
dados_loop

##          Dados
## 1  0.5001109
## 2  0.4384388
## 3  0.5735661
## 4  0.4942515
## 5  0.4422534
## 6  0.4537714
## 7  0.5289987
## 8  0.5636228
## 9  0.3977546
## 10 0.5017746

```

Os Loops são importantes em estudos que utilizam reamostragens para realizar análises estatísticas. Através de reamostragens é possível construir intervalos de confiança e testar hipóteses não paramétricas. Para fixar como utilizar a função `loop()`, vamos demonstrar o teorema central do limite:

```

dados_loop=data.frame(matrix(nrow=1000,ncol = 4)) ### onde serão armazenados os dados
names(dados_loop)=c("Dados1","Dados2","Dados3","Dados4")

for(j in 1:1000){
n=runif(20,0,1)
media[j]=mean(n)
;
dados_loop$Dados1[1:5]=media ## Armazena as 5 primeras médias
}

```

```

dados_loop$Dados2[1:10]=media ## Armazena as 10 primeiras médias
dados_loop$Dados3[1:50]=media ## Armazena as 100 primeiras médias
dados_loop$Dados4[1:1000]=media ## Armazena as 1000 primeiras médias
}

par(mfrow=c(2,2))
hist(dados_loop$Dados1,main = "5",ylab="Frequência", xlab="Média")
hist(dados_loop$Dados2,main = "10",ylab="Frequência", xlab="Média")
hist(dados_loop$Dados3,main = "50",ylab="Frequência", xlab="Média")
hist(dados_loop$Dados4,main = "1000",ylab="Frequência", xlab="Média")

```

## 1.4 Construindo uma tabela

Com o que foi visto até agora, é possível construir uma tabela para armazenar dados ou resultados gerados em uma análise. São inúmeras as formas de fazer isso, porém vamos mostrar apenas uma (devido ao pouco tempo). Utilizaremos as funções `matrix()` e `data.frame()` para a tabela, e a função `names()` para dar nomes as colunas. A construção de tabelas é muito útil para armazenar resultados na área de trabalho.

```

Ex.gen=matrix(0,ncol=3,nrow=20)
Ex.gen=as.data.frame(Ex.gen)
Ex.gen=as.data.frame(matrix(0,ncol=3,nrow=20)) # Utilizando tudo em uma linha só
names(Ex.gen)=c("G1","G2","G3")
Yeld1=rnorm(20,10,1)
# gera 20 valores de uma distribuição normal com média 10 e desvio padrão 1
Yeld2=rnorm(20,40,3)
# gera 20 valores de uma distribuição normal com média 40 e desvio padrão 3
Yeld3=rnorm(20,25,2.5)
# gera 20 valores de uma distribuição normal com média 25 e desvio padrão 2,5
Ex.gen$G1=Yeld1
Ex.gen$G2=Yeld2
Ex.gen$G3=Yeld3
Ex.gen

```

	G1	G2	G3
## 1	9.552228	43.40519	25.61442
## 2	9.116528	34.59915	23.49314
## 3	9.029065	42.06568	25.70684
## 4	10.719547	41.35450	24.88697
## 5	10.687592	41.99486	22.45169
## 6	9.859372	38.73699	27.44829
## 7	10.165176	38.22968	25.60023
## 8	9.888961	40.44973	27.23555
## 9	10.822013	38.83267	23.31050
## 10	10.563233	37.92885	27.68391
## 11	10.363880	35.80376	25.33431

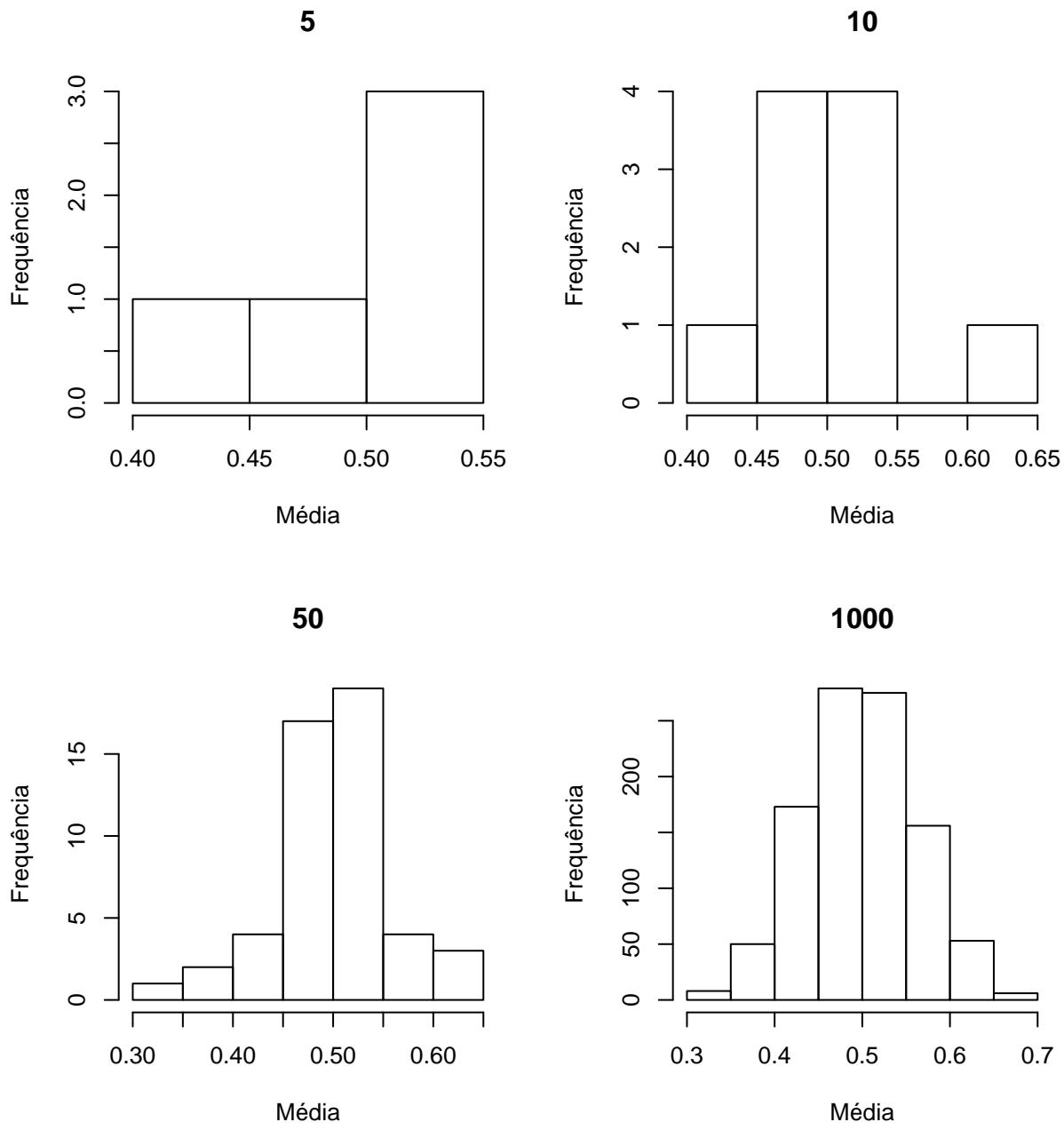


Figure 7: Demonstração do teorema central do limite

```

## 12 11.075796 41.16020 25.39570
## 13 9.572334 38.40035 20.36521
## 14 10.451615 34.61170 26.00877
## 15 9.628439 35.43038 27.48142
## 16 10.292712 39.23545 25.15524
## 17 9.648813 41.52780 29.42458
## 18 9.552438 40.25018 27.22714
## 19 11.568640 41.93661 26.91160
## 20 8.570283 40.63130 29.92856

```

## 1.5 Entrada de dados

A entrada de dados pode ser feita de várias maneiras, e em vários formatos. Porém daremos destaque as formas mais comuns. A forma mais simples (e mais trabalhosa) é digitar os dados diretamente no *console*, utilizando para isso a função `scan()`. Para importar dados digitados em extensão *.txt* utiliza-se a função `read.table()`. Por fim, para carregar arquivos em extensão *.xlsx* a maneira mais simples é utilizar o *Import Dataset* que encontra-se na área de trabalho.

```

data=scan() #Enter
1: 10
2: 20
3: 30
4: 40
5: # Duplo enter
data

```

A função `scan()` é muito trabalhosa e pouco utilizada. Caso o usuário queira carregar dados salvos em extensão *.txt* (bloco de notas), usando a função `read.table()`, deve-se ter o cuidado de mover o arquivo para o diretório previamente indicado. Como as colunas são identificadas por espaços, é importante que o usuário não utilize nomes compostos no cabeçalho ou no corpo da tabela. Quando isso ocorre, o *R* identifica o erro no console através da mensagem `Error in read.table("Dados_1.txt", header = TRUE) : more columns than column names.`

```

dados=read.table("Dados_1.txt",header=TRUE)
## Argumento header = TRUE indica a existência de cabeçalho

```

```

dados=read.table("Dados_2.txt",header=TRUE)
## Argumento header = TRUE indica a existência de cabeçalho
dados

```

```

##          Trat  Rep     Sta    num     peso
## 1 bcco_alb_imp  1 1132.789 0.00000 0.00000
## 2 bcco_alb_imp  2 1199.039 0.06250 0.61875
## 3 bcco_alb_imp  3 1265.189 0.06250 0.61875
## 4 bcco_alb_imp  4 1337.789 0.16525 1.48125

```

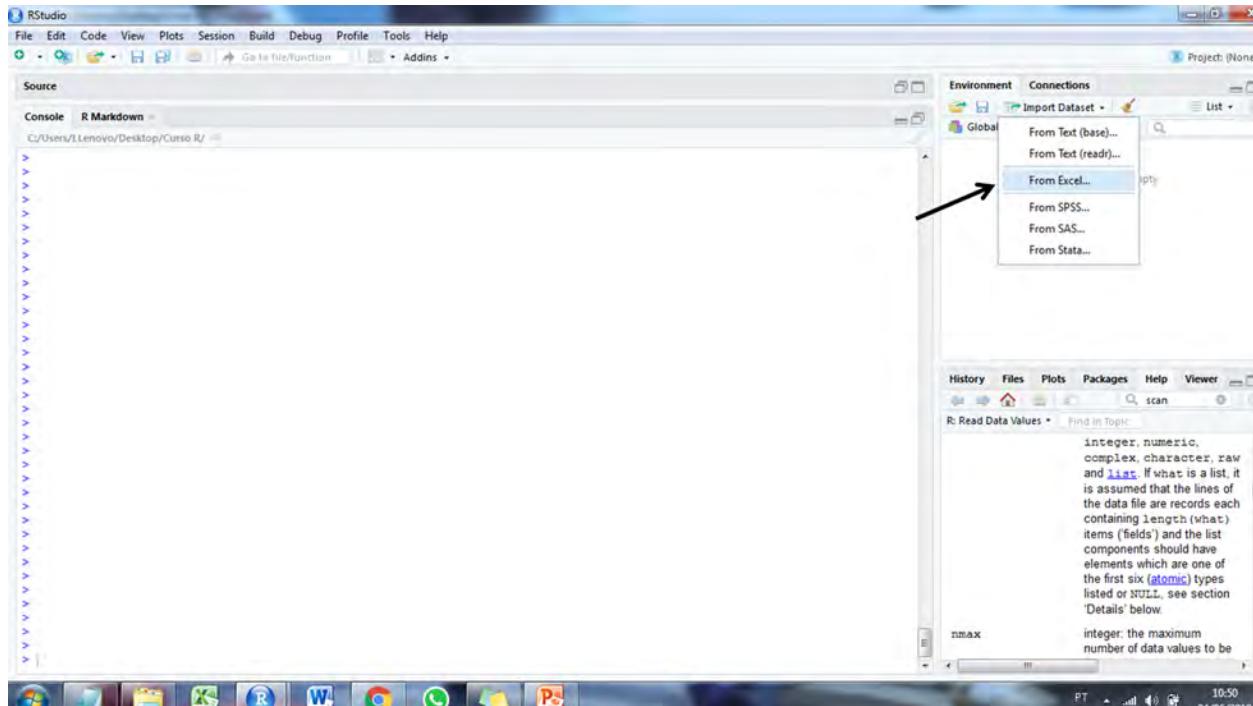


Figure 8: Importando dados de planilhas eletrônicas do Excel - Passo 1

A forma mais comum do pesquisador digitar seus dados é através de planilhas eletrônicas do Excel. Para carregar esses dados, basta ir em *Import Dataset* na área de trabalho. O passo a passo está descrito abaixo:

Também é possível carregar dados já existentes dentro do *software R*. Geralmente, os pacotes contém dados que são utilizados como exemplo de aplicação das suas funções.

```
head(iris) ## head() limita os valores que serão impressos no console

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5         1.4         0.2  setosa
## 2          4.9         3.0         1.4         0.2  setosa
## 3          4.7         3.2         1.3         0.2  setosa
## 4          4.6         3.1         1.5         0.2  setosa
## 5          5.0         3.6         1.4         0.2  setosa
## 6          5.4         3.9         1.7         0.4  setosa
```

## 1.6 Funções gráficas

Será dado destaque a funções gráficas genéricas, que são muito úteis na demonstração de comportamento de variáveis ou na análise gráfica dos resíduos de modelos da ANOVA, em regressão linear e não linear. Será demonstrado como utilizar as funções `plot()`, `hist()`, `qqplot()`, `box-plot()` e `curve()`.

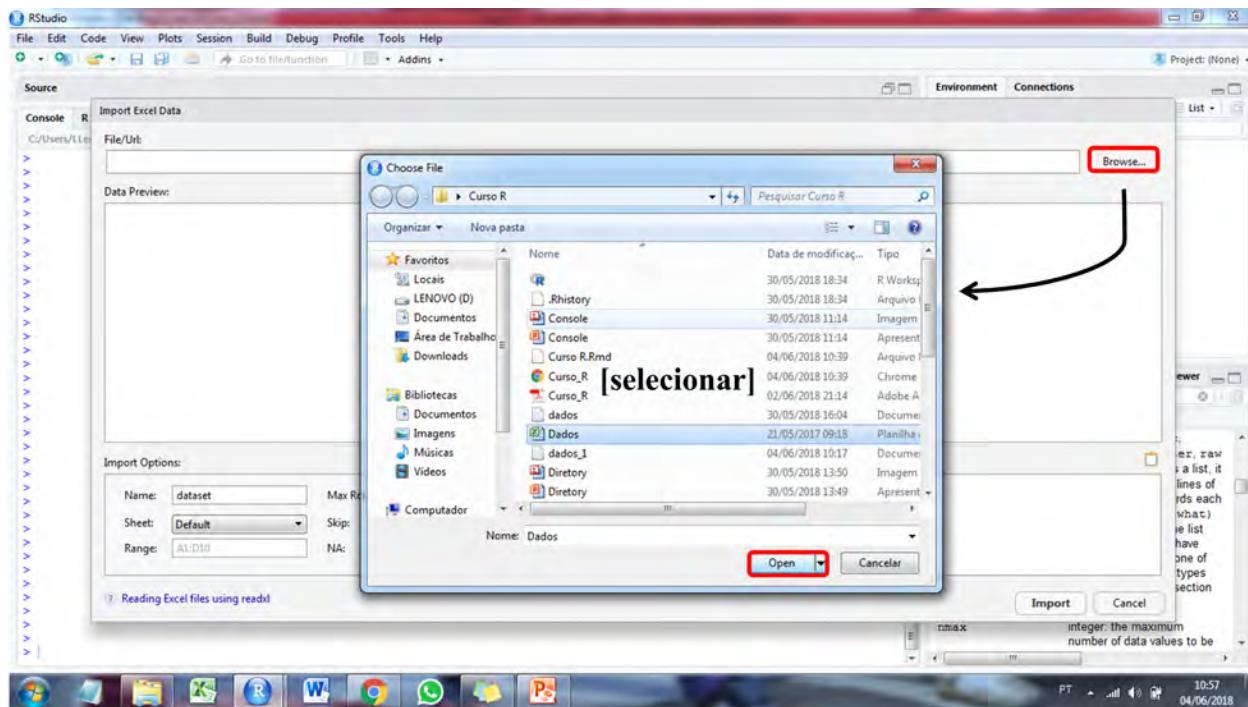


Figure 9: Importando dados de planilhas eletrônicas do Excel - Passo 2

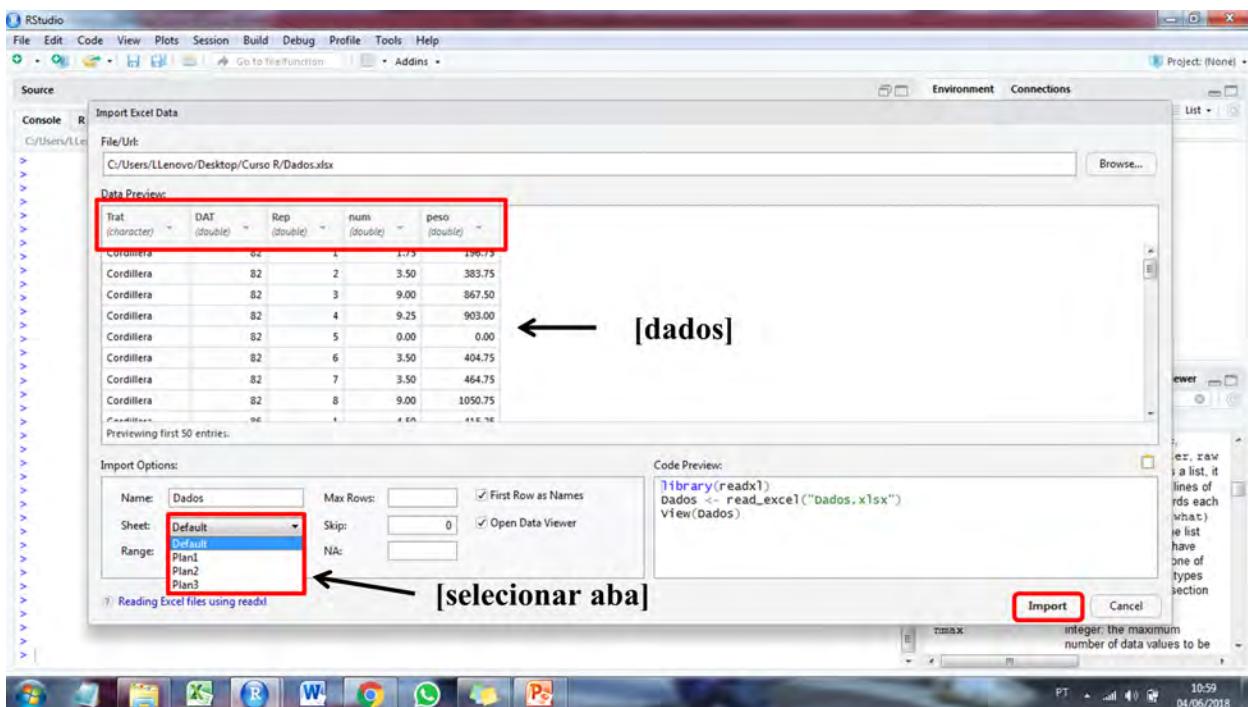


Figure 10: Importando dados de planilhas eletrônicas do Excel - Passo 3

### 1.6.1 Gráficos utilizando a função `plot()`

A função `plot()` pode ser usada para gerar gráficos com uma variável independente *vs* dependente. Um exemplo simples de aplicação é dado abaixo.

```
dados_2=read.table("Dados.txt",header=TRUE)
```

```
head(dados_2,n=6)
```

```
##      Ger Dias Classe
## 1 0.00    0     G1
## 2 2.50    2     G1
## 3 15.00   4     G1
## 4 25.75   6     G1
## 5 28.75   8     G1
## 6 30.25   10    G1
```

```
tail(dados_2,n=6)
```

```
##      Ger Dias Classe
## 27 49.50   4     G4
## 28 58.00   6     G4
## 29 64.25   8     G4
## 30 66.00   10    G4
## 31 67.00   12    G4
## 32 67.00   14    G4
```

Usando a função `plot()` é possível gerar um gráfico de *Ger vs Dias*. Utilizando o argumento `xlab`, `ylab` e `main()` é possível nomear a abcissa, a ordenada e inserir um título no gráfico, respectivamente. Com o argumento `type` é possível modificar o corpo do gráfico. O argumento `pch` modifica a representação (forma dos pontos) das observações e o argumento `lwd` a espessura de pontos ou linhas. Como serão gerados quatro gráficos, a função `par=mfrow(c(2,2))` será utilizada para dividir a tela onde os gráficos são gerados em quatro partes.

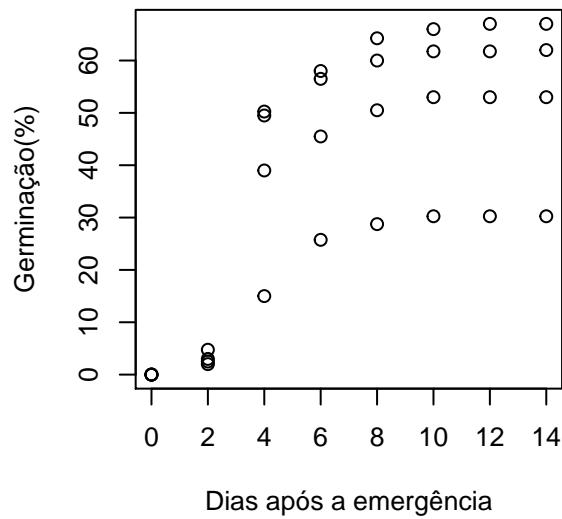
```
par(mfrow=c(2,2)) # divide a tela em 4 partes
```

```
plot(Ger~Dias, data=dados_2, xlab="Dias após a emergência",
      ylab="Germinação(%)", main = "Germinação ao longo dos dias")
plot(Ger~Dias, data=dados_2, xlab="Dias após a emergência",
      ylab="Germinação(%)", type="l", main = "Germinação ao longo dos dias")
plot(Ger~Dias, data=dados_2, xlab="Dias após a emergência",
      ylab="Germinação(%)", pch=2, main = "Germinação ao longo dos dias")
plot(Ger~Dias, data=dados_2, xlab="Dias após a emergência",
      ylab="Germinação(%)", type="l", lwd=3, main = "Germinação ao longo dos dias")
```

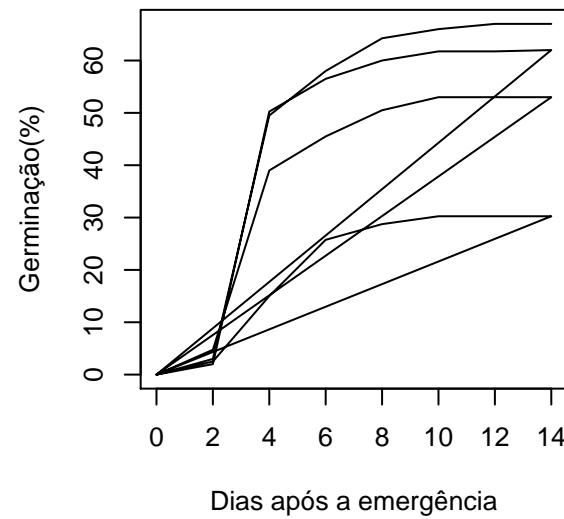
Ao dividir a tela, é possível separar os genótipos utilizando o argumento `subset()`. Também é possível alterar a coloração dos pontos ou linhas através do argumento `color()`.

```
par(mfrow=c(2,2))
plot(Ger~Dias, data=subset(dados_2,Classe=="G1"),xlab="Dias após a emergência"
      ,ylab="Germinação(%)",col="red")
```

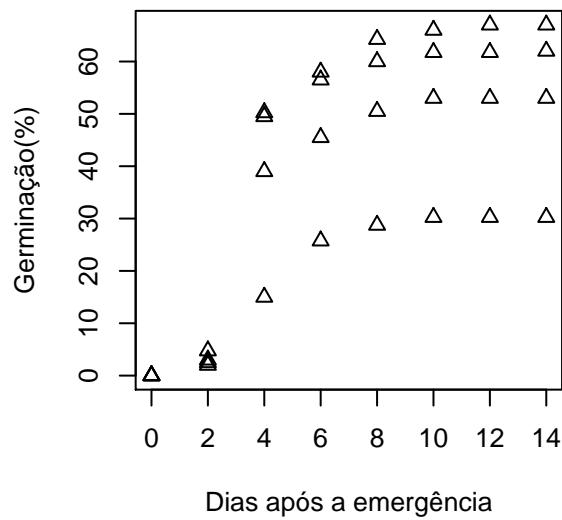
**Germinação ao longo dos dias**



**Germinação ao longo dos dias**



**Germinação ao longo dos dias**



**Germinação ao longo dos dias**

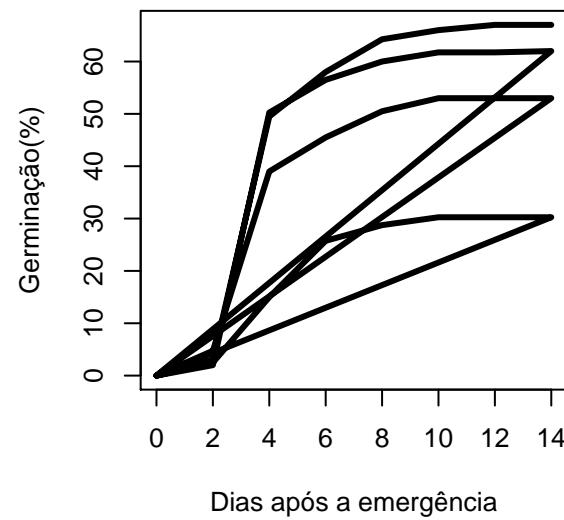


Figure 11: Dividindo a tela para plotagem de múltiplos gráficos

```

plot(Ger~Dias, data=subset(dados_2,Classe=="G2"),xlab="Dias após a emergência"
      ,ylab="Germinação(%)",col="black")
plot(Ger~Dias, data=subset(dados_2,Classe=="G3"),xlab="Dias após a emergência"
      ,ylab="Germinação(%)",col="green")
plot(Ger~Dias, data=subset(dados_2,Classe=="G4"),xlab="Dias após a emergência"
      ,ylab="Germinação(%)",col="blue")

```

É possível gerar um gráfico com as observações de apenas uma das classes. Posteriormente, as observações de uma segunda classe são adicionados através da função `point()`. Os pontos serão diferenciados pela cor (preto e vermelho) e pelo tipo de ponto (círculo e triângulo), e uma legenda será adicionada utilizando a função `legend()`. O argumento `pch` indica o tipo de ponto, `pch` as cores e `bty="n"` que a legenda não possui contorno.

```

par(mfrow=c(2,2))

### Legenda no canto inferior direito
plot(Ger~Dias, data=subset(dados_2,Classe=="G1"),
      xlab="Dias após a emergência", ylab="Germinação(%)", col="black")
points(Ger~Dias,data=subset(dados_2,Classe=="G2"),col="red", pch=2)
legend("bottomright", legend = c("Classe 1", "Classe 2"), pch=c(1,2),
       col = c("black", "red"), bty = "n")

### Legenda no canto inferior esquerdo
plot(Ger~Dias, data=subset(dados_2,Classe=="G1"),
      xlab="Dias após a emergência", ylab="Germinação(%)", col="black")
points(Ger~Dias,data=subset(dados_2,Classe=="G2"),col="red", pch=2)
legend("bottomleft", legend = c("Classe 1", "Classe 2"), pch=c(1,2),
       col = c("black", "red"), bty = "n")

### Legenda no canto superior direito
plot(Ger~Dias, data=subset(dados_2,Classe=="G1"),
      xlab="Dias após a emergência", ylab="Germinação(%)", col="black")
points(Ger~Dias,data=subset(dados_2,Classe=="G2"),col="red", pch=2)
legend("topright", legend = c("Classe 1", "Classe 2"), pch=c(1,2),
       col = c("black", "red"), bty = "n")

### Legenda no canto superior esquerdo
plot(Ger~Dias, data=subset(dados_2,Classe=="G1"),
      xlab="Dias após a emergência", ylab="Germinação(%)", col="black")
points(Ger~Dias,data=subset(dados_2,Classe=="G2"),col="red", pch=2)
legend("topleft", legend = c("Classe 1", "Classe 2"), pch=c(1,2),
       col = c("black", "red"), bty = "n")

```

Percebe-se que na função `plot()` as variáveis dependentes e independentes são declaradas como `Ger~Dias`. Porém, a função `plot()` pode receber como argumento uma outra função. Por exemplo, se declararmos como argumento uma função `aov()` (realiza a análise de variância) dentro da função `plot()`, ela retorna gráficos de diagnóstico de resíduos .

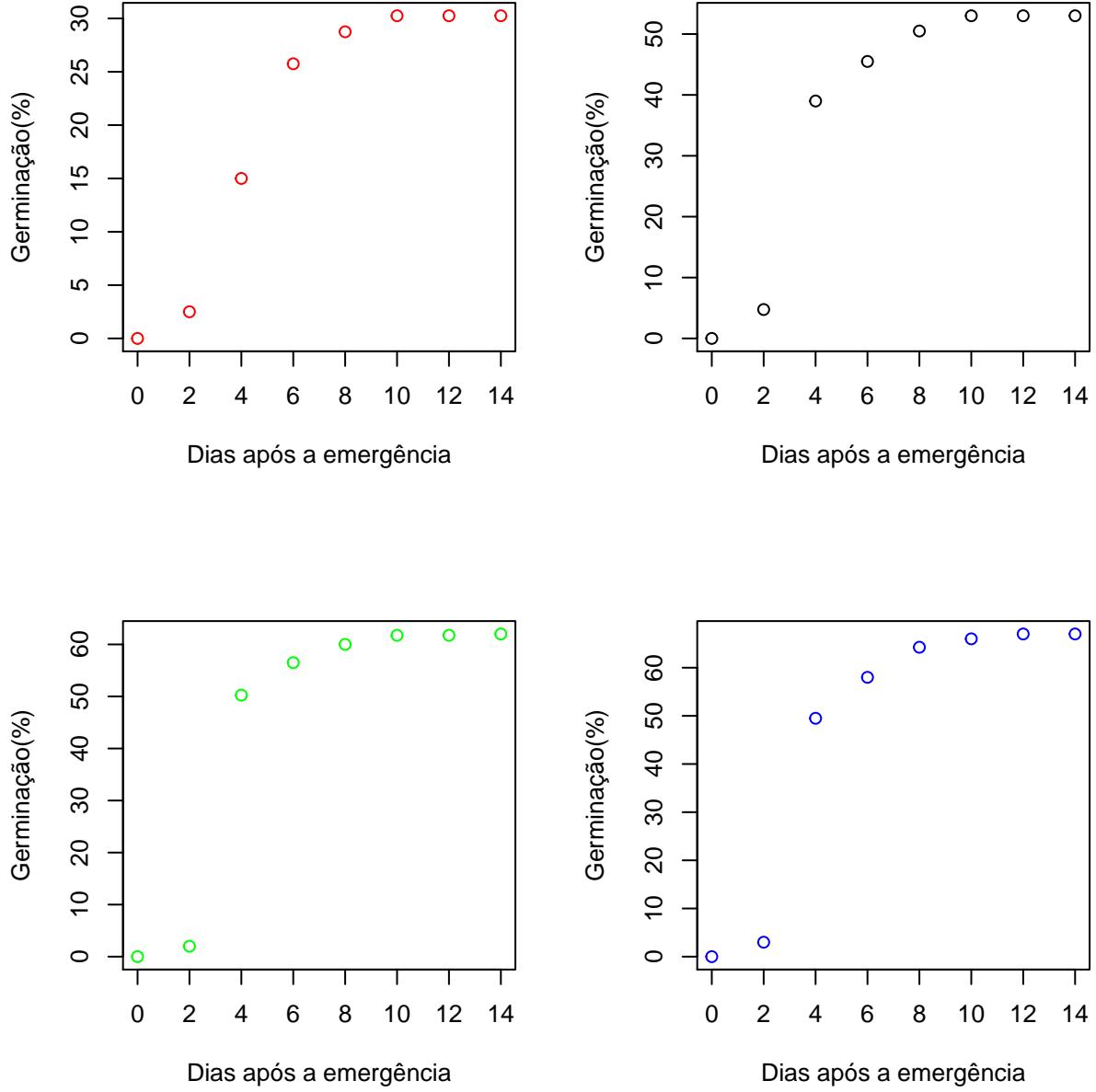


Figure 12: Alterando a coloração dos pontos ou linhas através do argumento color()

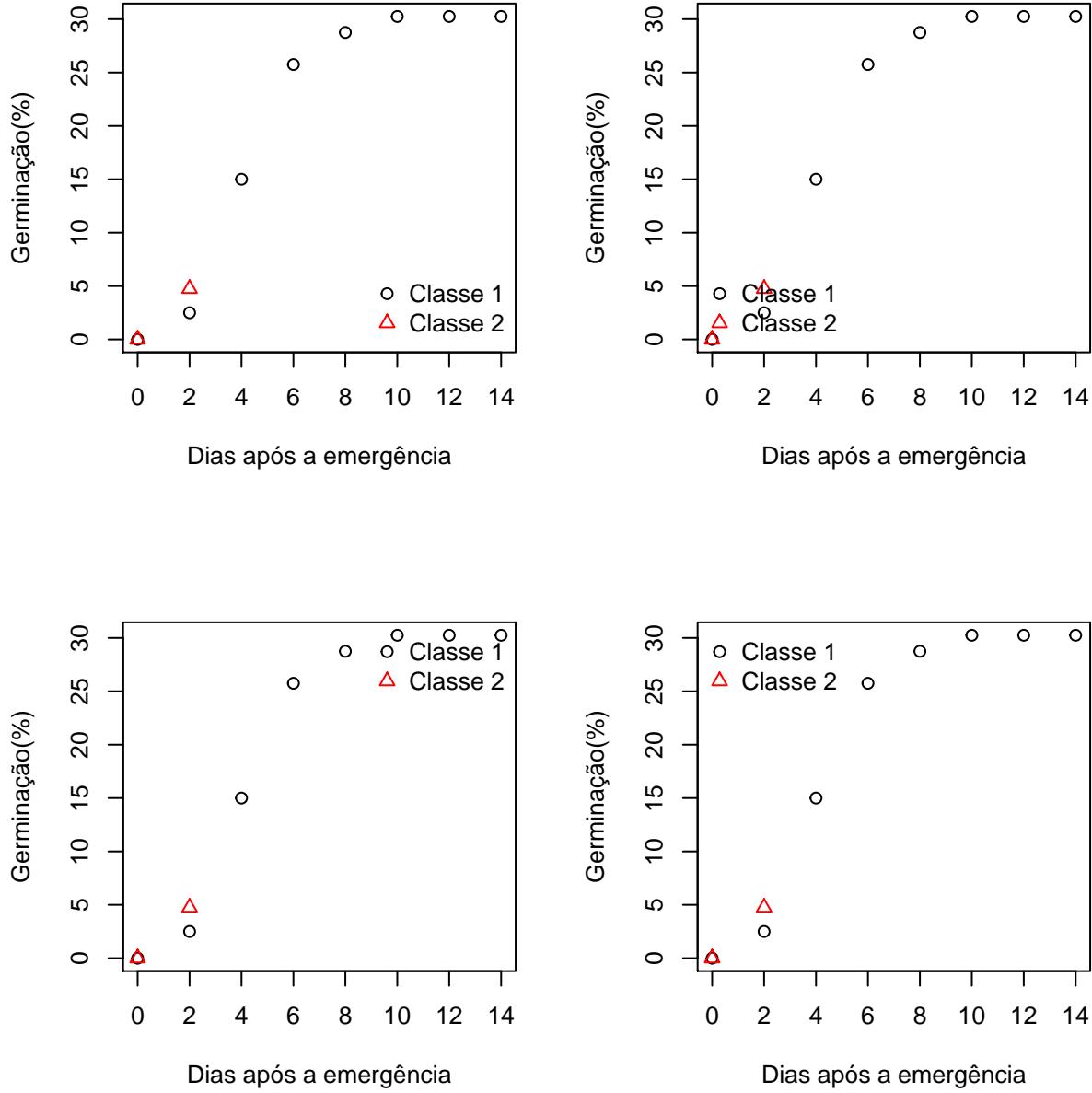


Figure 13: Modificando a posição da legenda em um gráfico

```

par(mfrow=c(2,2))
dados_3=read.table("Dados_3.txt",header=TRUE)
a=aov(a~Bloco+Trat, data=dados_3)
plot(a)

```

### 1.6.2 Histogramas utilizando a função hist()

Esta função retorna um histograma de frequências. A identificação dos eixos é realizada com os mesmos argumentos da função plot(). Utilizaremos o argumento `main = ""` para que o gráfico fique sem títulos. Por *default*, o gráfico é construído utilizando as frequências absolutas. Utilizando o argumento `freq=FALSE` as densidades de probabilidade são plotadas. Através da função `lines()` vamos adicionar linhas de uma distribuição normal ao histograma.

```

par(mfrow=c(2,2))
hist(dados_3$a, xlab = "Valores observados",
      ylab = "Frequência absoluta", main = "Histograma")

hist(dados_3$a, xlab = "Valores observados",
      ylab = "Frequência relativa", main = "Histograma", freq=FALSE)

norm=density(rnorm(64,411.5091,166.912))
hist(dados_3$a, xlab = "Valores observados",
      ylab = "Frequência relativa", main = "Histograma", freq=FALSE)
lines(norm)

hist(dados_3$a, xlab = "Valores observados",
      ylab = "Frequência relativa", main = "Histograma", freq=FALSE)
lines(norm, lty=2, lwd=2)

```

### 1.6.3 Gráficos quantil-quantil utilizando a função qqnorm()

Esta função é muito útil para verificar a normalidade dos resíduos da ANOVA e regressões lineares ou não lineares. A programação abaixo foi utilizada no artigo de Lucio and Sari (2017) para demonstrar a interpretação dos gráficos.

```

par(mfrow=c(2,2))
dados_esquerda=rbeta(1000,1,7)
dados_direita=rbeta(1000,7,1)
dados_normais=rnorm(1000,0.5,0.15)

hist(dados_normais,xlab = "Valores observados",
      ylab = "Frequência absoluta", main = "Normal")

qqnorm(dados_normais,main = "Normal")

```

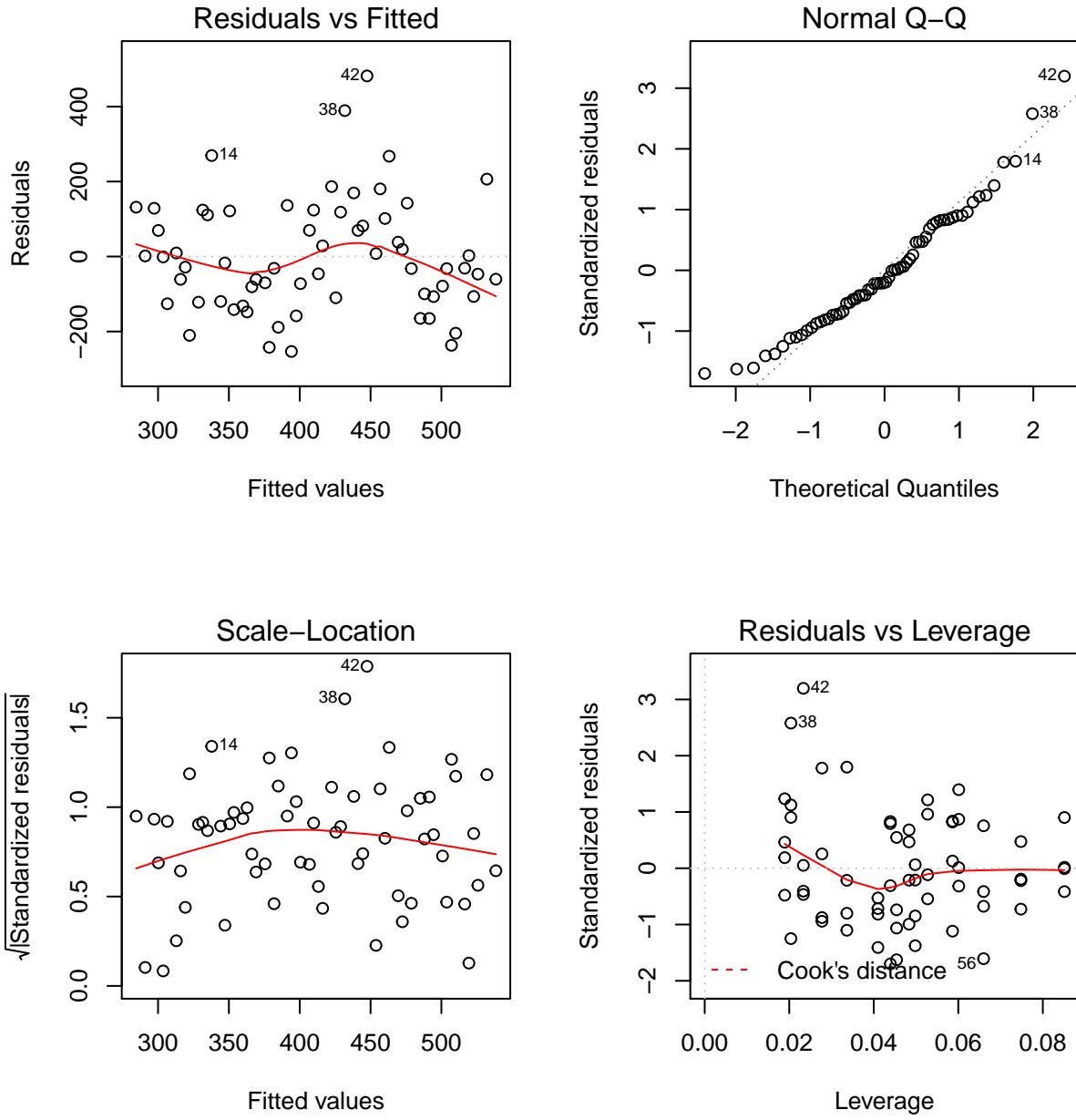


Figure 14: Gráficos de diagnóstico de resíduos utilizando a função `plot()`

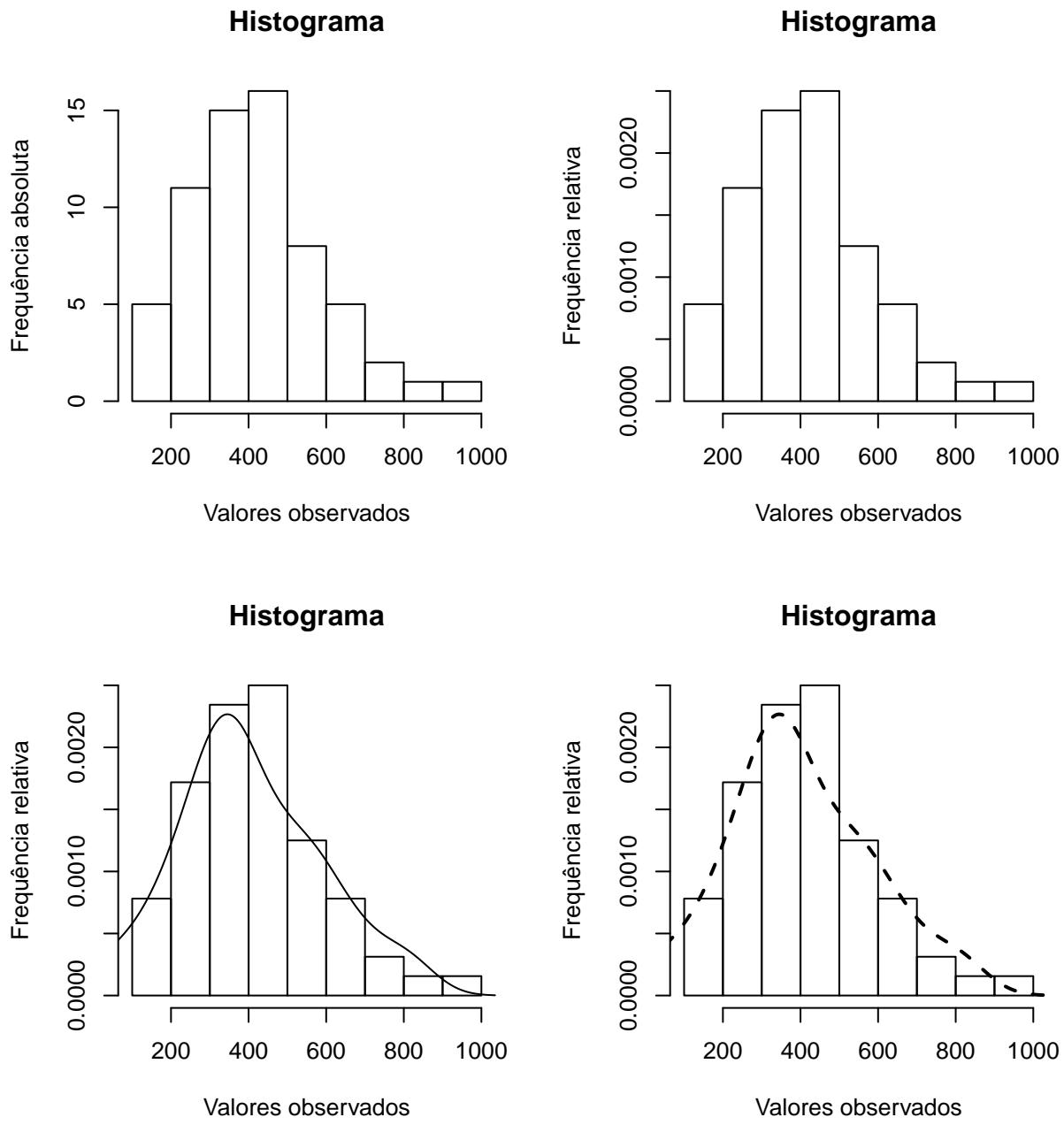


Figure 15: Histograma de frequencias

```

qqline(dados_normais)

hist(dados_direita,xlab = "Valores observados",
     ylab = "Frequência absoluta", main = "Assimétrico")

qqnorm(dados_direita,main = "Assimétrico", ylim=c(0.4, 1.2))
qqline(dados_direita)

```

#### 1.6.4 Gráfico de caixas usando a função `boxplot()`

O gráfico de caixas também é muito útil para verificar a distribuição de determinada variável. A caixa é delimitada pelo terceiro quartil (75%) e pelo primeiro quartil (25%). Uma linha (ou ponto) marca o terceiro quartil (mediana = 50%).

```

par(mfrow=c(2,2))
dados_3$Trat=as.factor(dados_3$Trat)
boxplot(dados_normais, main = "Normal")
boxplot(dados_direita, main = "Assimétrico à direita")
boxplot(dados_esquerda, main = "Assimétrico à esquerda")
boxplot(a~Trat,dados_3, main = "Avaliações no tempo") ## Plotando várias caixas

```

#### 1.6.5 Gráficos usando a função `curve()`

A função `curve` pode ser utilizada para traçar as curvas geradas por regressões lineares ou não lineares. Ela pode ser utilizada em conjunto com outras funções gráficas, como `plot()` por exemplo.

```

##### Polinômio de segundo grau
Y = c(64,53,46,56,39,46)
Trat = c(15,20,25,30,35,40)

Model=function(x,b0,b1,b2){
  b0+b1*x+b2*(x^2)
}

plot(Y~Trat, xlab="Doses", ylab="Produção", pch=16, ylim=c(10,100), xlim=c(15,40))
curve(Model(x,b0=107.128571,b1=-4.201429,b2=0.072857),add=TRUE,lty=2,lwd=2)
legend("top", legend = c("107.128571-4.201429x+0.072857x2"),bty="n")
legend("bottomleft", legend = c("Observado","Estimado"),
       lty=c(NA,2),pch=c(16,NA), lwd=c(2,2), bty="n")

##### Adicionando duas curvas
Model2=function(x,b0,b1){
  b0+b1*x
}
plot(Y~Trat, xlab="Doses", ylab="Produção", pch=16, ylim=c(10,100), xlim=c(15,40))

```

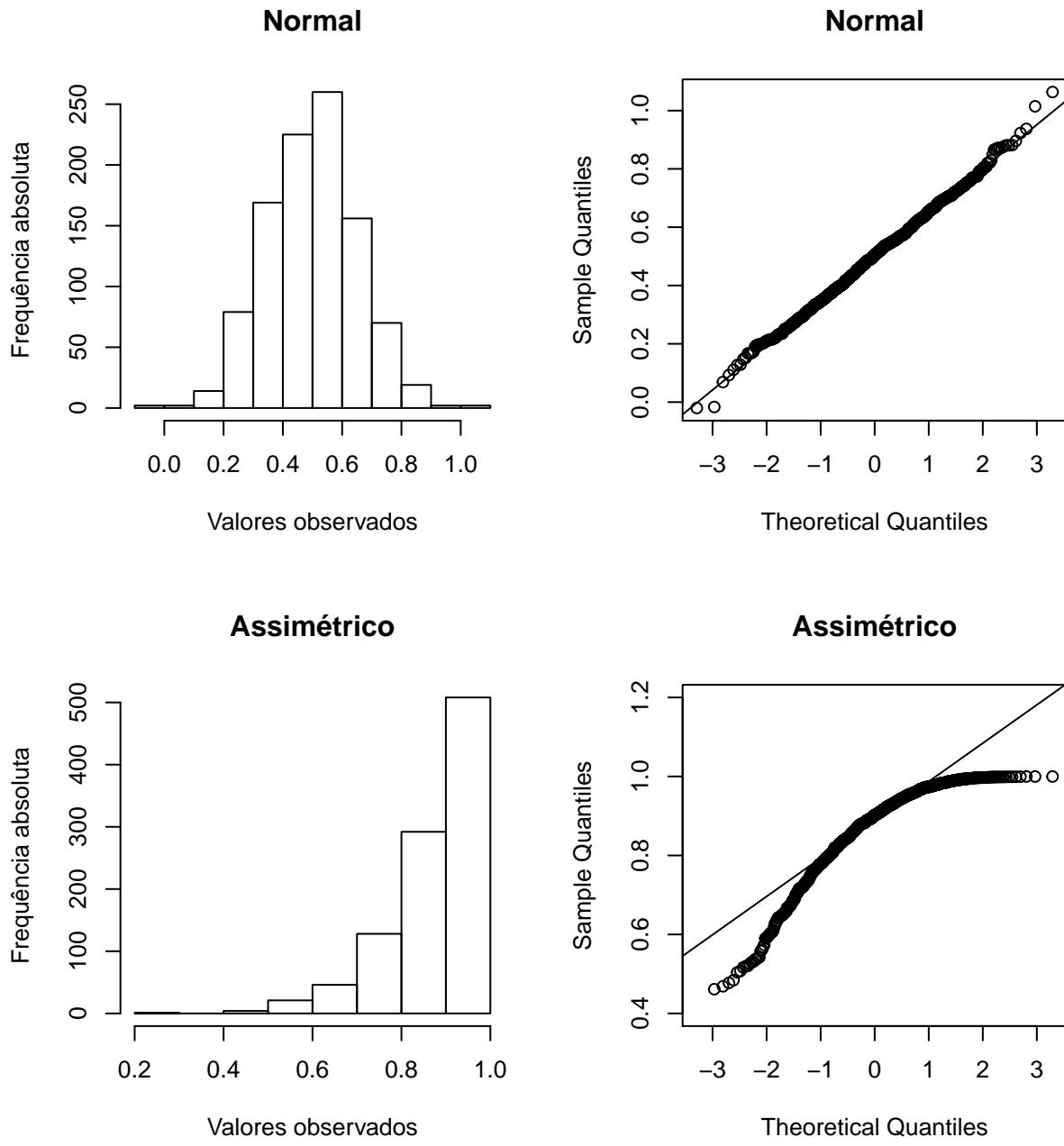
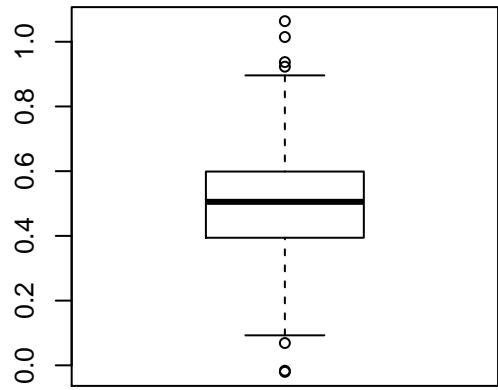
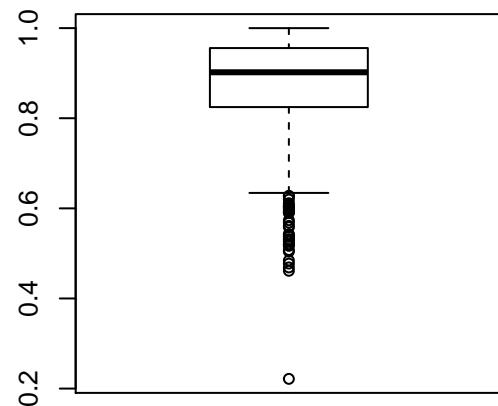


Figure 16: Gráficos quantil-quantil utilizando a função `qqnorm()`

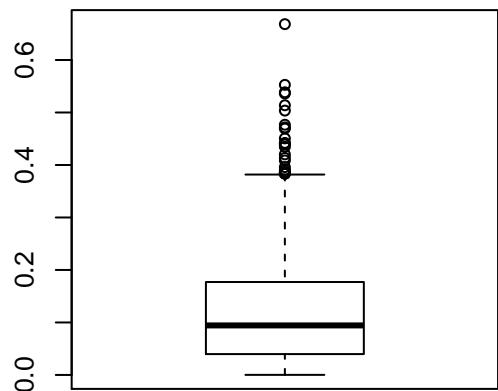
**Normal**



**Assimétrico à direita**



**Assimétrico à esquerda**



**Avaliações no tempo**

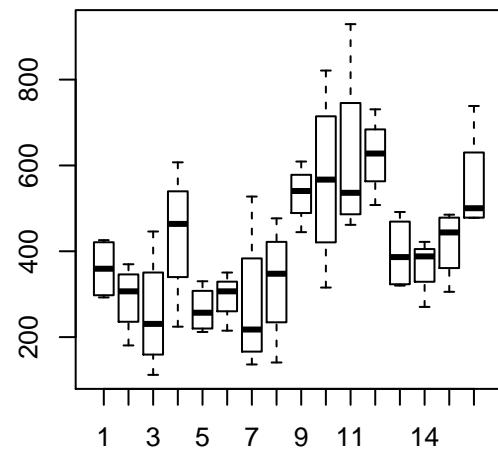


Figure 17: Gráfico de caixas usando a função boxplot()

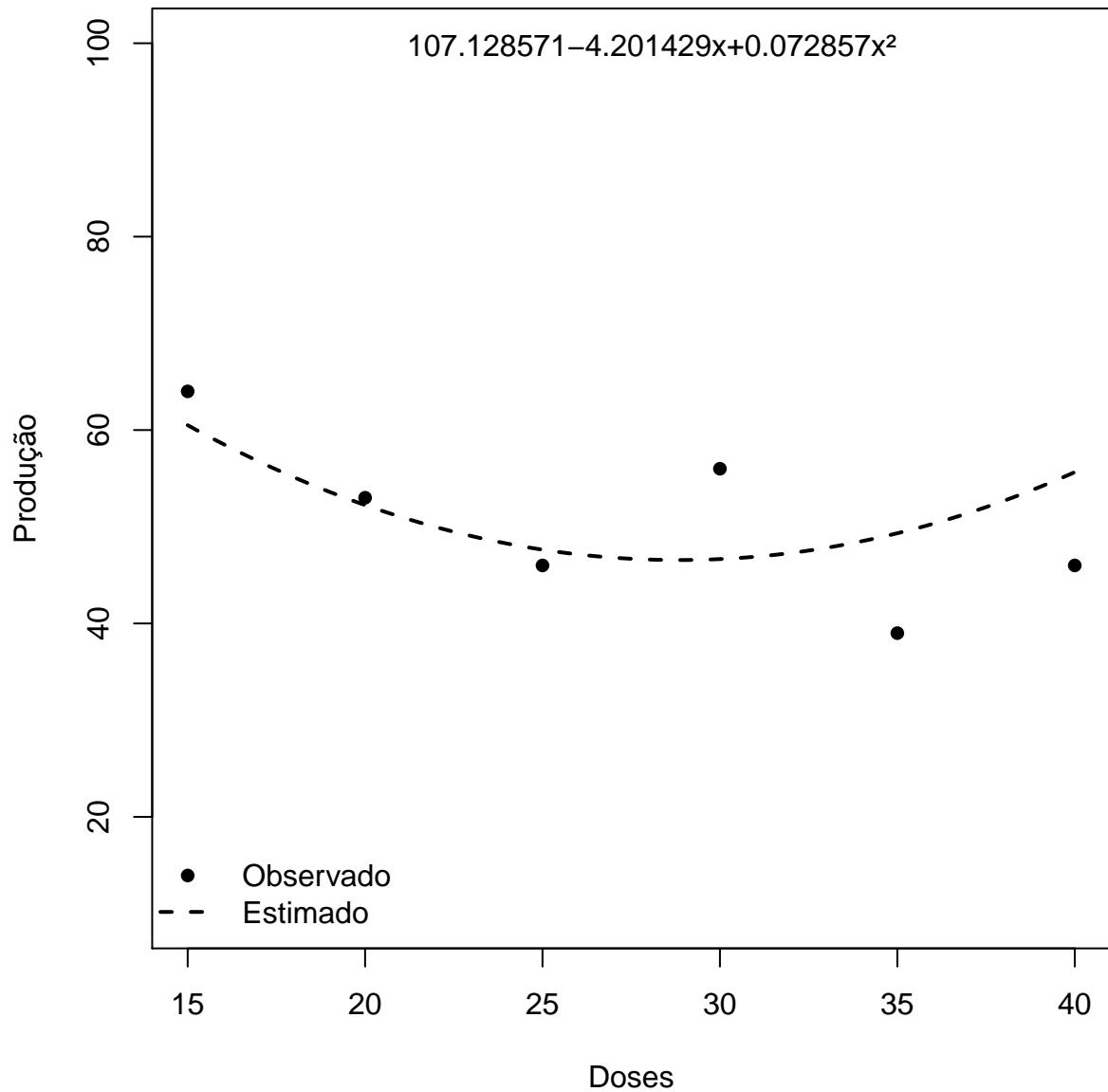


Figure 18: Gráficos usando a função curve()

```

curve(Model(x,b0=107.128571,b1=-4.201429,b2=0.072857),add=TRUE,lty=2,lwd=2)
curve(Model2(x,b0=57.342857,b1=-0.194286),add=TRUE,lty=3,lwd=2)
legend("bottomleft", legend = c("Observado", "1º Grau", "2º Grau"),
lty=c(NA,3,2),pch=c(16,NA,NA), lwd=c(2,2,2), bty="n")

```

## 1.7 Exportando dados

Será demonstrado nessa seção como exportar dados e tabelas gerados dentro do *software R*. Quanto a saída de resultados, será dado enfase a saídas com extensões .csv, .txt e .xlsx. Quanto aos gráficos, será mostrado como salvar figuras em alta resolução.

### 1.7.1 Exportando com diferentes extensões

#### Salvar em extensão .csv

Em arquivos .csv, os valores são separados por vírgula. Tabelas geradas no *R* podem ser salvos através da função `write.table()`. Será utilizado como exemplo uma ANOVA, onde o objetivo é exportar os *p*-valores do teste F para blocos e tratamentos.

```

mod1=aov(a~Trat+factor(Bloco), data=dados_3)
anova(mod1)

## Analysis of Variance Table
##
## Response: a
##              Df  Sum Sq Mean Sq F value    Pr(>F)
## Trat          15 1009319   67288  4.1587 0.0001013 ***
## factor(Bloco)  3   17728    5909  0.3652 0.7784260
## Residuals     45  728109   16180
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
names(anova(mod1))

## [1] "Df"      "Sum Sq"   "Mean Sq"  "F value" "Pr(>F)"
```

A função `names()` mostra o que é armazenado dentro do objeto `anova(mod1)`. utilizando \$ selecionamos `Pr(>F)`.

```

anova(mod1)$"Pr(>F)"

## [1] 0.0001012958 0.7784259554           NA
```

Por fim, armazenamos os valores utilizando a função `write.table()`, indicando onde os resultados estão armazenados, o nome do arquivo que será exportado (“NOME\_ARQUIVO.csv”).

```

p.value=cbind(c(anova(mod1)$"Pr(>F)"[1],anova(mod1)$"Pr(>F)"[2]))
write.table(p.value,"p.valor_mod1.csv",row.names = FALSE)
```

#### Salvar em extensão .txt

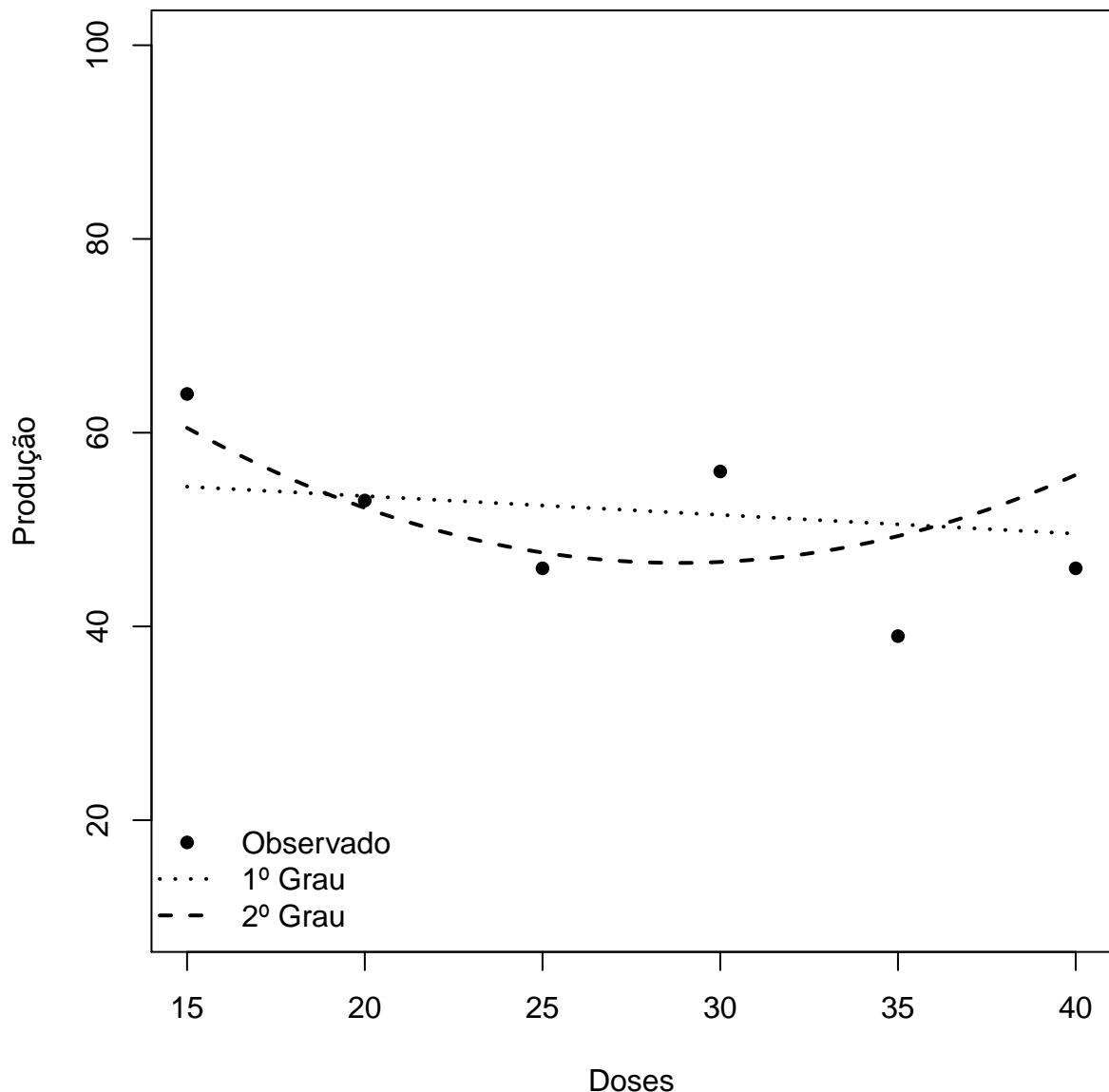


Figure 19: Gráficos usando a função curve()

Para salvar as saídas em extensão .txt utiliza-se a função `sink()`, indicando o nome do arquivo que será exportado (“NOME\_ARQUIVO.txt”).

```
sink("Modelo_1.txt")
cat("----- MODELO -----", "\n")
print(mod1)
cat("\n")
cat("----- ANOVA -----", "\n")
print(anova(mod1))
sink()
```

### Salvar em extensão . xls e .xlsx

Para salvar arquivos em extensão .xlsx usaremos a função `write.xlsx()` do pacote *xlsx*. É necessário também que o pacote *rJava* esteja instalado e seja carregado. Na função `write.xlsx()` é necessário indicar o nome do objeto que se deseja exportar e o “caminho” do diretório com o nome do arquivo (“NOME\_ARQUIVO.xls” ou “NOME\_ARQUIVO.xlsx”).

```
require(rJava)
require(xlsx)
write.xlsx(x22, "figuresTabela_1.xls")
write.xlsx(x22, "figuresTabela_1.xlsx")
```

### 1.7.2 Exportanto gráficos

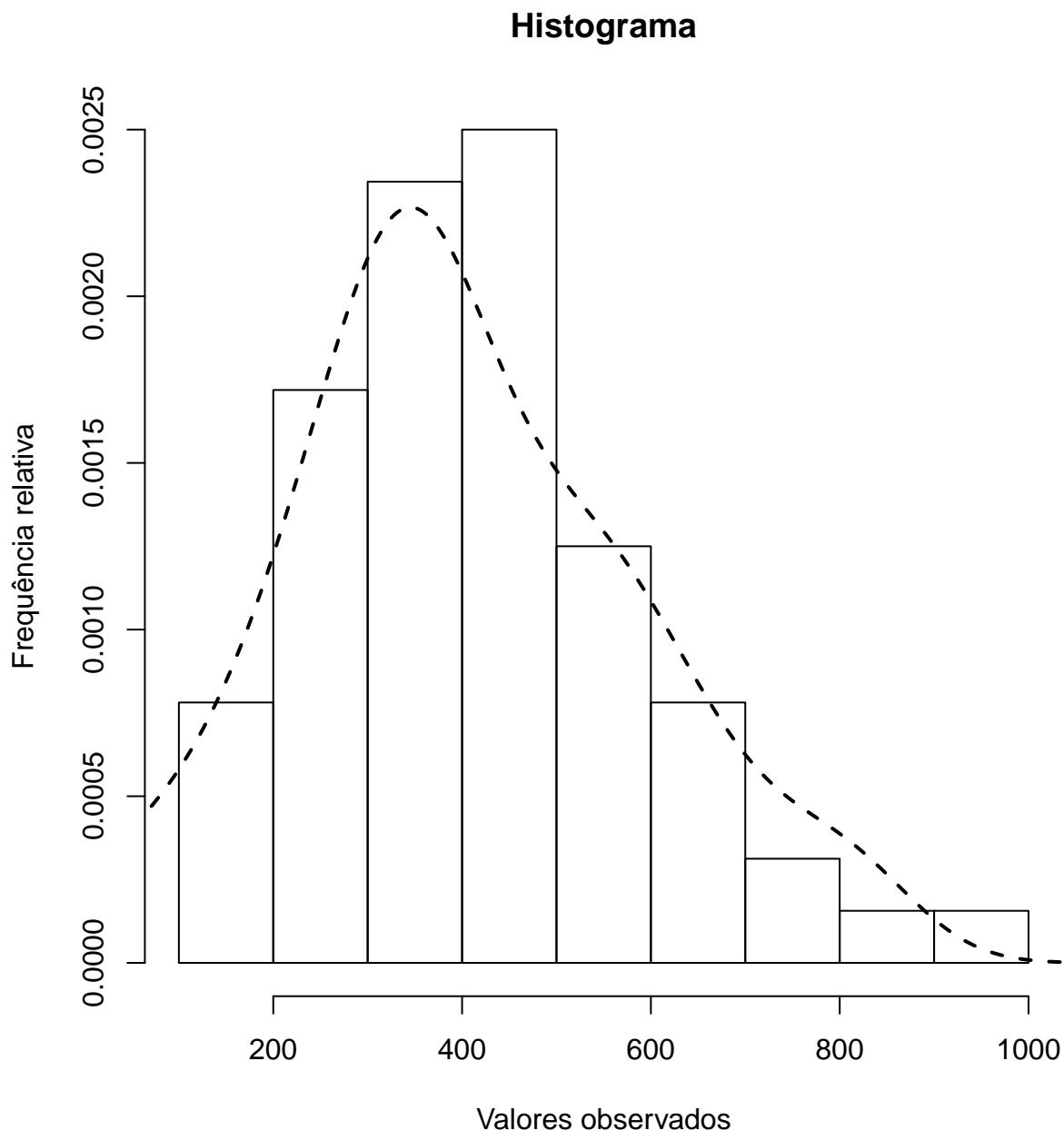
Os gráficos podem ser exportados clicando em *Export* no *output* dos gráficos. Você pode escolher entre salvar como imagem ou como PDF. Os formatos de imagem disponíveis são: .PNG, .TIFF, .JPEG, .BMP , .SVG e .ESP . A outra opção é salvar em um arquivo PDF . A principal diferença entre estes formatos é o método de renderização utilizado na formação do gráfico. Existem basicamente dois métodos de renderização de imagens: *raster images* e *vector-based images*. Imagens rasterizadas usam muitos pixels coloridos ou blocos de construção individuais para formar uma imagem completa. JPEGs, GIFs e PNGs e TIFFs são tipos de imagem raster mais comuns. Como as imagens raster são criadas usando um número fixo de pixels coloridos, elas não podem ser redimensionadas drasticamente sem comprometer sua resolução. Quando esticados para caber em um espaço que eles não foram projetados para preencher, seus pixels ficam visivelmente granulados e a imagem é distorcida. É importante que você salve os arquivos *raster* precisamente nas dimensões necessárias e com a devida resolução para uma boa apresentação. Para salvar estas imagens é necessário informar a *Density of Pixels per Inch (DPI)* , ou seja, quantos pontos por polegada quadrada deverá conter a imagem. Quanto maior este valor, maior será a qualidade da imagem e também maior será seu tamanho (em Mb).

Imagens vetoriais (*vector-based graphics*), por outro lado, permitem mais flexibilidade. Construídos usando fórmulas matemáticas em vez de blocos coloridos individuais, os tipos de arquivos vetoriais, como .PDF e .EPS \*, são excelentes para criar gráficos de alta resolução sem que seu redimensionamento prejudique a qualidade do gráfico. A maioria das revistas científicas aceitam todos estes tipos de formatos. Na dúvida, escolha sempre o formato .PDF (ou .EPS)!

```

hist(dados_3$a, xlab = "Valores observados",
     ylab = "Frequência relativa", main = "Histograma", freq=FALSE)
lines(norm, lty=2, lwd=2)

```



Uma alternativa prática para salvar em pdf as imagens é através da função `pdf()`. Os argumentos `width` e `height` correspondem a dimensão da figura (em polegadas) e `pointsize` corresponde ao tamanho da fonte.

```

pdf("Figura1.pdf", width = 9, height = 8, pointsize = 25)
hist(dados_3$a, xlab = "Valores observados",

```

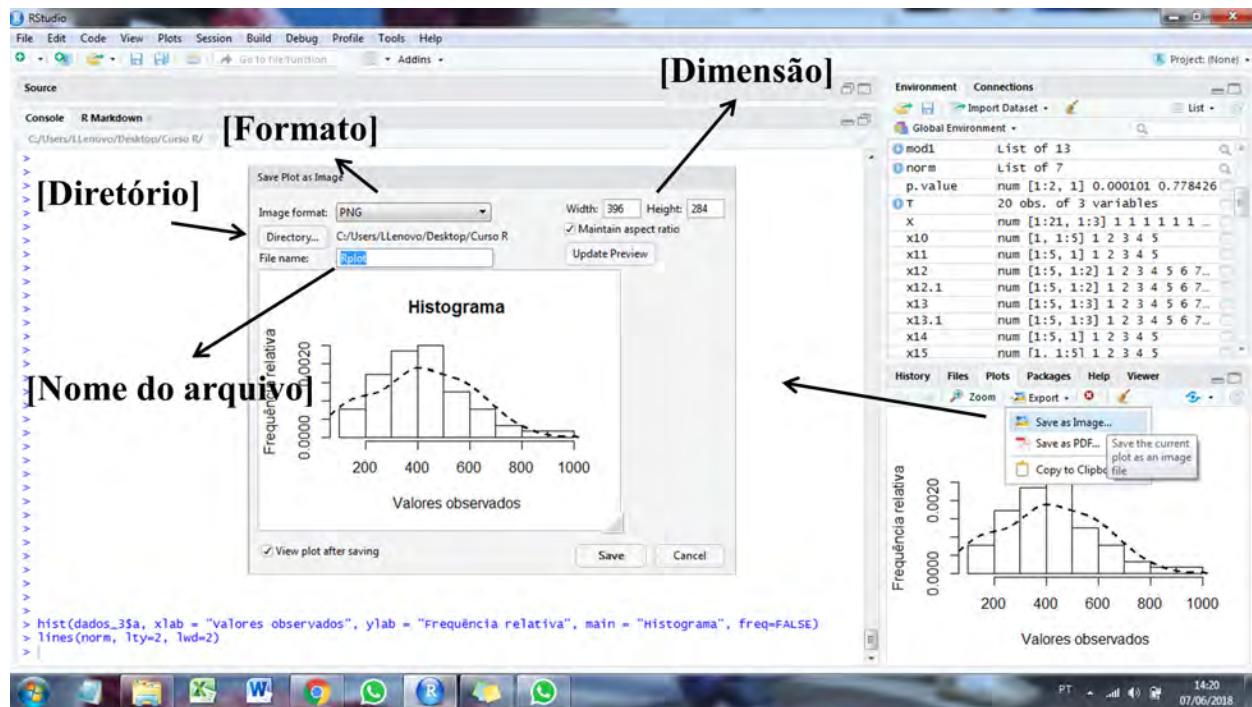


Figure 20: Salvando figuras como imagens

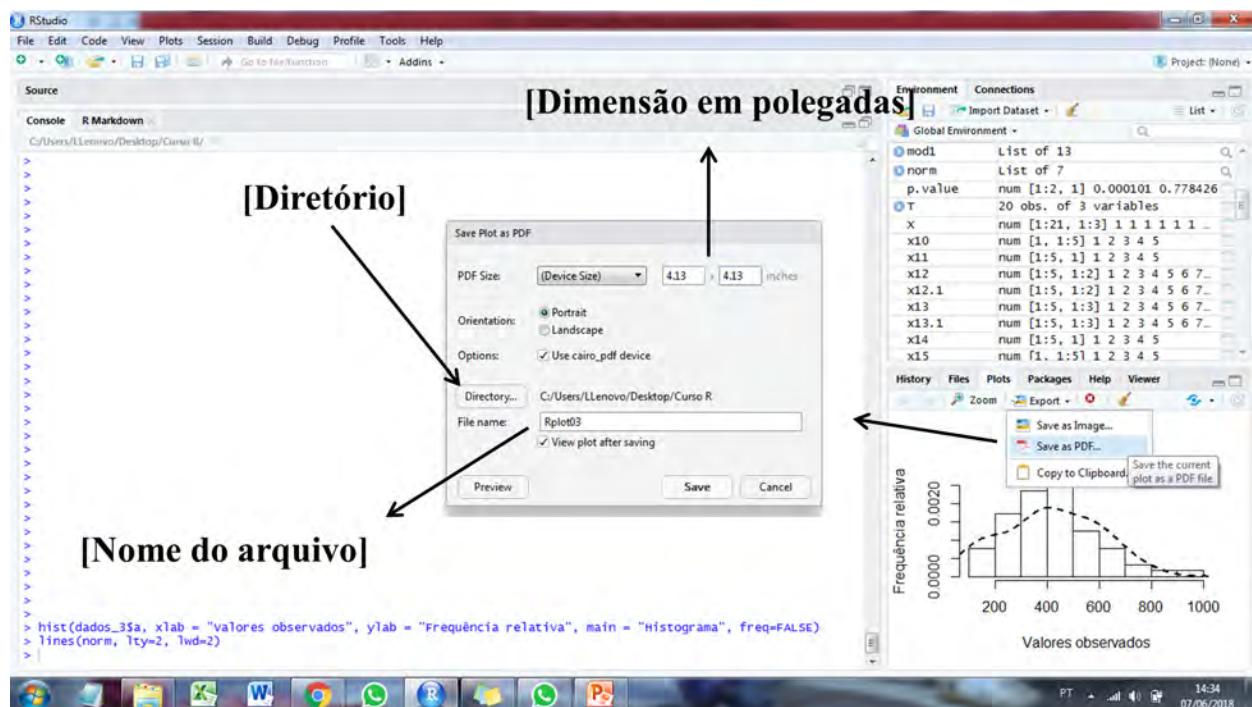


Figure 21: Salvando figuras em PDF

```

ylab = "Frequênci a relativa", main = "Histograma", freq=FALSE)
lines(norm, lty=2, lwd=2)
dev.off()

É importante notar que a figura só será salva após o comando dev.off() for executado. Portanto, para gerar várias figuras ao mesmo tempo, basta colocá-las uma embaixo da outra:

pdf("Figura2.pdf", width = 9, height = 8, pointsize = 25)
##### Polinômio de segundo grau
Y = c(64, 53, 46, 56, 39, 46)
Trat = c(15, 20, 25, 30, 35, 40)

Model=function(x,b0,b1,b2){
  b0+b1*x+b2*(x^2)
}
plot(Y~Trat, xlab="Doses", ylab="Produção", pch=16, ylim=c(10,100), xlim=c(15,40))
curve(Model(x,b0=107.128571,b1=-4.201429,b2=0.072857),add=TRUE,lty=2,lwd=2)
legend("top", legend = c("107.128571-4.201429x+0.072857x²"), bty="n")
legend("bottomleft", legend = c("Observado", "Estimado"),
       lty=c(NA,2),pch=c(16,NA), lwd=c(2,2), bty="n")

##### Adicionando duas curvas
Model2=function(x,b0,b1){
  b0+b1*x
}
plot(Y~Trat, xlab="Doses", ylab="Produção", pch=16, ylim=c(10,100), xlim=c(15,40))
curve(Model(x,b0=107.128571,b1=-4.201429,b2=0.072857),add=TRUE,lty=2,lwd=2)
curve(Model2(x,b0=57.342857,b1=-0.194286),add=TRUE,lty=3,lwd=2)
legend("bottomleft", legend = c("Observado", "1º Grau", "2º Grau"),
       lty=c(NA,3,2),pch=c(16,NA,NA), lwd=c(2,2,2), bty="n")
dev.off()

```

## Exportando figuras como imagens em alta qualidade

Com as funções `tiff()` , `png()` , `jpeg()` e `bmp()` é possível salvar gráficos como imagem em alta resolução. Como na função `pdf()` , `width` e `height` correspondem a dimensão. Porém, nestas funções é possível escolher a unidade através do argumento `units` e a resolução através do argumento `res`.

```

tiff(filename="Figura3.tiff", width=16, height=16, units="cm", pointsize=12,
      "lzw", res=1200)
hist(dados_3$a, xlab = "Valores observados", ylab = "Frequênci a relativa",
      main = "Histograma", freq=FALSE)
lines(norm, lty=2, lwd=2)
dev.off()

png(filename="Figura3.png", width=16, height=16, units="cm", pointsize=12, res=1200)
hist(dados_3$a, xlab = "Valores observados", ylab = "Frequênci a relativa",

```

```

  main = "Histograma", freq=FALSE)
lines(norm, lty=2, lwd=2)
dev.off()

jpeg(filename="Figura3.jpeg",width=16,height=16,units="cm",pointsize=12,res=1200)
hist(dados_3$a, xlab = "Valores observados", ylab = "Frequência relativa",
      main = "Histograma", freq=FALSE)
lines(norm, lty=2, lwd=2)
dev.off()

bmp(filename="Figura3.bmp",width=16,height=16,units="cm",pointsize=12,res=1200)
hist(dados_3$a, xlab = "Valores observados", ylab = "Frequência relativa",
      main = "Histograma", freq=FALSE)
lines(norm, lty=2, lwd=2)
dev.off()

```

## 2 Análise de dados experimentais

Nesta seção será abordado aspectos relacionados a análise de experimentos agrícolas. Ressalta-se que como o tempo é curto, será dado enfase a testes paramétricos. Dividiremos essa seção em três partes:

**Parte 1:** medidas de tendência central e de variabilidade. Intervalos de confiança para média. Testes de hipóteses para verificar a igualdade entre médias de uma ou duas amostras.

**Parte 2:** delineamentos experimentais inteiramente casualizado (DIC) e blocos ao acaso (DBC). pressupostos dos modelos estatísticos, testes complementares (média e regressão) e transformação de dados.

**Parte 3:** experimentos fatoriais e experimentos com parcelas subdivididas.

### 2.1 Estatística básica

Nesta seção mostraremos como calcular medidas de tendência central e medidas de variabilidade. As medidas de tendência central são valores que representam um conjunto de dados. Entre as mais comuns podemos citar a média, mediana e moda.

```

mean(dados_normais) ## média

## [1] 0.5002164

median(dados_normais) ## mediana

## [1] 0.5052466

```

O *R* calcula a média e mediana através das funções `mean()` e `median()`, porém não calcula a moda. No blog Ridículas, mantido pelo LEG da UFPR, dois métodos são discutidos.

Incentivamos a leitura do material. Já o desvio padrão e a variância podem ser obtidas com as funções `sd()` e `var()`:

```
sd(dados_normais)
```

```
## [1] 0.1520033
```

```
var(dados_normais)
```

```
## [1] 0.02310502
```

Para calcular o coeficiente de variação, pode-se usar a função `function()`:

```
CV=function(dados){  
  class=class(dados)  
  if(class=="numeric"){ #Indica que os dados devem ser numéricos  
    media=mean(dados)  
    sd=sd(dados)  
    CV=sd/media*100  
  }  
  return(CV) ## Valor que será retornado pela função  
}
```

```
CV(dados_normais)
```

```
## [1] 30.38752
```

A distribuição dos dados pode ser determinada através de histograma de frequências, QQ-Plos e Box-Plot (conforme visto anteriormente). Estatísticas como a amplitude, erro padrão da média, intervalo de confiança, entre outros, podem ser obtidas pela função `stat.desc()` do pacote *pastecs*. A vantagem deste pacote é que um objeto com muitas variáveis (*data.frame*, por exemplo) pode ser utilizada como argumento.

```
require(pastecs)  
head(Ex.gen) ## dados de três genótipos
```

```
##          G1          G2          G3  
## 1  9.552228 43.40519 25.61442  
## 2  9.116528 34.59915 23.49314  
## 3  9.029065 42.06568 25.70684  
## 4 10.719547 41.35450 24.88697  
## 5 10.687592 41.99486 22.45169  
## 6  9.859372 38.73699 27.44829
```

```
tail(Ex.gen) ## 20 observações
```

```
##          G1          G2          G3  
## 15  9.628439 35.43038 27.48142  
## 16 10.292712 39.23545 25.15524  
## 17  9.648813 41.52780 29.42458  
## 18  9.552438 40.25018 27.22714
```

```

## 19 11.568640 41.93661 26.91160
## 20 8.570283 40.63130 29.92856

```

```
stat.desc(Ex.gen)
```

	G1	G2	G3
## nbr.val	20.0000000	20.0000000	20.0000000
## nbr.null	0.0000000	0.0000000	0.0000000
## nbr.na	0.0000000	0.0000000	0.0000000
## min	8.57028299	34.59914616	20.36521294
## max	11.56864033	43.40518930	29.92855937
## range	2.99835735	8.80604314	9.56334643
## sum	201.12866490	786.58501441	516.66406245
## median	10.02706829	39.74281494	25.66062964
## mean	10.05643324	39.32925072	25.83320312
## SE.mean	0.16726596	0.58712128	0.50941242
## CI.mean.0.95	0.35009168	1.22885896	1.06621244
## var	0.55955804	6.89422797	5.19002021
## std.dev	0.74803612	2.62568619	2.27816159
## coef.var	0.07438384	0.06676166	0.08818734

Testes de aderência a distribuições teóricas também são de grande utilizada para as ciências agrárias. O teste de Shapiro-Wilk, através da função `shapiro.test()` , é amplamente utilizada para realizar o teste de normalidade dos dados. Para testar a aderência a outras distribuições teóricas, o teste de Kolmogorov-Smirnov (função `ks.test()`) é uma alternativa.

```

Amostra1=rnorm(100,5,10) ## Gera uma amostra com distribuição normal
Amostra2=rpois(100,12) ## Gera uma amostra com distribuição Poisson
Amostra3=rnorm(100,5,10)
shapiro.test(Amostra1)

```

```

##
## Shapiro-Wilk normality test
##
## data: Amostra1
## W = 0.98232, p-value = 0.2008
shapiro.test(Amostra2)

```

```

##
## Shapiro-Wilk normality test
##
## data: Amostra2
## W = 0.96473, p-value = 0.008869
ks.test(Amostra1,Amostra2)

```

```

##
## Two-sample Kolmogorov-Smirnov test
##
```

```

## data: Amostra1 and Amostra2
## D = 0.48, p-value = 1.972e-10
## alternative hypothesis: two-sided
ks.test(Amostra1,Amostra3)

##
## Two-sample Kolmogorov-Smirnov test
##
## data: Amostra1 and Amostra3
## D = 0.08, p-value = 0.9062
## alternative hypothesis: two-sided

```

Uma maneira de verificar a normalidade graficamente através de um gráfico dos valores observados *vs* uma distribuição teórica.

```

hist(dados_normais, xlab="",ylab="",freq=FALSE,main="", nclass = 20)
mean=mean(dados_normais)
sd=sd(dados_normais)
pois=density(rpois(1000,mean))
lines(pois)

```

## 2.2 Intervalos de confiança e teste de hipóteses

A estimativa por intervalo não fornece idéia da margem de erro cometida ao estimar determinado parâmetro (Ferreira 2009). Por isso, para verificar se uma dada hipótese  $H_0$  (de igualdade) é ou não verdadeira, deve-se utilizar intervalos de confiança ou testes de hipóteses. A construção destes intervalos, e as particularidades dos testes de hipóteses, serão discutidos a seguir. Recomendamos como literatura o livro *Estatística Básica* do prof. Daniel Furtado Ferreira da UFV (ver aqui). Para verificar a normalidade dos dados, as funções `shapiro.test()` e `ks.test()` e os gráficos Qq-Plot são de grande utilidade.

### Normalidade dos dados

É importante ressaltar que o curso terá como enfase a análise de dados com distribuição normal. Será demonstrado como testar hipóteses para uma e duas médias pelo teste  $t$  de Student, o que exige que os dados tenham distribuição normal univariada (já discutido anteriormente) ou bivariada (dados emparelhados). Para testar a normalidade bivariada, basta testar a normalidade da diferença entre as variáveis:

```

Amostra1=rnorm(100,10,10)
Amostra2=rnorm(100,10,24)
Amostra3=Amostra1-Amostra2
shapiro.test(Amostra3)

```

```

##
## Shapiro-Wilk normality test
##
## data: Amostra3

```

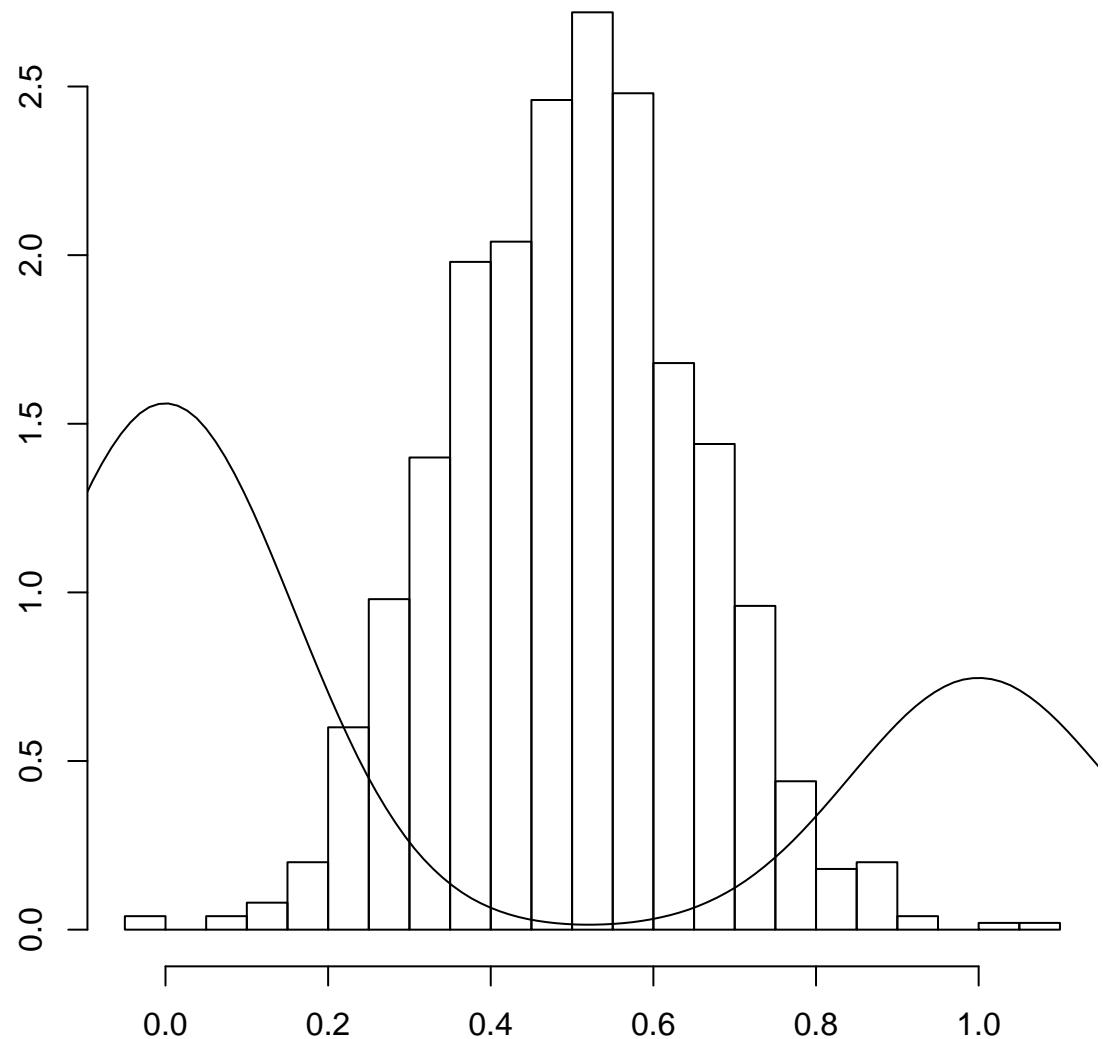


Figure 22: Gráfico dos valores observados vs uma distribuição teórica

```
## W = 0.98829, p-value = 0.5298
```

### 2.2.1 Intervalo de confiança

A partir de um intervalo que tenha alta probabilidade de conter o valor paramétrico, é possível diferenciar duas estimativas (Ferreira 2009). Devido ao pouco tempo, será dado enfase a construção de intervalos de confiança para a média. O intervalo de confiança de uma média amostral de 95% é dado por:

$$P \left[ \bar{X} - t_{\alpha/2} \frac{S}{\sqrt{n}} \leq \mu \leq \bar{X} + t_{\alpha/2} \frac{S}{\sqrt{n}} \right] = 1 - \alpha$$

Na expressão acima,  $\bar{X}$  é a média,  $S$  é o desvio padrão e  $-t_{\alpha/2}$  e  $+t_{\alpha/2}$  são os quantis inferior e superior, respectivamente, da distribuição  $t$  de Student. O intervalo acima indica que o valor do parâmetro ( $\alpha$ ) tem 95% de chance de estar contido no intervalo. Ressalta-se que a expressão acima está relacionada com a precisão e não com a acurácia da estimativa. Para calcular esse intervalo, podemos utilizar a função `t.test()`.

```
result=t.test(dados_normais)
result$conf.int ## Intervalo de confiança
```

```
## [1] 0.4907838 0.5096489
## attr(,"conf.level")
## [1] 0.95
result$estimate ## média
```

```
## mean of x
## 0.5002164
```

O intervalo de confiança é, por *default*, de 95%. Poém, pode-se modificar através do argumento `conf.level`.

```
result=t.test(dados_normais,conf.level = 0.99)
result1=t.test(dados_normais,conf.level = 0.90)
result$conf.int ## Intervalo de confiança
```

```
## [1] 0.4878112 0.5126215
## attr(,"conf.level")
## [1] 0.99
result1$conf.int ## Intervalo de confiança
```

```
## [1] 0.4923026 0.5081301
## attr(,"conf.level")
## [1] 0.9
```

Para gerar os gráficos de intervalo de confiança será utilizada a função `ggplot()` do pacote `ggplot2`.

```

require(ggplot2)

result=t.test(dados_normais)

dados_IC=data.frame(
  "Factor"=c("Amostra 1"),
  "LL"=c(result$conf.int[1]), # armazena o limite inferior
  "Mean"=c(mean(dados_normais)), # armazena a média
  "UL"=c(result$conf.int[2]) # armazena o limite superior
)

ggplot(data=dados_IC, aes(y = Mean, x = Factor, colour=Factor)) +
  ## indica os dados e as variáveis
  geom_point(size = 2.5,color="red")+
  ## adiciona um ponto ao gráfico
  geom_errorbar(aes(## adiciona a barra de erro
    ymax = UL, ## valor mínimo
    ymin = LL), ## valor máximo
    width = 0.1,
    color="red")+
  coord_flip()+
  ## "inverte" o "x" e o "y"
  expand_limits(y=c(0.45,0.55)) ## declarar "invertido"

```

Para fazer um gráfico com mais de uma variável:

```

require(ggplot2)

Amostra1=rnorm(100,10,10)
Amostra2=rnorm(100,10,24)
Amostra3=rnorm(100,25,15)
Amostra4=rnorm(100,20,30)

results1=t.test(Amostra1)
results2=t.test(Amostra2)
results3=t.test(Amostra3)
results4=t.test(Amostra4)

require(ggplot2)
dados_IC=data.frame(
  "Factor"=c("Amostra 1", "Amostra 2", "Amostra 3", "Amostra 4"),
  "LL"=c(results1$conf.int[1], results2$conf.int[1], results3$conf.int[1],
          results4$conf.int[1]),
  "Mean"=c(mean(Amostra1), mean(Amostra2), mean(Amostra3),
           mean(Amostra4)),
  "UL"=c(results1$conf.int[2], results2$conf.int[2], results3$conf.int[2],
          results4$conf.int[2]))

```

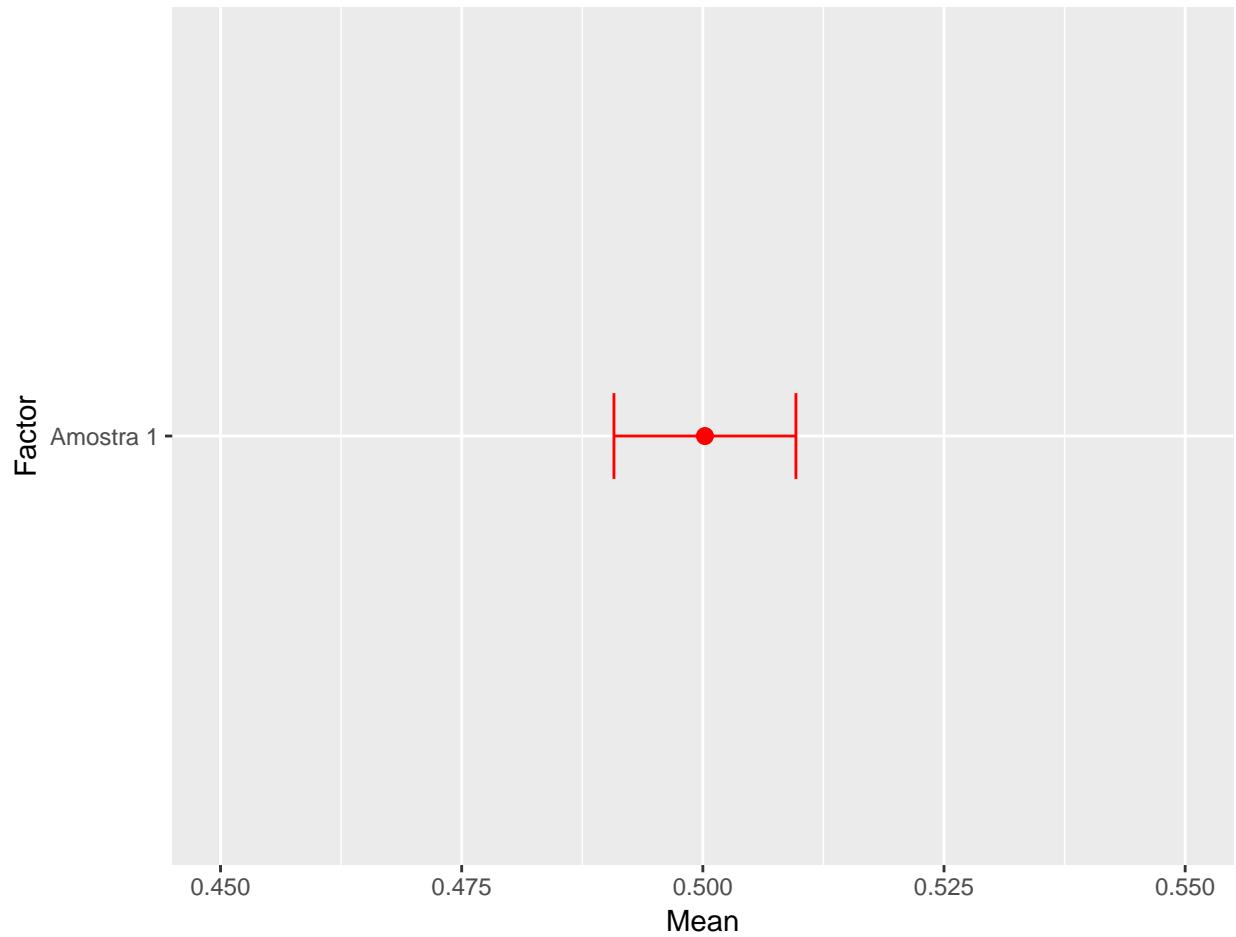


Figure 23: Gráficos de intervalo de confiança (uma variável)

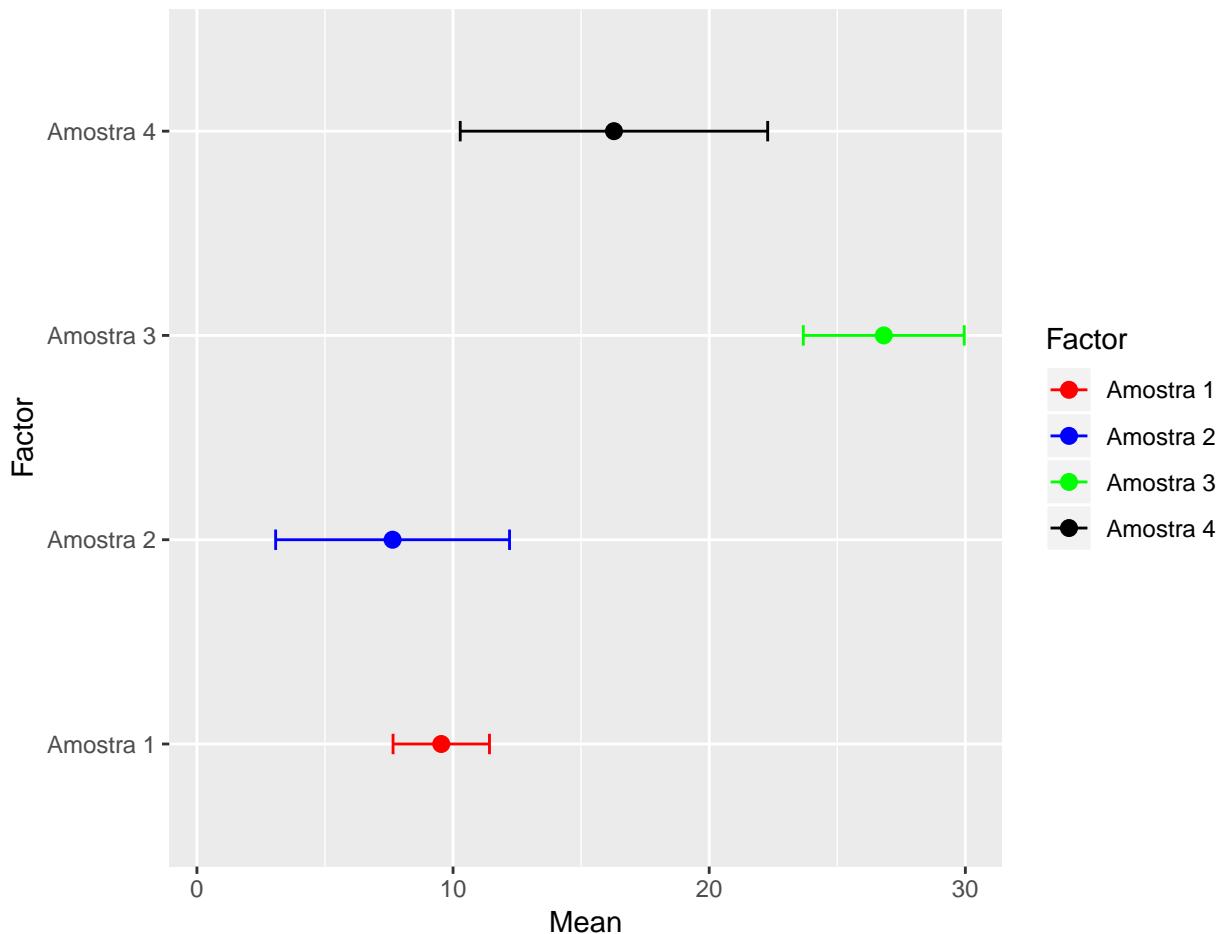


Figure 24: Gráficos de intervalo de confiança (mais de uma variável)

```

## Gráfico com barras de erros
ggplot(data=dados_IC, aes(y = Mean, x = Factor, colour=Factor)) +
  # indica os dados e as variáveis
  geom_point(size = 2.5) + # adiciona um ponto ao gráfico
  geom_errorbar(aes(# adiciona a barra de erro
    ymax = UL, # valor mínimo
    ymin = LL), # valor máximo
    width = 0.1) +
  scale_color_manual(values = c("red","blue","green","black")) + ## Diferencia por cores
  coord_flip() + ## "inverte" o "x" e o "y"
  expand_limits(y=c(0.4,0.6)) ## declarar "invertido"

## Gráfico de barras com barras de erros
g<- ggplot(data=dados_IC, aes(y = Mean, x = Factor)) +
  geom_bar(aes(fill = Factor), stat="identity", position="dodge") +
  ## Armazena o gráfico em "g"
  expand_limits(y=c(0,30))

```

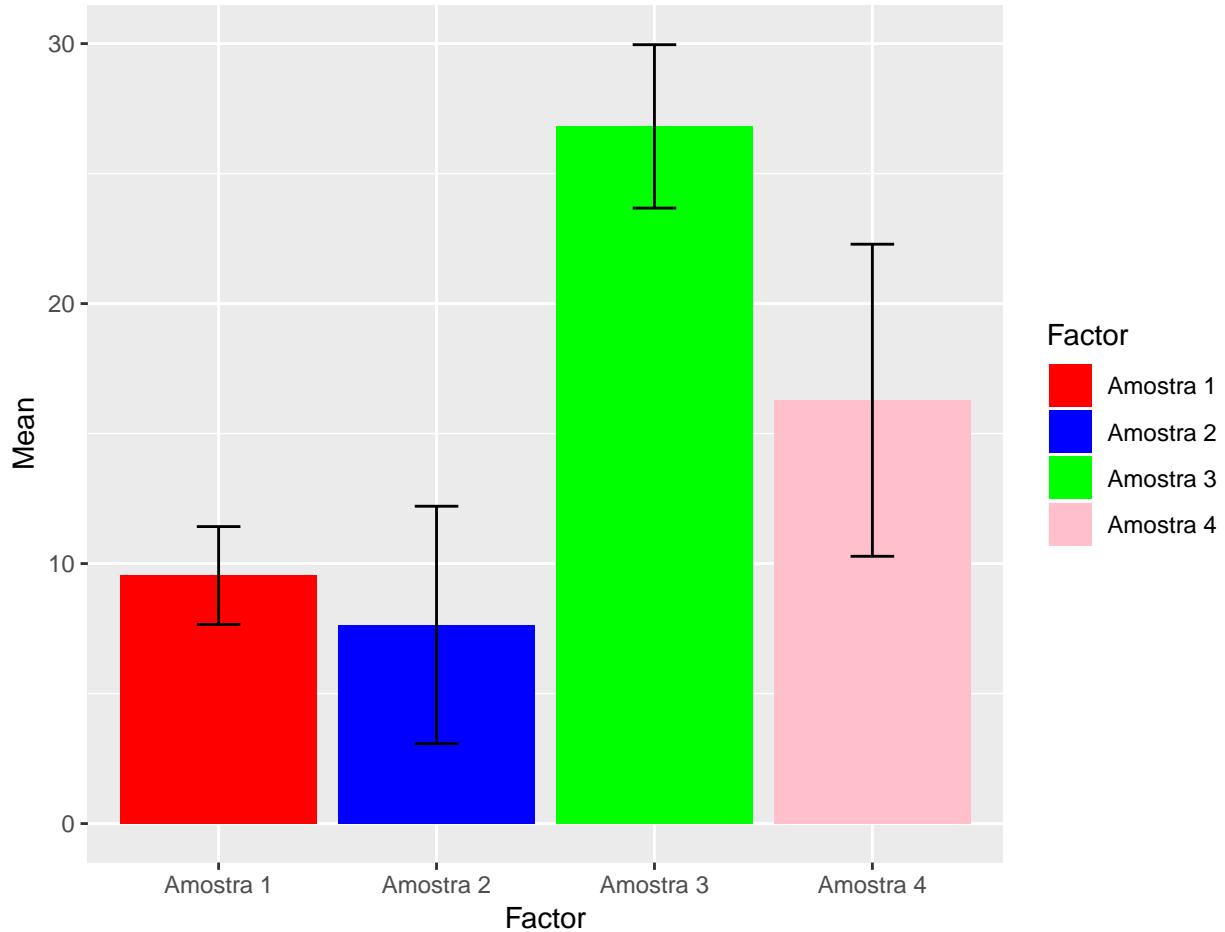


Figure 25: Gráficos de intervalo de confiança (mais de uma variável)

```
cbPalette <- c("red", "blue", "green", "pink") ## armazenando as cores
g+scale_fill_manual(values=cbPalette)+  
geom_errorbar(aes(# adiciona a barra de erro  
ymax = UL, # valor mínimo  
ymin = LL), # valor máximo  
width = 0.2)
```

## 2.2.2 Teste de hipóteses

Os testes de hipóteses aqui demonstrados tem como objetivo a) verificar se determinada amostra difere ou não de zero ( $H_0 : \mu = 0$ ) e b) se duas amostras são ou não iguais ( $H_0 : \mu_1 = \mu_2$ ). Para testar as hipóteses pode-se utilizar a função `t.test()`:

```
t.test(Amostra1) # testa se a amostra difere de zero
```

```
##  
## One Sample t-test
```

```

## 
## data: Amostra1
## t = 10.057, df = 99, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  7.65778 11.42241
## sample estimates:
## mean of x
## 9.540094

t.test(Amostra1,Amostra2) # testa se as amostras difrem entre si

```

```

## 
## Welch Two Sample t-test
## 
## data: Amostra1 and Amostra2
## t = 0.76359, df = 131.71, p-value = 0.4465
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -3.022947 6.823999
## sample estimates:
## mean of x mean of y
## 9.540094 7.639569

```

O teste t pode ser utilizado para testar tanto hipóteses do tipo  $H_A : \mu > 0$  ou  $H_A : \mu < 0$ . Porém, para isso, devemos utilizar o argumento `alternative` para indicar que o teste utilizado é unilateral.

```
t.test(Amostra1, alternative="greater") # unilateral a direita
```

```

## 
## One Sample t-test
## 
## data: Amostra1
## t = 10.057, df = 99, p-value < 2.2e-16
## alternative hypothesis: true mean is greater than 0
## 95 percent confidence interval:
## 7.964975 Inf
## sample estimates:
## mean of x
## 9.540094

```

```
t.test(Amostra1, alternative="less") # unilateral a esquerda
```

```

## 
## One Sample t-test
## 
## data: Amostra1
## t = 10.057, df = 99, p-value = 1

```

```

## alternative hypothesis: true mean is less than 0
## 95 percent confidence interval:
##      -Inf 11.11521
## sample estimates:
## mean of x
## 9.540094

```

Outro pressuposto para realizar o teste t é a homogeneidade das variâncias. Quando as variâncias são heterogêneas, o grau de liberdade utilizado é calculado pela aproximação de Welch-Satterthwaite:

$$\nu \cong \frac{\left(\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}\right)^2}{\frac{\left(\frac{S_1^2}{n_1}\right)^2}{n_1-1} + \frac{\left(\frac{S_2^2}{n_2}\right)^2}{n_2-1}}$$

```
var.test(Amostra1, Amostra2) ## Teste F para variâncias
```

```

##
## F test to compare two variances
##
## data: Amostra1 and Amostra2
## F = 0.16996, num df = 99, denom df = 99, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
## 0.1143565 0.2526011
## sample estimates:
## ratio of variances
## 0.1699605

t.test(Amostra1, Amostra2, var.equal = FALSE) ## Por default, usa Welch-Satterthwaite

##
## Welch Two Sample t-test
##
## data: Amostra1 and Amostra2
## t = 0.76359, df = 131.71, p-value = 0.4465
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -3.022947 6.823999
## sample estimates:
## mean of x mean of y
## 9.540094 7.639569

Amostra1 = rnorm(30, 10, 5)
Amostra2 = rnorm(30, 18, 5)

var.test(Amostra1, Amostra2)

```

```

##  

## F test to compare two variances  

##  

## data: Amostra1 and Amostra2  

## F = 0.69763, num df = 29, denom df = 29, p-value = 0.3377  

## alternative hypothesis: true ratio of variances is not equal to 1  

## 95 percent confidence interval:  

## 0.3320469 1.4657160  

## sample estimates:  

## ratio of variances  

## 0.6976292  

t.test(Amostra1, Amostra2, var.equal = TRUE) ## Quando variâncias são iguais

##  

## Two Sample t-test  

##  

## data: Amostra1 and Amostra2  

## t = -5.8942, df = 58, p-value = 2.035e-07  

## alternative hypothesis: true difference in means is not equal to 0  

## 95 percent confidence interval:  

## -9.824851 -4.843396  

## sample estimates:  

## mean of x mean of y  

## 10.45246 17.78658

```

As formas de comparação discutidas acima consideram as amostras como sendo independentes entre si. Para dados emparelhados, deve-se utilizar o argumento `paired= TRUE`.

```

Amostra1 = rnorm(30,10,5)  

Amostra2 = rnorm(30,15,5)

var.test(Amostra1, Amostra2) ## Teste F para variâncias

##  

## F test to compare two variances  

##  

## data: Amostra1 and Amostra2  

## F = 0.67484, num df = 29, denom df = 29, p-value = 0.2953  

## alternative hypothesis: true ratio of variances is not equal to 1  

## 95 percent confidence interval:  

## 0.3212015 1.4178423  

## sample estimates:  

## ratio of variances  

## 0.674843

t.test(Amostra1, Amostra2, var.equal = TRUE, paired = TRUE) ## Emparelhadas

##

```

```

## Paired t-test
##
## data: Amostra1 and Amostra2
## t = -5.2117, df = 29, p-value = 1.408e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -8.102724 -3.535511
## sample estimates:
## mean of the differences
## -5.819117

t.test(Amostra1, Amostra2, var.equal = TRUE, paired = FALSE) ## Independentes

##
## Two Sample t-test
##
## data: Amostra1 and Amostra2
## t = -4.5077, df = 58, p-value = 3.235e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -8.403204 -3.235030
## sample estimates:
## mean of x mean of y
## 10.23740 16.05652

```

Observa-se que existe uma diferença no valor da estatística teste, nos graus de liberdade, no valor tabelado e, consequentemente, no  $p$ -valor. Para duas amostras independentes, o teste para a diferença é dado por

$$t_c = \frac{\bar{X}_1 - \bar{X}_2}{S \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \sim t_{(\alpha, \nu)}$$

Onde  $\alpha$  é a probabilidade de erro,  $\nu$  é o grau de liberdade ( $n^o$  total de observações-2),  $S$  é a média ponderada do desvio padrão,  $\bar{X}_1$  e  $\bar{X}_2$  são a média das amostras 1 e 2, respectivamente, e  $n_1$  e  $n_1$  e  $n_2$  são os tamanhos de amostra da amostra 1 e 2, respectivamente. No resultado acima, observamos que  $\nu = 58$ .

No caso de amostras pareadas, a estatística teste é dada por

$$t_c = \frac{\bar{d} - \mu_0}{\frac{S_d}{\sqrt{n}}} \sim t_{(\alpha, \nu)}$$

Onde  $\alpha$  é a probabilidade de erro,  $\nu$  é o grau de liberdade ( $n^o$  de diferenças-1),  $\bar{d}$  é a média das diferenças,  $S_d$  é o desvio padrão das diferenças e  $n$  é o número de diferenças. No resultado acima, observamos que  $\nu = 29$ .

## 2.3 Delineamentos básicos

As análises realizadas até agora tinham como objetivo verificar a existência de diferenças entre as médias de duas amostras. Porém, uma das análises estatísticas mais comuns, quando deseja-se estudar o efeito de “grupos de fatores” sobre determinado fenômeno, é a análise da variância (ANOVA). A ANOVA atribui a diversos fatores partes da variabilidade dos dados (Casella 2008).

Os delineamentos experimentais também são parte importante da ANOVA. Será dado mais destaque aos mais comuns: o delineamento inteiramente casualizado (DIC) e o blocos ao acaso (DBC). O bloqueamento tem como objetivo remover parte da variabilidade. Como não se deseja encotrar diferença entre os blocos, análises complementares não são realizadas para este fator.

Nessa seção será demostrado como analisar dados experimentais utilizando estes dois delineamentos. Em um primeiro momento, serão demonstrados experimentos unifatoriais e, mais adiante, experimentos bifatoriais com e sem parcelas subdivididas. Formas de como verificar se os pressupostos do modelo estatístico estão sendo cumpridos, e formas de contornar este problema caso estejam sendo violados, também serão demonstrados.

Por fim, após a análise da variância, serão mostrados os testes complementares utilizados para tratamentos quali e quantitativos.

### Princípios básicos

Os princípios básicos da experimentação são a *casualização* e a *repetição*. A repetição possibilita que o erro seja estimado, e a casualização que eles sejam independentes.

### Pressupostos

Independente do delineamento, os pressupostos do modelo estatístico são que os erros são independentes, homocedásticos e normais:

$$\varepsilon \sim N(0, \mathbf{I}\sigma^2)$$

As formas de realizar esse diagnóstico é através de testes estatísticos e gráficos de diagnósticos. Uma ferramenta para contornar o problema de violação dos pressupostos é a transformação dos dados. Existem várias formas de transformar os dados (cada uma adequada a um caso específico), mas será dado enfase à transformação Box-Cox.

### Estimação

A estimativa dos parâmetros é realizado pelo método dos mínimos quadrados. Devido a matriz delineamento ser de posto incompleto (modelo superparametrizado), uma restrição deve ser imposta ao fator tratamentos para que a estimativa do efeito dos fatores seja única ( $\sum t_i = 0$ ). Reparametrizações ou combinações lineares também podem ser utilizados para garantir a unicidade dos parâmetros (Rencher and Schaalje 2008).

### Reparametrização, condições marginais ou combinações lineares?

As funções `lm()` e `aov()` são usualmente utilizadas para realizar a análise da variância, e ambas utilizam a reparametrizam o modelo para estimar os parâmetros. Vamos ver isso no

exemplo hipotético abaixo:

```
#### Considerando os dados abaixo
dados=data.frame(
  "Tratamentos"=c(rep(1:4,each=4)),
  "Y"=dados_3$a[1:16]
)
dados

##      Tratamentos      Y
## 1          1 416.1087
## 2          1 292.4907
## 3          1 425.9184
## 4          1 302.5734
## 5          2 369.7512
## 6          2 180.5917
## 7          2 322.3873
## 8          2 290.9101
## 9          3 254.9141
## 10         3 111.9849
## 11         3 206.6612
## 12         3 446.0961
## 13         4 455.4335
## 14         4 607.2445
## 15         4 224.3256
## 16         4 472.1252

#### Usando a função aov()
mod1=aov(Y~factor(Tratamentos), data=dados)
mod1$coefficients

##             (Intercept) factor(Tratamentos)2 factor(Tratamentos)3
##               359.27278           -68.36271          -104.35871
## factor(Tratamentos)4
##               80.50939

mod2=lm(Y~factor(Tratamentos), data=dados)
mod2$coefficients

##             (Intercept) factor(Tratamentos)2 factor(Tratamentos)3
##               359.27278           -68.36271          -104.35871
## factor(Tratamentos)4
##               80.50939

#### Abordagem matricial
y=as.matrix(dados_3$a[1:16])
x=as.matrix(cbind(rep(1,each=16),c(0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0),
                 c(0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0),
                 c(0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1))) ## reparametrização
x
```

```

##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    1    0    0    0
## [3,]    1    0    0    0
## [4,]    1    0    0    0
## [5,]    1    1    0    0
## [6,]    1    1    0    0
## [7,]    1    1    0    0
## [8,]    1    1    0    0
## [9,]    1    0    1    0
## [10,]   1    0    1    0
## [11,]   1    0    1    0
## [12,]   1    0    1    0
## [13,]   1    0    0    1
## [14,]   1    0    0    1
## [15,]   1    0    0    1
## [16,]   1    0    0    1

beta=solve(t(x)%*%x)%*%t(x)%*%y
beta
```

```

##      [,1]
## [1,] 359.27278
## [2,] -68.36271
## [3,] -104.35871
## [4,]  80.50939
```

Abaixo é apresentado o calculo da ANOVA de `mod1` através de uma abordagem matricial. Portanto verifica-se que a reparametrização é o método utilizado pelas funções `aov()` e `lm()`.

```

## Anova com uma abordagem matricial
SQtrat=t(beta)%*%t(x)%*%y-((sum(y))^2)/16
SQtotal=t(y)%*%y-((sum(y))^2)/16
SQRes = SQtotal-SQtrat
Fc=(SQtrat/3)/(SQRes/12)
SQtrat

##      [,1]
## [1,] 79680.8

SQRes

##      [,1]
## [1,] 169787.4
Fc

##      [,1]
## [1,] 1.87719
```

```
anova(mod1)

## Analysis of Variance Table
##
## Response: Y
##           Df Sum Sq Mean Sq F value Pr(>F)
## factor(Tratamentos) 3 79681  26560  1.8772 0.1873
## Residuals          12 169787  14149
```

```
anova(mod2)

## Analysis of Variance Table
##
## Response: Y
##           Df Sum Sq Mean Sq F value Pr(>F)
## factor(Tratamentos) 3 79681  26560  1.8772 0.1873
## Residuals          12 169787  14149
```

Demonstrar como o procedimento de estimação e cálculo da ANOVA serviu apenas para verificar de maneira didática como as funções no *R* contornam o problema singularidade da matriz delineamento. Conforme já relatado acima, as funções `aov()` e `lm()` podem ser utilizadas para realizar a ANOVA. Porém, será dado enfase as funções do pacote *ExpDes*, cujas funções possibilitam analisar dados uni e bifatoriais, e este último com e sem parcelas subdivididas. Esse pacote tem dentro dele as principais funções que são necessárias para realizar a ANOVA (entre elas a `aov()`), o que facilita a obtenção dos resultados.

### 2.3.1 Delineamento inteiramente casualizado (DIC)

O DIC é um delineamento adequado para áreas uniformes (parcelas são uniformes), onde não há necessidade de controle local (bloqueamento). Com esse delineamento, os tratamentos devem ser distribuídos aleatoriamente nas parcelas.

O modelo do DIC é dado por

$$Y_{ij} = m + t_i + \varepsilon_{ij}$$

Onde  $m$  é a média geral do experimento,  $t_i$  é o efeito de tratamentos e  $\varepsilon_{ij}$  é o erro experimental. Experimentos em DIC serão executados através da função `crd()` do pacote *ExpDes*. Os argumentos desta função são:

Argumento	Descrição
<code>trat</code>	Objeto contendo os tratamentos
<code>resp</code>	Objeto contendo a variável resposta
<code>quali</code>	Se TRUE, o tratamento é qualitativo ( <i>default</i> )
<code>mcomp</code>	Indicar o teste complementar (Tukey é <i>default</i> )
<code>nl</code>	Indica se uma regressão deve ser ajustada (FALSE é o <i>default</i> )
<code>hvar</code>	Teste de homogeneidade da variância (Bartlett é o <i>default</i> )

Argumento	Descrição
<b>sigT</b>	Significância da comparação múltipla
<b>sigF</b>	Significância do teste F

Para maiores detalhes, ver o pdf do pacote (aqui) ou ir em ajuda (digitar `?ExpDes` no console).

### 2.3.1.1 DIC com fatores qualitativos

Quando os fatores são qualitativos, a análise complementar é realizada através de testes de médias. A função `crd()` do pacote *ExpDes* retorna a tabela da ANOVA, a análise de pressupostos (normalidade e homogeneidade) e o teste de comparação de médias.

```
require(ExpDes)
print(dados_3, digits = 3) ## Banco de dados
```

```
##   Trat Bloco    a      b      c    XPI    XPAM    XPDM    XPDA
## 1     1    1  416  6.12 0.00208 2938  2306  3571  4039
## 2     1    2  292  6.26 0.00266 2351  1857  2845  3211
## 3     1    3  426  6.47 0.00214 3020  2405  3634  4089
## 4     1    4  303  6.87 0.00228 3008  2431  3584  4012
## 5     2    1  370  7.70 0.00246 3136  2599  3672  4069
## 6     2    2  181  8.52 0.00344 2478  2095  2861  3144
## 7     2    3  322  7.48 0.00243 3076  2534  3617  4018
## 8     2    4  291  7.90 0.00278 2896  2410  3383  3744
## 9     3    1  255  7.43 0.00279 2694  2216  3172  3527
## 10    3    2  112  7.44 0.00302 2460  2025  2896  3218
## 11    3    3  207  7.62 0.00299 2546  2106  2986  3311
## 12    3    4  446  7.23 0.00235 3076  2516  3636  4050
## 13    4    1  455  7.60 0.00259 2936  2427  3445  3822
## 14    4    2  607  7.60 0.00259 2936  2427  3445  3822
## 15    4    3  224  7.73 0.00279 2772  2300  3244  3594
## 16    4    4  472  7.08 0.00233 3043  2477  3609  4028
## 17    5    1  330  6.50 0.00229 2838  2263  3413  3839
## 18    5    2  212  7.03 0.00248 2832  2302  3362  3755
## 19    5    3  228  6.95 0.00285 2440  1978  2903  3245
## 20    5    4  285  5.99 0.00217 2756  2151  3362  3810
## 21    6    1  215  9.04 0.00346 2614  2233  2995  3277
## 22    6    2  308  7.08 0.00245 2885  2348  3422  3820
## 23    6    3  305  6.71 0.00213 3150  2531  3769  4227
## 24    6    4  350 10.60 0.00349 3038  2660  3415  3694
## 25    7    1  136  8.13 0.00327 2487  2084  2890  3188
## 26    7    2  196  6.64 0.00249 2669  2140  3199  3591
## 27    7    3  528  5.05 0.00178 2828  2090  3566  4112
## 28    7    4  239  9.64 0.00353 2734  2360  3107  3384
## 29    8    1  141 10.69 0.00401 2662  2334  2990  3233
## 30    8    2  328  8.59 0.00307 2826  2377  3275  3608
```

```

## 31    8    3 477  7.94 0.00276 2878 2401 3355 3709
## 32    8    4 367  7.13 0.00243 2937 2395 3479 3881
## 33    9    1 534  5.29 0.00191 2773 2081 3465 3977
## 34    9    2 445  5.43 0.00206 2641 2000 3282 3757
## 35    9    3 609  5.11 0.00177 2892 2146 3637 4189
## 36    9    4 547  5.32 0.00191 2785 2096 3475 3985
## 37   10    1 315  5.35 0.00194 2756 2077 3435 3938
## 38   10    2 821  5.07 0.00179 2837 2101 3574 4120
## 39   10    3 608  5.90 0.00202 2912 2261 3562 4044
## 40   10    4 526  5.12 0.00188 2728 2026 3430 3950
## 41   11    1 511  6.59 0.00257 2564 2052 3077 3456
## 42   11    2 929  5.28 0.00189 2795 2098 3492 4009
## 43   11    3 462  5.05 0.00178 2828 2090 3566 4112
## 44   11    4 562  5.81 0.00215 2707 2094 3320 3775
## 45   12    1 637  5.46 0.00198 2757 2092 3422 3915
## 46   12    2 731  6.12 0.00209 2934 2302 3565 4032
## 47   12    3 508  6.41 0.00238 2689 2136 3242 3651
## 48   12    4 618  6.02 0.00212 2845 2223 3468 3929
## 49   13    1 492  5.65 0.00202 2799 2147 3452 3935
## 50   13    2 447  5.75 0.00192 3003 2315 3690 4199
## 51   13    3 320  7.18 0.00266 2698 2203 3193 3559
## 52   13    4 326  6.04 0.00229 2635 2061 3209 3634
## 53   14    1 389  6.52 0.00236 2763 2206 3321 3734
## 54   14    2 388  6.63 0.00248 2672 2141 3203 3596
## 55   14    3 422  5.64 0.00195 2888 2213 3562 4061
## 56   14    4 270  7.60 0.00275 2764 2285 3243 3598
## 57   15    1 472  5.91 0.00217 2731 2122 3339 3789
## 58   15    2 306  6.18 0.00250 2477 1949 3005 3396
## 59   15    3 485  6.33 0.00219 2886 2285 3486 3931
## 60   15    4 416  7.06 0.00262 2692 2190 3194 3567
## 61   16    1 522  7.31 0.00251 2912 2387 3437 3826
## 62   16    2 479  6.17 0.00209 2952 2322 3582 4048
## 63   16    3 739  7.47 0.00257 2910 2397 3423 3803
## 64   16    4 478  6.34 0.00231 2744 2173 3314 3736

trat=as.matrix(dados_3$Trat)
resp=as.matrix(dados_3$a)
mod3=crd(trat,resp,sigF = 0.05)

```

```

## -----
## Analysis of Variance Table
## -----
##          DF   SS   MS   Fc   Pr>Fc
## Treatment 15 1009319 67288 4.3305 5.1207e-05
## Residuals 48 745837 15538
## Total     63 1755156
## -----

```

```

## CV = 30.29 %
##
## -----
## Shapiro-Wilk normality test
## p-value: 0.1899147
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered
## -----
## -----
## Homogeneity of variances test
## p-value: 0.3466402
## According to the test of bartlett at 5% of significance, residuals can be considered
## -----
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      12      623.5205
## a      11      615.8117
## ab     10      567.6311
## ab     16      554.2897
## ab     9       533.6724
## ab     4       439.7822
## ab     15      419.6284
## ab     13      396.133
## ab     14      367.076
## ab     1       359.2728
## ab     8       328.184
## b      6       294.6677
## b      2       290.9101
## b      7       274.764
## b      5       263.8872
## b      3       254.9141
## -----

```

É possível extrair os erros através de `mod3$residuos`. Também é possível fazer diagnóstico dos pressupostos do modelo estatístico através de gráficos utilizando a função `plotres()`:

```
plotres(mod3)
```

O *p*-valor do teste de Shapiro-Wilk mostrou que os resíduos seguem uma distribuição, o que pode ser confirmado pelo QQ-Plot. Além disso, a dispersão dos resíduos indica que eles são homogêneos. Para escolher qual teste utilizar para verificar a homocedasticidade dos erros, basta indicar a opção `levene` no argumento `hvar`. Lembrando que o *default* da função é o teste de Bartlett.

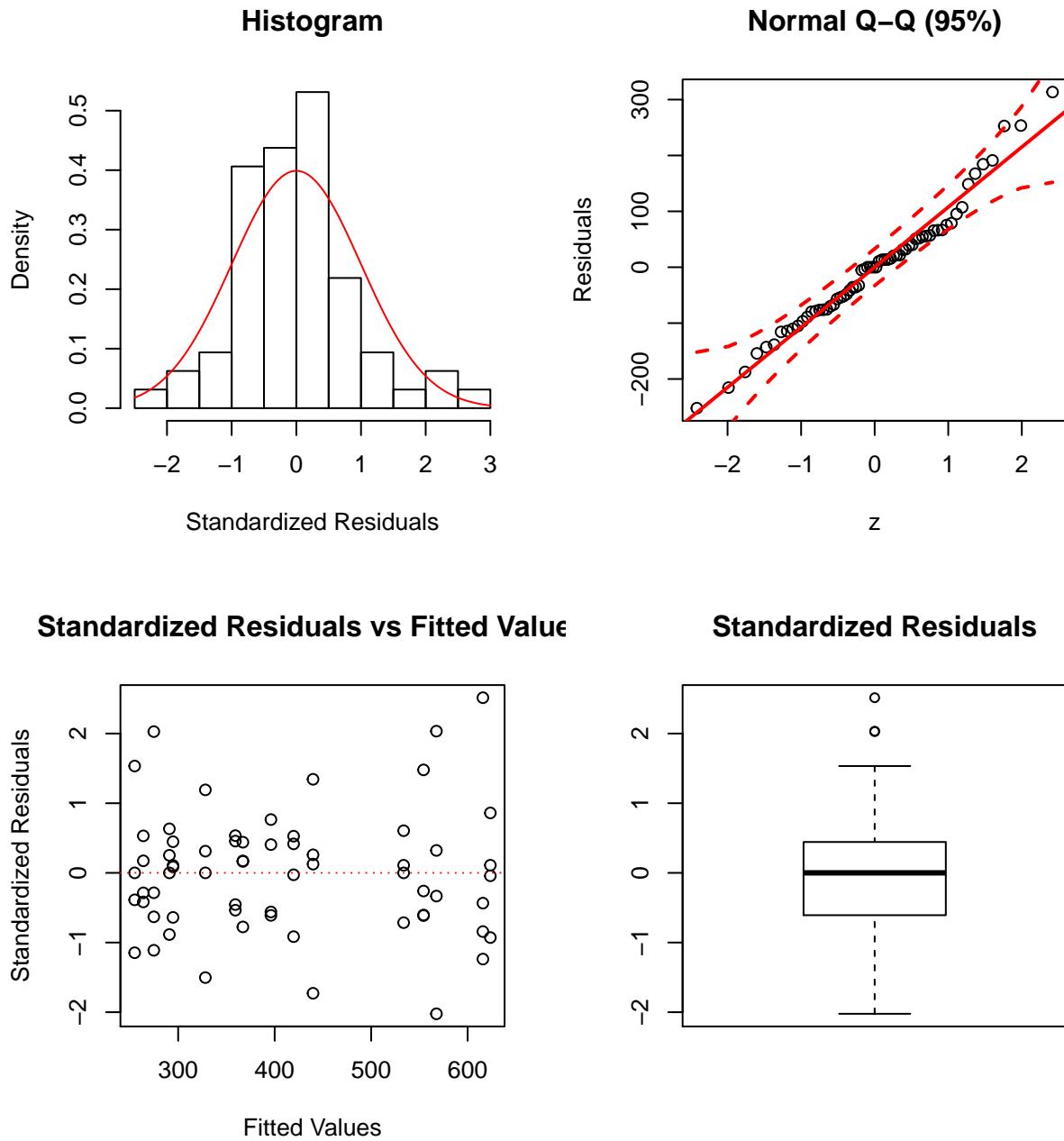


Figure 26: Gráfico de resíduos gerado pela função `plotres()`

```

mod3.1=crd(trat,resp,sigF = 0.05,mcomp = "sk", hvar = "levene") ## Bartlett

## -----
## Analysis of Variance Table
## -----
##          DF      SS      MS      Fc     Pr>Fc
## Treatment 15 1009319 67288 4.3305 5.1207e-05
## Residuals 48 745837 15538
## Total     63 1755156
## -----
## CV = 30.29 %
## 
## -----
## Shapiro-Wilk normality test
## p-value: 0.1899147
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered
## homocedastic!
## -----
## 
## -----
## Homogeneity of variances test
## p-value: 0.0001074615
## WARNING: at 5% of significance, residuals can not be considered homocedastic!
## -----
## 
## Scott-Knott test
## -----
##    Groups Treatments     Means
## 1       a           12 623.5205
## 2       a           11 615.8117
## 3       a           10 567.6311
## 4       a           16 554.2897
## 5       a            9 533.6724
## 6       b            4 439.7822
## 7       b           15 419.6284
## 8       b           13 396.1330
## 9       b           14 367.0760
## 10      b            1 359.2728
## 11      b            8 328.1840
## 12      b            6 294.6677
## 13      b            2 290.9101
## 14      b            7 274.7640
## 15      b            5 263.8872
## 16      b            3 254.9141
## -----

```

## Comparação múltipla

Em relação a comparação múltipla, o pacote utiliza o teste de Tukey como padrão para comparar médias. Para utilizar o teste de Scott-Knott para comparar as médias, basta indicar a opção `sk` no argumento `mcomp`.

```
mod3.2=crd(trat,resp,sigF = 0.05,mcomp = "sk") ## Scott-Knott

## -----
## Analysis of Variance Table
## -----
##          DF      SS      MS      Fc     Pr>Fc
## Treatment 15 1009319 67288 4.3305 5.1207e-05
## Residuals 48 745837 15538
## Total     63 1755156
## -----
## CV = 30.29 %
## 
## -----
## Shapiro-Wilk normality test
## p-value: 0.1899147
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered
## 
## -----
## Homogeneity of variances test
## p-value: 0.3466402
## According to the test of bartlett at 5% of significance, residuals can be considered
## 
## -----
## Scott-Knott test
## 
## -----
##   Groups Treatments    Means
## 1       a           12 623.5205
## 2       a           11 615.8117
## 3       a           10 567.6311
## 4       a           16 554.2897
## 5       a            9 533.6724
## 6       b            4 439.7822
## 7       b           15 419.6284
## 8       b           13 396.1330
## 9       b           14 367.0760
## 10      b            1 359.2728
## 11      b            8 328.1840
## 12      b            6 294.6677
## 13      b            2 290.9101
## 14      b            7 274.7640
## 15      b            5 263.8872
## 16      b            3 254.9141
```

```
## -----
```

## Gráfico de resultados: uma proposta

```
medias_mod3.2=mod3.2$means ## Armazenando as médias
medias_mod3.2=medias_mod3.2[rev(order(medias_mod3.2$resp)),] ## Ordenando o banco de dados

grupos=c(rep("a",5),rep("b",11)) ## Armazenando letras

medias_mod3.2=data.frame(
  "trat"=factor(medias_mod3.2[,1]),
  "resp"=c(medias_mod3.2[,2]),
  "grupos"=grupos
)

ggplot(data=medias_mod3.2, aes(y = resp, x = trat)) + ## indica daods e variáveis
geom_bar(aes(fill = factor(trat)), stat="identity", position ="dodge") +
expand_limits(y=c(0,800))+ ## aumenta os limites do eixo y
geom_text(
  aes(label=grupos), ## indica letras em cima dos gráficos
  position = position_dodge(0.8),
  vjust = -0.3, size = 3.5)+
guides(fill = FALSE) ## Remove legenda
```

### 2.3.1.2 DIC com fatores quantitativos

Vimos anteriormente que os fatores qualitativos são comparados através de comparações de médias. No caso de fatores quantitativos, o comum é utilizar regressões como análise complementar. Neste tipo de análise a  $SQ_{Trat}$  é decomposta, e cada polinômio explicará parte desta soma de quadrados. O maior grau significativo do polinômio determinará qual a regressão escolhida. Para implementar essa análise, utilizando a função `crd()`, basta indicar como `FALSE` os argumentos `quali` e `nl`. Os dados utilizados nesta seção e na seção de regressão podem ser baixados aqui.

```
require(readxl)
Reg=read_excel("D:/Desktop/final/Reg.xls")
crd(Reg$DOSE,Reg$Y, quali=FALSE,nl=FALSE,sigT=0.05,sigF = 0.05)

## -----
## Analysis of Variance Table
## -----
##          DF      SS      MS      Fc      Pr>Fc
## Treatment  6 11803.9 1967.32 25.805 8.3498e-07
## Residuals 14 1067.3   76.24
## Total     20 12871.2
## -----
## CV = 11.56 %
##
```

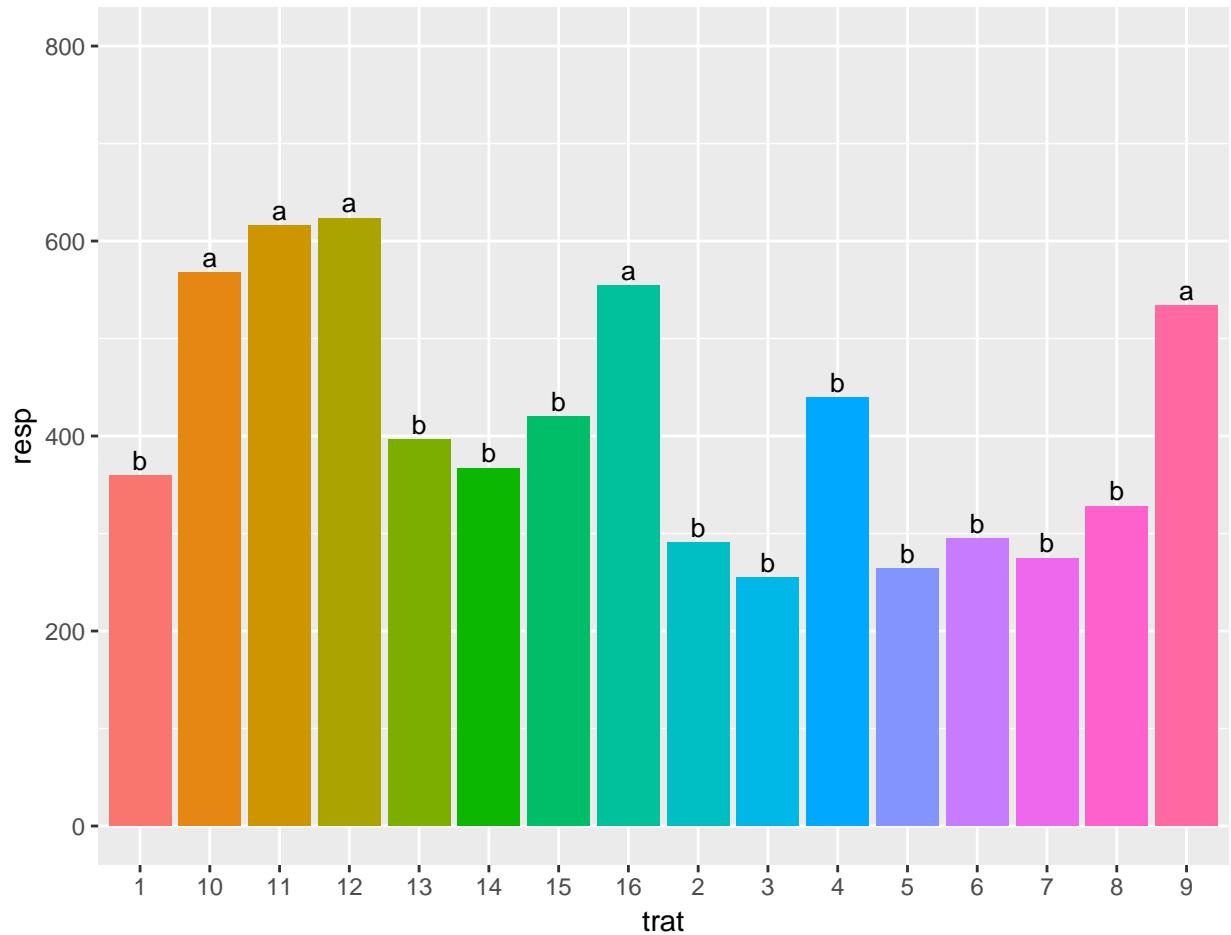


Figure 27: Gráfico de barras gerado pela função ggplot()

```

## -----
## Shapiro-Wilk normality test
## p-value: 0.7855485
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered
## -----
## 
## -----
## Homogeneity of variances test
## p-value: 0.4249146
## According to the test of bartlett at 5% of significance, residuals can be considered
## -----
## 
## Adjustment of polynomial models of regression
## -----
## 
## Linear Model
## =====
##   Estimate Standard.Error   tc   p.value
## -----
##   b0 44.5595      3.4349    12.9725    0
##   b1  2.0643      0.1905    10.8341    0
## -----
## 
## R2 of linear model
## -----
## 0.758112
## -----
## 
## Analysis of Variance of linear model
## =====
##          DF     SS       MS      Fc   p.value
## -----
##  Linear Effect 1 8,948.6790 8,948.6790 117.38    0
##  Lack of fit   5 2,855.2260 571.0452   7.49  0.00131
##  Residuals     14 1,067.3330  76.2381
## -----
## 
## 
## Quadratic Model
## =====
##   Estimate Standard.Error   tc   p.value
## -----
##   b0 28.3095      4.4002    6.4336  0.00002
##   b1  5.9643      0.6870    8.6818    0
##   b2 -0.1300      0.0220   -5.9088 0.00004
## -----

```

```

## 
## R2 of quadratic model
## -----
## 0.983609
## -----
## 
## Analysis of Variance of quadratic model
## =====
##          DF      SS       MS      Fc    p.value
## -----
## Linear Effect  1 8,948.6790 8,948.6790 117.38     0
## Quadratic Effect 1 2,661.7500 2,661.7500 34.91   4e-05
## Lack of fit    4 193.4762   48.3691    0.63  0.64625
## Residuals      14 1,067.3330   76.2381
## -----
## -----
## 
## Cubic Model
## =====
##      Estimate Standard.Error    tc    p.value
## -----
## b0 30.4206      4.8577     6.2623  0.00002
## b1 4.5569       1.5344     2.9698  0.0101
## b2 -0.0033      0.1254    -0.0266 0.9792
## b3 -0.0028      0.0027    -1.0258 0.3224
## -----
## 
## R2 of cubic model
## -----
## 0.990405
## -----
## 
## Analysis of Variance of cubic model
## =====
##          DF      SS       MS      Fc    p.value
## -----
## Linear Effect  1 8,948.6790 8,948.6790 117.38     0
## Quadratic Effect 1 2,661.7500 2,661.7500 34.91   4e-05
## Cubic Effect    1  80.2222   80.2222    1.05  0.32239
## Lack of fit    3 113.2540   37.7513    0.5   0.69143
## Residuals      14 1,067.3330   76.2381
## -----
## -----

```

O polinômio de segunda grau deve ser o escolhido, pois ele foi o maior grau significativo ( $p$ -valor menor que 0,05). Os desvios da regressão nada mais são do que a *FALTA DE AJUSTE* dos modelos. Porém, como é comum em ciências agrárias, ajusta-se apenas polinômios até a

terceira ordem, devido a dificuldade de interpretação de polinômios com ordens superiores. É importante ressaltar que aqui estamos falando de **regressões polinomiais**. Como veremos mais pra frente, a falta de ajuste é inaceitável para modelos de regressão múltipla. O valor de  $R^2$  é dado pela razão entre a  $SQ_{\text{Regressão}}$  e a  $SQ_{\text{Tratamento}}$ . No exemplo acima, o  $R^2$  da regressão quadrática (selecionada) é:

$$R^2 = \frac{SQ_{\text{Regressão}}}{SQ_{\text{Tratamento}}} = \frac{8948,679 + 2661,7500}{11803,9} = 0,9836$$

Percebe-se claramente pelos resultados acima que a significância do grau do polinômio está diretamente relacionado com quanto ele “contribui” para explicar a  $SQ_{\text{Trat}}$ . Abaixo vamos reproduzir o exemplo para os usuários das funções `aov()` ou `lm()`. Este exemplo é apenas ilustrativo, por isso não entraremos em maiores detalhes.

```
Reg=read_excel("D:/Desktop/final/Reg.xls")
Total=aov(Y~1,data=Reg)
Saturado=aov(Y~factor(DOSE),data=Reg)
Linear=lm(Y~DOSE,data=Reg)
Quadratica=lm(Y~DOSE+I(DOSE^2),data=Reg)
Cubica=lm(Y~DOSE+I(DOSE^2)+I(DOSE^3),data=Reg)
anova(Total,Linear,Quadratica,Cubica,Saturado)

## Analysis of Variance Table
##
## Model 1: Y ~ 1
## Model 2: Y ~ DOSE
## Model 3: Y ~ DOSE + I(DOSE^2)
## Model 4: Y ~ DOSE + I(DOSE^2) + I(DOSE^3)
## Model 5: Y ~ factor(DOSE)
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1     20 12871.2
## 2     19 3922.6  1    8948.7 117.3780 3.433e-08 ***
## 3     18 1260.8  1    2661.8  34.9136 3.810e-05 ***
## 4     17 1180.6  1      80.2   1.0523    0.3224
## 5     14 1067.3  3     113.3   0.4952    0.6914
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

A forma mais prática de analisar uma regressão é através de gráficos. A função `crd()` fornece essa alternativa, através da função `graficos()`. Nesta função existe dois argumentos fundamentais: `a`, onde indicamos o objeto que contém a saída da análise do experimento e `grau`, onde indicamos o grau do polinômio.

```
require(ExpDes)
require(readxl)
Reg=read_excel("D:/Desktop/final/Reg.xls")
crd_Reg=crd(Reg$DOSE,Reg$Y, quali=FALSE,nl=FALSE,sigT=0.05,sigF = 0.05)

## -----
```

```

## Analysis of Variance Table
## -----
##          DF      SS      MS      Fc     Pr>Fc
## Treatment 6 11803.9 1967.32 25.805 8.3498e-07
## Residuals 14 1067.3   76.24
## Total     20 12871.2
## -----
## CV = 11.56 %
## 
## -----
## Shapiro-Wilk normality test
## p-value: 0.7855485
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered
## -----
## 
## -----
## Homogeneity of variances test
## p-value: 0.4249146
## According to the test of bartlett at 5% of significance, residuals can be considered
## -----
## 
## Adjustment of polynomial models of regression
## -----
## 
## Linear Model
## =====
##          Estimate Standard.Error   tc    p.value
## -----
## b0 44.5595       3.4349    12.9725    0
## b1  2.0643       0.1905    10.8341    0
## -----
## 
## R2 of linear model
## -----
## 0.758112
## -----
## 
## Analysis of Variance of linear model
## =====
##          DF      SS      MS      Fc     p.value
## -----
## Linear Effect 1 8,948.6790 8,948.6790 117.38    0
## Lack of fit   5 2,855.2260 571.0452   7.49  0.00131
## Residuals     14 1,067.3330 76.2381
## -----
## 
## -----

```

```

## 
## Quadratic Model
## =====
##   Estimate Standard.Error   tc   p.value
## -----
## b0 28.3095      4.4002    6.4336  0.00002
## b1  5.9643      0.6870    8.6818    0
## b2 -0.1300      0.0220   -5.9088  0.00004
## -----
## 
## R2 of quadratic model
## -----
## 0.983609
## -----
## 
## Analysis of Variance of quadratic model
## =====
##           DF     SS      MS      Fc   p.value
## -----
## Linear Effect  1 8,948.6790 8,948.6790 117.38    0
## Quadratic Effect 1 2,661.7500 2,661.7500 34.91 4e-05
## Lack of fit    4 193.4762   48.3691    0.63  0.64625
## Residuals       14 1,067.3330  76.2381
## -----
## 
## 
## Cubic Model
## =====
##   Estimate Standard.Error   tc   p.value
## -----
## b0 30.4206      4.8577    6.2623  0.00002
## b1  4.5569      1.5344    2.9698  0.0101
## b2 -0.0033      0.1254   -0.0266  0.9792
## b3 -0.0028      0.0027   -1.0258  0.3224
## -----
## 
## R2 of cubic model
## -----
## 0.990405
## -----
## 
## Analysis of Variance of cubic model
## =====
##           DF     SS      MS      Fc   p.value
## -----
## Linear Effect  1 8,948.6790 8,948.6790 117.38    0

```

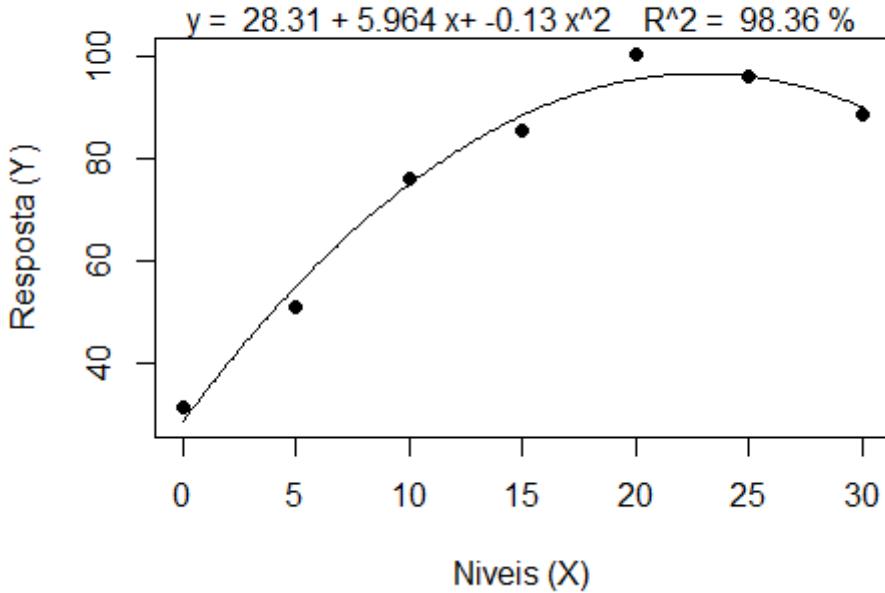


Figure 28: Regressão quadrática obtida no delineamento DIC

```
## Quadratic Effect 1 2,661.7500 2,661.7500 34.91 4e-05
## Cubic Effect      1    80.2222     80.2222    1.05 0.32239
## Lack of fit       3   113.2540    37.7513    0.5  0.69143
## Residuals         14 1,067.3330   76.2381
## -----
## -----
## Armazenando o objeto que contém a saída
graphics(crd_Reg, degree = 2)
```

### Gráfico de resultados: uma proposta

Utilizando a função `ggplot2` é possível fazer gráficos mais elegantes. Abaixo, vamos fazer o mesmo gráfico porém utilizando a função `ggplot()`. Neste gráfico, vamos traçar o intervalo de confiança dos valores preditos:

```
require(ggplot2)
require(dplyr)
g2=ggplot(data=Reg, aes(x=DOS, y=Y))+ ## Indicar dados e variáveis
  geom_point(color="darkgreen",size=2) ## Adiciona e edita os pontos

g2+geom_smooth(method = "lm", formula = y~poly(x,2),color="black",fill="grey")+
  ## Intervalo de confiança
  theme_bw() ## fundo branco
```

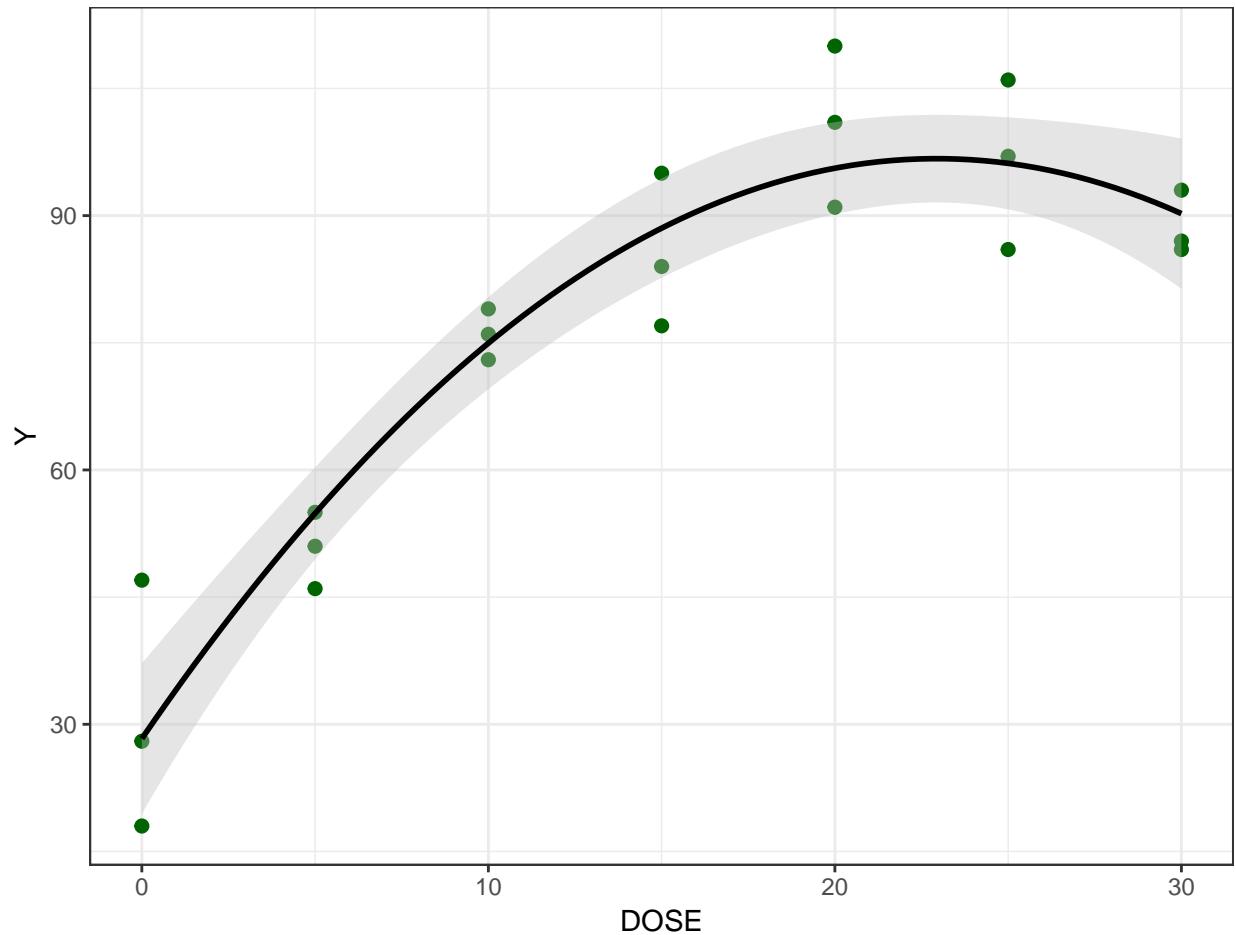


Figure 29: Gráfico de linhas gerado pela função ggplot()

Este é um exemplo simples de uso da função `ggplot()`. Porém, como visto até aqui, esta função (mesmo em aplicações simples) tem suas complexidades. Pensando nisso, o pacote *Curso R* foi desenvolvido para facilitar a vida dos usuários do curso. A partir dos experimentos fatoriais, apenas as funções deste pacote serão utilizados.

### 2.3.2 Delineamento blocos ao acaso (DBC)

No delineamento blocos ao acaso existe uma restrição (fonte de variação) na sua área experimental, e por isso as parcelas não são homogêneas entre si. As parcelas dos blocos devem ser homogêneas, porém os bloco devem ser heterogêneos entre eles. O bloqueamento tem como objetivo reduzir o erro experimental, “transferindo” parte do erro experimental para efeito de bloco. O modelo do DBC é dado por

$$Y_{ij} = m + b_j + t_i + \epsilon_{ij}$$

Onde  $m$  é a média geral do experimento,  $b_j$  é o efeito de bloco,  $t_i$  é o efeito de tratamentos e  $\epsilon_{ij}$  é o erro experimental. No pacote *ExpDes*, este delineamento é executado pela função `rbd()`. Os argumentos desta função são:

Argumento	Descrição
<code>trat</code>	Objeto contendo os tratamentos
<code>bloco</code>	Objeto contendo os blocos
<code>resp</code>	Objeto contendo a variável resposta
<code>quali</code>	Se TRUE, o tratamento é qualitativo ( <i>default</i> )
<code>mcomp</code>	Indicar o teste complementar (Tukey é <i>default</i> )
<code>nl</code>	Indica se uma regressão deve ser ajustada (FALSE é o <i>default</i> )
<code>hvar</code>	Teste de homogeneidade da variância (ONeill e Mathews é o <i>default</i> )
<code>sigT</code>	Significância da comparação múltipla
<code>sigF</code>	Significância do teste F

Percebe-se que apenas um argumento foi adicionado a função: `bloco`.

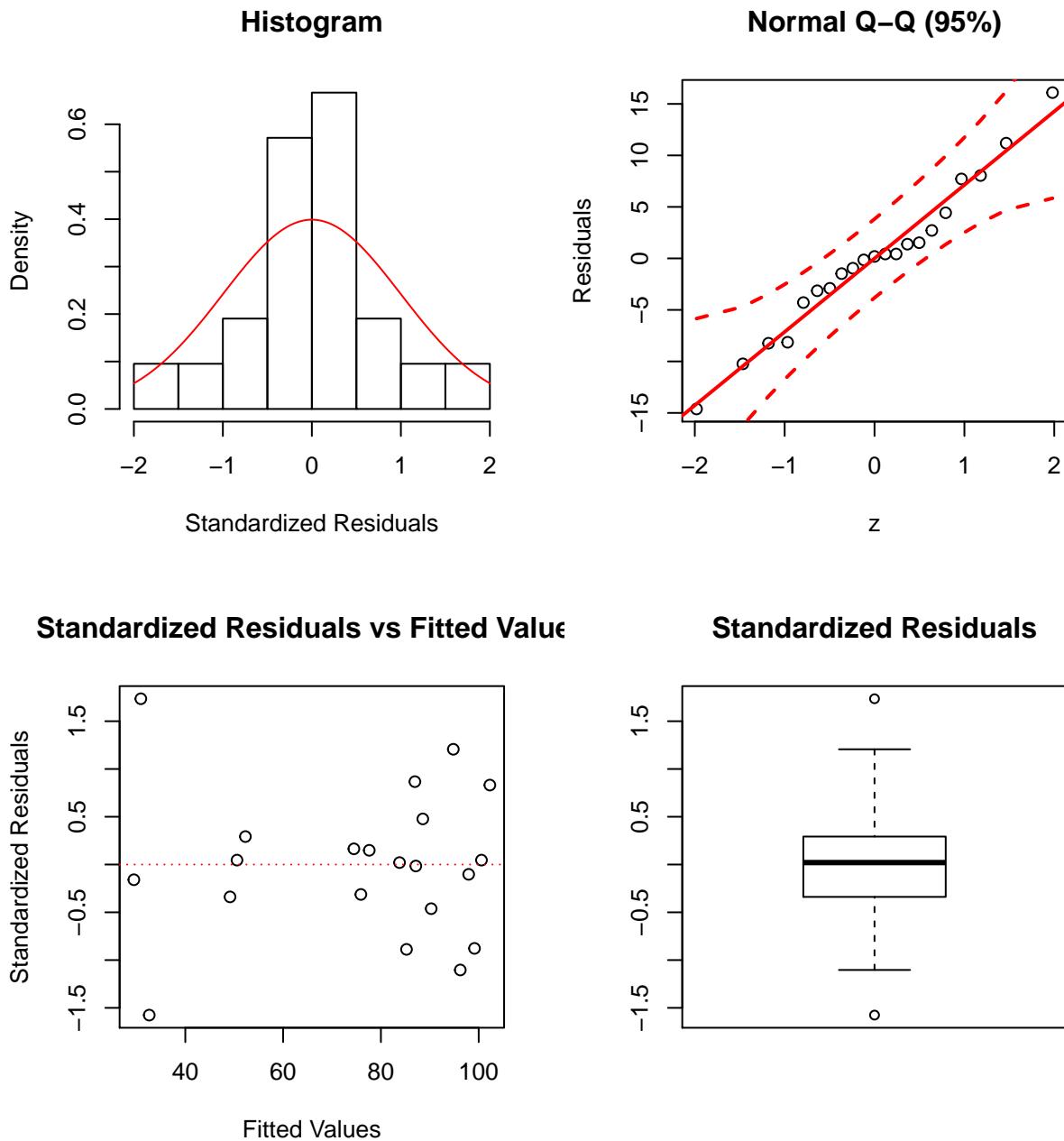
#### 2.3.2.1 DBC com fatores qualitativos

Vamos realizar a ANOVA e fazer uma análise gráfica dos resíduos :

```
mod4=rbd(Reg$TRAT,Reg$BLOCO,Reg$Y,quali=TRUE,mcomp="tukey",sigF = 0.05)
```

```
## -----
## Analysis of Variance Table
## -----
##          DF      SS      MS      Fc   Pr>Fc
## Treatment 6 11803.9 1967.32 22.8610 0.00001
## Block      2    34.7   17.33  0.2014 0.82028
## Residuals 12 1032.7   86.06
```

```
## Total      20 12871.2
## -----
## CV = 12.28 %
##
##
## -----
## Shapiro-Wilk normality test
## p-value: 0.9025366
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be cons
## -----
## 
## -----
## Homogeneity of variances test
## p-value: 0.6303247
## According to the test of oneillmathews at 5% of significance, the variances can be co
## -----
## 
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      5   100.6667
## a      6   96.33333
## a      7   88.66667
## a      4   85.33333
## ab     3   76
## bc     2   50.66667
## c      1   31
## -----
plotres(mod4)
```



Apesar do gráfico mostrar a tendência da distribuição ser leptocurtica, o *p*-valor do teste de Shapiro-Wilk mostrou que os resíduos seguem uma distribuição. Além disso, a dispersão dos resíduos indica que eles são homogêneos. A comparação das médias e a forma de apresentação dos resultados não mudam em relação ao delineamento inteiramente casualizados.

### 2.3.2.2 DBC com fatores quantitativos

Para realizar a análise de regressão utilizando a função `rbd()` basta indicar como `FALSE` os argumentos `quali` e `nl`.

```

require(readxl)
Reg=read_excel("D:/Desktop/final/Reg.xls")
rbd(Reg$DOSE,Reg$BLOCO,Reg$Y, quali=FALSE,nl=FALSE,sigT=0.05,sigF = 0.05)

## -----
## Analysis of Variance Table
## -----
##          DF      SS      MS      Fc    Pr>Fc
## Treatment 6 11803.9 1967.32 22.8610 0.00001
## Block     2   34.7   17.33  0.2014 0.82028
## Residuals 12 1032.7   86.06
## Total     20 12871.2
## -----
## CV = 12.28 %
## 
## -----
## Shapiro-Wilk normality test
## p-value: 0.9025366
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered
## -----
## 
## -----
## Homogeneity of variances test
## p-value: 0.6303247
## According to the test of oneillmathews at 5% of significance, the variances can be considered
## -----
## 
## Adjustment of polynomial models of regression
## -----
## 
## Linear Model
## =====
##      Estimate Standard.Error   tc    p.value
## -----
## b0 44.5595      3.6494    12.2101    0
## b1  2.0643      0.2024    10.1974    0
## -----
## 
## 
## R2 of linear model
## -----
## 0.758112
## -----
## 
## Analysis of Variance of linear model
## =====
##          DF      SS      MS      Fc    p.value

```

```

## -----
## Linear Effect 1 8,948.6790 8,948.6790 103.99 0
## Lack of fit 5 2,855.2260 571.0452 6.64 0.0035
## Residuals 12 1,032.6670 86.0556
## -----
## -----
## 
## Quadratic Model
## =====
##   Estimate Standard.Error tc p.value
## -----
## b0 28.3095      4.6750    6.0556 0.0001
## b1  5.9643      0.7299    8.1716 0
## b2 -0.1300      0.0234   -5.5615 0.0001
## -----
## 
## R2 of quadratic model
## -----
## 0.983609
## -----
## 
## Analysis of Variance of quadratic model
## =====
##           DF SS MS Fc p.value
## -----
## Linear Effect 1 8,948.6790 8,948.6790 103.99 0
## Quadratic Effect 1 2,661.7500 2,661.7500 30.93 0.00012
## Lack of fit 4 193.4762 48.3691 0.56 0.69475
## Residuals 12 1,032.6670 86.0556
## -----
## 
## 
## Cubic Model
## =====
##   Estimate Standard.Error tc p.value
## -----
## b0 30.4206      5.1610    5.8943 0.0001
## b1  4.5569      1.6302    2.7953 0.0162
## b2 -0.0033      0.1333   -0.0250 0.9804
## b3 -0.0028      0.0029   -0.9655 0.3533
## -----
## 
## R2 of cubic model
## -----
## 0.990405
## -----

```

```

## 
## Analysis of Variance of cubic model
## =====
##          DF      SS       MS      Fc   p.value
## -----
## Linear Effect  1 8,948.6790 8,948.6790 103.99    0
## Quadratic Effect 1 2,661.7500 2,661.7500 30.93  0.00012
## Cubic Effect    1  80.2222   80.2222    0.93  0.35334
## Lack of fit     3 113.2540   37.7513    0.44  0.72946
## Residuals        12 1,032.6670  86.0556
## -----
## -----
## -----

```

### 2.3.3 Transformação de dados

Em todos os exemplos apresentados até aqui, os resíduos devem cumprir os seguintes pressupostos: normalidade, homocedasticidade e independência:

$$\varepsilon \sim N(0, I\sigma^2)$$

Esses pressupostos são necessários para que o teste F seja utilizado na análise de variância. Sob normalidade dos resíduos e hipótese nula  $H_0$ , a razão entre as somas de quadrado de tratamento e resíduo tem distribuição F (Rencher and Schaalje 2008). Já em condições de não normalidade dos resíduos, o poder do teste (probabilidade de rejeitar  $H_0$ ) é reduzido. Apesar disso, não há grandes mudanças no erro tipo I quando a pressuposição de normalidade é violada (Senoglu and Tiku 2001), e por isso ele é considerado robusto.

Apesar do teste F de ser robusto a desvios da normalidade, é comum que ela seja cumprida para que o teste seja aplicado. Quando as pressuposições não são cumpridas, um dos procedimentos mais comum é transformar os dados. A transformação Box-Cox (Box and Cox 1964) é uma das mais comuns. Ela consiste em transformar os valores de  $Y_i$  por  $Y_i(\lambda)$ , sendo o valor de  $\lambda$  estimado por máxima verossimilhança. Após a transformação de  $Y_i$  por  $Y_i(\lambda)$  os dados seguem distribuição normal com variância constante.

A função `boxcox()`, do pacote *MASS*, pode ser utilizada para estimar o valor de  $\lambda$ . Uma sequência de valores de  $\lambda$  são estimados, e o escolhido é aquele que maximiza a função de log-verossimilhança:

```
mod5=rbd(dados_3$Trat,dados_3$Bloco, dados_3$b,sigF = 0.05)
```

```

## -----
## Analysis of Variance Table
## -----
##          DF      SS       MS      Fc   Pr>Fc
## Treatment 15 60.455 4.0303 5.0617 0.00001
## Block      3  2.359  0.7862 0.9874 0.40726
## Residuals 45 35.831  0.7962

```

```

## Total      63 98.644
## -----
## CV = 13.16 %
##
## -----
## Shapiro-Wilk normality test
## p-value: 0.02285176
## WARNING: at 5% of significance, residuals can not be considered normal!
## -----
## 
## -----
## Homogeneity of variances test
## p-value: 0.02448204
## WARNING: at 5% of significance, residuals can not be considered homocedastic!
## -----
## 
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a     8    8.587804
## ab    6    8.356202
## abc   2    7.902417
## abcd  4    7.502882
## abcd  3    7.432862
## abcd  7    7.364687
## abcd  16   6.821353
## abcd  5    6.620139
## abcd  14   6.597912
## abcd  1    6.430431
## abcd  15   6.370027
## bcd   13   6.156438
## cd    12   6.000802
## cd    11   5.681959
## d     10   5.357343
## d     9    5.285228
## -----

```

Os pressupostos de normalidade e homocedasticidade foram violados neste exemplo. O próximo passo é encontrar o valor de  $\lambda$  para transformar as variáveis, utilizando para isso a função `boxcox()`.

```

require(MASS)
boxcox(b~Trat,data=dados_3,
       lambda=seq(-5,2,length=20)) ## Indica os valores de lambda

```

Percebe-se que o intervalo de  $\lambda$  não cruza pelo valor zero, indicando que a transformação  $\log$  não é a mais adequada. Vamos focar no valor que maximiza a função de log-verossimilhança, delimitando o valor de  $\lambda$  através do argumento `lambda`.

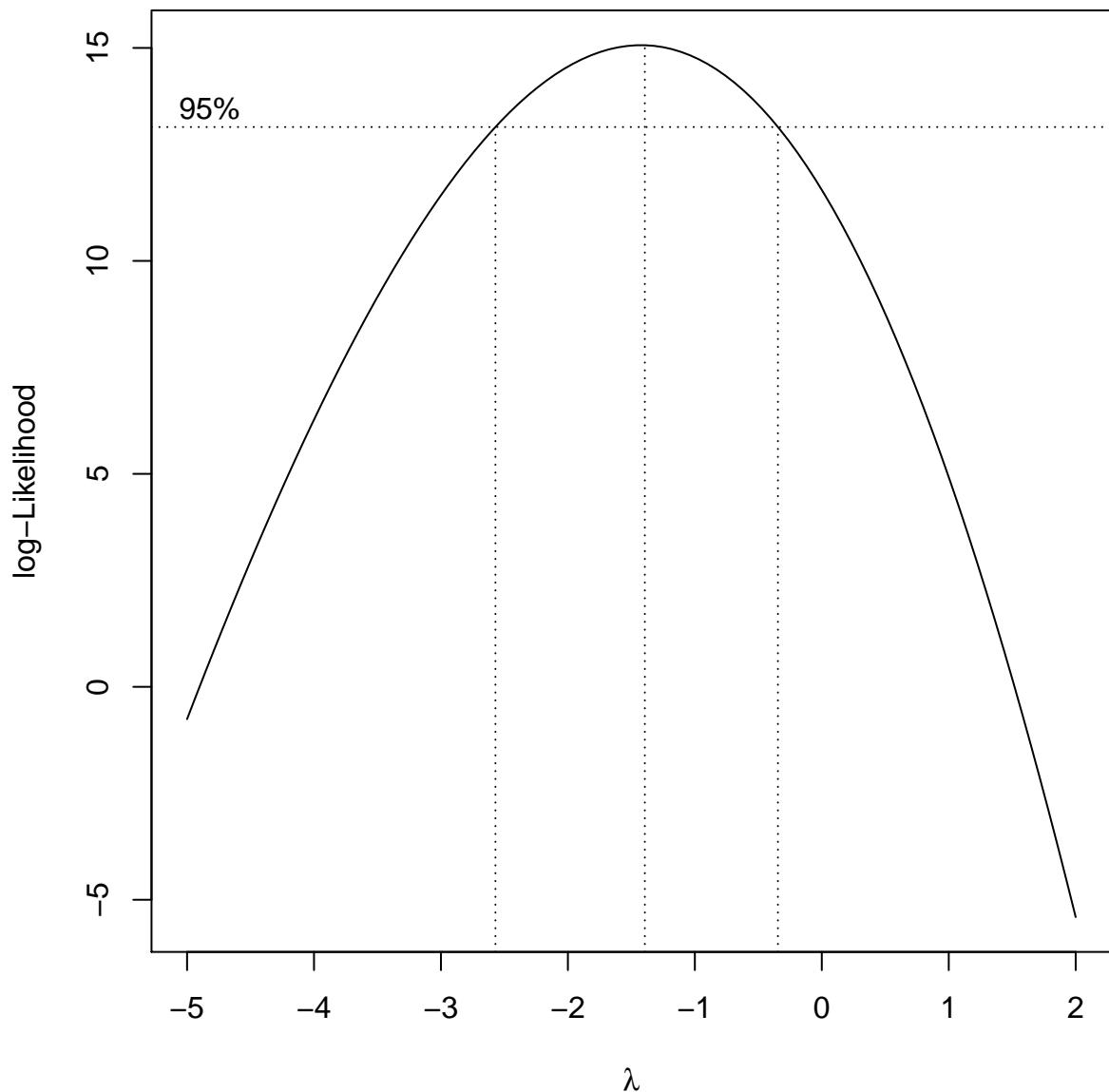
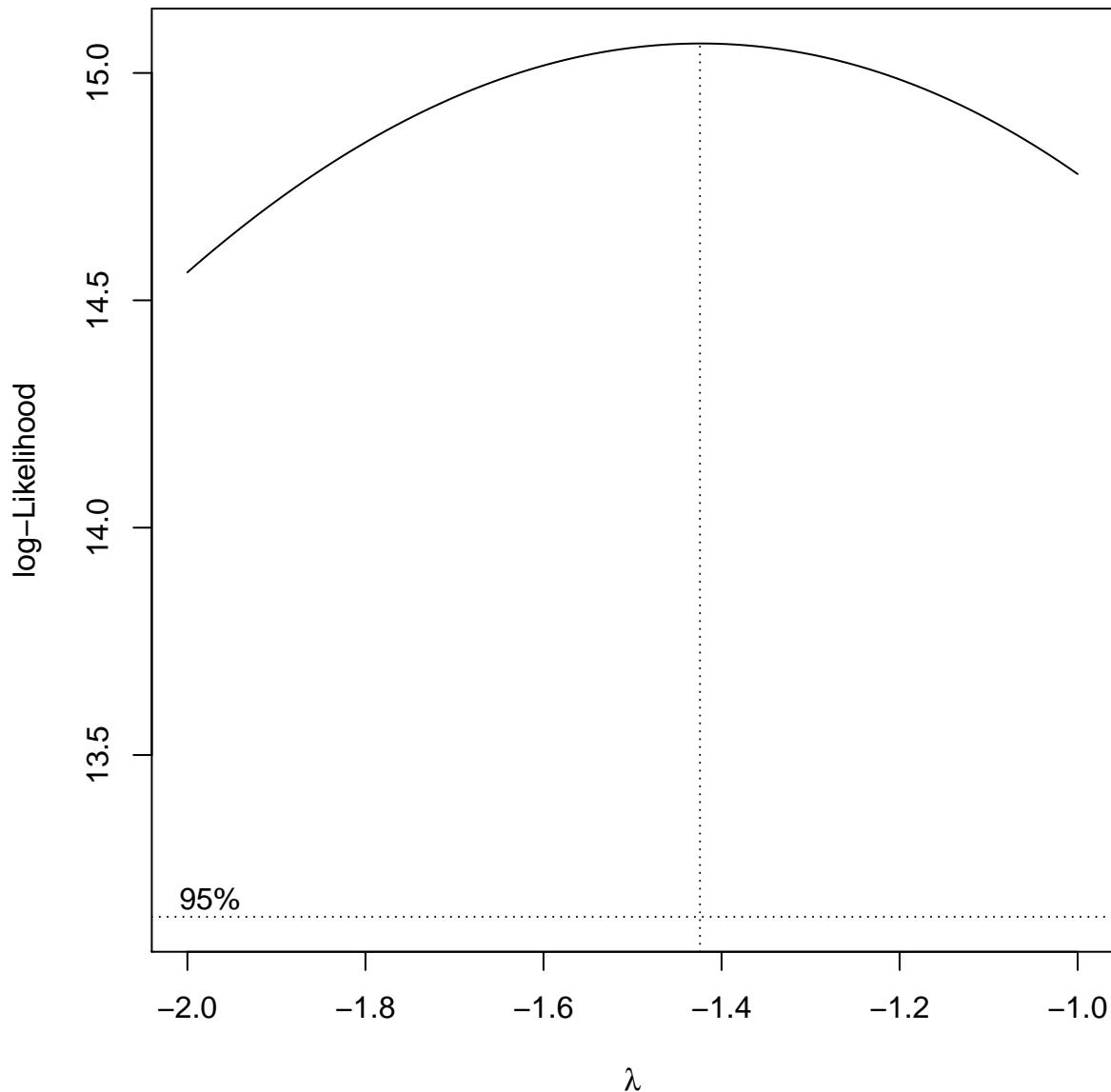


Figure 30: Gráfico gerado pela função boxcox() para identificar o valor de lambda

```

require(MASS)
boxcox(b~Trat,data=dados_3,
       lambda=seq(-2,-1,length=20)) ## "Aproximando" os valores de lambda

```



Uma forma mais prática de encontrar o valor de  $\lambda$  que maximiza a função de log-verossimilhança é utilizar a função `locator()`. Após executar a função abaixo, basta clicar com o cursor sobre o ponto que maximiza a função para que as coordenadas sejam mostradas no console.

```
locator(n=1)
```

O valor de  $\lambda$  que maximiza a função de log-verossimilhança é aproximadamente -1,4. Então, elevamos o valor da variável resposta por  $\lambda = -1,4$ :

```
mod5.1=rbd(dados_3$Trat,dados_3$Bloco, dados_3$b^-1.4,sigF = 0.05)
```

```
## -----
## Analysis of Variance Table
## -----
##          DF      SS       MS      Fc    Pr>Fc
## Treatment 15 0.0121282 0.00080855 6.3362 0.00000
## Block      3 0.0002768 0.00009228 0.7231 0.54348
## Residuals  45 0.0057424 0.00012761
## Total      63 0.0181474
## -----
## CV = 15.65 %
## 
## -----
## Shapiro-Wilk normality test
## p-value: 0.1670394
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be cons
## -----
## 
## -----
## Homogeneity of variances test
## p-value: 0.2008088
## According to the test of oneillmathews at 5% of significance, the variances can be co
## -----
## 
## Tukey's test
## -----
## Groups Treatments Means
## a     9   0.09728835
## ab    10   0.09595175
## ab    11   0.08938781
## abc   12   0.08184798
## abcd   13   0.07968353
## abcd   15   0.0753837
## abcd   1   0.07410317
## abcd   14   0.07260402
## abcd   5    0.07140962
## abcd   16   0.06882471
## bcd    7    0.06735454
## cd    3    0.06034415
## cd    4    0.05963573
## cd    2    0.05556879
```

```

##   cd      6  0.05419642
##   d      8  0.05109492
## -----

```

Os pressupostos de normalidade e homocedasticidade foram cumpridos após a transformação. Ressalta-se, porém, que os valores das médias que devem ser apresentados como resultado final (em tabelas ou gráficos) são os originais, e não os transformados.

## 2.4 Experimentos bifatoriais

Experimentos fatoriais são muito comuns nas ciências agrárias, pois permitem o estudo de dois ou mais fatores em um mesmo experimento. Diversas são as vantagens em se conduzir um experimento deste tipo. Dentre elas, podemos citar a redução de custos, quando comparado à realizar um experimento para cada fator, a otimização da área experimental e dos tratos culturais, bem como a possibilidade de identificar o efeito de dois ou mais fatores sobre a magnitude da variável resposta. Esta é, talvez, a principal vantagem destes experimentos. Ao memo tempo, no entanto, é a fonte de um dos maiores desafios encontrados no meio acadêmico. O surgimento de uma terceira fonte de variação, conhecida por interação.

Vamos considerar como exemplo, um experimento que avaliou a influencia de dois fatores, digamos  $\alpha$  e  $\tau$ , em uma determinada variável resposta. O modelo estatístico considerado neste tipo de experimento é:  $y_{ijk} = \mu + \beta_k + \alpha_i + \tau_j + (\alpha\tau)_{ij} + \varepsilon_{ijk}$ , onde  $y_{ijk}$  é o valor observado da combinação do  $i$ -ésimo nível do fator  $\alpha$  com o  $j$ -ésimo nível do fator  $\tau$  no  $k$ -ésimo bloco;  $\mu$  é a média geral;  $\beta_k$  é o efeito do bloco  $k$ ;  $\alpha_i$  é o efeito do  $i$ -ésimo nível de  $\alpha$ ;  $\tau_j$  é o efeito do  $j$ -ésimo nível de  $\tau$ ;  $(\alpha\tau)_{ij}$  é o efeito da interação do  $i$ -ésimo nível de  $\alpha$  com o  $j$ -ésimo nível de  $\tau$ ; e  $\varepsilon_{ijk}$  é o erro aleatório associado a  $y_{ijk}$ , assumindo  $\varepsilon_{ijk} \stackrel{iid}{\sim} N(0, \sigma^2)$ .

Basicamente, estes fatores podem ser divididos em dois tipos: qualitativos e quantitativos. Um fator qualitativo é, como o nome já diz, relacionado a *qualidade*, ou seja, diferentes em tipo mas não em quantidade. Como exemplo, podemos citar cultivares, defensivos agrícolas, práticas de manejo, etc. Um fator quantitativo, por outro lado, é caracterizado pela *quantidade* utilizada no experimento. Podemos citar, por exemplo, doses de adubação. Cabe ressaltar que o termo *fatorial* não indica um delineamento experimental, mas uma forma de arranjo de tratamentos na área parcela. Estes experimentos podem ser conduzidos tanto em DIC quanto DBC. Assim, em cada repetição/bloco, o tratamento a ser aplicado é a combinação dos níveis dos dois fatores. A combinação de diferentes tipos de fatores não influenciará a análise de variância dos dados, no entanto resultará em algumas particularidades nas análises complementares, como será visto ao longo desta seção.

## 2.5 Download dos dados

Nesta seção, serão analisados dados de experimentos bifatoriais com diferentes tipos de fatores (qualitativos e quantitativos) na presença de interação significativa e não significativa. Em todos os exemplos, será considerado o delineamento de blocos completos casualizados, tendo como variável resposta, o rendimento de grãos (RG). Para isto, utilizaremos cinco conjuntos

de dados que serão detalhados a seguir. Maiores informações sobre estes dados podem ser encontradas em Olivoto T. and Sari (2018a).

Acrônimo	Descrição
FAT1_CI	Fator 1 <b>qualitativo</b> (híbridos), com quatro níveis; Fator 2 <b>quantitativo</b> (doses de N), com cinco níveis, com interação significativa
FAT1_SI	Fator 1 <b>qualitativo</b> (híbridos), com dois níveis; Fator 2 <b>quantitativo</b> (doses de N), com cinco níveis, sem interação significativa
FAT2_CI	Fator 1 <b>qualitativo</b> (fontes de N), com quatro níveis; Fator 2 <b>qualitativo</b> (híbridos), com quatro níveis, com interação significativa
FAT2_SI	Fator 1 <b>qualitativo</b> (fontes de N), com três níveis; Fator 2 <b>qualitativo</b> (híbridos), com quatro níveis, sem interação significativa
FAT3	Fator 1 <b>quantitativo</b> (doses de N), com quatro níveis; Fator 2 <b>quantitativo</b> (doses de K), com cinco níveis, com interação significativa

```
FAT1_CI = read.csv("https://data.mendeley.com/datasets/stwhhd6tj3/1/files/37322c9c-2ce1-4a8e-8f3a-1a2a2a2a2a2a")
FAT1_SI = read.csv("https://data.mendeley.com/datasets/stwhhd6tj3/1/files/91608396-4a8e-8f3a-1a2a2a2a2a2a")
FAT2_CI = read.csv("https://data.mendeley.com/datasets/stwhhd6tj3/1/files/20369627-565a-41a635cf-9f53-4bbf-8f3a-1a2a2a2a2a2a")
FAT2_SI = read.csv("https://data.mendeley.com/datasets/stwhhd6tj3/1/files/41a635cf-9f53-4bbf-8f3a-1a2a2a2a2a2a")
FAT3 = read.csv("https://data.mendeley.com/datasets/stwhhd6tj3/1/files/f1e416bc-abd3-4bbf-8f3a-1a2a2a2a2a2a")
```

O pacote **cursoR** será utilizado nesta seção e na seção biometria-aplicada-ao melhoramento-genetico-de-plantas. Este pacote contém funções para análise multivariada com diversas opções gráficas. As principais funções implementadas no pacote até o momento estão listadas abaixo. Maiores informações podem ser encontradas em T. Olivoto (2018a).

Função	Descrição
<b>CIcorr.mat()</b>	Intervalo de confiança para o coeficiente de correlação de Pearson (matriz)
<b>CIcorr.val()</b>	Intervalo de confiança para o coeficiente de correlação de Pearson (valor)
<b>corr.plot()</b>	Diferentes opções para visualização gráfica de uma matriz de correlação
<b>corr.SS()</b>	Planejamento do tamanho de amostra para o coeficiente de correlação
<b>distdend()</b>	Medidas de dissimilaridade genética e confecção de dendrogramas
<b>ge_stats()</b>	Procedimentos estatísticos para análise de interação genótipo vs ambiente
<b>k_means()</b>	Agrupamento k-means com base em uma tabela de dupla entrada
<b>partial.corr()</b>	Coeficiente de correlação parcial
<b>pass()</b>	Converte uma variável para diferentes tipos (numerico, fator, lógico, etc.)

Função	Descrição
<code>path.coeff()</code>	Análise de trilha com algoritmo para seleção de variáveis
<code>path.diagram()</code>	Diagrama de trilha personalizável
<code>pca()</code>	Análise de componentes principais com diversas opções gráficas
<code>plot_fatcurves()</code>	Ajuste de curvas em experimentos bifatoriais
<code>plot_fatmeans()</code>	Gráficos de barras em experimentos bifatoriais
<code>plot_supresp()</code>	Gráficos de superfície de resposta
<code>plot_uni()</code>	Gráfico para ajuste de curvas em experimentos unifatoriais
<code>plot_supresp()</code>	Gráficos de superfície de resposta
<code>summary_ge_stats()</code>	Resume um objeto da classe <code>ge_stats</code>
<code>summary.path.coeff()</code>	Resume um objeto da classe <code>path.coeff</code>
<code>supresp()</code>	Análise de superfície de resposta
<code>supresp_col()</code>	Cores personalizadas para gráficos de superfície de resposta

## 2.6 Qualitativo vs quantitativo

### 2.6.1 Com interação significativa

O conjunto de dados utilizado neste exemplo será o **FAT1\_CI**. Por se tratar de um experimento factorial com um fator qualitativo (híbrido) e outro quantitativo (dose), convém confeccionar um gráfico com o rendimento observado de cada híbrido em cada dose. Este gráfico, além de servir como ferramenta para identificar possíveis *outliers*, também nos permite identificar a resposta de cada híbrido. Cabe salientar que este é um gráfico meramente ilustrativo. A análise estatística dos dados será realizada posteriormente.

- Visualização dos dados

```
library(ggplot2)
library(cursoR)
library(ExpDes)
p = ggplot(FAT1_CI, aes(x = DOSEN, y = RG)) + # cria um objeto ggplot
  geom_point(aes(colour = factor(HIBRIDO)), size = 1.5) + # adiciona pontos
  geom_smooth(aes(colour = factor(HIBRIDO)), method = "loess") + # adiciona banda
  theme_bw() # adiciona tema preto e branco
p

library(plotly)
ggplotly(p) # gera um grafico iterativo para visualização no R
```

Rendimento observado de cada híbrido em cada dose de nitrogênio

O gráfico acima representa o rendimento de grãos de cada híbrido em cada dose de nitrogênio. Inferências quanto a significância estatística tanto para os efeitos principais (de dose e híbrido) quanto para a interação destes fatores não podem ser feitas neste momento. Como se trata de um gráfico interativo, basta percorrer o ponteiro do mouse nos pontos para obter as informações detalhadas da produtividade. Para isolar um único híbrido, clique na legenda

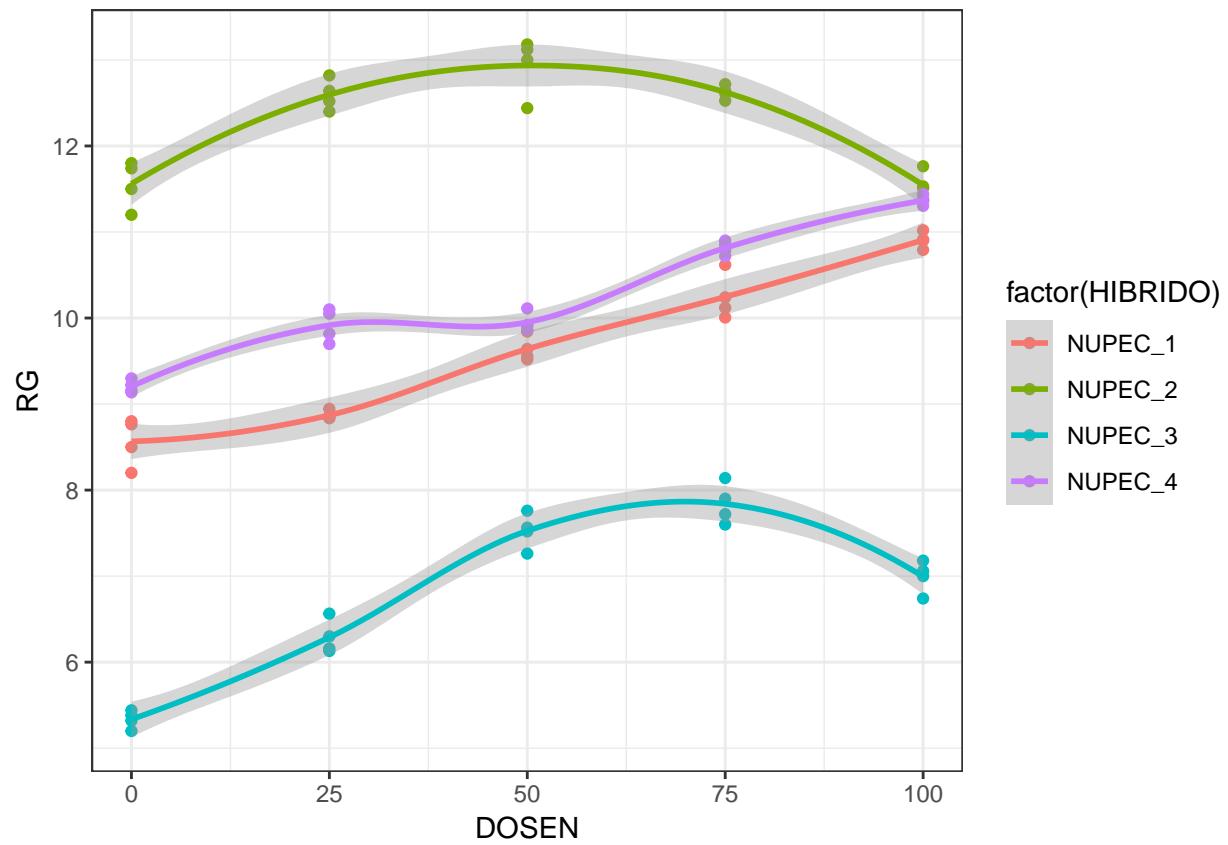


Figure 31: Rendimento observado de cada híbrido em cada dose de nitrogênio

do gráfico. Maiores informações com relação à sintaxe `ggplot2` para confecção de gráficos de linhas podem ser encontradas aqui.

- Análise estatística dos dados

Nesta seção, utilizaremos os pacotes `cursoR` (T. Olivoto 2018a) e `ExpDes` (Ferreira, Cavalcanti, and Nogueira 2018) para análise estatística dos dados. O pacote `ExpDes` contém funções úteis para análise de variância de experimentos nos delineamentos experimentais e arranjo de tratamentos mais conhecidos. A função do pacote `ExpDes` utilizada neste exemplo será `fat2.rbd()` a qual analisa dados de experimentos bifatoriais em delineamento de blocos completos casualizados. Uma explicação detalhada será dada agora ao passo em que nos outros experimentos serão apresentadas apenas as linhas de comandos e uma breve discussão. A delcaração dos argumentos na função é simples. Basta informarmos o nome das colunas do nosso arquivo correspondentes aos argumentos requeridos pela função. Por exemplo `factor1` e `fator2`, representam os fatores em estudo, que, neste caso, são híbridos e doses de nitrogênio. No argumento `block` informamos o nome da coluna de nossos dados correspondente aos blocos. O mesmo para o argumento `resp` que é a variável resposta, no caso, o rendimento de grãos.

Definidos a entrada de dados, precisamos declarar quais as análises complementares a serem realizadas, em caso de significância estatística. no argumento `quali`, informamos qual o tipo de fator em estudo. A declaração é um vetor lógico de comprimento 2. Em nosso caso, ao declararmos `quali = c(TRUE, FALSE)` estamos informando que o primeiro fator (no caso híbrido) é qualitativo e o segundo fator (dose de N) quantitativo. O argumento `mcomp` é utilizado para informar o teste de comparação (ou agrupamento ) de médias utilizado. Em nosso exemplo, vamos utilizar o teste tukey. No argumento `fac.names` é possível informar nomes específicos para os fatores. Os argumentos `sigF` e `sigT` são utilizados para informar a probabilidade de erro assumida na análise de variância e nas análises de comparação de médias, respectivamente. Ambos tem padrão 5%.

O resultado da função `fat2.rbd()` irá depender da significância das fontes de variação do experimento. Neste caso em específico, em caso de interação significativa (o que já sabemos devido ao título da subseção), uma regressão é ajustada para cada híbrido e as médias de cada híbrido são comparadas em cada dose de nitrogênio. Vamos ao exemplo.

```
library(ExpDes)
attach(FAT1_CI)

F1_CI = fat2.rbd(factor1 = HIBRIDO,
                  factor2 = DOSEN,
                  block = BLOCO,
                  resp = RG,
                  quali = c(TRUE, FALSE),
                  mcomp = "tukey",
                  fac.names = c("HIBRIDO", "DOSE"),
                  sigT = 0.05,
                  sigF = 0.05)

## -----
## Legend:
## FACTOR 1: HIBRIDO
```

```

## FACTOR 2: DOSE
## -----
## 
## Analysis of Variance Table
## -----
##          DF   SS    MS   Fc  Pr>Fc
## Block      3  0.37  0.122  4.2 0.009578
## HIBRIDO    3 304.85 101.617 3478.6 0.000000
## DOSE       4 31.33  7.831  268.1 0.000000
## HIBRIDO*DOSE 12 18.01  1.500   51.4 0.000000
## Residuals   57  1.67  0.029
## Total       79 356.21
## -----
## CV = 1.76 %
## 
## -----
```

## Shapiro-Wilk normality test  
## p-value: 0.095938  
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered approximately normally distributed.

```

## 
## -----
```

## Significant interaction: analyzing the interaction

```

## -----
```

## Analyzing HIBRIDO inside of each level of DOSE

```

## -----
```

## -----

## Analysis of Variance Table

```

##          DF   SS    MS   Fc  Pr.Fc
## Block      3  0.36656  0.12219  4.1827 0.0096
## DOSE       4 31.32538  7.83135  268.0829   0
## HIBRIDO:DOSE 0  3 79.07881  26.3596  902.343   0
## HIBRIDO:DOSE 25 3 81.74143  27.24714  932.7253   0
## HIBRIDO:DOSE 50 3 59.46093  19.82031  678.4896   0
## HIBRIDO:DOSE 75 3 46.79345  15.59782  533.9451   0
## HIBRIDO:DOSE 100 3 55.78144  18.59381  636.5041   0
## Residuals   57  1.66511  0.02921
## Total       79 356.21311
## -----
```

##

##

##

```

##  HIBRIDO  inside of the level  0  of  DOSE
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      2   11.56
## b      4   9.2035
## c      1   8.566
## d      3   5.335
## -----
## 
## 
##  HIBRIDO  inside of the level  25  of  DOSE
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      2   12.595
## b      4   9.9165
## c      1   8.8705
## d      3   6.2885
## -----
## 
## 
##  HIBRIDO  inside of the level  50  of  DOSE
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      2   12.935
## b      4   9.9505
## b      1   9.63975
## c      3   7.52625
## -----
## 
## 
##  HIBRIDO  inside of the level  75  of  DOSE
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      2   12.625
## b      4   10.8145
## c      1   10.246
## d      3   7.84
## -----

```

```

##  

##  

##  HIBRIDO  inside of the level 100 of DOSE  

## -----  

## Tukey's test  

## -----  

## Groups Treatments Means  

## a      2    11.5465  

## a      4    11.367  

## b      1    10.906  

## c      3     6.995  

## -----  

##  

##  

##  

## Analyzing DOSE inside of each level of HIBRIDO  

## -----  

## -----  

## Analysis of Variance Table  

## -----  

##          DF      SS      MS      Fc  Pr.Fc  

## Block        3  0.36656  0.12219    4.1827 0.0096  

## HIBRIDO      3 304.85101 101.617 3478.5573    0  

## DOSE:HIBRIDO NUPEC_1  4  14.86175  3.71544   127.187    0  

## DOSE:HIBRIDO NUPEC_2  4   6.79944  1.69986   58.1897    0  

## DOSE:HIBRIDO NUPEC_3  4  16.21950  4.05487   138.8066    0  

## DOSE:HIBRIDO NUPEC_4  4  11.44973  2.86243   97.9869    0  

## Residuals    57   1.66511  0.02921  

## Total        79 356.21311  

## -----  

##  

##  

##  

##  

##  DOSE  inside of the level NUPEC_1  of HIBRIDO  

## -----  

## Adjustment of polynomial models of regression  

## -----  

##  

##  

## Linear Model  

## ======  

## Estimate Standard.Error    tc    p.value  

## -----  

## b0  8.4345      0.0662    127.4185    0  

## b1  0.0242      0.0011    22.4077    0  

## -----  

##  

##
```

```

## R2 of linear model
## -----
## 0.986938
## -----
## 
## Analysis of Variance of linear model
## =====
##          DF   SS      MS   Fc   p.value
## -----
## Linear Effect 1 14.6676 14.6676 502.1    0
## Lack of fit   3  0.1941  0.0647  2.22  0.0962
## Residuals     57 1.6651  0.0292
## -----
## -----
## 
## Quadratic Model
## =====
##      Estimate Standard.Error   tc   p.value
## -----
## b0  8.5128        0.0804 105.8458    0
## b1  0.0180        0.0038   4.7126  0.00002
## b2  0.0001        0.00004   1.7138  0.0920
## -----
## 
## 
## R2 of quadratic model
## -----
## 0.992711
## -----
## 
## Analysis of Variance of quadratic model
## =====
##          DF   SS      MS   Fc   p.value
## -----
## Linear Effect  1 14.6676 14.6676 502.1    0
## Quadratic Effect 1 0.0858  0.0858  2.94  0.092
## Lack of fit    2  0.1083  0.0542  1.85  0.16592
## Residuals      57 1.6651  0.0292
## -----
## 
## 
## Cubic Model
## =====
##      Estimate Standard.Error   tc   p.value
## -----
## b0  8.5539        0.0848 100.8177    0
## b1  0.0062        0.0086   0.7155  0.4772

```

```

## b2 0.0004      0.0002      1.7852  0.0796
## b3 -0.000002    0          -1.5209  0.1338
## -----
## 
## R2 of cubic model
## -----
## 0.997258
## -----
## 
## Analysis of Variance of cubic model
## -----
##           DF   SS     MS   Fc  p.value
## -----
## Linear Effect 1 14.6676 14.6676 502.1  0
## Quadratic Effect 1 0.0858  0.0858  2.94  0.092
## Cubic Effect   1 0.0676  0.0676  2.31  0.13382
## Lack of fit    1 0.0408  0.0408  1.4   0.24246
## Residuals      57 1.6651  0.0292
## -----
## -----
## 
## DOSE inside of the level NUPEC_2 of HIBRIDO
## -----
## Adjustment of polynomial models of regression
## -----
## 
## Linear Model
## -----
##   Estimate Standard.Error   tc  p.value
## -----
## b0 12.2517      0.0662      185.0832  0
## b1 0.00001      0.0011      0.0111  0.9912
## -----
## 
## R2 of linear model
## -----
## 0.000001
## -----
## 
## Analysis of Variance of linear model
## -----
##           DF   SS     MS   Fc  p.value
## -----
## Linear Effect 1  0      0      0  0.99118
## Lack of fit   3 6.7994 2.2665 77.59  0

```

```

## Residuals      57 1.6651 0.0292
## -----
## -----
## 
## Quadratic Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
##   b0 11.5550      0.0804    143.6710    0
##   b1  0.0557      0.0038     14.6290    0
##   b2 -0.0006      0.00004   -15.2523    0
## -----
## 
## 
## R2 of quadratic model
## -----
## 0.999458
## -----
## 
## 
## Analysis of Variance of quadratic model
## =====
##          DF   SS     MS     Fc    p.value
## -----
## Linear Effect 1    0     0     0    0.99118
## Quadratic Effect 1 6.7958 6.7958 232.63    0
## Lack of fit   2 0.0037 0.0018  0.06  0.93889
## Residuals      57 1.6651 0.0292
## -----
## 
## 
## 
## Cubic Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
##   b0 11.5623      0.0848    136.2751    0
##   b1  0.0536      0.0086     6.2132    0
##   b2 -0.0005      0.0002   -2.2739  0.0268
##   b3 -0.000000      0       -0.2720  0.7866
## -----
## 
## 
## R2 of cubic model
## -----
## 0.999775
## -----
## 
## 
## Analysis of Variance of cubic model
## =====

```

```

##                   DF   SS     MS      Fc    p.value
## -----
## Linear Effect    1    0      0      0    0.99118
## Quadratic Effect 1  6.7958 6.7958 232.63    0
## Cubic Effect     1  0.0022 0.0022  0.07  0.78662
## Lack of fit      1  0.0015 0.0015  0.05  0.81994
## Residuals        57 1.6651 0.0292
## -----
## -----
## 
## 
## DOSE  inside of the level  NUPEC_3  of  HIBRIDO
## -----
## Adjustment of polynomial models of regression
## -----
## 
## 
## Linear Model
## =====
##   Estimate Standard.Error   tc    p.value
## -----
##   b0  5.8227      0.0662    87.9613    0
##   b1  0.0195      0.0011   18.0264    0
## -----
## 
## 
## R2 of linear model
## -----
##  0.585259
## -----
## 
## 
## Analysis of Variance of linear model
## =====
##                   DF   SS     MS      Fc    p.value
## -----
## Linear Effect  1  9.4926 9.4926 324.95    0
## Lack of fit    3  6.7269 2.2423 76.76    0
## Residuals      57 1.6651 0.0292
## -----
## 
## 
## 
## Quadratic Model
## =====
##   Estimate Standard.Error   tc    p.value
## -----
##   b0  5.1768      0.0804    64.3666    0
##   b1  0.0712      0.0038   18.6715    0
##   b2 -0.0005      0.00004   -14.1389   0

```

```

## -----
## 
## R2 of quadratic model
## -----
## 0.945309
## -----
## 
## Analysis of Variance of quadratic model
## =====
##          DF   SS    MS     Fc   p.value
## -----
## Linear Effect  1  9.4926 9.4926 324.95   0
## Quadratic Effect 1  5.8398 5.8398 199.91   0
## Lack of fit    2  0.8870 0.4435 15.18  1e-05
## Residuals      57 1.6651 0.0292
## -----
## -----
## 
## Cubic Model
## =====
##      Estimate Standard.Error   tc   p.value
## -----
## b0  5.3211        0.0848   62.7150   0
## b1  0.0298        0.0086   3.4503  0.0011
## b2  0.0006        0.0002   2.9085  0.0052
## b3 -0.00001       0        -5.3396   0
## -----
## 
## R2 of cubic model
## -----
## 0.996661
## -----
## 
## Analysis of Variance of cubic model
## =====
##          DF   SS    MS     Fc   p.value
## -----
## Linear Effect  1  9.4926 9.4926 324.95   0
## Quadratic Effect 1  5.8398 5.8398 199.91   0
## Cubic Effect    1  0.8329 0.8329 28.51   0
## Lack of fit    1  0.0542 0.0542  1.85  0.1787
## Residuals      57 1.6651 0.0292
## -----
## -----
## 
## 
```

```

## DOSE inside of the level NUPEC_4 of HIBRIDO
## -----
## Adjustment of polynomial models of regression
## -----
## 
## Linear Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
##   b0  9.2054      0.0662   139.0636    0
##   b1  0.0209      0.0011   19.3345    0
## -----
## 
## R2 of linear model
## -----
## 0.953756
## -----
## 
## Analysis of Variance of linear model
## =====
##          DF   SS     MS     Fc    p.value
## -----
## Linear Effect 1 10.9202 10.9202 373.82    0
## Lack of fit   3 0.5295  0.1765  6.04  0.0012
## Residuals     57 1.6651  0.0292
## -----
## 
## 
## Quadratic Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
##   b0  9.2781      0.0804   115.3611    0
##   b1  0.0151      0.0038    3.9578  0.0002
##   b2  0.0001      0.00004   1.5918  0.1170
## -----
## 
## R2 of quadratic model
## -----
## 0.960221
## -----
## 
## Analysis of Variance of quadratic model
## =====
##          DF   SS     MS     Fc    p.value
## -----

```

```

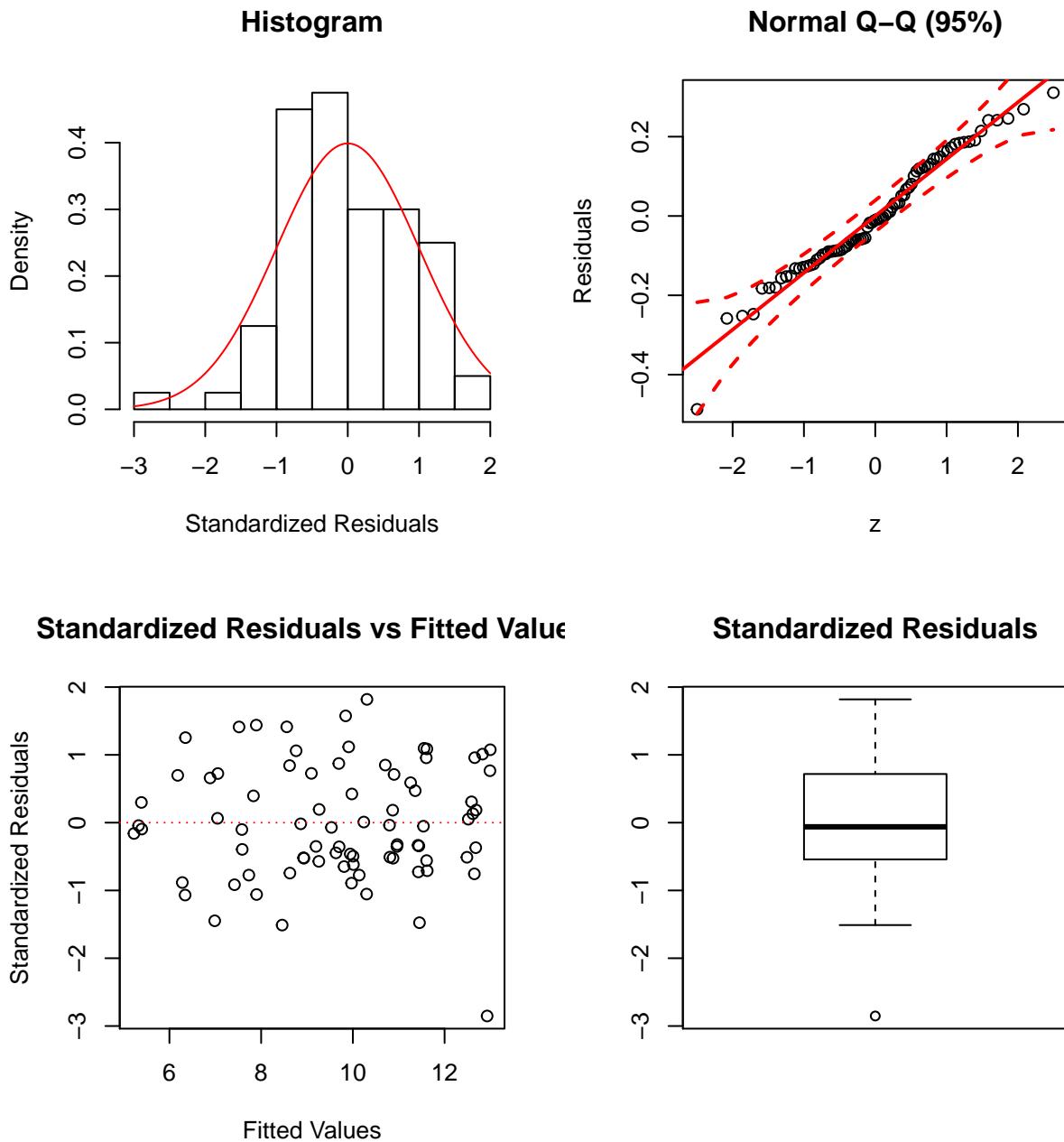
## Linear Effect      1 10.9202 10.9202 373.82      0
## Quadratic Effect 1 0.0740  0.0740   2.53  0.11695
## Lack of fit       2 0.4555  0.2277   7.8   0.00102
## Residuals         57 1.6651  0.0292
## -----
## -----
## 
## Cubic Model
## =====
##   Estimate Standard.Error    tc   p.value
## -----
## b0  9.2414      0.0848    108.9198     0
## b1  0.0256      0.0086     2.9672  0.0044
## b2 -0.0002      0.0002    -1.0756  0.2866
## b3  0.000002     0        1.3599  0.1792
## -----
## 
## R2 of cubic model
## -----
## 0.964939
## -----
## 
## Analysis of Variance of cubic model
## =====
##          DF   SS     MS    Fc   p.value
## -----
## Linear Effect  1 10.9202 10.9202 373.82      0
## Quadratic Effect 1 0.0740  0.0740   2.53  0.11695
## Cubic Effect    1 0.0540  0.0540   1.85  0.17922
## Lack of fit     1 0.4014  0.4014  13.74  0.00048
## Residuals       57 1.6651  0.0292
## -----
## 
## 
detach(FAT1_CI)

```

- Análise dos resíduos

De acordo com o teste de normalidade Shapiro-Wilk (5% de erro), os resíduos podem ser considerados normais. A função `plotres()` do pacote `ExpDes` é utilizada para a plotagem dos resíduos do modelo ajustado. É possível observar que apenas um ponto foi considerado como *outlier*.

```
plotres(F1_CI)
```



A análise de variância acima indicou efeitos significativos tanto para os efeitos principais, quanto para a interação. Assim, as análises complementares que foram realizadas foram (i) a comparação das médias pelo teste Tukey em cada nível da dose de N; e (ii) uma regressão polinomial ajustada para cada híbrido. Por padrão, o máximo grau do polinômio ajustado é 3 (modelo cúbico).

- Comparação das médias dos híbridos em cada dose de nitrogênio.

As comparações de médias são apresentadas após a análise de variância, na saída acima. Neste momento, utilizaremos a função `plot_fatmeans()` do pacote `cursoR` para plotar as médias dos híbridos em cada dose de nitrogênio. A apresentação gráfica de resultados, mesmo

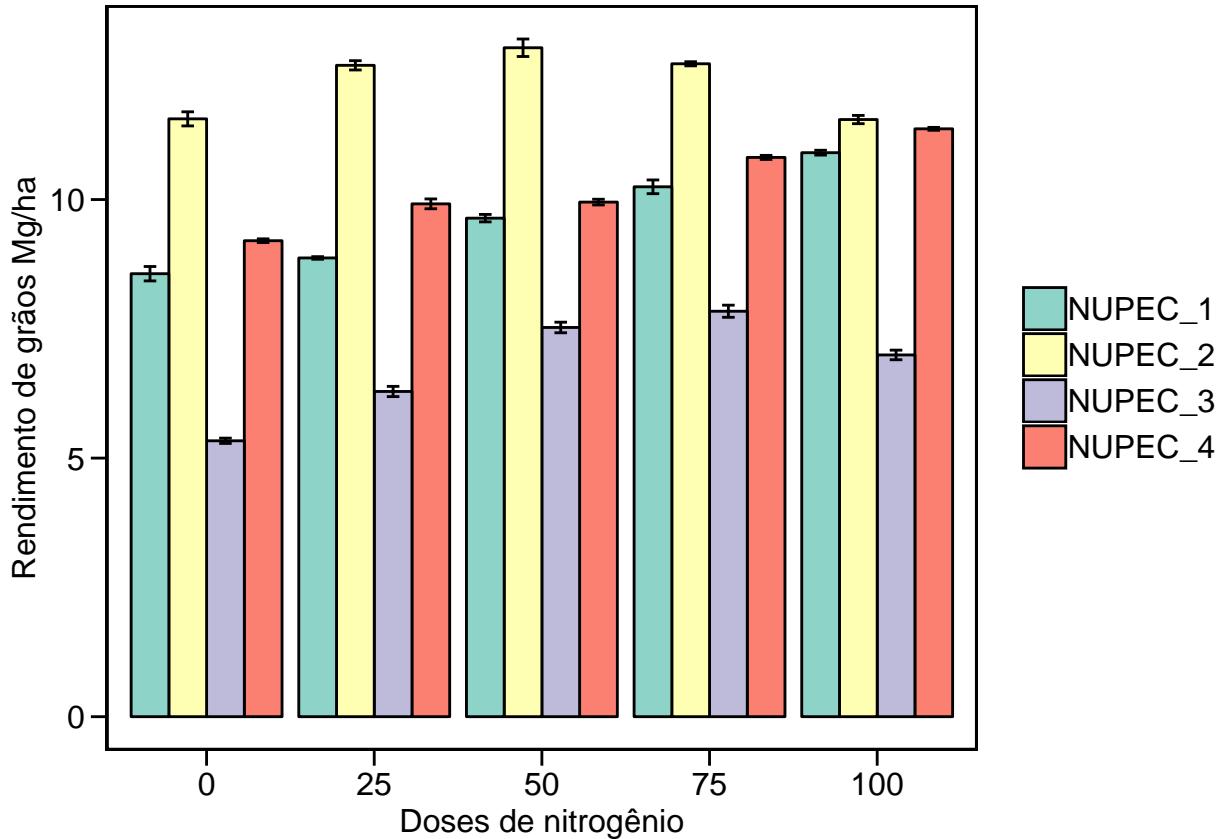


Figure 32: Gráfico das médias dos híbridos em cada dose de nitrogênio gerado pela função `plot_fatmeans()`

considerando médias, é uma alternativa interessante à tabela, pois permite uma interpretação mais clara e intuitiva dos resultados. Para fazer isto, primeiramente, precisamos criar um novo conjunto de dados (`FAT1_CI_means`), considerando a coluna **DOSEN** como um fator. Isto é necessário, pois na importação dos dados, por padrão, dados numéricos não são importados como fator. A função `pass()`, também do pacote `cursoR` irá ser utilizada para este fim.

```
library(cursoR)
FAT1_CI_means = pass(FAT1_CI, "DOSEN", as.factor) # tornando a coluna DOSEN um fator
plot_fatmeans(FAT1_CI_means,
              measurevar = "RG",
              groupvars = c("DOSEN", "HIBRIDO"),
              verbose = FALSE,
              xlab = "Doses de nitrogênio",
              ylab = "Rendimento de grãos Mg/ha",
              palette = "Set3",
              legend.position = "right")
```

- Ajuste de regressão para cada híbrido

No exemplo anterior, apresentamos as médias dos híbridos em cada dose de nitrogênio. Agora, criaremos um gráfico com o grau do polinômio significativo ajustado de cada híbrido. O

grau a ser ajustado deve ser identificado na saída da ANOVA , apresentada acima. Para fins didáticos um resumo é fornecido abaixo.

Híbrido	Polinômio	Equação
NUPEC_1	Linear	$y = 8.4345 + 0.0242 \times x, R^2 = 0.986$
NUPEC_2	Quadrático	$y = 11.555 + 0.0557 \times x - 0.0006 \times x^2, R^2 = 0.999$
NUPEC_3	Quadrático	$y = 5.1768 + 0.0712 \times x - 0.0005 \times x^2, R^2 = 0.945$
NUPEC_4	Linear	$y = 9.2054 + 0.0209 \times x, R^2 = 0.986$

Utilizando uma equação, é possível estimar a produtividade para uma dose de nitrogênio específica não testada, desde que ela esteja dentro do intervalo estudado. Para isto, basta substituir o  $x$  na equação pela dose a ser testada. Por exemplo para sabermos qual seria a produtividade do híbrido *NUPEC\_1* se tivéssemos aplicado 60 kg de N ha<sup>-1</sup> basta resolver:  $y = 8.4345 + 0.0242 \times 60$ , resultando em  $y = 9.886$ . A interpretação deste resultado, no entanto, deve ser cautelosa. Inconscientemente, concluiríamos que a produtividade do híbrido aumentaria 0.0242 Mg ha<sup>-1</sup> a cada kg de nitrogênio aplicado por hectare. Este fato, no entanto, não é observado na prática. Por exemplo, a produtividade não irá aumentar infinitamente a medida em que se aumenta a dose de nitrogênio aplicado. A única conclusão válida, neste caso, é que a produtividade aumenta linearmente até 100 kg de N ha<sup>-1</sup>. Este resultado se deu em virtude de as doses testadas não terem sido o suficiente para identificar um outro comportamento na variável testada. Nestes casos, indica-se para estudos futuros aumentar o número de doses. Quando não se conhece o intervalo de dose em que a variável em estudo apresenta uma resposta explicável, estudos pilotos podem ser realizados. Neste caso, testar-se-iam o mesmo número de tratamentos (número de doses), no entanto com um intervalo maior entre as doses (por exemplo, 0, 100, 200, 300 e 400 kg de N ha<sup>-1</sup>). Possivelmente, nesta amplitude, o comportamento da produtividade não seria linear, pois em uma determinada dose, a produtividade estabilizaria.

Semelhante ao exemplo das médias nas doses de nitrogênio, utilizaremos a função `plot_fatcurves()` para plotar, agora, uma regressão ajustada para cada híbrido. Para isto, precisaremos definir a coluna híbrido como fator para que uma regressão possa ser ajustada para cada um de seus níveis. Os argumentos a serem informados são os seguintes: `data`, o conjunto de dados (neste caso *FAT1\_CI\_reg*); `x` e `y`, as colunas dos dados correspondentes aos eixos x e y do gráfico, respectivamente; `group` a coluna que contém os níveis dos fatores em que as regressões serão ajustadas; `fit` um vetor de comprimento igual ao número de níveis da coluna informada em `group`. O número indicado em cada posição do vetor, corresponde ao grau do polinômio ajustado (máximo grau ajustado = 4). Em nosso exemplo, utilizaremos `fit = c(1, 2, 2, 1)` para ajustar uma regressão linear para os híbridos *NUPEC\_1* e *NUPEC\_4*, uma regressão quadrática para os híbridos *NUPEC\_2* e *NUPEC\_3*. Outro possível valor neste argumento é declaração de apenas um número. Quando isto ocorre, a função identifica que uma única regressão deve ser ajustada para todos os níveis do determinado fator. Isto é útil e utilizaremos a seguir em casos de interação não significativa.

```
library(cursoR)
FAT1_CI_reg = cursoR::pass(FAT1_CI, "HIBRIDO", as.factor)
plot_fatcurves(data = FAT1_CI_reg,
                x = "DOSEN",
```

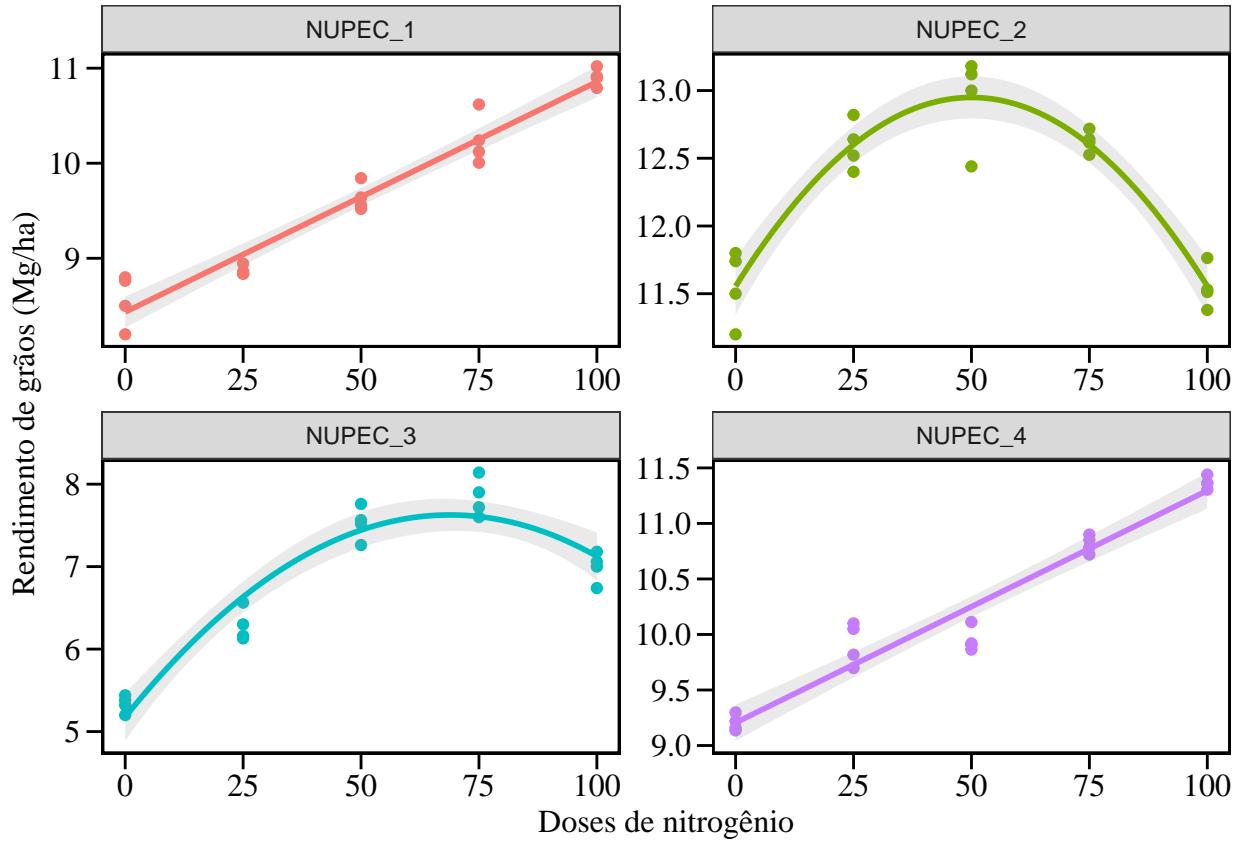


Figure 33: Gráfico com uma regressão ajustada para cada híbrido gerado pela função `plot_fatcurves()`

```

y = "RG",
group = "HIBRIDO",
fit = c(1,2,2,1),
xlab = "Doses de nitrogênio",
ylab = "Rendimento de grãos (Mg/ha)",
grid = T,
col = T,
size.shape = 1.5,
fontfam = "serif")

```

Observando-se a figura acima, é possível identificar o comportamento quadrático da variável resposta dos híbridos *NUPEC\_2* e *NUPEC\_3*. Para estes híbridos, houve um incremento positivo na produtividade até um ponto, posteriormente observa-se que a produtividade tendeu a reduzir. Uma explicação biológica para esta redução seria que o excesso de nitrogênio aplicado proporcionou um alto vigor vegetativo as plantas, podendo ter ocorrido competição entre as plantas por água, luz e outros nutrientes, ou até mesmo tombamento das plantas. O ponto em X (dose) em que a produtividade é máxima é chamado de máxima eficiência técnica (MET) e pode ser estimado por:

$$MET = \frac{-\beta_1}{2 \times \beta_2}$$

Tomando o híbrido *NUPEC\_2* como exemplo, temos:

$$MET = \frac{-0.0557}{2 \times -0.0006} = 46.41$$

Logo, a dose que proporciona a máxima produtividade é aproximadamente 46 kg de N ha<sup>-1</sup>. Assim para sabermos qual é esta produtividade estimada, basta substituir o *x* da equação por 46.41, resultando em  $y_{máx} = 12.84$  Mg ha<sup>-1</sup>.

Outro ponto importante que é possível de estimar utilizando uma equação de segundo grau, é a máxima eficiência econômica (MEE), ou seja, a dose máxima, neste caso de nitrogênio, em que é possível aplicar obtendo-se lucro. Este ponto é importante, pois a partir de uma certa dose, os incrementos em produtividade não compensariam o preço pago pelo nitrogênio aplicado. Este ponto pode ser facilmente estimado por

$$MEE = MET + \frac{u}{2 \times b2 \times m}$$

onde *u* e *m* são os preços da ureia e do milho em grão, respectivamente, na mesma unidade utilizada para a estimativa da equação (neste caso, preço da ureia por kg e preço do milho por tonelada). Considerando o preço de custo da ureia como R\$ 1,35 por kg e o preço de venda do milho a R\$ 600,00 por tonelada, substituindo-se na formula obtém-se:

$$MEE = 46.41 + \frac{1.35}{2 \times (-0.0006) \times 600} = 44.535$$

Assim, a dose máxima de nitrogênio que em que os incrementos de produtividade são lucrativos é de ~45 N ha<sup>-1</sup>.

## 2.6.2 Sem interação significativa

O conjunto de dados utilizado neste exemplo será o **FAT1\_SI**. Do mesmo modo do exemplo anterior, iremos confeccionar um gráfico prévio para visualização dos dados.

- Visualização dos dados

```
p = ggplot(FAT1_SI, aes(x = DOSEN, y = RG)) +
  geom_point(aes(colour = factor(HIBRIDO)), size = 1.5) +
  geom_smooth(aes(colour = factor(HIBRIDO)), method = "loess") +
  theme_bw()

p
```

```
ggplotly(p) # gera o grafico iterativo
```

Característica de produção em um experimento bifatorial sem interação significativa

- análise estatística dos dados.

A função **fat2.rdb()** será novamente utilizada neste exemplo. A declaração dos argumentos é idêntica ao exemplo anterior. O que mudará aqui, é a maneira com que as análises complementares serão realizadas, visto que, como já sabemos, a interação não será significativa.

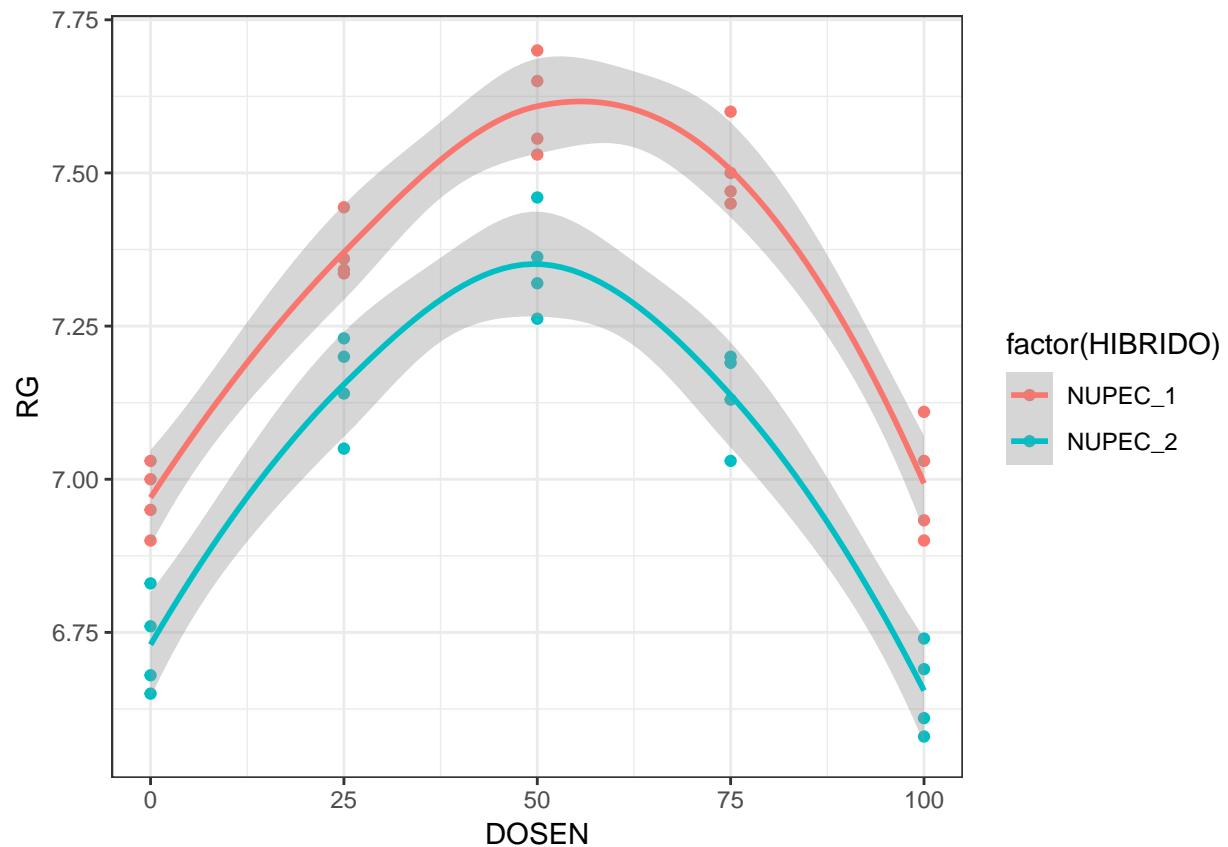


Figure 34: Característica de produção em um experimento bifatorial sem interação significativa

```

attach(FAT1_SI)
F1_SI = fat2.rbd(factor1 = HIBRIDO,
                  factor2 = DOSEN,
                  block = BLOCO,
                  resp = RG,
                  quali = c(TRUE, FALSE),
                  mcomp = "tukey",
                  fac.names = c("HIBRIDO", "DOSE"),
                  sigT = 0.05,
                  sigF = 0.05)
## -----
## Legend:
## FACTOR 1: HIBRIDO
## FACTOR 2: DOSE
## -----
## 
## 
## Analysis of Variance Table
## -----
##          DF      SS      MS      Fc   Pr>Fc
## Block      3 0.0257 0.00857  1.593 0.2141
## HIBRIDO    1 0.8054 0.80542 149.758 0.0000
## DOSE        4 2.7775 0.69438 129.110 0.0000
## HIBRIDO*DOSE 4 0.0345 0.00862  1.602 0.2025
## Residuals   27 0.1452 0.00538
## Total       39 3.7883
## -----
## CV = 1.03 %
## 
## -----
## Shapiro-Wilk normality test
## p-value: 0.3316718
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be cons
## -----
## 
## No significant interaction: analyzing the simple effect
## -----
## HIBRIDO
## Tukey's test
## -----
## Groups Treatments Means
## a     NUPEC_1     7.28955
## b     NUPEC_2     7.00575
## -----
## 
## 
## DOSE

```

```

## Adjustment of polynomial models of regression
## -----
## 
## Linear Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
## b0  7.1463      0.0201    355.8227    0
## b1  0.00003     0.0003     0.0823  0.9350
## -----
## 
## R2 of linear model
## -----
## 0.000013
## -----
## 
## Analysis of Variance of linear model
## =====
##          DF   SS     MS     Fc    p.value
## -----
## Linear Effect 1 0.00004 0.00004  0.01    0.935
## Lack of fit   3 2.7775  0.9258  172.15    0
## Residuals      27 0.1452  0.0054
## -----
## 
## 
## Quadratic Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
## b0  6.8326      0.0244    280.0053    0
## b1  0.0251      0.0012     21.7294    0
## b2 -0.0003      0.00001   -22.6358    0
## -----
## 
## R2 of quadratic model
## -----
## 0.992149
## -----
## 
## Analysis of Variance of quadratic model
## =====
##          DF   SS     MS     Fc    p.value
## -----
## Linear Effect   1 0.00004 0.00004  0.01    0.935
## Quadratic Effect 1 2.7557  2.7557  512.38    0

```

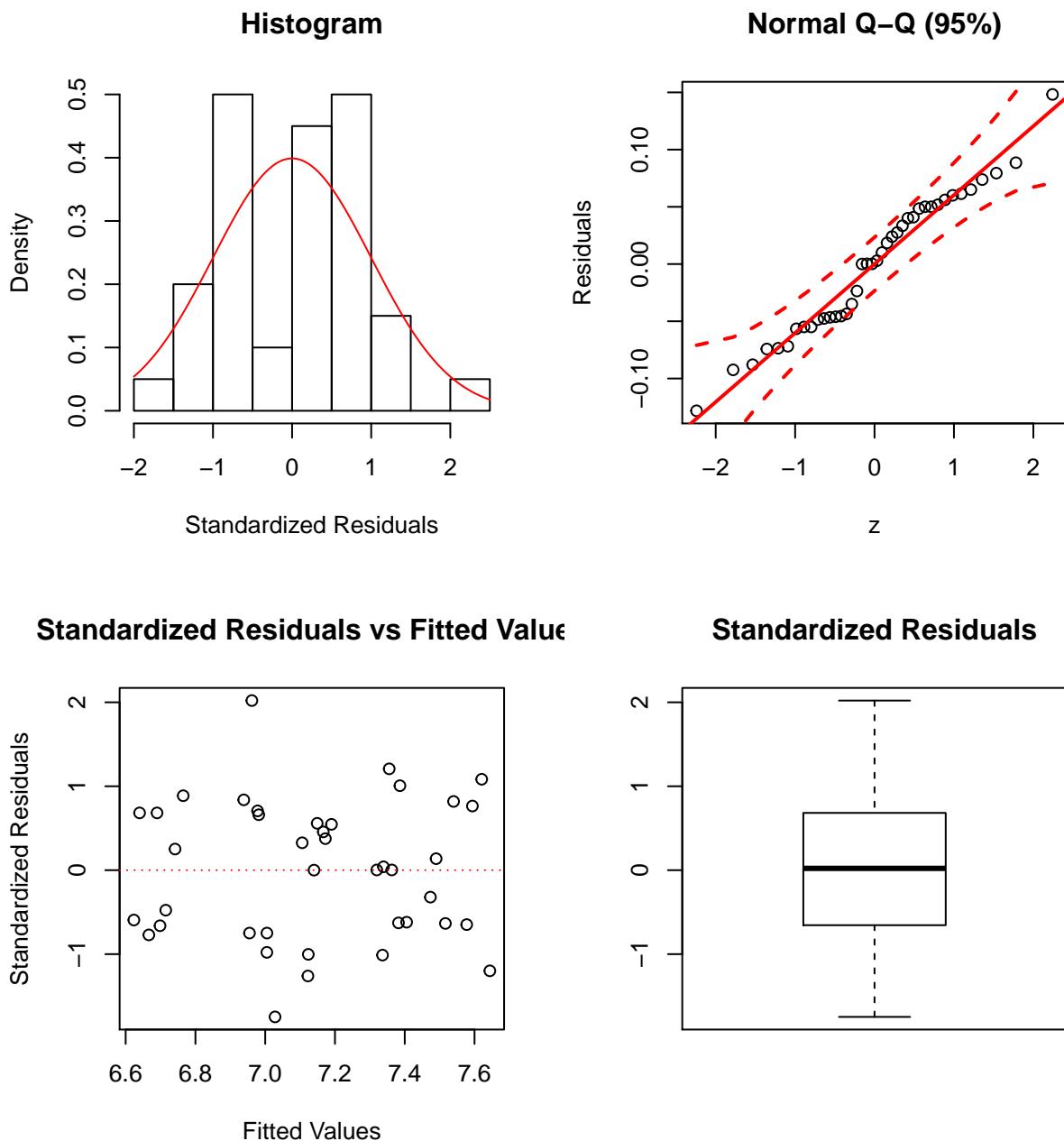
```

## Lack of fit      2  0.0218  0.0109   2.03  0.15126
## Residuals       27  0.1452  0.0054
## -----
## -----
## 
## Cubic Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
##   b0   6.8469      0.0257  265.9774     0
##   b1   0.0210      0.0026     8.0278     0
##   b2   -0.0001     0.0001   -2.0545  0.0497
##   b3  -0.000001      0       -1.7426  0.0928
## -----
## 
## 
## R2 of cubic model
## -----
## 0.998029
## -----
## 
## Analysis of Variance of cubic model
## =====
##          DF   SS     MS    Fc    p.value
## -----
## Linear Effect  1  0.00004 0.00004  0.01  0.935
## Quadratic Effect 1  2.7557  2.7557  512.38     0
## Cubic Effect    1  0.0163  0.0163   3.04  0.09279
## Lack of fit     1  0.0055  0.0055   1.02  0.32195
## Residuals       27  0.1452  0.0054
## -----
## 
## 
detach(FAT1_SI)

```

- Análise dos resíduos

```
plotres(F1_SI)
```



Como a interação não foi significativa, proceder-se-a a comparação de médias dos dois híbridos considerando a média de todas as doses de nitrogênio, e o ajuste de apenas uma regressão para os dois híbridos.

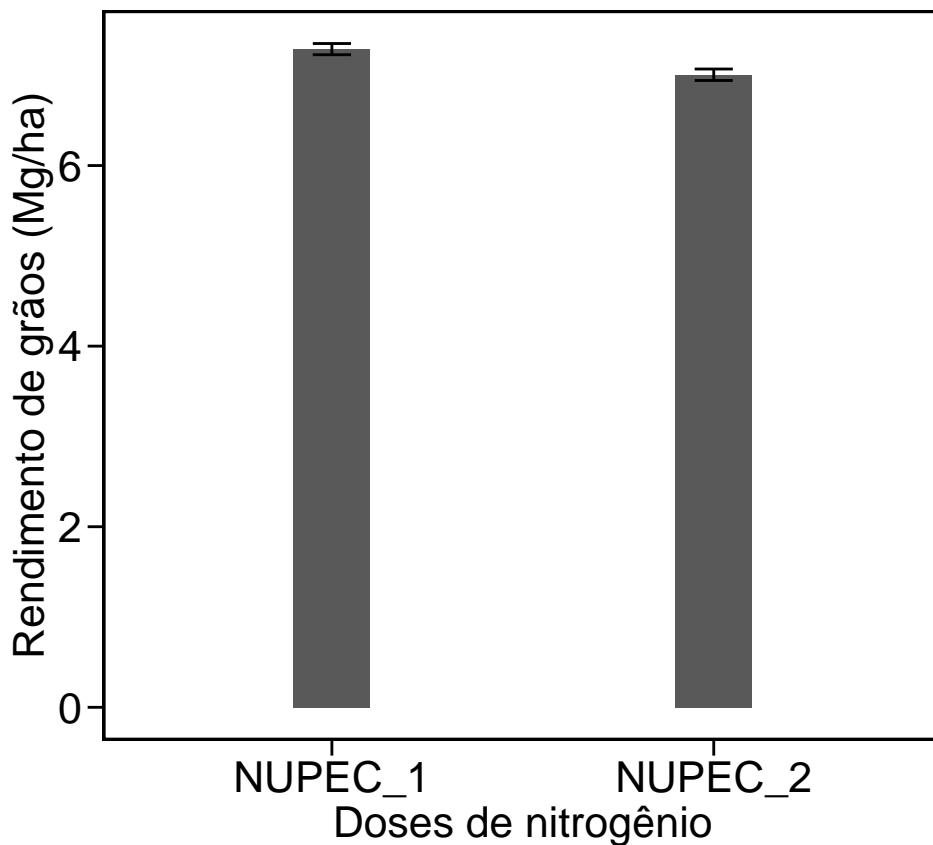
- comparação de médias dos híbridos considerando a média de todas as doses de N.

```
## -----
## HIBRIDO
## Tukey's test
## -----
```

```

## Groups Treatments Means
## a      NUPEC_1      7.28955
## b      NUPEC_2      7.00575
## -----
FAT1_SI_means = pass(FAT1_SI, "HIBRIDO", as.factor)
plot_fatmeans(FAT1_SI_means,
               measurevar = "RG",
               groupvars = c("HIBRIDO"),
               verbose = FALSE,
               xlab = "Doses de nitrogênio",
               ylab = "Rendimento de grãos (Mg/ha)",
               cex = 16,
               width.bar = 0.2,
               width.erbar = 0.1)

```



- ajuste de uma única regressão para os dois híbridos.

Diferentemente do exemplo anterior, para o ajuste de uma única regressão para os dois híbridos, a única alteração que precisamos fazer será declarar apenas um valor no argumento `fit`. Neste exemplo, ajustaremos uma regressão quadrática, que foi o grau do polinômio significativo observado no procedimento da análise de variância. Assim basta declararmos `fit = 2`

```

## Quadratic Model
## =====
##   Estimate Standard.Error    tc    p.value
## -----
## b0  6.8326      0.0244   280.0053    0
## b1  0.0251      0.0012   21.7294    0
## b2 -0.0003      0.00001  -22.6358    0
## -----
## R2 of quadratic model
## -----
## 0.992149
## -----
## Analysis of Variance of quadratic model
## =====
##           DF   SS     MS    Fc    p.value
## -----
## Linear Effect 1 0.00004 0.00004  0.01  0.935
## Quadratic Effect 1 2.7557  2.7557  512.38  0
## Lack of fit    2 0.0218  0.0109   2.03  0.15126
## Residuals       27 0.1452  0.0054
## -----
FAT1_SI_reg = pass(FAT1_SI, "HIBRIDO", as.factor)
plot_fatcurves(data = FAT1_SI_reg,
                 x = "DOSEN",
                 y = "RG",
                 group = "HIBRIDO",
                 fit = 2,
                 xlab = "Doses de nitrogênio",
                 ylab = "Rendimento de grãos (Mg/ha)",
                 grid = F,
                 cex = 16,
                 col = T,
                 size.shape = 1.5,
                 fontfam = "serif")

```

## 2.7 Qualitativo vs qualitativo

Nos dois exemplos anteriores, vimos exemplos de análise de experimentos bitaforiais na presença de um fator qualitativo e outro quantitativo . Nesta seção, utilizaremos as mesmas funções, no entanto, na análise de dados de experimentos bifatoriais com dois fatores qualitativos. Assim, em caso de interação significativa, é necessária comparação de dos níveis de um fator fixando um nível do segundo fator, e vice versa.

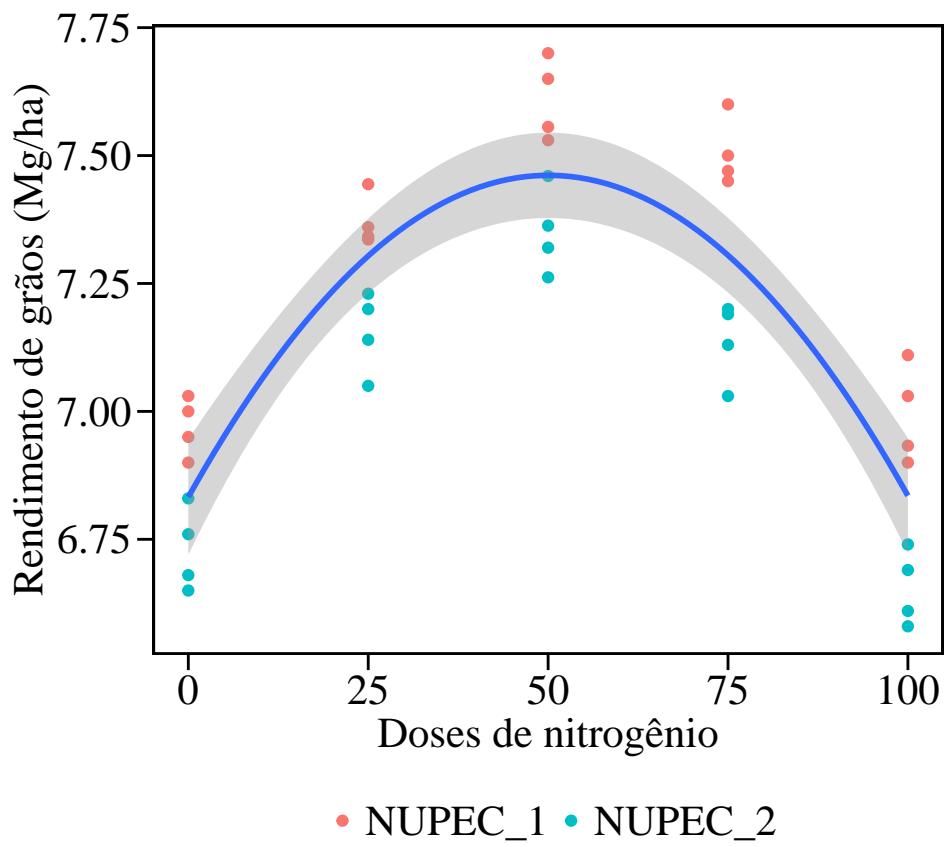


Figure 35: Curvas ajustadas para híbrido utilizando a função `plot_fatcurves()`

### 2.7.1 Com interação significativa

O conjunto de dados utilizado neste exemplo será o **FAT2\_CI**. A análise de variância é realizada utilizando a mesma função anterior. Neste exemplo, no entanto, precisaremos mudar um dos valores lógicos declarados no argumento `quali`. Como agora temos dois fatores qualitativos, para a correta análise precisamos declarar `quali = c(TRUE, TRUE)`. Vamos ao exemplo.

- Análise de variância

```
attach(FAT2_CI)
fat2.rbd(factor1 = HIBRIDO,
          factor2 = FONTEN,
          block = BLOCO,
          resp = RG,
          quali = c(TRUE, TRUE),
          mcomp = "tukey",
          fac.names = c("HIBRIDO", "FONTEN"),
          sigT = 0.05,
          sigF = 0.05)
## -----
## Legend:
## FACTOR 1: HIBRIDO
## FACTOR 2: FONTEN
## -----
## 
## 
## Analysis of Variance Table
## -----
##           DF   SS    MS   Fc Pr>Fc
## Block      3  2.55  0.850  1.84 0.15283
## HIBRIDO    3 33.19 11.063 23.99 0.00000
## FONTEN     3 731.71 243.902 528.93 0.00000
## HIBRIDO*FONTEN 9 151.46 16.829 36.50 0.00000
## Residuals   45 20.75  0.461
## Total       63 939.66
## -----
## CV = 5.77 %
## 
## -----
## Shapiro-Wilk normality test
## p-value: 0.129189
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be cons
## -----
## 
## 
##
```

```

## Significant interaction: analyzing the interaction
## -----
## 
## Analyzing HIBRIDO inside of each level of FONTEN
## -----
## 
## Analysis of Variance Table
## -----
##                               DF      SS       MS      Fc   Pr.Fc
## Block                      3  2.55051  0.85017  1.8437 0.1528
## FONTEN                     3 731.70503 243.90168 528.9296      0
## HIBRIDO:FONTEN AmonioA    3  64.21549  21.40516 46.4196      0
## HIBRIDO:FONTEN NitratoA   3  11.60853  3.86951  8.3915 2e-04
## HIBRIDO:FONTEN SulfatoA   3   0.51203  0.17068  0.3701 0.7749
## HIBRIDO:FONTEN Ureia      3 107.23754  35.74585 77.5191      0
## Residuals                  45 20.75054  0.46112
## Total                      63 939.65583
## -----
## 
## 
## 
## HIBRIDO inside of the level AmonioA of FONTEN
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a     3   13.51395
## b     1   11.26163
## c     4    9.702
## d     2    8.085
## -----
## 
## 
## HIBRIDO inside of the level NitratoA of FONTEN
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a     3   8.93592
## ab    4   7.884
## bc    1   7.4466
## c     2   6.57
## -----
## 
## 
## HIBRIDO inside of the level SulfatoA of FONTEN

```

```

##  

## According to the F test, the means of this factor are statistical equal.  

## -----  

##      Levels      Means  

## 1       1    16.7420  

## 2       2    17.1975  

## 3       3    17.1945  

## 4       4    17.1630  

## -----  

##  

##  

##  

##  HIBRIDO  inside of the level Ureia  of FONTEN  

## -----  

## Tukey's test  

## -----  

## Groups Treatments Means  

## a     4    15.237  

## b     2    12.6975  

## c     3    10.1007  

## d     1     8.41725  

## -----  

##  

##  

##  

## Analyzing FONTEN inside of each level of HIBRIDO  

## -----  

## -----  

## Analysis of Variance Table  

## -----  

##  

##          DF      SS      MS      Fc  Pr.Fc  

## Block           3  2.55051  0.85017  1.8437 0.1528  

## HIBRIDO         3 33.18888 11.06296 23.9913      0  

## FONTEN:HIBRIDO NUPEC_1  3 206.29612 68.76537 149.1258      0  

## FONTEN:HIBRIDO NUPEC_2  3 277.34805 92.44935 200.4873      0  

## FONTEN:HIBRIDO NUPEC_3  3 166.03794 55.34598 120.0243      0  

## FONTEN:HIBRIDO NUPEC_4  3 233.48380 77.82793 168.7791      0  

## Residuals        45 20.75054  0.46112  

## Total            63 939.65583  

## -----  

##  

##  

##  

##  

##  FONTEN  inside of the level NUPEC_1  of HIBRIDO  

## -----  

## Tukey's test  

## -----
```

```

## Groups Treatments Means
## a      3   16.742
## b      1   11.26163
## c      4   8.41725
## c      2   7.4466
## -----
## 
## 
## 
##  FONTEN  inside of the level  NUPEC_2  of  HIBRIDO
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      3   17.1975
## b      4   12.6975
## c      1   8.085
## d      2   6.57
## -----
## 
## 
## 
##  FONTEN  inside of the level  NUPEC_3  of  HIBRIDO
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      3   17.1945
## b      1   13.51395
## c      4   10.1007
## c      2   8.93592
## -----
## 
## 
## 
##  FONTEN  inside of the level  NUPEC_4  of  HIBRIDO
## -----
## Tukey's test
## -----
## Groups Treatments Means
## a      3   17.163
## b      4   15.237
## c      1   9.702
## d      2   7.884
## -----
detach(FAT2_CI)

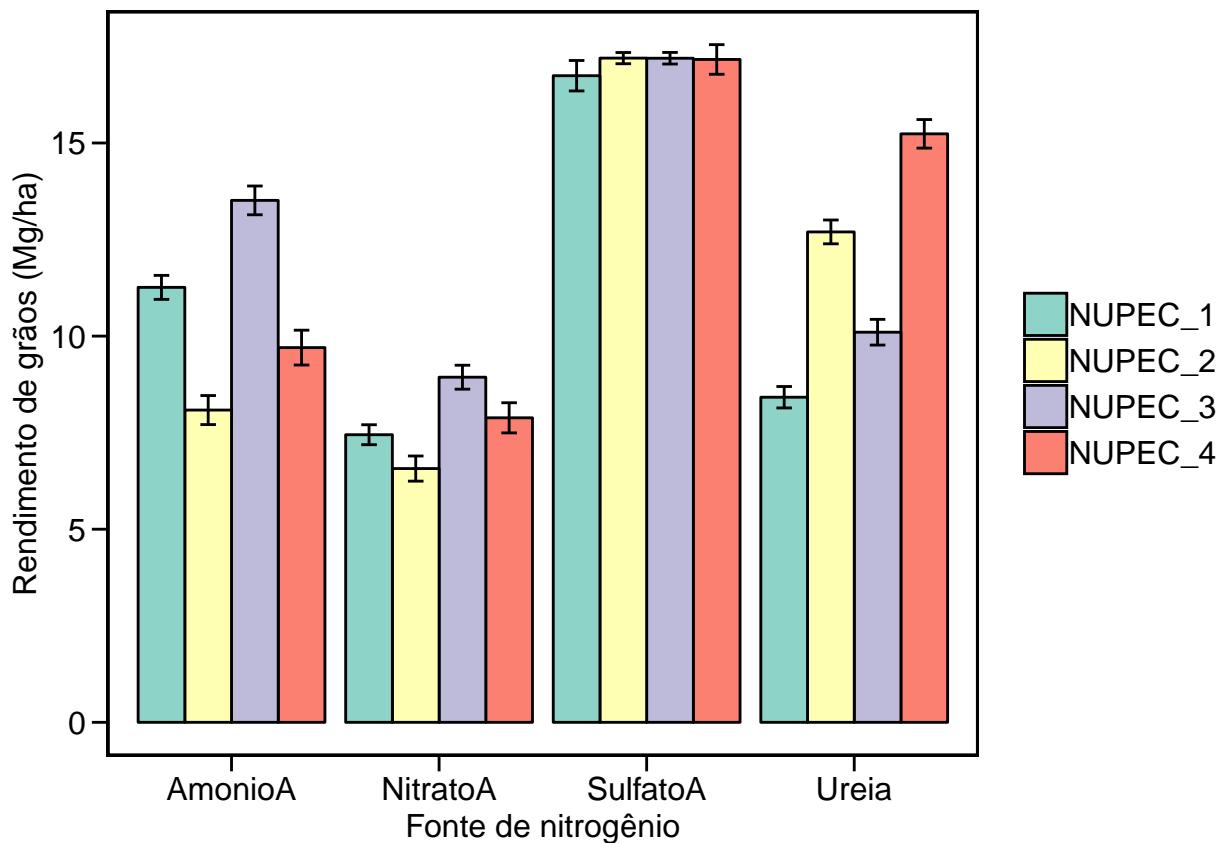
```

Como a interação foi significativa, o próximo passo é a comparação das médias considerando os efeitos da interação. Os valores das médias fixando os níveis de cada fator são, por padrão,

calculados pela função `fat2.rbd()`. A apresentação destas médias em um trabalho científico, por exemplo, pode ser por meio de tabelas, ou gráficos. A função abaixo permite a confecção de gráficos personalizáveis. Basta informar `groupvars = c("FONTEN", "HIBRIDO")` para que a função reconheça que é necessário apresentar os fatores separadamente. Para maiores detalhes, veja `?plot_fatmeans`.

- Plotagem das médias considerando a interação

```
plot_fatmeans(FAT2_CI,
               measurevar = "RG",
               groupvars = c("FONTEN", "HIBRIDO"),
               xlab = "Fonte de nitrogênio",
               ylab = "Rendimento de grãos (Mg/ha)",
               verbose = FALSE,
               palette = "Set3",
               legend.position = "right")
```



Após confeccionar o gráfico, é possível salvá-lo como imagem e inserir as letras correspondentes ao teste Tukey. Por exemplo pode-se considerar letras minúsculas para comparação das médias dos híbridos em cada fonte de nitrogênio e letras maiúsculas para comparar o efeito da fonte de nitrogênio em cada híbrido.

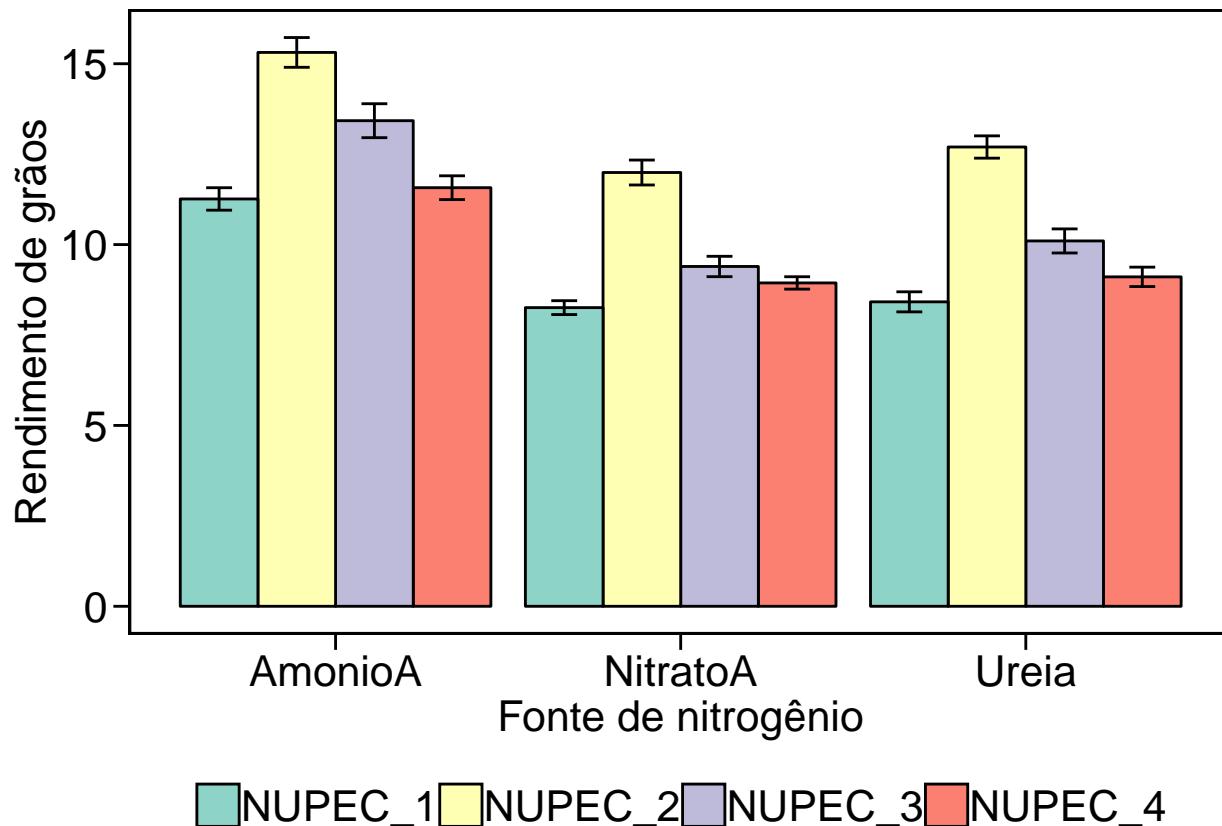


Figure 36: Característica da produção em um experimento bifatorial sem interação significativa (agrupado por fonte de N)

### 2.7.2 Sem interação significativa

Como exemplo de análise de um experimento bifatorial com dois fatores qualitativos sem interação significativa, utilizaremos o conjunto de dados **FAT2\_SI**. Já sabemos que a interação não será significativa neste exemplo. Os dois próximos gráficos nos ajudam a compreender porque.

```
plot_fatmeans(FAT2_SI,
    measurevar = "RG",
    groupvars = c("FONTEN",
                 "HIBRIDO"),
    invert = FALSE,
    verbose = FALSE,
    cex = 16,
    palette = "Set3",
    xlab = "Fonte de nitrogênio",
    ylab = "Rendimento de grãos")

plot_fatmeans(FAT2_SI,
    measurevar = "RG",
```

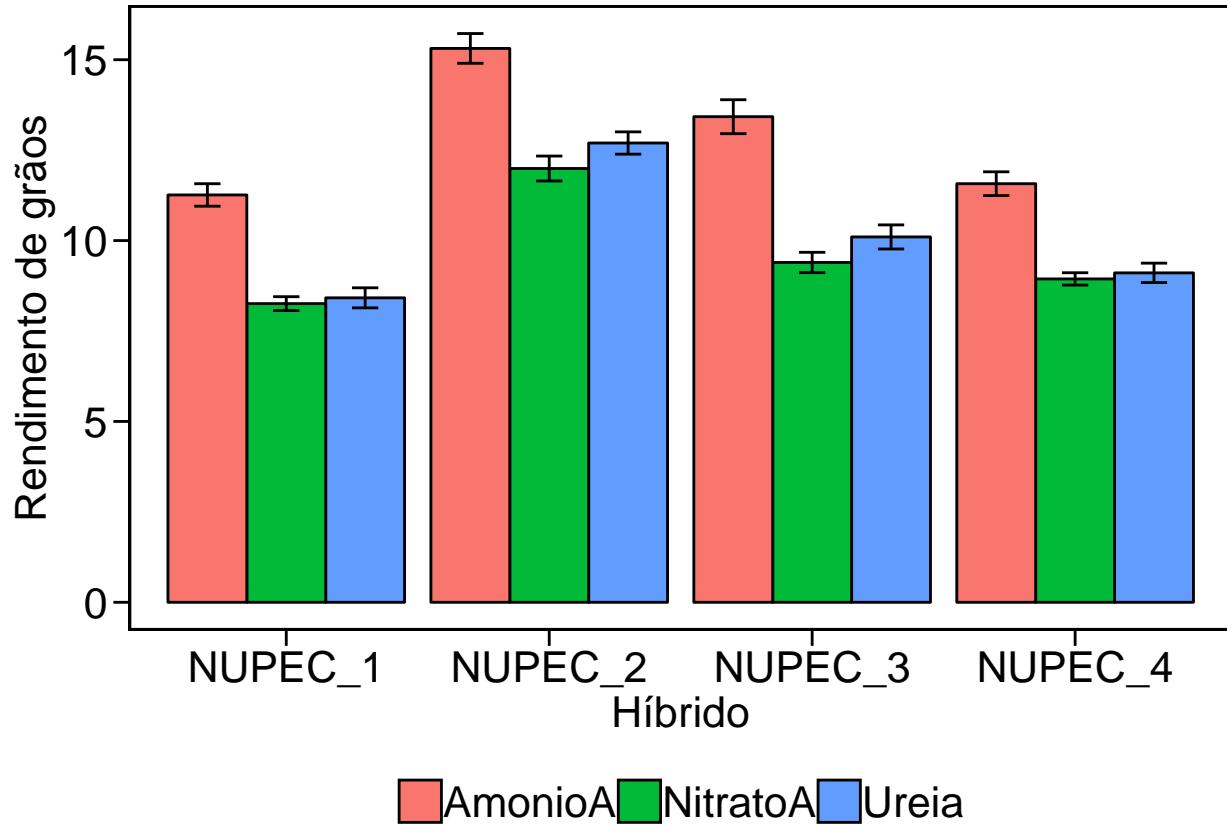


Figure 37: Característica da produção em um experimento bifatorial sem interação significativa (agrupado por fonte híbrido)

```
groupvars = c("FONTEN",
             "HIBRIDO"),
invert = TRUE,
verbose = FALSE,
cex = 16,
palette = "Set3",
xlab = "Híbrido",
ylab = "Rendimento de grãos")
```

É possível identificar que a magnitude da variável resposta, de um fator não é alterada pelo nível do outro fator. Por exemplo, nota-se que o híbrido *NUPEC\_2* parece ter uma média maior que os outros híbridos, independentemente da fonte de nitrogênio utilizada. Do mesmo modo, a fonte de nitrogênio *AmonioA* parece proporcionar maior produtividade, independentemente do híbrido testado. Estas afirmações, no entanto, só poderão ser confirmadas pela análise de variância e posterior comparação de médias.

Maiores informações com relação à sintaxe ggplot2 para gráfico de barras podem ser encontradas aqui.

- Análise de variância

```

attach(FAT2_SI)
fat2.rbd(factor1 = HIBRIDO,
          factor2 = FONTEN,
          block = BLOCO,
          resp = RG,
          quali = c(TRUE, TRUE),
          mcomp = "tukey",
          fac.names = c("HIBRIDO",
                      "FONTEN"),
          sigT = 0.05,
          sigF = 0.05)
## -----
## Legend:
## FACTOR 1: HIBRIDO
## FACTOR 2: FONTEN
## -----
## 
## 
## Analysis of Variance Table
## -----
##           DF   SS    MS   Fc Pr>Fc
## Block      3 0.692 0.231 0.549 0.65236
## HIBRIDO    3 114.046 38.015 90.508 0.00000
## FONTEN     2 99.415 49.707 118.345 0.00000
## HIBRIDO*FONTEN 6 2.362 0.394 0.937 0.48173
## Residuals  33 13.861 0.420
## Total      47 230.376
## -----
## CV = 5.96 %
## 
## 
## Shapiro-Wilk normality test
## p-value: 0.08895984
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be cons
## -----
## 
## No significant interaction: analyzing the simple effect
## -----
## HIBRIDO
## Tukey's test
## -----
## Groups Treatments Means
## a      NUPEC_2      13.33417
## b      NUPEC_3      10.97337
## c      NUPEC_4      9.873333

```

```

##      c    NUPEC_1      9.311875
## -----
## 
## FONTEN
## Tukey's test
## -----
## Groups Treatments Means
## a      AmonioA     12.89291
## b      Ureia      10.08074
## b      NitratoA     9.645917
## -----
detach(FAT2_SI)

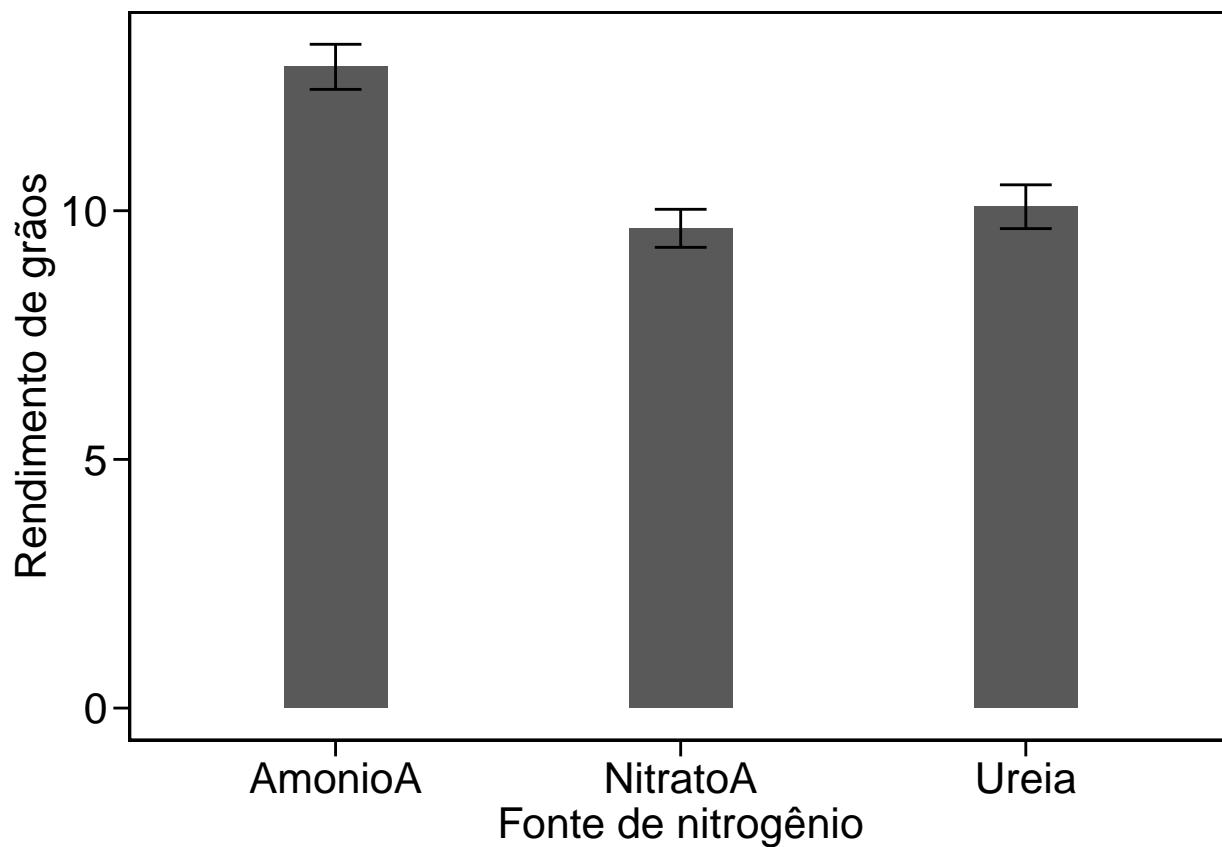
```

Conforme já sabido, a interação HIBRIDO x FONTEN não foi significativa a 5% de probabilidade de erro. Assim, o procedimento a ser realizado é a comparação das médias para os fatores principais apenas. Os dois gráficos abaixo mostram as médias dos fatores principais. Para que a média de um determinado fator seja plotada considerando a média dos níveis do outro fator, basta indicar apenas um fator no argumento `groupvars` da função. No primeiro exemplo, é declarado `groupvars = c("FONTEN")`, indicando que a média deve ser plotada considerando as fontes de nitrogênio. De modo oposto, quando declaramos `groupvars = c("HIBRIDO")`, as médias são plotadas considerando os híbridos.

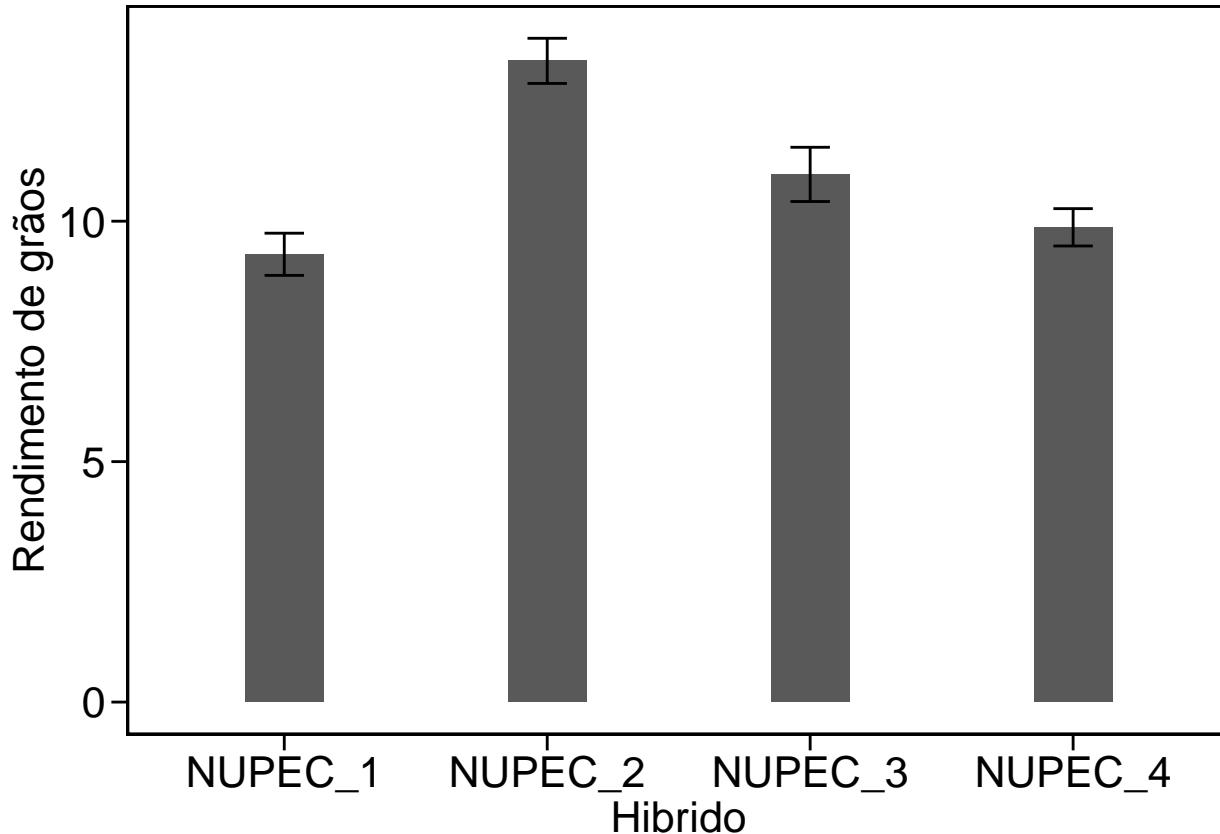
```

plot_fatmeans(FAT2_SI,
               measurevar = "RG",
               groupvars = c("FONTEN"),
               verbose = FALSE,
               cex = 16,
               width.bar = 0.3,
               width.erbar = 0.15,
               xlab = "Fonte de nitrogênio",
               ylab = "Rendimento de grãos")

```



```
plot_fatmeans(FAT2_SI,
               measurevar = "RG",
               groupvars = c("HIBRIDO"),
               verbose = FALSE,
               cex = 16,
               width.bar = 0.3,
               width.erbar = 0.15,
               xlab = "Hibrido",
               ylab = "Rendimento de grãos")
```



## 2.8 Quantitativo vs quantitativo

Neste exemplo, iremos avaliar dados de um experimento que testou dois fatores quantitativos. O conjunto de dados utilizado é o **FAT3**. A análise de variância, neste caso, é realizada da mesma maneira que os exemplos anteriores, o que muda agora é que a análise complementar, em caso de interação significativa será diferente. Por se tratar de dois fatores quantitativos, neste caso, doses de nitrogênio e de potássio na cultura do milho, é de se esperar que, em caso de interação significativa, haja uma combinação de doses ótimas que proporcione a maior magnitude da variável resposta (rendimento de grãos). Assim, uma análise de superfície de resposta é a mais indicada para este tipo de experimento.

Para a análise neste exemplo, utilizaremos a função `supresp()`. Nesta função estão implementadas as seguintes rotinas: análise de variância, ajuste da equação de superfície de resposta, determinação dos pontos críticos, estatísticas de ajuste e análise residual. O procedimento para a análise é simples. Os seguintes argumentos precisam ser declarados: `data`, o conjunto de dados; `factor1` o nome da coluna com o primeiro fator; `factor2` o nome da coluna com o segundo fator; `block` o nome da coluna com os blocos; e `resp` o nome da coluna com a variável resposta que se deseja analizar. Neste exemplo, vamos armazenar os resultados no objeto `sresp`. Este procedimento é necessário para a posterior análise gráfica.

- Análise estatística

```

sresp = supresp(FAT3,
                 factor1 = "DOSEN",
                 factor2 = "DOSEK",
                 block = "BLOCO",
                 resp = "RG")

## -----
## Resultado da análise de variância
## Modelo: Y = m + bk + Ai + Dj + (AD)ij + eijk
## -----
## 
##          Df Sum Sq Mean Sq F value Pr(>F)
## BLOCO      3    158     53   3.621  0.0183 *
## DOSEN      3   65978   21993 1515.063 < 2e-16 ***
## DOSEK       4   11817    2954  203.513 < 2e-16 ***
## DOSEN:DOSEK 12   2363     197   13.563 1.21e-12 ***
## Residuals   57    827      15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## -----
## Teste Shapiro-Wilk para normalidade de resíduos:
## W = 0.946194 p-valor = 0.002100403
## 
## A interação DOSEN*DOSEK foi significativa!
## p-Valor = 1.206e-12
## Verifique as probabilidades abaixo.
## Valores de probabilidade de erro obtidos na ANOVA
## BLOCO = 1.8336e-02
## DOSEN = 2.6968e-54
## DOSEK = 4.9278e-33
## DOSEN*DOSEK = 1.206e-12
## 
## Anova do modelo de superfície de resposta
## RG ~ DOSEN * DOSEK + I(DOSEN^2) + I(DOSEK^2)
## <environment: 0x000000009975b800>
## 
## Analysis of Variance Table
## 
## Response: RG
##          Df Sum Sq Mean Sq F value Pr(>F)
## DOSEN      1   803.8   803.8   3.1304  0.09862 .
## DOSEK      1  1634.6  1634.6   6.3657  0.02436 *
## I(DOSEN^2)  1 12731.3 12731.3 49.5811 5.853e-06 ***
## I(DOSEK^2)  1  1274.5  1274.5   4.9636  0.04280 *
## DOSEN:DOSEK 1     0.3     0.3   0.0011  0.97441

```

```

## Residuals    14  3594.9   256.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## -----
## Resumo dos parâmetros estimados do modelo de superfície de resposta
## 
## Call:
## lm(formula = F2, data = data2)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max
## -26.420 -10.060    1.834    7.189   27.470
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -3.178e+02  7.291e+01 -4.359 0.000654 ***
## DOSEN        1.475e+01  2.181e+00  6.762 9.15e-06 ***
## DOSEK         1.006e+00  5.423e-01  1.855 0.084830 .  
## I(DOSEN^2)   -1.121e-01  1.593e-02 -7.041 5.85e-06 ***
## I(DOSEK^2)   -7.633e-03  3.426e-03 -2.228 0.042797 *  
## DOSEN:DOSEK  1.973e-04  6.043e-03  0.033 0.974411    
## ---    
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 16.02 on 14 degrees of freedom
## Multiple R-squared:  0.8206, Adjusted R-squared:  0.7565 
## F-statistic: 12.81 on 5 and 14 DF,  p-value: 8.241e-05
## 
## -----
## Equação de superfície de resposta
## Y = B0 + B1*A + B2*D + B11*A^2 + B22*D^2 + B12*A*D
## 
## Parâmetros estimados
## B0: -317.8340786
## B1: 14.7502633
## B2: 1.0056943
## B11: -0.1121344
## B22: -0.0076331
## B12: 0.0001973
## 
## Matriz de parametros (A)
## -0.1121344  9.87e-05
## 9.87e-05   -0.0076331
## 
## Inversa da matrix dos parâmetros (invA)
## -8.9179679  -0.1152744

```

```

## -0.1152744      -131.0091259
## -----
## Vetor dos parâmetros B1 e B2 (X)
## B1: 14.7502633
## B2: 1.0056943
## -----
## Equação para estimativa das doses ótimas (A e D)
## -0.5*(invA*X)
## Autovalor 1: -0.007633
## Autovalor 2: -0.112135
## O ponto ótimo é de máxima!
## -----
## O ponto máximo é obtido com as seguintes doses ótimas:
## Dose ótima (DOSEN): 65.8292
## Dose ótima (DOSEK): 66.7277
## -----
## Equação de superfície de resposta ajustada
## A = DOSEN
## D = DOSEK
##  $y = -317.83408 + 14.75026A + 1.00569D - 0.11213A^2 - 0.00763D^2 + 2e-04AD$ 
## -----
## Estatísticas de ajuste
## Consulte a função gof do pacote hidroGOF
## https://cran.r-project.org/web/packages/hydroGOF/hydroGOF.pdf
##         statistics
## ME          0.00
## MAE         11.07
## MSE         179.74
## RMSE        13.41
## NRMSE %    41.30
## PBIAS %    0.00
## RSR          0.41
## rSD          0.91
## NSE          0.82
## mNSE         0.57
## rNSE         0.86
## d            0.95
## md           0.78
## rd           0.96
## cp           0.74
## r            0.91
## R2           0.82
## bR2          0.81
## KGE          0.87
## VE           0.93
## -----

```

```

## Resultados dos valores observados, preditos e residuais
## -----
##   DOSEN DOSEK N      RG predicted residuals
## 1     45    0 4 126.000  118.8555  7.144479
## 2     45    25 4 143.250  139.4492  3.800836
## 3     45    50 4 152.250  150.5014  1.748621
## 4     45    75 4 158.000  152.0122  5.987836
## 5     45   100 4 152.500  143.9815  8.518479
## 6     60    0 4 151.200  163.4977 -12.297721
## 7     60    25 4 171.900  184.1654 -12.265364
## 8     60    50 4 182.700  195.2916 -12.591579
## 9     60    75 4 178.850  196.8764 -18.026364
## 10    60   100 4 162.500  188.9197 -26.419721
## 11    75    0 4 165.000  157.6794  7.320579
## 12    75    25 4 183.750  178.4211  5.328936
## 13    75    50 4 208.250  189.6213  18.628721
## 14    75    75 4 218.750  191.2801  27.469936
## 15    75   100 4 206.250  183.3974  22.852579
## 16    90    0 4 103.320  101.4006  1.919379
## 17    90    25 4 117.465  122.2163 -4.751264
## 18    90    50 4 124.845  133.4905 -8.645479
## 19    90    75 4 128.825  135.2233 -6.398264
## 20    90   100 4 118.090  127.4146 -9.324621
## -----
## Shapiro-Wilk normality test
## p-value: 0.833218
## According to Shapiro-Wilk normality test at 5% of significance, residuals can be considered
## -----

```

Os resultados acima são gerados pela função. Podemos observar que a interação DOSEN x DOSEK foi significativa, assim a metodologia de superfície de resposta foi corretamente utilizada. A equação de superfície considerada é um modelo com termos de segunda ordem, onde a variável resposta é estimada considerando dois preditores, neste caso doses de nitrogênio e doses de potássio. Considerando estes fatores como  $A$  e  $D$  o seguinte modelo é ajustado:  $Y_i = \beta_0 + \beta_1 A_i + \beta_2 D_i + \beta_3 A_i^2 + \beta_4 D_i^2 + \beta_5 A_i D_i + \epsilon_i$ . Os parâmetros estimados estão em *Parâmetros estimados*.

As doses ótimas são estimadas pela seguinte equação:

$$-0.5 \times (\mathbf{A}^{-1} \mathbf{X})$$

, Onde

$$\mathbf{A} = \begin{pmatrix} \beta_3 & \beta_5/2 \\ \beta_5/2 & \beta_4 \end{pmatrix}$$

e

$$\mathbf{X} = \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}$$

Em nosso exemplo,

$$\mathbf{A} = \begin{pmatrix} -0.11213 & 9.865e-05 \\ 9.865e-05 & -0.00763 \end{pmatrix}; \mathbf{A}^{-1} = \begin{pmatrix} -8.91796 & -0.1152 \\ -0.11527 & -131.009 \end{pmatrix}$$

e

$$\mathbf{X} = \begin{pmatrix} 14.7502 \\ 1.00569 \end{pmatrix}$$

Assim

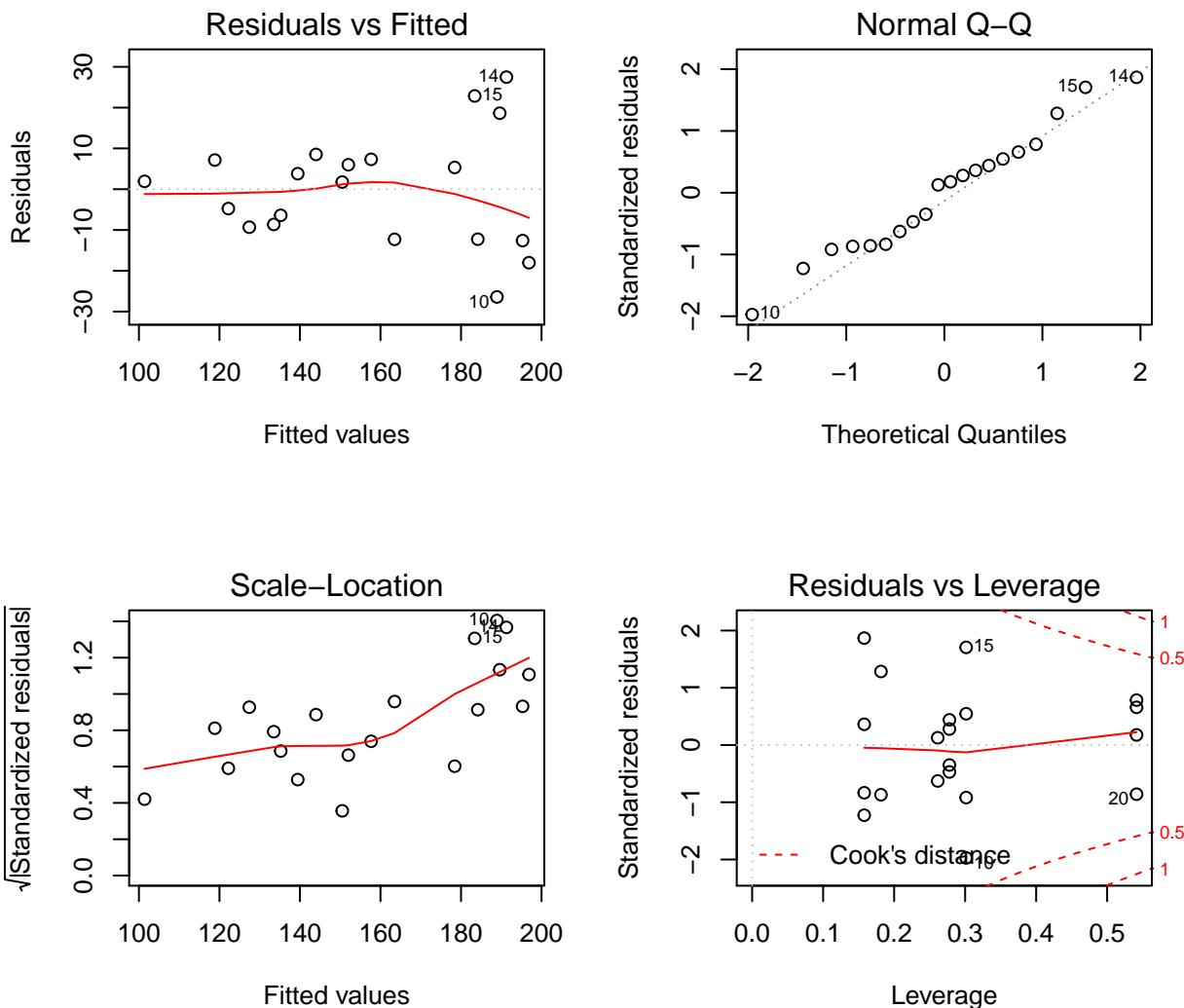
$$-0.5 \times \left[ \begin{pmatrix} -8.91796 & -0.1152 \\ -0.11527 & -131.009 \end{pmatrix} \times \begin{pmatrix} 14.7502 \\ 1.00569 \end{pmatrix} \right] = \begin{pmatrix} 65.8292 \\ 66.7277 \end{pmatrix}$$

- Implementação gráfica da superfície de resposta

A função `plot_supresp()` do pacote `cursoR` proporciona uma análise gráfica poderosa da superfície de resposta gerada pela função `supresp()`. Vimos anteriormente que era importante armazenar os resultados desta última função em um objeto. Na ocasião, armazenamos no objeto `sresp`. Agora é possível utilizá-lo para a confecção dos gráficos. A função `plot_supresp()` tem apenas um argumento obrigatório: `x` que é o objeto criado pela função `supresp()`. Por `default` o tipo do gráfico gerado é do tipo *surface plot*. Outros tipos de gráficos podem ser gerados utilizando o argumento `type`. Se `type = "residuals"`, um gráfico com a análise residual do modelo é gerado. Se `type = "contour"`, um gráfico de contorno é gerado. Os gráficos são coloridos por padrão. No entanto, diversas opções quanto a personalização estão disponíveis. para maiores informações veja `?plot_supresp`.

- Gráfico dos resíduos

```
plot_supresp(sresp,
              type = "residuals")
```



- Gráfico de superfície padrão

```
plot_supresp(sresp)
```

- Gráfico de superfície personalizado

```
plot_supresp(sresp,
  type = "surface",
  theta = 60,
  leg.length = 0.75,
  zlim = c(40, 250),
  leg.dist = -0.12,
  leg.desl = 0.05,
  xlab = "Dose de Nitrogênio",
  ylab = "Dose de Potássio",
  zlab = "Rendimento de grãos",
```

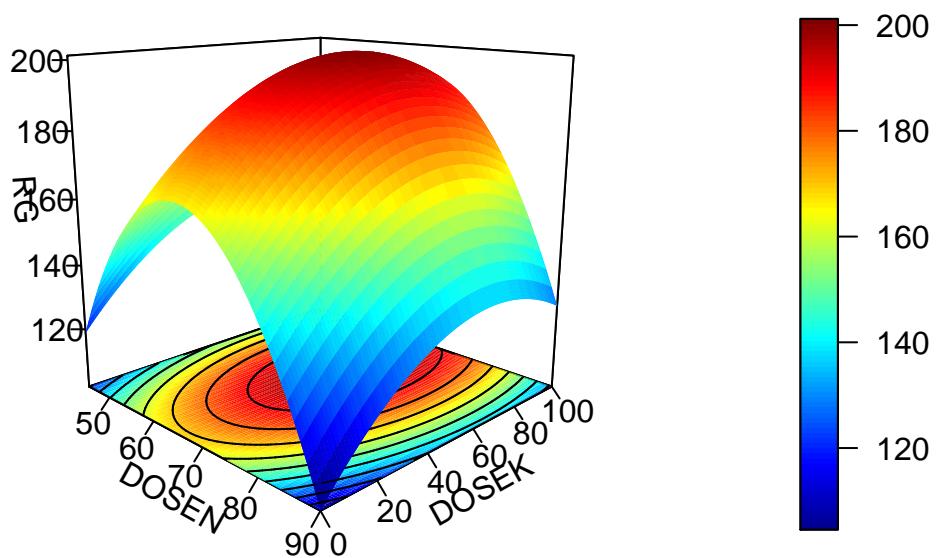
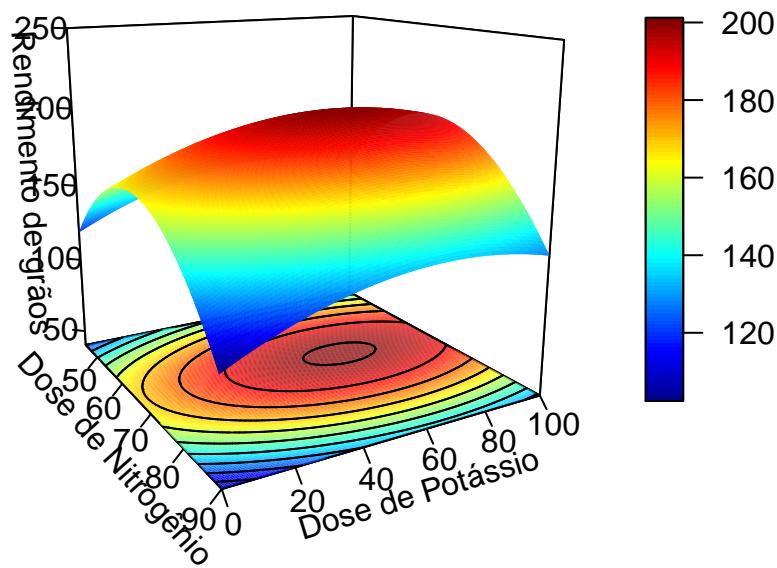


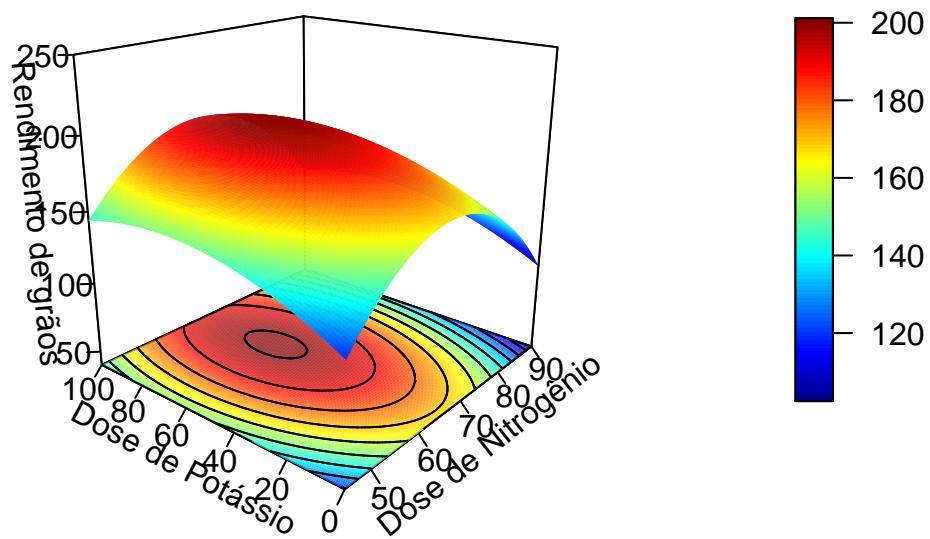
Figure 38: Gráfico de superfície de resposta utilizando a função `plot_supresp()`

```
resfac = 3)
```



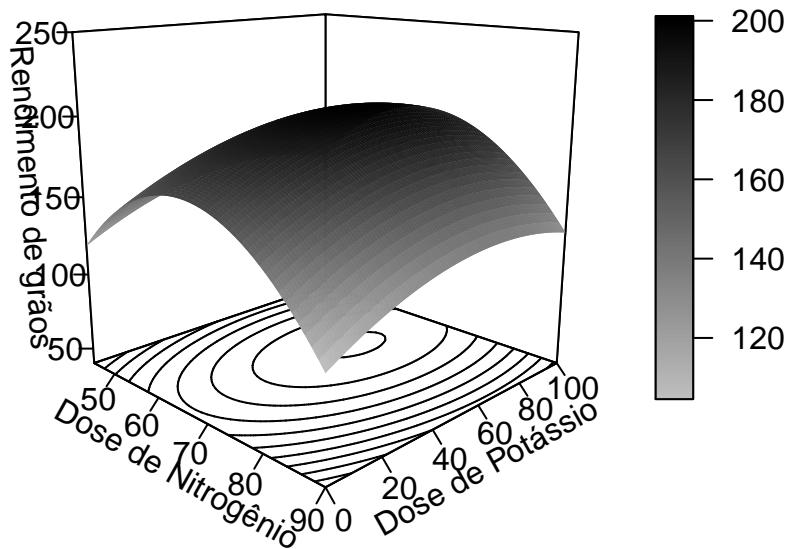
- Outra perspectiva (theta e phi)

```
plot_supresp(sresp,
              type = "surface",
              theta = 310,
              phi = 20,
              leg.length = 0.75,
              zlim = c(40, 250),
              leg.dist = -0.00,
              leg.desl = 0.05,
              xlab = "Dose de Nitrogênio",
              ylab = "Dose de Potássio",
              zlab = "Rendimento de grãos",
              resfac = 3)
```



- escala de cinza (`supresp_col()`)

```
plot_supresp(sresp,
             type = "surface",
             leg.length = 0.75,
             zlim = c(40, 250),
             leg.dist = -0.12,
             leg.desl = 0.05,
             xlab = "Dose de Nitrogênio",
             ylab = "Dose de Potássio",
             zlab = "Rendimento de grãos",
             resfac = 1,
             image = F,
             col = supresp_col())
```



- gráfico de contornos padrão

```
plot_supresp(sresp,
              type = "contour")
```

- gráfico de contornos personalizado

```
plot_supresp(sresp,
              type = "contour",
              xlab = "Dose de Nitrogênio",
              ylab = "Dose de Potássio",
              resfac = 3,
              cex.axis = 1.4,
              cex.lab = 1.4)
```

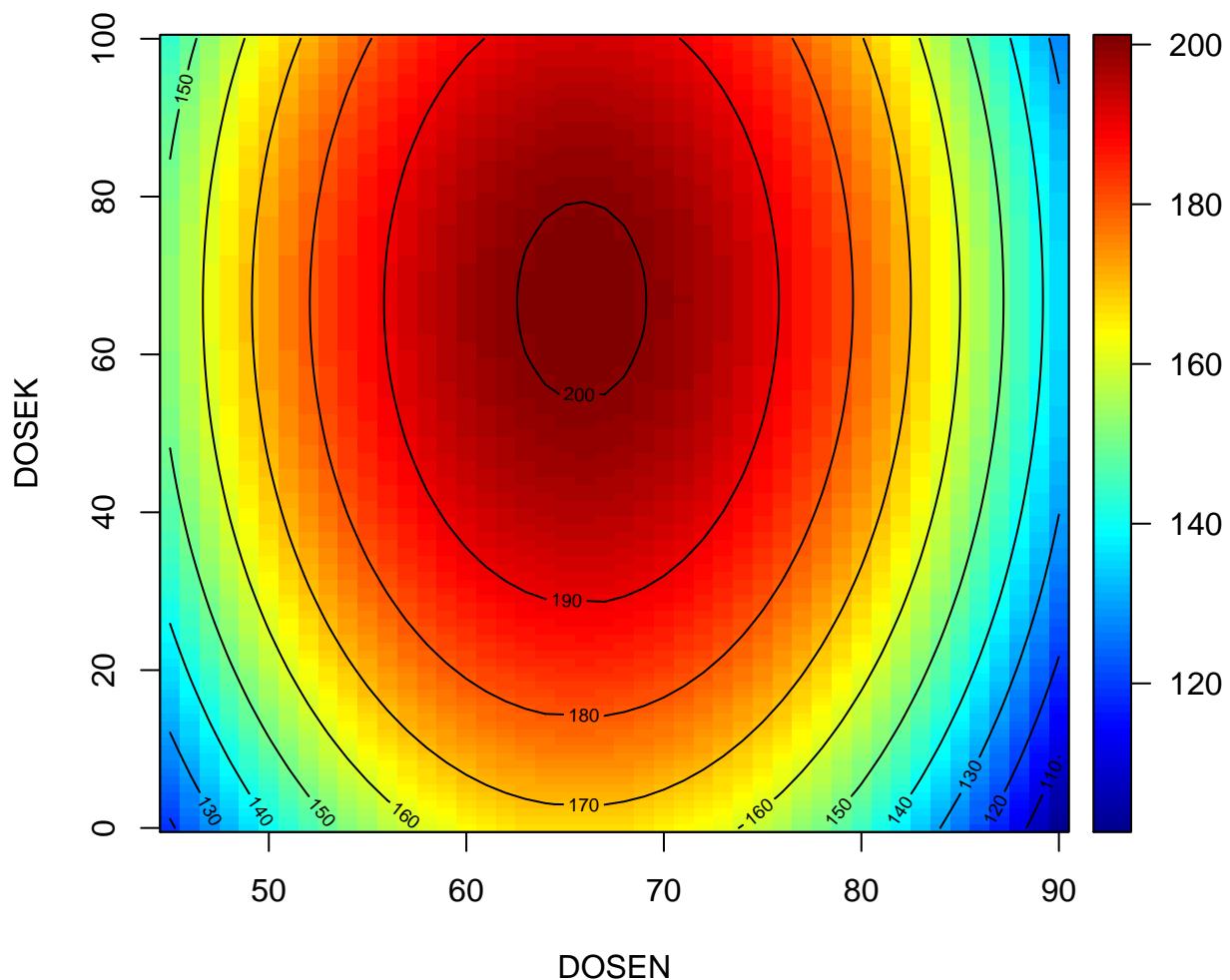
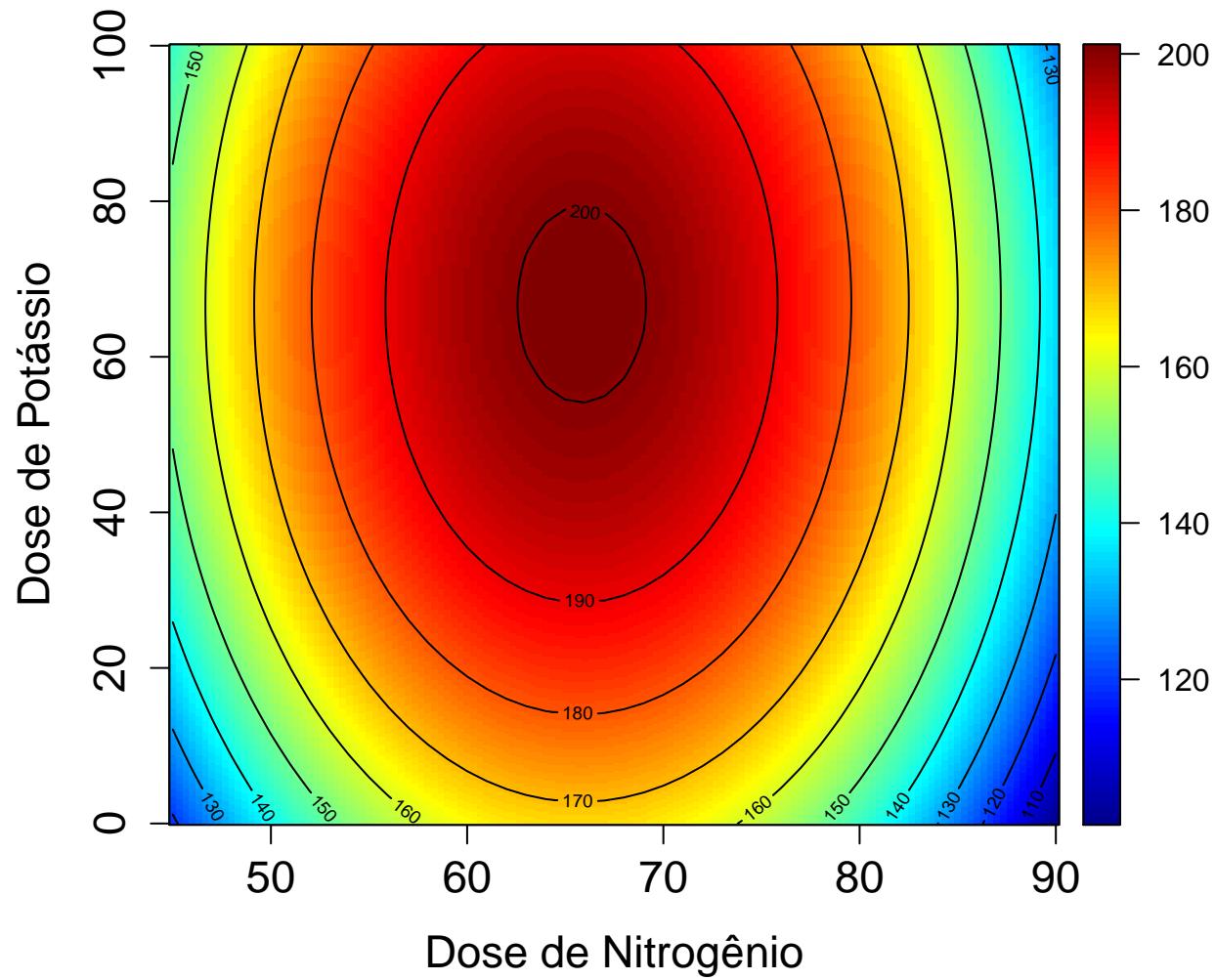
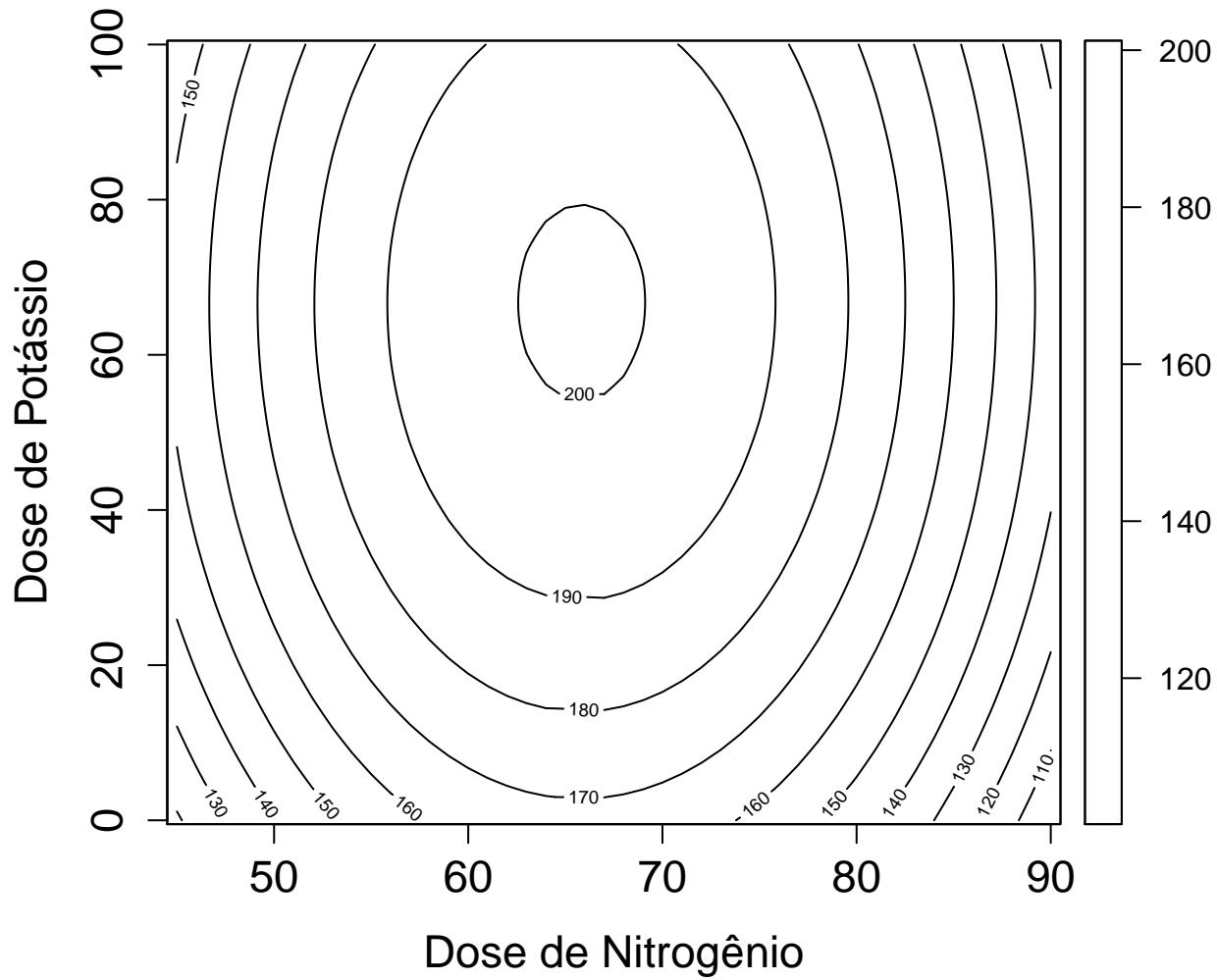


Figure 39: Gráfico de contornos utilizando a função `plot_supresp()`



- gráfico de contornos escala de cinza

```
plot_supresp(sresp,
             type = "contour",
             xlab = "Dose de Nitrogênio",
             ylab = "Dose de Potássio",
             cex.axis = 1.4,
             cex.lab = 1.4,
             col = supresp_col(alpha = 0))
```



## 2.9 Experimentos em parcelas subdivididas

Experimentos fatoriais são úteis devido a possibilidade de se testar dois ou mais fatores em um mesmo experimento. Uma desvantagem deste tipo de experimento é que cada bloco deve receber todos os tratamentos, ou seja, todas as combinações dos níveis dos dois fatores. Assim, o número de parcelas no experimento e consequentemente o tamanho da área experimental crece drasticamente na medida em que são incluídos fatores ou níveis de fatores no experimento. Uma maneira de se contornar isto, é a condução de experimentos em parcelas subdivididas.

Parcelas subdivididas são um caso especial de estrutura de tratamentos fatorial em que um fator é alocado na parcela principal e outro fator é alocado na subparcela. Este tipo de estrutura de tratamentos pode ser utilizada quando um fator é de difícil instalação em pequenas parcelas, como por exemplo, a semeadura mecanizada ou um sistema de irrigação,

e o segundo fator pode ser alocado em parcelas mais pequenas, como um doses de nitrogênio, por exemplo.

Diferentemente do modelo do fatorial tradicional, o modelo estatístico para análise de experimentos em parcelas subdivididas conta com mais uma fonte de variação. Vamos considerar como exemplo, um experimento que avaliou a influencia de dois fatores, digamos  $\alpha$  e  $\tau$ , em uma determinada variável resposta, agora, conduzido em parcelas subdivididas, onde o fator  $\alpha$  foi alocado na parcela principal e o fator  $\tau$  alocado na subparcela. O modelo estatístico considerado neste tipo de experimento é:  $y_{ijk} = \mu + \alpha_i + \beta_k + \eta_{ik} + \tau_j + (\alpha\tau)_{ij} + \varepsilon_{ijk}$ , onde  $y_{ijk}$  é a variável resposta observada;  $\mu$  é a média geral;  $\alpha_i$  é o efeito do  $i$ -ésimo nível de  $\alpha$ ;  $\beta_k$  é o efeito do bloco  $k$ ;  $\eta_{ik}$  é o erro de parcela, mais conhecido como erro a; assumido  $\varepsilon_{ijk} \stackrel{iid}{\sim} N(0, \sigma^2_\eta)$ ;  $\tau_j$  é o efeito do  $j$ -ésimo nível de  $\tau$ ;  $(\alpha\tau)_{ij}$  é o efeito da interação do  $i$ -ésimo nível de  $\alpha$  com o  $j$ -ésimo nível de  $\tau$ ; e  $\varepsilon_{ijk}$  é o erro da subparcela, mais conhecido como erro b, assumindo  $\varepsilon_{ijk} \stackrel{iid}{\sim} N(0, \sigma^2)$ .

```

attach(FAT2_SI)
split2.rbd(factor1 = HIBRIDO,
            factor2 = FONTEN,
            block = BLOCO,
            resp = RG,
            quali = c(TRUE, TRUE),
            mcomp = "tukey",
            fac.names = c("HIBRIDO", "FONTEN"),
            sigT = 0.05,
            sigF = 0.05)

## -----
## Legend:
## FACTOR 1 (plot): HIBRIDO
## FACTOR 2 (split-plot): FONTEN
## -----
## 
## -----
## $`Analysis of Variance Table`  

##          DF      SS      MS      Fc Pr(>Fc)  

## HIBRIDO        3 114.046 38.015 158.360 <2e-16 ***  

## Block          3   0.692  0.231   0.961  0.4523  

## Error a         9   2.161  0.240  

## FONTEN         2  99.415 49.707 101.962 <2e-16 ***  

## HIBRIDO*FONTEN 6   2.362  0.394   0.808  0.5741  

## Error b        24  11.700  0.488  

## Total          47 230.376  

## ---  

## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## -----
## CV 1 = 4.506098 %
## CV 2 = 6.421463 %

```

```

## 
## No significant interaction: analyzing the simple effects
## -----
## HIBRIDO
## Tukey's test
## -----
## Groups Treatments Means
## a      NUPEC_2      13.33417
## b      NUPEC_3      10.97337
## c      NUPEC_4      9.873333
## c     NUPEC_1      9.311875
## -----
## 
## 
## FONTEN
## Tukey's test
## -----
## Groups Treatments Means
## a      AmonioA     12.89291
## b      Ureia       10.08074
## b      NitratoA    9.645917
## -----
detach(FAT2_SI)

```

### 3 Análise de regressão

#### 3.1 Regressão Linear

A análise de regressão tem como objetivo verificar como uma variável independente influencia a resposta de uma variável dependente. A análise de regressão é amplamente utilizada em ciências agrárias e pode ser dividida em simples ou múltipla . Na regressão simples, apenas uma variável dependente é declarada no modelo:

$$Y_i = \beta_0 + \beta_1 x + \varepsilon_i$$

Onde  $Y_i$  é a variável dependente,  $x$  é a variável independente,  $\beta_0$  é o intercepto,  $\beta_1$  é a inclinação da reta e  $\varepsilon$  é o erro. Na regressão linear múltipla, mais de uma variável dependente é declarada no modelo:

$$Y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k + \varepsilon_i$$

Onde  $Y_i$  é a variável dependente,  $x_1, x_2, \dots, x_k$  são as variáveis independentes,  $\beta_0, \beta_1, \beta_2, \dots, \beta_k$  são os parâmetros da regressão e  $\varepsilon$  é o erro. Então, uma regressão por ser descrita genericamente por (Draper and Smith 1998)

$$Y_i = f(x) + \varepsilon_i$$

Onde  $Y_i$  é a variável dependente,  $f(x)$  é a função resposta do modelo e  $\varepsilon$  é o erro.

### 3.1.1 Estimação

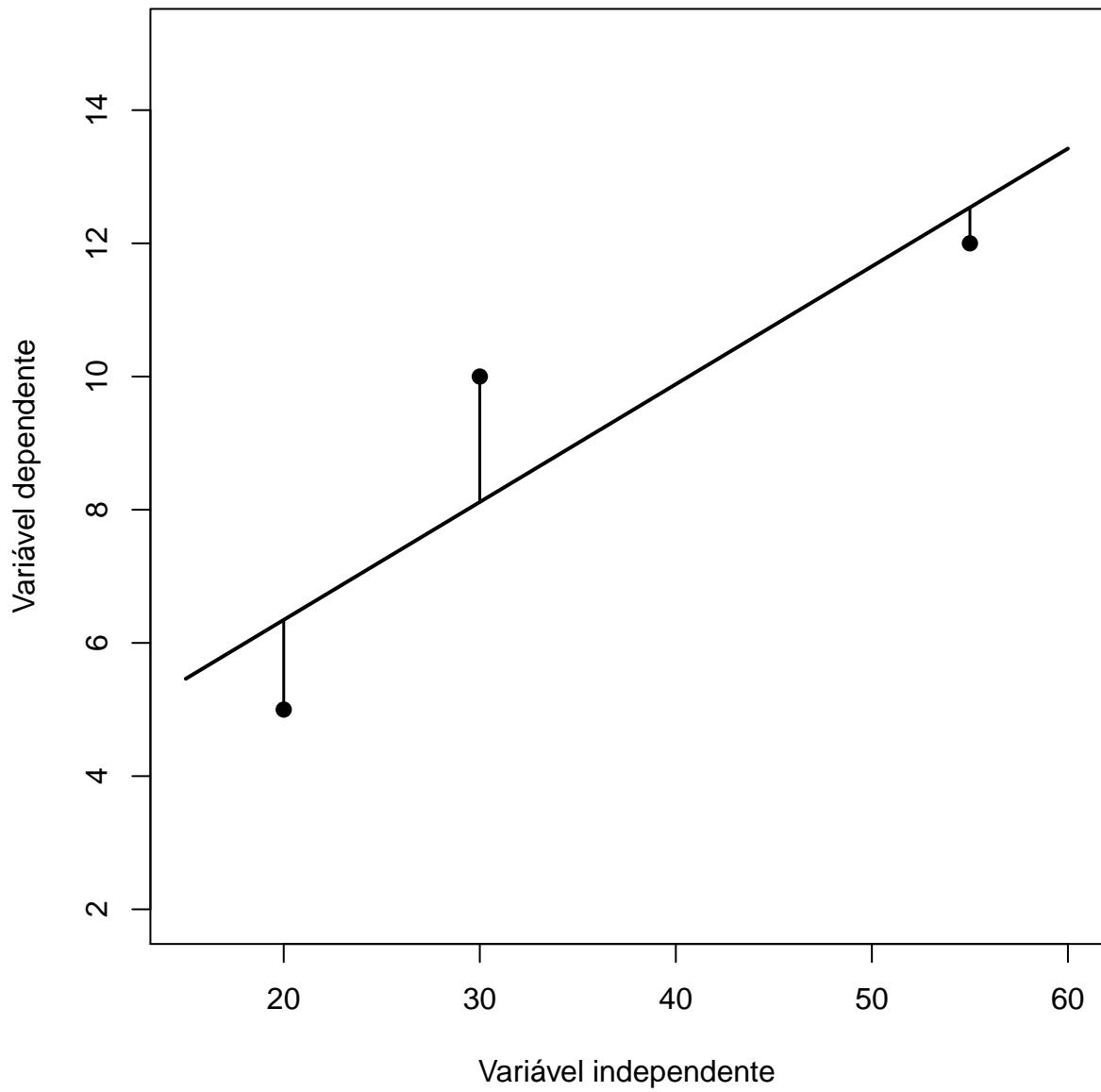
Uma das formas de estimar os parâmetros em regressão é minimizando os erros. Os erros são a diferença entre o valor estimado pela função resposta ( $f(x)$ ) e o valor observado ( $Y_i$ ). Então, devemos encontrar valores para  $\beta$  que minimizem estes erros, representados pela distância entre a reta estimada e os valores observados.

```
model=function(x,b0,b1){
  b0+b1*x} ## modelo genérico

X=c(20,55,30)
Y=c(5,12,10)
mod6=lm(Y~X)
Y.1=mod6$fitted.values ## valores estimados

b0=mod6$coefficients[1]
b1=mod6$coefficients[2]

## Gráfico
plot(Y~X, xlab="Variável independente", ylab="Variável dependente",
      ylim=c(2,15), xlim=c(15,60), pch=16, cex=1.2)
curve(model(x,b0=b0,b1=b1), add=TRUE, lwd=2)
segments(x0=20,y0=5,x1=20,y1=Y.1[1],lwd=1.5) ## erro 1
segments(x0=30,y0=10,x1=30,y1=Y.1[3],lwd=1.5) ## erro 2
segments(x0=55,y0=12,x1=55,y1=Y.1[2],lwd=1.5) ## erro 3
```



Na figura acima, reta é traçada de modo que as distâncias entre ela e os pontos seja mínimo. Essa distância é obtida, pelo método dos mínimos quadrados , minimizando a soma de quadrados dos resíduos (uma vez que a soma dos resíduos é igual a zero).

$$S = \varepsilon' \varepsilon = \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 = 0$$

Para encontrar os valores dos parâmetros que minimiza essa soma de quadrados basta resolver o sistema de equações normais , obtida após derivar  $S$  em relação aos parâmetros. A resolução

deste sistema fornece estimativas não viesadas dos parâmetros  $\hat{\beta}$ .

$$\begin{aligned} \mathbf{X}'\mathbf{X}\beta &= \mathbf{X}'\mathbf{Y} \\ \hat{\beta} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \end{aligned}$$

Percebe-se que as estimativas dos parâmetros não tem nenhuma relação com os pressupostos de normalidade, homocedasticidade e independência dos resíduos. Porém, cumprir pressupostos é importante para testar hipóteses e construir intervalos de confiança. Outro aspecto importante na estimação é a necessidade da matriz  $\mathbf{X}'\mathbf{X}$  ser não singular, pois assim é possível inverter essa matriz e resolver o sistema de equações normais obtendo parâmetros únicos. O sistema de equações normais também pode ser resolvido utilizando a inversa generalizada. Neste último caso, os valores dos parâmetros que resolvem o sistema de equações não são únicos. Para maiores detalhes sobre como estimar os parâmetros de regressões lineares, ver Draper and Smith (1998), Kutner et al. (2005) e Rencher and Schaalje (2008).

Citamos acima que uma matriz  $\mathbf{X}'\mathbf{X}$  não singular é necessário para obtermos os parâmetros da nossa regressão. A não singularidade da matriz está relacionada com quanto as variáveis independentes estão correlacionadas. Quando as variáveis independentes estão aproximadamente (ou perfeitamente, o que é praticamente impossível) relacionadas dizemos que há elevada **multicolinearidade**. O principal problema da multicolinearidade está relacionado com as estimativas dos parâmetros. Quando ela é elevada, um conjunto de funções resposta minimiza os erros e prediz, com precisão, os valores observados.

Quando há multicolinearidade, é possível resolver o sistema de equações normais utilizando a inversa generalizada de Moore-Penrose, através da função `ginv()` do pacote *MASS*. Utilizando a inversa generalizada minimiza-se a soma dos quadrados, porém não garante-se que os parâmetros sejam únicos. Então, qualquer inferências sobre como as variáveis se relacionam passa a ser duvidosa (Kutner et al. 2005).

Para demonstrar como a multicolinearidade afeta a estimativa dos parâmetros, utilizamos um exemplo hipotético de Kutner et al. (2005). Execute a programação em casa e veja os resultados.

```
X1=c(2,8,6,10)
X2=c(6,9,8,10)
cor(X1,X2) ## correlação entre as variáveis independentes

#### Minimizando a soma de quadrados
require (nls2)
resultados=data.frame(matrix(ncol=4,nrow = 20))
names(resultados)=c("b0","b1","b2","sigma")

for(i in 1:20){
  Y=c(23,83,63,103)
  X1=c(2,8,6,10)
  X2=c(6,9,8,10)
  grid = expand.grid(list(
    b0 = seq(-i,i, by=0.1),
    b1 = seq(-i,i, by=0.1),
    b2 = seq(-i,i, by=0.1),
    sigma = seq(0.1,1, by=0.1)
  ))
  resultados[i,]=nls2(Y~b0+b1*X1+b2*X2+sigma*X1*X2, start=list(b0=0, b1=0, b2=0, sigma=1), algorithm="port")
}
```

```

b1 = seq(-i, i, by = 0.1),
b2 = seq(-i, i, by = 0.1)
)) ## Armazenando um conjunto de valores que quero dar aos parâmetros

Resp = nls2(Y~b0+b1*X1+b2*X2,
start = grid,
algorithm = "brute-force")

b0=summary(Resp)$coefficients[1,1]
b1=summary(Resp)$coefficients[2,1]
b2=summary(Resp)$coefficients[3,1]
sigma=summary(Resp)$sigma
;
resultados$b0[i]=b0
resultados$b1[i]=b1
resultados$b2[i]=b2
resultados$sigma[i]=sigma
}
resultados

```

Quando há multicolinearidade , conjuntos de diferentes parâmetros resolvem o sistema de equações normais e minimizam a soma de quadrados. Por isso, relacionar a resposta da variável dependente em função das variáveis independentes passa a ser impossível. Por isso, por exemplo, que a multicolinearidade é importante na análise de trilha. A relação entre as variáveis na análise de trilha é determinada com base no valor dos coeficientes de trilha, que nada mais são do que parâmetros de uma regressão múltipla.

Não entraremos em detalhe sobre a multicolinearidade nesta parte do curso, e deixaremos o diagnóstico para a seção de análise multivariada. Ressaltamos também que a capacidade preditiva dos modelos, ou seja, a capacidade do modelo em estimar determinado valor observado com precisão, não é afetada pela multicolinearidade.

### 3.1.2 Ajustando regressões com a função lm()

A função `lm()` é utilizada para ajustar regressões lineares simples e múltipla. Os argumentos mais importantes desta função são a `formula`, onde indicamos a função resposta; e `data`, onde indicamos o banco de dados. Vamos utilizar um exemplo simples retirado de Schenider, Schenider, and Souza (2009):

```

require(readxl)
Reg_F1=read_excel("D:/Desktop/final/Reg_F1.xlsx")
Reg_F1

## # A tibble: 13 x 4
##       Y     X1     X2     X3
##   <dbl> <dbl> <dbl> <dbl>
## 1     1     15      5      3     12
## 2     2     10      4      2     10
## 3     3     12      6      4     11
## 4     4     14      8      5     13
## 5     5     16      9      6     14
## 6     6     18      11     7     15
## 7     7     20      13     8     16
## 8     8     22      15     9     17
## 9     9     24      17     10    18
## 10   10     26      19     11    19
## 11   11     28      21     12    20
## 12   12     30      23     13    21
## 13   13     32      25     14    22

```

```

##   2    27     8     2     5
##   3    10    12    14     8
##   4     2     2     4    14
##   5    39    16     5     7
##   6    32    10     2     9
##   7    35    14     5     3
##   8    14     9    11     2
##   9     8     6    10     3
##  10    18    10     8    15
##  11     3     7     9    13
##  12     7     6     8    16
##  13    24    12    10    23

mod7=lm(Y~X1+X2+X3,data=Reg_F1)
mod7.1=lm(Y~1,data=Reg_F1)
anova(mod7.1,mod7) ## Verificar a significância do modelo

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ 1
## Model 2: Y ~ X1 + X2 + X3
##   Res.Df   RSS Df Sum of Sq    F    Pr(>F)
## 1     12 1834.00
## 2      9  97.47  3    1736.5 53.449 4.653e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Através da função `anova(mod7.1,mod7)` pode-se verificar se o modelo é ou não significativo. A hipótese  $H_0 = 0$  é rejeitada e conclui-se que o modelo explica o comportamento da variável resposta. Através da função `summary()` obtém-se o resultado do teste t para os parâmetros do modelo. A hipótese testada neste caso é  $H_0 : \beta = \mathbf{0}$  vs  $H_A : \beta \neq \mathbf{0}$ . Por fim, a função `anova()` retorna um teste F que possibilita verificar a contribuição de cada parâmetro em explicar a variabilidade da variável resposta.

```

summary(mod7)

##
## Call:
## lm(formula = Y ~ X1 + X2 + X3, data = Reg_F1)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -5.4062 -2.2378 -0.3066  1.6321  4.8468
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.86871   3.32628   2.366   0.0422 *
## X1          2.72881   0.25198 10.830 1.84e-06 ***

```

```

## X2          -1.92182   0.25540  -7.525 3.60e-05 ***
## X3          -0.09752   0.15686  -0.622   0.5496
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.291 on 9 degrees of freedom
## Multiple R-squared:  0.9469, Adjusted R-squared:  0.9291
## F-statistic: 53.45 on 3 and 9 DF,  p-value: 4.653e-06

anova(mod7) ## Quanto cada variável "contribui" para a significância

## Analysis of Variance Table
##
## Response: Y
##             Df  Sum Sq Mean Sq  F value    Pr(>F)
## X1          1 1102.77 1102.77 101.8264 3.318e-06 ***
## X2          1  629.58  629.58  58.1331 3.243e-05 ***
## X3          1     4.19     4.19   0.3865   0.5496
## Residuals  9   97.47   10.83
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Entre os parâmetros do modelo, apenas  $\hat{\beta}_3$  não foi significativo, indicando que não há necessidade dele ser incluído no modelo. Através do teste F é possível verificar qual o modelo (com ou sem  $\beta_3$ ) é o mais parcimonioso. O teste F é dado por:

$$F_{calc} = \frac{SQ_{Erro}(\Omega) - SQ_{Erro}(\omega)/GL_{Erro}(\Omega) - GL_{Erro}(\omega)}{QM_{Erro}(\omega)}$$

Onde,  $SQ_{Erro}(\Omega)$  e  $SQ_{Erro}(\omega)$  são as somas de quadrados dos resíduos nos modelos completo e reduzido, respectivamente;  $GL_{Erro}(\omega)$  e  $GL_{Erro}(\Omega)$  são os graus de liberdade do resíduo do modelo completo e reduzido, respectivamente; e  $QM_{Erro}(\omega)$  é o quadrado médio do resíduo do modelo completo. Podemos realizar o teste F utilizando a função `anova()`:

```

mod7=lm(Y~X1+X2+X3,data=Reg_F1)
mod8=lm(Y~X1+X2,data=Reg_F1)
anova(mod8,mod7)

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ X1 + X2
## Model 2: Y ~ X1 + X2 + X3
##   Res.Df      RSS Df Sum of Sq    F Pr(>F)
## 1      10 101.655
## 2       9  97.469  1     4.186 0.3865 0.5496

```

Como ambos modelos são estatisticamente iguais, opta-se pelo modelo reduzido. O modelo que melhor se ajustou aos dados foi  $Y = 7.73 + 2.76X_1 - 1.94X_2$ , com  $R^2$  superior a 90%.

```

anova(mod8)

## Analysis of Variance Table
##
## Response: Y
##          Df  Sum Sq Mean Sq F value    Pr(>F)
## X1        1 1102.77 1102.77 108.481 1.093e-06 ***
## X2        1  629.58  629.58  61.933 1.359e-05 ***
## Residuals 10  101.66   10.17
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

summary(mod8)

##
## Call:
## lm(formula = Y ~ X1 + X2, data = Reg_F1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.6055 -2.1927 -0.6734  2.0716  4.0917
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.7266    2.6866   2.504  0.0312 *
## X1          2.7596    0.2394  11.530 4.25e-07 ***
## X2         -1.9376    0.2462  -7.870 1.36e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.188 on 10 degrees of freedom
## Multiple R-squared:  0.9446, Adjusted R-squared:  0.9335
## F-statistic: 85.21 on 2 and 10 DF,  p-value: 5.232e-07

```

### 3.1.3 Seleção de variáveis

Foi mostrado brevemente um exemplo de como ajustar e selecionar variáveis. Porém, no exemplo apresentado, utilizou-se somente três variáveis. No entanto, quando há um elevado número de variáveis, selecioná-las torna-se um trabalho um pouco mais complexo. Nestes casos é necessário utilizar algoritmos de seleção. Os mais comuns são o *Forward*, *Backward* e *Stepwise*.

#### Forward

No método *forward* parte-se de um modelo com uma variável independente, que é aquela que possui maior correlação amostral com a variável dependente. Posteriormente, realiza-se o teste F para verificar se ela é realmente significativa. A segunda variável independente com maior correlação amostral com a variável dependente é adicionada ao modelo, e um

teste F parcial verifica a significância. As variáveis são adicionadas enquanto o F parcial for significativo.

```
cor(Reg_F1$X1,Reg_F1$Y) ## maior correlação
## [1] 0.7754301
cor(Reg_F1$X2,Reg_F1$Y) ## segunda maior correlação
## [1] -0.4558018
cor(Reg_F1$X3,Reg_F1$Y) ## terceira maior correlação
## [1] -0.2460537
mod_for1=lm(Y~1,data=Reg_F1)
mod_for2=lm(Y~X1,data=Reg_F1) ## Adiciona X1
mod_for3=lm(Y~X1+X2,data=Reg_F1) ## Adiciona X2
mod_for4=lm(Y~X1+X2+X3,data=Reg_F1) ## Adiciona X3
anova(mod_for1,mod_for2,mod_for3,mod_for4) ## Seleciona o modelo

## Analysis of Variance Table
##
## Model 1: Y ~ 1
## Model 2: Y ~ X1
## Model 3: Y ~ X1 + X2
## Model 4: Y ~ X1 + X2 + X3
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1     12 1834.00
## 2     11  731.23  1   1102.77 101.8264 3.318e-06 ***
## 3     10  101.66  1     629.58  58.1331 3.243e-05 ***
## 4      9   97.47  1      4.19   0.3865    0.5496
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## Backward

No método *backward* parte-se do modelo completo. As variáveis candidatas a serem eliminadas são determinadas através do teste F parcial, como se elas fossem (hipoteticamente) as últimas a serem incluídas no modelo.

```
## F parcial para X3
mod_back.x3.1=lm(Y~X1+X2,data=Reg_F1)
mod_back.x3=lm(Y~X1+X2+X3,data=Reg_F1)
anova(mod_back.x3.1,mod_back.x3) ## Menor F parcial

## Analysis of Variance Table
##
## Model 1: Y ~ X1 + X2
## Model 2: Y ~ X1 + X2 + X3
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
```

```

## 1      10 101.655
## 2      9  97.469  1      4.186 0.3865 0.5496

## F parcial para X2
mod_back.x2.1=lm(Y~X1+X3,data=Reg_F1)
mod_back.x2=lm(Y~X1+X2+X3,data=Reg_F1)
anova(mod_back.x2.1,mod_back.x2)

## Analysis of Variance Table
##
## Model 1: Y ~ X1 + X3
## Model 2: Y ~ X1 + X2 + X3
##   Res.Df   RSS Df Sum of Sq    F    Pr(>F)
## 1     10 710.70
## 2      9  97.47  1    613.23 56.624 3.598e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## F parcial para X1
mod_back.x1.1=lm(Y~X2+X3,data=Reg_F1)
mod_back.x1=lm(Y~X1+X2+X3,data=Reg_F1)
anova(mod_back.x1.1,mod_back.x1) ## Maior F parcial

## Analysis of Variance Table
##
## Model 1: Y ~ X2 + X3
## Model 2: Y ~ X1 + X2 + X3
##   Res.Df   RSS Df Sum of Sq    F    Pr(>F)
## 1     10 1367.60
## 2      9  97.47  1    1270.1 117.28 1.836e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

### Elimina a 3, depois a 2 e depois a 1
mod_back1=lm(Y~X1+X2+X3,data=Reg_F1)
mod_back2=lm(Y~X1+X2,data=Reg_F1)
anova(mod_back2,mod_back1) ## elimina X3

## Analysis of Variance Table
##
## Model 1: Y ~ X1 + X2
## Model 2: Y ~ X1 + X2 + X3
##   Res.Df   RSS Df Sum of Sq    F    Pr(>F)
## 1     10 101.655
## 2      9  97.469  1      4.186 0.3865 0.5496

mod_back2=lm(Y~X1+X2,data=Reg_F1)
mod_back3=lm(Y~X1,data=Reg_F1)
anova(mod_back3,mod_back2) ## não elimina X2 e seleciona

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ X1
## Model 2: Y ~ X1 + X2
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1     11 731.23
## 2     10 101.66  1    629.58 61.933 1.359e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

### Stepwise

O método *stepwise* utiliza características dos métodos *forward* e *backward*. Neste método parte-se de um modelo composto pela variável com maior correlação amostral. A cada variável adicionada por *forward*, é realizado um *backward* para retirar uma das variáveis previamente adicionadas.

```

## Passo 1
mod_step1.0=lm(Y~1,data=Reg_F1)
mod_step1=lm(Y~X1,data=Reg_F1)
anova(mod_step1.0,mod_step1) ## adiciona X1

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ 1
## Model 2: Y ~ X1
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1     12 1834.00
## 2     11 731.23  1    1102.8 16.589 0.001842 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Passo 2
mod_step2=lm(Y~X1,data=Reg_F1)
mod_step2.1=lm(Y~X1+X2,data=Reg_F1)
anova(mod_step2,mod_step2.1) ## adiciona X2

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ X1
## Model 2: Y ~ X1 + X2
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1     11 731.23
## 2     10 101.66  1    629.58 61.933 1.359e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
mod_step2.2=lm(Y~X2,data=Reg_F1)
anova(mod_step2.2,mod_step2.1) ## mantém X1 no modelo

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ X2
## Model 2: Y ~ X1 + X2
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1     11 1452.98
## 2     10 101.66  1    1351.3 132.93 4.251e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Passo 3
mod_step3=lm(Y~X1+X2,data=Reg_F1)
mod_step3.1=lm(Y~X1+X2+X3,data=Reg_F1)
anova(mod_step3,mod_step3.1) ## não adiciona X3, seleciona o modelo

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ X1 + X2
## Model 2: Y ~ X1 + X2 + X3
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1     10 101.655
## 2      9  97.469  1    4.186 0.3865 0.5496

```

As análises acima foram demonstradas apenas para detalhar o funcionamento dos algoritmos de seleção, utilizando F parcial. O F parcial é muito rigoroso, e por isso muitas vezes o pesquisador usa valores de  $\alpha$  maiores que 5%. Além disso, várias funções do *R* estão disponíveis para selecionar variáveis utilizando diferentes diferentes critérios. A função `stepAIC()` do pacote *MASS* seleciona variáveis utilizando o critério de informação de Akaike (AIC) como critério de seleção. Através do argumento `direction` é possível indicar o algoritmo a ser utilizado (*forward*, *backward* ou *stepwise*). Por default, a função `stepAIC()` utiliza o algoritmo *backward*. Para utilizar *forward* ou *stepwise*, é necessário indicar os modelos completos e reduzido no argumento `scope`.

```

require(readxl)
Reg_cherry=read_excel("D:/Desktop/final/Reg_Cherry.xlsx")

```

```

head(Reg_cherry)

```

```

## # A tibble: 6 x 6
##       Y     X1     X2     X3     X4     X5
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  3.14  2.60  2.84  1.15  1     1.98
## 2  3.36  2.72  2.95  1.21  1     2.15
## 3  3.31  2.76  2.90  1.18  1.18   2.12
## 4  3.26  2.64  2.78  1.12  1.04   2.14
## 5  3.03  2.53  2.65  1.10  0.845  1.93
## 6  3.23  2.59  2.76  1.11  0.954  2.12

```

```

mod9=lm(Y~X1+X2+X3+X4+X5, data=Reg_cherry) # O ponto indica que todas as variáveis estão incluídas
summary(mod9)

##
## Call:
## lm(formula = Y ~ X1 + X2 + X3 + X4 + X5, data = Reg_cherry)
##
## Residuals:
##    Min      1Q  Median      3Q     Max 
## -0.095586  0.000014  0.000245  0.000457  0.039900 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  0.004561  0.007787   0.586   0.558    
## X1          -0.007276  0.006604  -1.102   0.271    
## X2           0.007407  0.006016   1.231   0.219    
## X3           0.996903  0.013019  76.575 <2e-16 ***  
## X4           0.001987  0.003895   0.510   0.610    
## X5           0.997701  0.003282 304.022 <2e-16 ***  
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.005704 on 341 degrees of freedom
## Multiple R-squared:  0.9992, Adjusted R-squared:  0.9992 
## F-statistic: 8.99e+04 on 5 and 341 DF,  p-value: < 2.2e-16

```

São candidatas a permanecer no modelo apenas as variáveis  $X_3$  e  $X_5$ .

```

require(MASS)
stepAIC(mod9,direction =c("backward"))

```

```

## Start: AIC=-3579.65
## Y ~ X1 + X2 + X3 + X4 + X5
##
##              Df Sum of Sq     RSS     AIC
## - X4       1  0.00001 0.01110 -3581.4
## - X1       1  0.00004 0.01113 -3580.4
## - X2       1  0.00005 0.01114 -3580.1
## <none>            0.01110 -3579.6
## - X3       1  0.19079 0.20188 -2574.9
## - X5       1  3.00738 3.01848 -1636.4
##
## Step: AIC=-3581.38
## Y ~ X1 + X2 + X3 + X5
##
##              Df Sum of Sq     RSS     AIC
## - X1       1  0.0000  0.0111 -3582.2

```

```

## - X2     1    0.0000  0.0111 -3582.0
## <none>           0.0111 -3581.4
## - X3     1    0.1912  0.2023 -2576.2
## - X5     1   13.1318 13.1429 -1127.9
##
## Step: AIC=-3582.2
## Y ~ X2 + X3 + X5
##
##          Df Sum of Sq    RSS    AIC
## - X2     1    0.0000  0.0112 -3583.8
## <none>           0.0111 -3582.2
## - X3     1    0.2112  0.2223 -2545.5
## - X5     1   13.4001 13.4112 -1122.9
##
## Step: AIC=-3583.78
## Y ~ X3 + X5
##
##          Df Sum of Sq    RSS    AIC
## <none>           0.0112 -3583.8
## - X3     1    1.0352  1.0463 -2010.0
## - X5     1   13.7857 13.7968 -1115.0
##
## Call:
## lm(formula = Y ~ X3 + X5, data = Reg_cherry)
##
## Coefficients:
## (Intercept)      X3      X5
## 0.0008518    0.9997819   0.9995747

```

### 3.1.4 Falta de ajuste

Quando várias observações são realizadas para cada variável independente (experimentos com repetição, por exemplo), é necessário verificar a falta de ajuste. Nestes casos, o erro é dividido em duas partes: a) o erro puro, que consiste na diferença entre a média e as observações em cada variável; b) falta de ajuste, que é a diferença entre a média da variável independente e o valor ajustado pela regressão.

$$Y_{ij} - \hat{Y}_i = (Y_{ij} - \bar{Y}_{i\cdot}) + (\bar{Y}_{i\cdot} - \hat{Y}_i)$$

Onde  $\hat{Y}_{ij} - Y_{ij}$  é o erro do modelo,  $(Y_{ij} - \bar{Y}_j)$  é o erro puro e  $(\hat{Y}_{ij} - \bar{Y}_j)$  é a falta de ajuste.

```

require(readxl)
Reg=read_excel("D:/Desktop/final/Reg.xls")
head(Reg)

## # A tibble: 6 x 4

```

```

##      TRAT BLOCO DOSE      Y
##      <dbl> <dbl> <dbl> <dbl>
## 1      1     1     0    28
## 2      1     2     0    47
## 3      1     3     0    18
## 4      2     1     5    46
## 5      2     2     5    51
## 6      2     3     5    55

##### Ajustando uma regressão linear
mod10=lm(Y~DOSE, data=Reg)
## estima a reta = falta de ajuste
mod10.1=lm(Y~factor(DOSE), data=Reg)
## retorna a média das variáveis independentes = erro puro
anova(mod10,mod10.1)

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ DOSE
## Model 2: Y ~ factor(DOSE)
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1     19 3922.6
## 2     14 1067.3  5    2855.2 7.4903 0.001308 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

A significância do teste F indica que o modelo linear não é adequado para representar a relação entre as variáveis dependentes e independentes. Isso indica que o modelo ajustado “não se aproxima” satisfatoriamente da média das variáveis independentes, e que a falta de ajuste é elevada quando comparado ao erro puro. Agora, vamos ajustar um modelo quadrático:

```

##### Ajustando uma polinomio de segundo grau
mod11=lm(Y~DOSE+I(DOSE^2), data=Reg)
## estima a reta = falta de ajuste
mod11.1=lm(Y~factor(DOSE), data=Reg)
## retorna a média das variáveis independentes = erro puro
anova(mod11,mod11.1)

```

```

## Analysis of Variance Table
##
## Model 1: Y ~ DOSE + I(DOSE^2)
## Model 2: Y ~ factor(DOSE)
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1     18 1260.8
## 2     14 1067.3  4    193.48 0.6344 0.6462

```

A não significância do teste F indica que o modelo quadrático é adequado para representar a relação entre as variáveis dependentes e independentes. Aqui o exemplo é apresentado para um ajuste de polinômios, mas sua aplicação se estende a qualquer regressão (linear simples,

múltipla ou regressão não linear).

### 3.1.5 Análise dos resíduos

Na análise de regressão, os resíduos devem ser normalmente distribuídos, homocedásticos e independentes. O diagnóstico é realizado por testes estatísticos ou através de análise gráfica.

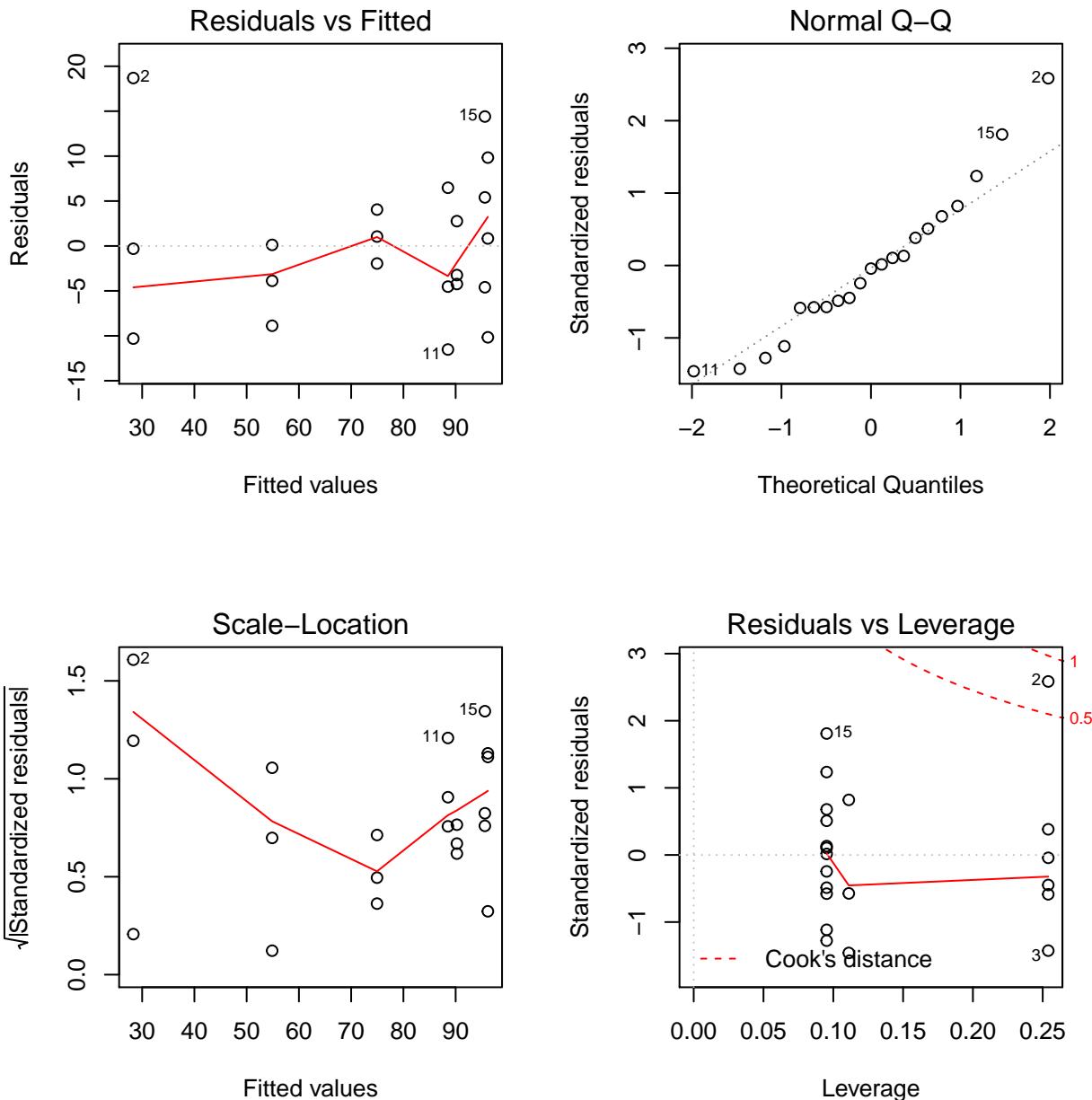
```
mod11=lm(Y~DOSE+I(DOSE^2), data=Reg)
residuos=residuals(mod11)
shapiro.test(residuos) ## Normalidade

##
## Shapiro-Wilk normality test
##
## data: residuos
## W = 0.95358, p-value = 0.3972

bartlett.test(residuos~DOSE,data=Reg) ## Homocedasticidade

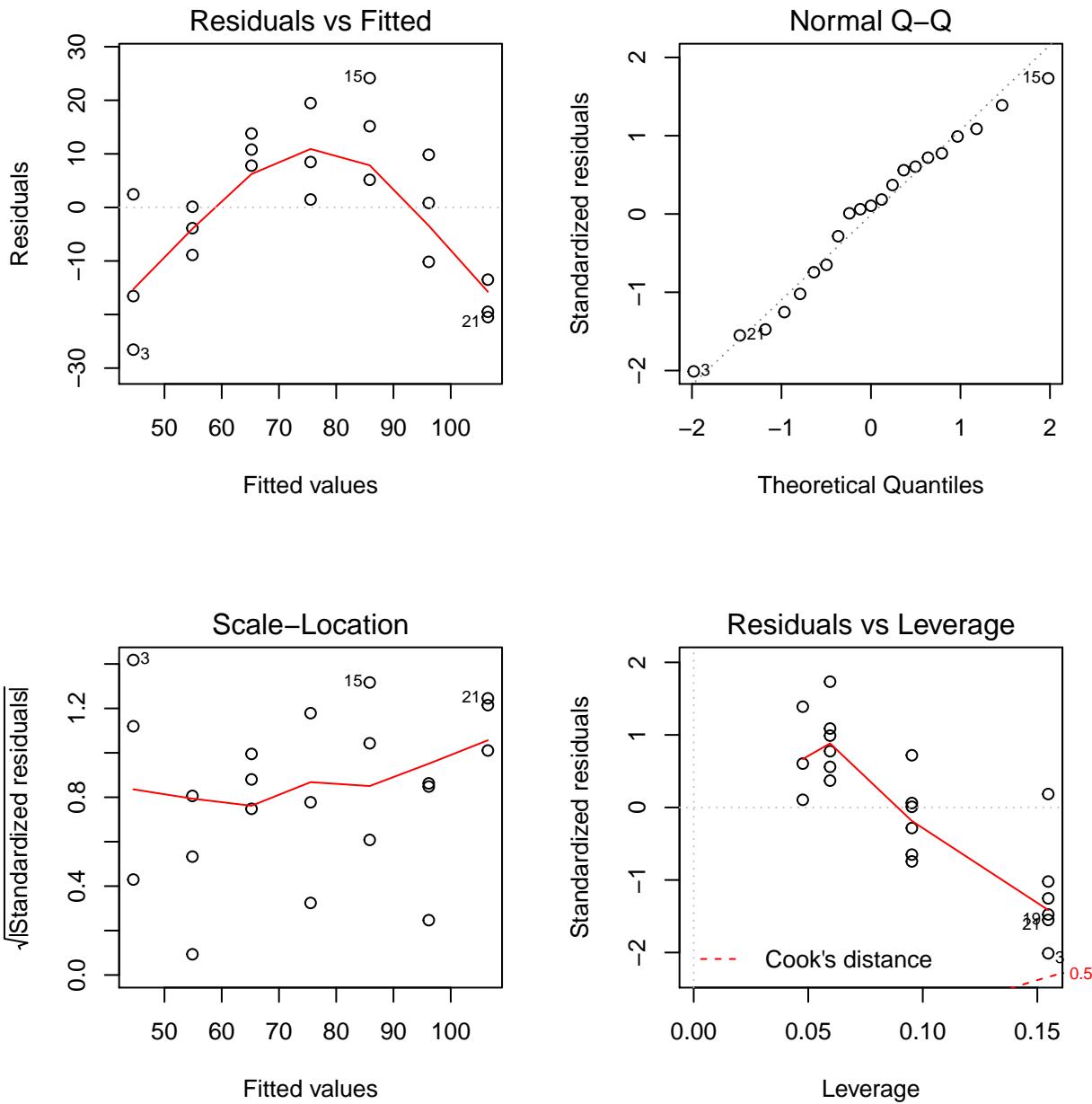
##
## Bartlett test of homogeneity of variances
##
## data: residuos by DOSE
## Bartlett's K-squared = 5.9846, df = 6, p-value = 0.4249

par(mfrow=c(2,2))
plot(mod11) ## análise gráfica dos resíduos
```



A análise dos resíduos também pode indicar a necessidade de adicionar variáveis explicativas ao modelo. Por exemplo, vamos ajustar um modelo linear a dados que tem (conhecidamente) comportamento quadrático.

```
mod10=lm(Y~DOSE, data=Reg)
residuos=residuals(mod10)
par(mfrow=c(2,2))
plot(mod10) ## análise gráfica dos resíduos
```



Percebe-se, pelo gráfico *residuals vs fitted*, que os resíduos não são aleatoriamente distribuídos em torno de zero. A distribuição sistemática dos resíduos indica que uma variável que explica consideravelmente a variabilidade dos dados não foi incluída no modelo (no caso o termo quadrático do polinômio).

### 3.1.6 Pontos influentes

As observações influentes podem ser mensuradas através da *distância de Cook*, *DFBETA* e *DFFITS*. O *DFFITS* e a *distância de Cook* medem a influência das observações sobre a

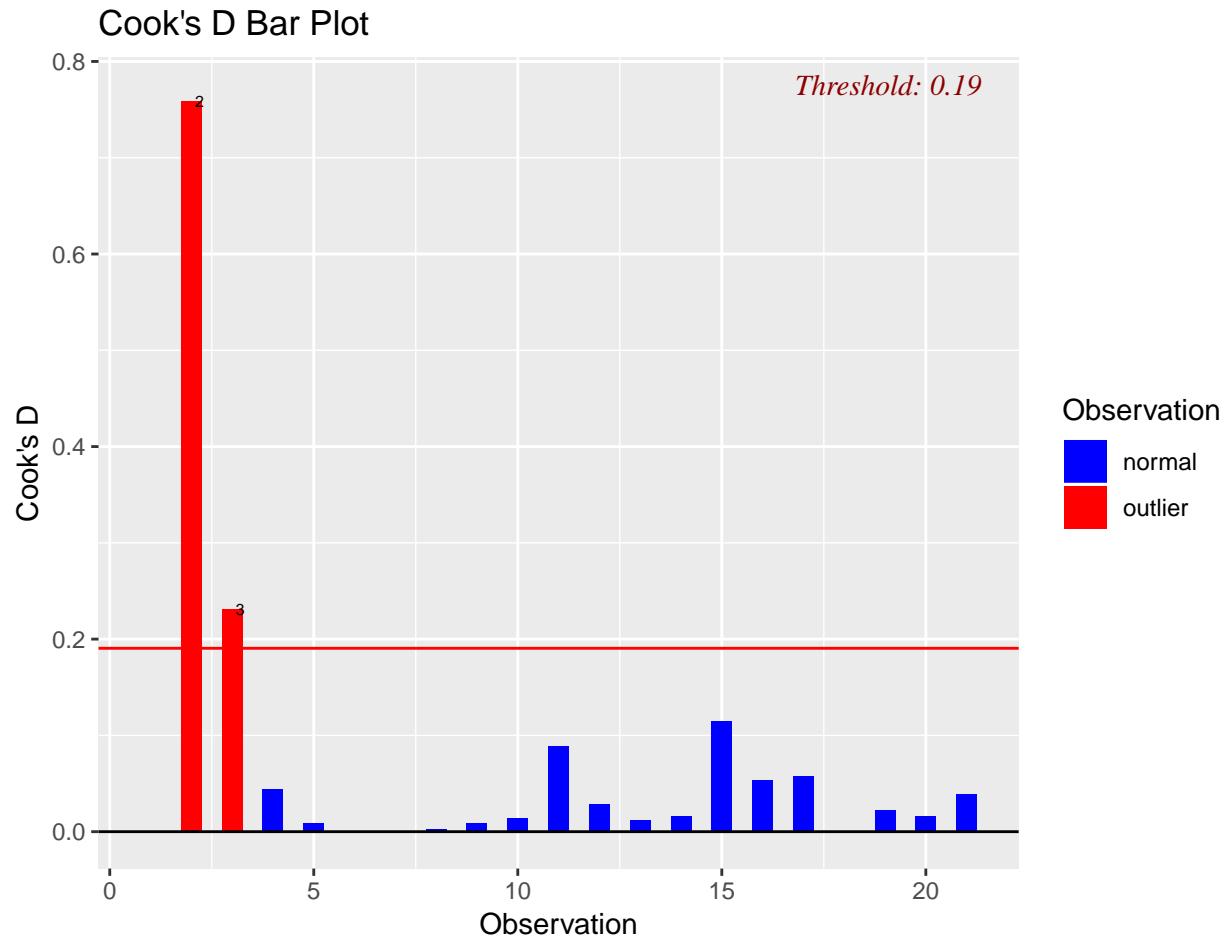


Figure 40: Distância de Cook representando a influencia dos pontos na predição das variáveis

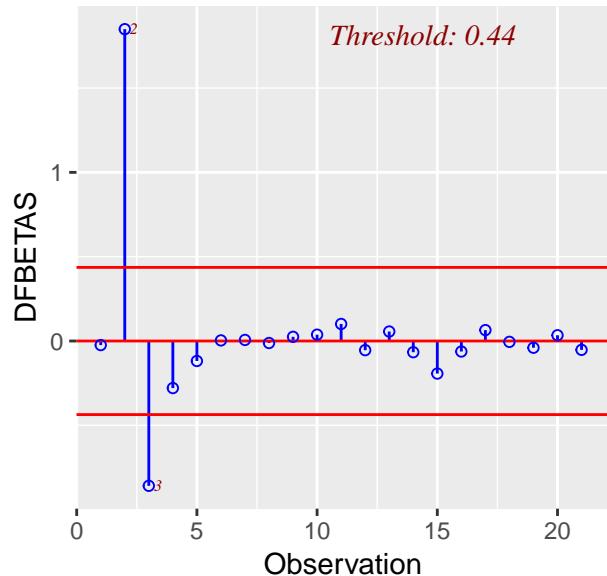
predição das variáveis; e o *DFBETA* mede a influência destas observações sobre as estimativas dos parâmetros.

Vamos utilizar as funções gráficas do pacote `olsrr` para fazer o diagnóstico dos pontos infuentes.

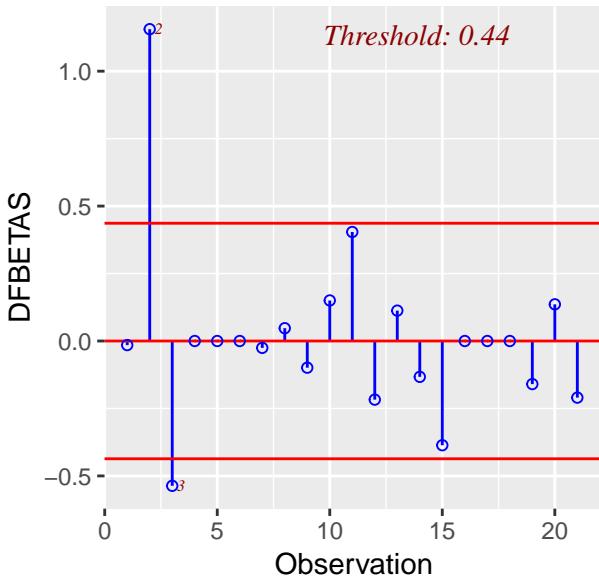
```
require(olsrr)
mod11=lm(Y~DOSE+I(DOSE^2), data=Reg)
olsrr::ols_plot_cooksd_bar(mod11) ## Distância de Cook

olsrr::ols_plot_dfbetas(mod11) ## DFBetas
```

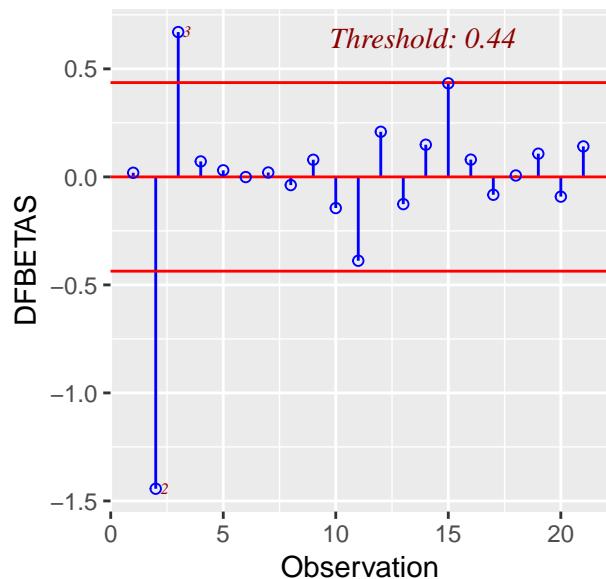
## Influence Diagnostics for (Intercept)



## Influence Diagnostics for I(DOSE')

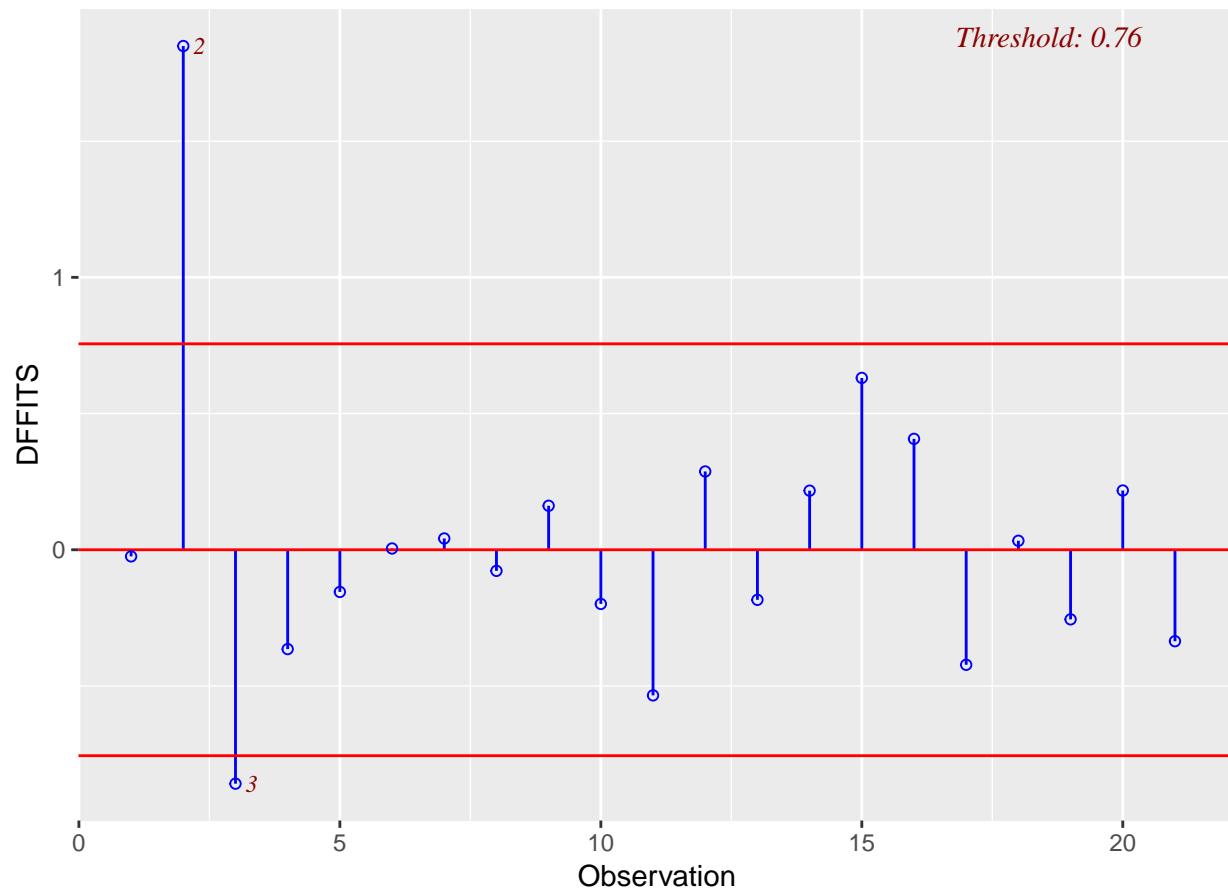


## Influence Diagnostics for DOSE



```
olsrr::ols_plot_dffits(mod11) ## DFFits
```

## Influence Diagnostics for Y



Reg

```
## # A tibble: 21 x 4
##   TRAT BLOCO DOSE     Y
##   <dbl> <dbl> <dbl> <dbl>
## 1     1     1     1     0    28
## 2     2     1     2     0    47
## 3     3     1     3     0    18
## 4     4     2     1     5    46
## 5     5     2     2     5    51
## 6     6     2     3     5    55
## 7     7     3     1    10    76
## 8     8     3     2    10    73
## 9     9     3     3    10    79
## 10   10     4     1    15    84
## # ... with 11 more rows
```

As observações 2 e 3 são as que mais influenciam os valores preditos e as estimativas dos parâmetros. Realmente, o valor da observação 2 é demasiadamente grande e o valor da observação 3 é demasiadamente pequeno. Os limites pela *distância de Cook*, *DFBETA* e

*DFFITS* para realizar o diagnóstico são  $\frac{4}{n} = \frac{4}{21} = 0,19$ ,  $\frac{4}{\sqrt{n}} = \frac{4}{\sqrt{21}} = 0,44$  e  $2 \times \sqrt{\frac{p}{n}} = 2 \times \sqrt{\frac{3}{21}} = 0,76$ , respectivamente.

## 3.2 Regressão não linear

Uma regressão é dita não linear quando os parâmetros não encontram-se de forma aditiva no modelo. Devido a isso, o sistema de equações normais  $(\mathbf{X}'\mathbf{X})^{-1}\beta = \mathbf{X}'\mathbf{Y}$  não pode ser resolvido analiticamente, e os parâmetros precisam ser estimados utilizando métodos iterativos.

### 3.2.1 Estimação

A estimação dos parâmetros é realizado pelo método dos mínimos quadrados . O método iterativo utilizado nos softwares R e SAS (dois dos softwares estatísticos mais utilizados no mundo) é o de *Gauss-Newton*. Este método utiliza aproximações lineares de Taylor de primeira ordem da função resposta, dada por

$$f(x, \theta) = f(x, \theta^0) + \frac{\partial f(x, \theta^0)}{\partial \theta} (\theta - \theta^0)$$

Essa aproximação linear de primeira ordem pode ser simplificada por  $f(\theta) = f(\theta^0) + \mathbf{F}(\theta - \theta^0)$ . Substituindo-a na função que minimiza a soma de quadrados, temos:

$$\begin{aligned} S(\theta) &= \sum_{i=1}^n (y_i - f(\hat{\theta}))^2 \\ S(\theta) &= \sum_{i=1}^n (y_i - f(\theta^0) - \mathbf{F}(\theta - \theta^0))^2 \\ S(\theta) &= \sum_{i=1}^n (\varepsilon_i - \mathbf{F}(\theta - \theta^0))^2 \end{aligned}$$

Percebe-se que a matriz  $\mathbf{F}$ , que substitui  $\mathbf{X}$ , é sempre dependente de um dos parâmetros do modelo, e por isso o sistema de equações não tem solução analítica. O sistema de equações não linear acima é resolvido por  $\theta - \theta^0 = (\mathbf{F}'\mathbf{F})^{-1}\mathbf{F}'\varepsilon$ , que reorganizada como  $\theta = \theta^0 + (\mathbf{F}'\mathbf{F})^{-1}\mathbf{F}'\varepsilon$ , corresponde ao primeiro passo do método iterativo de *Gauss-Newton*. Para algoritimo seja iniciado, um valor inicial para os parâmetros deve ser declarado. O processo é repetido até obter convergência, que ocorre quando os valores estimados em cada passo são próximos um dos outros.

### 3.2.2 Ajustando o modelo com a função `nls()`

A função `nls()` pode ser utilizada para ajustar modelos não lineares. Os principais argumentos da função são: i) `formula`, onde o modelo é declarado; ii) `data`, onde os dados são declarados e iii) `start`, que é uma lista com os valores iniciais dos parâmetros.

#### Valores iniciais dos parâmetros

O primeiro passo da análise é encontrar os valores iniciais dos parâmetros. O método gráfico é útil para cumprir esse objetivo. Valores dos parâmetros são declarados até o ponto em que a curva gerada se aproxime dos valores observados. A programação que será apresentada foi obtida no blog Ridículas, mantido pelo LEG da UFPR. Utilizaremos como exemplo o modelo logístico (uma de suas parametrizações), dado por

$$Y_i = \frac{\beta_1}{1 + e^{(\beta_2 - \beta_3 t_i)}} + \varepsilon_i$$

```
require(readxl)
nls_tomato=read_excel("D:/Desktop/final/nls_tomato.xlsx")
nls_tomato=subset(nls_tomato,Genotipo=="Cordillera") ## Selecionado só um genótipo

## Valores iniciais
plot(num~DAT,data=nls_tomato) # Plotando os valores observados

logi<-function(x,b1,b2,b3){
  b1/(1+exp(b2-b3*x))
} ## modelo logístico

b1= 30 # valor inicial de b1 "bem errado"
b2= 19 # valor inicial de b2 "bem errado"
b3= 0.2 # valor inicial de b3 "bem errado"

curve(logi(x,b1,b2,b3),add=T,col=2) ## traçando a curva com os parâmetros declarados

## Manipulando os valores
require(manipulate) ## Permite criar gráficos "móvels"

manipulate({
  plot(num~DAT,data=nls_tomato)
  curve(logi(x,b1=b1,b2=b2,b3=b3),add=TRUE)
  start<-list(b1=b1,b2=b2,b3=b3)},
  b1=slider(1,30,initial=30), ## valores que irão variar para b0
  b2=slider(5,40,initial=19), ## valores que irão variar para b1
  b3=slider(0.2,1.5,initial=0.2) ## valores que irão variar para b2
)
start()
```

## Ajustando o modelo

Os valores iniciais serão armazenados na lista `start`, e esta será declarada no argumento `start` da função `nls()`. A função `summary()` retorna o teste de Wald para os parâmetros.

```
start=list(b1=30,b2=19,b3=0.2)
nls1=nls(num~b1/(1+exp(b2-b3*DAT)),
         data=nls_tomato, ## indica os dados
         start=start) ## indica os valores iniciais
```

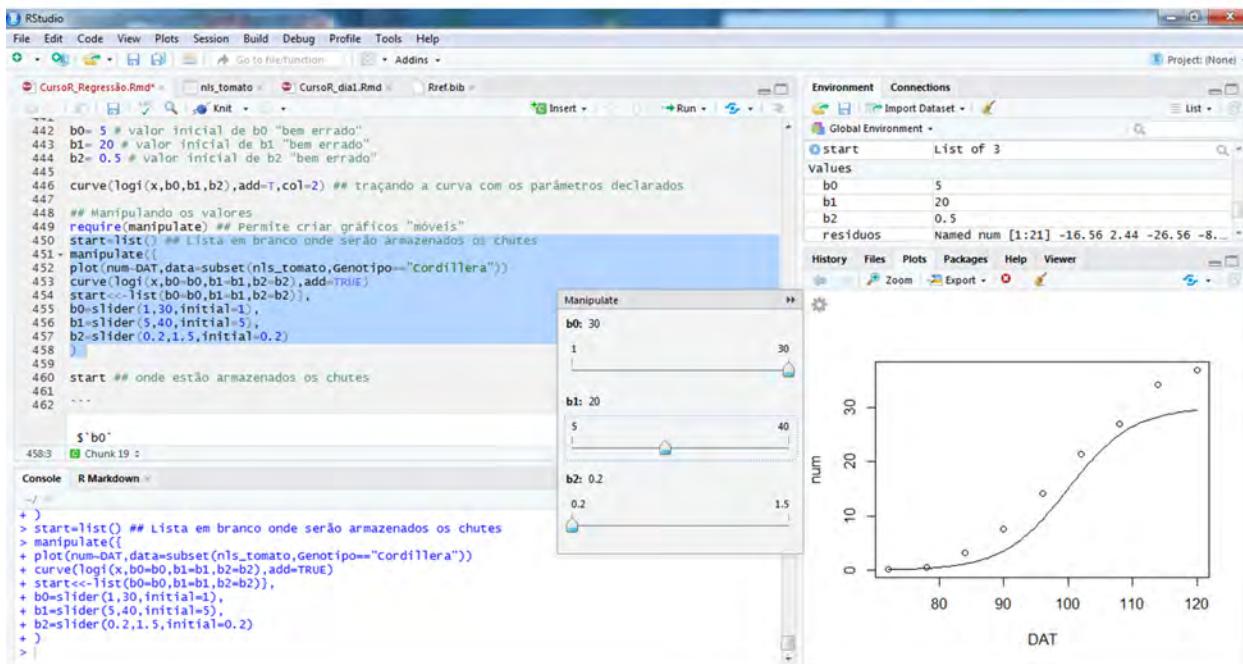


Figure 41: Manipulando os valores dos parâmetros

```
summary(nls1)
```

```
##
## Formula: num ~ b1/(1 + exp(b2 - b3 * DAT))
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
##   b1 39.68150  1.50505 26.37 1.96e-07 ***
##   b2 13.53979  0.94913 14.27 7.42e-06 ***
##   b3  0.13390  0.01018 13.15 1.19e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9108 on 6 degrees of freedom
##
## Number of iterations to convergence: 7
## Achieved convergence tolerance: 3.652e-06
```

### 3.2.3 Análise dos resíduos

Os resíduos dos modelos não lineares também deve ser normalmente distribuídos, homocedásticos e independentes. Os testes estatísticos e a análise dos resíduos seguem os mesmos princípios dos modelos lineares.

```

#### Normalidade
res_nls1=residuals(nls1)
shapiro.test(res_nls1)

##
## Shapiro-Wilk normality test
##
## data: res_nls1
## W = 0.92624, p-value = 0.4464

### Passando a função nls para lm
nls1_grad=attr(nls1$m$fitted(),"gradient") # obtém matriz gradiente
nls1_lm=lm(num~1+nls1_grad, data=nls_tomato)

### Homogeneidade
require(lmtest) # pacote para carregar teste de Breusch-Pagan (homogeneidade)
bttest(nls1_lm) # teste de Breusch-Pagan (homogeneidade)

##
## studentized Breusch-Pagan test
##
## data: nls1_lm
## BP = 1.5734, df = 2, p-value = 0.4554

### Independência
require (car) # pacote para carregar teste de DW (independência)
durbinWatsonTest(nls1_lm) # teste de DW (independência)

## lag Autocorrelation D-W Statistic p-value
##    1      0.08163667     1.763778   0.184
## Alternative hypothesis: rho != 0

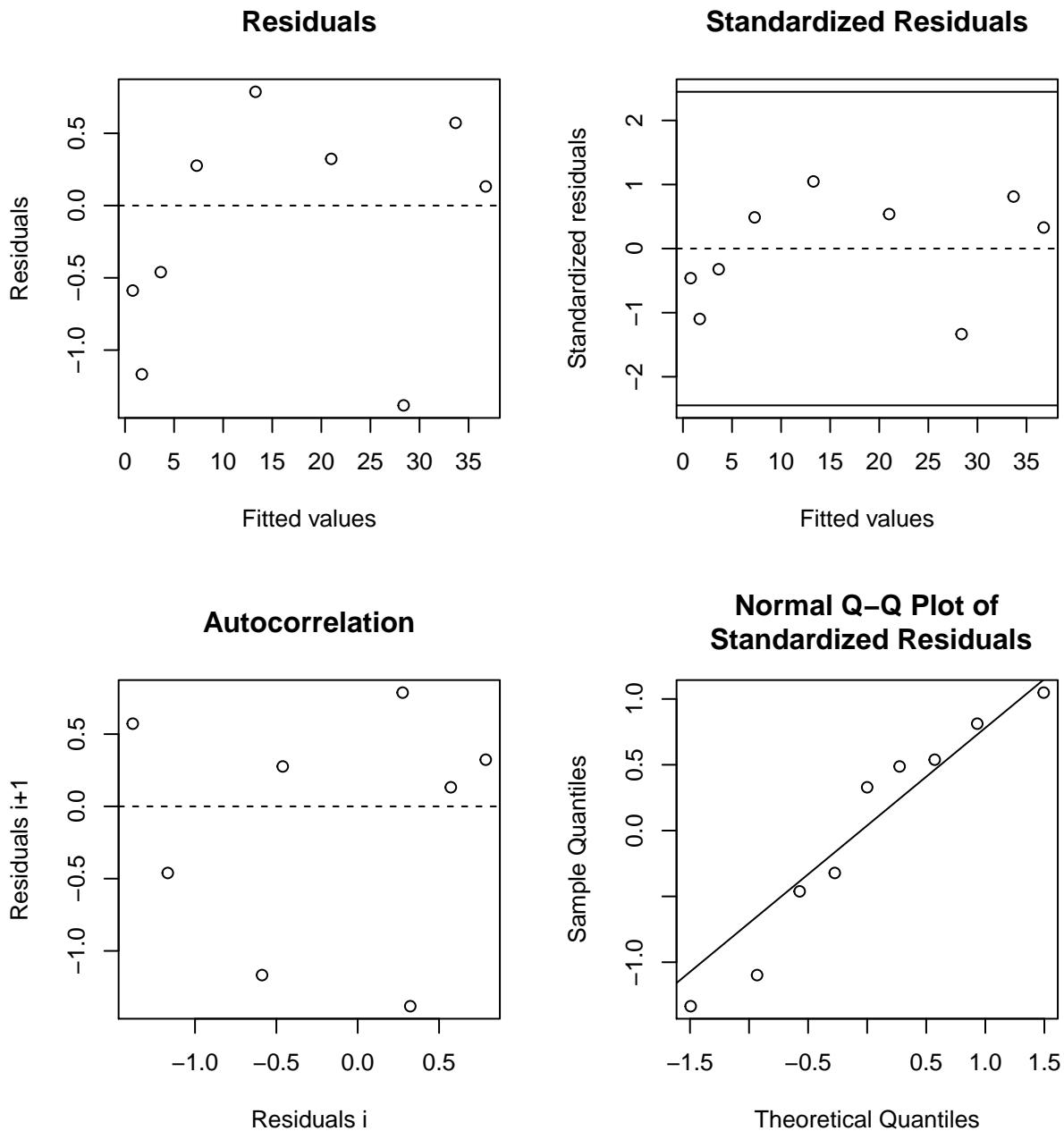
```

O cumprimento dos pressupostos dos resíduos não afeta a estimativa dos parâmetros, mas é de extrema importância para construir intervalos de confiança e testar hipóteses. Percebe-se, no nosso exemplo, que os pressuposto foram cumpridos ( $p\text{-valor}>0,05$ ). Para realizar a análise gráfica dos resíduos pode-se utilizar a função `nlsResiduals` do pacote `nlstools`.

```

require(nlstools)
res1_nls1 = nlsResiduals(nls1)
plot(res1_nls1)

```



No exemplo acima, apenas uma observação foi realizada para cada variável independente (dias após o transplante, no caso). Por isso optou-se por utilizar o teste de *Breusch-Pagan* para realizar o diagnóstico de homocedasticidade dos resíduos. Quando mais de uma observação é realizada em cada variável independente, pode-se utilizar os testes de *Bartlett* ou *Levene*.

```
### Carrgando os dados
nls_eggplant=read_excel("D:/Desktop/final/nls_eggp.xlsx")
nls_eggplant=subset(nls_eggplant, Estufa=="1") ## Selecionado só um genótipo
```

```
### Ajustando o modelo
```

```

start=list(b1=10,b2=6.7,b3=0.073)
nls2=nls(num~b1/(1+exp(b2-b3*DAT)),
         data=nls_eggplant, ## indica os dados
         start=start) ## indica os valores iniciais
summary(nls2)

##
## Formula: num ~ b1/(1 + exp(b2 - b3 * DAT))
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## b1 10.223999  0.377492 27.084 < 2e-16 ***
## b2  6.765817  0.686703  9.853 2.35e-11 ***
## b3  0.072527  0.008245   8.796 3.63e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6438 on 33 degrees of freedom
##
## Number of iterations to convergence: 6
## Achieved convergence tolerance: 3.861e-06
#### Normalidade
res_nls2=residuals(nls2)
shapiro.test(res_nls2)

##
## Shapiro-Wilk normality test
##
## data: res_nls2
## W = 0.96901, p-value = 0.3991
####
## Homogeneidade
bartlett.test(res_nls2~DAT,data=nls_eggplant)

##
## Bartlett test of homogeneity of variances
##
## data: res_nls2 by DAT
## Bartlett's K-squared = 11.741, df = 5, p-value = 0.03851
####
## Independência
nls2_grad=attr(nls2$m$fitted(),"gradient") # obtém matriz gradiente
nls2_lm=lm(num~-1+nls2_grad, data=nls_eggplant)

####
## Independência
require (car) # pacote para carregar teste de DW (independência)
durbinWatsonTest(nls2_lm) # teste de DW (independência)

```

```

## lag Autocorrelation D-W Statistic p-value
##      1          0.1801403     1.621292   0.114
## Alternative hypothesis: rho != 0

```

### 3.2.4 Medidas de não linearidade

Conforme vimos acima, a estimação dos parâmetros é realizada pelo método dos mínimos quadrados utilizando aproximações lineares de Taylor da função resposta. É esta aproximação linear que garante que os parâmetros estimados sejam próximos de não viesados (só serão não viesados assintoticamente). Modelos com aproximação linear pobre tendem a ter parâmetros muito viesados, o que impede que eles sejam utilizados para explicar determinado fenômeno biológico (eles têm ininterpretação biológica). As medidas de curvatura de Bates e Watts são amplamente utilizadas para avaliar o grau de não linearidade da função resposta. Para maiores detalhes, ver Bates and Watts (1988), cap. 7 e Seber and Wild (2003), cap. 4. Essas medidas são facilmente implementadas através da função `rms.curv()` do pacote *MASS*.

```

## Chute inicial e modelo
start=list(b1=30,b2=19,b3=0.2)
nls1=nls(num~b1/(1+exp(b2+b3*DAT)),
         data=nls_tomato, ## indica os dados
         start=start)

## Medidas de não linearidade
nls1_hess=deriv3(~b1/(1+exp(b2+b3*DAT)),c("b1","b2","b3"),
                 function(DAT,b1,b2,b3) NULL) ## hessiana
nls1_hess.1=nls(num~nls1_hess(DAT,b1,b2,b3),data=nls_tomato,start=start)

rms.curv(nls1_hess.1)

## Parameter effects: c^theta x sqrt(F) = 0.806
##           Intrinsic: c^iota x sqrt(F) = 0.1235

```

Os valores que a função `rms.curv()` retorna são  $c^\theta \times \sqrt{F_{\alpha;p,n-p}}$  e  $c^\iota \times \sqrt{F_{\alpha;p,n-p}}$ . Valores baixos destas duas medidas indicam que a função tem boa aproximação linear e, consequentemente, os parâmetros são próximos de ser não viesados.

### 3.2.5 Comparação de parâmetros

Os parâmetros em um modelo podem ser comparados utilizando variáveis *dummy*. Através desta técnica, a ocorrência de um determinado fator é associado a um nova variável. Como os modelos são aninhados, pode-se utilizar o teste F para verificar a significância do parâmetro (e, consequentemente, do fator) associado a esta variável.

Vamos ao exemplo. Suponha que queiramos comparar a produção e a taxa de produção de frutos de dois genótipos de tomate. Sabemos que quando acumuladas, a produção de olerícolas tem comportamento sigmoide (Lucio A. D., Nunes, and Rego 2016a Lucio A. D., Nunes, and

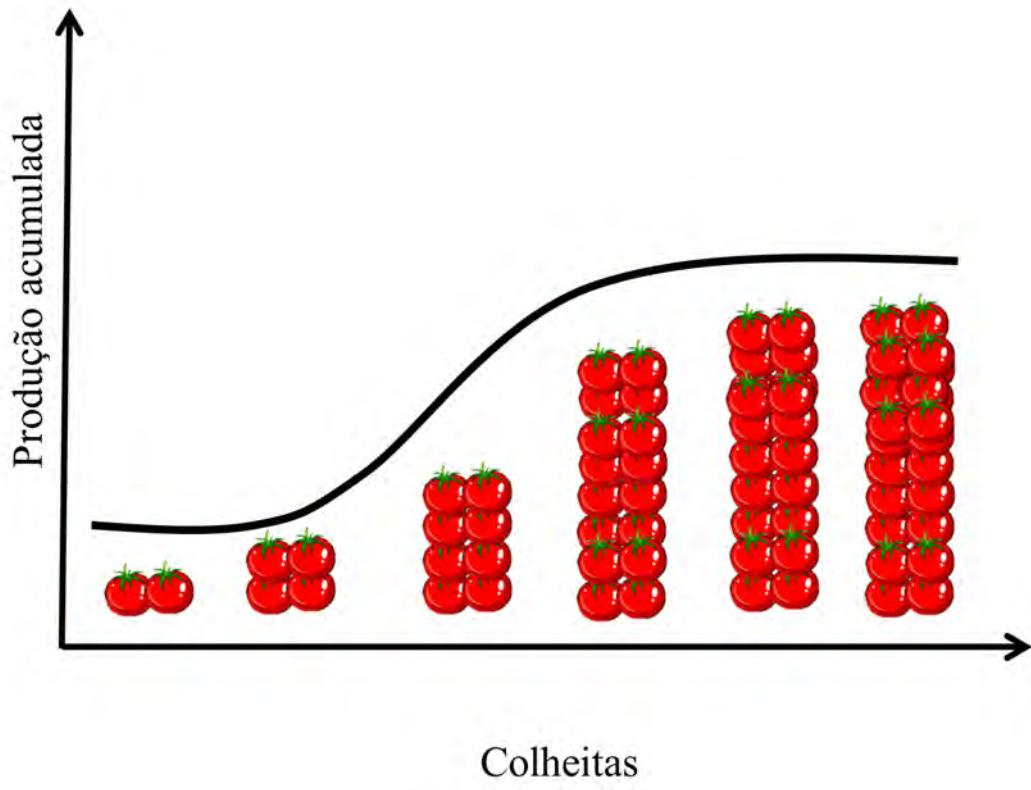


Figure 42: Comportamento da produção acumulada em olerícolas

Rego (2016b) Lucio et al. (2016)), o que permite determinar essas características através dos parâmetros de um modelo logístico:

$$Y_i = \frac{\beta_1}{1 + e^{(\beta_2 - \beta_3 t_i)}} + \varepsilon_i$$

No modelo acima,  $\beta_1$  representa a assíntota, e está associada a produção total dos genótipos;  $\beta_3$  é a taxa de produção de frutos, e está associada a precocidade produtiva dos genótipos (Sari 2018). Vamos testar as seguintes hipóteses:

$$\begin{aligned} H_0 &: \beta_{11} = \beta_{12} \\ H_A &: \beta_{11} \neq \beta_{12} \end{aligned}$$

$$\begin{aligned} H_0 &: \beta_{31} = \beta_{32} \\ H_A &: \beta_{31} \neq \beta_{32} \end{aligned}$$

## Testando a hipótese $H_0 : \beta_{11} = \beta_{12}$

Associamos os fatores a novas variáveis através de uma nova coluna no banco de dados. No nosso caso, a coluna “Completo” associa o fator aos parâmetros do modelo logístico. Então, como o modelo logístico possui 3 parâmetros, ao associarmos variáveis dummys ao fator genótipo (são dois genótipos), o modelo completo passa a ter 6 parâmetros.

Para tentar esta hipótese, associamos variáveis dummy s apenas aos parâmetros  $\beta_2$  e  $\beta_3$ . Neste caso, o modelo passa a ter 5 parâmetros (1  $\beta_1$ , 2  $\beta_2$  e 2  $\beta_3$ ). Podemos comparar esse modelo reduzido com o modelo completo de 6 parâmetros. Se o teste F der significativo, concluímos que há influência do fator genótipo.

```
## Lendo os dados
require(xlsx)
nls_tomato=read_excel("D:/Desktop/final/nls_tomato.xlsx")
nls_tomato$Genotipo=as.factor(nls_tomato$Genotipo)
nls_tomato$Completo=as.factor(nls_tomato$Completo)
nls_tomato$Reduzido=as.factor(nls_tomato$Reduzido)

nls_tomato

## # A tibble: 18 x 6
##   Genotipo    DAT     peso   num Completo Reduzido
##   <fct>      <dbl>   <dbl> <dbl> <fct>    <fct>
## 1 Gaucho      72     5.71  0.075 A        A
## 2 Gaucho      78    125.   0.995 A        A
## 3 Gaucho      84    318.   3.07  A        A
## 4 Gaucho      90   1206.   8.16  A        A
## 5 Gaucho      96   2018.  12.7   A        A
## 6 Gaucho     102   2429.  15.3   A        A
## 7 Gaucho     108   2720.  17.1   A        A
## 8 Gaucho     114   2895.  18.4   A        A
## 9 Gaucho     120   3030.  19.7   A        A
## 10 Cordillera 72    16.4   0.2    B        A
## 11 Cordillera 78    52.2   0.55   B        A
## 12 Cordillera 84    311.   3.18   B        A
## 13 Cordillera 90    804.   7.58   B        A
## 14 Cordillera 96   1632.  14.1   B        A
## 15 Cordillera 102   2521.  21.3   B        A
## 16 Cordillera 108   3164.  27     B        A
## 17 Cordillera 114   3892.  34.2   B        A
## 18 Cordillera 120   4135.  36.9   B        A

## Modelo completo
tomato_completo=nls(num~b1[Completo] / (1+exp(b2[Completo]-b3[Completo]*DAT)) ,
                     data=nls_tomato,
                     start=list(
                     b1=c(18.94,39.68),
                     b2=c(16.04,13.54),
```

```

b3=c(0.17,0.13)))

## Modelo completo
tomato_reduzido.b1=nls(num~b1[Reduzido]/(1+exp(b2[Completo]-b3[Completo]*DAT)),
                       data=nls_tomato,
                       start=list(
                         b1=c(18.94),
                         b2=c(16.04,13.54),
                         b3=c(0.17,0.13)))

anova(tomato_reduzido.b1,tomato_completo)

```

```

## Analysis of Variance Table
##
## Model 1: num ~ b1[Reduzido]/(1 + exp(b2[Completo] - b3[Completo] * DAT))
## Model 2: num ~ b1[Completo]/(1 + exp(b2[Completo] - b3[Completo] * DAT))
##   Res.Df Res.Sum Sq Df Sum Sq F value    Pr(>F)
## 1      13    55.636
## 2      12    7.833  1 47.802  73.228 1.875e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Os valores da assintota diferem estatisticamente entre si. Percebe-se claramente que o genótipo Cordillera foi mais produtivo que o genótipo Gaúcho.

**Testando a hipótese  $H_0 : \beta_{31} = \beta_{32}$**

```

## Modelo completo
tomato_completo=nls(num~b1[Completo]/(1+exp(b2[Completo]-b3[Completo]*DAT)),
                      data=nls_tomato,
                      start=list(
                        b1=c(18.94,39.68),
                        b2=c(16.04,13.54),
                        b3=c(0.17,0.13)))

## Modelo completo
tomato_reduzido.b3=nls(num~b1[Completo]/(1+exp(b2[Completo]-b3[Reduzido]*DAT)),
                       data=nls_tomato,
                       start=list(
                         b1=c(18.94,39.68),
                         b2=c(16.04,13.54),
                         b3=c(0.16)))

```

```
anova(tomato_reduzido.b3,tomato_completo)
```

```

## Analysis of Variance Table
##
## Model 1: num ~ b1[Completo]/(1 + exp(b2[Completo] - b3[Reduzido] * DAT))

```

```

## Model 2: num ~ b1[Completo]/(1 + exp(b2[Completo] - b3[Completo] * DAT))
##   Res.Df Res.Sum Sq Df Sum Sq F value    Pr(>F)
## 1      13   10.1479
## 2      12    7.8335  1  2.3144  3.5454 0.08417 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Os valores da taxa de produção de frutos não diferem estatisticamente entre si.

### 3.2.6 Representação gráfica dos modelos

Percebe-se claramente que um modelo comum aos dois genótipos não é possível (assintota diferem entre si). Por isso, optou-se por gerar um modelo em separado para cada genótipo. Podemos representar isso graficamente utilizando a função `plot()`.

```

logi<-function(x,b1,b2,b3){
  b1/(1+exp(b2-b3*x))
}

plot(num~DAT,data=subset(nls_tomato,Genotipo=="Gaúcho") ,ylim=c(0,50) ,
     xlab="Dias após o transplante (DAT)" ,
     ylab="Número de frutos por planta",pch=1)
points(num~DAT,data=subset(nls_tomato,Genotipo=="Cordillera") ,pch=2)

curve(logi(x,b1=summary(tomato_completo)$coefficients[1,1] ,
            b2=summary(tomato_completo)$coefficients[3,1] ,
            b3=summary(tomato_completo)$coefficients[5,1]),add=TRUE)

curve(logi(x,b1=summary(tomato_completo)$coefficients[2,1] ,
            b2=summary(tomato_completo)$coefficients[4,1] ,
            b3=summary(tomato_completo)$coefficients[6,1]),add=TRUE,lty=2)

legend("bottomright", legend=c("Gaúcho", "Cordillera"), lty=c(1,2) ,
       pch=c(1,2),bty="n")

```

### 3.3 Regressão bisegmentada com platô

Continuaremos tomindo como exemplo a produção de olerícolas de múltiplas colheitas para exemplificar o uso deste tipo de regressão. Os modelos com platô são modelos bi-segmentados cujo primeiro segmento descreve o crescimento da produção até determinado ponto, e um segundo segmento que descreve a estabilização da produção (platô). Podemos representar um modelo linear-platô por:

$$Y_i = \begin{cases} \beta_0 + \beta_1 x + \varepsilon_i, & \text{se } X_i < X_0 \\ P, & \text{se } X_i \geq X_0 \end{cases}$$

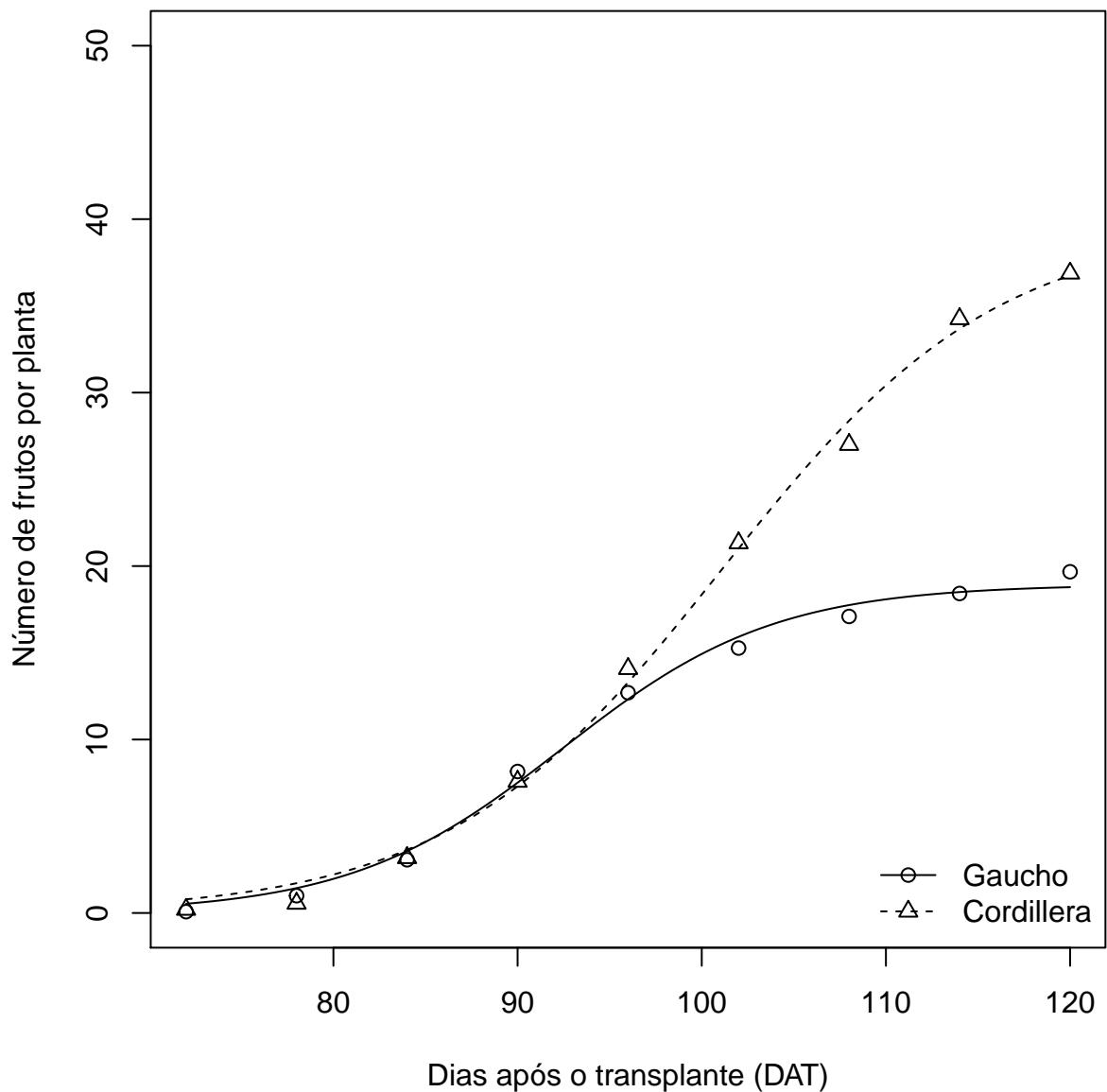


Figure 43: Representação gráfica em modelos não lineares

```

### Carregando os dados
plato_tomato=read_excel("D:/Desktop/final/plato_tomato.xlsx")

plato_cordillera=nls(num~(b0+b1*DAT*(DAT<=P))+
                      (b1*P*(DAT>P)),
                      data=subset(plato_tomato, Genotipo=="Cordillera"),
                      start=list(b0=5, b1=11/8, P=15))

summary(plato_cordillera)

##
## Formula: num ~ (b0 + b1 * DAT * (DAT <= P)) + (b1 * P * (DAT > P))
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
##   b0    4.83854   0.24586 19.68 0.000287 ***
##   b1    1.33203   0.04761 27.98 0.000100 ***
##   P     9.28641   0.25091 37.01 4.34e-05 ***
##   ---
##   Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2693 on 3 degrees of freedom
##
## Number of iterations to convergence: 4
## Achieved convergence tolerance: 6.131e-08

plato_SantaCl=nls(num~(b0+b1*DAT*(DAT<=P))+
                      (b1*P*(DAT>P)),
                      data=subset(plato_tomato, Genotipo=="Santa.Clara"),
                      start=list(b0=5, b1=11/8, P=15))

summary(plato_SantaCl)

##
## Formula: num ~ (b0 + b1 * DAT * (DAT <= P)) + (b1 * P * (DAT > P))
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
##   b0    2.50000   0.42213  5.922 0.009618 **
##   b1    0.71484   0.05641 12.673 0.001060 **
##   P     14.36066  0.89938 15.967 0.000534 ***
##   ---
##   Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5045 on 3 degrees of freedom
##
## Number of iterations to convergence: 2

```

```
## Achieved convergence tolerance: 1.835e-09
```

O genótipo Cordillera produz a uma taxa de 1,33 frutos/dia até os ~9 dias após o início das colheitas, quando começa a estabilizar a produção (17,19 frutos). Já o genótipo Santa Clara produz a uma taxa de 0,71 frutos/dia até os ~14 dias após o início das colheitas, quando começa a estabilizar a produção (12,69 frutos). Podemos verificar a taxa de produção e o ponto de estabilização através de variáveis dummy .

```
### Carregando os dados
```

```
plato_tomato=read_excel("D:/Desktop/final/plato_tomato.xlsx")
```

```
plato_tomato$Genotipo=as.factor(plato_tomato$Genotipo)
```

```
plato_tomato$Completo=as.factor(plato_tomato$Completo)
```

```
plato_tomato$Reduzido=as.factor(plato_tomato$Reduzido)
```

```
### Modelo completo
```

```
plato_completo=nls(num~(b0[Completo]+b1[Completo]*DAT*(DAT<=P[Completo]))+  
                    (b1[Completo]*P[Completo]*(DAT>P[Completo])),  
                    data=plato_tomato,  
                    start=list(b0=c(4.8,2.5),  
                               b1=c(1.33,0.71),  
                               P=c(9.28,14.36)))
```

```
### Modelo reduzido (taxa de produção)
```

```
plato_reduzido.b1=nls(num~(b0[Completo]+b1[Reduzido]*DAT*(DAT<=P[Completo]))+  
                    (b1[Reduzido]*P[Completo]*(DAT>P[Completo])),  
                    data=plato_tomato,  
                    start=list(b0=c(4.8,2.5),  
                               b1=c(1),  
                               P=c(9.28,14.36)))
```

```
anova(plato_reduzido.b1,plato_completo)
```

```
## Analysis of Variance Table
```

```
##
```

```
## Model 1: num ~ (b0[Completo] + b1[Reduzido] * DAT * (DAT <= P[Completo])) + (b1[Reduzido] * P[Completo] * (DAT > P[Completo]))
```

```
## Model 2: num ~ (b0[Completo] + b1[Completo] * DAT * (DAT <= P[Completo])) + (b1[Completo] * P[Completo] * (DAT > P[Completo]))
```

```
##   Res.Df Res.Sum Sq Df Sum Sq F value    Pr(>F)
```

```
## 1      7    9.6880
```

```
## 2      6    0.9813  1 8.7068  53.237 0.0003379 ***
```

```
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
### Modelo reduzido (Platô)
```

```
plato_reduzido.P=nls(num~(b0[Completo]+b1[Completo]*DAT*(DAT<=P[Reduzido]))+  
                     (b1[Completo]*P[Reduzido]*(DAT>P[Reduzido])),  
                     data=plato_tomato,  
                     start=list(b0=c(4.8,2.5),  
                                b1=c(1.33,0.71),  
                                P=c(9.28,14.36)))
```

```
P=c(12)))
```

```
anova(plato_reduzido.P,plato_completo)
```

```
## Analysis of Variance Table
##
## Model 1: num ~ (b0[Completo] + b1[Completo] * DAT * (DAT <= P[Reduzido])) + (b1[Completo] * DAT)
## Model 2: num ~ (b0[Completo] + b1[Completo] * DAT * (DAT <= P[Completo])) + (b1[Completo] * DAT)
##   Res.Df Res.Sum Sq Df Sum Sq F value    Pr(>F)
## 1      7     8.0519
## 2      6     0.9813  1 7.0706  43.233 0.0005936 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Percebe-se que a taxa de produção do genótipo Cordillera é significativamente superior a taxa de produção do genótipo Santa Clara. Como consequência, o momento de estabilização da produção ocorre mais precocemente no genótipo Cordillera. O exemplo foi realizado com dados de produção de olerícolas, mas pode ser adaptado para qualquer estudo (desde que tenha este comportamento).

### Representação gráfica dos modelos

```
plot(num~DAT,data=subset(plato_tomato,Genotipo=="Cordillera"),ylim=c(0,20),
      xlab="Dias após o transplante (DAT)",
      ylab="Número de frutos por planta",pch=1,lwd=2)
points(num~DAT,data=subset(plato_tomato,Genotipo=="Santa.Clara"),pch=2,lwd=2)

segments(y0=4.8385, x0=0, y1=17.2083, x1=9.28641,lty=1,lwd=2)
segments(y0=17.2083, x0=9.28641, y1=17.2083, x1=20,lty=1,lwd=2)

segments(y0=2.5000, x0=0, y1=12.7655, x1=14.3606,lty=2,lwd=2)
segments(y0=12.7655, x0=14.3606, y1=12.7655, x1=20,lty=2,lwd=2)

legend("bottomright", legend=c("Cordillera", "Santa Clara"), lty=c(1,2),
       pch=c(1,2), lwd=c(2,2), bty="n")
```

## 4 Biometria aplicada ao melhoramento genético de plantas

### 4.1 Associação entre variáveis

Conhecer o grau de associação entre caracteres é de fundamental importância em um programa de melhoramento genético vegetal. Esta importância aumenta, principalmente se algum caractere desejável é de difícil mensuração, ou apresenta baixa herdabilidade. O coeficiente de correlação produto-momento ( $r$ ) de (Pearson 1920), vem sendo amplamente utilizado para este fim. Embora o mérito desta análise seja atribuído à Karl Pearson, o método foi

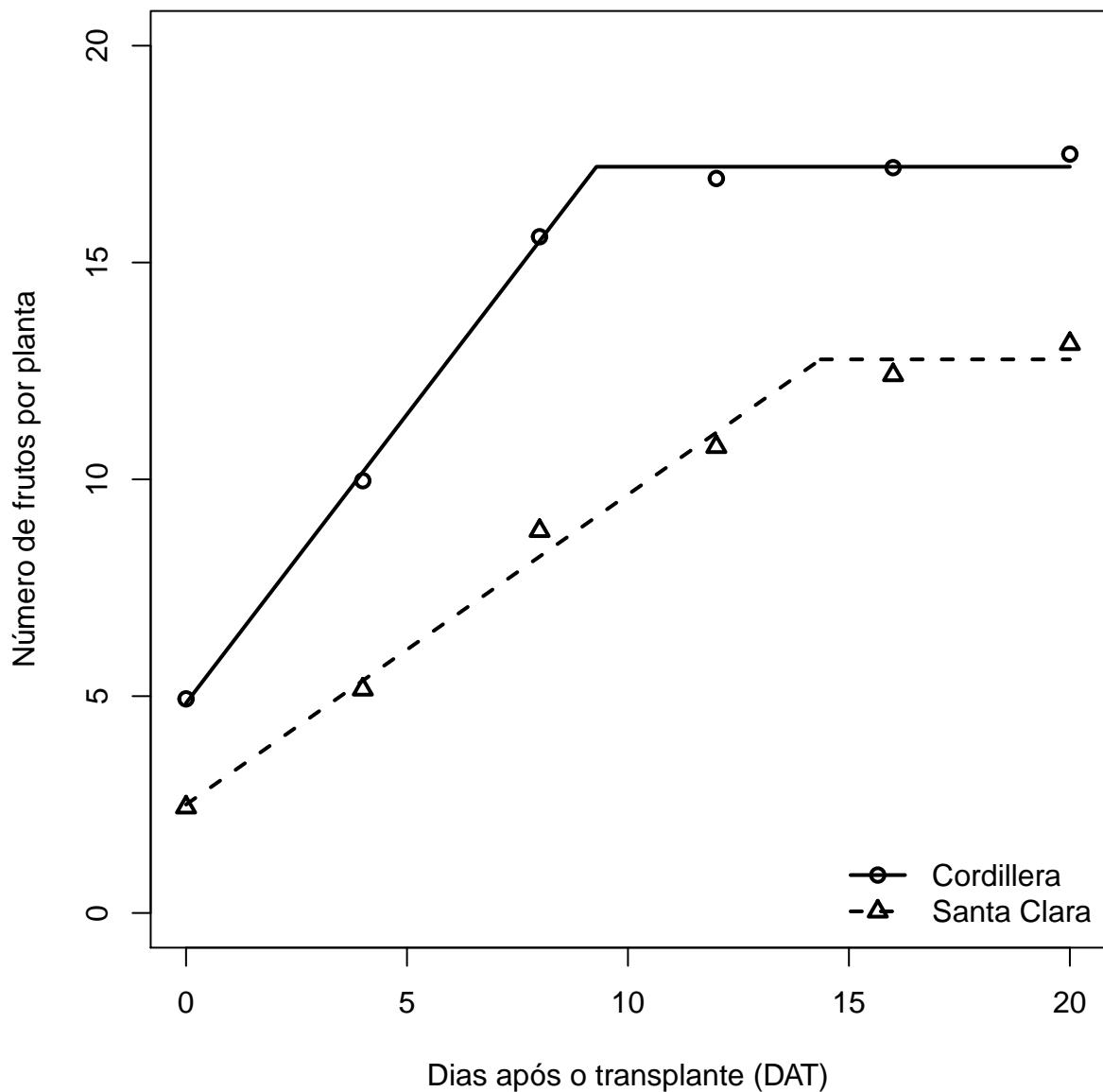


Figure 44: Representação gráfica de regressão bisegmentada com platô

originalmente concebido por Francis Galton, que definiu o termo correlação como o seguinte: *duas variáveis são ditas correlacionadas quando a variação de uma é acompanhada na média, mais ou menos a variação da outra, e no mesmo sentido* (Galton 1888).

## 4.2 Download dos dados

Nesta sessão, e na sessão de análise multivariada iremos utilizar o conjunto de dados *Data from a maize experiment for multivariate analysis* (Olivoto T. and Sari 2018b). Este conjunto de dados contém dados de 15 variáveis avaliadas em 13 híbridos de milho, que foram conduzidos em quatro ambientes. Em cada ambiente, um delineamento de blocos completos casualizados com três blocos e cinco observações por bloco foi utilizado. O download dos dados pode ser realizado pelo seguinte código.

```
# Importing data
datafactors = read.csv("https://data.mendeley.com/datasets/42xhfv7bmj/1/files/0cf1cc65-7

dataset = datafactors[,5:ncol(datafactors)]
datacor = dataset[,1:8]
```

## 4.3 Correlação linear simples

A estimativa do coeficiente de correlação linear de Pearson leva em consideração a covariância entre duas variáveis, representadas aqui por XY dividida pelo produto dos respectivos desvios padrões de X e de Y, conforme o seguinte modelo:

$$r = \frac{\sum_{i=1}^n \{(X_i - \bar{X})(Y_i - \bar{Y})\}}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2} \sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2}}$$

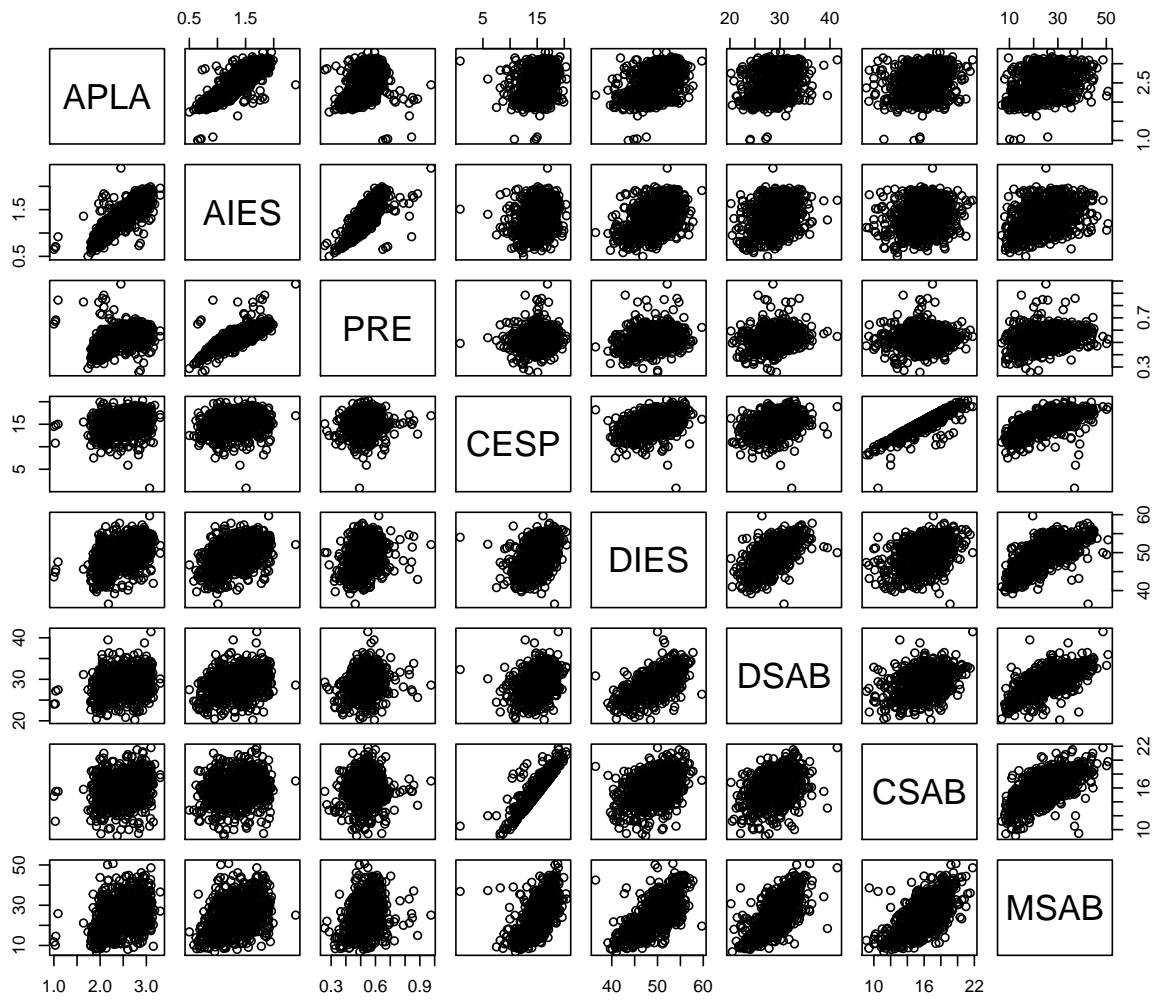
onde  $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$  e  $\bar{Y} = \frac{1}{n} \sum_{i=1}^n Y_i$ .

Esta sessão é focada em apresentar funções básicas e avançadas para visualização gráfica de associações e estimativas do coeficiente de correlação. Para este fim, utilizaremos o conjunto de dados *datacor*.

### 4.3.1 Visualização gráfica

A seguinte função proporciona uma visualização gráfica de todos os pares de correlação possíveis (*scatter-plot*)

```
pairs(datacor)
```



#### 4.3.2 Estimativa dos coeficientes de correlação

```
# Extracting variables
library(agricolae)

##
## Attaching package: 'agricolae'
## The following objects are masked from 'package:ExpDes':
##       lastC, order.group, tapply.stat
options(digits = 3)

# Computing correlation
corr = correlation(datacor)
```

```

# Imprimindo os coeficientes
print(corr$correlation)

##          APLA     AIES      PRE     CESP     DIES     DSAB     CSAB     MSAB
## APLA 1.00 0.84 0.32 0.23 0.47 0.24 0.21 0.38
## AIES 0.84 1.00 0.77 0.21 0.46 0.30 0.16 0.38
## PRE  0.32 0.77 1.00 0.10 0.28 0.24 0.05 0.24
## CESP 0.23 0.21 0.10 1.00 0.40 0.30 0.90 0.55
## DIES 0.47 0.46 0.28 0.40 1.00 0.65 0.41 0.67
## DSAB 0.24 0.30 0.24 0.30 0.65 1.00 0.34 0.66
## CSAB 0.21 0.16 0.05 0.90 0.41 0.34 1.00 0.60
## MSAB 0.38 0.38 0.24 0.55 0.67 0.66 0.60 1.00

# Imprimindo os p-valores
options(digits = 2)
print(corr$pvalue)

##          APLA     AIES      PRE     CESP     DIES     DSAB     CSAB     MSAB
## APLA 1.0e+00 0.0e+00 0.0e+00 3.0e-11 0.0e+00 7.5e-12 6.6e-09 0.0e+00
## AIES 0.0e+00 1.0e+00 0.0e+00 4.5e-09 0.0e+00 0.0e+00 6.5e-06 0.0e+00
## PRE  0.0e+00 0.0e+00 1.0e+00 5.3e-03 2.9e-15 1.0e-11 1.6e-01 1.3e-11
## CESP 3.0e-11 4.5e-09 5.3e-03 1.0e+00 0.0e+00 0.0e+00 0.0e+00 0.0e+00
## DIES 0.0e+00 0.0e+00 2.9e-15 0.0e+00 1.0e+00 0.0e+00 0.0e+00 0.0e+00
## DSAB 7.5e-12 0.0e+00 1.0e-11 0.0e+00 0.0e+00 1.0e+00 0.0e+00 0.0e+00
## CSAB 6.6e-09 6.5e-06 1.6e-01 0.0e+00 0.0e+00 0.0e+00 1.0e+00 0.0e+00
## MSAB 0.0e+00 0.0e+00 1.3e-11 0.0e+00 0.0e+00 0.0e+00 0.0e+00 1.0e+00

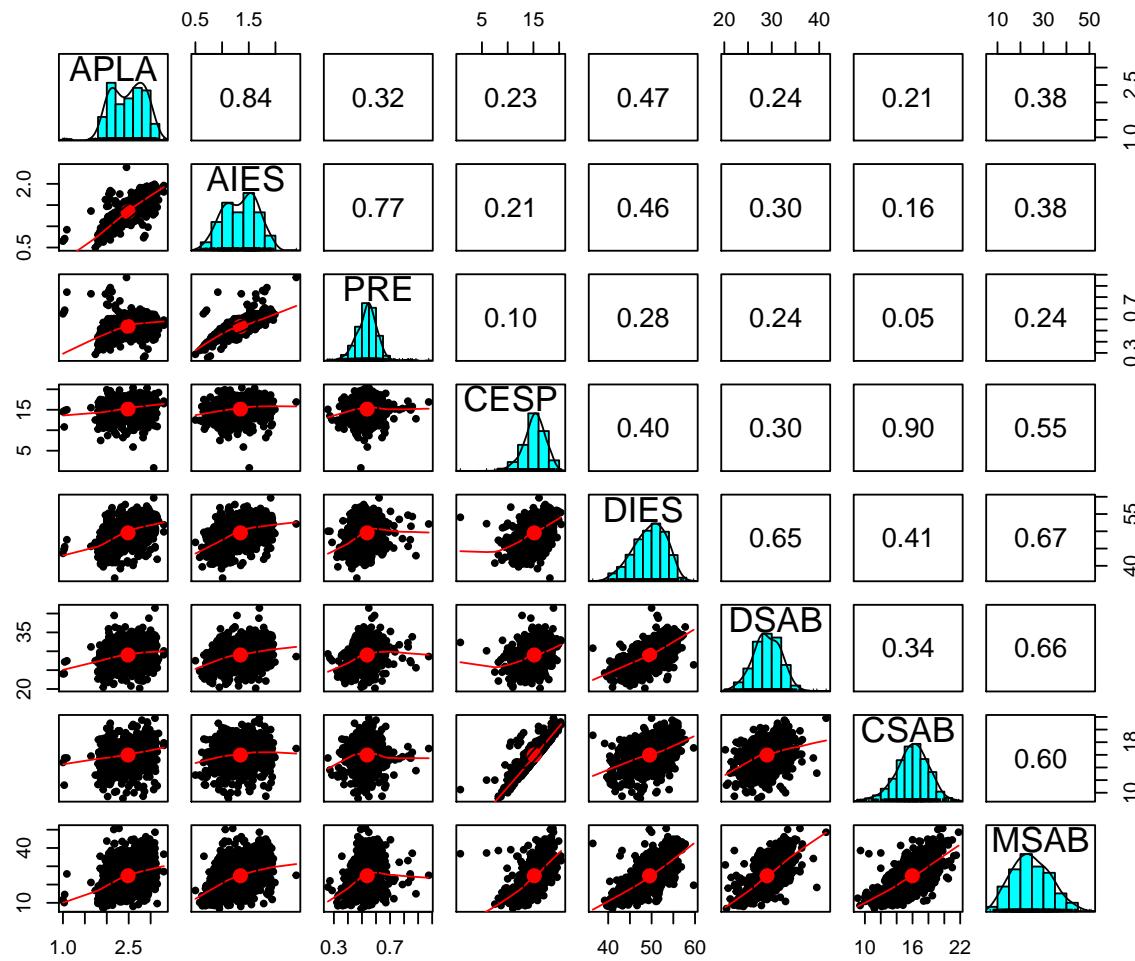
```

### 4.3.3 Combinando visualização gráfica e numérica (I)

```

# Criando o gráfico utilizando o conjunto de dados datacor
psych::pairs.panels(datacor)

```



Na figura acima, os pontos observados são plotados na diagonal inferior. Na diagonal, é apresentada a função densidade de probabilidade normal e um histgrama de cada variável. A diagonal superior contém os coeficientes de correlação.

#### 4.3.4 Combinando visualização gráfica e numérica (II)

A função `corr.plot()` do pacote `cursoR` retorna um gráfico semelhante ao anterior, no entanto possui diversas opções, tais como mudança no tamanho da letra dependendo da magnitude da correlação e indicação de cores para correlações significativas.

```
# Criando o gráfico utilizando o conjunto de dados dacor
library(cursoR)
corr.plot(dacor,
          signcol = "green",
          sizepoint = 1,
          minsize = 3,
          maxsize = 4)
```

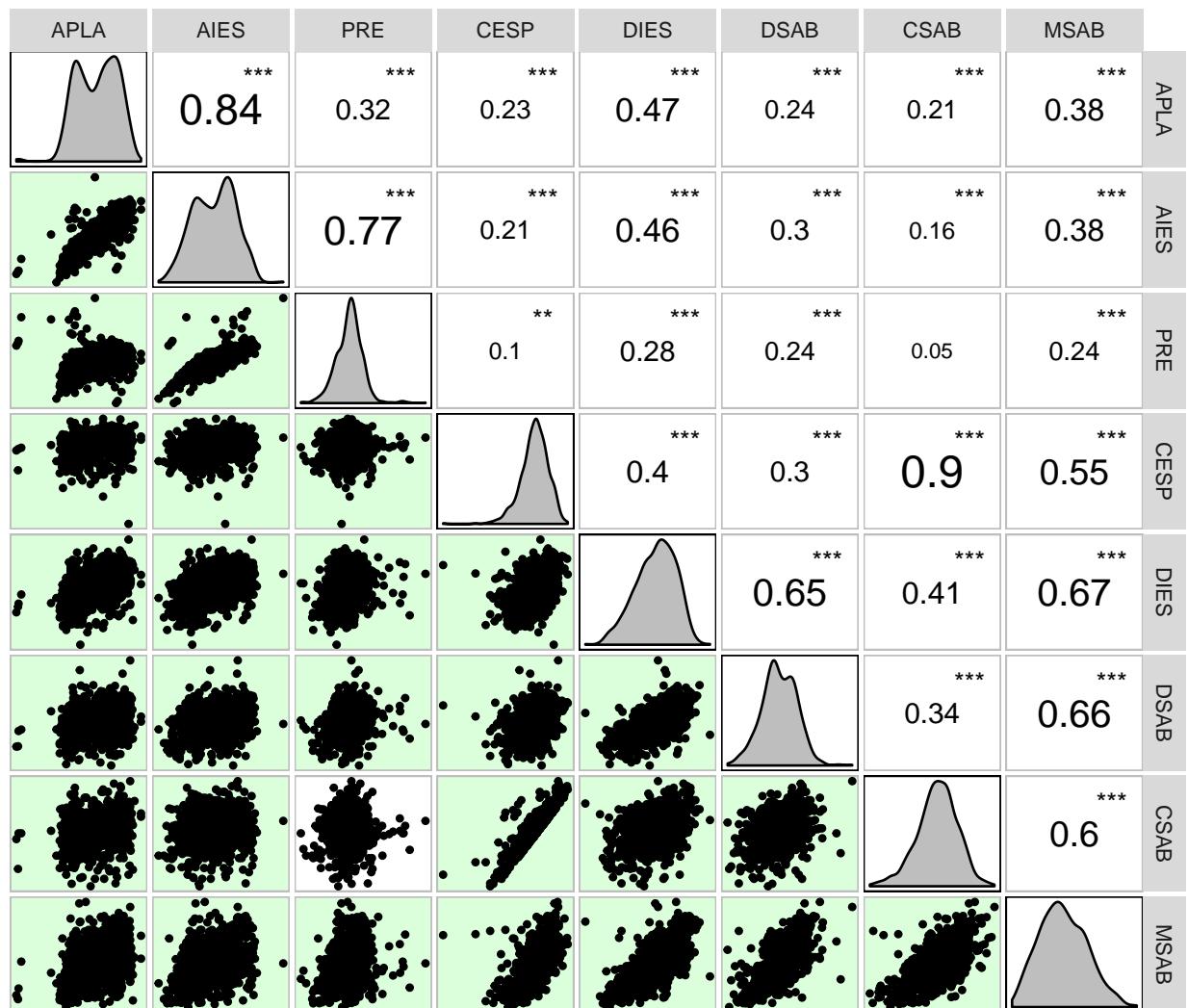


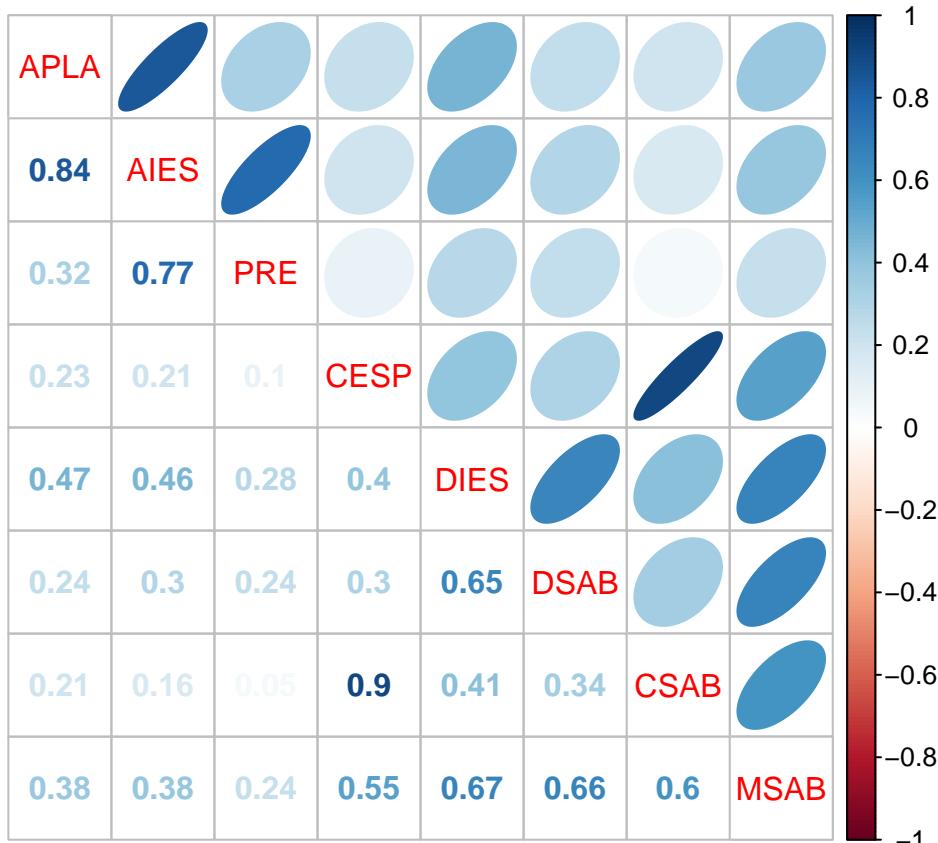
Figure 45: Scatter plot de uma matriz de correlação de Pearson

#### 4.3.5 Combinando visualização gráfica e numérica (III)

A função `corrplot.mixed()` do pacote `corrplot` também é uma boa opção para visualização gráfica. As correlações são indicadas por uma escala de cores e também pelo formato da elipse (quanto mais alta a correlação, mais fina a elipse)

```
library(corrplot)
# Criando a matrix de correlação utilizando o conjunto de dados 'datacor'.
# 8 variáveis / 26 combinações
correl = cor(datacor)

# Criando o gráfico
corrplot.mixed(correl,
                upper = "ellipse",
                lower = "number",
                number.digits = 2)
```



#### 4.3.6 Combinando visualização gráfica e numérica (IV)

```
library(corrplot)
# Criando a matrix de correlação utilizando o conjunto de dados dataset.
# 15 variáveis / 105 combinações
```

```
correl = cor(dataset)

# calculando os p-valores
pval = cor.mtest(dataset)$p
```

A função `corrplot()` do pacote `corrplot` permite uma poderosa personalização. Esta função tem a vantagem de apresentar um elevado número de combinações em um gráfico claro e intuitivo.

```
# Criando o gráfico
corrplot(correl,
         method = "pie",
         p.mat = pval,
         sig.level = 0.05,
         insig = "blank",
         type = "lower",
         diag = F,
         tl.col = "black",
         tl.srt = 45)
```

```
# Criando o gráfico
corrplot(correl,
         method = "ellipse",
         p.mat = pval,
         sig.level = 0.05,
         insig = "blank",
         type = "upper",
         diag = F,
         tl.col = "black",
         tl.srt = 45)
```

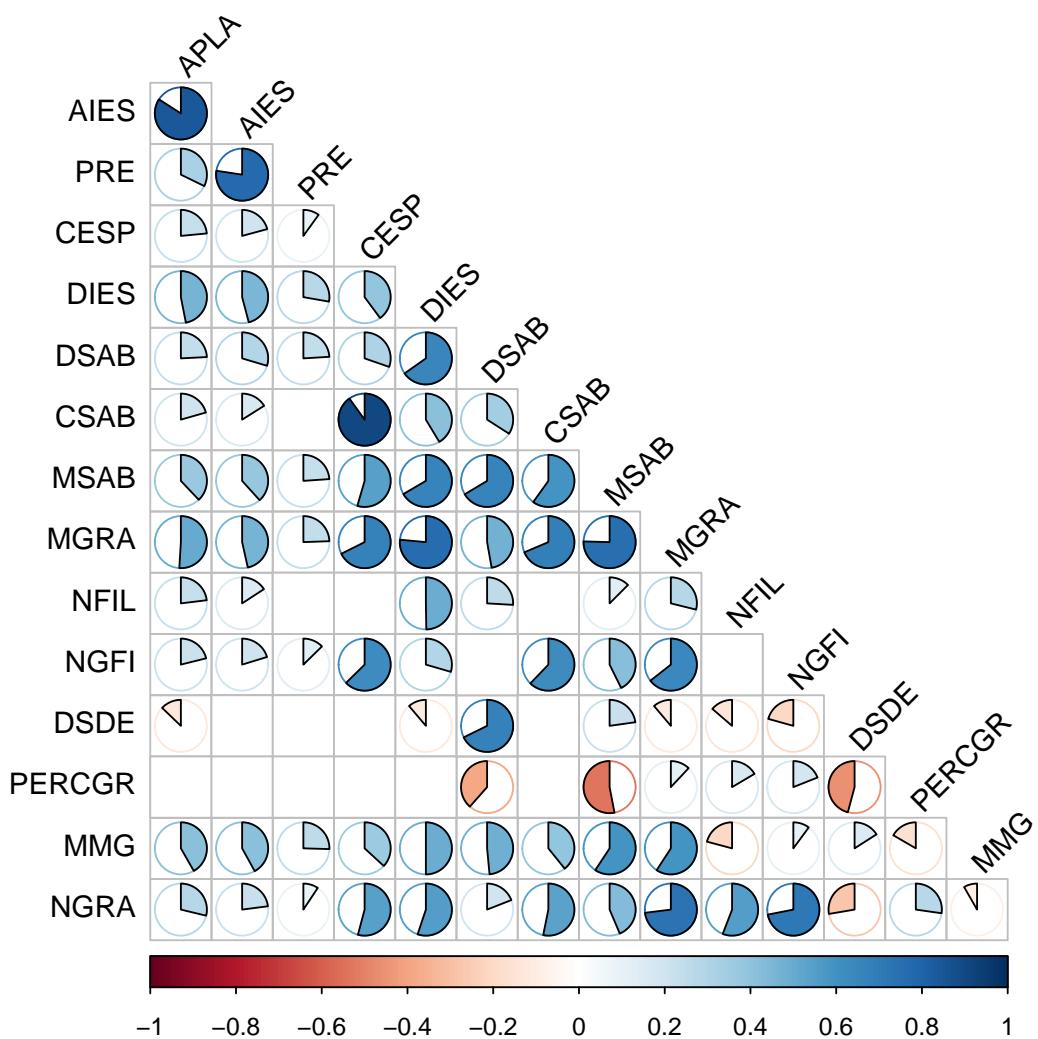
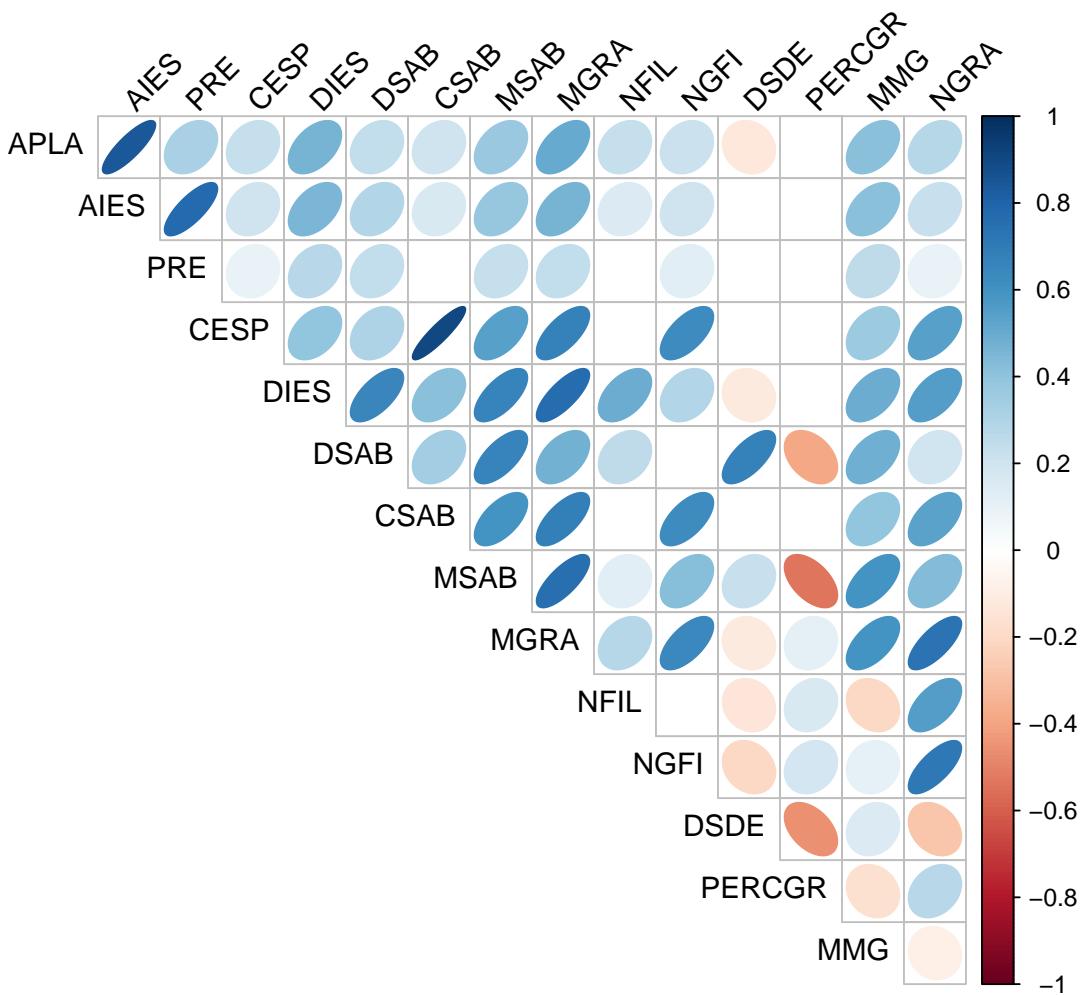


Figure 46: Gráfico de pizza de uma matriz de correlação de Pearson



Maiores detalhes de outros argumentos utilizados na função `corrplot()` podem ser encontrados aqui.

#### 4.3.7 Intervalo de confiança não paramétrico

A seguinte função computa o intervalo de confiança não paramétrico do coeficiente de correlação de Pearson de acordo com o seguinte modelo proposto por Olivoto et al. (2018).

$$CI = 0.45304^r \times 2.25152 \times n^{-0.50089}$$

onde  $CI$  é a semi-amplitude do intervalo de confiança do coeficiente de correlação (5% de erro);  $r$  é o coeficiente de correlação, em valor absoluto; e  $n$  é o tamanho amostral utilizado para estimar o coeficiente de correlação.

- Calculando o intervalo de confiança com base em um arquivo de variáveis

```

library(cursoR)
CI = CIcorr.mat(datacor)

# Imprimindo as primeiras 10 entradas
head(CI, n = 10)

##   Corr     CI    LL    UL
## 1  0.84  0.041  0.80  0.88
## 2  0.32  0.062  0.26  0.38
## 3  0.23  0.067  0.17  0.30
## 4  0.47  0.055  0.41  0.52
## 5  0.24  0.066  0.18  0.31
## 6  0.21  0.068  0.14  0.27
## 7  0.38  0.059  0.32  0.44
## 8  0.77  0.043  0.73  0.82
## 9  0.21  0.068  0.14  0.28
## 10 0.46  0.056  0.40  0.51

```

A saída acima retorna os seguintes valores: **Corr** = coeficiente de correção calculado; **CI** = semi-amplitude do intervalo de confiança (5% de erro); **LL** = limite inferior; **UL** = limite superior.

- Calculando o intervalo de confiança com base em valores declarados

```
CIcorr.val(r = 0.34, n = 50)
```

```

## -----
## Nonparametric 95% half-width confidence interval
## -----
## Level of significance: 5%
## Correlation coefficient: 0.34
## Sample size: 50
## Confidence interval: 0.2424
## True parameter range from: 0.0976 to 0.5824
## -----

```

#### 4.3.8 Planejamento do tamanho de amostra

A função **CIcorr.mat()** pode ser utilizada para planejar o tamanho da amostra necessário para calcular coeficiente de correlação de Pearson. com uma determinada semi-amplitude do intervalo de confiança 95%. A estimativa é baseada no seguinte modelo:

Com base em uma magnitude pré assumida de associação ( $r$ ), o tamanho da amostra necessário para uma semi-amplitude do intervalo de confiança 95% desejada pode ser calculado por:

$$n = [CI/(0.45304^r \times 2.25152)]^{-0.50089}$$

onde  $n$  é o tamanho da amostra calculado ;  $CI$  é a semi-amplitude do intervalo de confiança

desejado; e  $r$  é o coeficiente de correlação assumido. Olivoto et al. (2018) sugeriram considerar  $r$  nesta equação igual a 0. Uma vez que o tamanho da amostra será calculado para uma correlação nula, combinações com  $r > |0|$  apresentarão intervalo de confiança menor do que o informado no argumento  $CI$ .

```
corr.SS(CI = 0.1, r = 0)

## -----
## Sample size planning for correlation coefficient
## -----
## Level of significance: 5%
## Correlation coefficient: 0
## 95% half-width CI: 0.1
## Required sample size: 501
## -----
```

#### 4.4 Correlação parcial

Em certos casos, o coeficiente de correlação linear simples pode nos levar a equívocos na interpretação da associação entre duas variáveis, pois este não considera a influência das demais variáveis contidas no conjunto de dados. O coeficiente de correlação parcial é uma técnica baseada em operações matriciais que nos permite identificar a associação entre duas variáveis retirando-se os efeitos das demais variáveis presentes (Anderson 2003). Uma maneira generalizada para a estimativa do coeficiente de correlação parcial entre duas variáveis ( $i$  e  $j$ ) é por meio da matriz de correlação simples que envolve estas duas variáveis e  $m$  outras variáveis das quais queremos retirar o efeito. A estimativa do coeficiente de correlação parcial entre  $i$  e  $j$  excluído o efeito de  $m$  outras variáveis é dado por:

$$r_{ij.m} = \frac{-a_{ij}}{\sqrt{a_{ii}a_{jj}}}$$

onde  $r_{ij.m}$  é o coeficiente de correlação parcial entre as variável  $i$  e  $j$ , sem o efeito das outras  $m$  outras variáveis;  $a_{ij}$  é o elemento da ordem  $ij$  da inversa da matriz de correlação simples;  $a_{ii}$  e  $a_{jj}$  são os elementos de ordens  $ii$  e  $jj$ , respectivamente, da inversa da matriz de correlação simples.

A matriz de coeficientes de correlação parcial pode ser facilmente obtida utilizando a função `partial.corr()` disponível no pacote `cursoR`. A entrada dos dados pode ser realizada de duas maneiras. (i) utilizando os dados com as observações de cada variável; ou (ii) utilizando uma matriz de correlação linear simples pré estimada. Em nosso exemplo, vamos utilizar os mesmos dados utilizados nas funções anteriores (`datacor`).

```
partial = partial.corr(datacor)
```

A função `partial.corr()` retorna 4 objetos: `linear.mat` que contém a matriz de correlação linear simples; `partial.mat` que contém a matriz de correlações parciais, `results` que contém um data frame contendo todas as combinações de correlação com seus respectivos testes de hipótese e `call` que contém os argumentos informados na função. Vamos agora aos resultados.

```

options(digits = 2)
# correlação linear simples
partial$linear.mat

##      APLA AIES  PRE CESP DIES DSAB CSAB MSAB
## APLA 1.00 0.84 0.32 0.23 0.47 0.24 0.21 0.38
## AIES 0.84 1.00 0.77 0.21 0.46 0.30 0.16 0.38
## PRE  0.32 0.77 1.00 0.10 0.28 0.24 0.05 0.24
## CESP 0.23 0.21 0.10 1.00 0.40 0.30 0.90 0.55
## DIES 0.47 0.46 0.28 0.40 1.00 0.65 0.41 0.67
## DSAB 0.24 0.30 0.24 0.30 0.65 1.00 0.34 0.66
## CSAB 0.21 0.16 0.05 0.90 0.41 0.34 1.00 0.60
## MSAB 0.38 0.38 0.24 0.55 0.67 0.66 0.60 1.00

# correlação parcial
partial$partial.mat

##          APLA     AIES     PRE     CESP     DIES     DSAB     CSAB     MSAB
## APLA 1.0000 0.9833 -0.9564 0.0111 0.1529 -0.097 -0.0051 0.0205
## AIES 0.9833 1.0000 0.9805 0.0016 -0.0963 0.068 -0.0087 0.0060
## PRE -0.9564 0.9805 1.0000 0.0183 0.1042 -0.050 -0.0197 0.0069
## CESP 0.0111 0.0016 0.0183 1.0000 0.0415 -0.048 0.8563 -0.0232
## DIES 0.1529 -0.0963 0.1042 0.0415 1.0000 0.406 0.0084 0.2424
## DSAB -0.0971 0.0681 -0.0504 -0.0477 0.4058 1.000 -0.0154 0.4138
## CSAB -0.0051 -0.0087 -0.0197 0.8563 0.0084 -0.015 1.0000 0.2693
## MSAB 0.0205 0.0060 0.0069 -0.0232 0.2424 0.414 0.2693 1.0000

# resultados
partial$results

##          linear partial      t    prob
## APLA x AIES 0.84 0.9833 150.348 0.0e+00
## APLA x PRE  0.32 -0.9564 -90.988 0.0e+00
## APLA x CESP 0.23 0.0111   0.307 7.6e-01
## APLA x DIES 0.47 0.1529   4.298 1.9e-05
## APLA x DSAB 0.24 -0.0971  -2.709 6.9e-03
## APLA x CSAB 0.21 -0.0051  -0.141 8.9e-01
## APLA x MSAB 0.38 0.0205   0.570 5.7e-01
## AIES x PRE  0.77 0.9805 138.713 0.0e+00
## AIES x CESP 0.21 0.0016   0.043 9.7e-01
## AIES x DIES 0.46 -0.0963 -2.687 7.4e-03
## AIES x DSAB 0.30 0.0681   1.896 5.8e-02
## AIES x CSAB 0.16 -0.0087 -0.241 8.1e-01
## AIES x MSAB 0.38 0.0060   0.166 8.7e-01
## PRE x CESP  0.10 0.0183   0.509 6.1e-01
## PRE x DIES  0.28 0.1042   2.912 3.7e-03
## PRE x DSAB  0.24 -0.0504 -1.403 1.6e-01
## PRE x CSAB  0.05 -0.0197 -0.547 5.8e-01

```

```

## PRE x MSAB    0.24  0.0069   0.191 8.5e-01
## CESP x DIES   0.40  0.0415   1.155 2.5e-01
## CESP x DSAB   0.30 -0.0477  -1.328 1.8e-01
## CESP x CSAB   0.90  0.8563   46.073 0.0e+00
## CESP x MSAB   0.55 -0.0232  -0.644 5.2e-01
## DIES x DSAB   0.65  0.4058   12.337 0.0e+00
## DIES x CSAB   0.41  0.0084   0.233 8.2e-01
## DIES x MSAB   0.67  0.2424   6.943 8.1e-12
## DSAB x CSAB   0.34 -0.0154  -0.428 6.7e-01
## DSAB x MSAB   0.66  0.4138   12.630 0.0e+00
## CSAB x MSAB   0.60  0.2693   7.769 2.5e-14

```

## 4.5 Análise de trilha

Nesta sessão, primeiramente uma breve introdução à análise de trilha é apresentada. Algumas dificuldades, como , por exemplo, a presença de multicolinearidade e possíveis soluções para contorná-la serão discutidas. Posteriormente exemplos numéricos serão realizados utilizando funções do pacote **cursoR**.

### 4.5.1 Introdução

A análise de trilha vem se destacando na área do melhoramento genético, pois a seleção para melhoria de um caractere desejável que possui difícil mensuração e baixa herdabilidade, pode ser realizada indiretamente por outro caractere, diretamente associado a este, mas que apresente alta herdabilidade e seja de fácil mensuração. Esta técnica é baseada em ideias originalmente desenvolvidas por Sewall Wright (Wright 1921), no entanto desde sua concepção até a consolidação do método, algumas divergências quanto a fidedignidade do método matemático que explica as relações de causa e efeito foram observadas. Em 1922, Henry E. Niles, em um artigo publicado na revista *Genetics* intitulado *Correlation, Causation and Wright's theory of path coefficients*, fez uma crítica ao método proposto por Wright, afirmando que a base filosófica do método dos coeficientes de trilha era falha. Niles, testando o método de Wright, evidenciou em alguns de seus resultados coeficientes superiores a |1|, afirmando [...] *estes resultados são ridículos* e que Wright teria de fornecer provas bem mais convincentes do que ele estava apresentando Niles (1922).

No ano seguinte, Sewall Wright em seu artigo intitulado *The theory of path coefficients: a reply to Niles's criticism*, publicado na mesma revista *Genetics*, consolida seu método ao concluir que Niles pareceu se basear em conceitos matemáticos incorretos, resultado de uma falha em reconhecer que um coeficiente de trilha não é uma função simétrica de duas variáveis, mas que ele necessariamente tem direção. Este autor conclui seu trabalho afirmando que a análise de trilha não fornece uma fórmula geral para deduzir relações causais a partir do conhecimento das correlações. Ela é, no entanto, dentro de certas limitações, um método de avaliar as consequências lógicas de uma hipótese de relação causal em um sistema de variáveis correlacionadas. Acrescenta ainda que as críticas oferecidas por Niles em nada invalidam a teoria do método ou sua aplicação (Wright 1923). Atualmente, o método estatístico é consolidado, e utilizado mundialmente em diversas áreas da ciência.

#### 4.5.2 Estimação

A decomposição das correlações lineares em efeitos diretos e indiretos de um conjunto de  $p$ -variáveis explicativas pode ser realizada a partir da derivação do sistema de equações normais  $X'X\beta = X'Y$  utilizado para a estimativa dos parâmetros de regressão múltipla do modelo. Assim, a estimativa de  $\beta$  é dada por:  $\hat{\beta} = X'X^{-1}X'Y$ , onde  $\beta$  é o vetor dos coeficiente de regressão parcial ( $\beta_1, \beta_2, \beta_3, \dots, \beta_p$ ) para  $p + 1$ ;  $X'X^{-1}$  é a inversa da matriz de correlação linear entre as variáveis explicativas e  $X'Y$  é a matriz de correlação de cada variável explicativa, com a variável dependente.

Após a estimativa dos coeficientes de regressão ( $\beta_p$ ), os efeitos diretos e indiretos do conjunto de  $p$ -variáveis explicativas podem ser estimados. Considere o seguinte exemplo, onde um conjunto de variáveis explicativas ( $a, b, c$  e  $d$ ) são utilizadas para explicar as relações de causa e efeito na resposta de uma variável dependente ( $y$ ). Após as estimativas dos coeficientes de regressão parcial ( $\beta_1, \beta_2, \beta_3, \beta_4$ ), os efeitos diretos e indiretos de  $a$  sobre  $y$  são dados por:  $r_{a:y} = \beta_1 + \beta_{2_{ra:b}} + \beta_{3_{ra:c}} + \beta_{4_{ra:d}}$  onde  $r_{a:y}$  é a correlação linear entre  $a$  e  $y$ ,  $\beta_1$  é o efeito direto de  $a$  em  $Y$ ;  $\beta_{2_{ra:b}}$  é o efeito indireto de  $a$  em  $y$  via  $b$ ,  $\beta_{3_{ra:c}}$  é o efeito indireto de  $a$  em  $y$  via  $c$  e  $\beta_{4_{ra:d}}$  é o efeito indireto de  $a$  sobre  $y$  via  $d$ . Regressões semelhantes são utilizadas para estimativa dos efeitos de  $b, c$  e  $d$ .

#### 4.5.3 Multicolinearidade

Embora esta análise revele associações de causa e efeito, sua estimativa é baseada em princípios de regressão múltipla. Assim, as estimativas dos parâmetros podem ser enviesadas devido a natureza complexa dos dados, em que a resposta da variável dependente está ligada a um grande número de variáveis explicativas, que são muitas vezes correlacionadas ou multicolineares entre si (Graham 2003). Assim, sempre que duas supostas variáveis explicativas se apresentam altamente associadas, é difícil estimar as relações de cada variável explicativa individualmente, uma vez que vários parâmetros resolvem o sistema de equações normais. A esta particularidade é atribuída o nome de multicolinearidade (Blalock 1963).

Os principais meios utilizados para identificar o grau de multicolinearidade em uma matriz de variáveis explicativas são os seguintes:

- Número de condição (CN): O número de condição é calculado pela razão entre o maior e menor autovalor ( $\lambda$ ) da matriz de correlação  $X'X$ , de acordo com a expressão

$$NC = \frac{\lambda_{\text{Max}}}{\lambda_{\text{Min}}}$$

O grau de multicolinearidade é considerado fraco se  $NC \leq 100$ , moderado se  $100 \leq NC \leq 1000$  e severo quando  $NC > 1000$ .

- Determinante da matriz  $X'X$  (D): O determinante da cada matriz de correlação é estimado pelo produto de seus respectivos autovalores, para  $\lambda_j > 0$ , de acordo com a expressão

$$D_{\mathbf{x}'\mathbf{x}} = \prod_{j=1}^p \lambda_j$$

Um determinante muito próximo a zero indica dependência linear entre as características explicativas, indicando problemas graves de multicolinearidade.

- Fator de inflação de variância (VIF): Os (VIFs) são utilizados para medir o quanto a variância dos coeficientes de regressão estimados ( $\beta_k$ ) foi inflada em comparação a quando os caracteres explicativos não são linearmente associados. A estimativa do VIF do  $k$ -ésimo elemento de  $\beta$  é dada pela soma dos quocientes do quadrado de cada componente do autovetor pelo seu respectivo autovalor associado, de acordo com a expressão

$$\text{VIF}_{\beta_k} = \left( \frac{(\text{EV}_{KC1})^2}{\lambda_1} + \frac{(\text{EV}_{KC2})^2}{\lambda_2} + \dots + \frac{(\text{EV}_{KCp})^2}{\lambda_p} \right)$$

onde  $\text{VIF}_{\beta_k}$  é o fator de inflação de variância o  $k$ -ésimo elemento de  $\beta$  para  $k = 1, 2, \dots, p$ ;  $\text{EV}_{KC1}$  é o componente do  $k$ -ésimo autovetor para  $k = 1, 2, \dots, p$  e  $C = 1, 2, \dots, p$ ; e  $\lambda$  é o autovalor associado ao respectivo autovetor para  $\lambda = 1, 2, \dots, p$ . Os VIFs também podem ser considerados como os elementos da diagonal da inversa da matriz  $X'X$ . Considera-se que a presença de VIFs maiores que 10 é um indicativo de multicolinearidade .

#### 4.5.4 Métodos para ajustar a multicolinearidade

Embora os problemas relacionados a multicolinearidade se apresente como uma dificuldade na estimativa de coeficientes de trilha, algumas medidas podem ser tomadas visando mitigar seus efeitos indesejáveis, quando esta for detectada pelos métodos acima citados. Sabe-se atualmente, que a exclusão das variáveis responsáveis por inflar a variância de um coeficiente de regressão é um dos métodos mais indicados para reduzir a multicolinearidade em matrizes de variáveis explicativas (Olivoto T., Souza, et al. 2017). A identificação destas variáveis, no entanto, pode ser uma tarefa difícil. Recentemente, Olivoto T., Nardino, et al. (2017) propuseram a utilização de procedimentos stepwise juntamente com análise de trilha sequencial visando identificar um conjunto de variáveis com alto poder explicativo, mas que não se apresentem altamente correlacionadas. Quando a exclusão das variáveis responsáveis não é um procedimento considerado pelo pesquisador, por exemplo, devido a um número reduzido de variáveis explicativa, ou pela importância em conhecer seus efeitos, uma terceira opção é realizar a análise de trilha com todas as variáveis explicativas, porém com a inclusão de um pequeno valor nos elementos da diagonal  $X'X$ , conhecida como regressão em crista. Este procedimento, no entanto superestima os efeitos diretos, principalmente daquelas variáveis com alto VIF. (Olivoto T., Souza, et al. 2017).

#### 4.5.5 Análise tradicional

Esta sessão está focada principalmente em três objetivos. (i) diagnóstico da multicolinearidade ; (ii) seleção de variáveis preditoras; e (iii) estimação dos coeficientes de trilha. Embora esta

seja a sequência correta a ser seguida para a estimativa dos coeficientes de trilha, utilizaremos somente a função `path.coeff()`, que possibilita todas estas abordagens. Para isto, os conjunto de dados `dataset` e `datafactors` serão utilizados.

```
library(cursoR)
pathtodas = path.coeff(dataset, resp = "MGRA")
```

```
## Multicolinearidade severa! NC = 2643.543. Considere incluir um fator de correção ou d
```

Com a função acima, estimamos os coeficientes de trilha considerando a variável *MGRA* como dependente, e todas as outras variáveis restantes do conjunto de dados *dataset* como explicativas. A função `summary.path.coeff()` é utilizada para resumir os resultados da análise.

```
summary.path.coeff(pathtodas, digits = 2)
```

```
## -----
## Multicollinearity diagnosis and goodness-of-fit
## -----
## Condition number = 2643.5429
## Determinant = 0
## R-square = 0.9792
## Residual = 0.0208
## Response = MGRA
## -----
## 
## -----
## Variance inflation factors
## -----
##          VIF
## APLA    44.7
## AIES    96.5
## PRE     31.6
## CESP    5.8
## DIES    142.9
## DSAB    244.0
## CSAB    6.3
## MSAB    17.6
## NFIL    2.9
## NGFI    3.6
## DSDE    141.0
## PERCGR   7.3
## MMG     7.8
## NGRA    13.6
## -----
## 
## -----
## Eigenvalues and eigenvectors
```

```

## -----
##   Eigenvalues    APLA     AIES      PRE     CESP     DIES     DSAB     CSAB
## 1      5.0091 -0.274 -0.2845 -0.188 -0.3218 -0.36080 -0.2920 -0.3243
## 2      2.5455  0.021 -0.0530 -0.107  0.1391  0.02102 -0.3453  0.1084
## 3      1.9767  0.399  0.4956  0.409 -0.3234  0.10948 -0.0773 -0.3663
## 4      1.4659 -0.108 -0.1851 -0.191 -0.2187  0.27573  0.3289 -0.1908
## 5      0.8734  0.201 -0.1792 -0.552 -0.0478  0.32156 -0.0387 -0.0073
## 6      0.7407  0.060  0.0065 -0.053 -0.1620  0.00068 -0.3059 -0.1088
## 7      0.5696  0.662  0.1788 -0.475  0.0820 -0.36297 -0.0424  0.0718
## 8      0.3922  0.049 -0.0647 -0.168 -0.4660  0.05142  0.1684 -0.4004
## 9      0.1775 -0.111 -0.0464  0.080 -0.0715 -0.45180 -0.2715 -0.0124
## 10     0.1188 -0.037 -0.0029  0.039 -0.1250 -0.26614 -0.1401  0.0031
## 11     0.0933 -0.016 -0.0301 -0.028  0.6710 -0.03087 -0.0065 -0.7313
## 12     0.0294  0.027 -0.0654  0.037 -0.0370  0.02821 -0.0445 -0.0040
## 13     0.0059  0.504 -0.7480  0.422  0.0037  0.02346 -0.0250  0.0029
## 14     0.0019 -0.026  0.0345 -0.018  0.0039  0.51606 -0.6784 -0.0093
##   MSAB     NFIL     NGFI     DSDE    PERCGR     MMG     NGRA
## 1 -0.377 -0.1335 -0.2611 -0.0368  0.0621 -0.265 -0.284
## 2 -0.180  0.2152  0.3230 -0.4706  0.4522 -0.263  0.396
## 3 -0.150  0.1706 -0.2115 -0.2094  0.1490  0.037 -0.087
## 4  0.044  0.6574 -0.1894  0.1618 -0.0841 -0.299  0.238
## 5  0.061  0.0049 -0.2786 -0.3589  0.1162  0.508 -0.185
## 6  0.375 -0.0354  0.1208 -0.4079 -0.7027 -0.196  0.082
## 7 -0.072  0.0603  0.0203  0.3109 -0.0728 -0.216  0.022
## 8  0.144 -0.2907  0.5822  0.1588  0.1924  0.142  0.173
## 9  0.159  0.5870  0.2189  0.1086 -0.0039  0.501 -0.123
## 10     0.432 -0.1938 -0.5162  0.0885  0.2248  0.091  0.578
## 11     0.064 -0.0061 -0.0188  0.0087 -0.0194  0.053  0.065
## 12    -0.640 -0.0546 -0.0223  0.0688 -0.4000  0.376  0.519
## 13     0.048 -0.0068 -0.0031  0.0219  0.0323 -0.038 -0.045
## 14     0.047 -0.0041  0.0057  0.5149  0.0282 -0.036 -0.042
## -----
## 
## 
## -----
## Variables with the largest weight in the eigenvalue of smallest magnitude
## -----
## [1] "DSAB > DIES > DSDE > MSAB > NGRA > MMG > AIES > PERCGR > APLA > PRE > CSAB > NGF"
## -----
## 
## 
## -----
## Direct (diagonal) and indirect (off-diagonal) effects
## -----
##   APLA     AIES      PRE     CESP     DIES     DSAB     CSAB
## APLA  0.10001  0.08409  0.03224  0.02350  0.04694  0.02419  0.02059
## AIES -0.09812 -0.11671 -0.09033 -0.02428 -0.05356 -0.03456 -0.01875
## PRE   0.02129  0.05113  0.06606  0.00658  0.01833  0.01588  0.00329

```

```

## CESP  0.00213  0.00189  0.00090  0.00907  0.00362  0.00275  0.00817
## DIES  0.12759  0.12476  0.07545  0.10835  0.27187  0.17730  0.11242
## DSAB -0.08438 -0.10329 -0.08387 -0.10575 -0.22747 -0.34881 -0.11885
## CSAB  0.00167  0.00130  0.00040  0.00731  0.00336  0.00277  0.00812
## MSAB  0.19469  0.19647  0.12267  0.27972  0.34148  0.34079  0.30727
## NFIL -0.00142 -0.00097 -0.00018 -0.00016 -0.00307 -0.00160 -0.00012
## NGFI  0.00071  0.00068  0.00043  0.00211  0.00099  0.00019  0.00209
## DSDE -0.03221 -0.01316  0.01109  0.00549 -0.02801  0.16990  0.01462
## PERCGR 0.01105 -0.00458 -0.01907  0.00942 -0.01504 -0.12691 -0.01196
## MMG   0.14716  0.14798  0.08976  0.12991  0.17632  0.17109  0.13826
## NGRA  0.11948  0.09534  0.03871  0.22505  0.22919  0.07914  0.22103
##      MSAB    NFIL    NGFI    DSDE    PERCGR    MMG    NGRA
## APLA  0.03795  0.02308  0.02110 -0.01285  0.00335  0.04173  0.0288
## AIES -0.04469 -0.01843 -0.02365  0.00612  0.00162 -0.04897 -0.0268
## PRE   0.01579  0.00197  0.00837  0.00292 -0.00382  0.01681  0.0062
## CESP  0.00495  0.00024  0.00568  0.00020  0.00026  0.00334  0.0049
## DIES  0.18093  0.13523  0.08020 -0.03036 -0.01240  0.13592  0.1499
## DSAB -0.23166 -0.09040 -0.01992 -0.23630  0.13425 -0.16922 -0.0664
## CSAB  0.00486  0.00015  0.00505  0.00047 -0.00029  0.00318  0.0043
## MSAB  0.51313  0.06414  0.21901  0.11688 -0.27206  0.30414  0.2237
## NFIL -0.00077 -0.00616 -0.00038  0.00084 -0.00102  0.00129 -0.0034
## NGFI  0.00144  0.00021  0.00336 -0.00070  0.00064  0.00034  0.0024
## DSDE  0.05713 -0.03437 -0.05186  0.25079 -0.11494  0.04000 -0.0694
## PERCGR -0.17482  0.05437  0.06263 -0.15112  0.32972 -0.05482  0.0898
## MMG   0.20903 -0.07359  0.03566  0.05625 -0.05864  0.35266 -0.0292
## NGRA  0.18119  0.23215  0.29887 -0.11500  0.11322 -0.03447  0.4156
## -----

```

De acordo com o NC, VIF e Determinante da matriz, a multicolinearidade na matriz das variáveis explicativa é severa. Por exemplo, foram observados oito VIFs > 10. A saída indica quais são as variáveis que apresentam maior peso último autovalor. Em ordem de importância, estas variáveis são: NGFI > PERCGR > AIES > APLA > CESP > DIES > MSAB > NFIL > PRE > DSAB > MMG > NGRA > DSDE > CSAB. Conforme discutido, temos basicamente duas opções para contornar os problemas da elevada multicolinearidade em nossos dados. Excluir as variáveis responsáveis pela multicolinearidade, ou manter todas as variáveis e incluir um fator de correção na diagonal da matrix  $X'X$ . Vamos começar pela última opção.

#### 4.5.6 Incluindo um fator de correção (k)

A variância dos coeficientes de regressão é reduzida quando quando um valor  $k$  para  $0 < k \leq 1$  é incluído na diagonal da matriz de correlação  $X'X$ . Esta técnica é conhecida como regressão em crista, ou *ridge regression*, em Inglês (Hoerl and Kennard 1976). Com esta técnica, a estimativa dos coeficientes de regressão é dado por:  $\beta = (X'X + Ik)^{-1}X'Y$ . A escolha da magnitude de  $k$ , no entanto, deve ser cautelosa. Sugere-se que o valor a ser incluído seja aquele menor possível que estabilize os coeficientes de regressão ( $\beta_p$ ). Felizmente, grande parte deste trabalho já foi realizado quando utilizamos a função `path.coeff()` na sessão anterior.

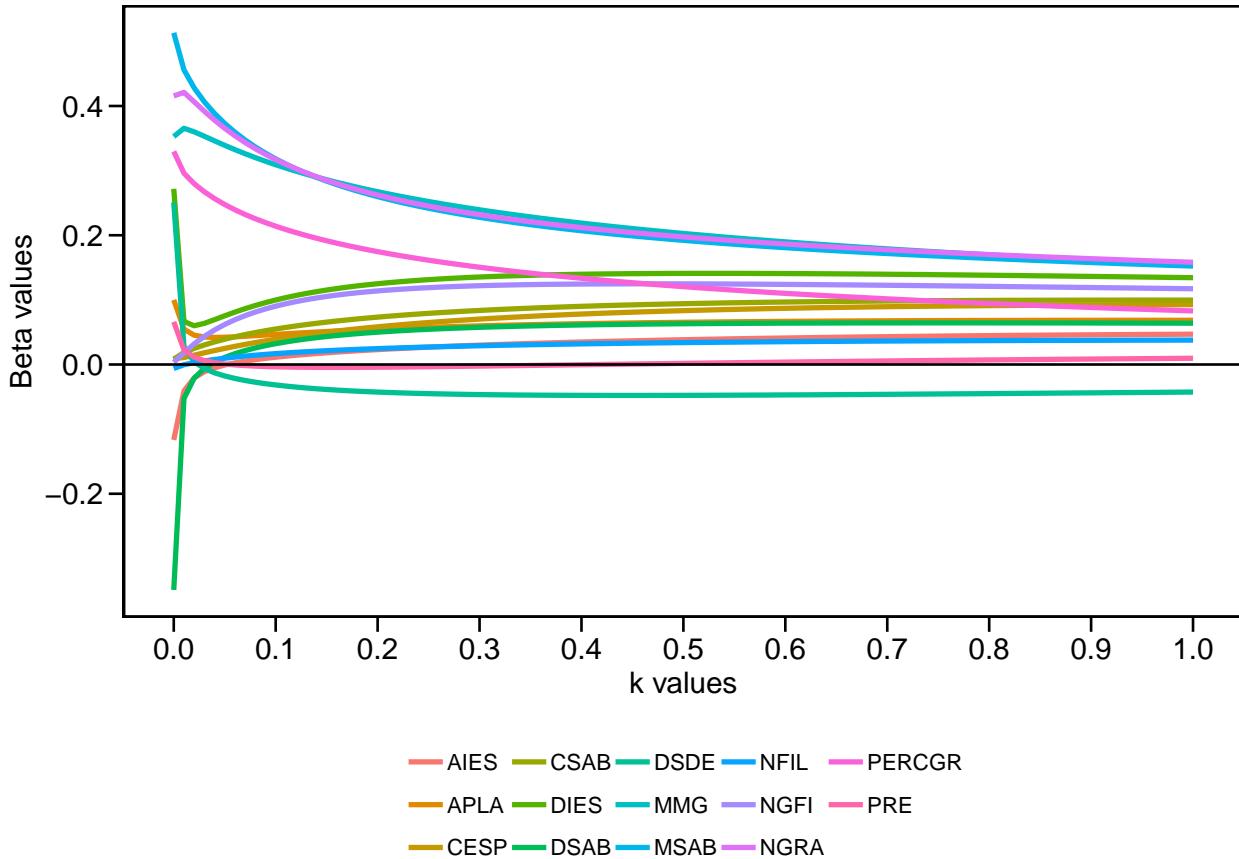


Figure 47: Valores de beta obtidos com 101 valores de  $k$

Um conjunto de estimativas de  $\beta_p$  foi estimado com 101 valores de  $k$ , ( $k = 0, 0.01, \dots, 1$ ) dois gráficos foram confeccionados. Um gráfico interativo e um gráfico e um gráfico que pode ser salvo como imagem ou pdf. Neste documento, apresentaremos o gráfico interativo gerado. Para vizualizarmos este gráfico é simples. Vamos fazer isto.

```
pathtodas$iterplot # gráfico interativo para visualização no R
```

Valores de beta obtidos com 101 valores de  $k$

```
pathtodas$plot
```

O gráfico acima nos proporciona uma interpretação sobre qual é o valor de  $k$  mais indicado a ser utilizado. Em nosso exemplo, no entanto, esta escolha ainda será difícil. Podemos notar que quatro variáveis apresentam valores de  $\beta$  maiores que o restante das variáveis. (para identificar estas variáveis, passe o cursor do mouse sobre as linhas. Também é possível isolar uma variável com um duplo clique na respectiva legenda). Para fins didáticos, escolheremos, por enquanto, o valor de  $k$  igual a 0.06, valor no qual os coeficientes de regressão da maioria das variáveis se estabiliza.

Para incluir este valor de correção, utilizaremos novamente a função `path.coeff()`, no entanto, agora, incluiremos o argumento `correction = 0.06`.

```
pathtodas_k = path.coeff(dataset,
                         resp = "MGRA",
                         correction = 0.06)
```

## A multicolinearidade pode ser considerada fraca. NC = 81.898. Observe os outros indicadores.

O aviso gerado no ajuste do modelo nos informou que a multicolinearidade agora pode ser considerada fraca, devido ao NC ser menor que 100. No entanto, é recomendado observar os outros indicadores. Para fazer isto, basta utilizar a função `summary.path.coeff()` novamente.

```
summary.path.coeff(pathtodas_k, digits = 2)
```

```
## -----
## Multicollinearity diagnosis and goodness-of-fit
## -----
## Condition number = 81.8985
## Determinant = 2.073e-05
## R-square = 0.9466
## Residual = 0.0534
## Response = MGRA
## -----
## 
## -----
## Variance inflation factors
## -----
##          VIF
## APLA    4.8
## AIES    8.8
## PRE     3.6
## CESP    3.7
## DIES    6.0
## DSAB    8.2
## CSAB    4.0
## MSAB    6.1
## NFIL    2.2
## NGFI    2.7
## DSDE    5.1
## PERCGR  2.9
## MMG     3.3
## NGRA    5.3
## -----
## 
## -----
## Eigenvalues and eigenvectors
## -----
##      Eigenvalues   APLA     AIES     PRE     CESP     DIES     DSAB     CSAB
## 1       5.069 -0.274 -0.2845 -0.188 -0.3218 -0.36080 -0.2920 -0.3243
```

```

## 2      2.606  0.021 -0.0530 -0.107  0.1391  0.02102 -0.3453  0.1084
## 3      2.037  0.399  0.4956  0.409 -0.3234  0.10948 -0.0773 -0.3663
## 4      1.526 -0.108 -0.1851 -0.191 -0.2187  0.27573  0.3289 -0.1908
## 5      0.933  0.201 -0.1792 -0.552 -0.0478  0.32156 -0.0387 -0.0073
## 6      0.801  0.060  0.0065 -0.053 -0.1620  0.00068 -0.3059 -0.1088
## 7      0.630 -0.662 -0.1788  0.475 -0.0820  0.36297  0.0424 -0.0718
## 8      0.452  0.049 -0.0647 -0.168 -0.4660  0.05142  0.1684 -0.4004
## 9      0.238 -0.111 -0.0464  0.080 -0.0715 -0.45180 -0.2715 -0.0124
## 10     0.179 -0.037 -0.0029  0.039 -0.1250 -0.26614 -0.1401  0.0031
## 11     0.153 -0.016 -0.0301 -0.028  0.6710 -0.03087 -0.0065 -0.7313
## 12     0.089  0.027 -0.0654  0.037 -0.0370  0.02821 -0.0445 -0.0040
## 13     0.066  0.504 -0.7480  0.422  0.0037  0.02346 -0.0250  0.0029
## 14     0.062 -0.026  0.0345 -0.018  0.0039  0.51606 -0.6784 -0.0093

##      MSAB    NFIL    NGFI    DSDE   PERCGR    MMG    NGRA
## 1     -0.377 -0.1335 -0.2611 -0.0368  0.0621 -0.265 -0.284
## 2     -0.180  0.2152  0.3230 -0.4706  0.4522 -0.263  0.396
## 3     -0.150  0.1706 -0.2115 -0.2094  0.1490  0.037 -0.087
## 4      0.044  0.6574 -0.1894  0.1618 -0.0841 -0.299  0.238
## 5      0.061  0.0049 -0.2786 -0.3589  0.1162  0.508 -0.185
## 6      0.375 -0.0354  0.1208 -0.4079 -0.7027 -0.196  0.082
## 7      0.072 -0.0603 -0.0203 -0.3109  0.0728  0.216 -0.022
## 8      0.144 -0.2907  0.5822  0.1588  0.1924  0.142  0.173
## 9      0.159  0.5870  0.2189  0.1086 -0.0039  0.501 -0.123
## 10     0.432 -0.1938 -0.5162  0.0885  0.2248  0.091  0.578
## 11     0.064 -0.0061 -0.0188  0.0087 -0.0194  0.053  0.065
## 12     -0.640 -0.0546 -0.0223  0.0688 -0.4000  0.376  0.519
## 13     0.048 -0.0068 -0.0031  0.0219  0.0323 -0.038 -0.045
## 14     0.047 -0.0041  0.0057  0.5149  0.0282 -0.036 -0.042

## -----
## 
## -----
## Variables with the largest weight in the eigenvalue of smallest magnitude
## -----
## [1] "DSAB > DIES > DSDE > MSAB > NGRA > MMG > AIES > PERCGR > APLA > PRE > CSAB > NGF
## -----
## 
## -----
## Direct (diagonal) and indirect (off-diagonal) effects
## -----
##      APLA     AIES     PRE     CESP     DIES     DSAB     CSAB
## APLA  0.04539  0.03600  0.01380  1.0e-02  0.02010  0.01036  8.8e-03
## AIES  0.00210  0.00264  0.00193  5.2e-04  0.00114  0.00074  4.0e-04
## PRE   -0.00022 -0.00053 -0.00072 -6.8e-05 -0.00019 -0.00016 -3.4e-05
## CESP  0.00659  0.00584  0.00280  3.0e-02  0.01118  0.00851  2.5e-02
## DIES  0.03805  0.03721  0.02250  3.2e-02  0.08594  0.05287  3.4e-02
## DSAB  0.00405  0.00495  0.00402  5.1e-03  0.01091  0.01773  5.7e-03

```

```

## CSAB 0.00882 0.00689 0.00213 3.9e-02 0.01773 0.01461 4.5e-02
## MSAB 0.13634 0.13759 0.08591 2.0e-01 0.23914 0.23865 2.2e-01
## NFIL 0.00264 0.00181 0.00034 3.1e-04 0.00569 0.00297 2.2e-04
## NGFI 0.01461 0.01403 0.00877 4.3e-02 0.02043 0.00396 4.3e-02
## DSDE 0.00277 0.00113 -0.00095 -4.7e-04 0.00241 -0.01463 -1.3e-03
## PERCGR 0.00802 -0.00333 -0.01384 6.8e-03 -0.01092 -0.09215 -8.7e-03
## MMG 0.13873 0.13951 0.08462 1.2e-01 0.16622 0.16129 1.3e-01
## NGRA 0.10174 0.08119 0.03297 1.9e-01 0.19516 0.06739 1.9e-01
##      MSAB    NFIL    NGFI    DSDE    PERCGR    MMG    NGRA
## APLA 0.01625 0.00988 9.0e-03 -0.00550 1.4e-03 0.01787 1.2e-02
## AIES 0.00095 0.00039 5.1e-04 -0.00013 -3.5e-05 0.00105 5.7e-04
## PRE -0.00016 -0.00002 -8.6e-05 -0.00003 3.9e-05 -0.00017 -6.3e-05
## CESP 0.01530 0.00075 1.8e-02 0.00061 8.0e-04 0.01034 1.5e-02
## DIES 0.05396 0.04033 2.4e-02 -0.00906 -3.7e-03 0.04054 4.5e-02
## DSAB 0.01111 0.00433 9.6e-04 0.01133 -6.4e-03 0.00811 3.2e-03
## CSAB 0.02567 0.00081 2.7e-02 0.00250 -1.6e-03 0.01681 2.3e-02
## MSAB 0.38091 0.04491 1.5e-01 0.08185 -1.9e-01 0.21299 1.6e-01
## NFIL 0.00143 0.01213 7.1e-04 -0.00157 1.9e-03 -0.00239 6.4e-03
## NGFI 0.02956 0.00432 7.3e-02 -0.01432 1.3e-02 0.00700 5.0e-02
## DSDE -0.00492 0.00296 4.5e-03 -0.02288 9.9e-03 -0.00344 6.0e-03
## PERCGR -0.12694 0.03948 4.5e-02 -0.10973 2.5e-01 -0.03981 6.5e-02
## MMG 0.19706 -0.06937 3.4e-02 0.05303 -5.5e-02 0.35241 -2.8e-02
## NGRA 0.15430 0.19769 2.5e-01 -0.09792 9.6e-02 -0.02935 3.8e-01
## -----

```

Com a inclusão do fator de correção ( $k = 0.06$ ) na diagonal de  $X'X$ , a multicolinearidade ficou à níveis aceitáveis, ou seja,  $NC < 100$  e nenhum  $VIF > 10$ . O determinante da matriz, no entanto, ainda é muito baixo. Inevitavelmente, temos agora duas opções para obtermos um determinante com valores mais aceitáveis. A primeira é aumentar o valor de  $k$ , digamos, para 0.1. Isto iria reduzir ainda mais o nível de multicolinearidade em nossa matriz, no entanto, o viés na estimativa dos coeficientes aumentaria. A segunda (e mais razoável) opção, é a exclusão de alguma variável do conjunto de variáveis explicativas. Por exemplo, podemos considerar as variáveis com maior peso nos últimos autovalores, ou aquelas com maior VIF. AIES e APLA apresentam alta correlação (veja a seção Estimativas dos coeficientes de correlação), assim poderíamos manter apenas uma destas variáveis na análise. PERCGR é uma co-variável ( $MGRA/(MGRA+MSAB)$ ). Assim, é razoável excluí-la também. A mesma interpretação pode ser considerada para DSDE. Esta é uma co-variável (razão do diâmetro do sabugo e diâmetro da espiga). Vamos considerar então a exclusão de três variáveis. PERCGRA, AIES e DSDE.

#### 4.5.7 Excluindo variáveis

O ajuste do novo modelo excluindo estas variáveis é facilmente realizado. Para isto, iremos utilizar dois argumentos da função `path.coeff()` não vistos até agora: `pred` e `exclude`. As variáveis informadas em `pred` podem ser as variáveis preditoras (*default*) ou as variáveis a serem excluídas, se `exclude = TRUE`. Vamos ao exemplo.

```

path_exclude = path.coeff(dataset,
                          resp = "MGRA",
                          pred = c("PERCGR", "AIES", "DSDE"),
                          exclude = TRUE)

## A multicolinearidade pode ser considerada fraca. NC = 56.513. Observe os outros indicadores
summary.path.coeff(path_exclude, digits = 2)

## -----
## Multicollinearity diagnosis and goodness-of-fit
## -----
## Condition number = 56.5128
## Determinant = 0.00032016
## R-square = 0.9643
## Residual = 0.0357
## Response = MGRA
## -----
## 
## -----
## Variance inflation factors
## -----
##      VIF
## APLA 1.6
## PRE  1.2
## CESP 5.8
## DIES 4.6
## DSAB 2.7
## CSAB 6.3
## MSAB 3.5
## NFIL 2.9
## NGFI 3.6
## MMG  3.8
## NGRA 7.2
## -----
## 
## -----
## Eigenvalues and eigenvectors
## -----
##      Eigenvalues   APLA     PRE     CESP     DIES     DSAB     CSAB     MSAB     NFIL
## 1      4.654 -0.246 -0.147 -0.3583 -0.374 -0.2933 -0.3635 -0.390 -0.146
## 2      1.792 -0.209 -0.284  0.2245 -0.164 -0.3479  0.2110 -0.165  0.160
## 3      1.577  0.180  0.094 -0.3190  0.293  0.1288 -0.3315 -0.076  0.681
## 4      0.992  0.458  0.683 -0.0377 -0.133 -0.3999 -0.1214 -0.162 -0.156
## 5      0.698  0.680 -0.608 -0.0078  0.026 -0.3111 -0.0063 -0.081  0.035
## 6      0.424  0.091  0.196  0.5023 -0.096 -0.0037  0.3918 -0.418  0.289
## 7      0.313 -0.409  0.053 -0.0178  0.489 -0.4831 -0.0416 -0.264  0.131

```

```

## 8      0.211 -0.081  0.093 -0.0184 -0.163 -0.5089  0.1016  0.710  0.162
## 9      0.164  0.077 -0.020  0.0618  0.559 -0.0757  0.0145 -0.030 -0.561
## 10     0.093  0.043  0.043 -0.6470  0.099 -0.0233  0.7267 -0.082  0.019
## 11     0.082 -0.035  0.021 -0.2063 -0.363  0.1450  0.0696 -0.164 -0.148
##      NGFI    MMG    NGRA
## 1   -0.296 -0.259 -0.322
## 2    0.414 -0.468  0.434
## 3   -0.166 -0.286  0.258
## 4    0.269 -0.032  0.080
## 5   -0.058  0.241 -0.031
## 6   -0.498 -0.055 -0.168
## 7    0.029  0.516  0.045
## 8   -0.393 -0.048  0.045
## 9   -0.363 -0.408  0.238
## 10   0.063 -0.087 -0.144
## 11   -0.314  0.356  0.724
##
## -----
## -----
## Variables with the largest weight in the eigenvalue of smallest magnitude
## -----
## [1] "NGRA > DIES > MMG > NGFI > CESP > MSAB > NFIL > DSAB > CSAB > APLA > PRE"
## -----
## -----
## Direct (diagonal) and indirect (off-diagonal) effects
## -----
##      APLA      PRE      CESP      DIES      DSAB      CSAB      MSAB
## APLA  0.01934  0.00623  0.00454  0.00908  0.00468  4.0e-03  0.00734
## PRE   -0.00051 -0.00158 -0.00016 -0.00044 -0.00038 -7.9e-05 -0.00038
## CESP   0.00094  0.00040  0.00401  0.00160  0.00121  3.6e-03  0.00218
## DIES   0.01455  0.00860  0.01235  0.03100  0.02022  1.3e-02  0.02063
## DSAB   -0.00411 -0.00409 -0.00515 -0.01109 -0.01700 -5.8e-03 -0.01129
## CSAB   0.00151  0.00037  0.00660  0.00303  0.00250  7.3e-03  0.00439
## MSAB   0.02219  0.01398  0.03189  0.03893  0.03885  3.5e-02  0.05849
## NFIL   -0.00328 -0.00042 -0.00038 -0.00707 -0.00369 -2.7e-04 -0.00178
## NGFI   0.00497  0.00298  0.01476  0.00695  0.00135  1.5e-02  0.01005
## MMG    0.24717  0.15077  0.21820  0.29614  0.28736  2.3e-01  0.35109
## NGRA   0.20687  0.06703  0.38967  0.39682  0.13703  3.8e-01  0.31373
##      NFIL      NGFI      MMG      NGRA
## APLA   4.5e-03  0.00408  0.0081  0.00556
## PRE   -4.7e-05 -0.00020 -0.0004 -0.00015
## CESP   1.1e-04  0.00251  0.0015  0.00217
## DIES   1.5e-02  0.00914  0.0155  0.01709
## DSAB   -4.4e-03 -0.00097 -0.0082 -0.00324
## CSAB   1.4e-04  0.00456  0.0029  0.00390

```

```

## MSAB 7.3e-03 0.02497 0.0347 0.02550
## NFIL -1.4e-02 -0.00089 0.0030 -0.00794
## NGFI 1.5e-03 0.02355 0.0024 0.01693
## MMG -1.2e-01 0.05989 0.5923 -0.04912
## NGRA 4.0e-01 0.51747 -0.0597 0.71963
## -----

```

Vamos à uma interpretação das três abordagens realizadas até agora, utilizando o resumo apresentado na tabela abaixo.

Estatística	Convencional	Incluindo $k$	Excluindo variáveis
Condition number	2655.0945	81.7588	51.9513
Determinant	0	2.224e-05	0.00037237
Largest VIF	245.4	8.8	6.3
R-square	0.9839	0.9522	0.9753
Residual	0.0161	0.0478	0.0247
Response	MGRA	MGRA	MGRA

Conforme também observado por Hoerl and Kennard (1970) e Olivoto T., Souza, et al. (2017), a exclusão de variáveis responsáveis pela multicolinearidade foi mais eficiente que a inclusão do  $k$ , proporcionando menores níveis de multicolinearidade e maior precisão do modelo (maior  $R^2$  e menor residual).

Vimos que tanto a identificação das variáveis responsáveis pela multicolinearidade quanto o ajuste do modelo declarando preditores específicos é um procedimento relativamente simples utilizando a função `path.coeff()`. Mas, e se algum procedimento estatístico-computacional facilitasse ainda mais essa tarefa? Vamos, a partir de agora, considerar isso.

Olivoto T., Nardino, et al. (2017) sugeriram a utilização de regressões stepwise para seleção de um conjunto de preditores com mínima multicolinearidade em análise de trilha. Esta opção está disponível na função `path.coeff()`. Baseado em um algoritmo heurístico iterativo executado pelo argumento `brutestepwise = TRUE`, um conjunto de preditores com mínima multicolinearidade é selecionado com base nos valores de VIF. Posteriormente, uma série de regressões stepwise são ajustadas. A primeira regressão stepwise é ajustada considerando  $p - 1$  variáveis preditoras selecionadas, sendo  $p$  o número de variáveis selecionadas no processo iterativo. O segundo modelo ajusta uma regressão considerando  $p - 2$  variáveis selecionadas, e assim por diante até o último modelo, que considera apenas duas variáveis selecionadas. Vamos ao exemplo.

#### 4.5.8 Utilizando regressões stepwise para seleção de variáveis

```

path_step = path.coeff(dataset,
                      resp = "MGRA",
                      brutstepwise = TRUE)

```

```

## The brutestepwise algorithm have selected a set of 11 predictors with largest VIF =

```

```

## Adjusting the model 1 with 10 predictor variables ( 11.11 % concluded)
## Adjusting the model 2 with 9 predictor variables ( 22.22 % concluded)
## Adjusting the model 3 with 8 predictor variables ( 33.33 % concluded)
## Adjusting the model 4 with 7 predictor variables ( 44.44 % concluded)
## Adjusting the model 5 with 6 predictor variables ( 55.56 % concluded)
## Adjusting the model 6 with 5 predictor variables ( 66.67 % concluded)
## Adjusting the model 7 with 4 predictor variables ( 77.78 % concluded)
## Adjusting the model 8 with 3 predictor variables ( 88.89 % concluded)
## Adjusting the model 9 with 2 predictor variables ( 100 % concluded)
## Done!
##
##
## -----
## Summary of the adjusted models
##
## -----
##      Model AIC Numpred   CN Determinant     R2 Residual maxVIF
## 2  Model 9 7077       2  1.8    0.9130  0.78    0.223    1.1
## 3  Model 8 5688       3  6.7    0.4538  0.96    0.037    2.2
## 4  Model 7 5684       4 20.7    0.1439  0.96    0.037    4.7
## 5  Model 6 5678       5 22.1    0.1247  0.96    0.037    5.1
## 6  Model 5 5675       6 30.1    0.0540  0.96    0.037    5.7
## 7  Model 4 5672       7 24.5    0.0460  0.96    0.036    4.0
## 8  Model 3 5669       8 36.7    0.0132  0.96    0.036    7.1
## 9  Model 2 5667       9 37.7    0.0091  0.96    0.036    7.1
## 10 Model 1 5669      10 44.6   0.0016  0.96    0.036    7.2
## -----

```

Três objetos são criados por esta função: `Summary`, `Models` e `Selectedpred`. O objeto `Summary` contém um resumo do procedimento, listando o número do modelo, o valor do AIC, o diagnóstico da multicolinearidade e os valores de R2 e residual. O objeto `Models`, contém todos os modelos ajustados, e o objeto `Selectedpred`, contém o nome das variáveis preditoras selecionadas no processo iterativo. Podemos notar que o algoritmo selecionou um conjunto com 11 preditores que apresenta multicolinearidade em níveis aceitáveis. Assim, qualquer um destes modelos poderia ser utilizado sem maiores problemas em relação à isto. O procedimento stepwise realizado com diferentes números de variáveis selecionados também permite a seleção de um modelo mais parcimonioso. Por exemplo, o modelo 8 (com três variáveis preditoras) apresentou o mesmo poder explicativo do modelo 1 (com dez variáveis preditoras), no entanto com menor nível de multicolinearidade. A seleção do modelo a ser utilizado ficará a critério do pesquisador. Por exemplo, pode ser interessante discutir os efeitos de variáveis não incluídas no modelo 8. Para fins didáticos, consideraremos como o modelo selecionado o modelo 6.

```

path_step$Models[["Model 6"]]$Predictors

## [1] "DIES"    "NGFI"    "MMG"     "NGRA"    "PERCGR"

```

Os coeficientes de trilha, bem como todos os outros indicadores são encontrados no objeto `path_step$Models[["Model 6"]]`. O pacote `cursoR` também conta com uma função para

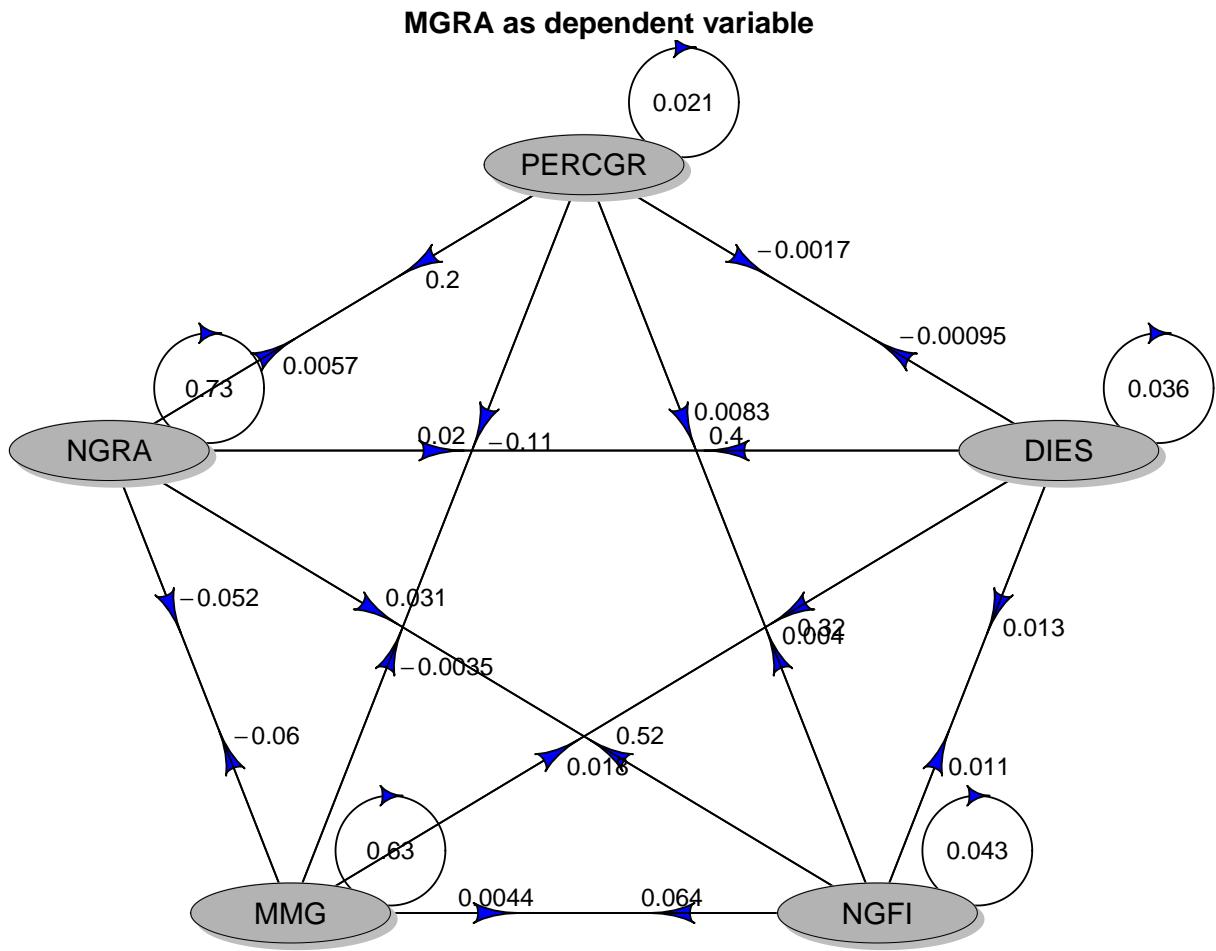


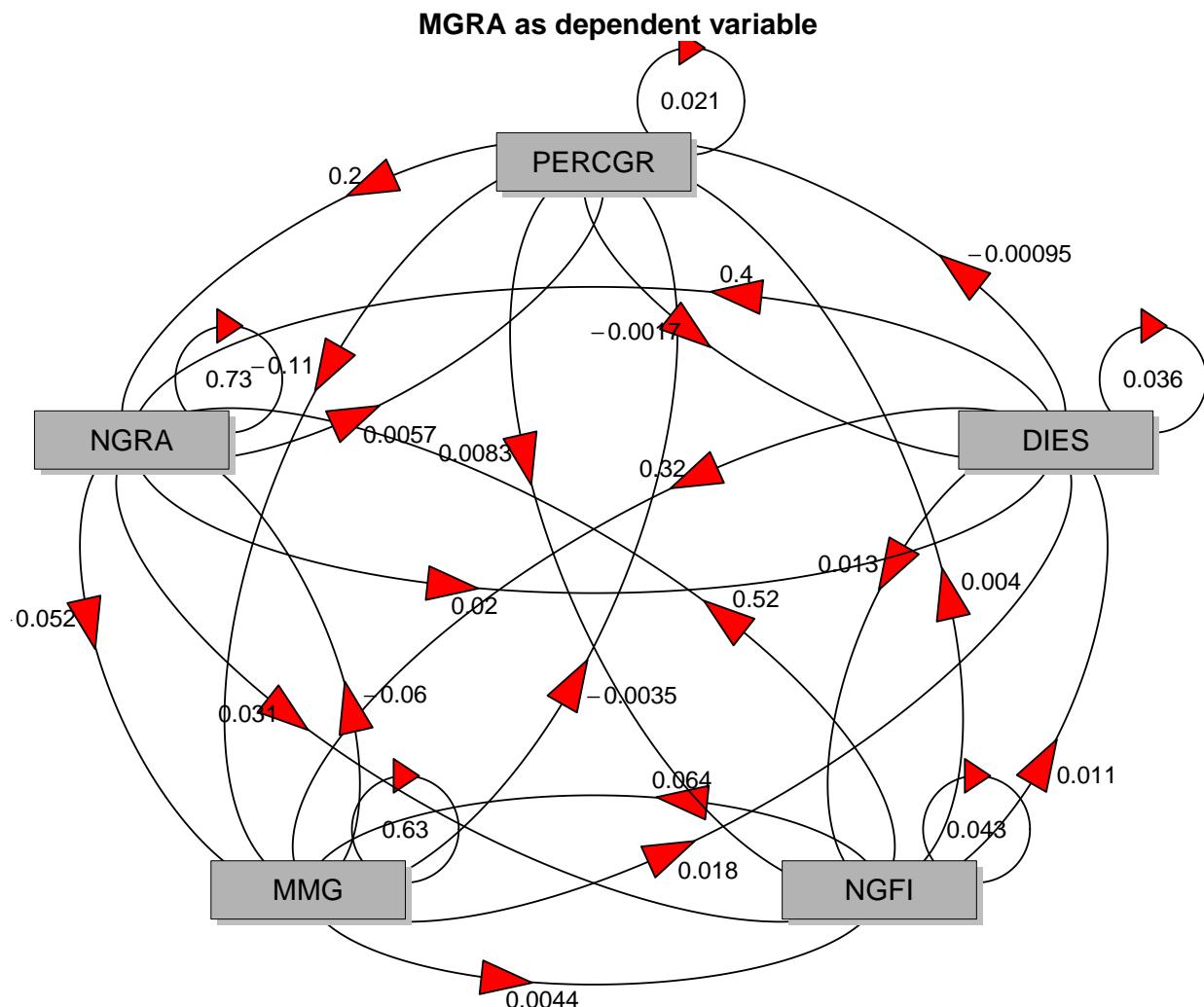
Figure 48: Diagrama de trilha utilizando a função path.diagram()

confecção de diagramas de trilha considerando um modelo ajustado. Vamos construir dois diagramas com o modelo 6, salvo em `path_step` utilizando a função `path.diagram()`.

```

# Criando o gráfico 1
path.diagram(path_step$Models[["Model 6"]],
             digits = 2,
             curve = 0,
             arr.col = "blue",
             arr.type = "curved",
             arr.pos = 0.35,
             arr.length = 0.5,
             arr.width = 0.2,
             box.prop = 0.3,
             box.type = "ellipse",
             box.col = "gray70")
  
```

```
# Criando o gráfico 2
path.diagram(path_step$Models[["Model 6"]],
  digits = 2,
  curve = 0.2,
  arr.col = "red",
  arr.type = "triangle",
  arr.pos = 0.4,
  arr.length = 0.6,
  arr.width = 0.4,
  box.prop = 0.3,
  box.type = "rect",
  box.col = "gray70")
```



Até agora utilizamos a função `path.coeff()` para realizar a análise de trila baseado em dados fenotípicos. É importante observar que o conjunto de dados *dataset* contém 780 observações, que são oriundos do experimento multi-ambiente conduzido com híbridos de milho (4 locais x

13 híbridos x 3 blocos x 5 plantas avaliadas por bloco).

Um fato importante e que merece atenção, é que a estrutura do nosso experimento precisa ser considerada na estimativa dos coeficientes de trilha. Assim, um modelo considerando o design experimental adequado precisa ser ajustado e a análise de trilha deve ser realizada com os valores preditos (ou com o residual do modelo).

Para implementar esta abordagem, iremos considerar um modelo misto, considerando genótipos como de efeito aleatório e ambiente como de efeito fixo. A função `WAASB()` do pacote `WAASB` será utilizada. Esta função é discutida em detalhes na sessão Ajustando o modelo. Primeiramente, precisaremos criar um novo conjunto de dados com a média das cinco plantas por parcela.

```
# declarando os fatores do modelo
library(cursoR)
datafactors = pass(data = datafactors,
                     var = c(1:4),
                     type = as.factor)

library(plyr)
# criando um arquivo com as médias das cinco plantas em cada repetição
MEDIAS = ddply(datafactors,
               ~AMB*HIB*REP,
               summarise,
               APLA = mean(APLA),
               AIES = mean(AIES),
               PRE = mean(PRE),
               CESP = mean(CESP),
               DIES = mean(DIES),
               DSAB = mean(DSAB),
               CSAB = mean(CSAB),
               MSAB = mean(MSAB),
               MGRA = mean(MGRA),
               NFIL = mean(NFIL),
               NGFI = mean(NGFI),
               DSDE = mean(DSDE),
               PERCGR = mean(PERCGR),
               MMG = mean(MMG),
               NGRA = mean(NGRA))

library(WAASB)

# Obtendo os valores preditos (via modelo misto)
APLA = WAASB(MEDIAS, resp = "APLA")$BLUPgge$Predicted
AIES = WAASB(MEDIAS, resp = "AIES")$BLUPgge$Predicted
PRE = WAASB(MEDIAS, resp = "PRE")$BLUPgge$Predicted
CESP = WAASB(MEDIAS, resp = "CESP")$BLUPgge$Predicted
DIES = WAASB(MEDIAS, resp = "DIES")$BLUPgge$Predicted
```

```

DSAB = WAASB(MEDIAS, resp = "DSAB")$BLUPgge$Predicted
CSAB = WAASB(MEDIAS, resp = "CSAB")$BLUPgge$Predicted
MSAB = WAASB(MEDIAS, resp = "MSAB")$BLUPgge$Predicted
MGRA = WAASB(MEDIAS, resp = "MGRA")$BLUPgge$Predicted
NGRA = WAASB(MEDIAS, resp = "NGRA")$BLUPgge$Predicted
NFIL = WAASB(MEDIAS, resp = "NFIL")$BLUPgge$Predicted
NGFI = WAASB(MEDIAS, resp = "NGFI")$BLUPgge$Predicted
DSDE = WAASB(MEDIAS, resp = "DSDE")$BLUPgge$Predicted
PERCGR = WAASB(MEDIAS, resp = "PERCGR")$BLUPgge$Predicted
MMG = WAASB(MEDIAS, resp = "MMG")$BLUPgge$Predicted

# Criando um arquivo com todas as variáveis
DATAMISTO = data.frame(cbind(APLA, AIES, PRE, CESP, DIES, DSAB, CSAB,
                               MSAB, MGRA, NGRA, NFIL, NGFI, DSDE, PERCGR, MMG))

```

Após criado o arquivo *DATAMISTO*, utilizaremos a função `path.coeff()` novamente, utilizando o algorítmo para seleção de variáveis preditoras.

```

path_misto = path.coeff(DATAMISTO,
                        resp = "MGRA",
                        brutstepwise = TRUE)

## The brutestepwise algorithm have selected a set of 8 predictors with largest VIF =
## Adjusting the model 1 with 7 predictor variables ( 16.67 % concluded)
## Adjusting the model 2 with 6 predictor variables ( 33.33 % concluded)
## Adjusting the model 3 with 5 predictor variables ( 50 % concluded)
## Adjusting the model 4 with 4 predictor variables ( 66.67 % concluded)
## Adjusting the model 5 with 3 predictor variables ( 83.33 % concluded)
## Adjusting the model 6 with 2 predictor variables ( 100 % concluded)
## Done!
##
##
## -----
## Summary of the adjusted models
## -----
##      Model AIC NumPred   CN Determinant    R2 Residual maxVIF
## 2 Model 6 386        2  2.8     0.770 0.86    0.138    1.3
## 3 Model 5 303        3  2.1     0.866 0.97    0.027    1.1
## 4 Model 4 301        4  4.9     0.537 0.98    0.025    1.6
## 5 Model 3 302        5  7.5     0.240 0.98    0.024    2.2
## 6 Model 2 303        6 21.2     0.084 0.98    0.024    3.6
## 7 Model 1 304        7 31.3     0.033 0.98    0.024    4.4
## -----

```

Neste ponto, realizamos uma análise de trilha utilizando dados genotípicos. Assim, tanto os efeitos de tratamentos e erro experimental foram considerados, diferentemente de quando utilizamos os dados fenotípicos.

## 4.6 Análise multivariada

No melhoramento genético de plantas, diversas variáveis são mensuradas em cada genótipo, visando maior segurança na distinção de tais genótipos. Embora em alguns casos possa fazer sentido isolar cada variável e estudá-la separadamente, geralmente, uma análise que englobe todas as variáveis fornece um maior número de informações. Como todo o conjunto de variáveis é medido em cada genótipo, as variáveis serão relacionadas em maior ou menor grau. Consequentemente, se cada variável é analisada isoladamente, a estrutura completa dos dados pode não ser revelada. A análise multivariada é a análise estatística simultânea de uma coleção de variáveis que utilizaria informações sobre as relações entre estas. É muito provável que a análise de cada variável separadamente não revele padrões interessantes que a análise multivariada proporciona.

A concepção da análise multivariada é provavelmente o trabalho realizado por Francis Galton e Karl Pearson no final do século XIX sobre a quantificação da relação entre descendentes e características parentais e o desenvolvimento do coeficiente de correlação Galton (1888). Naquele tempo, o processamento computacional era muito limitado para suportar o peso das vastas quantidades de aritmética envolvidas na aplicação dos métodos multivariados que estavam sendo propostos. Assim, os desenvolvimentos eram principalmente matemáticos e a pesquisa multivariada era, na época, em grande parte, um ramo de álgebra linear. No entanto, a chegada e rápida expansão do uso de computadores eletrônicos na segunda metade do século XX, levou à crescente aplicação prática dos métodos existentes de análise multivariada, renovando o interesse no desenvolvimento de novas técnicas.

Nos primeiros anos do século XXI, a ampla disponibilidade de computadores pessoais e laptops relativamente baratos e extremamente poderosos, aliados a softwares estatísticos flexíveis fez com que todos os métodos de análise multivariada pudessem ser aplicados rotineiramente, mesmo para grandes conjuntos de dados, como os gerados em um programa de melhoramento genético; por exemplo, dados de marcadores moleculares e sequenciamento gênico.

### 4.6.1 Componentes principais

O objetivo básico da análise de componentes principais é descrever a variação em um conjunto de variáveis correlacionadas  $x^T = (x_1, \dots, x_q)$ , em termos de um novo conjunto de variáveis não correlacionadas,  $y^T = (y_1, \dots, y_q)$ , onde cada variável é uma combinação linear das variáveis  $x$ . As novas variáveis são ordenadas em ordem decrescente de “importância”, no sentido de que  $y_1$  é responsável pelo máximo possível da variação dos dados originais entre todas as combinações lineares de  $x$ . Então  $y_2$  é escolhido para explicar o máximo possível da variação restante, sujeito a ser não correlacionado com  $y_1$ , e assim por diante. As novas variáveis definidas por este processo,  $y_1, \dots, y_q$ , são os componentes principais. A principal vantagem da análise de componentes principais é que os primeiros componentes serão responsáveis por uma proporção substancial da variação nas variáveis originais, e podem, consequentemente, serem usados para fornecer um resumo de dimensões inferiores dessas variáveis.

Nesta sessão utilizaremos o conjunto de dados *datafactor* para realizar as aplicações da análise de componentes principais. Para isto, precisamos, primeiramente, organizar os dados de modo que tenhamos uma matriz de dupla entrada, contendo os genótipos nas linhas e as variáveis nas colunas.

- Organizando o arquivo para análise

```
library(dplyr)
library(plyr)

nvars = 15
gru = c("HIB")
namvar = names(datafactors[, (ncol(datafactors)-nvars+1):ncol(datafactors)])
datapca = data.frame(datafactors[(match(c(gru,namvar), names(datafactors)))])

datapca = datapca %>%
  dplyr::group_by(HIB) %>%
  dplyr::summarise_all(funns(m = mean(., na.rm = T)))
datapca = data.frame(datapca)
rownames(datapca) = datapca[,1]
datapca = datapca[-1]
names(datapca) = names(datafactors)[5:19]
```

Neste ponto, temos um conjunto de dados chamado *datapca*. Vamos visualizar a estrutura deste conjunto (primeiras 10 linhas e 10 variáveis)

```
options(digits = 4)
head(datapca[,1:10], n = 10)

##      APLA    AIES     PRE    CESP    DIES    DSAB    CSAB    MSAB    MGRA    NFIL
## H1  2.621 1.504 0.5705 15.10 51.18 30.06 15.71 26.72 183.7 16.60
## H10 2.315 1.262 0.5450 15.12 48.43 28.41 15.90 22.80 163.9 15.60
## H11 2.390 1.266 0.5266 15.22 48.77 28.27 16.01 22.60 167.2 15.63
## H12 2.438 1.282 0.5192 14.28 48.59 28.24 14.81 22.61 157.5 16.33
## H13 2.539 1.353 0.5318 15.03 50.56 29.39 15.83 26.04 179.8 17.43
## H2   2.604 1.378 0.5252 15.34 50.92 29.25 16.03 25.66 187.0 16.67
## H3   2.595 1.411 0.5379 14.53 49.45 28.51 15.70 22.91 169.5 15.80
## H4   2.580 1.425 0.5458 15.73 49.24 28.58 16.52 25.75 184.3 15.50
## H5   2.568 1.371 0.5303 15.58 49.90 29.37 16.59 27.70 183.6 16.13
## H6   2.558 1.415 0.5528 15.76 51.53 30.25 16.57 27.27 188.0 16.30

options(digits = 7)
```

De posse do arquivo, vamos agora realizar a análise de componentes principais utilizando a função `pca()` do pacote `cursoR`. Para maiores detalhes sobre os argumentos disponíveis na função, veja `?pca`.

- plotando os autovalores

```
cursoR::pca(datapca,
  type = "eigen",
  results = F,
  geom = "line",
  repel = T,
  xlab = "Número de componentes principais",
```

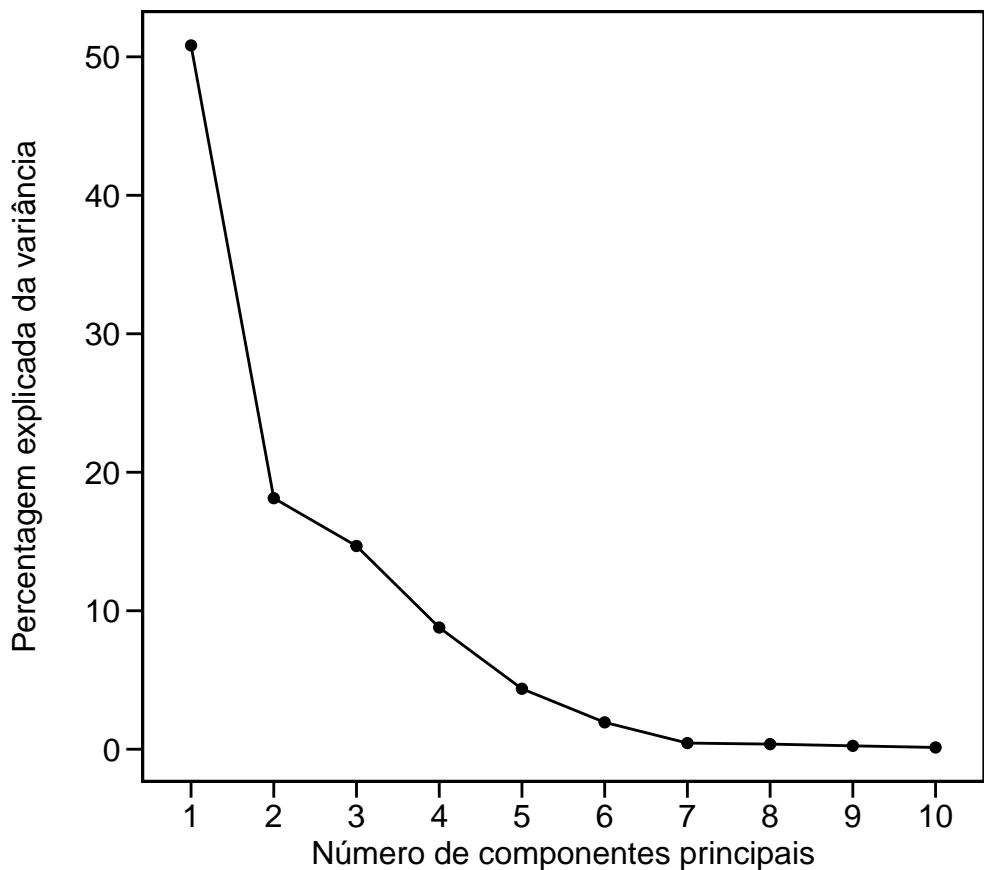


Figure 49: Autovalores da matriz de correlação obtidos pela função `pca()`

```
ylab = "Percentagem explicada da variância",
theme = "journal")
```

Declarando o argumento `results = F|FALSE` apenas o gráfico escolhido no argumento `type` é gerado. Neste exemplo, declaramos que o gráfico a ser mostrado deveria ser o gráfico contendo a variância explicada por cada componente principal.

Uma das principais vantagens da análise de componentes principais é que o padrão de variabilidade pode ser representada graficamente, utilizando gráficos conhecidos como *biplots*. Nos próximos tés comandos, serão plotados os escores dos genótipos, os escores das variáveis e os escores dos genótipos juntamente com os das variáveis. Este último é mais comumente utilizado.

- plotando os escores dos genótipos

```
cursoR::pca(datapca,
  type = "gen",
  results = F,
  geom = c("text", "point"),
```

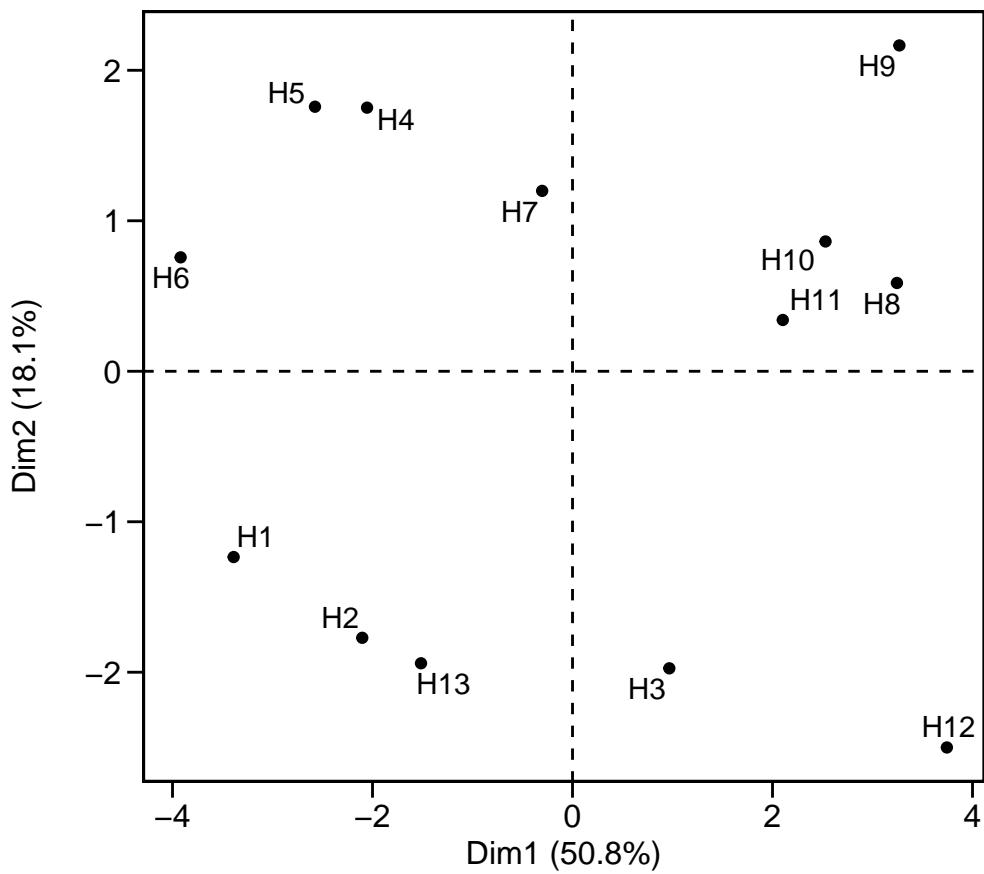


Figure 50: Escores dos genótipos nos dois primeiros PCA obtidos pela função pca()

```
repel = TRUE,
theme = "journal")
```

- plotando os escores das variáveis

```
cursoR::pca(datapca,
  type = "var",
  results = F,
  geom = c("text", "point", "arrow"),
  col.var = "contrib",
  gradient.cols = c("blue", "red"),
  repel = TRUE,
  theme = "journal")
```

- biplot, escores de genótipos e variáveis

```
cursoR::pca(datapca,
  type = "biplot",
```

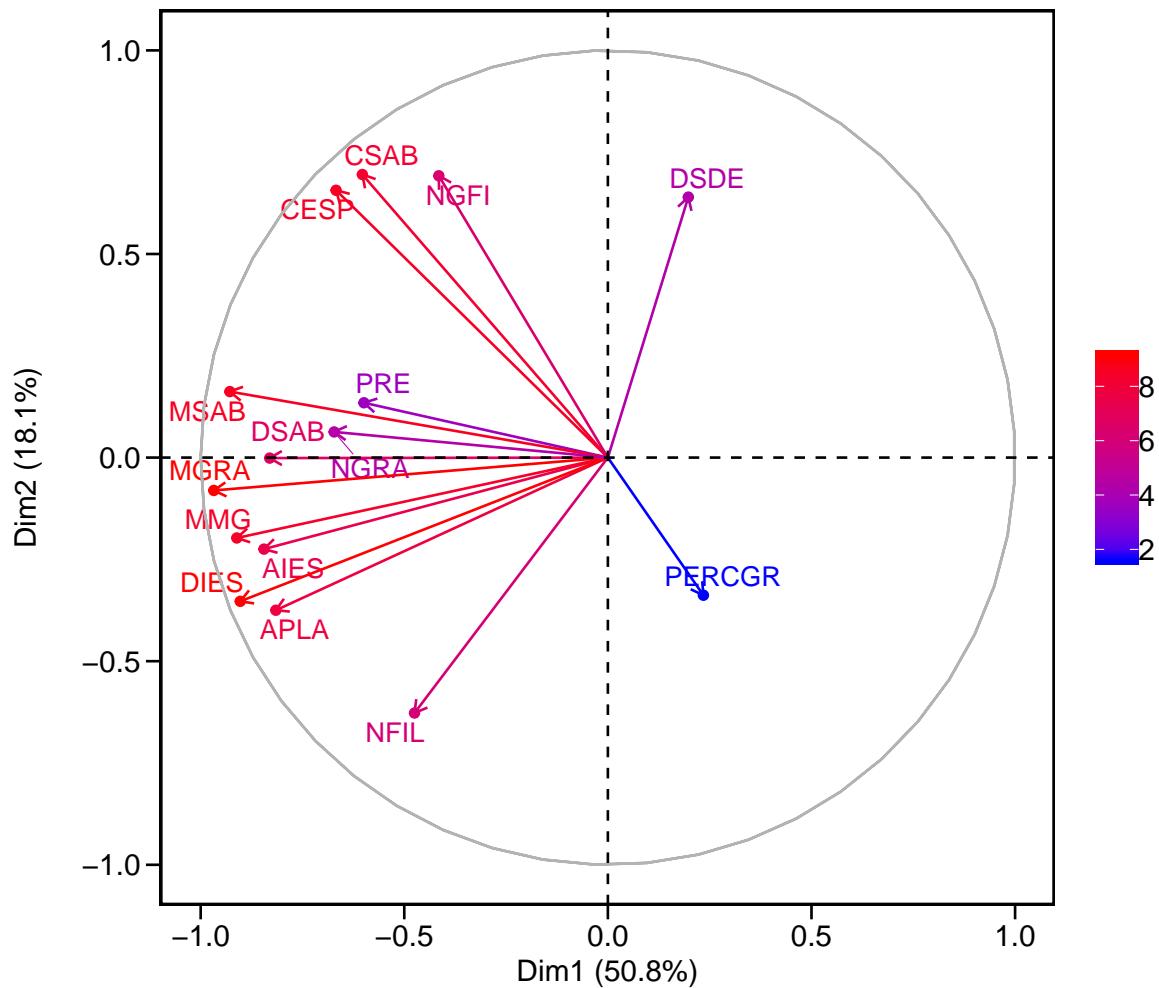


Figure 51: Escores das variáveis nos dois primeiros PCA obtidos pela função `pca()`

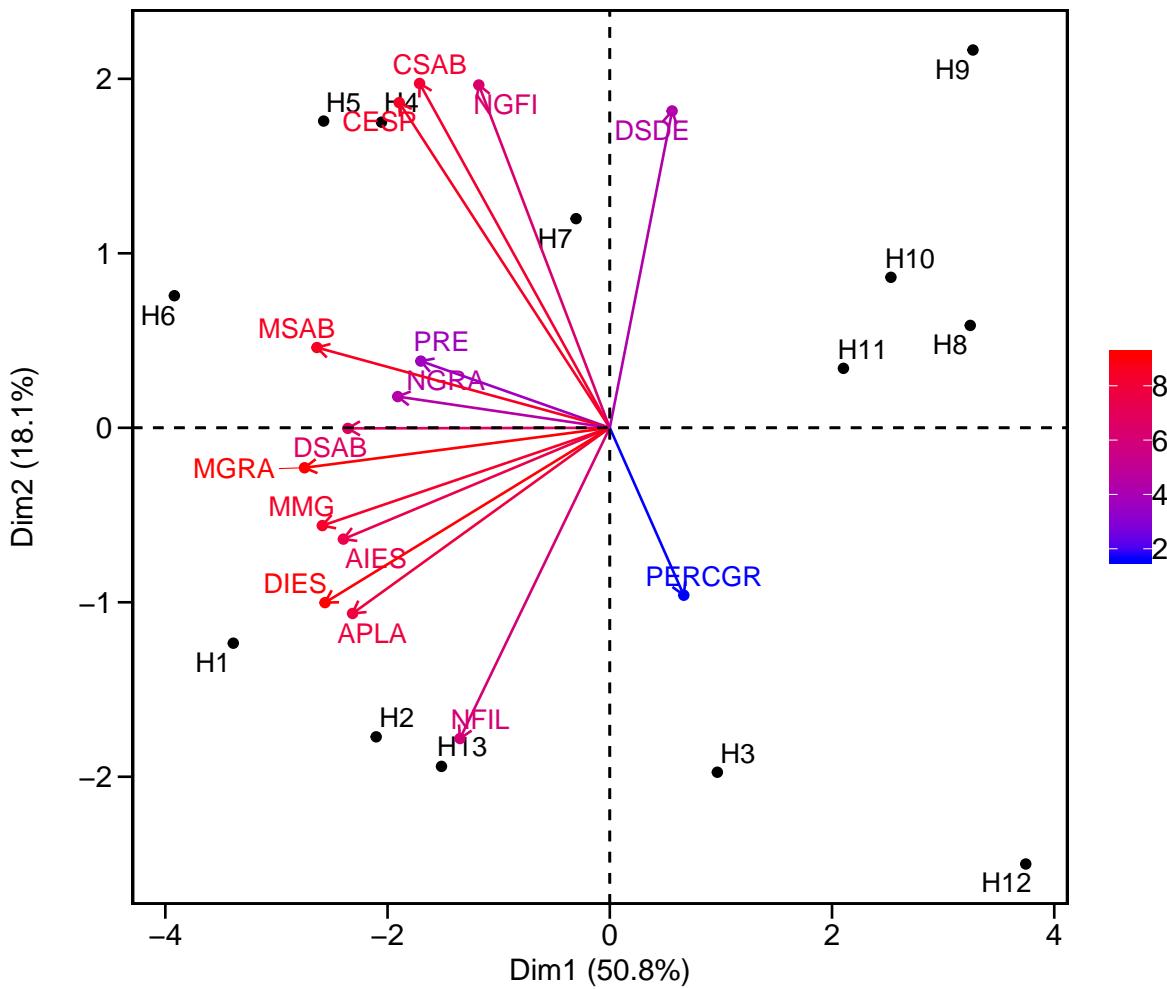


Figure 52: Escores dos genótipos e das variáveis nos dois primeiros PCA obtidos pela função `pca()`

```

results = F,
geom.ind  = c("text", "point"),
geom.var  = c("text", "point", "arrow"),
col.var = "contrib",
gradient.cols = c("blue", "red"),
repel = TRUE,
theme = "journal")

```

Nos biplots acima, os escores plotados foram em relação à média dos genótipos. No entanto, a estrutura original dos dados é um experimento fatorial. Em casos em que é do interesse do pesquisador realizar uma análise para cada ambiente, por exemplo, a estrutura do banco de dados precisa ser modificada e alguns novos argumentos precisam ser inseridos na função. Vamos ao exemplo.

- Organizando o banco de dados para uma análise considerando os ambientes como fatores

```

library(dplyr)
library(plyr)
nvars = 15
gru = c("HIB", "AMB")
namvar = names(datafactors[, (ncol(datafactors)-nvars+1):ncol(datafactors)])
datapca2 = data.frame(datafactors[(match(c(gru,namvar), names(datafactors)))] )
datapca2 = datapca2 %>%
  dplyr::group_by(HIB, AMB) %>%
  dplyr::summarise_all(funs(m = mean(., na.rm = T)))
datapca2 = data.frame(datapca2)
datapca2$x <- paste(datapca2$HIB, datapca2$AMB)
rownames(datapca2) = datapca2$x
datapca2 = datapca2[-(match(c("HIB", "x"), names(datapca2)))]
names(datapca2) = names(datafactors)[c(1, 5:19)]

groups = as.factor(datapca2$AMB)
datapca2 = datapca2[,-1]

```

Neste ponto, criamos um novo conjunto de dados, onde as médias foram calculadas para cada tratamento, ou seja, combinação de genótipos e ambientes. Vamos visualizar os dados (primeiras 10 linhas e 10 variáveis)

```

options(digits = 4)
head(datapca2[,1:10], n = 10)

##      APLA     AIES     PRE    CESP    DIES    DSAB    CSAB    MSAB    MGRA    NFIL
## H1 A1  2.723 1.6833 0.6256 15.43 51.08 28.05 15.71 23.49 202.7 16.27
## H1 A2  2.930 1.7953 0.6122 14.97 51.87 31.52 15.49 30.20 188.2 17.07
## H1 A3  2.197 1.0953 0.4972 14.83 50.59 30.90 15.81 26.84 156.8 15.87
## H1 A4  2.635 1.4440 0.5469 15.15 51.16 29.78 15.81 26.37 187.3 17.20
## H10 A1 2.783 1.6207 0.5837 16.10 53.17 31.37 16.79 24.59 192.4 16.67
## H10 A2 2.049 0.9873 0.4936 15.53 46.73 26.80 16.30 26.34 159.9 14.00
## H10 A3 2.038 1.0113 0.5029 13.99 43.87 24.78 15.18 12.28 120.6 15.33
## H10 A4 2.389 1.4267 0.5996 14.89 49.96 30.70 15.32 28.02 182.7 16.40
## H11 A1 2.749 1.5787 0.5742 16.63 48.92 28.98 17.18 23.61 188.3 15.20
## H11 A2 2.147 1.0187 0.4749 15.12 47.32 27.23 15.65 24.29 163.6 13.73

options(digits = 7)

```

De posse dos novos dados, podemos agora confeccionar um novo biplot . \* Biplot, escores de genótipos e variáveis

```

cursoR::pca(datapca2,
  type = "biplot",
  results = F,
  col.ind = groups, # declara que a cor dos genótipos deve ser por ambiente
  palette = c("blue", "red", "gray", "green"), # vetor de cores (n = nº de ambient
  addEllipses = TRUE, # adiciona elipses nos centroides de cada ambiente

```

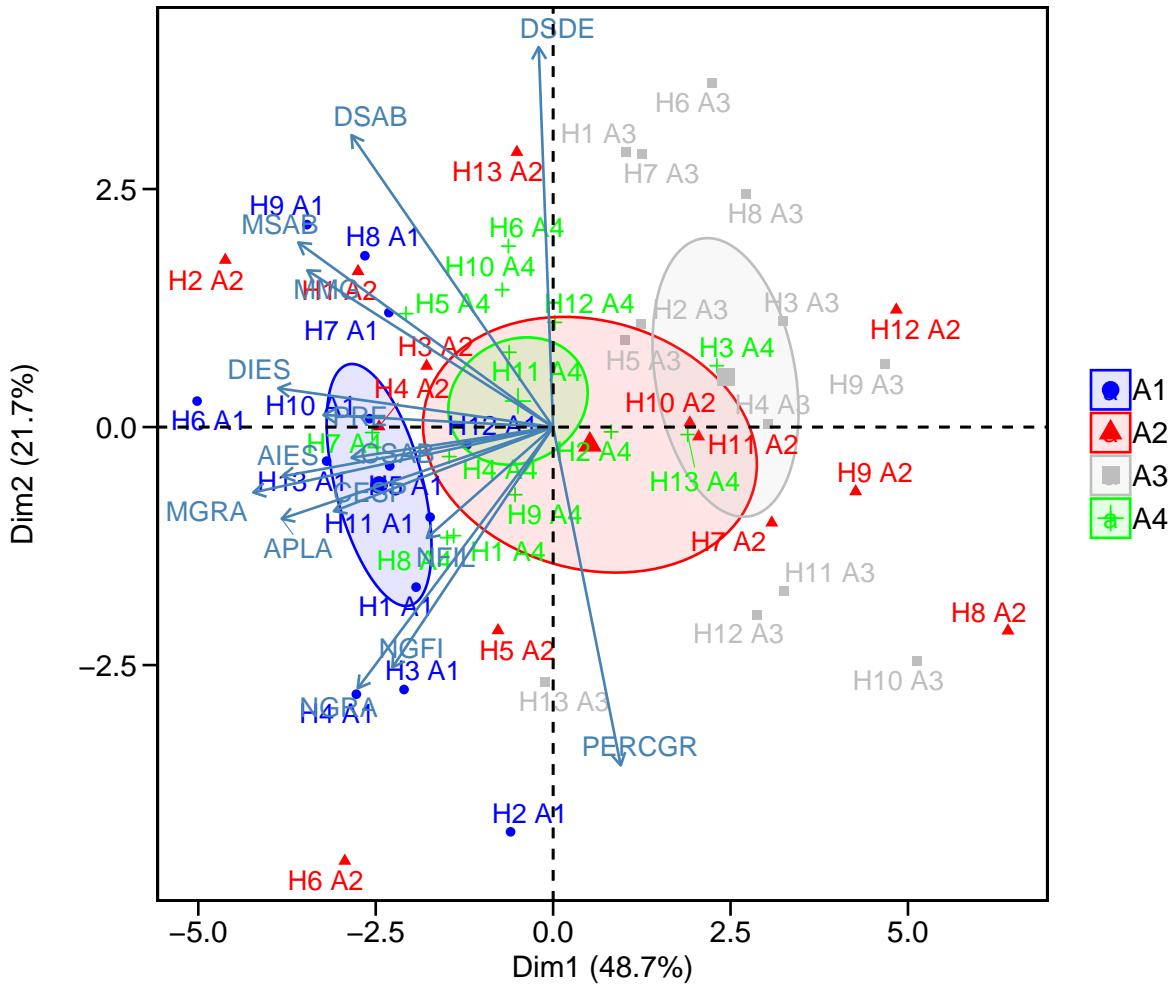


Figure 53: Escores dos genótipos e das variáveis nos dois primeiros PCA obtidos pela função `pca()` considerando os efeitos de ambiente

```
ellipse.type = "confidence", # elipse representa a confiança
legend.title = "Groups", # título da legenda
repel = TRUE, # Argumento para evitar sobreposição das legendas
theme = "journal")
```

#### 4.6.2 Análise de agrupamento hierárquico

A análise de agrupamento é um procedimento multivariado muito útil no melhoramento de plantas. O princípio básico é agrupar indivíduos (genótipos) de acordo com suas semelhanças (variáveis analizadas). Esta sessão é focada na estimativa de matrizes de similaridade e na implementação de algoritmos aglomerativos de agrupamento hierárquicos para confecção de dendrogramas.

As seguintes etapas precisam seguidas para realizar o agrupamento hierárquico aglomerativo usando o a função `dstdend()` do pacote `cursoR`.

1. Preparação dos dados
2. Computar uma medida de (di)similaridade entre cada par de genótipos no conjunto de dados.
3. Usar uma função de ligação para agrupar os genótipos no dendrograma, com base nas informações de distância geradas na etapa 1. Genótipos que estão próximos são agrupados usando a função de ligação.
4. Determinar o ponto de corte para formação dos grupos.

Os dados que serão utilizados são os mesmos daqueles utilizados para realizar a análise de componentes principais. O primeiro procedimento é estimar uma matriz de similaridade entre os genótipos testados. Existem muitos métodos para calcular as informações de (di)similaridade. As opções incluídas na função são: “euclidean” (padrão), “pearson”, “kendall”, “spearman”, “maximum”, “manhattan”, “canberra”, “binary”, “minkowski” ou “gower”. Para maiores informações veja `?dstdend`.

- Computando a matriz de distâncias

```
library(cursoR)
dist = dstdend(datapca,
               results = TRUE,
               distmethod = "euclidean",
               dendrogram = FALSE)

options(digits = 2)
dist$distances

##      H1   H10  H11  H12  H13  H2   H3   H4   H5   H6   H7   H8   H9
## H1    0 49.2  37 55.9 39.8 22 28.6 34.5 42.5 11 25.0 50.6 63.5
## H10   49  0.0  14  8.3 42.9 49 29.2 46.0 48.1 51 26.9  8.8 16.6
## H11   37 13.7   0 20.0 40.1 40 16.2 41.0 45.3 40 13.6 14.4 27.1
## H12   56  8.3  20  0.0 49.1 56 34.0 52.9 54.4 59 32.7  8.6  9.7
## H13   40 42.9  40 49.1  0.0 21 48.0  9.1  6.8  33 40.8 49.9 58.2
## H2    22 48.8  40 56.0 20.8   0 40.9 14.2 22.0 13 34.5 53.6 64.9
## H3    29 29.2  16 34.0 48.0 41  0.0 46.8 53.0 36  7.9 26.5 39.3
## H4    34 46.0  41 52.9  9.1 14 46.8  0.0  8.7  26 39.7 52.5 61.9
## H5    42 48.1  45 54.4  6.8 22 53.0  8.7  0.0 34 45.7 55.2 63.3
## H6    11 51.3  40 58.5 32.5 13 36.1 26.2 34.1  0 30.9 54.4 66.7
## H7    25 26.9  14 32.7 40.8 35  7.9 39.7 45.7 31  0.0 26.3 39.2
## H8    51  8.8  14  8.6 49.9 54 26.5 52.5 55.2 54 26.3  0.0 13.2
## H9    63 16.6  27  9.7 58.2 65 39.3 61.9 63.3 67 39.2 13.2  0.0

options(digits = 7)
```

A matriz acima é a matriz de distância Euclidiana entre os genótipos estimada com as 15 variáveis do nosso exemplo. Observe que geralmente é recomendado padronizar as variáveis no conjunto de dados antes de realizar a análise subsequente. A padronização torna as variáveis comparáveis, quando são medidas em diferentes escalas. Por exemplo, uma variável pode

medir a altura em metros e outra variável pode medir o peso em kg. O argumento `scale = TRUE` satisfaz esta opção.

No exemplo anterior, apenas a matriz foi calculada. Vamos agora utilizar outros argumentos da função para a confecção do dendrograma. Como na estimativa das distâncias, diversos algorítimos aglomerativos estão disponíveis para a confecção do dendrograma. Um resumo é dado abaixo:

- “complete”: a distância entre dois clusters é definida como o valor máximo de todas as distâncias entre entre os elementos no cluster 1 e os elementos no cluster 2. Ele tende a produzir clusters mais compactos.
- “single”: a distância entre dois clusters é definida como o valor mínimo de todas as distâncias entre pares os elementos no cluster 1 e os elementos no cluster 2. Ele tende a produzir clusters longos.
- “average” ou UPGMA: a distância entre dois clusters é definida como a distância média entre os elementos no cluster 1 e os elementos no cluster 2.
- “centroid” ou UPGMC: A distância entre dois grupos é definida como a distância entre o centroide do cluster 1 (um vetor médio de comprimento igual a  $p$  variáveis) e o centroide do cluster 2.
- “ward.D”: minimiza a soma de quadrados dentro do cluster. Em cada etapa, o par de clusters com distância mínima entre clusters é mesclado.
- Confeccionando o dendrograma com o método *average*.

```
distdend(datapca,
          results = FALSE,
          distmethod = "euclidean",
          dendrogram = TRUE,
          clustmethod = "average")
```

No dendrograma exibido acima, cada *folha* corresponde a um genótipo. À medida que subimos na *árvore*, genótipos que são semelhantes uns aos outros são combinados em *ramos*, que vão sendo fundidos a uma altura cada vez maior. A altura da fusão, fornecida no eixo vertical, indica a (di)similaridade/distância entre dois genótipos. Quanto maior a altura da fusão, menos semelhantes são os genótipos.

**Observe que as inferências sobre a proximidade de dois genótipos podem ser realizadas apenas com base na altura em que as ramificações que contêm esses dois genótipos são fundidas. Não podemos usar a proximidade de dois genótipos ao longo do eixo horizontal como critério de similaridade.**

Após a confecção do dendrograma, convém avaliar se as distâncias (ou seja, as alturas) na árvore refletem as distâncias originais com precisão. Uma maneira de medir o quanto bem o dendrograma gerado reflete seus dados é calcular a correlação entre as distâncias cofenéticas e os dados de distância originais. No procedimento anterior o dendrograma não foi mostrado, no entanto, o coeficiente de correlação cofenético foi calculado. Para isto basta incluir o seguinte comando:

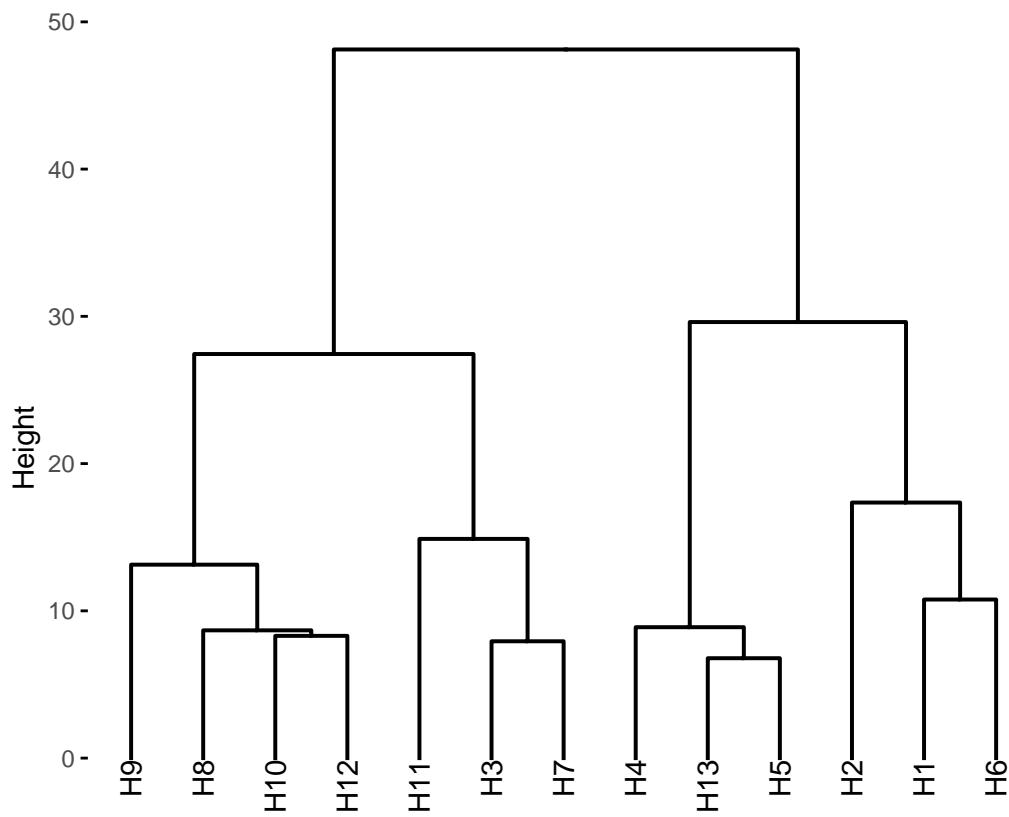


Figure 54: Dendrograma gerado pela função distdend()

```
dist$cophenetic
```

```
## [1] 0.865619
```

Quanto mais próximo o valor do coeficiente de correlação for de 1, mais precisamente o dendrograma refletirá as distâncias originais. Valores acima de 0,75 são considerados bons. O método de ligação “average”, ou UPGMA parece produzir altos valores dessa estatística. Esta pode ser uma das razões por que ele é tão popular.

A função `distdend()` também conta com um algorítimo de seleção de variáveis que considera os pesos dos autovetores associados aos autovalores de menor magnitude para excluir uma determinada variável. Para maiores informações, veja `?distdend`. Quando declaramos o argumento `selvar = TRUE` na função, um algorítimo de seleção de variáveis é executado. O objetivo é selecionar um grupo de variáveis que mais contribuem para explicar a variabilidade dos dados originais. Digamos que, algumas poucas variáveis pudessem ser utilizadas para agrupar os nossos tratamentos sem que haja perda de informação, recursos humanos e financeiros poderiam ser poupadados, pois ao envés de avaliarmos 15 variáveis (em nosso exemplo), poderíamos avaliar somente aquelas que realmente contribuem para a distinção dos tratamentos. Vamos ao exemplo.

```
library(cursoR)
dist = distdend(datapca,
                 results = TRUE,
                 selvar = TRUE,
                 distmethod = "euclidean",
                 dendrogram = TRUE)

## Calculating model 1 with 15 variables. ' AIES ' excluded in this step ( 7.1 %).
## Calculating model 2 with 14 variables. ' PRE ' excluded in this step ( 14.3 %).
## Calculating model 3 with 13 variables. ' DSDE ' excluded in this step ( 21.4 %).
## Calculating model 4 with 12 variables. ' APLA ' excluded in this step ( 28.6 %).
## Calculating model 5 with 11 variables. ' DSAB ' excluded in this step ( 35.7 %).
## Calculating model 6 with 10 variables. ' NFIL ' excluded in this step ( 42.9 %).
## Calculating model 7 with 9 variables. ' PERCGR ' excluded in this step ( 50 %).
## Calculating model 8 with 8 variables. ' CESP ' excluded in this step ( 57.1 %).
## Calculating model 9 with 7 variables. ' CSAB ' excluded in this step ( 64.3 %).
## Calculating model 10 with 6 variables. ' DIES ' excluded in this step ( 71.4 %).
## Calculating model 11 with 5 variables. ' MGRA ' excluded in this step ( 78.6 %).
## Calculating model 12 with 4 variables. ' MSAB ' excluded in this step ( 85.7 %).
## Calculating model 13 with 3 variables. ' NGFI ' excluded in this step ( 92.9 %).
## Calculating model 14 with 2 variables. ' MMG ' excluded in this step ( 100 %).
## Done!
##
##
## -----
## Summary of the adjusted models
## -----
##      Model excluded cophenetic remaining cormantel      pvmantel
```

```

## 1 Model 1      - 0.8656190      15 1.0000000 0.000999001
## 2 Model 2      AIES 0.8656191    14 1.0000000 0.000999001
## 3 Model 3      PRE  0.8656191    13 1.0000000 0.000999001
## 4 Model 4      DSDE 0.8656191   12 1.0000000 0.000999001
## 5 Model 5      APLA 0.8656189   11 1.0000000 0.000999001
## 6 Model 6      DSAB 0.8655939   10 0.9999996 0.000999001
## 7 Model 7      NFIL 0.8656719    9 0.9999982 0.000999001
## 8 Model 8      PERCGR 0.8657259  8 0.9999977 0.000999001
## 9 Model 9      CESP 0.8657904   7 0.9999972 0.000999001
## 10 Model 10     CSAB 0.8658997  6 0.9999964 0.000999001
## 11 Model 11     DIES 0.8658274  5 0.9999931 0.000999001
## 12 Model 12     MGRA 0.8643556  4 0.9929266 0.000999001
## 13 Model 13     MSAB 0.8640355  3 0.9927593 0.000999001
## 14 Model 14     NGFI 0.8648384  2 0.9925396 0.000999001
##
## -----
## 
## Suggested variables to be used in the analysis
## -----
## The distance were calculated based on the variables in Model 10 .
## The variables included in this model were...
## DIES MSAB MGRA NGFI MMG NGRA
## -----

```

A saída acima nos mostra o progresso do algoritmo, indicando qual foi a variável excluída em cada passo. Isto pode ser omitido, indicando `verbose = FALSE` na função. O resumo do modelo (*Summary of the adjusted models*) nos fornece duas informações muito úteis. Primeira: ao reduzir o número de variáveis utilizadas na estimativa das distâncias, o coeficiente de correlação cofenético não reduziu significativamente, apresentando variação apenas na terceira casa decimal. Vamos ver o gráfico gerado.

```
dist[["cofgrap"]]
```

A segunda, e talvez mais importante, é a correlação de mantel realizada com a matriz de distâncias em cada passo da análise com a matriz de distâncias inicial, que foi computada com todas as variáveis. Percebe-se que utilizando apenas duas variáveis, as distâncias calculadas foram praticamente idênticas ( $r = 0.992$ ) às distâncias calculadas com todas as variáveis. Por padrão, o algoritmo estima a matriz de distâncias e gera o dendrograma (se `dendrogram = TRUE`) considerando as variáveis do modelo com maior coeficiente de correlação cofenética. Em nosso exemplo, as variáveis utilizadas foram DIES, MSAB, MGRA, NGFI, MMG e NGRA. Veja que reduzimos em um terço o número de variáveis necessárias para diferenciar os tratamentos, sem perda de informação. Salienta-se, no entanto, que este resultado foi obtido neste exemplo. Um conjunto de dados diferente poderá apresentar resultados distintos. Cabe ao pesquisador, assim, decidir qual é o melhor modelo a ser utilizado.

Para fins didáticos, iremos presseguir com nossos exemplos, estimando as distâncias somente com as variáveis selecionadas no processo anterior. Para isto, precisaremos criar um novo conjunto de dados, extraiendo apenas as variáveis de nosso interesse.

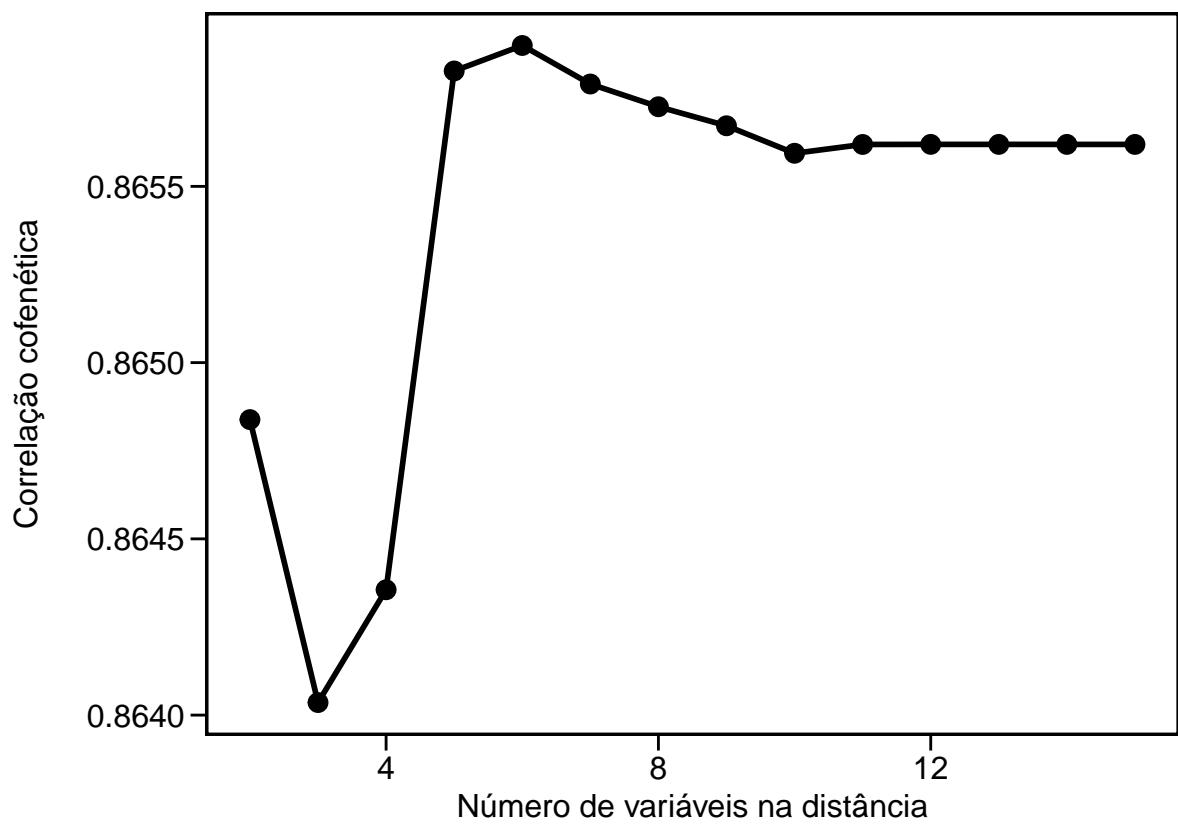


Figure 55: Coeficiente de correlação cofenética em matrizes de distância estimadas com diferentes números de variáveis

```

datadist = datapca[,c("DIES", "MSAB", "MGRA", "NGFI", "MMG", "NGRA")]
head(datadist, n = 10)
##          DIES     MSAB     MGRA     NGFI      MMG     NGRA
## H1  51.17650 26.72237 183.7469 32.20000 364.6268 506.5500
## H10 48.43300 22.80458 163.8930 32.40000 319.9602 503.6333
## H11 48.77400 22.60115 167.2397 33.00000 332.9150 500.6500
## H12 48.58717 22.60875 157.4735 30.38333 315.7981 501.7500
## H13 50.56450 26.03742 179.8382 30.98333 340.2383 537.6500
## H2  50.91883 25.66495 186.9990 31.88333 356.1710 526.5333
## H3  49.44883 22.91384 169.4556 31.38333 345.7283 491.2333
## H4  49.24417 25.74671 184.2604 35.00000 346.1070 535.3333
## H5  49.89933 27.69686 183.6348 33.93333 340.9189 541.7000
## H6  51.52800 27.26551 187.9994 32.83333 363.1051 516.2167

```

Um dos problemas com o agrupamento hierárquico é que ele não nos informa quantos clusters existem ou onde cortar o dendrograma para formar os clusters. Você pode cortar a árvore hierárquica a uma determinada altura, digamos, na média das distâncias, no entanto esta decisão é puramente impírica. Por exemplo, se o ponto de corte for muito alto, tendemos a agrupar genótipos que podem, de fato, não ser semelhantes. Um ponto de corte muito baixo, por outro lado, pode resultar em fracassos na seleção, pois consideramos que os genótipos são distintos, podendo não o serem. Procedimentos estatísticos à exemplo de Milligan and Cooper (1985), Scott and Symons (1971), Krzanowski and Lai (1988), Halkidi, Batistakis, and Vazirgiannis (2001), e Hubert and Arabie (1985) são recomendados para esta escolha. Estes algorítimos ainda não estão implementados na função, no entanto o pacote `NbClust` (Charrad et al. 2014) pode ser utilizado para este fim. Por padrão, a função fornece o ponto de corte calculado pelo método de Mojena (Mojena 1977).

Além disto, a função `distdend()` também conta com um procedimento baseado em reamostragens bootstrap que calcula a probabilidade de erro de cada galho do dendrograma . Declarando `pvclust = TRUE` e `nboot = 500` um procedimento bootstrap com 500 reamostragens é realizado para a estimativa de p-valores. Para maiores detalhes sobre o método veja Suzuki and Shimodaira (2006). Declarando `alpha = 0.95` indicamos que somente grupos com este nível de confiança serão considerados significativos. Neste exemplo, vamos declarar `results = FALSE` e `dendrogram = FALSE`. Assim, somente o gráfico com os grupos será apresentado.

```

cut = distdend(datadist,
                results = TRUE,
                dendrogram = TRUE,
                pvclust = TRUE,
                nboot = 500,
                alpha = 0.95,
                distmethod = "euclidean",
                clustmethod = "average")

```

```

## Bootstrap (r = 0.5)... Done.
## Bootstrap (r = 0.5)... Done.
## Bootstrap (r = 0.67)... Done.
## Bootstrap (r = 0.67)... Done.

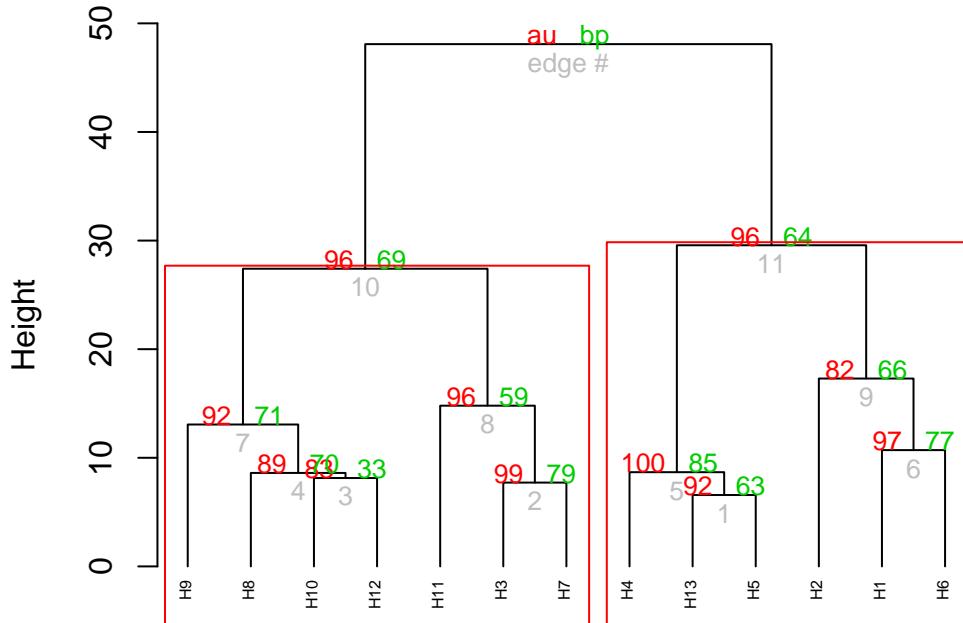
```

```

## Bootstrap (r = 0.83)... Done.
## Bootstrap (r = 1.0)... Done.
## Bootstrap (r = 1.0)... Done.
## Bootstrap (r = 1.17)... Done.
## Bootstrap (r = 1.17)... Done.
## Bootstrap (r = 1.33)... Done.

```

### Cluster dendrogram with AU/BP values (%)



Distance: euclidean  
 Cluster method: average

```
cut$cut
```

```
## [1] 32.28318
```

O resultado do procedimento bootstrap indicou a formação de dois grupos, com probabilidade de erro de 5%. Assim, nas próximas funções, serão demonstrados os diferentes dendrogramas que podem ser gerados

```

distdend(datadist,
  results = FALSE,
  distmethod = "euclidean",
  clustmethod = "average",
  nclust = 2,
  cex = 1,
  lwd = 1.3,

```

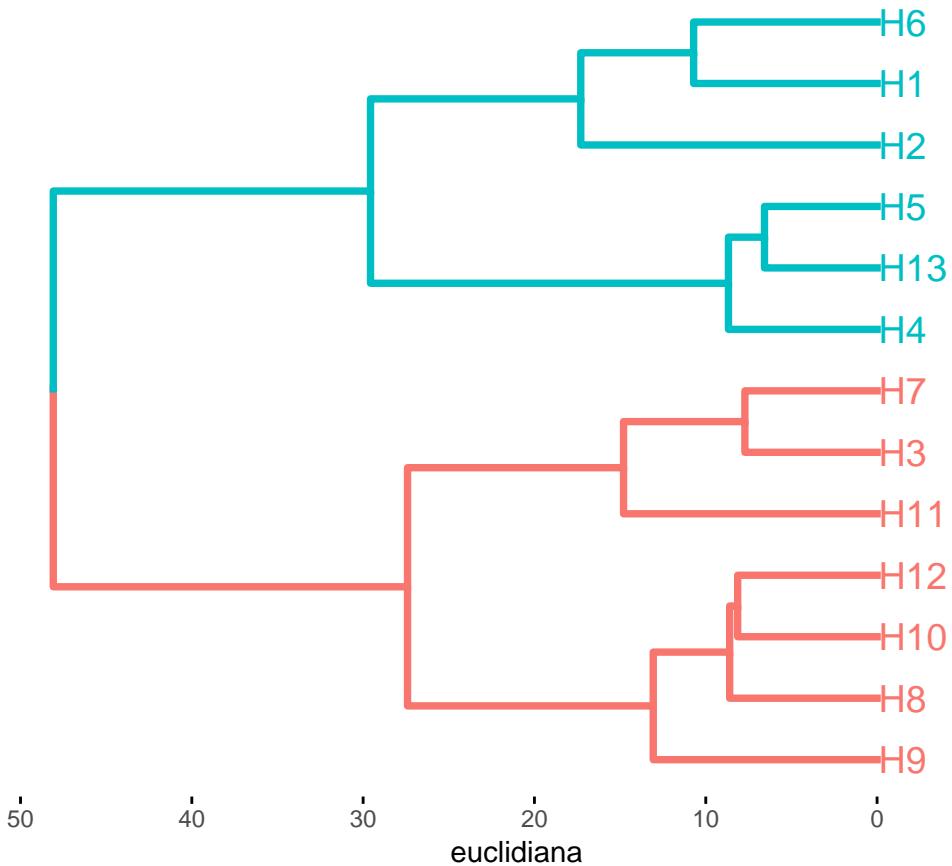


Figure 56: Procedimento bootstrap indicando o número de clusters a serem formados no dendrograma

```

ylab = "euclidiana",
type = "rectangle",
horiz = T)

distdend(datadist,
  results = FALSE,
  distmethod = "euclidean",
  clustmethod = "average",
  nclust = 2,
  rect = TRUE,
  ylab = "euclidiana",
  color_labels_by_k = F,
  k_colors = c("black", "black", "black"),
  horiz = F)

distdend(datadist,
  results = FALSE,

```

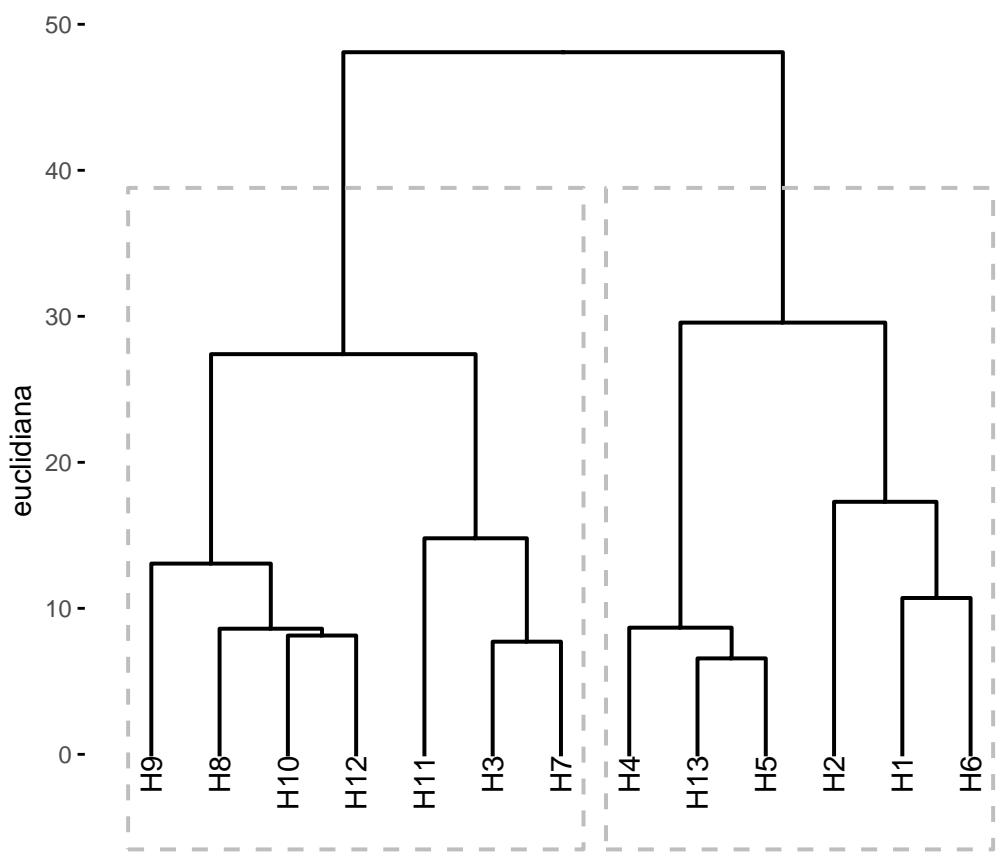


Figure 57: Dendrograma personalizado com cores indicando os grupos

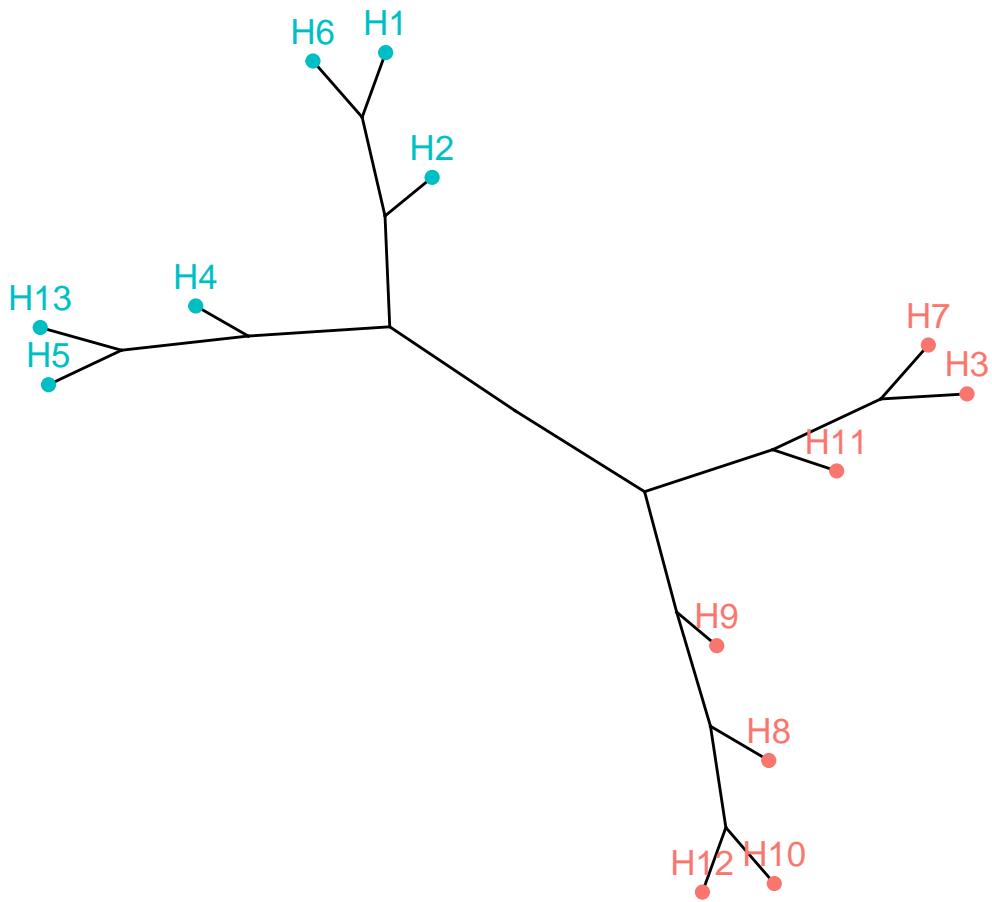


Figure 58: Árvore filogenética representando a matriz de distâncias

```

distmethod = "euclidean",
clustmethod = "average",
nclust = 2,
cex = 1.2,
type = "phylogenetic")

distdend(datadist,
  results = FALSE,
  distmethod = "euclidean",
  clustmethod = "average",
  nclust = 2,
  type = "circular")

```

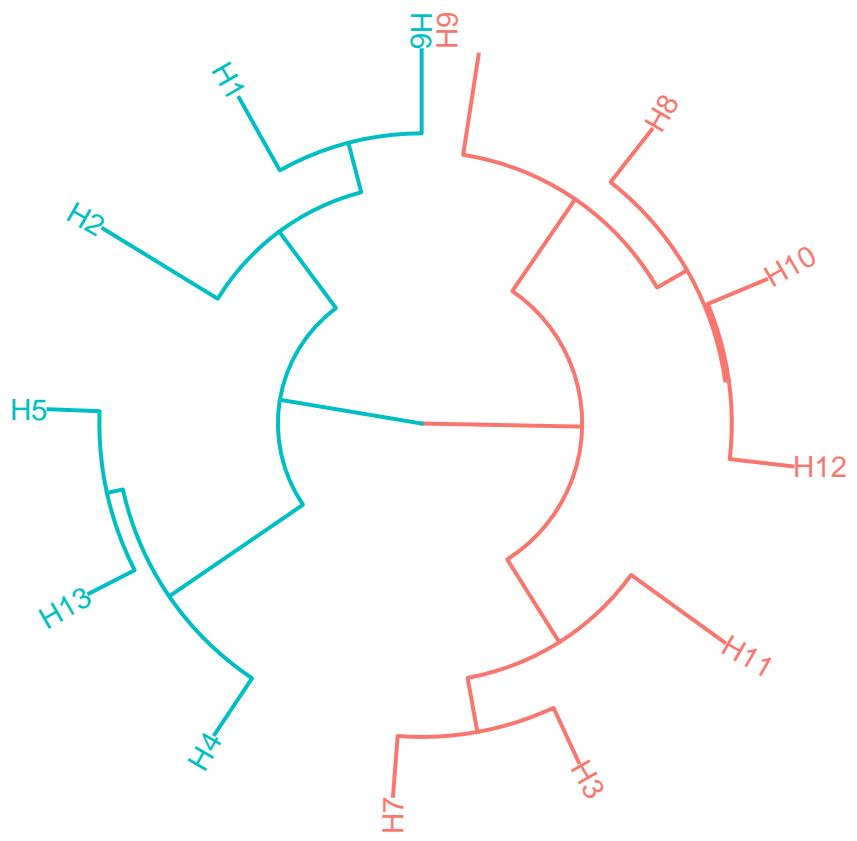


Figure 59: Dendrograma circular representando a matriz de distâncias

#### 4.6.3 k-means

A idéia básica por trás do agrupamento k-means consiste em definir clusters para que a variação total dentro do cluster seja minimizada. Existem vários algoritmos k-means disponíveis. O algoritmo padrão e o mais utilizado é o algoritmo de Hartigan-Wong (Hartigan and Wong 1977), que define a variação total dentro do cluster como a soma das distâncias Euclidianas quadráticas entre os itens e o centróide correspondente. Os passos básicos para a análise são os que seguem:

1. Especifique o número de clusters (K) a serem criados;
  2. Selecione aleatoriamente k objetos do conjunto de dados como os centros de clusters iniciais ou médias;
  3. Atribui cada observação ao seu centróide mais próximo, com base na distância euclidiana entre o objeto e o centróide;
  4. Para cada um dos k clusters, atualize o centróide do cluster calculando os novos valores médios de todos os pontos de dados no cluster. O centróide do k-ésimo cluster é um vetor de comprimento p ( $p = \text{número de variáveis}$ ) contendo as médias de todas as variáveis para as observações no k-ésimo cluster;
  5. Iterativamente, minimize a soma de quadrados total dentro do cluster; isto é, iterar as etapas 3 e 4 até que as atribuições do cluster parem de mudar ou até que o número máximo de iterações sejam atingidas. Por padrão, o software R usa 10 como o valor padrão para o número máximo de iterações.
- Exemplo

```
library(ggplot2)
kmeans = k_means(datadist,
                  nclust = 3)
```

## 4.7 Interação genótipo vs ambiente

Esta sessão tem como foco principal a análise estatística de dados oriundos de ensaios de melhoramento genético. Assim, os maiores esforços serão direcionados para duas importantes áreas da estatística relacionadas com estes tipos de experimentos: (i) análise multivariada ; e (ii) interação genótipo *vs* ambiente (GEI) . Como algumas funções específicas deste tipo de análise não são encontradas em pacotes, ou muitas vezes, a realização de uma determinada análise depende de funções de diferentes pacotes, utilizaremos nesta sessão, principalmente, o pacote WAASB (T. Olivoto 2018b) e cursoR, apresentado anteriormente na seção Experimentos-bifatoriais. O pacote WAASB foi desenvolvido em linguagem R e distribuídos sob a licença GPL 3.0. Nossa principal motivação para o desenvolvimento deste pacote foi fornecer funções amigáveis e confiáveis para serem utilizadas na análise de dados de experimentos voltados ao melhoramento genético de plantas. O pacote contém funções para análise GEI utilizando modelos mistos (Piepho 1994) e análise AMMI (Gauch 2013). As principais funções do pacote são apresentadas abaixo.

Função	Descrição
WAASB()	Análise da interação GEI via modelos mistos com biplots para a interação
WAASBYratio()	Permite atribuir pesos para produtividade e estabilidade em modelos mistos
WAAS.AMMI()	Análise da interação GEI via análise AMMI tradicional
WAASratio.AMMI()	Permite atribuir pesos para produtividade e estabilidade em análise AMMI
validation.blup()	Procedimento de validação cruzada em modelos mistos
validation.AMMIF()	Procedimento de validação cruzada para membros de modelos da família AMMI
predict.AMMI()	Predição da variável resposta baseada na análise AMMI
predict.AMMImean()	Predição da variável resposta baseada na análise AMMI com dados médios
plot.blup()	Gráfico de médias preditas via modelos mistos
plot.validation.AMMIF()	Gráfico para representar o erro de predição dos modelos
plot.WAASBY()	Gráfico para seleção de genótipos considerando pesos
plot.WAASBYratio()	Mapas de calor com diferentes interpretações para seleção de genótipos
summary.WAASB()	Resume um objeto WAASB
plot.eigen()	Gráfico com os autovalores da matriz de interação BLUP

Uma cultura pode ser vista como um sistema complexo com resultados (por exemplo, rendimento de grãos) que são afetados por informações genéticas, fisiológicas, pedoclimáticas e de manejo. Melhoristas e geneticistas se esforçam continuamente para aumentar a produtividade das culturas visando suprir a demanda mundial cada vez maior por alimentos. É na fase final de um programa de melhoramento de plantas que muito esforço e recursos precisam ser investidos na avaliação dos genótipos ( $G$ ) a serem selecionados. Geralmente algumas centenas de genótipos precisam ser avaliados em um grande número de ambientes ( $E$ ). Estes ensaios são conhecidos como ensaios multi-ambientes (EMA) e os dados destes experimentos resultam em uma matriz  $M$  de dimensões  $G \times E$ . É nesta fase do processo que surge um dos maiores desafios da análise de EMA: compreender a interação genótipo  $\times$  ambiente (GEI) buscando novas formas de explorá-la e utilizá-la a favor da seleção de genótipos com estabilidade produtiva satisfatória.

## 4.8 Download dos dados

Os dados utilizados neste módulo são oriundos de um ensaio de aveia com 10 genótipos conduzidos em cinco ambientes. Em cada ambiente um delineamento de blocos completos casualizados com três blocos e uma observação por bloco foi utilizado. Sete variáveis foram avaliadas. Ao longo desta sessão, utilizaremos o rendimento de grãos (**RG**) como exemplo.

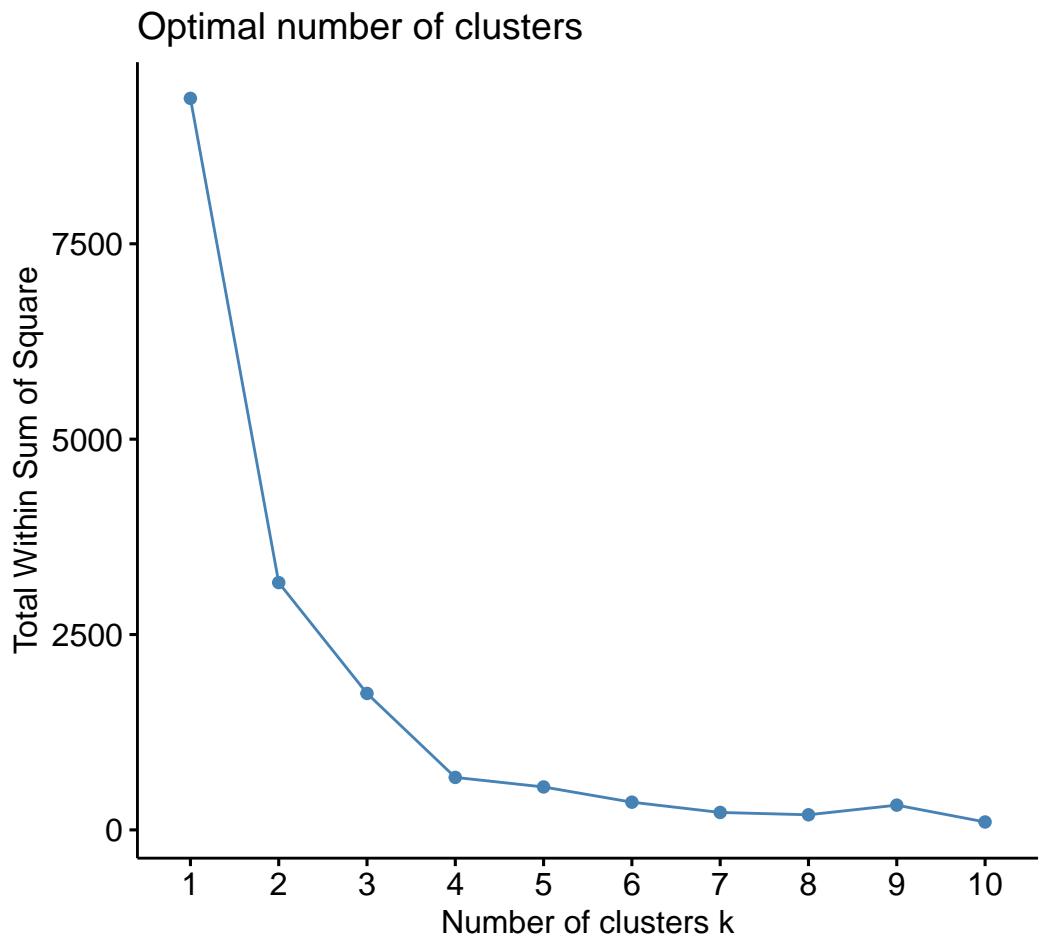


Figure 60: Gráfico para agrupamento k-means gerado pela função kmeans()

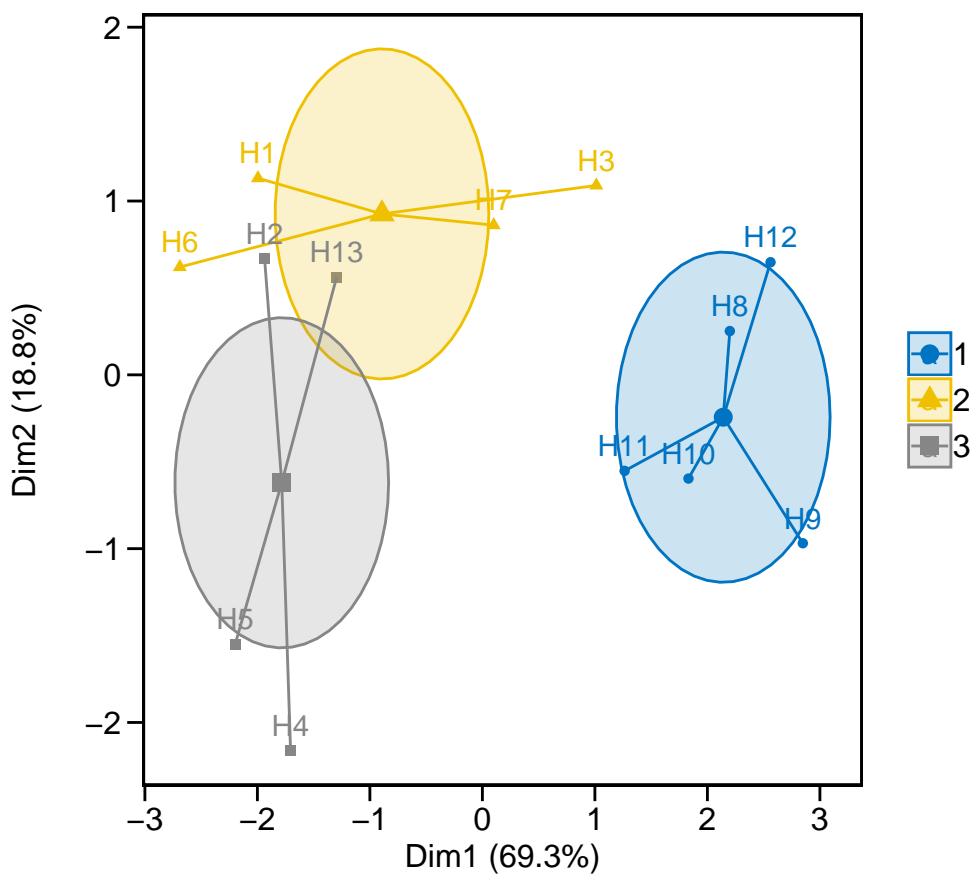


Figure 61: Gráfico para agrupamento k-means gerado pela função kmeans()

Para download dos dados, basta rodar o seguinte código no console R. Online, os dados estão disponíveis em Olivoto T. and Sari (2018a).

```
# Importando os dados
interaction = read.csv("https://data.mendeley.com/datasets/f83v482wkx/1/files/d6abb3ff-5

# observando a estrutura do banco de dados
head(interaction)

##   AMB GEN BLOCO      RG     PH    MMG DEF DFM DEM APLA
## 1  E1  G1       1 2167.00 44.93 31.28  91  41 132   90
## 2  E1  G1       2 2503.04 46.90 32.92  92  42 134   93
## 3  E1  G1       3 2427.32 47.75 32.28  92  39 131  103
## 4  E1  G2       1 3207.50 45.20 29.00  92  38 130   87
## 5  E1  G2       2 2932.90 45.30 30.48  92  40 132   85
## 6  E1  G2       3 2564.84 45.49 29.60  91  40 131   75
```

## 4.9 Compreendendo a interação genótipo vs ambiente

Nesta sessão, utilizaremos a função `ge_stats()` do pacote `cursoR`. Esta função computa uma análise individual para cada ambiente, uma ANOVA conjunta, bem como algumas estatísticas da GEI. Para maiores detalhes, utilize `?ge_stats` no console R. Os argumentos que precisaremos informar na função são: `data` (no nosso caso o arquivo `interaction`), `resp` (a variável resposta, *RG*) `gen` (o nome da coluna correspondente aos genótipos *GEN*) `env` (o nome da coluna relacionada aos ambientes *AMB*) e `rep` (o nome da coluna correspondente às repetições *BLOCO*). Vamos ao exemplo.

```
library(cursoR)
stats = ge_stats(interaction, resp = "RG", gen = "GEN", env = "AMB", rep = "BLOCO")
```

Com esta função, salvamos no objeto `stats` todas as saídas geradas pela função. Agora vamos, passo-a-passo, visualizar estas saídas.

### 4.9.1 Análise individual

A análise de variância individual é calculada para cada ambiente considerando o modelo  $y_{ij} = \mu + \alpha_i + \beta_j + \varepsilon_{ij}$ , onde  $y_{ij}$  é a variável resposta observada no  $i$ -ésimo genótipo alocado no  $j$ -ésimo bloco,  $\mu$  é a média geral do experimento,  $\alpha_i$  é o efeito do  $i$ -ésimo genótipo,  $\beta_j$  é o efeito do  $j$ -ésimo bloco, e  $\varepsilon_{ij}$  é o erro aleatório associado à  $y_{ij}$ , onde  $\varepsilon_{ij} \stackrel{iid}{\sim} N(0, \sigma^2)$ .

```
options(digits = 2)
stats$individual

## $individual
##   ENV Mean MSblock MSgen MSres Fcal(Blo) Pr>F(Blo) Fcal(Gen) Pr>F(Gen)
## 1  E1 2521   65222 337386 144008      0.45     0.643      2.3  5.9e-02
## 2  E2 3180   698289 206625 178515      3.91     0.039      1.2  3.8e-01
## 3  E3 3675   116460 531337 137694      0.85     0.446      3.9  7.1e-03
```

```

## 4 E4 2507 67066 407701 231050 0.29 0.751 1.8 1.5e-01
## 5 E5 3107 165496 550423 66759 2.48 0.112 8.2 8.5e-05
## 6 E6 3910 218807 526119 66357 3.30 0.060 7.9 1.1e-04
## 7 E7 2663 159851 134643 58566 2.73 0.092 2.3 6.3e-02
## CV(%) h2 AS R2
## 1 15.1 0.57 0.76 0.70
## 2 13.3 0.14 0.37 0.54
## 3 10.1 0.74 0.86 0.79
## 4 19.2 0.43 0.66 0.64
## 5 8.3 0.88 0.94 0.89
## 6 6.6 0.87 0.93 0.89
## 7 9.1 0.57 0.75 0.70
##
## $MSEratio
## [1] 3.9

```

#### 4.9.2 Médias e efeitos da interação

```

options(digits = 3)
stats$ge_means$ge_means

##      E1   E2   E3   E4   E5   E6   E7
## G1  2366 3041 3494 2297 3231 4171 2809
## G10 1974 3146 4271 2786 2716 3366 2485
## G2  2902 3231 3716 2558 2268 3827 2541
## G3  2889 3608 4134 2771 2910 4127 2985
## G4  2589 3190 3302 2344 3821 3783 2701
## G5  2188 3140 3379 2351 3175 3469 2433
## G6  2301 3291 3403 2337 2976 3569 2336
## G7  2774 2613 3019 1759 3229 4048 2666
## G8  2899 3445 4138 3043 3535 4812 2908
## G9  2326 3095 3896 2827 3210 3934 2768

```

Esta saída contém as médias dos genótipos nos ambientes avaliados. Estas são as médias dos  $B$ -Blocos de cada ambiente.

```

options(digits = 3)
stats$ge_means$ge_effects

##      E1      E2      E3      E4      E5      E6      E7
## G1 -132.6 -116.9 -158.61 -188.11 145.7 282.36 168.2
## G10 -429.4  83.1  712.98  395.82 -273.9 -427.58 -61.1
## G2  455.4 125.4 114.63 125.57 -764.7 -8.73 -47.5
## G3  102.3 162.7 193.09 -2.07 -462.3 -49.48 55.9
## G4   44.2 -13.6 -396.79 -187.11 690.5 -151.57 14.3
## G5 -128.5 164.1 -92.25  48.14 272.1 -237.59 -26.0
## G6  -26.9 304.2 -78.78  22.59  62.0 -148.55 -134.6

```

```

## G7    461.2 -359.1 -448.15 -540.27  329.9  345.58  210.9
## G8   -80.7 -194.6     3.08   76.15 -31.6  442.18 -214.5
## G9  -265.0 -155.4  150.79  249.29   32.4  -46.60   34.6

```

Esta é a matriz dos efeitos da interação  $\hat{z}_{ij}$ , estimada por:  $\hat{z}_{ij} = y_{ij} - \bar{y}_i - \bar{y}_j + \bar{y}_{..}$ . Como será discutido posteriormente, esta é a matriz que é utilizada na decomposição por valores singulares na análise AMMI.

```

options(digits = 3)
stats$ge_means$gge_effects

```

```

##          E1      E2      E3      E4      E5      E6      E7
## G1   -154.9 -139.3 -180.9 -210.4  123.3  260.0  145.87
## G10  -546.6 -34.1  595.7  278.6 -391.1 -544.8 -178.34
## G2    381.1  51.1   40.3   51.2 -839.1 -83.1 -121.88
## G3    367.9  428.3  458.7  263.6 -196.7  216.2  321.50
## G4     67.9   10.1 -373.1 -163.4  714.2 -127.9   37.97
## G5   -332.4 -39.8 -296.2 -155.8   68.2 -441.5 -229.89
## G6   -219.9  111.2 -271.8 -170.4 -131.0 -341.6 -327.66
## G7   253.0 -567.3 -656.3 -748.4  121.7  137.4   2.69
## G8   378.7  264.8  462.5  535.6  427.8  901.6  244.91
## G9  -194.8 -85.1  221.1  319.6  102.6   23.7  104.82

```

Esta é a matriz dos efeitos de genótipos somado ao efeito da interação  $\hat{z}_{g+ij}$ , estimada por:  $\hat{z}_{g+ij} = y_{ij} - \mu + (y_{ij} - \bar{y}_i - \bar{y}_j + \bar{y}_{..})$ . Esta matriz é utilizada na decomposição por valores singulares na análise GGE (Yan et al. 2007), não apresentada neste material.

#### 4.9.3 ANOVA conjunta

o modelo com efeito de interação utilizado na análise conjunta dos dados é  $y_{ijk} = \mu + \alpha_i + \tau_j + (\alpha\tau)_{ij} + \gamma(\tau)_{jk} + \varepsilon_{ijk}$ , onde  $y_{ijk}$  é a variável resposta observada do genótipo  $i$  ( $i = 1, 2, \dots, G$ ) no bloco  $k$  ( $k = 1, 2, \dots, B$ ) de o ambiente ( $j = 1, 2, \dots, E$ );  $\mu$  é a média geral do experimento;  $\alpha_i$  é o principal efeito do genótipo  $i$ ;  $\tau_j$  é o principal efeito do ambiente  $j$ ;  $(\alpha\tau)_{ik}$  é o efeito de interação do genótipo  $i$  com o ambiente  $j$ ;  $\gamma(\tau)_{jk}$  é o efeito do do bloco  $k$  no ambiente  $j$ ;  $\varepsilon_{ijk}$  é o erro aleatório médio  $\varepsilon_{ij} \stackrel{iid}{\sim} N(0, \sigma^2)$ . Considerando todos os efeitos como fixos, exceto  $\varepsilon_{ijk}$ , a média do genótipo  $i$  no ambiente  $j$  ( $y_{ij}$ ) é a média aritmética B-bloco do genótipo  $i$  no ambiente  $j$ .  $y_{ij} = \sum_k y_{ijk}/B$ , ( $k = 1, 2, \dots, B$ ).

```

stats$anovaconj

```

```

## Analysis of Variance Table
##
## Response: Yield
##              Df  Sum Sq Mean Sq F value Pr(>F)
## Gen           9  9010451 1001161     7.94 3.1e-09 ***
## Env           6  56087042  9347840    74.11 < 2e-16 ***
## Rep:Env      14  2982380  213027     1.69 0.06575 .
## Env:Gen      54 15237655  282179     2.24 0.00012 ***

```

```

## Residuals 126 15893067 126135
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

#### 4.9.4 medidas de estabilidade

Além das análises individual e conjunta, a função também retorna algumas informações importantes que podem ser utilizadas em outros procedimentos, como matrizes de médias e efeitos da GEI. Medidas de estabilidade paramétricas baseadas em ANOVA como o método tradicional (Yates and Cochran 1938), a Ecovalência de Wricke (1962) e variância de Shukla (1972) são computadas e salvas no dataframe *stab\_meas*.

```
stats$stab_meas
```

	G	Mean	GenSS	Var	CV	Ecov	ShuklaVar	Ecov.perc
## 1	G1	3058	7736758	1289460	21.4	663092	34291	4.35
## 2	G10	2963	9616368	1602728	24.7	3353537	221127	22.01
## 3	G2	3006	6601798	1100300	20.1	2517614	163077	16.52
## 4	G3	3346	6478813	1079802	17.9	880613	49396	5.78
## 5	G4	3104	6083580	1013930	18.7	2083813	132952	13.68
## 6	G5	2877	5125136	854189	18.5	556290	26874	3.65
## 7	G6	2888	5560972	926829	19.2	432108	18250	2.84
## 8	G7	2872	8671919	1445320	24.2	3321378	218894	21.80
## 9	G8	3540	9125055	1520842	20.1	878172	49227	5.76
## 10	G9	3151	6324297	1054050	18.8	551037	26509	3.62

O método de regressão proposto por Eberhart and Russell (1966) também é computado e os resultados salvos na lista *ER* que contém a ANOVA e os coeficientes de regressão.

```
stats$ER$ANOVA
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
## Total	69	80335148	1164278		
## Gen	9	9010451	1001161	3.44	0.0023 **
## Env + (Gen x Env)	60	71324696	1188745		
## Env (linear)	1	56087042	56087042		
## Gen x Env(linear)	9	706590	78510	0.27	0.9799
## Pooled deviation	50	14531065	290621		
## G1	5	567432	113486	1.00	0.4203
## G10	5	3334464	666893	5.87	5.8e-05 ***
## G2	5	2414018	482804	4.25	0.0012 **
## G3	5	880608	176122	1.55	0.1778
## G4	5	1968426	393685	3.47	0.0055 **
## G5	5	508093	101619	0.90	0.4863
## G6	5	421845	84369	0.74	0.5924
## G7	5	3318407	663681	5.85	6.2e-05 ***
## G8	5	567941	113588	1.00	0.4198
## G9	5	549830	109966	0.97	0.4392

```

## Pooled error      140 15893067   113522
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
stats$ER$Regression

##      G Mean Slope    LCI   UCI R.Sqr RMSE     SSE   Delta   sdij
## G1    G1 3058 1.131 0.7649 1.50 0.927 194 567432 189144 -4216
## G10   G10 2963 1.058 0.1719 1.94 0.653 471 3334464 1111488 180252
## G2    G2 3006 0.864 0.1099 1.62 0.634 401 2414018 804673 118889
## G3    G3 3346 0.999 0.5435 1.45 0.864 242 880608 293536 16662
## G4    G4 3104 0.857 0.1755 1.54 0.676 362 1968426 656142 89183
## G5    G5 2877 0.907 0.5613 1.25 0.901 184 508093 169364 -8172
## G6    G6 2888 0.957 0.6419 1.27 0.924 168 421845 140615 -13922
## G7    G7 2872 0.977 0.0927 1.86 0.617 470 3318407 1106136 179182
## G8    G8 3540 1.235 0.8694 1.60 0.938 195 567941 189314 -4182
## G9    G9 3151 1.015 0.6547 1.37 0.913 191 549830 183277 -5390

```

Dois tipos de gráficos, um iterativo (*iterplot*) e um que pode ser salvo em pdf com alta resolução (*plot*) são confeccionados considerando o índice ambiental e a produtividade de cada genótipo.

#### 4.9.5 Regressão individual

```
stats$plot
```

```
stats$iterplot #gráfico interativo para visualização no R
```

Regressão individual da produtividade com o índice ambiental

A visualização deste gráfico só é possível em um arquivo HTML como este ou no próprio ambiente R. Caso deseje salvar este mesmo gráfico em um arquivo .pdf, por exemplo, basta utilizar o arquivo compartilhável, salvo, neste caso, no arquivo *stats\$plot*.

Todos estes resultados podem ser resumidos utilização a função *summary\_ge\_stats()*. Por exemplo, é possível salvar estes resultados em um arquivo de texto, utilizando a função abaixo

```
summary_ge_stats(stats, export = TRUE)
```

## 4.10 Interação genótipo vs ambiente (AMMI)

Métodos que combinam diferentes princípios estatísticos ganharam espaço na análise de EMA por volta da década 1960, com destaque especial ao estudo de Gollob (1968), que propôs um método que combina os benefícios da análise de fatores e análise de variância em um único método para estudar a estabilidade. Naquela época este método era conhecido como FANOVA. Atualmente este mesmo método foi popularizado por Gauch and Zobel (1988) com o acrônimo AMMI.

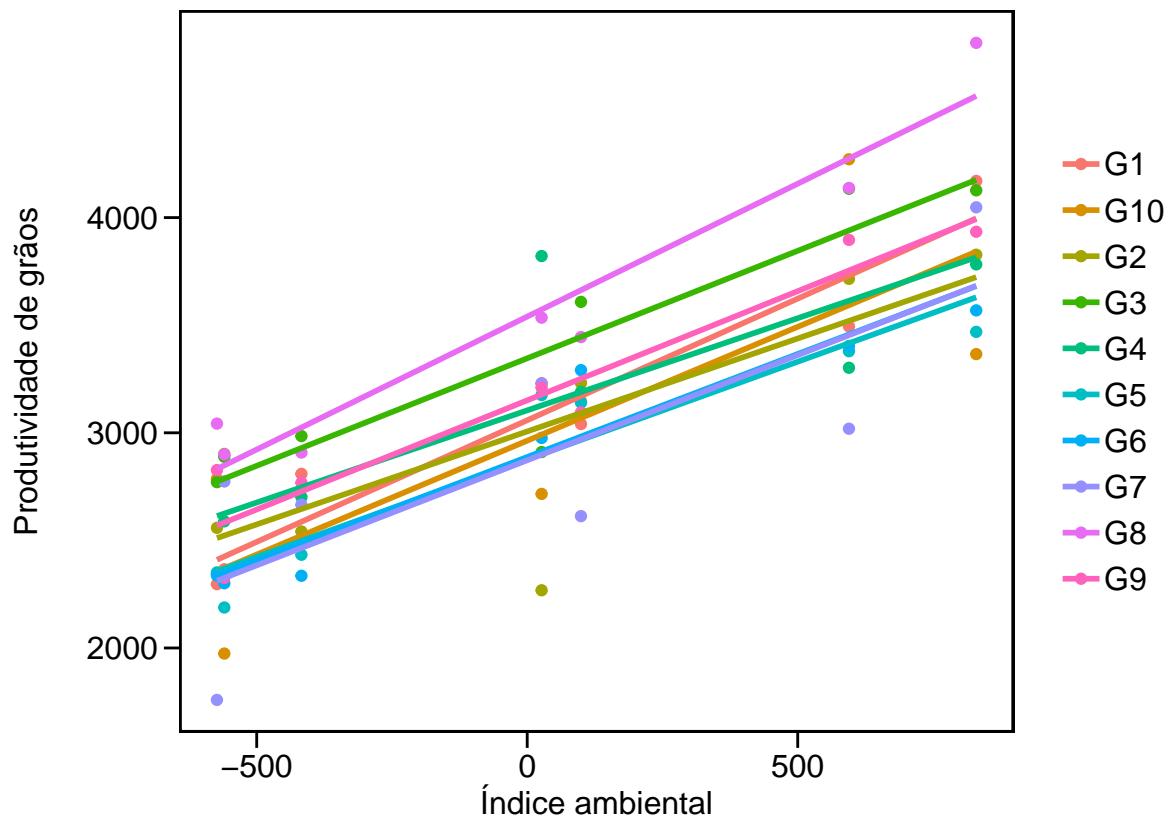


Figure 62: Regressão individual da produtividade com o índice ambiental

A análise AMMI utiliza análise aditiva de variância aos fatores principais (genótipo e ambiente) e decomposição por valores singulares ao residual do modelo aditivo, isto é, o efeito da GEI somado ao erro experimental. Esta matriz dos efeitos não aditivos, então, pode ser aproximadamente exibida por meio de biplots Gabriel (1971). Este método tem ganhado destaque nas últimas décadas, principalmente devido a rápida evolução computacional, o que tornou possível as complexas decomposições de matrizes de alta ordem.

De posse de uma matriz de dupla entrada oriunda de EMA, a estimativa da variável resposta do  $i$ -ésimo genótipo no  $j$ -ésimo ambiente é obtida utilizando AMMI de acordo com o seguinte modelo:

$$y_{ij} = \mu + \alpha_i + \tau_j + \sum_{k=1}^k \lambda_k a_{ik} t_{jk} + \rho_{ij} + \varepsilon_{ij}$$

onde  $\lambda_k$  é o valor singular para o  $k$ -ésimo eixo do componente principal;  $a_{ik}$  é o  $i$ -ésimo elemento do  $k$ -ésimo autovetor de genótipos;  $t_{jk}$  é o  $j$ -ésimo elemento do  $k$ -ésimo autovetor de ambientes. Um resíduo  $\rho_{ij}$  permanece, se todos os  $k$ -PCAs não são considerados, onde  $k = \min(G - 1; E - 1)$ .

#### 4.10.1 Ajustando o modelo

O modelo AMMI pode ser ajustado utilizando a função `WAAS` do pacote WAASB. Vamos assumir no momento apenas o ajuste do modelo. Posteriormente serão discutidos os outros argumentos desta função.

```
library(WAASB)
# Ajustando o modelo AMMI para os dados
AMMI = WAAS.AMMI(interaction,
                   resp = "RG")

# Imprimindo a ANOVA
options(digits = 2)
print(AMMI$anova)

##          Df  Sum Sq Mean Sq F value      Pr(>F) Percent Accumul
## ENV        6 5.6e+07 9347840    43.88 0.0000000275089   .
## REP(ENV)   14 3.0e+06 213027     1.69 0.0657500081587   .
## GEN         9 9.0e+06 1001161     7.94 0.0000000031208   .
## ENV:GEN   54 1.5e+07 282179     2.24 0.0001206407021   .
## PC1        14 8.6e+06 612698     4.86 0.000000000000000 56.3 56.3
## PC2        12 4.0e+06 332832     2.64 0.003500000000000 26.2 82.5
## PC3        10 1.7e+06 167672     1.33 0.221500000000000 11 93.5
## PC4         8 5.9e+05  74100     0.59 0.784700000000000 3.9 97.4
## PC5         6 2.8e+05  47068     0.37 0.896800000000000 1.9 99.3
## PC6         4 1.1e+05  28496     0.23 0.921100000000000 0.7 100
## Residuals 126 1.6e+07 126135       NA                 NA   .
## Total      209 9.9e+07 474692       NA                 NA   .
```

Observe que as somas de quadrados são as mesmas obtidos anteriormente, na ANOVA conjunta. Nesta abordagem, no entanto, a soma de quadrados da interação é decomposta em componentes principais, quais explicam, cada um, uma fração da soma de quadrados total da interação

#### 4.10.2 Escolha do número de termos multiplicativos

Conforme já discutido, a análise AMMI aplica a técnica de decomposição por valores singulares na matriz dos efeitos não aditivos do modelo ( $\mathbf{M}$ ). Logo, esta matriz pode ser aproximada pela pelo seguinte modelo:  $M = U\lambda V^T$ , onde onde  $\mathbf{U}$  é uma matriz  $g \times e$  contendo os vetores singulares de  $AA^T$  e formam a base ortonormal para os efeitos de genótipos;  $V^T$  é uma matriz  $e \times e$  que contém os vetores singulares de  $\mathbf{A}^T\mathbf{A}$  e formam a base ortonormal para os efeitos de ambientes; e  $\lambda$  é uma matriz diagonal  $e \times e$  contendo  $k$ -valores singulares de  $A^TA$ , onde  $k = \min(G - 1; E - 1)$ . Assim, diferentes modelos (dependendo do número de termos multiplicativos utilizados) podem ser utilizados para predizer o rendimento do genótipo  $i$  no ambiente  $j$ . A tabela abaixo mostra os possíveis modelos. No modelo AMMI0 apenas os efeitos aditivos são considerados. No modelo AMMI1, o primeiro termo multiplicativo é considerado, e assim por diante, até o modelo AMMIF, onde  $\min(G - 1; E - 1)$  termos são considerados.

Família AMMI	Resposta esperado do genótipo $i$ no ambiente $j$
AMMI0	$\hat{y}_{ij} = \bar{y}_{..} + \bar{y}_{.j} - \bar{y}_{..}$
AMMI1	$\hat{y}_{ij} = \bar{y}_{..} + \bar{y}_{.j} - \bar{y}_{..} + \lambda_1 a_{i1} t_{j1}$
AMMI2	$\hat{y}_{ij} = \bar{y}_{..} + \bar{y}_{.j} - \bar{y}_{..} + \lambda_1 a_{i1} t_{j1} + \lambda_2 a_{i2} t_{j2}$
...	
AMMIF	$\hat{y}_{ij} = \bar{y}_{..} + \bar{y}_{.j} - \bar{y}_{..} + \lambda_1 a_{i1} t_{j1} + \lambda_2 a_{i2} t_{j2} + \dots + \lambda_p a_{ip} t_{jp}$

A escolha do número de termos multiplicativos a ser utilizado é baseada em basicamente dois critérios de sucesso de análise: **Postdiscritive sucess** e **Predictive sucess**. Por definição, **Predictive sucess** significa literalmente a afirmação prévia do que acontecerá em algum momento futuro. Neste contexto, testes de validação cruzada (cross-validation) podem ser utilizadas para avaliar o sucesso preditivo dos membros de modelos da família AMMI. Por outro lado, **Postdiscritive sucess** significa fazer uma afirmação ou dedução sobre algo que aconteceu no passado. Na escolha do número de termos multiplicativos da análise AMMI este sucesso pode ser calculado utilizando testes como o proposto por Gollob (1968). O pacote WAASB permite estas duas abordagens.

- Postdiscritive sucess

No objeto `anova` gerado pela função `WAAS.AMMI()` testes de hipóteses são realizados e probabilidades de erro são atribuídas para cada modelo considerando a distribuição de graus de liberdade proposto por Gollob (1968). Assim é possível identificar qual é o número ideal de termos a ser considerado na predição. Em nosso exemplo, dois termos foram significativos a 5% de probabilidade de erro

- Predictive sucess

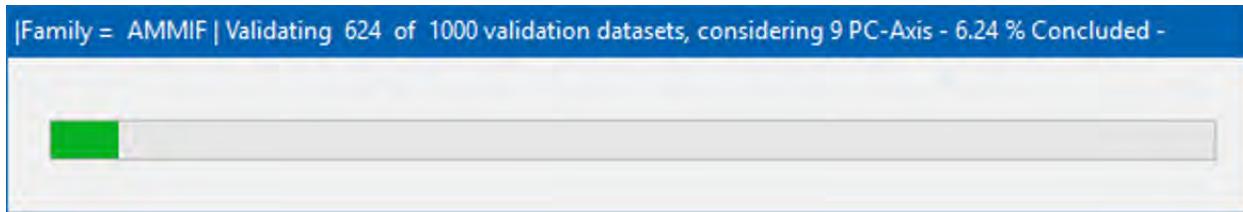


Figure 63: Barra de progresso.

O pacote WAASB fornece uma solução completa para cross-validation do modelo AMMI. Utilizando a função `validation.AMMIF()`, por exemplo, é possível realizar um teste de *cross-validation* para a família de modelos AMMI (AMMIO-AMMIF) usando dados com repetições. Automaticamente a primeira validação é realizada considerando a AMMIF (todos os eixos possíveis são usados). Considerando esse modelo, o conjunto de dados original é dividido em dois conjuntos de dados: dados de modelagem e dados de validação. O conjunto de dados “modelagem” possui todas as combinações (genótipo x ambiente) com o número de replicações informado no argumento `nrepval`. O conjunto de dados “validação” tem uma repetição. O diagrama ao lado representa o procedimento realizado. A divisão do conjunto de dados em dados de modelagem e validação depende do design informado. Considerando um delineamento de blocos completos casualizados (DBC) blocos completos são aleatoriamente selecionados dentro de ambientes, como sugerido por Piepho (1994). O bloco restante serve dados de validação. Se `design = "CRD"` for informado, assim declarando que um delineamento intericamente casualizado (DIC) foi usado, observações são aleatoriamente selecionadas para cada tratamento (combinação genótipo-vs-ambiente). Este é o mesmo procedimento sugerido por Gauch and Zobel (1988). Os valores estimados para o membro da família AMMI em estudo são então comparados com os dados de “validação” e um erro de predição  $\hat{z}_{ij}$  é estimado para cada tratamento. A raiz quadrada do quadrado médio do erro (RMSE) é calculado. Este procedimento é repetido  $n$  vezes, utilizando o argumento `nboot = n`. Ao final do procedimento, o algoritmo armazena as  $n$  estimativas do RMSE para o modelo em questão, e um novo modelo é então testado seguindo os mesmos passos.

Uma barra de progresso é mostrada por padrão. Assim, é possível verificar o status do processo. Se necessário, a barra de progresso pode ser desabilitada informando o argumento `progbar = FALSE` na função.

- Vamos agora ao exemplo.

```
# Validação cruzada para os membros da família AMMI
library(WAASB)
valida = validation.AMMIF(interaction,
                           resp = "RG",
                           nboot = 500,
                           nrepval = 2)

# Guardando os valores médios do RSME no objeto RMSEmed
RMSEmed = valida$RMSEmean

# Ordenando do mais preciso para o menos preciso
```

```
RMSEmed = RMSEmed[order(RMSEmed[,2], decreasing = F),]
```

Podemos lidar com os resultado armazenados em `valida` basicamente de duas formas distintas. A Primeira é imprimir os resultados no ambiente R usando `print(RMSEmed)`. Isso não é uma boa idéia, já que provavelmente iremos utilizar estes dados em outros outros programas, como o excel, por exemplo. Porém, vamos fazer isso agora.

```
# Imprimindo os valores médios do RMSE
options(digits = 5)
print(RMSEmed)
```

```
## MODEL mean
## 2 AMMI1 429.83
## 3 AMMI2 430.54
## 4 AMMI3 437.69
## 1 AMMI0 438.38
## 5 AMMI4 444.48
## 6 AMMI5 446.68
## 7 AMMIF 447.46
```

Em nosso exemplo, o modelo AMMI2 foi o que apresentou o menor RMSE, sendo o mais indicado para predizer o rendimento de grãos. Isto está de acordo com o teste de hipótese encontrado em `anova`.

A segunda opção é exportar os resultados para um arquivo editável, como um arquivo `.csv` ou `.xlsx`. Para isto, utilizaremos as funções abaixo.

```
# Exportando os resultados para um arquivo .csv
write.csv(RMSEmed, file = "myfile.csv")

# Exportando os resultados para um arquivo .xlsx
library(xlsx)
write.xlsx(RMSEmed, file = "myfile2.xlsx")
```

É também possível criar um gráfico com a distribuição das  $n$  estimativas do RMSE para cada modelo. Para isto iremos utilizar a função `plot.validation.AMMIF()`.

```
# Plotando os valores de RMSE
plot.validation.AMMIF(valida,
                      violin = TRUE,
                      col.boxplot = "gray75")
```

Cinco estatísticas são mostradas neste boxplot. A mediana, o primeiro e terceiro quartis (comprimento da caixa) e os valores que não ultrapassam  $1.5 \times$  a amplitude interquartílica (linhas que se estendem além da caixa). Dados além do fim dos bigodes são considerados outliers. Se a condição `violin = TRUE`, um gráfico de violino é adicionado ao boxplot. Um gráfico de violino é uma exibição compacta de uma distribuição contínua.

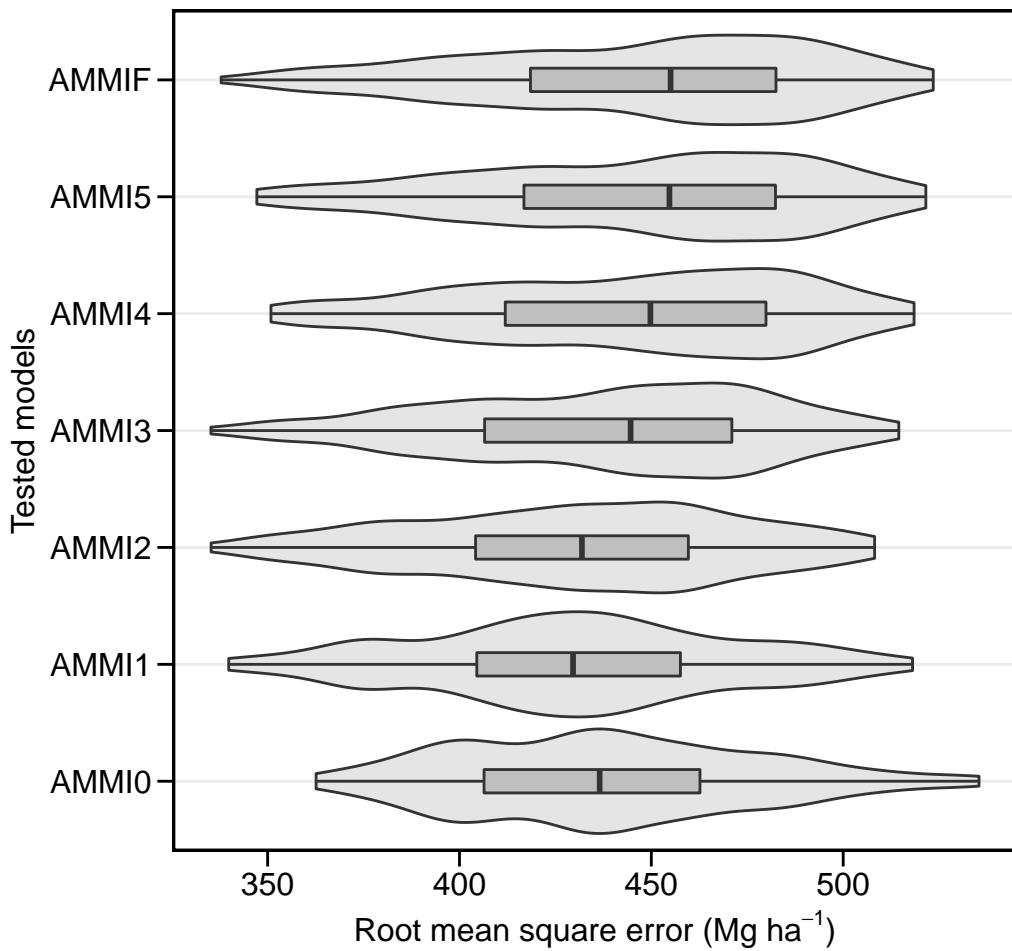


Figure 64: Gráfico boxplot demonstrando o RMSE obtido na validação cruzada dos models AMMI

#### 4.10.3 Biplots

```
options(digits = 2)
head(AMMI$model, n = 10)

##      type Code   Y    PC1    PC2    PC3    PC4    PC5    PC6 WAAS PctResp PctWAAS OrRes
## 1: GEN  G1 3058  8.4 -0.059 -8.7  1.84 -11.63 -3.62  5.7     86      97
## 2: GEN  G10 2963 -23.3 11.386 -3.6  8.75  2.20  5.98 19.5     84      91
## 3: GEN   G2 3006 -12.1 -21.491  8.0 -1.75  5.32 -4.01 15.0     85      93
## 4: GEN   G3 3346 -9.6 -9.512  4.1  3.22 -7.20  1.69  9.6     95      96
## 5: GEN   G4 3104 16.3 12.360  8.9 -0.21  5.09 -0.12 15.1     88      93
## 6: GEN   G5 2877  1.4 10.745  7.7 -2.49 -0.85 -2.26  4.4     81      98
## 7: GEN   G6 2888 -1.3  4.225 10.1 -9.25 -4.15  2.91  2.2     82      99
## 8: GEN   G7 2872 23.3 -10.907 -1.5  9.76  3.07  4.09 19.4     81      91
## 9: GEN   G8 3540  1.7 -3.920 -16.0 -12.53  3.72  4.23  2.4    100      99
## 10: GEN  G9 3151 -4.9  7.173 -8.8  2.68  4.43 -8.88  5.7     89      97
##      PesRes PesWAAS WAASY OrWAASY
## 1:      50      50    92      4
## 2:      50      50    87      9
## 3:      50      50    89      8
## 4:      50      50    95      2
## 5:      50      50    90      5
## 6:      50      50    90      7
## 7:      50      50    90      6
## 8:      50      50    86     10
## 9:      50      50    99      1
## 10:     50      50    93      3
```

Neste exemplo, os escores dos nove componentes principais são mostrados. O objeto `model` gerado pela função `WAAS.AMMI()` mostra os seguintes resultados: **type**, genótipo (GEN) ou ambiente (ENV); **Code**, o código atribuído a cada nível dos fatores; **Y**, a variável resposta (neste caso, o rendimento de grãos) **PC** são os escores dos componentes principais; e **WAAS** é a média ponderada dos escores absolutos, considerando o número de termos significativos do modelo, estimado pela seguinte equação:  $WAAS_i = \sum_{k=1}^s |PCA_{ik} \times EP_k| / \sum_{k=1}^s EP_k$ , onde  $WAAS_i$  é a média ponderada dos escores absolutos do  $i$ th genótipo;  $PCA_{ik}$  é o escore do  $i$ th genótipo no  $k$ th componente principal; e  $EP_k$  é a variância explicada pelo  $k$ th componente principal para  $k = 1, 2, \dots, s$ .

Para plotar os escores utilizaremos a função `plot.scores`.

biplot AMMI1

```
plot.scores(AMMI,
            type = 2,
            x.lab = "Produtividade de grãos (Mg/ha)")
```

biplot AMMI2

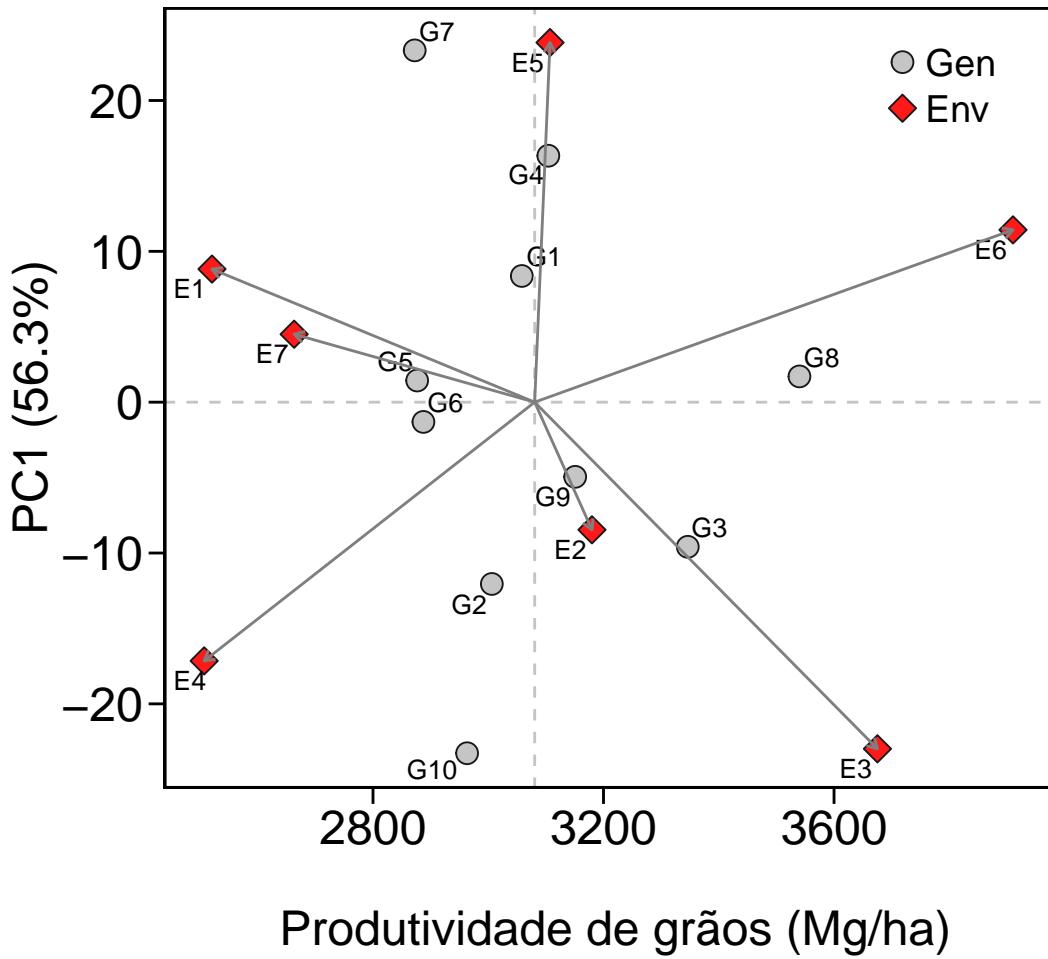


Figure 65: Biplot da produtividade de grãos com o PC1 gerado pela função plot.scores()

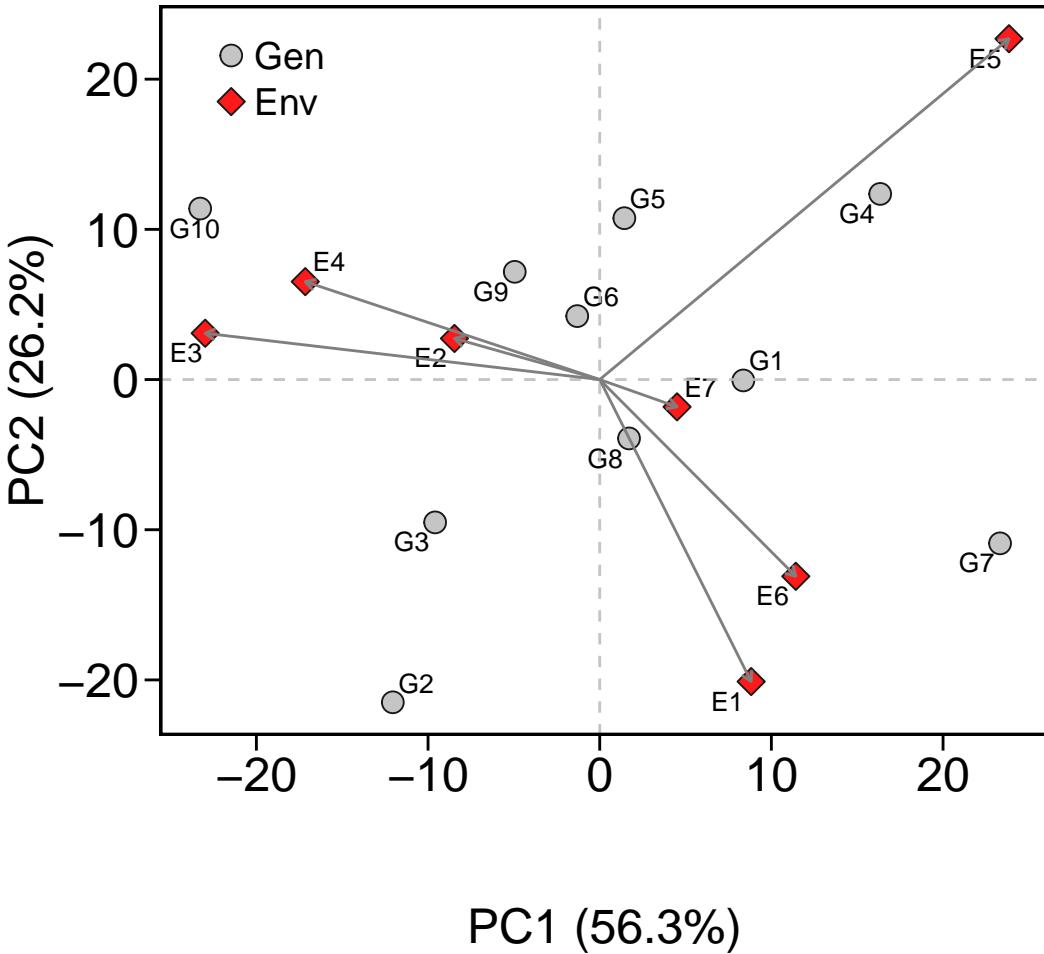


Figure 66: Biplot do PC1 com o PC2 gerado pela função `plot.scores()`

```
plot.scores(AMMI,
            type = 1,
            leg.position = "t1")
```

biplot WAAS x GY

```
plot.scores(AMMI,
            type = 3,
            x.lab = "Produtividade de grãos (Mg/ha)",
            y.lab = "Média ponderada dos escores absolutos")
```

Os quadrantes propostos no biplot acima representam quatro classificações propostas em relação à interpretação conjunta da produtividade e estabilidade. Os genótipos ou ambientes incluídos no quadrante I podem ser considerados genótipos instáveis ou ambientes com alta capacidade de discriminação, porém com produtividade abaixo da média geral. No quadrante II estão incluídos genótipos instáveis, porém, com produtividade acima da média geral. Os ambientes incluídos neste quadrante merecem atenção especial, pois, além de proporcionarem altas magnitudes da variável resposta, apresentam boa capacidade de discriminação. Os

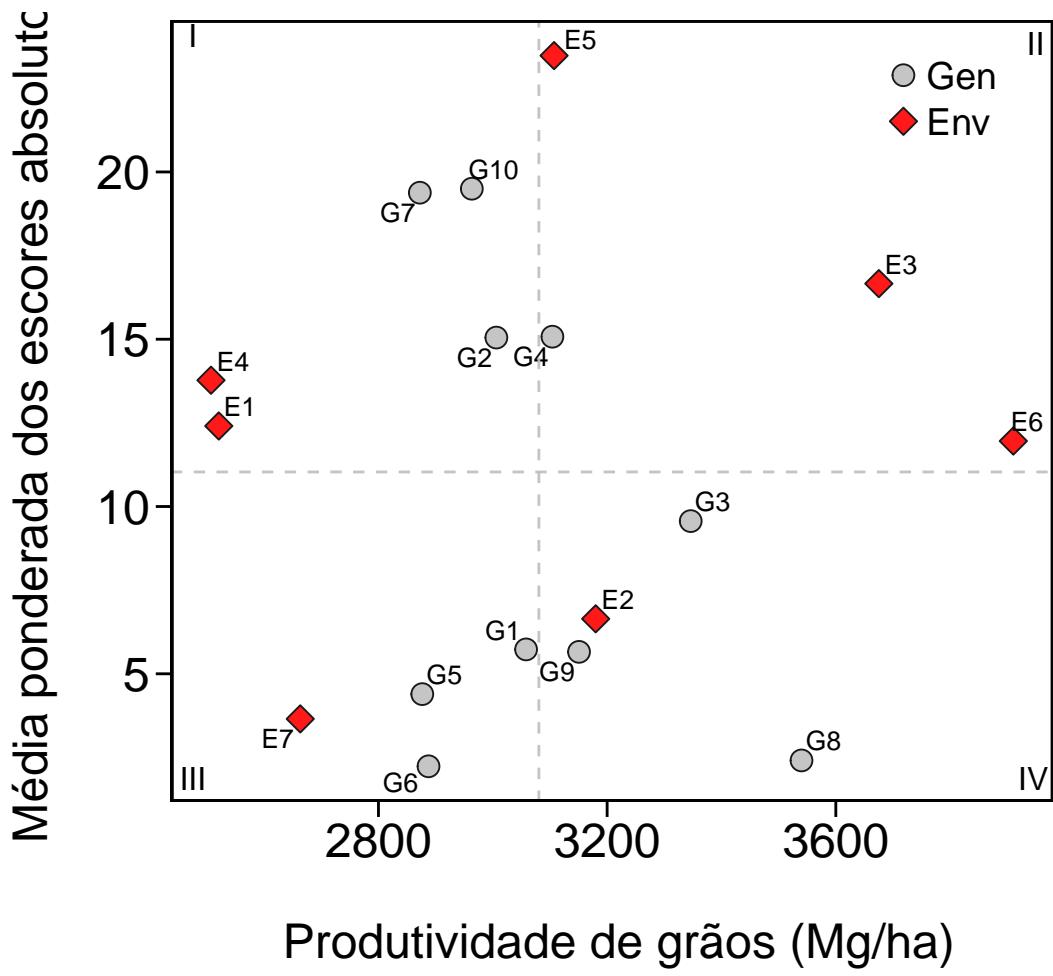


Figure 67: Biplot da produtividade de grãos com a média ponderada dos escores absolutos gerado pela função `plot.scores()`

genótipos dentro do quadrante III têm baixa produtividade, mas podem ser considerados estáveis devido aos baixos valores de WAAS. Quanto menor esse valor, mais estável o genótipo pode ser considerado. Os ambientes incluídos neste quadrante podem ser considerados pouco produtivos e com baixa capacidade de discriminação. Os genótipos dentro do quadrante IV são os genótipos que podem ser considerados como um genótipo “ideal” devido à alta magnitude da variável resposta e alta estabilidade (valores mais baixos da WAAS).

#### 4.10.4 Predição da variável resposta

A função `predict.AMMI()` pode ser usada para predizer a variável resposta em uma tabela dupla entrada (por exemplo, a produção de  $g$  genótipos em  $* n *$  ambientes) com base no modelo AMMI. Esta predição é baseada no número de termos multiplicativos declarados na função. Se `naxis = 0` for declarado, somente os efeitos principais (AMMIO) serão usados. Neste caso, a média predita será o valor predito na ANOVA convencional. Se `naxis = 1` o modelo AMMI1 (com um termo multiplicativo) é usado para predizer a variável resposta. Se `naxis = min (gen-1; env-1)`, o AMMIF é ajustada e o valor predito será a média de  $R$  repetições do genótipo  $i$  no ambiente  $j$ . O número de eixos a serem usados deve ser cuidadosamente escolhido.

- Predizendo o rendimento utilizando 2 termos multiplicativos

```
options(digits = 4)
predictAMMI = predict.AMMI(interaction,
                           resp = "RG",
                           naxis = 2)
head(predictAMMI, n = 10)

##   ENV GEN     Y  resOLS Ypred ResAMMI YpredAMMI AMMIO
## 1   E1  G1 2366 -132.57  2498    75.01      2573  2498
## 2   E1  G10 1974 -429.36  2403   -434.29     1969  2403
## 3   E1  G2 2902  455.40  2446   325.85     2772  2446
## 4   E1  G3 2889  102.26  2786   106.69     2893  2786
## 5   E1  G4 2589   44.20  2544  -104.41     2440  2544
## 6   E1  G5 2188 -128.51  2317  -203.41     2113  2317
## 7   E1  G6 2301  -26.86  2328   -96.52     2231  2328
## 8   E1  G7 2774  461.21  2313   425.09     2738  2313
## 9   E1  G8 2899  -80.74  2980   93.90     3074  2980
## 10  E1  G9 2326 -265.03  2591  -187.92     2403  2591
```

Apenas os primeiros dez valores são mostrados. Os seguintes valores são apresentados: **ENV** é o ambiente; **GEN** é o genótipo; **Y** é a variável de resposta; **resOLS** é o residual ( $\hat{z}_{ij}$ ) estimado pelo Mínimo Quadrado Ordinário (OLS), onde  $\hat{z}_{ij} = y_{ij} - \bar{y}_i - \bar{y}_j + \bar{y}_{..}$ ; **Ypred** é o valor previsto por OLS ( $\hat{y}_{ij} = y_{ij} - \hat{z}_{ij}$ ); **ResAMMI** é o residual estimado pelo modelo AMMI ( $\hat{a}_{ij}$ ) considerando o número de termos multiplicativos informado na função (neste caso 5), onde  $\hat{a}_{ij} = \lambda_1 a_{i1} t_{j1} + \lambda_2 a_{i2} t_{j2}$ ; **YpredAMMI** é o valor previsto pelo modelo AMMI ( $\hat{y}_{aij} = \bar{y}_i + \bar{y}_{..} - \bar{y}_j + \hat{a}_{ij}$ ); e **AMMIO** é o valor previsto quando nenhum termo multiplicativo é usado, ou seja,  $\hat{y}_{ij} = \bar{y}_i + \bar{y}_{..} - \bar{y}_j$ .

## 4.11 Modelos mistos na avaliação de MET

Desde a década de 1990, os modelos mistos vêm ganhando cada vez mais espaço na avaliação de MET, pois permitem a estimativa de parâmetros genéticos e ambientais, bem como a predição dos valores genotípicos de forma não-viciada (Smith, Cullis, and Thompson 2005). Da mesma forma, modelos mistos reduzem os ruídos de análises realizadas com dados desbalanceados e também de variáveis que não assumem aditividade, características frequentemente observadas em MET (Hu 2015). Na avaliação dos dados oriundos de MET é de interesse do pesquisador predizer o “verdadeiro” rendimento  $w_{ij}$  dado os rendimentos observados  $y_{ij}$ . Quando um fator é considerado fixo, as inferências são limitadas apenas aos níveis testados deste fator e os efeitos são estimados por Best Linear Unbiased Estimator, ou BLUE , ou seja, pela média aritmética. Para efeitos aleatórios, onde deseja-se expandir as inferências para uma população de tratamentos (ou ambientes), a predição é realizada por Best Linear Unbiased Predictor, ou BLUP (Henderson 1975).

### 4.11.1 Ajustando o modelo

A função `WAASB()` do pacote `WAASB` será utilizada para a análise dos dados de nosso exemplo utilizando modelos mistos. Para isto, o seguinte modelo será considerado:

$$y = Xb + Zg + Wi + \varepsilon$$

onde  $\mathbf{y}$  é o vetor de observações;  $\mathbf{b}$  é o vetor dos efeitos fixo de bloco (bloco dentro de ambiente), somados a média geral;  $\mathbf{g}$  é o vetor dos efeitos genotípicos (aleatórios), com  $g \sim NMV(0; I\hat{\sigma}_g^2)$ ;  $\mathbf{i}$  é o vetor dos efeitos da GEI (aleatório) com  $i \sim NMV(0; I\hat{\sigma}_i^2)$ ;  $\varepsilon$  é o vetor de erros aleatórios, com  $\varepsilon \sim NMV(0; I\hat{\sigma}_\varepsilon^2)$ .  $\mathbf{X}$ ,  $\mathbf{Z}$  e  $\mathbf{W}$  representam as matrizes design conhecidas que associam os parâmetros desconhecidos  $\mathbf{b}$ ,  $\mathbf{g}$  e  $\mathbf{i}$ , respectivamente, para o vetor  $\mathbf{y}$  de dados.  $\hat{\sigma}_g^2$ , é a variância genotípica;  $\hat{\sigma}_i^2$  é a variância da interação; e  $\hat{\sigma}_\varepsilon^2$  é a variância residual. A estimativa dos componentes de variância pode ser realizada sem maiores problemas utilizando ANOVA convencional quando os dados são balanceados. Quando este pressuposto não é cumprido, a estimativa baseada em Restricted Maximum Likelihood (REML) utilizando o algoritmo Expectation-Maximization (Dempster, Laird, and Rubin 1977) é a mais indicada.

```
# Função para predição via modelos mistos
misto = WAASB(interaction,
               resp = "RG",
               prob = 0.95)
```

- Teste de razão de máxima verossimilhanças (LRT)

```
print(misto$LRT)
```

	reducedG	Complete	ReducedGE	Complete
## Df	23	2.400e+01	23	2.400e+01
## AIC	3164	3.157e+03	3170	3.157e+03
## BIC	3241	3.238e+03	3247	3.238e+03

```

## logLik      -1559 -1.555e+03      -1562 -1.555e+03
## deviance    3118  3.109e+03      3124  3.109e+03
## Chisq        NA   9.066e+00       NA   1.482e+01
## Chi Df       NA   1.000e+00       NA   1.000e+00
## Pr(>Chisq)  NA   2.605e-03       NA   1.185e-04

```

O teste LRT indicou diferenças significativas para os efeitos aleatórios de genótipo e ambiente. Assim, a utilização de modelos mistos é indicada para predição do rendimento em nosso exemplo.

#### 4.11.2 Componentes de variância e parâmetros genéticos

```

print(misto$ESTIMATES)

##          Parameters             Values
## 1      GEI variance 52014.443913 (24.49% of fenotypic variance.)
## 2      Genotypic variance 34237.260292 (16.12% of fenotypic variance.)
## 3      Residual variance 126135.452823 (59.39% of fenotypic variance.)
## 4      Phenotypic variance           212387.15702799
## 5      Heritability            0.161202121498193
## 6      GEIr2                  0.2449039039872
## 7      Heribatility of means 0.661262156422673
## 8      Accuracy              0.813180273016183
## 9      rge                   0.291970104197959
## 10     CVg                  6.00648337213627
## 11     CVr                  11.5289395532015
## 12     CV ratio              0.520991834888083

```

No objeto ESTIMATES, além dos componentes de variância para os efeitos aleatórios declarados, alguns importantes parâmetros genéticos são mostrados. **Heribatility** é a herdabilidade no sentido amplo ( $h_g^2$ ) estimada por  $h_g^2 = \sigma_g^2 / (\sigma_g^2 + \sigma_i^2 + \sigma_e^2)$  onde ( $\sigma_g^2$ ) é a variância genotípica; ( $\sigma_i^2$ ) é a variância da interação GE; e ( $\sigma_e^2$ ) é a variância residual. **GEIr2** é o coeficiente de determinação do efeito da interação GE ( $r_i^2$ ) estimado por  $r_i^2 = \sigma_i^2 / (\sigma_g^2 + \sigma_i^2 + \sigma_e^2)$ ; **Heribatility of means** é a herdabilidade da média assumindo ausência de valores perdidos ( $h_{gm}^2$ ), estimada por  $h_{gm}^2 = \sigma_g^2 / [\sigma_g^2 + \sigma_i^2 / b + \sigma_e^2 / (nb)]$ , onde  $b$  e  $n$  são o número de blocos e parcelas, respectivamente; **Accuracy** é a acurácia de seleção ( $Ac$ ), estimada por  $Ac = \sqrt{h_{gm}^2}$ ; **rge** é a correlação genótipo-ambiente ( $r_{ge}$ ) estimada por  $r_{ge} = \sigma_g^2 / (\sigma_g^2 + \sigma_i^2)$ ; **CVg** é o coeficiente de variação genotípico, estimado por  $(\sigma_g^2 / \mu)^{0.5} \times 100$ , onde  $\mu$  é a média geral; **CVr** é o coeficiente de variação residual, estimado por  $(\sigma_e^2 / \mu)^{0.5} \times 100$ ; **CV ratio** é a razão entre o coeficiente de variação genotípico e residual.

#### 4.11.3 Detalhes sobre a análise

```
options (digits = 4)
print(misto$Details)

##      Parameters          Values
## 1   WgtResponse           50
## 2   WgtWAAS                50
## 3   Ngen                   10
## 4   Nenv                    7
## 5   OVmean            3080.5568
## 6   Min 1758.6667 (Genotype G7 in E4 )
## 7   Max 4812.0988 (Genotype G8 in E6 )
## 8   MinENV     Environment E4 (2507.1)
## 9   MaxENV     Environment E6 (3910.5)
## 10  MinGEN     Genotype G7 (2872.3913)
## 11  MaxGEN     Genotype G8 (3539.9741)
```

As seguintes informações são detalhadas nesta saída. **WgtResponse** é o peso da variável resposta na estimativa da WAASBY; **WgtWAAS** é o peso para a estabilidade ; **Ngen** é o número de genótipos; **Nenv** é o número de ambientes; **OVmean** é a média geral; **Min** é o mínimo valor observado (retornando o genótipo e ambiente); **Max** é o máximo valor observado; **MinENV** é o ambiente com a menor média; **MaxENV** é o ambiente com a maior média; **MinGEN** é o genótipo com a menor média; **MaxGEN** é o genótipo com a maior média.

#### 4.11.4 Médias preditas

- Imprimindo os BLUPs preditos para genótipos

```
options(digits = 4)
head(misto$BLUPgen, n = 10)

##      Rank GEN    BLUPg Predicted     LL     UL
## 1       1 G8    329.93     3410 3252 3569
## 2       2 G3    190.77     3271 3113 3430
## 3       3 G9     50.47     3131 2972 3290
## 4       4 G4     17.01     3098 2939 3256
## 5       5 G1    -16.03     3065 2906 3223
## 6       6 G2    -53.38     3027 2869 3186
## 7       7 G10   -84.20     2996 2838 3155
## 8       8 G6   -138.62     2942 2783 3101
## 9       9 G5   -146.44     2934 2775 3093
## 10      10 G7  -149.49     2931 2772 3090
```

Esta saída mostra a média predita para os genótipos testados. **BLUPg** é o efeito genotípico ( $\hat{g}_i$ ) estimado por  $\hat{g}_i = h_g^2(\bar{y}_i - \bar{y}_{..})$  onde  $h_g^2$  é o efeito *shrinkage* para genótipo, estimado por

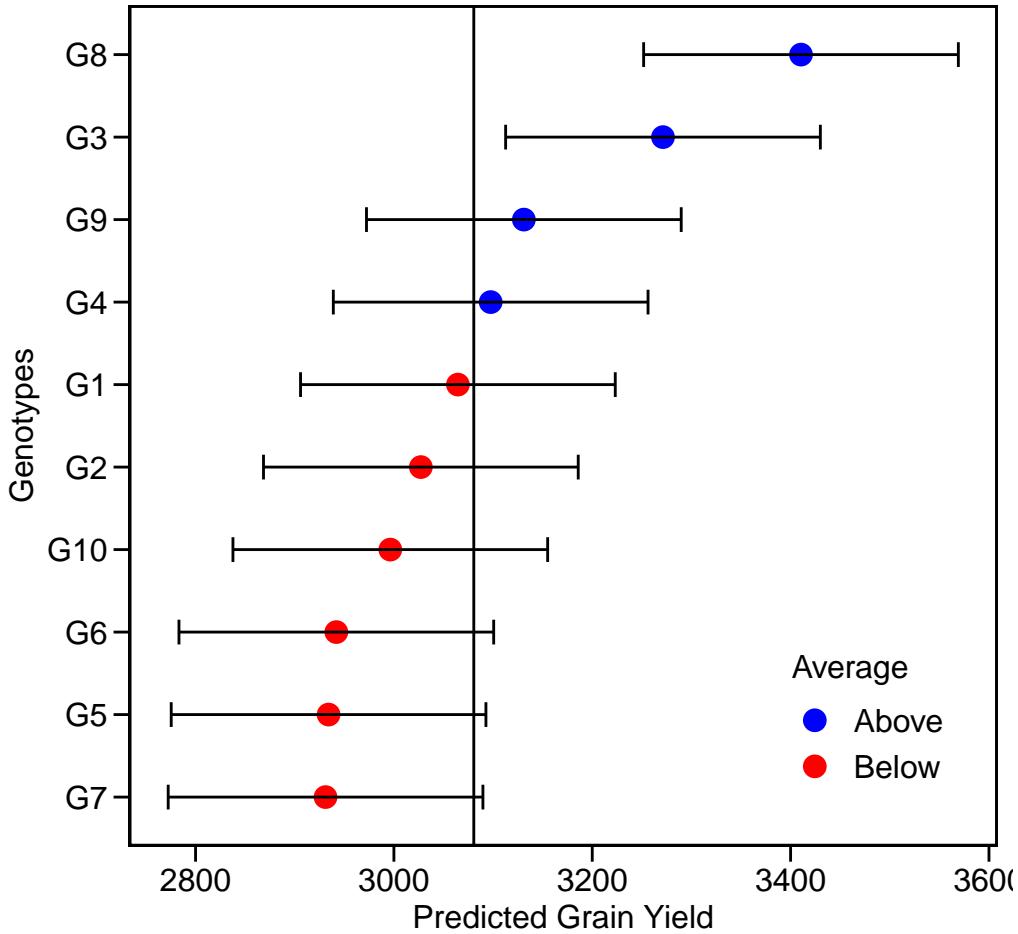


Figure 68: Médias preditas para genótipos considerando um modelo misto

$h_g^2 = (\hat{\sigma}_i^2 + E \hat{\sigma}_g^2) / (\hat{\sigma}_i^2 + \hat{\sigma}_\epsilon^2 + E \hat{\sigma}_g^2)$ . **Predicted** é a média predita por  $\hat{g}_i + \mu$  onde  $\mu$  é a média geral. **LL** e **UL** são os limites inferior e superior, respectivamente estimado por  $(\hat{g}_i + \mu) \pm CI$ . **CI** é o intervalo de confiança para predição BLUP, assumindo uma dada probabilidade de erro, onde  $CI = t \times \sqrt{((1 - Ac) \times \sigma_g^2)}$  onde  $t$  é o valor  $t$ -Student para um teste bilateral a uma dada probabilidade de erro;  $Ac$  é a acurácia de seleção; e  $\sigma_g^2$  é a variância genotípica.

- plotando os BLUPs preditos para genótipos

```
plot.blup(misto,
           leg.pos = c(0.85, 0.15))
```

- Imprimindo os BLUPs estimados para as combinações genótipo x ambiente (primeiras 10 entradas)

```
options (digits = 4)
print(misto$BLUPge[1:10,])
```

```
##   ENV GEN BLUPge    BLUPg BLUPg+ge Predicted     LL     UL
## 1   E1  G1  -76.79   -16.03   -92.82      2428  2269  2587
```

```

## 2   E1 G10 -255.71  -84.20  -339.91      2181 2022 2339
## 3   E1 G2   240.25  -53.38   186.86      2708 2549 2866
## 4   E1 G3    97.95  190.77   288.72      2809 2651 2968
## 5   E1 G4   28.14   17.01   45.14      2566 2407 2724
## 6   E1 G5 -102.85 -146.44  -249.29      2271 2113 2430
## 7   E1 G6  -44.94 -138.62  -183.56      2337 2178 2496
## 8   E1 G7  222.60 -149.49   73.11      2594 2435 2752
## 9   E1 G8   26.96  329.93   356.89      2878 2719 3036
## 10  E1 G9 -135.61   50.47  -85.14      2436 2277 2594

```

A saída acima os BLUPs estimados para as combinações genótipo x ambiente. **BLUPg** é o efeito genotípico, descrito acima; **BLUPge** é o efeito genotípico do genótipo  $i$  no ambiente  $j$  ( $\hat{g}_{ij}$ ) estimado por  $\hat{g}_{ij} = h_g^2(\bar{y}_i - \bar{y}_..) + h_{ge}^2(y_{ij} - \bar{y}_i - \bar{y}_{.j} + \bar{y}_..)$ , onde  $h_{ge}^2$  é o efeito *shrinkage* para a interação GE, estimado por  $h_{ge}^2 = \hat{\sigma}_i^2 / (\hat{\sigma}_i^2 + \hat{\sigma}_e^2)$ ; **BLUPg+ge** é  $BLUP_g + BLUP_{ge}$ ; **Predicted** é a média predita ( $\hat{y}_{ij}$ ) estimada por  $\hat{y}_{ij} = \bar{y}_{.j} + BLUP_{g+ge}$ .

## 4.12 AMMI ou BLUP? Decisão em cada caso!

Vimos anteriormente que um membro da família de modelos AMMI (AMMI2) é o mais preciso na predição da variável resposta em nosso exemplo. Após analizar-mos os mesmos dados utilizando modelos mistos, nos surge a seguinte questão. Qual é o método mais preciso para prever a variável resposta em nosso exemplo? A resposta a esta pergunta será dada agora. O pacote WAASB também conta com uma função para *cross-validation* considerando modelos mistos. A idéia é simples. Relizaremos uma validação cruzada semelhante àquela realizada no modelo AMMI e compararemos o RMSE do modelo BLUP com aqueles obtidos pela análise AMMI.

- Cross-validation para predição BLUP.

A função `validation.blup` realiza uma validação cruzada para experimentos com repetição usando modelos mistos. Por padrão, blocos completos são selecionados aleatoriamente em cada ambiente. O procedimento para calcular o RSME é idêntico ao apresentado na análise AMMI.

```

# cross-validation para predição BLUP
valida2 = validation.blup(interaction,
                           resp = "RG",
                           nboot = 500,
                           nrepval = 2)

```

- Concatenando os dados de validação do modelo AMMI e BLUP

```

options(digits = 4)
# Concatenando os valores
resultsval = rbind(valida$RMSEmean, valida2$RMSEmean)

# Ordenando em ordem crescente
resultsval = resultsval[order(resultsval[,2], decreasing = F),]

```

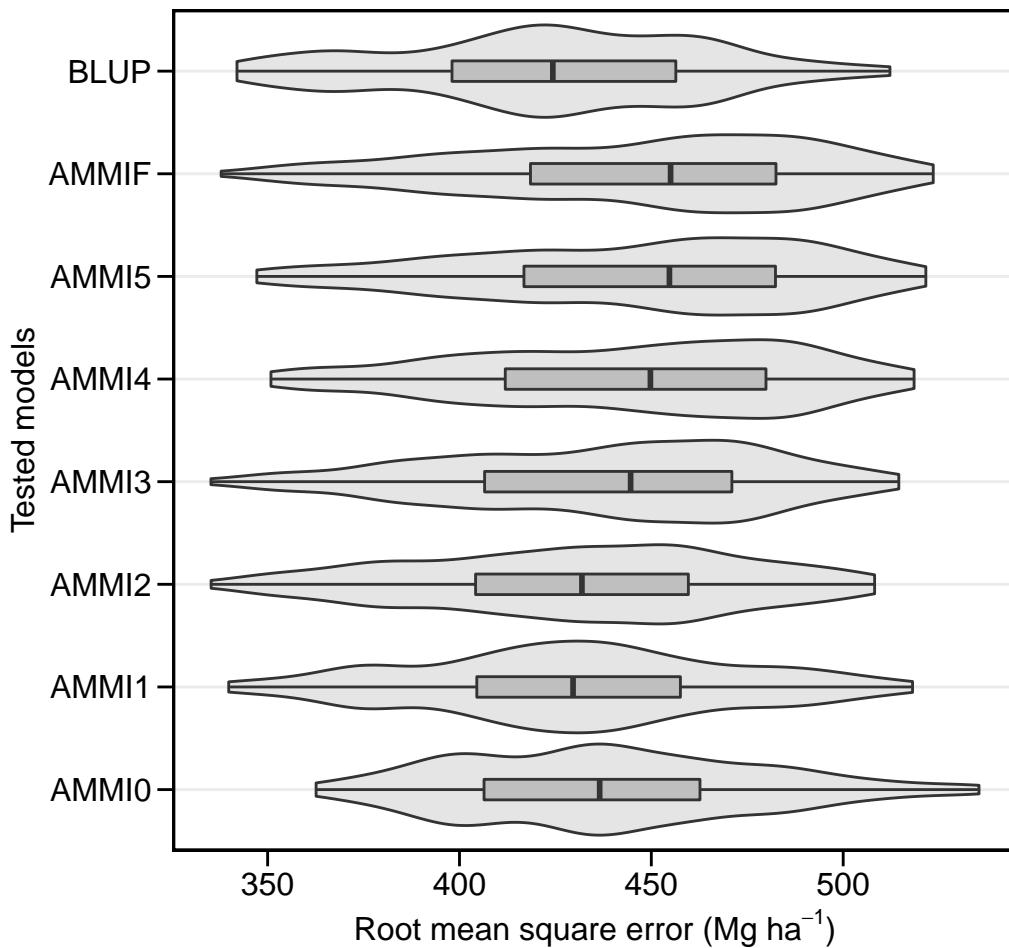
```
# Imprimindo a saída
print(resultsval)
```

```
##      MODEL  mean
## 11    BLUP 424.0
## 2     AMMI1 429.8
## 3     AMMI2 430.5
## 4     AMMI3 437.7
## 1     AMMIO 438.4
## 5     AMMI4 444.5
## 6     AMMI5 446.7
## 7     AMMIF 447.5
```

- Plotando os valores de RMSE

```
# Concatenando as 500 estimativas de RMSE de cada modelo
resultsplot = list(RMSE = rbind(valida$RMSE,
                                 valida2$RMSE))

# Plotando os valores
plot.validation.AMMIF(resultsplot,
                      violin = TRUE,
                      col.boxplot = "gray75")
```



#### 4.13 Combinando as vantagens dos métodos AMMI e BLUP

Os métodos AMMI e BLUP são dois dos métodos estatísticos mais utilizados para analisar dados de EMA. Tanto o BLUP quanto o AMMI podem ser vistos como dois métodos estatísticos distintos para alcançar o mesmo objetivo: distinguir o padrão da GEI do erro aleatório. A análise AMMI retém a maior parte do padrão GEI nos primeiros termos multiplicativos resultantes da SVD da matriz de efeitos residuais não aditivos, enquanto a maior parte do erro aleatório é retido nos últimos termos. Esse método, no entanto, tem limitações conhecidas inerentes aos modelos de efeitos fixos, incluindo a dificuldade no tratamento da heterogeneidade de variância experimentos desbalanceados. Por outro lado, o BLUP inicialmente estima os efeitos do modelo ANOVA e, em seguida, atribui pesos a esses efeitos. Devido ao seu efeito *shrinkage*, é um dos modelos de predição mais precisos e também pode ser usado com dados ausentes e variância heterogênea. Uma limitação do BLUP na análise de EMA, entretanto, é a dificuldade em modelar e interpretar os padrões da GEI. Piepho (1994) sugeriu “[...] Em trabalhos futuros, pode ser promissor combinar os princípios de AMMI e BLUP [...]”.

#### 4.13.1 Decomposição por valores singulares da matriz BLUP

Considere  $\mathbf{A}$  uma matriz  $g \times e$ , com os efeitos genotípicos preditos de  $g$  genótipos em  $e$  ambientes. Esta matriz também pode ser decomposta utilizando SVD, semelhante àquela realizada na análise AMMI com a matriz residual com os efeitos não aditivos. Neste ponto, dois problemas importantes da análise AMMI convencional são contornados: Primeiro, a matriz  $\mathbf{A}$  é composta dos efeitos “puros” da interação, ou seja, livre do ruído. Isto é possível devido ao efeito *shrinkage* do preditor BLUP. Assim é possível utilizar a SVD para modelar os efeitos genotípicos da GEI; O segundo, e talvez o mais importante, é que efeitos aleatórios podem ser incluídos no modelo, diferentemente da análise AMMI convencional. Considerando genótipo ou ambiente como fator aleatório (caracterizando assim um modelo misto) a GEI será sempre aleatória, o que possibilita a estimativa de  $\mathbf{A}$  e, consequentemente, a estimativa de escores para genótipos e ambientes nos k-possíveis termos multiplicativos (PCA).

Os escores podem ser facilmente obtidos utilizando novamente a função `WAASB()`.

- Estimando escores para genótipos e ambientes considerando um modelo misto.

```
# Estimando os escores
```

```
escmisto = WAASB(interaction,
                  resp = "RG",
                  prob = 0.95)
```

```
# Imprimindo os escores
```

```
print(escmisto$model[, c(1:10)])
```

	##	type	Code	Y	PC1	PC2	PC3	PC4	PC5	PC6	WAASB
## 1:	1:	GEN	G1	3058	190.06	-4.481	-116.16	59.81	-115.83	20.616	123.77
## 2:	2:	GEN	G10	2963	-526.58	220.064	-24.26	122.96	22.48	-2.584	350.58
## 3:	3:	GEN	G2	3006	-273.34	-394.208	150.93	-16.39	47.34	37.252	269.24
## 4:	4:	GEN	G3	3346	-222.18	-185.147	27.44	-27.23	-55.80	-116.218	174.94
## 5:	5:	GEN	G4	3104	371.94	231.084	93.91	-64.21	57.92	-66.447	277.05
## 6:	6:	GEN	G5	2877	36.64	211.394	125.20	-29.02	-11.92	29.553	91.11
## 7:	7:	GEN	G6	2888	-26.19	89.783	157.40	-100.76	-48.55	59.819	63.01
## 8:	8:	GEN	G7	2872	532.72	-198.660	28.48	140.52	29.73	9.010	350.04
## 9:	9:	GEN	G8	3540	30.56	-98.755	-306.83	-134.78	32.31	19.253	86.87
## 10:	10:	GEN	G9	3151	-113.62	128.924	-136.12	49.09	42.32	9.745	114.43
## 11:	11:	ENV	E1	2521	187.37	-394.147	119.05	-58.17	93.26	-61.994	221.79
## 12:	12:	ENV	E2	3180	-204.18	36.036	140.76	-189.23	-87.81	-49.823	148.47
## 13:	13:	ENV	E3	3675	-536.01	34.244	-161.25	54.50	19.40	-69.013	323.27
## 14:	14:	ENV	E4	2507	-403.50	97.431	-133.84	-109.58	74.86	5.230	266.65
## 15:	15:	ENV	E5	3107	530.43	400.357	-91.21	-61.05	39.95	-53.667	404.98
## 16:	16:	ENV	E6	3910	243.18	-282.014	-334.50	-80.83	-46.58	17.307	249.85
## 17:	17:	ENV	E7	2663	89.84	-54.610	-47.55	103.50	-51.90	-102.197	75.79

- Imprimindo os autovalores da matriz  $M$

```
print(escmisto$PCA)
```

```
##   PC Eigenvalue Proportion Accumulated
```

```

## 1 1     875489    54.220    54.22
## 2 2     410116    25.399    79.62
## 3 3     200368    12.409    92.03
## 4 4      75143     4.654    96.68
## 5 5     28849     1.787    98.47
## 6 6     24740     1.532   100.00

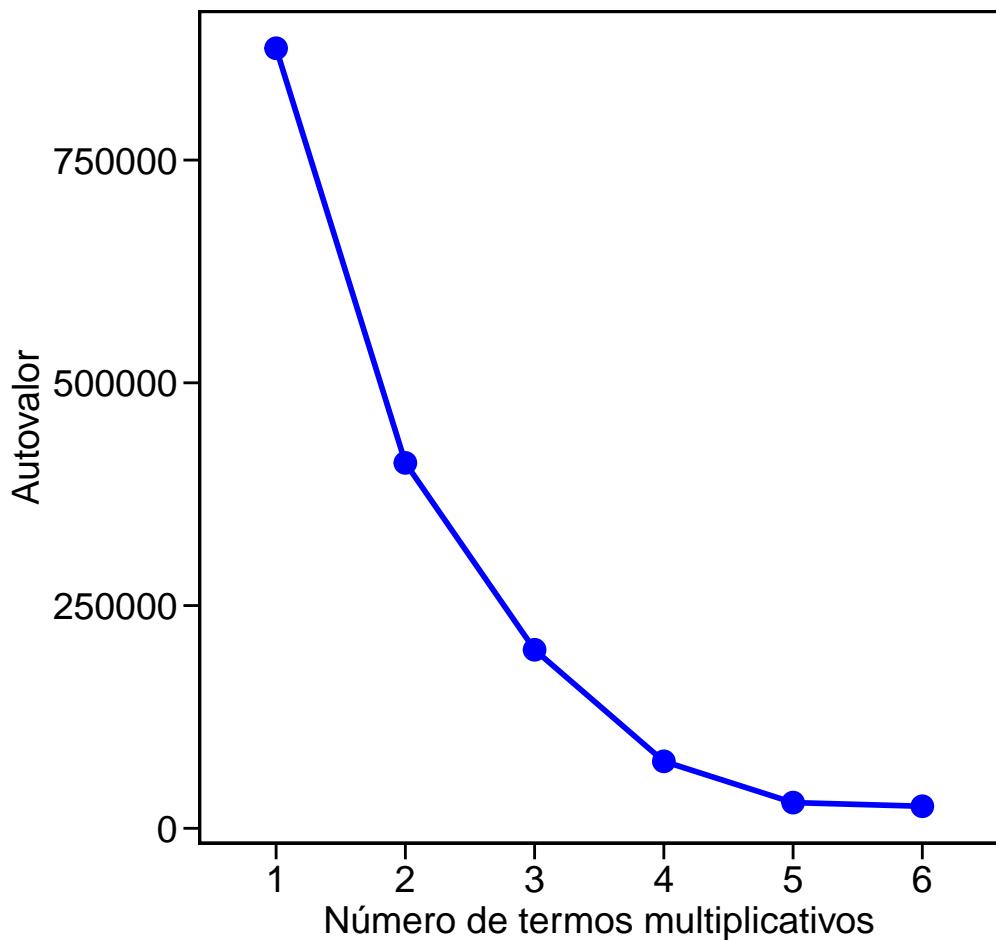
```

- Plotando os autovalores da matriz  $M$

```

plot.eigen(escmisto,
            size.lab = 14,
            size.tex = 14,
            x.lab = "Número de termos multiplicativos",
            y.lab = "Autovalor")

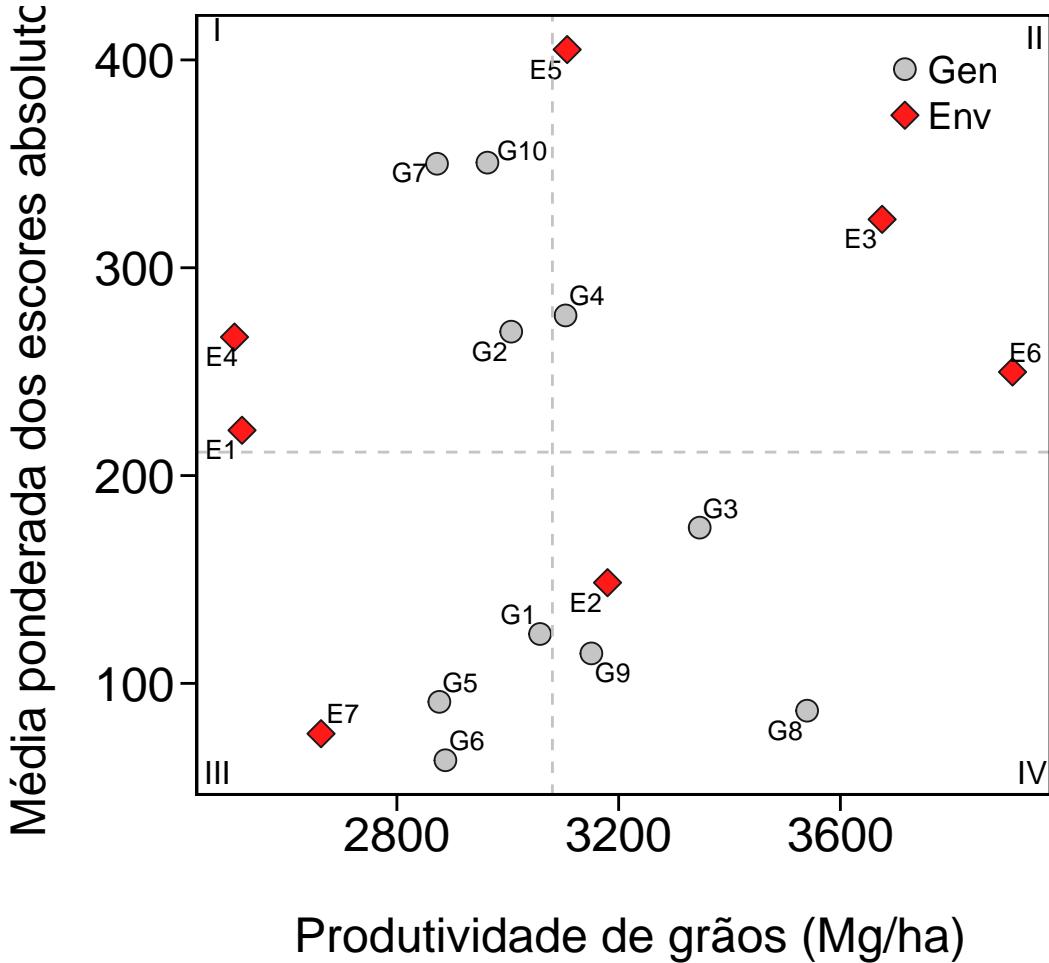
```



A saída acima mostra os autovalores e a proporção da variância explicada por cada eixo do componente principal da matriz de efeitos de interação BLUP.

#### 4.13.2 Interpretação conjunta da estabilidade e produtividade.

```
plot.scores(escmisto,
            type = 3,
            x.lab = "Produtividade de grãos (Mg/ha)",
            y.lab = "Média ponderada dos escores absolutos")
```



Este biplot tem a mesma interpretação daquele obtido pela análise AMMI convencional. A principal diferença é que aqui, todos os termos multiplicativos são utilizados para a estimativa da WAASB, diferentemente da análise AMMI, em que para nosso exemplo, apenas dois termos foram utilizados (termos significativos a 5% de probabilidade de erro). A figura abaixo compara o biplot obtido pela análise AMMI(a) e considerando o modelo misto (b). É possível notar que a magnitude da WAASB foi maior para o modelo misto, isto é devido aos valores dos BLUPs da interação serem maiores que o residual não aditivo obtido na análise AMMI. Os escores dos genótipos e ambientes, no entanto não apresentou mudanças significativas. Isto é devido, principalmente a alta percentagem explicada nos dois primeiros termos multiplicativos da análise AMMI, indicando uma predominância de interação GE simples em nosso exemplo.

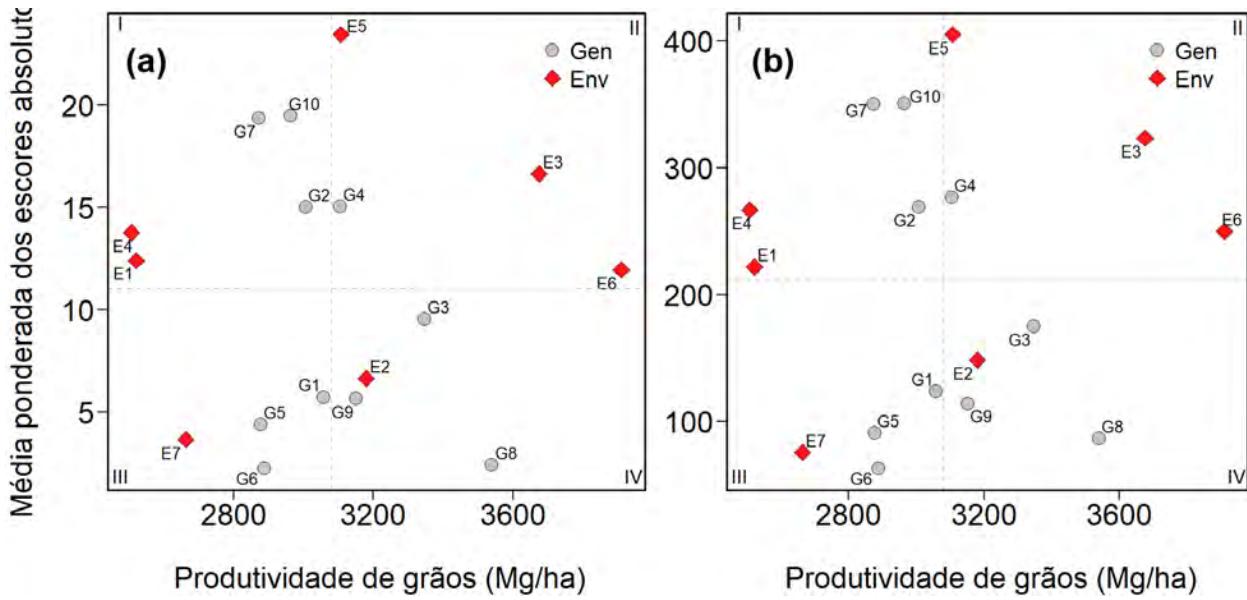


Figure 69: Biplot WAAS x RG (a) e WAASB x RG (b)

#### 4.13.3 Atribuindo pesos para a estabilidade e variável resposta

A função `WAASBratio()` considera tanto a estabilidade (média ponderada dos escores absolutos obtidos da SVD da matriz de efeitos de interação BLUP) quanto a produtividade para a classificação dos genótipos, considerando o seguinte modelo:

$$WAASBY_i = \frac{[W_{GY} \times \left( \left( \frac{GY_i}{GY_{max}} \right) \times 100 \right)] + [W_S \times \left( 100 - \frac{WAASB_i}{WAASB_{min}} \right)]}{W_{GY} + W_S}$$

onde  $WAASBY_i$  é a média ponderada dos escores absolutos e da produtividade do genótipo  $i$ ;  $W_{GY}$  é o peso para o rendimento de grãos;  $GY_i$  é o rendimento médio de grãos do genótipo  $i$ ;  $GY_{max}$  é o maior rendimento médio de grãos entre os genótipos;  $W_S$  é o peso para a estabilidade;  $WAASB_i$  é a média ponderada dos escores absolutos do genótipo  $i$ ; e  $WAASB_{min}$  é o menor valor da WAASB.

Esta função fornece a opção de atribuir pesos para estabilidade e produtividade na classificação dos genótipos. Isso é importante dependendo do objetivo de uma estratégia de seleção. Por exemplo, se o objetivo de um programa de melhoramento é selecionar um genótipo com alto rendimento (independentemente do desempenho de estabilidade), o genótipo com a primeira classificação em uma relação  $WAASB/GY = 0/100$  deve ser selecionado. A reciproca é verdadeira. Com o objetivo de selecionar um genótipo altamente estável (independentemente da produtividade), deve-se selecionar aquele genótipo com a primeira classificação em uma relação  $WAASB/GY = 100/0$ . Por padrão, o incremento na relação  $WAASB/GY$  é igual a 5 e os valores  $WAASBY$  são salvos quando a relação  $WAASB/GY$  é igual a 50/50. Portanto, vinte e uma combinações diferentes são calculadas, e para cada combinação, os genótipos são classificados em relação aos valores de  $WAASBY$ .

No exemplo a seguir, vamos supor que queremos obter as classificações alterando a relação

WAASB/GY em 10% em cada cenário e plotar os valores de WAASBY considerando uma relação WAASB / GY igual a 30/70. **Importante! O ARGUMENTO saveWAASY DEVE SER DIVISÍVEL POR increment!**

```
WAASBYratio = WAASBYratio(interaction,
                           resp = "RG",
                           increment = 10,
                           saveWAASY = 30)
```

- Imprimindo os valores da WAASBY

```
print(WAASBYratio$WAASY)
```

```
##      Code PesRes PesWAAS WAASY  Mean
## 1:    G7     70     30 85.13 below
## 2:    G5     70     30 86.45 below
## 3:    G6     70     30 86.80 below
## 4:   G10     70     30 86.93 below
## 5:    G2     70     30 88.16 below
## 6:    G1     70     30 89.88 below
## 7:    G4     70     30 90.06 above
## 8:    G9     70     30 91.76 above
## 9:    G3     70     30 95.34 above
## 10:   G8     70     30 99.59 above
```

- Plotando os valores da WAASBY

```
plot.WAASBY(WAASBYratio,
             legend.pos = c(0.9, 0.2),
             y.lab = "Genótipo")
```

- Imprimindo o ranqueamento dos genótipos dependendo do número de termos multiplicativos utilizado para estimar a WAASB.

```
print(WAASBYratio$hetdata)
```

```
##      6PCA 5PCA 4PCA 3PCA 2PCA 1PCA
## G1      5     5     5     5     5     5
## G10     10    10    10    10    10     9
## G2      7     7     7     7     7     7
## G3      6     6     6     6     6     6
## G4      8     8     8     8     8     8
## G5      3     3     3     3     3     3
## G6      1     1     1     1     1     1
## G7      9     9     9     9     9    10
## G8      2     2     2     2     2     2
## G9      4     4     4     4     4     4
```

- Heatmap ranks dos genótipos dependendo do número de PCA utilizados para estimar a WAASB

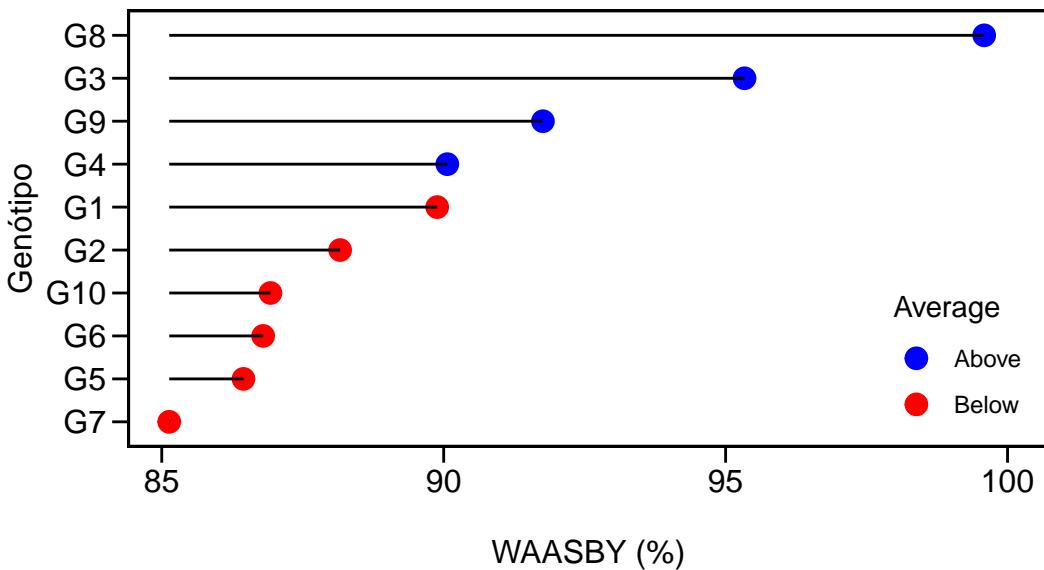


Figure 70: Índice WAASBY proposto para a classificação dos genótipos de acordo com pesos atribuídos para produtividade e estabilidade

```
plot.WAASBYratio(WAASBYratio,
                  type = 1,
                  key.lab = "Classificação",
                  x.lab = "Número de termos multiplicativos",
                  y.lab = "Genótipos")
```

Este *heatmap* mostra a classificação dos genótipos dependendo do número termos multiplicativos usados para estimar o índice WAASB. Um dendrograma baseado na distância euclidiana (linha e coluna) é usado para formação dos grupos.

- Imprimindo o ranqueamento dos genótipos dependendo do peso na relação WAASB/GY.

```
print(WAASBYratio$hetcomb)
```

	100/0	90/10	80/20	70/30	60/40	50/50	40/60	30/70	20/80	10/90	0/100
## G1	5	5	4	4	4	4	4	5	5	5	5
## G10	10	9	9	9	9	9	9	7	7	7	7
## G2	7	8	8	8	8	6	6	6	6	6	6
## G3	6	4	2	2	2	2	2	2	2	2	2
## G4	8	7	7	7	5	5	5	4	4	4	4
## G5	3	6	6	6	7	8	8	9	9	9	9
## G6	1	3	5	5	6	7	7	8	8	8	8
## G7	9	10	10	10	10	10	10	10	10	10	10
## G8	2	1	1	1	1	1	1	1	1	1	1
## G9	4	2	3	3	3	3	3	3	3	3	3

- Heatmap do ranqueamento dos genótipos dependendo do peso na relação WAASB/GY.

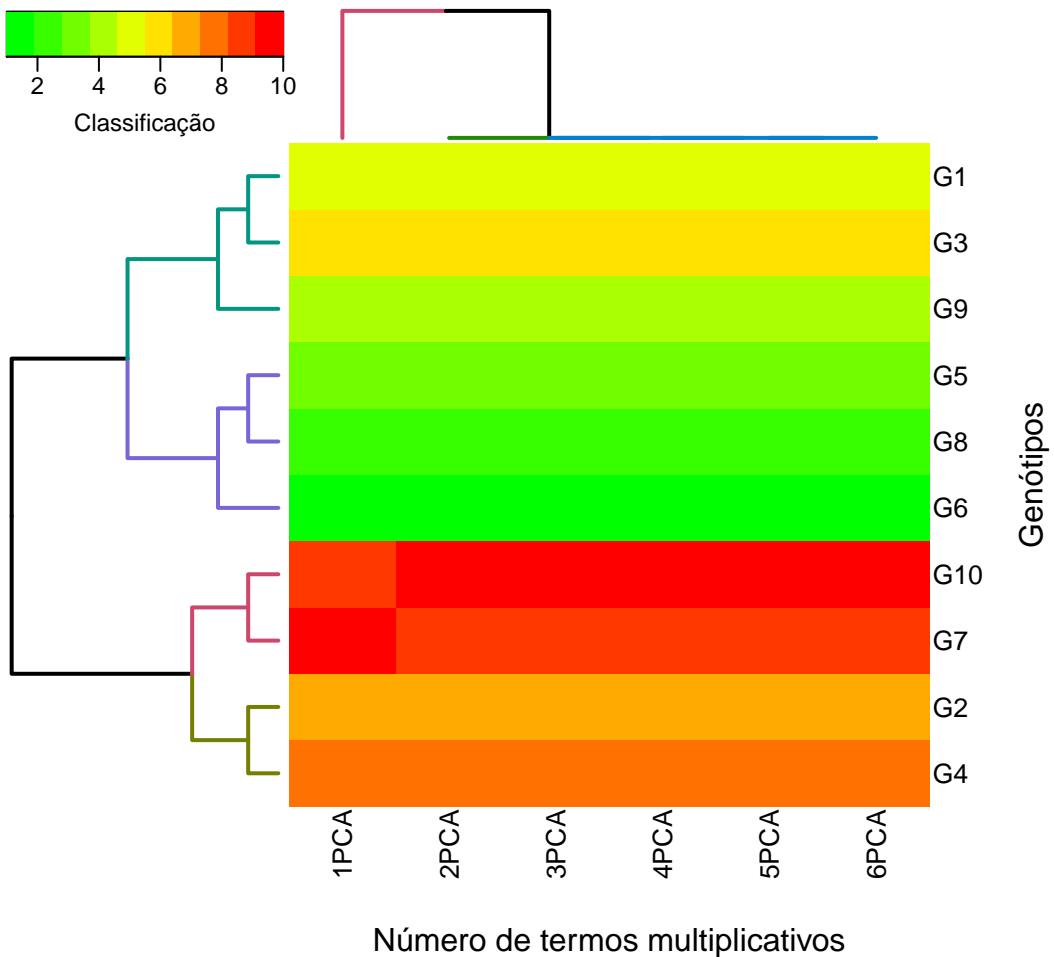


Figure 71: Ranqueamento dos genótipos dependendo do número de componentes principais considerados no modelo

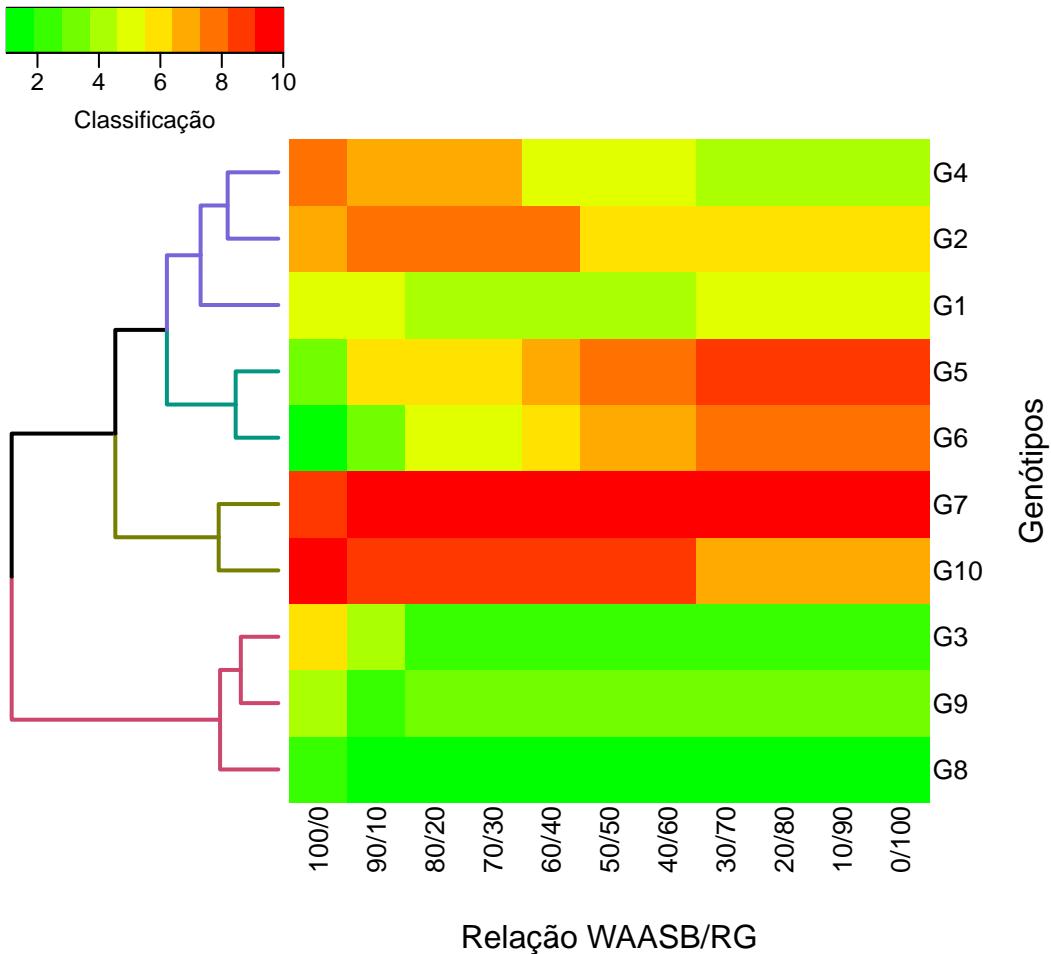


Figure 72: Ranqueamento dos genótipos dependendo do peso considerado para a produtividade e estabilidade.

```
plot.WAASBYratio(WAASBYratio,
                  type = 2,
                  key.lab = "Classificação",
                  x.lab = "Relação WAASB/RG",
                  y.lab = "Genótipos")
```

Este *heatmap* mostra a classificação dos genótipos dependendo da relação WAASB/GY considerada. As classificações obtidas com uma razão de 100/0 consideram exclusivamente a estabilidade para o ranqueamento dos genótipos. Por outro lado, uma relação 0/100 considera exclusivamente a produtividade para o ranqueamento dos genótipos.

## Referências

- Anderson, T. W. 2003. *An Introduction to Multivariate Statistical Analysis*. John Wiley & Sons.
- Bates, D. M., and D. G. Watts. 1988. *Nonlinear Regression Analysis and Its Applications*. John Wiley & Sons.
- Blalock, H. M. 1963. “Correlated Independent Variables: The Problem of Multicollinearity.” *Social Forces* 42: 233–37.
- Box, G. E. P., and D. R. Cox. 1964. “An Analysis of Transformations.” *J R Stat Soc Series B Stat Methodol* 26 (2): 211–52.
- Casella, G. 2008. *Statistical Design*. Springer.
- Charrad, M., Ghazzali N., Boiteau V., and A. Niknafs. 2014. “NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set.” *Journal of Statistical Software* 6: 1–36. doi:10.18637/jss.v061.i06.
- Dempster, A. P., Laird N. M., and D. B. Rubin. 1977. “Maximum Likelihood from Incomplete Data via the Em Algorithm.” *J. R. Stat. Soc. Ser. B* 39: 1–38.
- Draper, N. R., and H. Smith. 1998. *Applied Regression Analysis*. John Wiley & Sons.
- Eberhart, S. A., and W. A. Russell. 1966. “Stability Parameters for Comparing Varieties.” *Crop Science* 6: 36–40.
- Ferreira, D. F. 2009. *Estatística Basica*. Editora da UFLA.
- Ferreira, E.B., P.P. Cavalcanti, and D.A. Nogueira. 2018. *ExpDes: Experimental Designs*. <https://CRAN.R-project.org/package=ExpDes>.
- Gabriel, K.R. 1971. “The Biplot Graphic Display of Matrices with Application to Principal Component Analysis.” *Biometrika* 58: 453–67. doi:10.2307/2334381.
- Galton, F. 1888. “Co-Relations and Their Measurement, Chiefly from Anthropometric Data.” *Proc. R. Soc. Lond.* 45: 135–45.
- Gauch, H. G. 2013. “A Simple Protocol for Ammi Analysis of Yield Trials.” *Crop Science* 53: 1860–9. doi:10.2135/cropsci2013.04.0241.
- Gauch, H.G, and Zobel R.W. 1988. “Predictive and Postdictive Success of Statistical Analyses of Yield Trials.” *Theor. Appl. Genet.* 76: 1–10. doi:10.1007/BF00288824.
- Gollob, H. F. 1968. “A Statistical Model Which Combines Features of Factor Analytic and Analysis of Variance Techniques.” *Psychometrika* 33: 73–115. doi:10.1007/BF02289676.
- Graham, M. H. 2003. “Confronting Multicollinearity in Ecological Multiple Regression.” *Ecology* 84: 2809–15.
- Halkidi, M., Batistakis Y., and M. Vazirgiannis. 2001. “On Clustering Validation Techniques.” *Journal of Intelligent Information Systems* 17: 23–34. doi:10.2307/2531893.
- Hartigan, J. A., and M. A. Wong. 1977. “Hierarchical Grouping Methods and Stopping Rules:

- An Evaluation.” *J. R. Stat. Soc. Ser. C Appl. Stat.* 28: 100–108. doi:10.2307/2346830.
- Henderson, C.R. 1975. “Best Linear Unbiased Estimation and Prediction Under a Selection Model.” *Biometrics* 31: 423–47. doi:10.2307/2529430.
- Hoerl, A.E., and R.W. Kennard. 1970. “Ridge Regression: Biased Estimation for Nonorthogonal Problems.” *Technometrics* 12: 55–67. doi:10.1080/00401706.1970.10488634.
- . 1976. “Ridge Regression Iterative Estimation of the Biasing Parameter.” *Commun. Stat. Theory Methods* 9: 77–88. doi:10.1080/03610927608827333.
- Hu, X. 2015. “A Comprehensive Comparison Between Anova and Blup to Valuate Location-Specific Genotype Effects for Rape Cultivar Trials with Random Locations.” *F. Crop. Res.* 179: 144–49. doi:10.1016/j.fcr.2015.04.023.
- Hubert, L., and P. Arabie. 1985. “Comparing Partitions.” *Journal of Classification* 2: 193–218. doi:10.1007/BF01908075.
- Krzanowski, W. J., and Y. T. Lai. 1988. “A Criterion for Determining the Number of Groups in a Data Set Using Sum-of-Squares Clustering.” *Biometrics* 27: 23–34. doi:10.2307/2531893.
- Kutner, M. H., Nachtsheim C.J., Neter J., and W. Li. 2005. *Applied Linear Statistical Models*. McGraw-Hill/Irwin.
- Lucio, A. D., and B. G. Sari. 2017. “Planning and Implementing Experiments and Analyzing Experimental Data in Vegetable Crops - Problems and Solutions.” *Hortic. Bras.* 35 (3): 316–27. doi:10.1590/s0102-053620170302.
- Lucio, A. D., Nunes L. F., and F. Rego. 2016a. “Nonlinear Models to Describe Production of Fruit in Cucurbita Pepo and Capiscum Annum.” *Sci Hortic* 193: 286–93. doi:10.1590/s0102-053620160409.
- . 2016b. “Nonlinear Regression and Plot Size to Estimate Green Beans Production.” *Hortic. Bras.* 34 (4): 507–13. doi:10.1590/s0102-053620160409.
- Lucio, A. D., Sari B. G., Rodrigues M., Bevilaqua L. M., Voss H. M. G., Copetti D., and M. Fae. 2016. “Nonlinear Regression and Plot Size to Estimate Green Beans Production.” *Cienc. Rural* 46 (2): 233–41. doi:10.1590/0103-8478cr20150067.
- Milligan, G. W., and M. C. Cooper. 1985. “An Examination of Procedures for Determining the Number of Clusters in a Data Set.” *Bioinformatics* 50: 159–79. doi:10.1007/BF02294245.
- Mojena, R. 1977. “Hierarchical Grouping Methods and Stopping Rules: An Evaluation.” *The Computer Journal* 20: 359–63. doi:10.1093/comjnl/20.4.359.
- Niles, H. E. 1922. “Correlation, Causation and Wright’s Theory of ‘Path Coefficients’” *Genetics* 7: 258–73.
- Olivoto, T. 2018a. *CursoR: Curso R Do Grupo Experimentacao Da Ufsm-Sm.* <https://github.com/TiagoOlivoto/cursoR>.
- . 2018b. *WAASB: Weighted Average of Absolute Scores from Svd of Blup-Interaction Effects.* <https://github.com/TiagoOlivoto/WAASB>.
- Olivoto, T., and B. G. Sari. 2018a. “Data for Two-Way Analysis of Variance.” *Mendeley*

*Data* V1. doi:10.17632/stwhhd6tj3.1.

———. 2018b. “Data from a Maize Experiment for Multivariate Analysis.” *Mendeley Data* V1. doi:10.17632/42xhf7bmj.1.

Olivoto, T., Lucio A. D. C., Souza V. Q., Nardino M., Diel M. I., Sari B. G., Krysczun D. K., Meira D., and C. Meier. 2018. “Confidence Interval Width for Pearson’s Correlation Coefficient: A Gaussian-Independent Estimator Based on Sample Size and Strength of Association.” *Agronomy Journal* 110: 1–8. doi:10.2134/agronj2017.09.0566.

Olivoto, T., Nardino M., Carvalho I. R., Follmann D. N., Ferrari M., Szareski V. J., and Souza Pelegrin A. J. 2017. “REML/Blup and Sequential Path Analysis in Estimating Genotypic Values and Interrelationships Among Simple Maize Grain Yield-Related Traits.” *Gen. Mol. Res.* 12: 93–103. doi:10.5897/AJAR2016.11799.

Olivoto, T., Souza V. Q., Nardino M., Carvalho I. R., Ferrari M., Pelegrin A. J., Szareski V. J., and D. Schmidt. 2017. “Multicollinearity in Path Analysis: A Simple Method to Reduce Its Effects.” *Agron. J.* 109: 131–42. doi:10.2134/agronj2016.04.0196.

Pearson, K. 1920. “Notes on the History of Correlation.” *Biometrika* 13: 25–45.

Piepho, H.P. 1994. “Best Linear Unbiased Prediction (Blup) for Regional Yield Trials: A Comparison to Additive Main Effects and Multiplicative Interaction (Ammi) Analysis.” *Theor. Appl. Genet.* 89 (5): 647–54. doi:10.1007/BF00222462.

Pinheiro, J. C., and D. M. Bates. 2000. *Mixed-Effects Models in S an S-Plus*. Springer.

Rencher, A.C., and G. B. Schaalje. 2008. *Linear Models in Statistics*. John Wiley & Sons.

Sari, B. G. 2018. “Parametros Biologicos Da Producao de Tomateiro via Modelo Logistico.” PhD thesis, Santa Maria, RS, Brasil: Universidade Federal de Santa Maria.

Schenider, P. R., Schenider P. S. P., and C. A. M. Souza. 2009. *Analise de Regressao Aplicada a Engenharia Florestal*. FACOS, UFSM.

Scott, A. J., and M. J. Symons. 1971. “Clustering Methods Based on Likelihood Ratio Criteria.” *Biometrics* 27: 387–97. doi:10.2307/2529003.

Seber, G. A. F, and C. J. Wild. 2003. *Nonlinear Regression*. John Wiley & Sons.

Senoglu, B., and M. L. Tiku. 2001. “Analysis of Variance in Experimental Design with Nonnormal Error Distributions.” *Commun. Statist.-Theory Meth.* 30 (7): 1335–52. doi:doi.org/10.1081/STA-100104748.

Shukla, G. K. 1972. “Some Statistical Aspects of Partitioning Genotype-Environmental Components of Variability.” *Heredity* 29: 237–45. doi:10.1038/hdy.1972.87.

Smith, A.B., Cullis B.R., and R. Thompson. 2005. “The Analysis of Crop Cultivar Breeding and Evaluation Trials: An Overview of Current Mixed Model Approaches.” *J. Agric. Sci.* 143: 449–62. doi:10.2307/2334381.

Suzuki, R., and H. Shimodaira. 2006. “Pvclust: An R Package for Assessing the Uncertainty in Hierarchical Clustering.” *Bioinformatics* 22: 1540–2. doi:10.1093/bioinformatics/btl117.

Wricke, G. 1962. “On a Method of Understanding the Biological Diversity in Field Research.”

*Z. Pfl.-Zucht* 47: 92–146.

Wright, S. 1921. “Correlation and Causation.” *J. Agr. Res.* 20: 557–85.

\_\_\_\_\_. 1923. “The Theory of Path Coefficients a Reply to Niles’s Criticism.” *Genetics* 8: 239–55.

Yan, W., Kang M. S., Ma B., Woods S., and P. L. Cornelius. 2007. “GGE Biplot Vs. Ammi Analysis of Genotype-by-Environment Data.” *Crop Science* 47: 641–53. doi:10.2135/cropsci2006.06.0374.

Yates, F., and W. G. Cochran. 1938. “The Analysis of Groups of Experiments.” *The Journal of Agricultural Science* 28: 556–80. doi:10.1017/S0021859600050978.

## Índice remissivo

- .BMP, 41
- .EPS, 41
- .JPEG, 41
- .PDF, 41
- .SGV, 41
- .TIFF, 41
- agrupamento, 92, 215, 222, 228
- AMMI, 228, 238–240, 247, 252, 254, 255, 257
- amostras pareadas, 58
- análise multivariada, 208, 228
- ANOVA, 59, 105, 239, 241, 248
- anova(), 146
- aov, 59
- aov(), 29
- as.factor(), 18
- as.numeric(), 18
- atualização, 8
- backward, 149
- biplots, 210, 213, 214, 238, 245, 257
- BLUE, 248
- BLUP, 248, 250, 252, 254
- bmp(), 44
- Box-Cox, 83
- boxcox(), 83
- boxplot(), 35
- bty, 29
- causa e efeito, 191
- cbind(), 11
- citação de pacotes, 9
- class(), 17
- color, 27
- componentes de variância, 249
- conf.level(), 50
- console, 5
- corr.plot(), 181
- correlação, 148, 151, 176, 178, 186, 187
- correlação parcial, 188
- corrplot(), 184
- corrplot.mixed(), 183
- crd(), 62
- cursoR, 89
- curve(), 35
- CV, 46
- data.frame(), 12, 22
- DBC, 59, 78, 88, 178, 240
- dendrograma, 215, 217, 219, 220, 222, 223
- det(), 20
- determinante da matriz, 191
- dev.off(), 44
- diag(), 20
- DIC, 59, 62, 88, 240
- diretório, 5
- distdend(), 216, 219, 222
- distribuição normal, 48
- doses ótimas, 130
- DPI, 41
- eigen(), 20
- erro puro, 154
- estabilidade, 229, 235, 236, 245, 250, 257
- ExpDes, 62, 92
- exportar imagens, 41
- falta de ajuste, 72, 154, 155
- fat2.rbd(), 92, 107, 120
- fator de inflação de variância, 192, 199, 202
- fatores qualitativos, 63
- fatores quantitativos, 69
- for(), 21
- forward, 148
- funções, 16
- function(), 16, 46
- ge.stats(), 232
- ggplot(), 50, 76
- ginv(), 18, 144
- gráfico de contorno, 136
- gráfico de superfície, 132
- graficos(), 73
- hist(), 32
- homocedasticidade, 84, 88
- importar planilhas, 25
- imputs, 5
- interação, 88, 92, 116, 119, 121, 126, 228, 255

intervalo de confiança, 50, 76  
 jpeg(), 44  
 kmeans(), 228  
 ks.test, 47  
 library, 5  
 lines(), 32  
 lista, 14  
 lm(), 59, 145  
 locator(), 86  
 log-verossimilhança, 86  
 loop(), 21  
 loops, 21  
 LRT, 249  
 lwd, 27  
 mínimos quadrados, 143, 162, 168  
 Máxima eficiência econômica, 107  
 Máxima eficiência técnica, 106  
 média, 45  
 main(), 27  
 matrix(), 11, 22  
 matrizes, 10  
 mean(), 45  
 median(), 45  
 mediana, 45  
 multicolinearidade, 144, 145, 190–192, 195, 197, 199, 202, 203  
 número de condição, 191, 199  
 names(), 20, 39  
 nls(), 162, 163  
 normalidade, 48, 83, 88  
 operações matemáticas, 18  
 outputs, 5  
 pacotes, 9  
 par, 27  
 parâmetros, 59, 141, 143–146, 162, 163, 168, 170, 191  
 parâmetros genéticos, 248, 249  
 parcelas subdivididas, 139  
 partial.corr(), 188  
 pass(), 104  
 path.coeff(), 195, 199, 202  
 path.diagram(), 204  
 pca(), 209  
 pch, 27  
 pdf(), 42, 44  
 plot, 172  
 plot(), 27, 29, 32, 35  
 plot.fatcurves(), 105  
 plot.fatmeans, 120, 124  
 plot.fatmeans(), 103  
 plot.supresp, 131  
 plotres, 111  
 plotres(), 65, 102  
 png(), 44  
 poder do teste, 83  
 point(), 29  
 polinômio, 72  
 pontos influentes, 158  
 postdiscritive sucess, 239  
 predict.AMMI(), 247  
 predictive sucess, 240  
 pressuposições, 83  
 pressupostos, 45, 59, 63, 83, 84, 88, 144, 165  
 qqnorm(), 32  
 qualitativo, 88, 114  
 quantitativo, 88, 114  
 rbd, 80  
 rbind(), 11  
 read.table, 24  
 regressão, 105, 113  
 regressão múltipla, 141  
 regressão simple, 141  
 REML, 248  
 require, 5  
 resíduos, 25, 29, 32, 65, 78, 83, 102, 143, 147, 156, 157, 164  
 rms.curv(), 168  
 scan(), 24  
 Scott-Knott, 68  
 sd(), 46  
 seleção de variáveis, 192, 202, 207, 219  
 Shapiro-Wilk, 80  
 shapiro.test, 47  
 sink(), 41  
 sistema de equações normais, 143, 144, 162, 191

solve(), 18  
start, 162  
stat.desc(), 46  
stepAIC, 152  
stepwise, 151, 192, 202  
subset, 27  
subset(), 13  
summary, 146  
summary(), 163  
summary.path.coeff(), 193, 197  
supresp(), 126, 131

t(), 18  
t.test(), 50  
tamanho de amostra, 187  
testes de hipótese, 54  
tiff(), 44  
transformação de dados, 83  
Tukey, 68, 120  
type, 27

validation.AMMIF(), 240  
var, 46  
variáveis dummy, 170, 175  
vetores, 9

WAAS.AMMI(), 239  
WAASB(), 206, 248  
WAASBratio(), 258  
Welch-Satterthwaite, 56  
write.table(), 39  
write.xlsx(), 41

xlab, 27  
ylab, 27