

Uso do sistema R para análise de dados

Jefferson Vieira José, Dhonatan Diego Pessi, Timóteo Herculino da Silva Barros e André Pereira

2020-05-13

Contents

1	Pré requisitos	7
2	R Básico	9
2.1	Expressões	9
2.2	Valores Booleanos	10
2.3	Variáveis	10
2.4	Funções	11
2.5	Ajuda	11
2.6	Referência	14
3	Estruturas de Dados	15
3.1	Vetor	15
3.2	Matrizes	26
3.3	Fatores	36
3.4	Array	36
3.5	Data.frame	40
3.6	Lista	44
3.7	Referência	46
4	Entrada de dados	47
4.1	Onde os dados devem estar?	47
4.2	Entrando com dados	48
4.3	Salvar objetos de dados	55
4.4	Referência	56

5 Criando Gráficos com o R	57
5.1 Exemplos de gráficos com o R	58
5.2 Entrada de dados	84
5.3 Usando a função <code>plot()</code>	85
5.4 Histogramas	100
5.5 Gráficos de Barras	102
5.6 Boxplots	103
5.7 Cores	106
5.8 Interagindo com a Janela gráfica	107
5.9 Texto e tamanho do símbolo	108
5.10 Visualizar vários gráficos	108
5.11 Salvando gráficos	111
6 Gráficos com ggplot2	113
6.1 Personalizando os gráficos	122
6.2 Exemplos	137
6.3 Referência	150
7 Testes Estatísticos	151
7.1 Teste t de Student	151
7.2 Teste de variância	159
7.3 Teste para a normalidade - <code>shapiro.test()</code>	160
7.4 Teste U de Mann-Whitney	162
7.5 Covariância e Correlação	162
7.6 Outros testes	165
8 Analise de variância (ANOVA)	169
8.1 Delineamento inteiramente casualizado	170
8.2 Delineamento em bloco casualizado	185
8.3 Quadrado Latino	206
8.4 Regressão Linear simples	228
8.5 Regressão Linear Multipla	238
8.6 Regressão não linear	243

CONTENTS	5
9 Multivariada	249
10 Dados climáticos	255
10.1 Precipitação pluvial	255
10.2 Evapotranspiração	264
10.3 Dados temporais no R	291
10.4 Datas no R	337
10.5 Teste Mann-Kendall para Tendência	338
11 Sensoriamento remoto	351
11.1 Imagens de RPAs	351
11.2 Imagens de satélites	352
11.3 Curvas de nível e modelo 3D a partir do Modelo Digital de Elevação	356
11.4 LIDAR	357
11.5 Modelo Digital de Elevação MDE	359

Chapter 1

Pré requisitos

Material em construção.

Este material, em forma de notas de aula, foi escrito para a disciplina do Mestrado em Engenharia Agrícola, intitulado Uso do sistema R para análise de dados, no primeiro semestre de 2018. Estas notas de aulas é uma coletânea de apostilas, livros, sites, forum e cursos voltando ao sistema R. Foi utilizado desses materiais sua estrutura didática e rotinas que foram adaptados para o perfil da disciplina. O material consultado encontra-se referenciado no final de cada capítulo.

Chapter 2

R Básico

Este primeio capítulo foi baseado no curso on-line denominada *Code School Try R* e *Datacamp*. Foram realizadas modificações utilizando-se de outros materiais que se encontram referenciado no final desse capítulo.

Primeiramente iremos abordar as expressões básicas do R. Começaremos com comandos simples, como por exemplo, os comandos **números**, **strings** e valores **true/false**. Em seguida mostraremos como armazenar esses valores em variáveis e como transmitir as funções. Como obter ajuda sobre as funções e no final vamos carregar um arquivo.

2.1 Expressões

Vamos tentar algumas funções matemáticas simples. Digite o comando abaixo e aperte enter:

```
2+8
```

```
## [1] 10
```

Note que é impresso o resultado 10.

Digite a frase “Engenharia Agrícola”:

```
"Engenharia Agrícola"
```

```
## [1] "Engenharia Agrícola"
```

Agora tente multiplicar 6×5 ($*$ é o operador de multiplicação):

```
6*5
```

```
## [1] 30
```

2.2 Valores Booleanos

Algumas expressões retornam um “valor lógico”: TRUE ou FALSE e/ou “booleanos”. Vamos tentar digitar uma expressões que nos dê um valor lógico:

```
7<12
```

```
## [1] TRUE
```

E outro valor lógico (sinal duplo de igualdade):

```
6+5==10
```

```
## [1] FALSE
```

T e **F** são taquigrafia para TRUE e FALSE. Tente isso:

```
F==FALSE
```

```
## [1] TRUE
```

2.3 Variáveis

Você pode armazenar valores em uma variável para usar mais tarde. Digite **x** **<-** **28** para armazenar um valor em **x**:

```
x<-28
```

Tente dividir **x** por **4** (**/** é o operador da divisão):

```
x/4
```

```
## [1] 7
```

Você pode reatribuir qualquer valor a uma variável a qualquer momento. Tente atribuir “Engenharia Agrícola” em **x**:

```
x <- "Engenharia Agrícola"
```

Tente imprimir o valor atual de x:

```
x
```

```
## [1] "Engenharia Agrícola"
```

2.4 Funções

Você pode chamar uma **função** digitando seu nome, seguido de um ou mais argumentos para essa função entre parênteses.

Vamos tentar usar a função **sum()** para adicionar alguns números. Entrar com o comando:

```
sum (2, 4, 6)
```

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

Alguns argumentos têm nomes. Por exemplo, para repetir um valor 3 vezes você chamaria a função **rep** e forneceria seu argumento **times**:

```
rep("Engenharia Agrícola", times=3)
```

```
## [1] "Engenharia Agrícola" "Engenharia Agrícola" "Engenharia Agrícola"
```

Tente chamar a função **sqrt** para obter a raiz quadrada de 16:

```
sqrt(16)
```

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

2.5 Ajuda

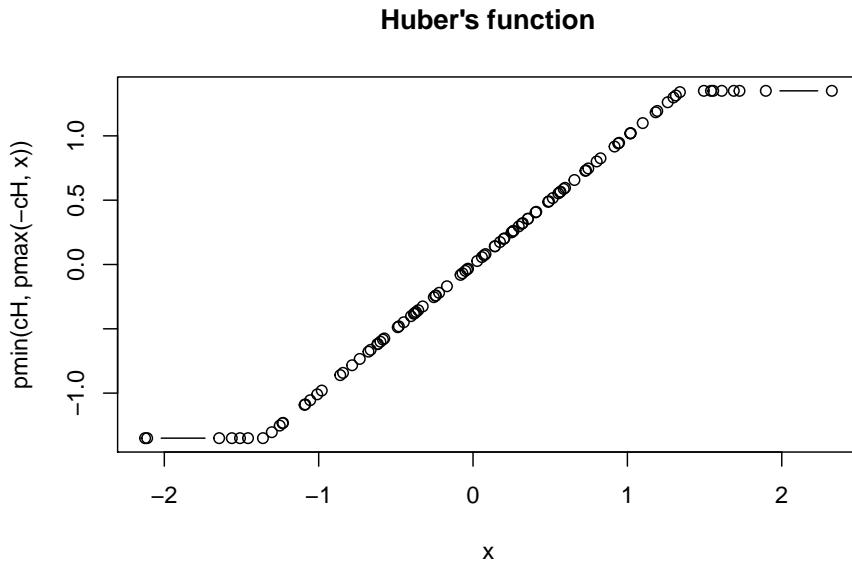
A função **help()** fornece ajuda para a função desejada. Tente exibir ajuda para a função **mean**:

```
help (mean)
```

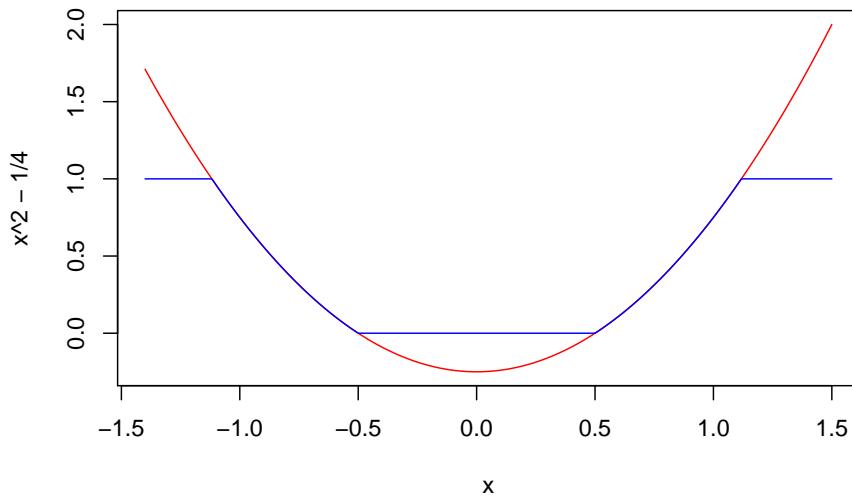
A função `example ()` traz exemplos de usos. Tente exibir exemplos para a função `min`:

```
example(min)
```

```
##  
## min> require(stats); require(graphics)  
##  
## min> min(5:1, pi) #-> one number  
## [1] 1  
##  
## min> pmin(5:1, pi) #-> 5 numbers  
## [1] 3.141593 3.141593 3.000000 2.000000 1.000000  
##  
## min> x <- sort(rnorm(100)); cH <- 1.35  
##  
## min> pmin(cH, quantile(x)) # no names  
## [1] -2.1246694 -0.5878653 0.1108181 0.7289706 1.3500000  
##  
## min> pmin(quantile(x), cH) # has names  
##          0%         25%         50%         75%        100%  
## -2.1246694 -0.5878653  0.1108181  0.7289706  1.3500000  
##  
## min> plot(x, pmin(cH, pmax(-cH, x)), type = "b", main = "Huber's function")
```



```
##  
## min> cut01 <- function(x) pmax(pmin(x, 1), 0)  
##  
## min> curve(      x^2 - 1/4, -1.4, 1.5, col = 2)
```



```
##  
## min> curve(cut01(x^2 - 1/4), col = "blue", add = TRUE, n = 500)  
##  
## min> ## pmax(), pmin() preserve attributes of *first* argument  
## min> D <- diag(x = (3:1)/4) ; n0 <- numeric()  
##  
## min> stopifnot(identical(D, cut01(D) ),  
## min+     identical(n0, cut01(n0)),  
## min+     identical(n0, cut01(NULL)),  
## min+     identical(n0, pmax(3:1, n0, 2)),  
## min+     identical(n0, pmax(n0, 4)))
```

2.6 Referência

MELO, M. P.; PETERNELI, L. A. **Conhecendo o R: Um visão mais que estatística**. Viçosa, MG: UFV, 2013. 222p.

Prof. Paulo Justiniando Ribeiro ><http://www.leg.ufpr.br/~paulojus/><

Prof. Adriano Azevedo Filho ><http://rpubs.com/adriano/esalq2012inicial><

Prof. Fernando de Pol Mayer ><https://fernandomayer.github.io/ce083-2016-2/><

Site Interativo Datacamp ><https://www.datacamp.com/><

Chapter 3

Estruturas de Dados

Este segundo capítulo foi baseado no curso on-line *Code School Try R* e no livro **Conhecendo o R: Um visão mais que estatística**, modificações foram realizadas utilizando outros materiais que se encontram referenciados no final desse capítulo.

3.1 Vetor

Um vetor é simplesmente uma lista de valores. A maneira mais simples de usar um vetor é usando o comando `c()`, que concatena elementos num mesmo objeto. Exemplo:

```
x<- c(2,3,5,7,11)  
x
```

```
## [1] 2 3 5 7 11
```

Os argumentos de `c()` podem ser tanto elementos únicos quanto outros objetos. Adicione três números no **vetor x**:

```
y<- c(x,13,17,19)  
y
```

```
## [1] 2 3 5 7 11 13 17 19
```

3.1.1 Vetores de Sequência

Se você precisar de um vetor com uma sequência de números, você pode criá-lo com a notação `start:end`. Vamos fazer um vetor com valores de 1 a 7:

```
1:7
```

```
## [1] 1 2 3 4 5 6 7
```

Uma maneira mais versátil de fazer sequências é chamar a função `seq`. Vamos fazer o mesmo com `seq()`:

```
seq(1:7)
```

```
## [1] 1 2 3 4 5 6 7
```

A função `seq` também permite que você use incrementos diferentes de 1. Experimente com as etapas de 0.5:

```
seq(1,7,0.5)
```

```
## [1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0
```

```
seq(7,1,-0.5)
```

```
## [1] 7.0 6.5 6.0 5.5 5.0 4.5 4.0 3.5 3.0 2.5 2.0 1.5 1.0
```

Todo objeto possui atributos intrínsecos, como tipo e tamanho. Com relação ao tipo, ele pode ser: **numérico**, **caractere**, **complexo** e **lógico**. Existem outros tipos, como por exemplo, funções ou expressões, porém esses não representam dados. As funções `mode()` e `length()` mostram o tipo e tamanho de um objeto, respectivamente:

```
z<-c(1,3,5,7,11)
mode (z)
```

```
## [1] "numeric"
```

```
length(z)
```

```
## [1] 5
```

```
a <- "Angela"
b<-TRUE;
c<-8i #objetos com tipos diferentes
mode(a);

## [1] "character"

mode(b);

## [1] "logical"

mode(c) #exibe os atributos "tipo" dos objetos

## [1] "complex"
```

Se o vetor é muito longo e não “cabe” em uma linha, o R vai usar as linhas seguintes para continuar imprimindo o vetor:

```
longo<-100:50 #sequência decrescente de 100 a 50
longo #exibe o conteúdo do objeto
```

```
## [1] 100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82
## [20] 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63
## [39] 62 61 60 59 58 57 56 55 54 53 52 51 50
```

Os números entre os colchetes não fazem parte do objeto e indicam a posição do vetor naquele ponto. Pode-se ver que [1] indica que o primeiro elemento do vetor estão naquela linha, e o [17] indica que a linha seguinte começa pelo décimo sétimo elemento do vetor e assim por diante.

Você pode recuperar um valor individual dentro de um vetor fornecendo seu índice numérico entre colchetes. Tente obter o valor 18:

```
longo[18]
```

```
## [1] 83
```

Muitas línguagens de programação iniciam índices de matriz em 0, mas os índices vetoriais de R começam em 1. Obtenha o primeiro valor digitando:

```
longo[1]
```

```
## [1] 100
```

Você pode atribuir novos valores dentro de um vetor existente. Tente mudar o terceiro valor **28**:

```
longo[3] <- 28
```

Se você adicionar novos valores no final, o vetor aumentará para acomodá-los. Vamos adicionar um valor no final:

```
longo[101] <- 83
```

Você pode usar um vetor entre os colchetes para acessar vários valores. Tente obter a primeira e a terceira palavra:

```
longo[c(1,3)]
```

```
## [1] 100 28
```

Isso significa que você pode recuperar intervalos de valores. Obter a segunda e a quarta palavra:

```
longo[2:4]
```

```
## [1] 99 28 97
```

Você também pode definir intervalos de valores. Apenas fornecendo os valores em um vetor. Adicione valores 5 a 7:

```
longo[5:7] <- c(42,52,75)
```

Agora tente acessar o oitavo valor do vetor:

```
longo[8]
```

```
## [1] 93
```

3.1.2 Nomes de vetores

Para esse desafio, criaremos um vetor de 3 itens, e na sequência iremos armazená-lo na variável solo. Você pode atribuir nomes aos elementos de um vetor passando um segundo vetor preenchido com os nomes com a função `names()`, assim:

```
solo <- 1:3  
names(solo) <- c("Argila", "Areia", "Silte")  
solo
```

```
## Argila  Areia  Silte  
##      1      2      3
```

Agora defina o valor atual para o *silte* para um valor diferente usando o nome em vez da posição:

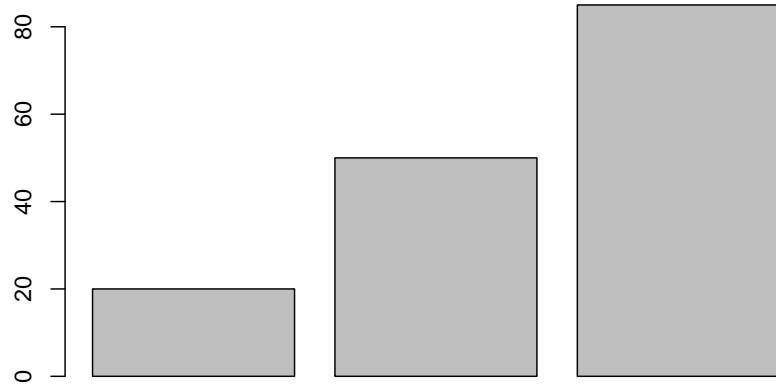
```
solo["Silte"] <- 20
```

3.1.3 Plotando um vetor

A função `barplot()` desenha um gráfico de barras com os valores de um vetor. Vamos criar um novo vetor e armazená-lo na variável chuva.

Agora tente passar o vetor para a função `barplot`:

```
chuva <- c(20, 50, 85)  
barplot(chuva)
```

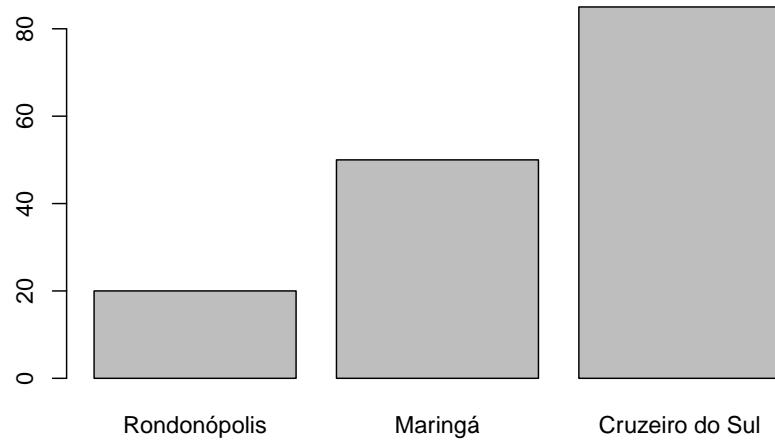


Se você atribuir nomes aos valores do vetor, o R usará esses nomes como rótulos no gráfico da barra. Vamos usar a função `names()` novamente:

```
names(chuva) <- c("Rondonópolis", "Maringá", "Cruzeiro do Sul")
```

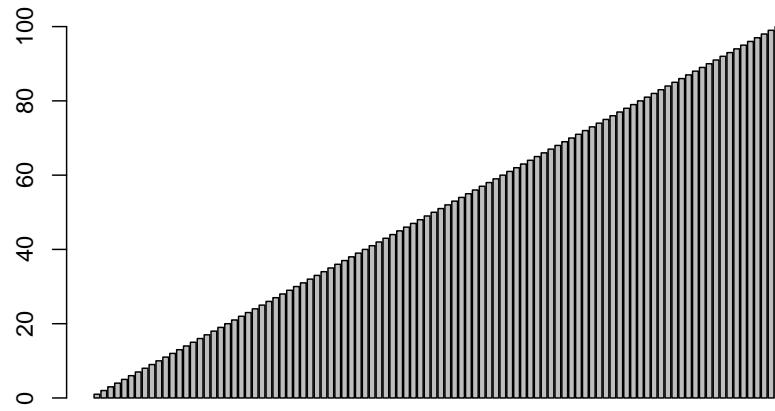
Se você digitar `barplot(chuva)` com o vetor novamente, você verá os rótulos:

```
barplot(chuva)
```



Tente chamar `barplot` em um vetor de números inteiros que variam de 1 a 100:

```
barplot(1:100)
```



3.1.4 Operações matemáticas

A maioria das operações aritméticas funcionam tão bem em vetores quanto em valores únicos. Vamos fazer outro vetor de exemplo para você trabalhar e armazená-lo na variável **a**.

Se você adicionar um escalar (um único valor) a um vetor, o escalar será adicionado a cada valor no vetor, retornando um novo vetor com os resultados. Tente adicionar 1 a cada elemento em nosso vetor:

```
a <- c(1, 2, 3)
a + 1
```

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

O mesmo se aplica na divisão, multiplicação ou qualquer outra aritmética básica. Tente dividir nosso vetor por 2:

```
a / 2
```

```
## [1] 0.5 1.0 1.5
```

Tente multiplicar nosso vetor por 2:

```
a*2
```

```
## [1] 2 4 6
```

Se você adicionar dois vetores, o R irá tirar cada valor de cada vetor e adicioná-los. Vamos fazer um segundo vetor para você experimentar e armazená-lo na variável **b**.

Tente adicioná-lo no vetor **a**:

```
b <- c(4,5,6)
a+b
```

```
## [1] 5 7 9
```

Agora tente subtrair **b** de **a**:

```
a-b
```

```
## [1] -3 -3 -3
```

Você também pode tirar dois vetores e comparar cada item. Veja quais valores nos vetores são iguais aos de um segundo vetor:

```
a == c(1, 99, 3)  
  
## [1] TRUE FALSE TRUE
```

Observe que o R não testou se os vetores inteiros eram iguais, mas verificou cada valor no vetor a contar o valor no mesmo índice no nosso novo vetor.

Verifique se cada valor nos vetores são menores que o valor correspondente em outro vetor:

```
a < c(1, 99, 3)  
  
## [1] FALSE TRUE FALSE
```

Funções que normalmente funcionam com escalares também podem operar em cada elemento de um vetor. Tente obter o seno de cada valor em nosso vetor:

```
sin(a)  
  
## [1] 0.8414710 0.9092974 0.1411200
```

Agora tente obter as raízes quadradas com a função `sqrt`:

```
sqrt(a)  
  
## [1] 1.000000 1.414214 1.732051
```

3.1.5 Parcelas de dispersão

A função `plot` leva dois vetores, um para valores X e um para valores Y, e desenha um gráfico com eles.

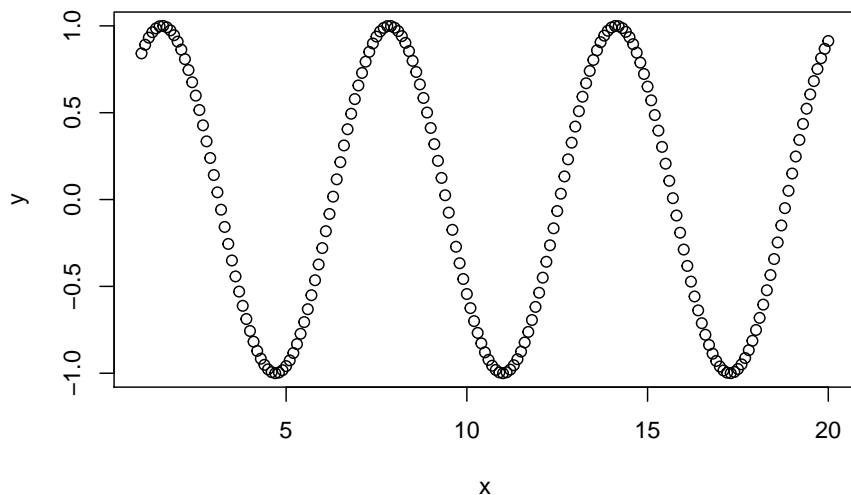
Vamos desenhar um gráfico que mostre a relação de números e seus senos.

Primeiro, precisaremos de alguns dados de amostra. Criaremos um vetor com alguns valores fracionários entre 0 e 20, e iremos armazená-lo na variável `x`. E na variável `y` um segundo vetor com os senos de `x`:

```
x <- seq(1, 20, 0.1)
y<-sin(x)
```

Em seguida basta chamar a função `plot` com seus dois vetores:

```
plot(x, y)
```



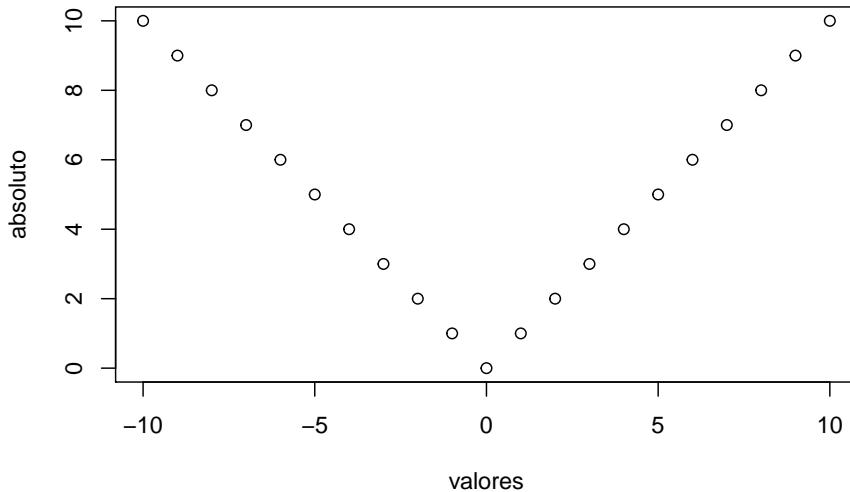
Observa-se sobre o gráfico que os valores do primeiro argumento (`x`) são usados para o eixo horizontal, e os valores do segundo (`y`) para o vertical.

Vamos criar um vetor com alguns valores negativos e positivos e armazená-lo na variável **valores**.

Também criaremos um segundo vetor com os valores absolutos do primeiro e armazená-lo na variável **absoluto**.

Tente traçar os vetores, com os **valores** absolutos no eixo horizontal e no eixo vertical:

```
valores <- -10:10
absoluto<- abs(valores)
plot(valores, absoluto)
```



3.1.6 Valores Faltantes

Às vezes um determinado valor não está disponível ao trabalhar com dados de amostra. Mas não é uma boa idéia apenas tirar esses valores. O R tem um valor que indica explicitamente uma amostra que não estava disponível: **NA**. Muitas funções que funcionam com vetores tratam esse valor especialmente.

Vamos criar um vetor com uma amostra ausente e armazená-lo na variável **a**.

Tente obter a soma de seus valores e veja qual é o resultado:

```
a <- c(1, 3, NA, 7, 9)
sum(a)
```

```
## [1] NA
```

A soma é considerada “*não disponível*” por padrão porque um dos valores do vetor foi **NA**.

Lembre-se desse comando para mostrar ajuda para uma função. Apresente a ajuda para a função **sum**:

```
help(sum)
```

Como você vê na documentação, `sum` pode tomar um argumento opcional `na.rm`. É configurado **FALSE** por padrão, mas se você configurá-lo com **TRUE**, todos os argumentos **NA** serão removidos do vetor antes do cálculo a ser executado.

Tente rondar `sum` novamente, com o `na.rm` conjunto para **TRUE**:

```
sum(a, na.rm = T)
```

```
## [1] 20
```

3.2 Matrizes

Há varias formas de criar uma matriz. O comando `matrix()` recebe um vetor como argumento e o transforma em uma matrix de acordo com as dimensões. Vamos fazer uma matriz de 3 linhas de altura por 4 colunas de largura com todos os seus campos definidos em 0.

```
matrix(0,3,4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]     0     0     0     0
## [2,]     0     0     0     0
## [3,]     0     0     0     0
```

Você também pode usar um vetor para inicializar o valor de uma matriz. Para preencher uma matriz de 3x4, você precisará de um vetor de 12 itens.

```
a <- (1:12)
```

```
print (a)
```

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

Agora chame a matrix com o vetor, o número de linhas e o número de colunas:

```
matrix (a,# chama o vetor
       3,# linha
       4) #coluna
```

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

Você também pode usar um vetor para inicializar o valor de uma matriz. Para preencher uma matriz 3x4, você precisará de um vetor de 12 itens:

```
a <- 1:12
a
```

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

Agora chame a **matrix** com o vetor, o número de linhas e o número de colunas:

```
matrix (a, 3, 4)
```

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

3.2.1 Outras formas

```
matrix (a, 3)
```

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

Note que as matrizes são preenchidas ao longo das colunas. Para que a matriz seja preenchida por linhas deve-se alterar o argumento **byrow**, que por padrão está definido como **FALSE**, passe para **TRUE**:

```
matrix(a, 3, byrow=T)
```

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

Os valores do vetor são copiados um por um para a nova matriz. Você também pode reformular o próprio **vetor** em uma **matriz**. Crie um vetor de 8 itens:

```
foliar <- 1:8
```

A função `dim` define as dimensões para uma matriz. Ele aceita um vetor com o número de linhas e o número de colunas a serem atribuídas. Atribua novas dimensões para `foliar` passando um vetor especificando 2 linhas e 4 colunas (`c(2, 4)`):

```
dim(foliar) <- c(2,4)
```

O vetor não é mais unidimensional. Foi convertido no local para uma matriz. Agora, use a função `matrix` para criar uma matriz **5x5**, com seus campos inicializados para qualquer valor que você desejar:

```
matrix (2,5,5)
```

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

3.2.2 Acesso a Matriz

Obter valores de matrizes não é diferente de vetores, você só precisa fornecer dois índices em vez de um. Abra a matriz `foliar`:

```
print (foliar)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]     1     3     5     7
## [2,]     2     4     6     8
```

Tente obter o valor da segunda linha na terceira coluna da matriz `foliar`:

```
foliar[2,3]
```

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

O valor da primeira linha da quarta coluna:

```
foliar[1,4]
```

```
## [1] 7
```

Você pode obter uma linha inteira da matriz omitindo o índice da coluna (mas mantenha a vírgula). Tente recuperar a segunda linha:

```
foliar[2,]
```

```
## [1] 2 4 6 8
```

Para obter uma coluna inteira, omita o índice da linha. Recupere a quarta coluna:

```
foliar[,4]
```

```
## [1] 7 8
```

Você pode ler várias linhas ou colunas, fornecendo um vetor ou sequência com seus índices. Tente recuperar as colunas de 2 a 4:

```
foliar[,2:4]
```

```
##      [,1] [,2] [,3]
## [1,]     3     5     7
## [2,]     4     6     8
```

O comando `summary` pode ser usado para obter informações da matriz

```
summary(foliar)
```

	V1	V2	V3	V4
## Min.	:1.00	Min. :3.00	Min. :5.00	Min. :7.00
## 1st Qu.	:1.25	1st Qu.:3.25	1st Qu.:5.25	1st Qu.:7.25
## Median	:1.50	Median :3.50	Median :5.50	Median :7.50
## Mean	:1.50	Mean :3.50	Mean :5.50	Mean :7.50
## 3rd Qu.	:1.75	3rd Qu.:3.75	3rd Qu.:5.75	3rd Qu.:7.75
## Max.	:2.00	Max. :4.00	Max. :6.00	Max. :8.00

Se desejar um resumo de todos os elementos da matriz, basta transformá-la em um vetor:

```
summary(as.vector(foliar))
```

```
##      Min. 1st Qu. Median     Mean 3rd Qu.     Max.
##      1.00    2.75    4.50    4.50    6.25    8.00
```

3.2.3 Visualizações em dados matriciais

Com um mapa de elevação. Tudo fica a 1 metro acima do nível do mar. Vamos criar uma matriz de 10 por 10 com todos os seus valores inicializados para 1:

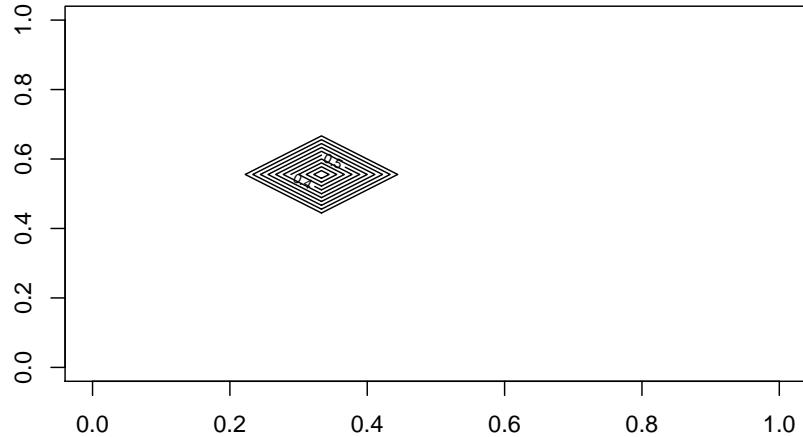
```
elevacao <- matrix (1,10,10)
```

Na quarta linha, sexta coluna, defina a elevação para 0:

```
elevacao [4, 6] <- 0
```

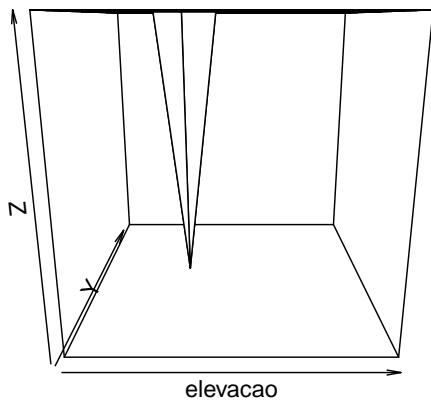
Mapa de contorno dos valores passando a matriz para a função `contour`:

```
contour(elevacao)
```



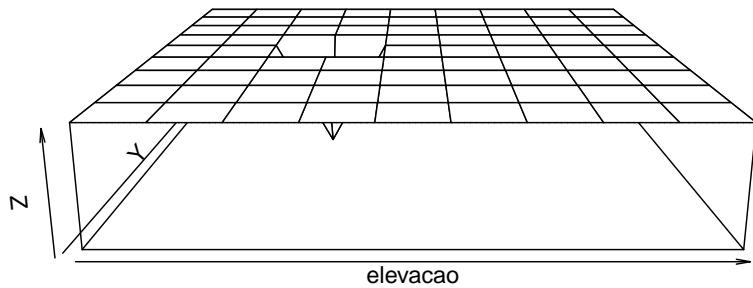
Criar um gráfico em perspectiva 3D com a função `persp`:

```
persp (elevacao)
```



Podemos consertar isso especificando nosso próprio valor para o parâmetro **expand**:

```
persp (elevacao, expand =0.2)
```



O R inclui alguns conjuntos de dados de amostra. Um deles é o *volcano* um mapa 3D de um vulcão adormecido da Nova Zelândia.

É simplesmente uma matriz de 87x61 com valores de elevão, mas mostra o poder das visualizações de matriz do R. Criar um mapa de calor:

```
contour(volcano)
```

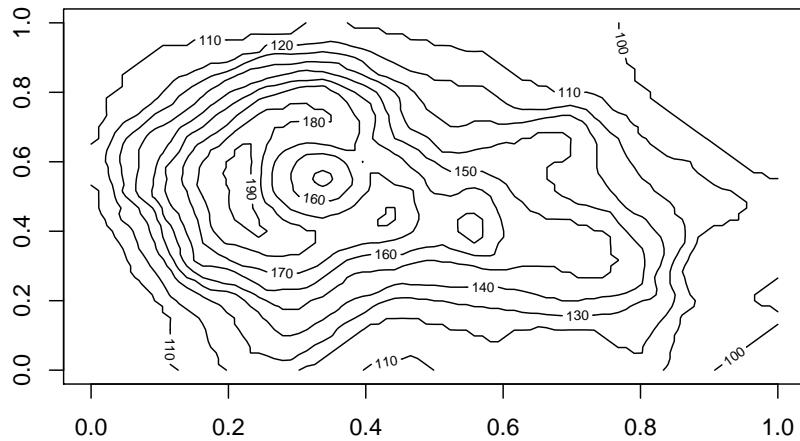
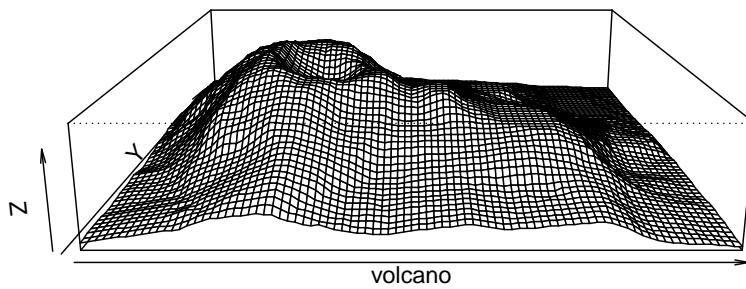


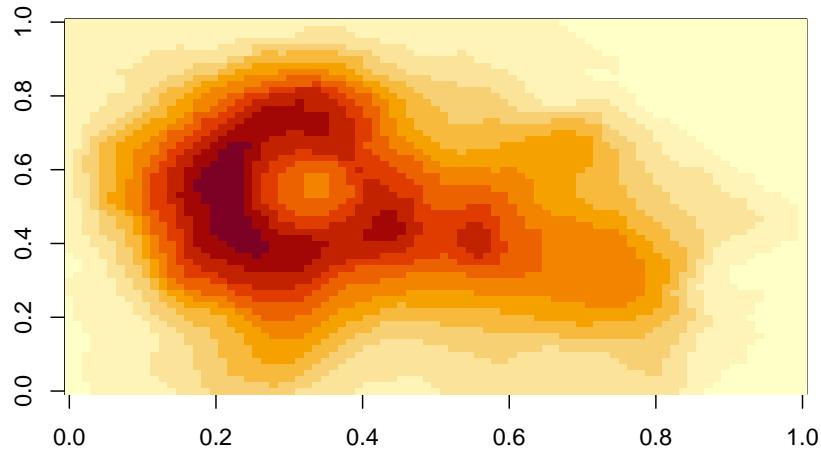
Gráfico em perspectiva:

```
persp(volcano, expand=0.2)
```



A função `image` cria um mapa de calor:

```
image(volcano)
```



3.2.4 Mais informações sobre construções de Matrizes

Há outros comandos que podem ser usados para construir matrizes como `cbind()` e `rbind ()`. Esses comandos concatenam colunas ou linhas, respectivamente na matriz (ou vetor):

```
a <- matrix (10:1,ncol=2) #construir uma matriz qualquer
a
```

```
##      [,1] [,2]
## [1,]    10    5
## [2,]     9    4
## [3,]     8    3
## [4,]     7    2
## [5,]     6    1
```

```
b <- cbind (a,1:5) #adicionar uma terceira coluna
b
```

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

```
## [1,] 10 5 1
## [2,] 9 4 2
## [3,] 8 3 3
## [4,] 7 2 4
## [5,] 6 1 5
```

```
c<- rbind(b,c(28,28,28))
c
```

```
## [,1] [,2] [,3]
## [1,] 10 5 1
## [2,] 9 4 2
## [3,] 8 3 3
## [4,] 7 2 4
## [5,] 6 1 5
## [6,] 28 28 28
```

Opcionalmente matrizes podem ter nomes associados às linhas e colunas (“rownames” e “colnames”). Cada um destes componentes da matrix é um vetor de nomes.

```
m1 <- matrix(1:12, ncol = 3)

dimnames(m1) <- list(c("L1", "L2", "L3", "L4"), c("C1", "C2", "C3"))
dimnames(m1)
```

```
## [[1]]
## [1] "L1" "L2" "L3" "L4"
##
## [[2]]
## [1] "C1" "C2" "C3"
```

Matrizes são muitas vezes utilizadas para armazenar frequências de cruzamentos entre variáveis. Desta forma é comum surgir a necessidade de obter os totais marginais, isto é a soma dos elementos das linhas e/ou colunas das matrizes, o que pode ser diretamente obtido com `margin.table()`:

```
margin.table(m1, margin = 1)
```

```
## L1 L2 L3 L4
## 15 18 21 24
```

```
margin.table(m1, margin = 2)

## C1 C2 C3
## 10 26 42

apply(m1, 2, median)

##   C1   C2   C3
## 2.5 6.5 10.5
```

3.3 Fatores

Os fatores são vetores em que os elementos pertencem a uma ou mais categorias temáticas. Por exemplo, ao criar um vetor de indicadores de “**tratamentos**” em uma análise de experimentos, devemos declarar este vetor como um “**fator**”. Pode criar um fator usando o comando **factor()**, ou o comando **gl**.

```
factor(rep(paste("T", 1:3, sep = ""), c(4, 4, 3)))

## [1] T1 T1 T1 T1 T2 T2 T2 T2 T3 T3 T3 T3
## Levels: T1 T2 T3

peso <- c(134.8, 139.7, 147.6, 132.3, 161.7, 157.7, 150.3, 144.7,
        160.7, 172.7, 163.4, 161.3, 169.8, 168.2, 160.7, 161.0,
        165.7, 160.0, 158.2, 151.0, 171.8, 157.3, 150.4, 160.4,
        154.5, 160.4, 148.8, 154.0)
trat <- rep(seq(0,300,50), each=4) #?each
dados <- data.frame(peso, trat=as.factor(trat))
```

3.4 Array

O conceito de array generaliza a idéia de matrix. Enquanto em uma matrix os elementos são organizados em duas dimensões (linhas e colunas), em um array os elementos podem ser organizados em um número arbitrário de dimensões. No R um array é definido utilizando a função **array()**:

```
ar1 <- array(1:24, dim = c(3, 4, 2))
ar1
```

```

## , , 1
##
##      [,1] [,2] [,3] [,4]
## [1,]     1     4     7    10
## [2,]     2     5     8    11
## [3,]     3     6     9    12
##
## , , 2
##
##      [,1] [,2] [,3] [,4]
## [1,]    13    16    19    22
## [2,]    14    17    20    23
## [3,]    15    18    21    24

```

Veja agora um exemplo de dados já incluído no R no formato de array. Para “carregar” e visualizar os dados digite:

```

data(Titanic)
Titanic

## , , Age = Child, Survived = No
##
##      Sex
## Class Male Female
## 1st     0      0
## 2nd     0      0
## 3rd    35     17
## Crew    0      0
##
## , , Age = Adult, Survived = No
##
##      Sex
## Class Male Female
## 1st   118      4
## 2nd   154     13
## 3rd   387     89
## Crew  670      3
##
## , , Age = Child, Survived = Yes
##
##      Sex
## Class Male Female
## 1st     5      1
## 2nd    11     13
## 3rd    13     14

```

```
##   Crew      0      0
##
## , , Age = Adult, Survived = Yes
##
##       Sex
## Class Male Female
##  1st    57    140
##  2nd    14     80
##  3rd    75     76
## Crew   192     20
```

Para obter maiores informações sobre estes dados digite: `help(Titanic)`

Agora vamos responder às seguintes perguntas, mostrando os comandos do R utilizados sobre o array de dados.

1. Quantas pessoas haviam no total?

```
sum(Titanic)
```

```
## [1] 2201
```

2. Quantas pessoas haviam na tripulação (crew)?

```
sum(Titanic[4, , ])
```

```
## [1] 885
```

3. Quantas pessoas sobreviveram e quantas morreram?

```
apply(Titanic, 4, sum)
```

```
##   No   Yes
## 1490  711
```

4. Quais as proporções de sobreviventes entre homens e mulheres?

```
margin.table(Titanic, margin = 1)
```

```
## Class
##  1st  2nd  3rd Crew
## 325 285  706  885
```

```
margin.table(Titanic, margin = 2)

## Sex
##   Male Female
##   1731    470
```

```
margin.table(Titanic, margin = 3)

## Age
## Child Adult
## 109 2092
```

```
margin.table(Titanic, margin = 4)

## Survived
##   No Yes
## 1490 711
```

Esta função admite ainda índices múltiplos que permitem outros resumos da tabela de dados. Foi demonstrado como obter o total de sobreviventes e não sobreviventes, separados por sexo e depois as porcentagens de sobreviventes para cada sexo. Exemplo:

```
margin.table(Titanic, margin = c(2, 4))

##           Survived
## Sex       No   Yes
##   Male   1364 367
##   Female 126 344

prop.table(margin.table(Titanic, margin = c(2, 4)), margin = 1)

##           Survived
## Sex       No      Yes
##   Male  0.7879838 0.2120162
##   Female 0.2680851 0.7319149

prop.table(margin.table(Titanic, margin = c(2, 1)), margin = 1)

##           Class
## Sex          1st     2nd     3rd      Crew
##   Male  0.10398614 0.10340843 0.29462738 0.49797805
##   Female 0.30851064 0.22553191 0.41702128 0.04893617
```

3.5 Data.frame

Os `data.frames` são muito semelhantes quando comparados às matrizes, pois têm linhas e colunas, portanto duas dimensões. Entretanto, diferentemente das matrizes, colunas diferentes podem armazenar elementos de tipos diferentes. Por exemplo, a primeira coluna pode ser numérica, enquanto a segunda constituída de caracteres. Cada coluna precisa ter o mesmo tamanho. Criar o vetor nomes:

```
nome <- c("Melissa José",
        "Jennifer Linhares",
        "Gedilene Ponciano",
        "Edinar da Silva",
        "Osmar Emidio",
        "Jeeziel Vieira")
```

Criar vetor idade:

```
idade <- c(17,18,16,15,15,18)
```

Criar vetor sexo (categoria=fator):

```
sexo <- factor(c("F", "F", "F", "F", "M", "M"))
```

Criar vetor altura:

```
alt <- c(180,170,160,150,140,168)
```

Reunir tudo em um `data.frame`:

```
dados <- data.frame(nome, idade, sexo, alt)
```

Ver atributos da tabela:

```
str(dados)
```

```
## 'data.frame':   6 obs. of  4 variables:
## $ nome : Factor w/ 6 levels "Edinar da Silva",...: 5 4 2 1 6 3
## $ idade: num  17 18 16 15 15 18
## $ sexo : Factor w/ 2 levels "F","M": 1 1 1 1 2 2
## $ alt  : num  180 170 160 150 140 168
```

Adicionar nome às linhas com o comando `row.names()`:

```

row.names(dados) <- c(1,2,3,4,5,6)
dados

##           nome  idade sexo alt
## 1      Melissa José    17   F 180
## 2 Jennifer Linhares    18   F 170
## 3 Gedilene Ponciano    16   F 160
## 4 Edinar da Silva     15   F 150
## 5 Osmar Emidio        15   M 140
## 6 Jeeziel Vieira      18   M 168

names(dados) <- c("Nome", "Idade", "Sexo", "altura")
dados

##           Nome Idade Sexo altura
## 1      Melissa José    17   F 180
## 2 Jennifer Linhares    18   F 170
## 3 Gedilene Ponciano    16   F 160
## 4 Edinar da Silva     15   F 150
## 5 Osmar Emidio        15   M 140
## 6 Jeeziel Vieira      18   M 168

```

3.5.1 Índice dos Data.frames

Buscar elementos:

```

dados[2,1] #elemento da linha 2, coluna 1

## [1] Jennifer Linhares
## 6 Levels: Edinar da Silva Gedilene Ponciano ... Osmar Emidio

dados[2,] #toda linha dois

##           Nome Idade Sexo altura
## 2 Jennifer Linhares    18   F 170

```

Repare que apesar de “Nomes” ter sido criado como vetor de caracteres, o R passou a entender como um fator dentro do data.frame:

```
dados[,1]
```

```
## [1] Melissa José      Jennifer Linhares Gedilene Ponciano Edinar da Silva
## [5] Osmar Emidio       Jeeziel Vieira
## 6 Levels: Edinar da Silva Gedilene Ponciano ... Osmar Emidio
```

Transformar para caracteres:

```
dados[,1] <- as.character(dados[,1])
dados[,1]
```

```
## [1] "Melissa José"      "Jennifer Linhares" "Gedilene Ponciano"
## [4] "Edinar da Silva"    "Osmar Emidio"       "Jeeziel Vieira"
```

Acessando os dados:

```
dados$Nome
```

```
## [1] "Melissa José"      "Jennifer Linhares" "Gedilene Ponciano"
## [4] "Edinar da Silva"    "Osmar Emidio"       "Jeeziel Vieira"
```

```
dados$Nome[3]
```

```
## [1] "Gedilene Ponciano"
```

```
dados$Nome [1:3]
```

```
## [1] "Melissa José"      "Jennifer Linhares" "Gedilene Ponciano"
```

```
str(dados)
```

```
## 'data.frame':   6 obs. of  4 variables:
## $ Nome  : chr  "Melissa José" "Jennifer Linhares" "Gedilene Ponciano" "Edinar da S
## $ Idade : num  17 18 16 15 15 18
## $ Sexo  : Factor w/ 2 levels "F","M": 1 1 1 1 2 2
## $ altura: num  180 170 160 150 140 168
```

3.5.2 Manipulando um Data.frame

Você pode manipular um data.frame adicionando ou eliminando colunas ou linhas, assim como em matrizes. Pode-se usar os comandos `cbind()` e `rbind()` para adicionar colunas e linhas respectivamente a um data.frame:

```

dados <- cbind (dados, #adicionar uma coluna
Conceito=c("A","A","A","C","A","B"))

dados <- rbind (dados, #adicionar uma linha
"7"= c("Caio Pinto", 21, "M", 172, "C"))
dados

##           Nome Idade Sexo altura Conceito
## 1      Melissa José   17    F    180       A
## 2 Jennifer Linhares   18    F    170       A
## 3 Gedilene Ponciano   16    F    160       A
## 4   Edinar da Silva   15    F    150       C
## 5      Osmar Emidio   15    M    140       A
## 6     Jeeziel Vieira   18    M    168       B
## 7      Caio Pinto     21    M    172       C

```

Assim como para vetores e matrizes, você pode selecionar um subgrupo de um data.frame e armazená-lo em um outro objeto ou utilizar índices como o sinal negativo para eliminar linhas ou colunas de um data.frame:

```

dados<- dados [1:6,] #selecionar linha de 1 a 6
dados<- dados [,-5] #excluir a quinta coluna
dados

```

```

##           Nome Idade Sexo altura
## 1      Melissa José   17    F    180
## 2 Jennifer Linhares   18    F    170
## 3 Gedilene Ponciano   16    F    160
## 4   Edinar da Silva   15    F    150
## 5      Osmar Emidio   15    M    140
## 6     Jeeziel Vieira   18    M    168

```

```

dados[dados$Sexo=="F",] #exibir só masculinos

```

```

##           Nome Idade Sexo altura
## 1      Melissa José   17    F    180
## 2 Jennifer Linhares   18    F    170
## 3 Gedilene Ponciano   16    F    160
## 4   Edinar da Silva   15    F    150

```

A ordenação das linhas de um **data.frame** segundo os dados contidos em determinadas colunas também é extremamente útil:

```

dados [order(dados$altura),]

##           Nome Idade Sexo altura
## 5      Osmar Emidio    15   M    140
## 4     Edinar da Silva    15   F    150
## 3  Gedilene Ponciano    16   F    160
## 6    Jeeziel Vieira    18   M    168
## 2 Jennifer Linhares    18   F    170
## 1    Melissa José     17   F    180

dados [rev(order(dados$altura)),]

##           Nome Idade Sexo altura
## 1    Melissa José     17   F    180
## 2 Jennifer Linhares    18   F    170
## 6    Jeeziel Vieira    18   M    168
## 3  Gedilene Ponciano    16   F    160
## 4     Edinar da Silva    15   F    150
## 5      Osmar Emidio    15   M    140

```

3.5.3 Separando um data.frame por grupos

```

split (dados, sexo)

## $F
##           Nome Idade Sexo altura
## 1    Melissa José     17   F    180
## 2 Jennifer Linhares    18   F    170
## 3  Gedilene Ponciano    16   F    160
## 4     Edinar da Silva    15   F    150
##
## $M
##           Nome Idade Sexo altura
## 5      Osmar Emidio    15   M    140
## 6    Jeeziel Vieira    18   M    168

```

3.6 Lista

Listas são objetos muito úteis, pois são usadas para combinar diferentes estruturas de dados em um mesmo objeto, ou seja, vetores, matrizes, arrays, data.frames e até mesmo outras listas:

```

pes <- list (idade=32, nome="Maria", notas=c(98,95,78), B=matrix(1:4,2,2))
pes

## $idade
## [1] 32
##
## $nome
## [1] "Maria"
##
## $notas
## [1] 98 95 78
##
## $B
##      [,1] [,2]
## [1,]     1     3
## [2,]     2     4

```

Listas são construídas com o comando `list ()`. Quando você exibe um objeto que é uma lista, cada componente é exibido com o seu nome `$` ou `[]`:

3.6.1 Alguns comandos que retornam listas

Muitos comandos do R retornam seu resultado na forma de listas. Um exemplo pode ser visualizado com o uso do comando `t.test()`, que retorna um objeto, sendo este uma lista:

```

x <- c(1,3,2,3,4)
y <- c(4,5,5,4,4)
tt <- t.test (x,y, var.equal=T)
tt

##
##  Two Sample t-test
##
## data:  x and y
## t = -3.182, df = 8, p-value = 0.01296
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -3.1044729 -0.4955271
## sample estimates:
## mean of x mean of y
##          2.6          4.4

```

Comprovar que é uma lista:

```
is.list(tt)
```

```
## [1] TRUE
```

```
mode (tt)
```

```
## [1] "list"
```

Exibir o componentes da lista:

```
names(tt)
```

```
## [1] "statistic"    "parameter"    "p.value"      "conf.int"     "estimate"
## [6] "null.value"   "stderr"       "alternative"  "method"      "data.name"
```

```
tt$conf.int #intervalo de confiança
```

```
## [1] -3.1044729 -0.4955271
## attr(,"conf.level")
## [1] 0.95
```

3.7 Referência

MELO, M. P.; PETERNELI, L. A. **Conhecendo o R: Um visão mais que estatística.** Viçosa, MG: UFV, 2013. 222p.

Prof. Paulo Justiniando Ribeiro ><http://www.leg.ufpr.br/~paulojus/><

Prof. Adriano Azevedo Filho ><http://rpubs.com/adriano/esalq2012inicial><

Prof. Fernando de Pol Mayer ><https://fernandomayer.github.io/ce083-2016-2/><

Site Interativo Datacamp ><https://www.datacamp.com/><

Chapter 4

Entrada de dados

Este terceiro capítulo foi baseado no livro **Conhecendo o R: Um visão mais que estatística**, e na página do **Prof. Paulo Justiniando Ribeiro** modificações foram realizadas utilizando outros materiais que se encontram referenciados no final desse capítulo.

O diretório de trabalho é aquele usado pelo R para gravar, ler, importar e exportar arquivos quando nenhum outro caminho é explicitado.

4.1 Onde os dados devem estar?

Para saber onde os diretórios estão basta digitar o comando `getwd()`:

```
getwd() #para verificar diretório de trabalho
```

```
## [1] "D:/livro/TudodoRa"
```

Caso queira alterar o diretório de trabalho para um outro lugar, digite o comando `setwd()`:

```
setwd("C:/R_Curso") #para altear o diretório de trabalho
```

Outra forma de mudar o caminho é com o comando:

```
caminho<-file.choose() # ou usando as teclhas shift + Crtl + H
```

Este comando irá abrir uma tela para que o usuário navegue nas pastas e escolha o arquivo a ser aberto.

Você pode exibir o conteúdo do diretório com o comando `dir()`:

```
dir()

##  [1] "_bookdown.yml"                      "_bookdown_files"
##  [3] "_main.Rmd"                           "_output.yml"
##  [5] "01-R_basico.Rmd"                     "02-Estrutura_basica.Rmd"
##  [7] "03-Entrada_dados.Rmd"                 "04-Criando_graficos.Rmd"
##  [9] "05-Criando_graficos_2.Rmd"            "06-Teste_T_corre.Rmd"
## [11] "07-Anova_dic.Rmd"                     "08-Anova_DBC.Rmd"
## [13] "09-Anova_QL.Rmd"                      "10-Regressao_L_M.Rmd"
## [15] "11-Regressao_N_linear.Rmd"            "12-Multivariada.Rmd"
## [17] "13-Dados_clima.Rmd"                   "14-Dados_tempor.Rmd"
## [19] "16-Tendencia_tempora.Rmd"              "17-Analise_imagens.Rmd"
## [21] "18-Analise_imagens_sentinel.Rmd"       "19-Referencia.Rmd"
## [23] "book.bib"                            "CHM.tif"
## [25] "docs"                                "ET_Abtew.csv"
## [27] "ET_BrutsaertStrickler.csv"             "ET_ChapmanAustralian.csv"
## [29] "ET_GrangerGray.csv"                    "ET_HargreavesSamani.csv"
## [31] "ET_JensenHaise.csv"                    "ET_Makkink.csv"
## [33] "ET_MattShuttleworth.csv"                "ET_McGuinnessBordne.csv"
## [35] "ET_Penman.csv"                         "ET_PenmanMonteith.csv"
## [37] "ET_PenPan.csv"                         "ET_PriestleyTaylor.csv"
## [39] "ET_Romanenko.csv"                      "ET_Turc.csv"
## [41] "ETP.txt"                               "image"
## [43] "index.Rmd"                            "LICENSE"
## [45] "packages.bib"                          "preamble.tex"
## [47] "README.md"                            "search_index.json"
## [49] "sentinel2.tif"                         "style.css"
## [51] "tab.xls"                               "TudodoR.Rmd"
## [53] "TudodoR.Rproj"                        "TudodoR_files"
```

4.2 Entrando com dados

O formato mais adequado vai depender do tamanho do conjunto de dados, e se os dados já existem em outro formato para serem importados ou se serão digitados diretamente no R.

A seguir são descritas formas de entrada de dados com indicação de quando cada uma das formas deve ser usada.

4.2.1 Vetores

Podemos entrar com dados definindo vetores com o comando `c()`, conforme visto no capítulo 3:

```
vetor <- c(2,5,7)
```

Esta forma de entrada de dados é conveniente quando se tem um pequeno número de dados. Quando os dados têm algum elemento repetido, números sequenciais podem ser usados com mecanismos do R para facilitar a entrada dos dados como vetores:

```
vetor <- rep(c(2,5), 5) # cria vetor repetindo 5 vezes 2 e 5 alternadamente
```

```
vetor
```

```
## [1] 2 5 2 5 2 5 2 5 2 5
```

```
vetor <- rep(c(5,8), each=3) # cria vetor repetindo 3 vezes 5 e depois 8
```

```
vetor
```

```
## [1] 5 5 5 8 8 8
```

4.2.2 Usando a função ‘scan’

Esta função coloca o modo prompt onde o usuário deve digitar cada dado seguido da tecla . Para encerrar a entrada de dados basta digitar duas vezes consecutivos. Veja o seguinte resultado:

```
y <- scan()
```

```
#1: 11
#2: 24
#3: 35
#4: 29
#5: 39
#6: 47
#7:
```

```
#Read 6 items
```

```
y
```

```
## numeric(0)
```

```
#[1] 11 24 35 29 39 47
```

Este formato é mais ágil que o anterior e mais conveniente para digitar vetores longos.

4.2.3 Copiar e colar usando `scan()`

Pode usar o recurso “copiar e colar” com o comando `scan`. Após copiar os dados (`crtl+C`), digite no **prompt/console** o comando `scan()`, aperte >ENTER<, depois cole o texto e, aperte >ENTER< novamente.

4.2.4 Lendo dados através da área de transferência

Funções como `scan()`, `read.table()` e outras podem ser usadas para ler os dados diretamente da área de transferência passando-se ao “*clipboard*” ao primeiro argumento:

4.2.5 Usando a função `edit`

O comando `edit(data.frame())` abre uma planilha para digitação de dados que são armazanados como data-frames:

```
dados <- edit(data.frame())
```

	trat	bloc	resp	var4	var5	var6
1	0	1	10			
2	0	2	11			
3	0	3	12			
4	50	1	31			
5	50	2	28			
6	50	3	29			
7	100	1	15			
8	100	2	16			
9	100	3	15			
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						

Se você precisar abrir novamente planilha com os dados, para fazer modificações e/ou inserir mais dados, use o comando `fix`:

```
fix(dados)
head(dados)
```

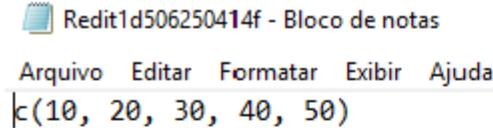
4.2.5.1 Exemplo 1

```
teste <- c(10,20,30,40,50)
teste
```

```
## [1] 10 20 30 40 50
```

Porém houve um erro: o último elemento deveria ser 60 e não 50, você não precisa criar novamente um objeto, use o comando `edit()`:

```
teste2 <- edit(teste)
```



4.2.5.2 Exemplo 2

Com uma planilha com três colunas de dados. Os valores numéricos da coluna poderiam ser importados para o R utilizando-se o mesmo processo descrito com o uso do comando `scan()`. Abra o arquivo: EVI-prec.xlsx.

Uma matrix com os dados poderá ser obtida com o comando `cbind`:

```
dados <- cbind(ano, chuva, evi)
```

O objeto dados é um **data.frame**:

```
is.data.frame(dados)
```

Transforme para um data.frame com o comando **as.data.frame**:

```
dados_m <- as.data.frame(dados)
```

Poderia usar o comando `data.frame` direto:

```
dados=data.frame (ano, chuva, evi)
```

4.2.6 Lendo dados de um arquivo texto

É muito importante ter os dados tabulados em um arquivo-texto ou em outros formatos que permitem a conversão para dados texto. O comando `read.table()` é extremamente útil por ler dados de um arquivo-texto no formato de um `data.frame`

Usando o Comando `read.table ()`

4.2.6.1 Exemplo 1

Como primeiro exemplo considere importar para o R os dados do arquivo texto: `exemplo1.txt`.

```
ex01 <- read.table("exemplo1.txt")  
  
#Use os comandos  
ex01  
class(ex01)  
names(ex01)  
dim(ex01)  
str(ex01)  
head(ex01)
```

4.2.6.2 Exemplo 2

Como primeiro exemplo considere importar para o R os dados do arquivo de texto: `exemplo2.txt`.

```
ex02 <- read.table("exemplo2.txt")  
ex02
```

Note que este arquivo difere do anterior em um aspecto: os nomes das variáveis estão na primeira linha. Para que o R considere isto corretamente, temos que informá-lo com o argumento `head=T`. Portanto para importar este arquivo usamos:

```
ex02 <- read.table("exemplo02.txt", head=T)
ex02
```

4.2.7 Dados do tipo CSV

Exemplo3.csv: Vamos utilizar um arquivo de tipo **CSV**:

```
ex03 <- read.table("exemplo3.csv.", head=T, sep=":", dec=",")
ex03
```

Note que este arquivo difere do primeiro em outros aspectos: *read.table*.

```
ex03 <- read.table(          # lê dados de um arquivo texto
  "exemplo3.csv",           # nome do arquivo ou o caminho c:/R.exemplo3.csv
  head=T,                 # primeira linha ? cabeçalho
  sep=":",                # separador de coluna
  dec=",")                 # vírgula como separador
ex03                      # exibe o objeto
```

1.sep: caractere utilizado para separação dos campos e valores. Normalmente é utilizado o ponto e vírgula (;

1.dec: caractere utilizado para separar as casas decimais. Normalmente ponto (.) ou vírgula (,).

1.header: TRUE, assume que a primeira linha da tabela contém rotulos das variáveis. ‘FALSE’, assume que os dados se iniciam na primeira linha.

4.2.8 A seguir listamos algumas destas funções:

1. *read.dbf()* para arquivos DBASE
2. *read.epiinfo()* para arquivos .REC do Epi-Info
3. *read.mtp()* para arquivos “Minitab Portable Worksheet”
4. *read.S()* para arquivos do S-PLUS, e *restore.data()* para “dumps” do S-PLUS
5. *read.spss()* para dados do SPSS
6. *read.systat()* para dados do SYSTAT
7. *read.dta()* para dados do STATA
8. *read.octave()* para dados do OCTAVE (um clone do MATLAB)
9. *read.csv(file, header = TRUE, sep = “,”, dec = “.”)*
10. *read.csv2(file, header = TRUE, sep = “;”, dec = “,”)*
11. *read.delim(file, header = TRUE, sep = “;”, dec = “.”)*
12. *read.delim2(file, header = TRUE, sep = “;”, dec = “,”)*

4.2.9 Lendo dados disponíveis na web

Exemplo 4: As funções permitem ler ainda dados diretamente disponíveis na web. Por exemplo os dados do exemplo1.txt poderiam ser lidos diretamente com o comando a seguir:

4.2.10 Lendo dados de uma planilha eletrônica

Com o **pacote xlsx** é possível ler os dados diretamente da planilha eletrônica do Excel.

```
install.packages("")  
require("xlsx")
```

O comando *read.xlsx()*, do **pacote xlsx**, lê o conteúdo de uma planilha eletrônica para o R com a estrutura de dados de um *data.frame*:

```
dados <- read.xlsx(  
  file="C:/R/EVI_Prec.xlsx",      # comando que lê planilhas  
  sheetName = "Conbea",          # nome da planilha  
  h=T)                          # sem cabeçalho
```

4.2.11 Exercícios

1. Baixe os seguintes arquivos:

- BanzattoQd1.2.3.txt
- BanzattoQd3.2.1.txt
- BanzattoQd3.4.1.txt

Coloque os arquivos em um local apropriado (de preferência no mesmo diretório de trabalho que você definiu no início da sessão), faça a importação usando a função de sua escolha, e confira a estrutura dos dados com `'str()'`:

4.3 Salvar objetos de dados

Salvar objetos de dados nos formatos **.txt** ou **.csv** função: **write.table** sintaxe da função: `write.table(x, file, sep="\"", dec="", rownames = T, col.names = T)`

Principais argumentos: 1. x - matriz ou data frame 1. file - nome do arquivo ou caminho do arquivo 1. sep - separador da coluna 1. dec - separador decimal

4.3.1 Outras funções

`write.csv() write.csv2() write.xlsx ()`

Exemplo: `write.xlsx(dados,“tabela salva.xlsx”)`

4.4 Referência

MELO, M. P.; PETERNELI, L. A. **Conhecendo o R: Um visão mais que estatística.** Viçosa, MG: UFV, 2013. 222p.

Prof. Paulo Justiniano Ribeiro ><http://www.leg.ufpr.br/~paulojus/><

Prof. Adriano Azevedo Filho ><http://rpubs.com/adriano/esalq2012inicial><

Prof. Fernando de Pol Mayer ><https://fernandomayer.github.io/ce083-2016-2/><

Chapter 5

Criando Gráficos com o R

Este capítulo foi baseado nos livros **Conhecendo o R: Um visão mais que estatística**.

AQUINO, J. A. **R para cientistas sociais**. - Ilhéus, BA: EDITUS, 2014. 157.

ANJOS, A. **Análise gráfica com uso do R**. Apostila. Dep. de Estatística da UFPR, 2016. 127p.

Sites:

<https://www.statmethods.net/index.html> <http://curso-r.github.io/index.html>
PET Estatística UFPR (2016). **labestData: Biblioteca de Dados para Aprendizado de Estatística**. R package version x.y-z.w.
<https://www.statmethods.net/index.html>

O R é uma poderosa ferramenta no que diz respeito à confeção de gráficos. Iremos abordar três categorias de comandos gráficos, com o uso do pacote básico do R o “graphics”. Alguns pacotes foram desenvolvidos especialmente para manipulação de gráficos, como *lattice*, *ggplot2*, *ggobi* e *rgl*.

O R possui diferentes funções geradoras de gráficos, e essas são classificados como:

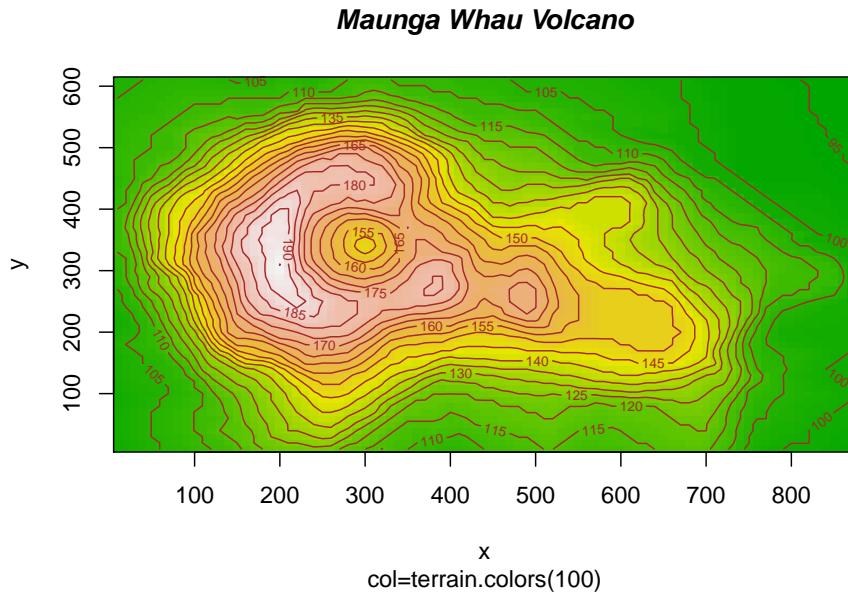
- *Funções gráficas de alto nível*: criam novos gráficos na janela, definindo eixos, título, etc. Exemplos: *plot*, *hist*, *image*, *contour*, *persp* etc.
- *Funções gráficas de baixo nível*: permitem adicionar novas informações em gráficos já criados, como novos dados, linhas etc. Exemplos: *points*, *lines*, *abline*, *polygon*, *legend* etc.
- *Funções gráficas iterativas*: permitem retirar ou adicionar informações aos gráficos já existentes, usando por exemplo o cursor do mouse. Exemplos: *locator* e *identify*.

5.1 Exemplos de gráficos com o R

Você pode ver alguns exemplos de gráficos que podem ser criados no R com os seguintes comandos:

```
demo(image)
```

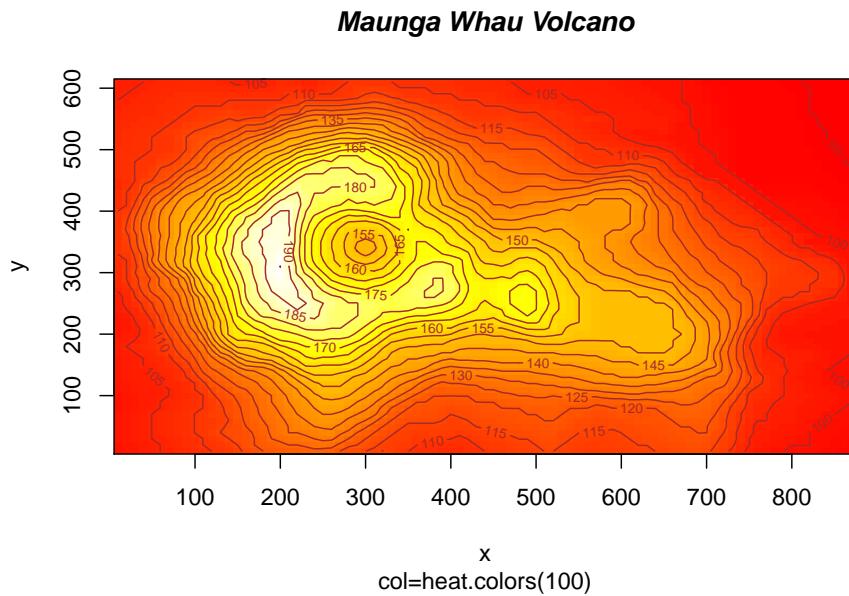
```
##  
##  
##  demo(image)  
##  ----- ~~~~~~  
##  
## > # Copyright (C) 1997-2009 The R Core Team  
## >  
## > require(datasets)  
##  
## > require(grDevices); require(graphics)  
##  
## > x <- 10*(1:nrow(volcano)); x.at <- seq(100, 800, by=100)  
##  
## > y <- 10*(1:ncol(volcano)); y.at <- seq(100, 600, by=100)  
##  
## >                      # Using Terrain Colors  
## >  
## > image(x, y, volcano, col=terrain.colors(100), axes=FALSE)
```



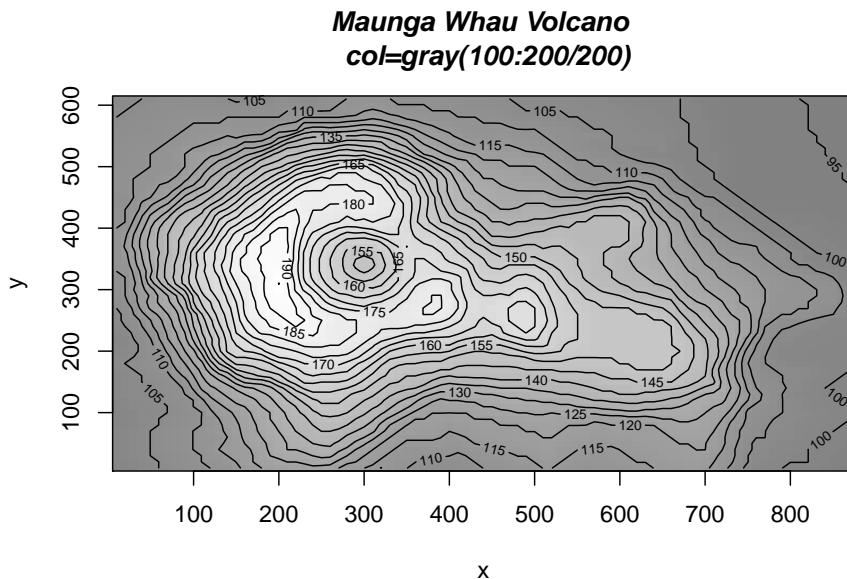
```

## 
## > contour(x, y, volcano, levels=seq(90, 200, by=5), add=TRUE, col="brown")
## 
## > axis(1, at=x.at)
## 
## > axis(2, at=y.at)
## 
## > box()
## 
## > title(main="Maunga Whau Volcano", sub = "col=terrain.colors(100)", font.main=4)
## 
## >                               # Using Heat Colors
## > 
## > image(x, y, volcano, col=heat.colors(100), axes=FALSE)

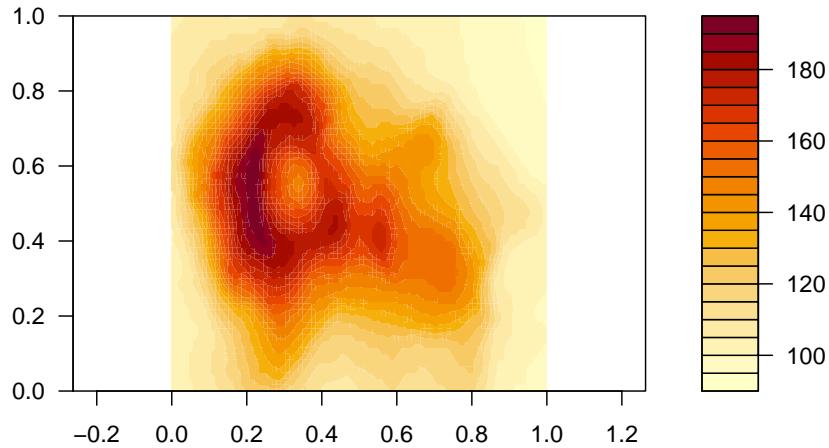
```



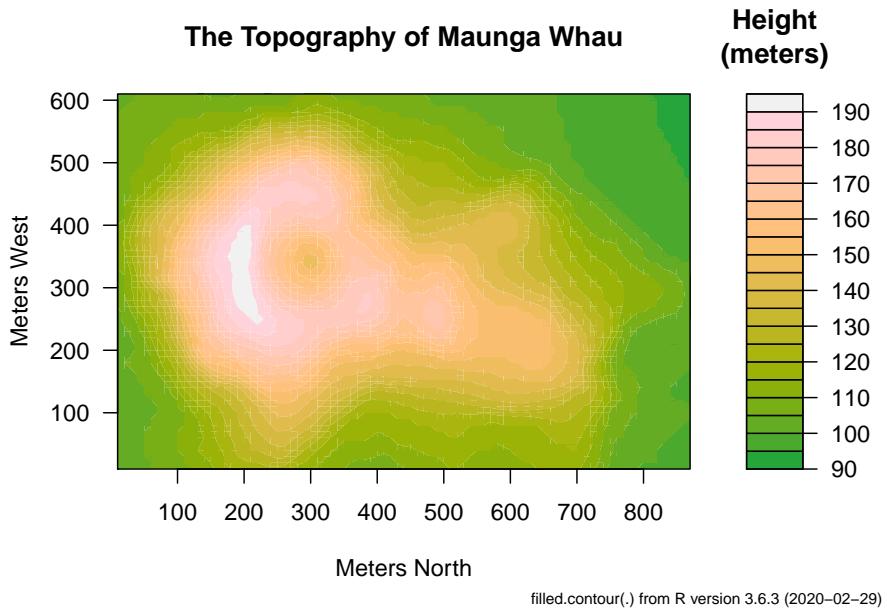
```
##> contour(x, y, volcano, levels=seq(90, 200, by=5), add=TRUE, col="brown")
##> axis(1, at=x.at)
##> axis(2, at=y.at)
##> box()
##> title(main="Maunga Whau Volcano", sub = "col=heat.colors(100)", font.main=4)
##> # Using Gray Scale
##>
##> image(x, y, volcano, col=gray(100:200/200), axes=FALSE)
```



```
##> contour(x, y, volcano, levels=seq(90, 200, by=5), add=TRUE, col="black")
##> axis(1, at=x.at)
##> axis(2, at=y.at)
##> box()
##> title(main="Maunga Whau Volcano \n col=gray(100:200/200)", font.main=4)
##> ## Filled Contours are even nicer sometimes :
##> example(filled.contour)
##> require("grDevices") # for colours
##> filled.contour(volcano, asp = 1) # simple
```



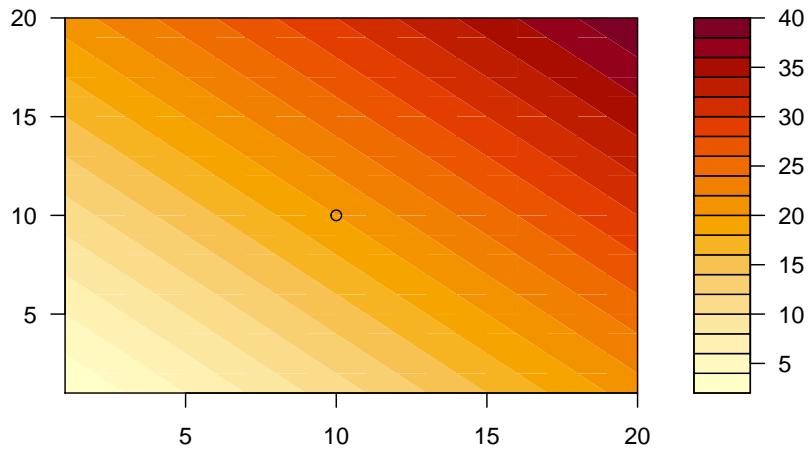
```
##  
## flld.c> x <- 10*1:nrow(volcano)  
##  
## flld.c> y <- 10*1:ncol(volcano)  
##  
## flld.c> filled.contour(x, y, volcano, color = function(n) hcl.colors(n, "terrain"),  
## flld.c+   plot.title = title(main = "The Topography of Maunga Whau",  
## flld.c+   xlab = "Meters North", ylab = "Meters West"),  
## flld.c+   plot.axes = { axis(1, seq(100, 800, by = 100))  
## flld.c+                           axis(2, seq(100, 600, by = 100)) },  
## flld.c+   key.title = title(main = "Height\n(meters)"),  
## flld.c+   key.axes = axis(4, seq(90, 190, by = 10))) # maybe also asp = 1
```



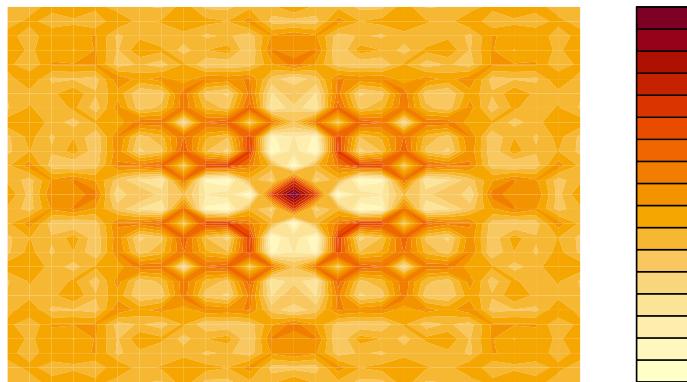
```

## 
## flld.c> mtext(paste("filled.contour(.) from", R.version.string),
## flld.c+      side = 1, line = 4, adj = 1, cex = .66)
## 
## flld.c> # Annotating a filled contour plot
## flld.c> a <- expand.grid(1:20, 1:20)
## 
## flld.c> b <- matrix(a[,1] + a[,2], 20)
## 
## flld.c> filled.contour(x = 1:20, y = 1:20, z = b,
##                         plot.axes = { axis(1); axis(2); points(10, 10) })

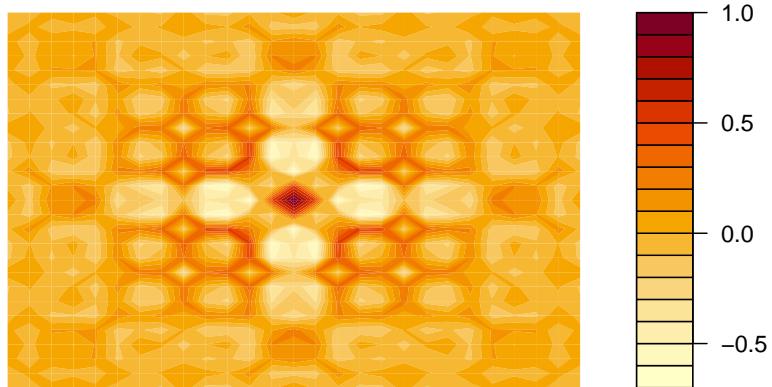
```



```
##  
## flld.c> ## Persian Rug Art:  
## flld.c> x <- y <- seq(-4*pi, 4*pi, len = 27)  
##  
## flld.c> r <- sqrt(outer(x^2, y^2, "+"))  
##  
## flld.c> filled.contour(cos(r^2)*exp(-r/(2*pi)), axes = FALSE)
```



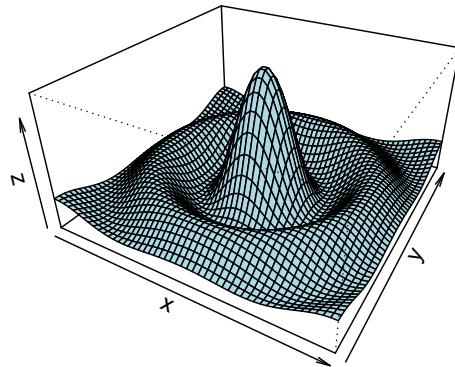
```
##  
## flld.c> ## rather, the key *should* be labeled:  
## flld.c> filled.contour(cos(r^2)*exp(-r/(2*pi)), frame.plot = FALSE,  
## flld.c+           plot.axes = {})
```



```
demo(persp)
```

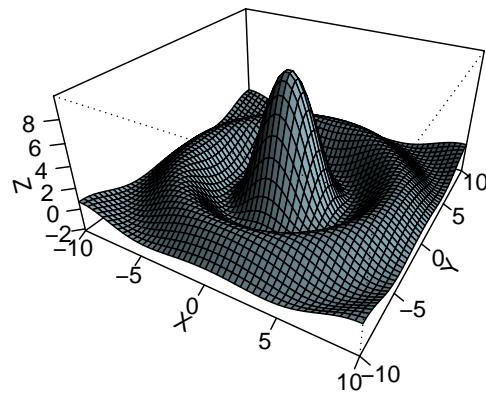
```
##  
##  
##  demo(persp)  
##  ----- ~~~~~~  
##  
## > ##### Demos for  persp()  plots  -- things not in  example(persp)  
## > ##### -----  
## >  
## > require(datasets)  
##  
## > require(grDevices); require(graphics)  
##  
## > ## (1) The Obligatory Mathematical surface.  
## > ##      Rotated sinc function.  
## >  
## > x <- seq(-10, 10, length.out = 50)  
##  
## > y <- x  
##  
## > rotsinc <- function(x,y)  
## + {  
## +     sinc <- function(x) { y <- sin(x)/x ; y[is.na(y)] <- 1; y }  
## +     10 * sinc( sqrt(x^2+y^2) )  
## + }  
##  
## > sinc.exp <- expression(z == Sinc(sqrt(x^2 + y^2)))  
##  
## > z <- outer(x, y, rotsinc)  
##  
## > oldpar <- par(bg = "white")  
##  
## > persp(x, y, z, theta = 30, phi = 30, expand = 0.5, col = "lightblue")
```

$$z = \text{Sinc}(\sqrt{x^2 + y^2})$$

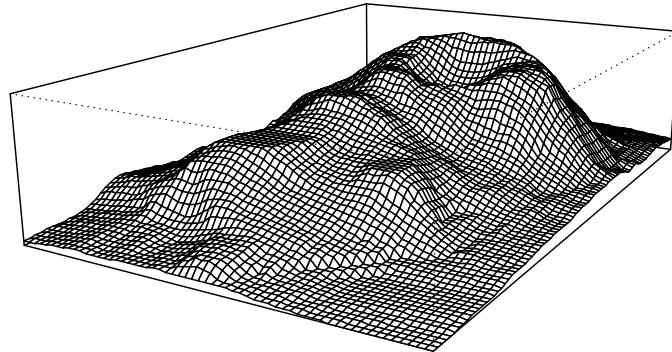


```
##  
## > title(sub=".")## work around persp+plotmath bug  
##  
## > title(main = sinc.exp)  
##  
## > persp(x, y, z, theta = 30, phi = 30, expand = 0.5, col = "lightblue",  
## +         ltheta = 120, shade = 0.75, ticktype = "detailed",  
## +         xlab = "X", ylab = "Y", zlab = "Z")
```

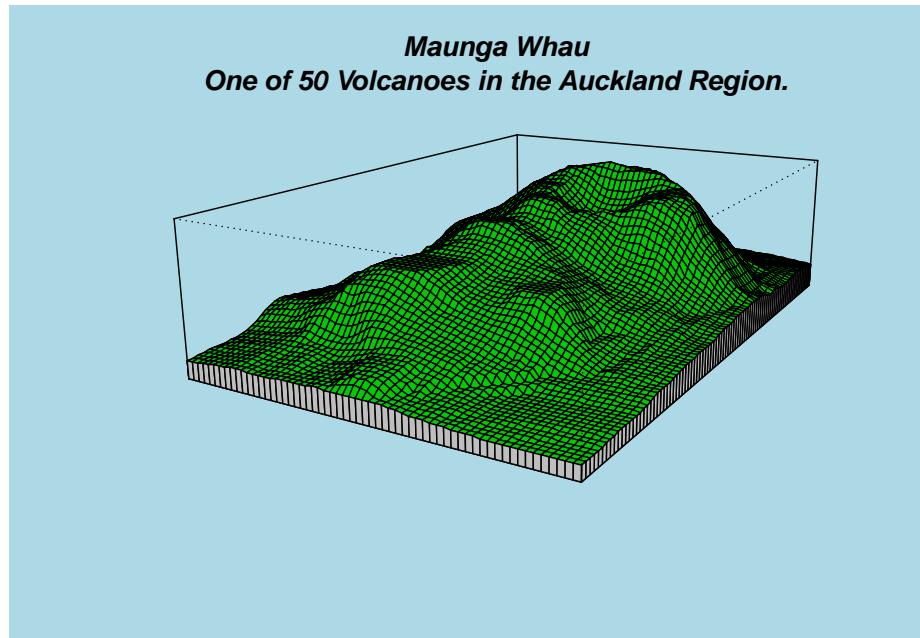
$$z = \text{Sinc}(\sqrt{x^2 + y^2})$$



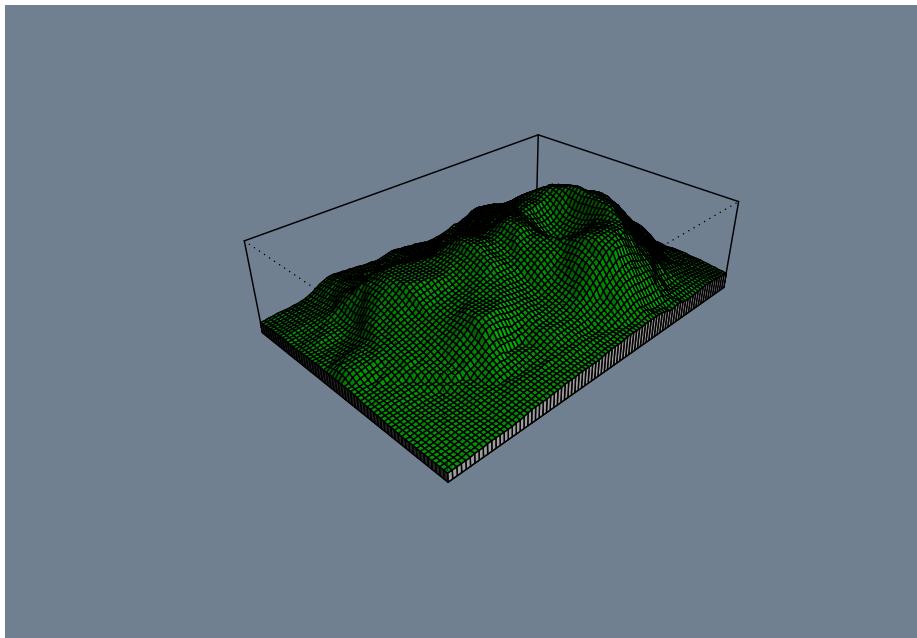
```
##
## > title(sub=".")## work around persp+plotmath bug
##
## > title(main = sinc.exp)
##
## > ## (2) Visualizing a simple DEM model
## >
## > z <- 2 * volcano          # Exaggerate the relief
##
## > x <- 10 * (1:nrow(z))    # 10 meter spacing (S to N)
##
## > y <- 10 * (1:ncol(z))    # 10 meter spacing (E to W)
##
## > persp(x, y, z, theta = 120, phi = 15, scale = FALSE, axes = FALSE)
```



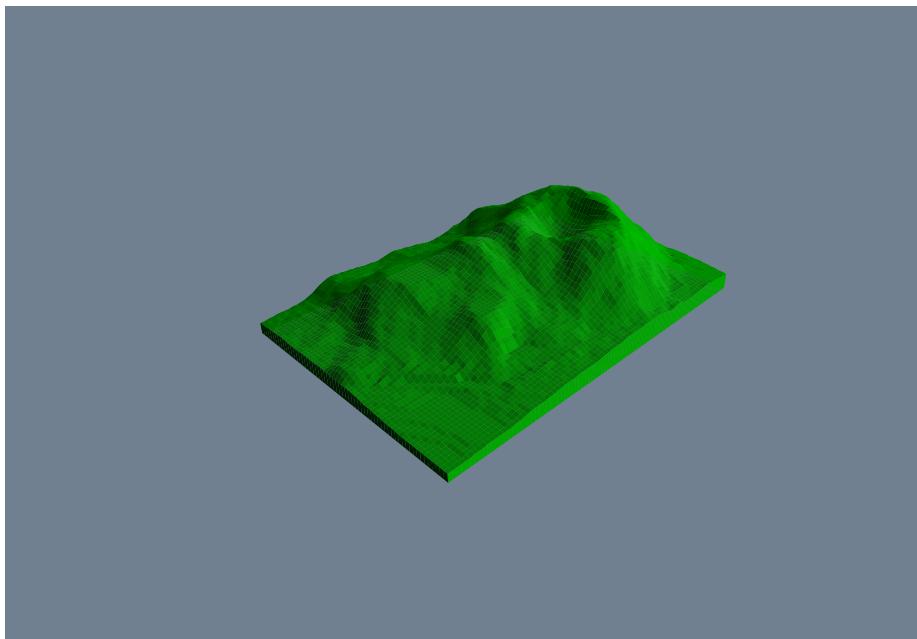
```
##  
## > ## (3) Now something more complex  
## > ##      We border the surface, to make it more "slice like"  
## > ##      and color the top and sides of the surface differently.  
## >  
## > z0 <- min(z) - 20  
##  
## > z <- rbind(z0, cbind(z0, z, z0), z0)  
##  
## > x <- c(min(x) - 1e-10, x, max(x) + 1e-10)  
##  
## > y <- c(min(y) - 1e-10, y, max(y) + 1e-10)  
##  
## > fill <- matrix("green3", nrow = nrow(z)-1, ncol = ncol(z)-1)  
##  
## > fill[ , i2 <- c(1,ncol(fill))] <- "gray"  
##  
## > fill[i1 <- c(1,nrow(fill)) , ] <- "gray"  
##  
## > par(bg = "lightblue")  
##  
## > persp(x, y, z, theta = 120, phi = 15, col = fill, scale = FALSE, axes = FALSE)
```



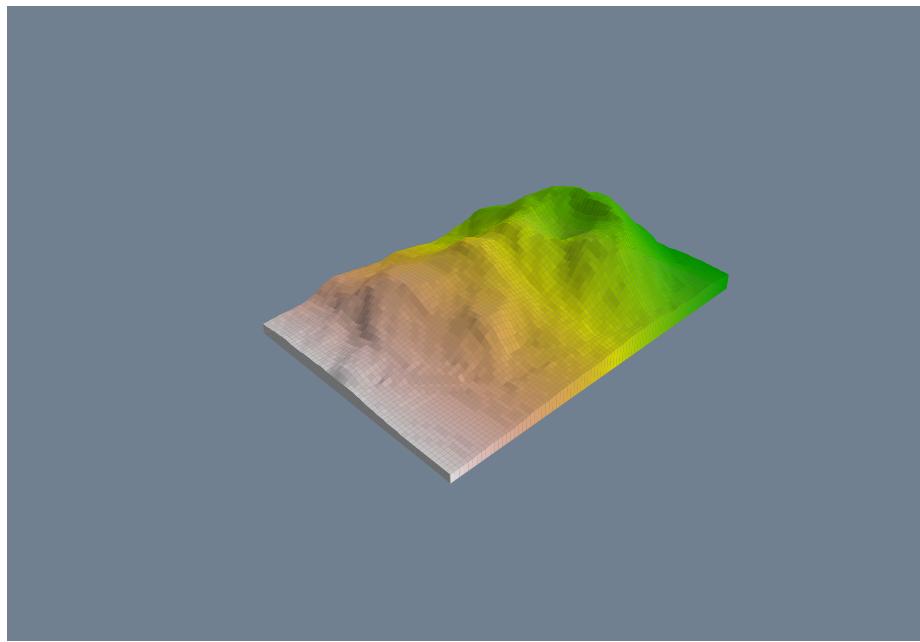
```
##  
## > title(main = "Maunga Whau\nOne of 50 Volcanoes in the Auckland Region.",  
## +         font.main = 4)  
##  
## > par(bg = "slategray")  
##  
## > persp(x, y, z, theta = 135, phi = 30, col = fill, scale = FALSE,  
## +         ltheta = -120, lphi = 15, shade = 0.65, axes = FALSE)
```



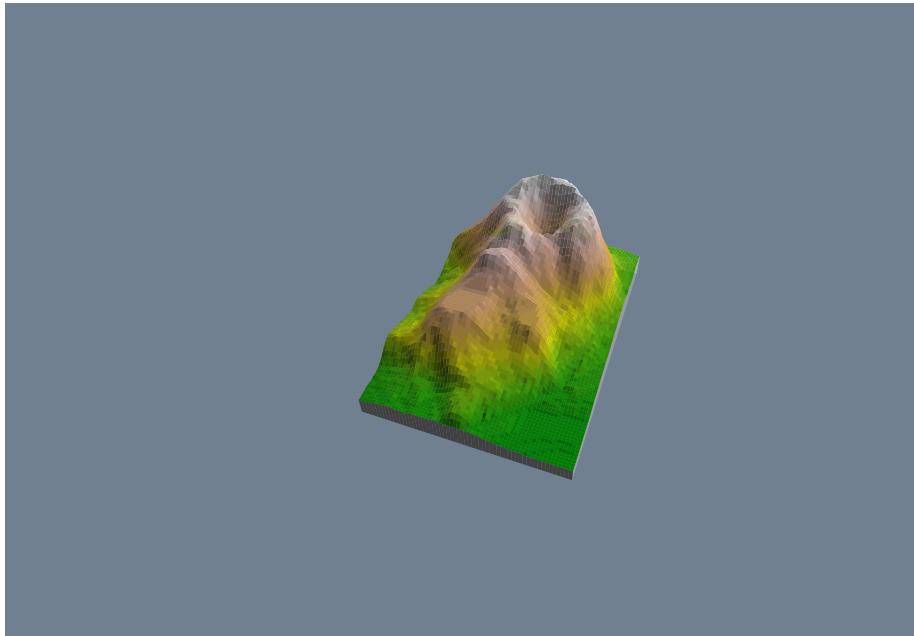
```
##  
## > ## Don't draw the grid lines : border = NA  
## > persp(x, y, z, theta = 135, phi = 30, col = "green3", scale = FALSE,  
## +         ltheta = -120, shade = 0.75, border = NA, box = FALSE)
```



```
##
## > ## `color gradient in the soil' :
## > fcol <- fill ; fcol[] <- terrain.colors(nrow(fcol))
##
## > persp(x, y, z, theta = 135, phi = 30, col = fcol, scale = FALSE,
## +         ltheta = -120, shade = 0.3, border = NA, box = FALSE)
```



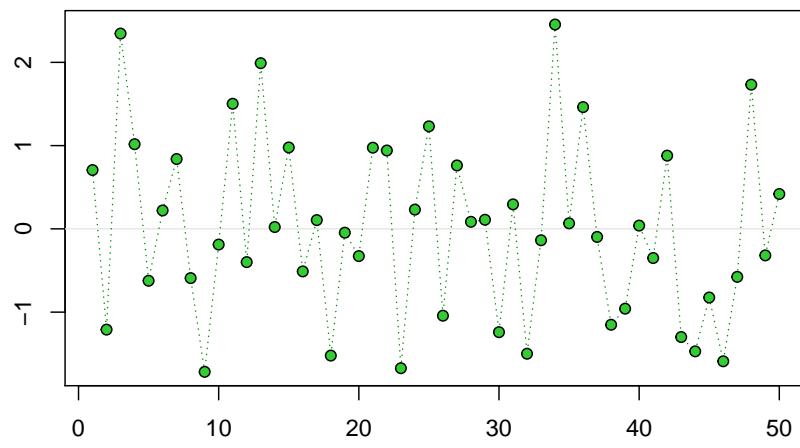
```
##
## > ## `image like' colors on top :
## > fcol <- fill
##
## > zi <- volcano[ -1,-1] + volcano[ -1,-61] +
## +           volcano[-87,-1] + volcano[-87,-61] ## / 4
##
## > fcol[-i1,-i2] <-
## +     terrain.colors(20)[cut(zi,
## +                           stats::quantile(zi, seq(0,1, length.out = 21)),
## +                           include.lowest = TRUE)]
##
## > persp(x, y, 2*z, theta = 110, phi = 40, col = fcol, scale = FALSE,
## +         ltheta = -120, shade = 0.4, border = NA, box = FALSE)
```



```
##  
## > ## reset par():  
## > par(oldpar)  
  
demo(graphics)  
  
##  
##  
##  demo(graphics)  
##  ----- ~~~~~~  
##  
## > # Copyright (C) 1997-2009 The R Core Team  
## >  
## > require(datasets)  
##  
## > require(grDevices); require(graphics)  
##  
## > ## Here is some code which illustrates some of the differences between  
## > ## R and S graphics capabilities. Note that colors are generally specified  
## > ## by a character string name (taken from the X11 rgb.txt file) and that line  
## > ## textures are given similarly. The parameter "bg" sets the background  
## > ## parameter for the plot and there is also an "fg" parameter which sets  
## > ## the foreground color.  
## >
```

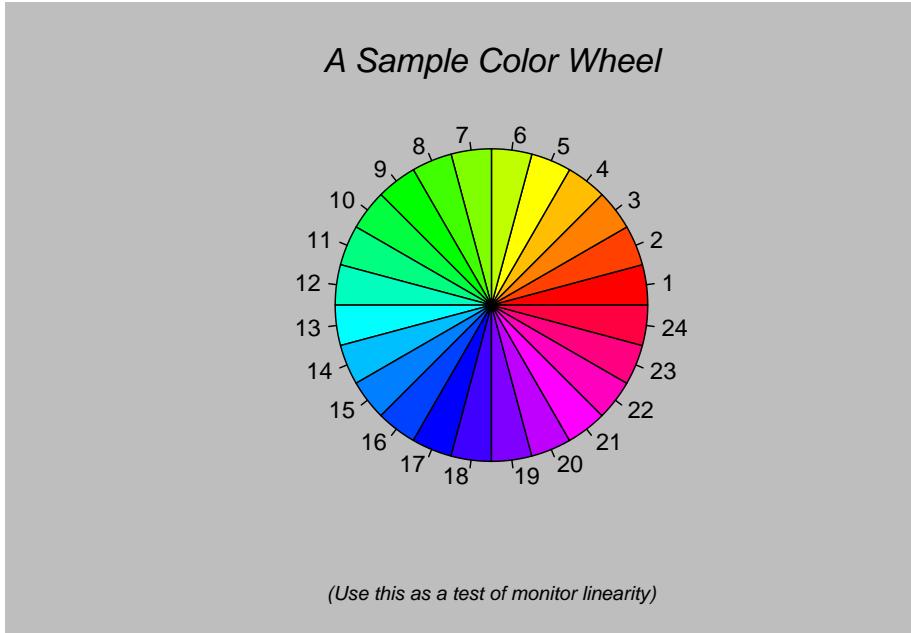
```
## >
## > x <- stats::rnorm(50)
##
## > opar <- par(bg = "white")
##
## > plot(x, ann = FALSE, type = "n")
```

Simple Use of Color In a Plot

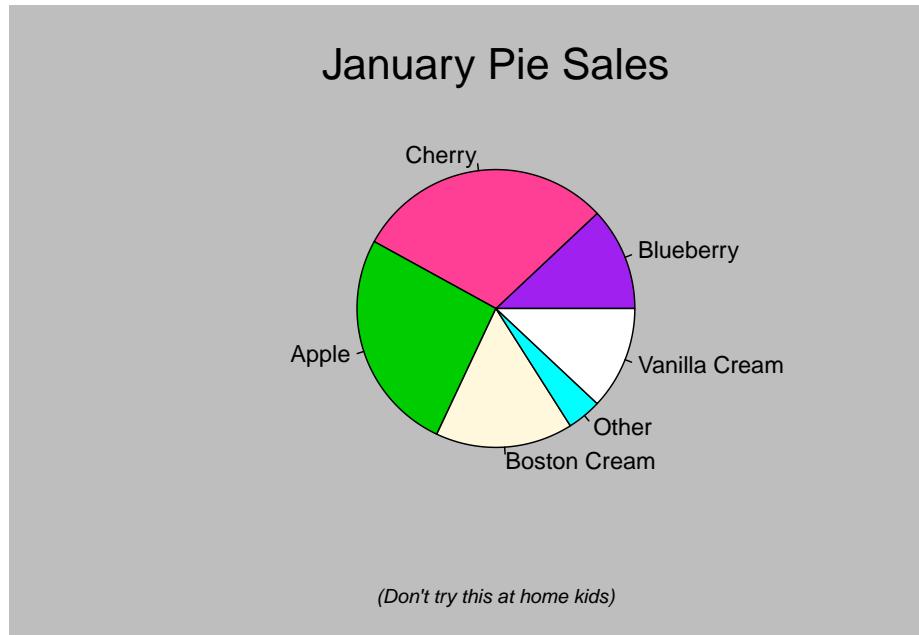


```
##
## > abline(h = 0, col = gray(.90))
##
## > lines(x, col = "green4", lty = "dotted")
##
## > points(x, bg = "limegreen", pch = 21)
##
## > title(main = "Simple Use of Color In a Plot",
## +       xlab = "Just a Whisper of a Label",
## +       col.main = "blue", col.lab = gray(.8),
## +       cex.main = 1.2, cex.lab = 1.0, font.main = 4, font.lab = 3)
##
## > ## A little color wheel. This code just plots equally spaced hues in
## > ## a pie chart. If you have a cheap SVGA monitor (like me) you will
## > ## probably find that numerically equispaced does not mean visually
## > ## equispaced. On my display at home, these colors tend to cluster at
## > ## the RGB primaries. On the other hand on the SGI Indy at work the
```

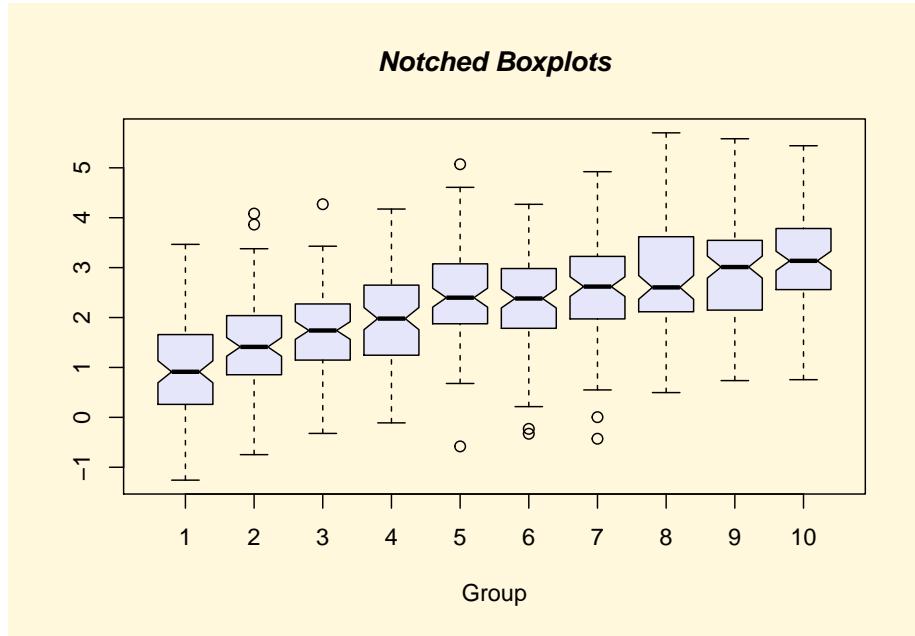
```
## > ## effect is near perfect.
## >
## > par(bg = "gray")
##
## > pie(rep(1,24), col = rainbow(24), radius = 0.9)
```



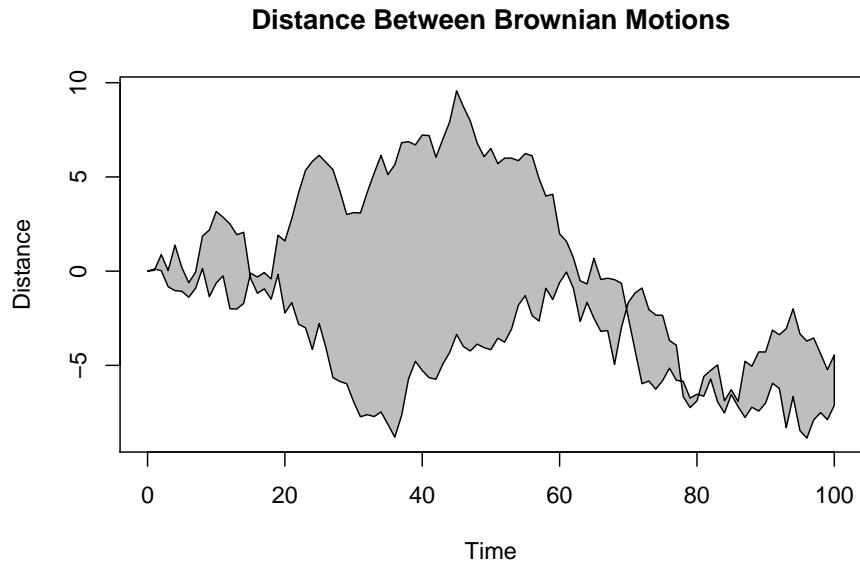
```
##
## > title(main = "A Sample Color Wheel", cex.main = 1.4, font.main = 3)
##
## > title(xlab = "(Use this as a test of monitor linearity)",
## +         cex.lab = 0.8, font.lab = 3)
##
## > ## We have already confessed to having these. This is just showing off X11
## > ## color names (and the example (from the postscript manual) is pretty "cute".
## >
## > pie.sales <- c(0.12, 0.3, 0.26, 0.16, 0.04, 0.12)
##
## > names(pie.sales) <- c("Blueberry", "Cherry",
## +                         "Apple", "Boston Cream", "Other", "Vanilla Cream")
##
## > pie(pie.sales,
## +      col = c("purple","violetred1","green3","cornsilk","cyan","white"))
```



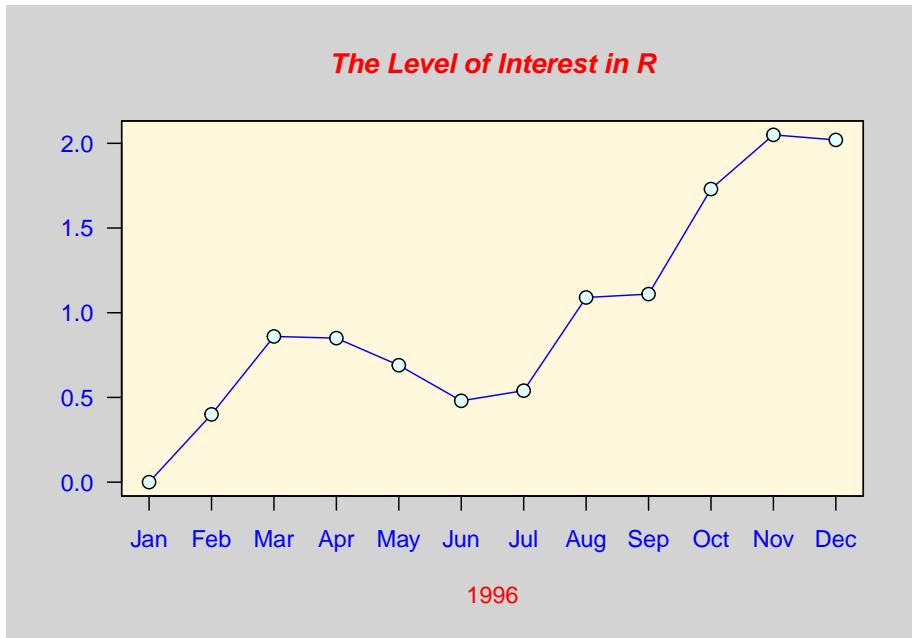
```
##> title(main = "January Pie Sales", cex.main = 1.8, font.main = 1)
##> title(xlab = "(Don't try this at home kids)", cex.lab = 0.8, font.lab = 3)
##> ## Boxplots: I couldn't resist the capability for filling the "box".
##> ## The use of color seems like a useful addition, it focuses attention
##> ## on the central bulk of the data.
##>
##> par(bg="cornsilk")
##> n <- 10
##> g <- gl(n, 100, n*100)
##> x <- rnorm(n*100) + sqrt(as.numeric(g))
##> boxplot(split(x,g), col="lavender", notch=TRUE)
```



```
##> title(main="Notched Boxplots", xlab="Group", font.main=4, font.lab=1)
##> ## An example showing how to fill between curves.
##>
##> par(bg="white")
##>
##> n <- 100
##>
##> x <- c(0,cumsum(rnorm(n)))
##>
##> y <- c(0,cumsum(rnorm(n)))
##>
##> xx <- c(0:n, n:0)
##>
##> yy <- c(x, rev(y))
##>
##> plot(xx, yy, type="n", xlab="Time", ylab="Distance")
```

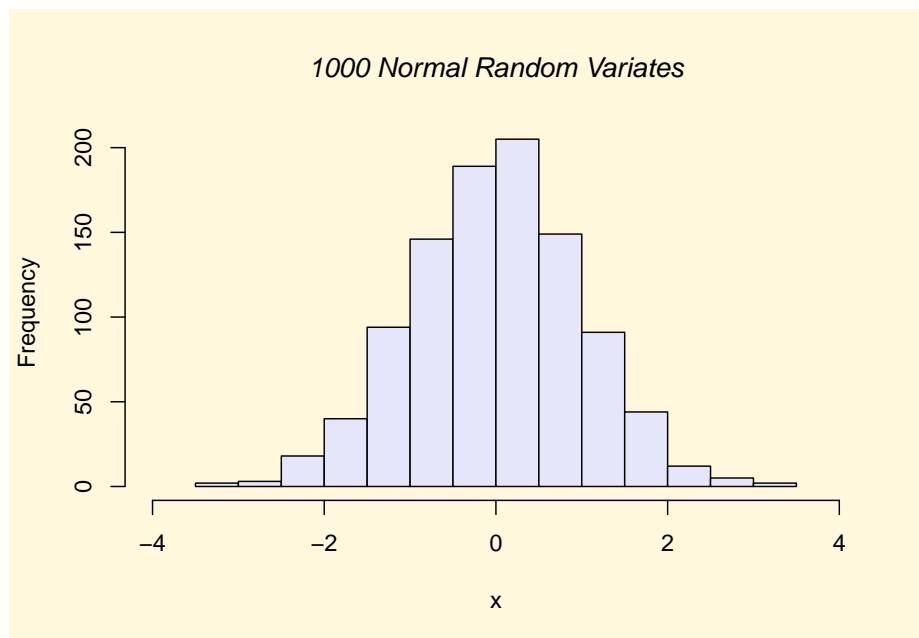


```
##  
## > polygon(xx, yy, col="gray")  
##  
## > title("Distance Between Brownian Motions")  
##  
## > ## Colored plot margins, axis labels and titles. You do need to be  
## > ## careful with these kinds of effects. It's easy to go completely  
## > ## over the top and you can end up with your lunch all over the keyboard.  
## > ## On the other hand, my market research clients love it.  
## >  
## > x <- c(0.00, 0.40, 0.86, 0.85, 0.69, 0.48, 0.54, 1.09, 1.11, 1.73, 2.05, 2.02)  
##  
## > par(bg="lightgray")  
##  
## > plot(x, type="n", axes=FALSE, ann=FALSE)
```

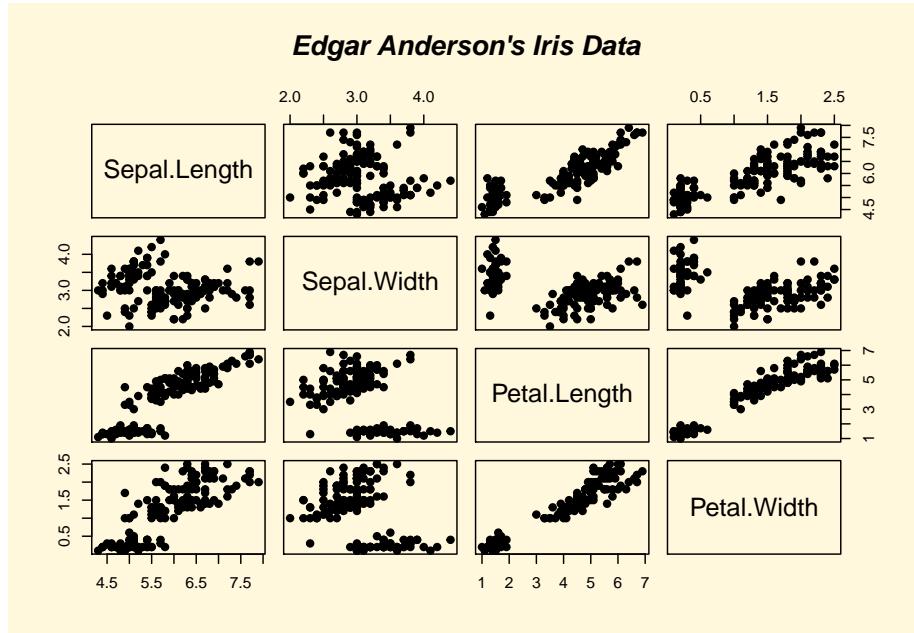


```
##  
## > usr <- par("usr")  
##  
## > rect(usr[1], usr[3], usr[2], usr[4], col="cornsilk", border="black")  
##  
## > lines(x, col="blue")  
##  
## > points(x, pch=21, bg="lightcyan", cex=1.25)  
##  
## > axis(2, col.axis="blue", las=1)  
##  
## > axis(1, at=1:12, lab=month.abb, col.axis="blue")  
##  
## > box()  
##  
## > title(main= "The Level of Interest in R", font.main=4, col.main="red")  
##  
## > title(xlab= "1996", col.lab="red")  
##  
## > ## A filled histogram, showing how to change the font used for the  
## > ## main title without changing the other annotation.  
## >  
## > par(bg="cornsilk")  
##  
## > x <- rnorm(1000)
```

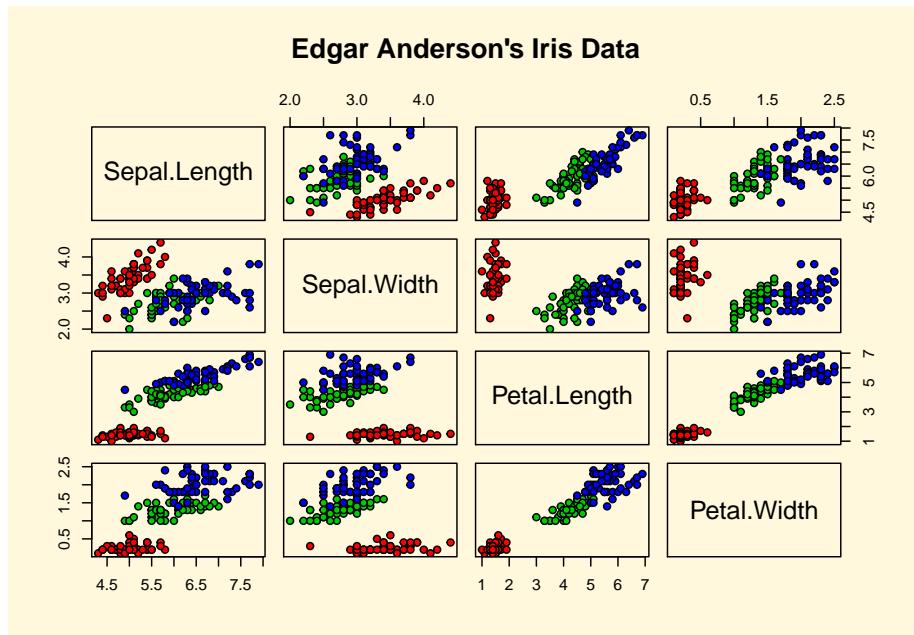
```
##  
## > hist(x, xlim=range(-4, 4, x), col="lavender", main="")
```



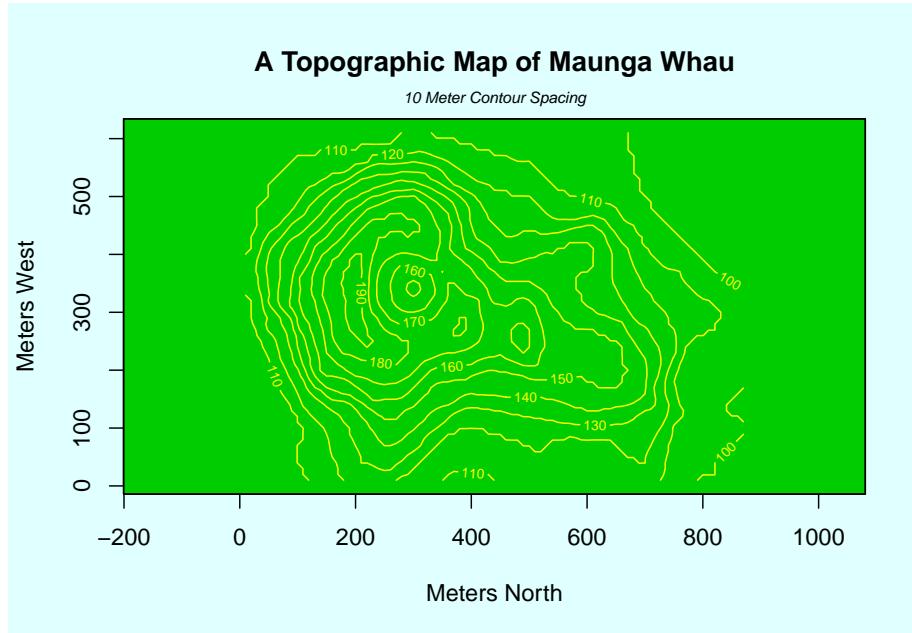
```
##  
## > title(main="1000 Normal Random Variates", font.main=3)  
##  
## > ## A scatterplot matrix  
## > ## The good old Iris data (yet again)  
## >  
## > pairs(iris[1:4], main="Edgar Anderson's Iris Data", font.main=4, pch=19)
```



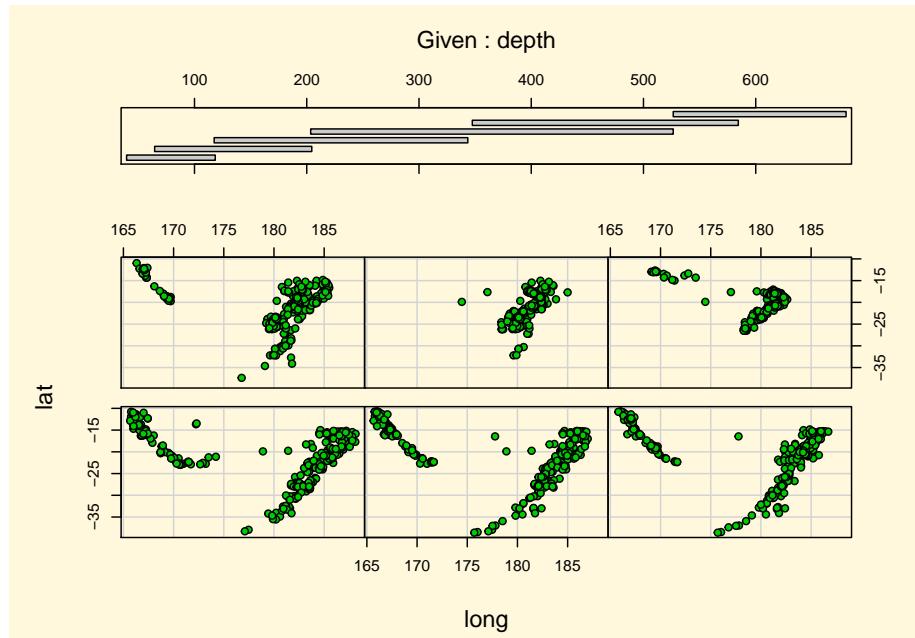
```
##> pairs(iris[1:4], main="Edgar Anderson's Iris Data", pch=21,
##+      bg = c("red", "green3", "blue")[unclass(iris$Species)])
```



```
##  
## > ## Contour plotting  
## > ## This produces a topographic map of one of Auckland's many volcanic "peaks".  
## >  
## > x <- 10*1:nrow(volcano)  
##  
## > y <- 10*1:ncol(volcano)  
##  
## > lev <- pretty(range(volcano), 10)  
##  
## > par(bg = "lightcyan")  
##  
## > pin <- par("pin")  
##  
## > xdelta <- diff(range(x))  
##  
## > ydelta <- diff(range(y))  
##  
## > xscale <- pin[1]/xdelta  
##  
## >yscale <- pin[2]/ydelta  
##  
## > scale <- min(xscale, yscale)  
##  
## > xadd <- 0.5*(pin[1]/scale - xdelta)  
##  
## > yadd <- 0.5*(pin[2]/scale - ydelta)  
##  
## > plot(numeric(0), numeric(0),  
## +       xlim = range(x)+c(-1,1)*xadd, ylim = range(y)+c(-1,1)*yadd,  
## +       type = "n", ann = FALSE)
```



```
##  
## > usr <- par("usr")  
##  
## > rect(usr[1], usr[3], usr[2], usr[4], col="green3")  
##  
## > contour(x, y, volcano, levels = lev, col="yellow", lty="solid", add=TRUE)  
##  
## > box()  
##  
## > title("A Topographic Map of Maunga Whau", font= 4)  
##  
## > title(xlab = "Meters North", ylab = "Meters West", font= 3)  
##  
## > mtext("10 Meter Contour Spacing", side=3, line=0.35, outer=FALSE,  
## +         at = mean(par("usr")[1:2]), cex=0.7, font=3)  
##  
## > ## Conditioning plots  
## >  
## > par(bg="cornsilk")  
##  
## > coplot(lat ~ long | depth, data = quakes, pch = 21, bg = "green3")
```



```
##  
## > par(opar)
```

5.2 Entrada de dados

Nesse tópico utilizaremos o arquivo de dados: `dadosfisio.csv`.

Dados físico hídrico de 3 solos com texturas diferentes:

Cod.	Solo	Areia	Silte	Argila
Z1	NITOSSOLO	122	121	757
Z2	LATOSSOLO	710	80	210
Z3	LATOSSOLO	892	10	98

Ler dados via web:

```
solo <- read.table("https://www.dropbox.com/s/zg7fyg1iewtji49/dadosfisio.csv?dl=1", se
```

Verificar a estrutura de dados:

```
str(solo)
```

```
## 'data.frame': 108 obs. of 16 variables:
## $ z      : int 1 1 1 1 1 1 1 1 1 ...
## $ x      : int 1 1 1 1 1 1 3 3 3 ...
## $ y      : int 1 3 5 7 9 11 1 3 5 7 ...
## $ cota   : num 9.15 8.95 8.78 8.59 8.48 8.41 8.93 8.76 8.58 8.48 ...
## $ ds     : num 1.5 1.47 1.47 1.39 1.38 ...
## $ cc     : num 0.398 0.382 0.351 0.372 0.356 ...
## $ ma     : num 0.129 0.153 0.185 0.188 0.208 ...
## $ ptotal : num 0.526 0.535 0.537 0.561 0.564 ...
## $ tibo   : num 46.1 19.2 172.8 96 30.7 ...
## $ tibe   : num 26.8 26.1 113.9 74.8 37.2 ...
## $ a      : num 926 384 275 1207 151 ...
## $ b      : num -0.529 -0.418 -0.131 -0.376 -0.227 ...
## $ X3     : num 518 243 238 798 118 ...
## $ X60    : num 153.2 92.7 176.5 335.4 69.9 ...
## $ X90    : num 106.2 69.4 161.2 258.4 59.8 ...
## $ X120   : num 73.6 52 147.2 199 51.1 ...
```

Resumo estatístico da coluna 5 a coluna 8 de todos os solos:

```
summary(solo[5:8])
```

	ds	cc	ma	ptotal
## Min.	:1.263	Min. :0.1501	Min. :0.004834	Min. :0.2257
## 1st Qu.	:1.500	1st Qu.:0.2505	1st Qu.:0.047689	1st Qu.:0.3090
## Median	:1.722	Median :0.2712	Median :0.081510	Median :0.3284
## Mean	:1.660	Mean :0.2998	Mean :0.090675	Mean :0.3905
## 3rd Qu.	:1.787	3rd Qu.:0.3579	3rd Qu.:0.129955	3rd Qu.:0.5269
## Max.	:1.960	Max. :0.4997	Max. :0.238551	Max. :0.6015

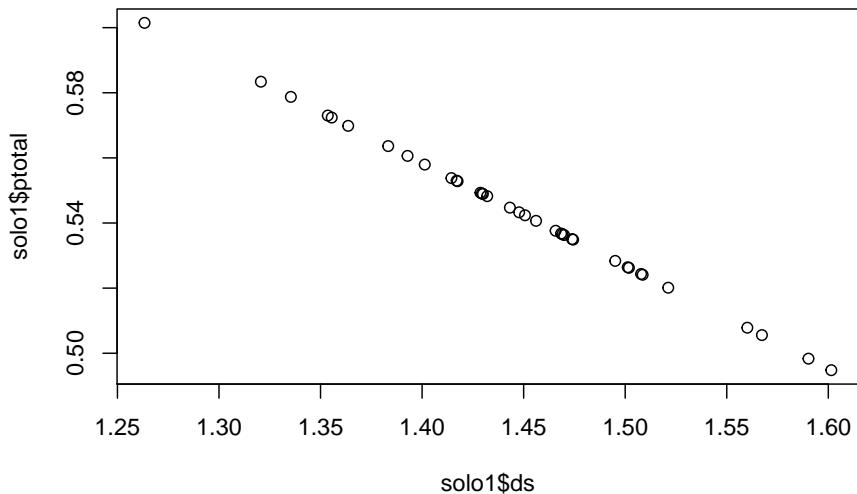
Neste exemplo vamos analisar cada solo separadamente usando o comando *subset()*:

```
solo1 <- subset(solo, z==1)
solo2 <- subset(solo, z==2)
solo3 <- subset(solo, z==3)
```

5.3 Usando a função *plot*()

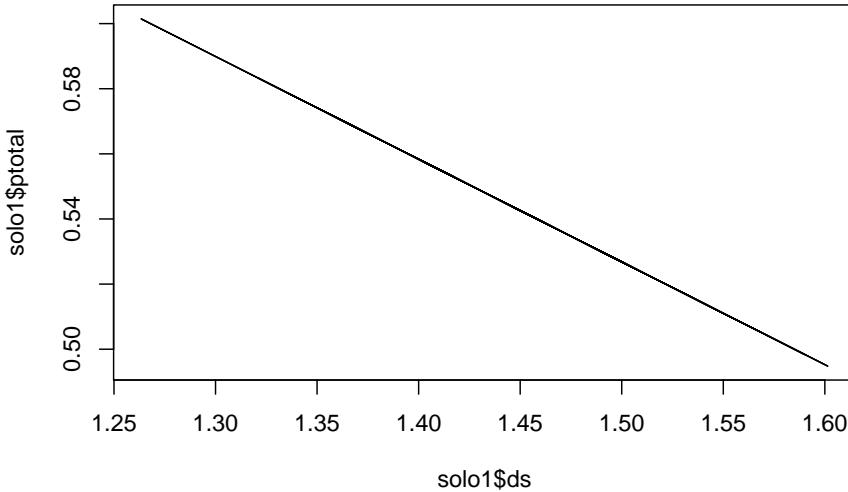
A função *plot()* inicia um novo gráfico. Em sua forma mais simples a função recebe valores de coordenadas *ds* (densidade do solo) e *ptotal* (porosidade total do solo) do solo z1:

```
plot(solo1$ds,solo1$ptotal)
```



Vamos inserir no gráfico linhas ligando os pontos. Use o argumento *type="l" na função `plot()`:

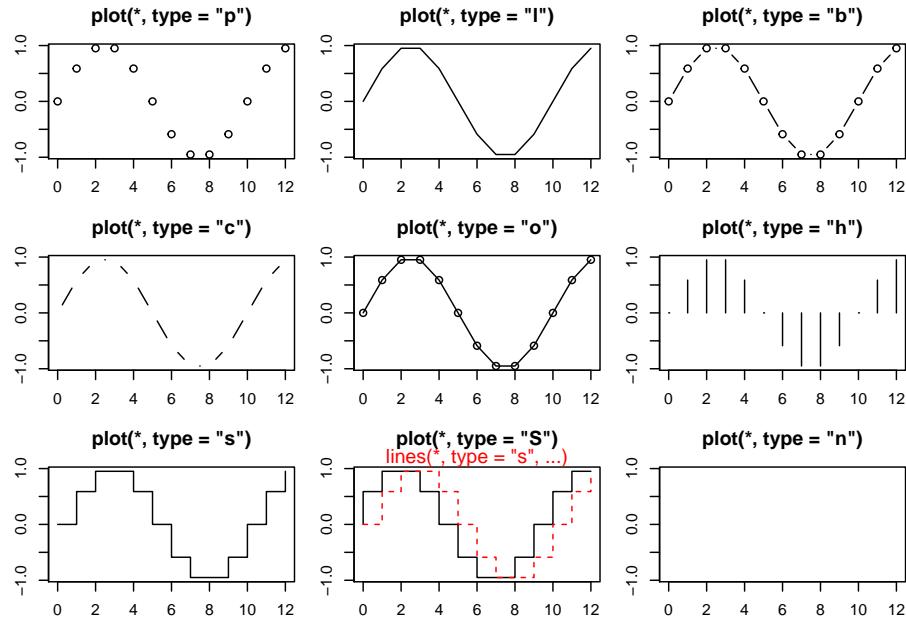
```
plot(solo1$ds,solo1$ptotal, type = "l")
```



Verifique outras opções para os gráficos:

- *type* = “*p*” especifica o tipo de plotagem
- “*p*”: pontos,
- “*l*”: linhas,
- “*b*”: pontos conectados por linhas,
- “*o*”: id. mas as linhas estão acima dos pontos,
- “*h*”: linhas verticais,
- “*s*”: passos, os dados são representados pelo topo das linhas verticais,
- “*S*”: id. mas os dados são representados pela parte inferior das linhas verticais

```
x <- 0:12
y <- sin(pi/5 * x)
op <- par(mfrow = c(3,3), mar = .1 + c(2,2,3,1))
for (tp in c("p", "l", "b", "c", "o", "h", "s", "S", "n")) {
  plot(y ~ x, type = tp, main = paste0("plot(*, type = \\"", tp, "\")"))
  if(tp == "S") {
    lines(x, y, type = "s", col = "red", lty = 2)
    mtext("lines(*, type = \"s\", ...)", col = "red", cex = 0.8)
  }
}
```

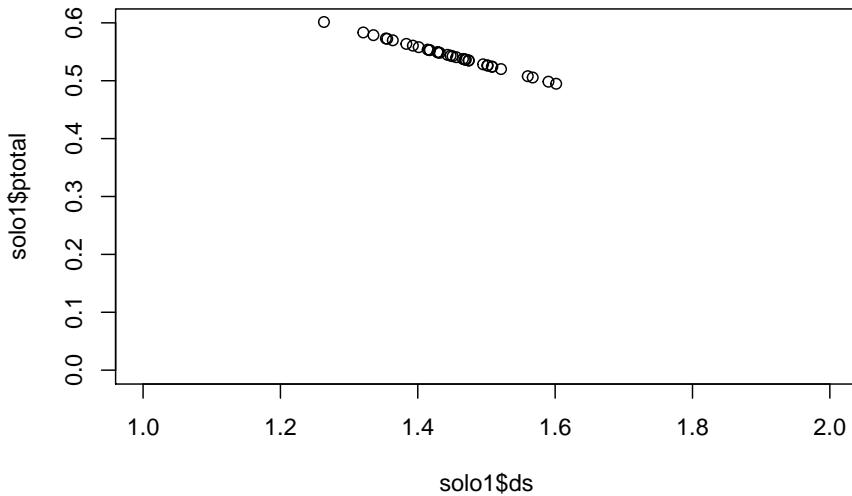


```
par(op)
```

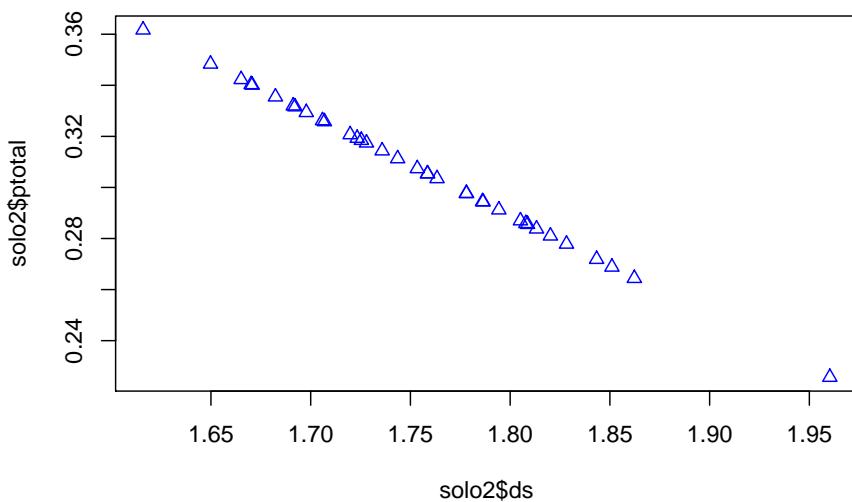
5.3.1 Mudando o padrão dos pontos pch=

Pode-se usar diferentes padrões para os pontos usando o argumento `pch=`. Diferentes tipos de símbolos são associados a diferentes números. Pode-se ainda usar caracteres como o símbolo desejado. Use a opção `pch =` para especificar símbolos a serem usados ao traçar pontos. Para os símbolos de 21 a 25, especifique a cor da borda (`col =`):

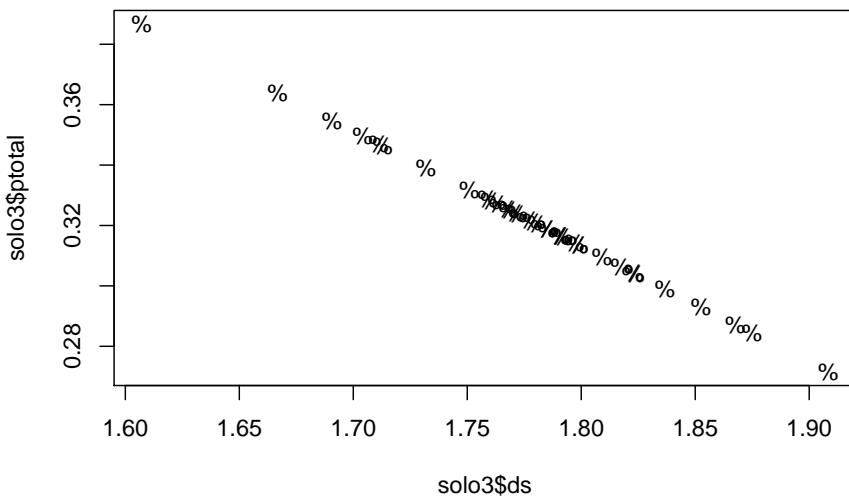
```
plot(solo1$ds,solo1$ptotal, pch=21, ylim = c(0,0.6), xlim = c(1,2))
```



```
plot(solo2$ds,solo2$ptotal,pch=2, col="blue")
```



```
plot(solo3$ds,solo3$ptotal,pch="%")
```

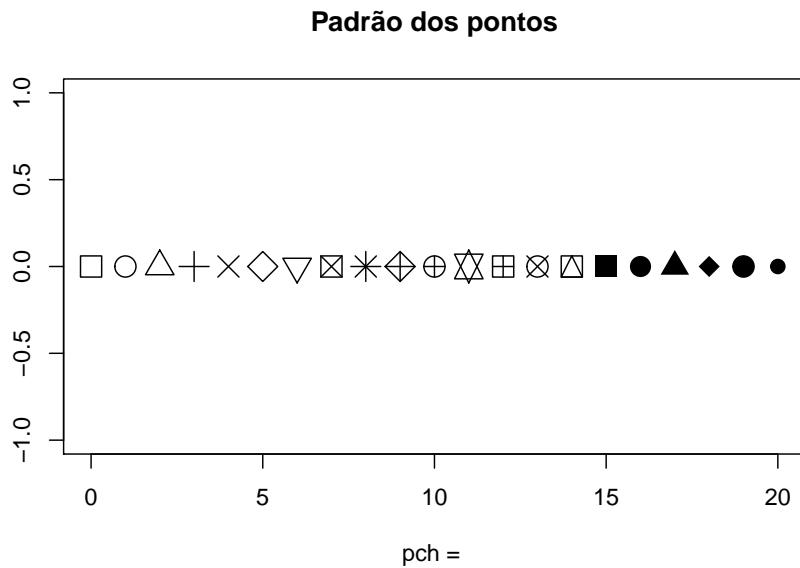


Neste exemplo acima note que foi adicionado o argumento `ylim` e `xlim` eles limitam os valores mínimos e máximos:

```
xlim=c(xmin, xmax) ylim=c(ymin, ymax)
```

Veja um exemplo do padrão dos pontos:

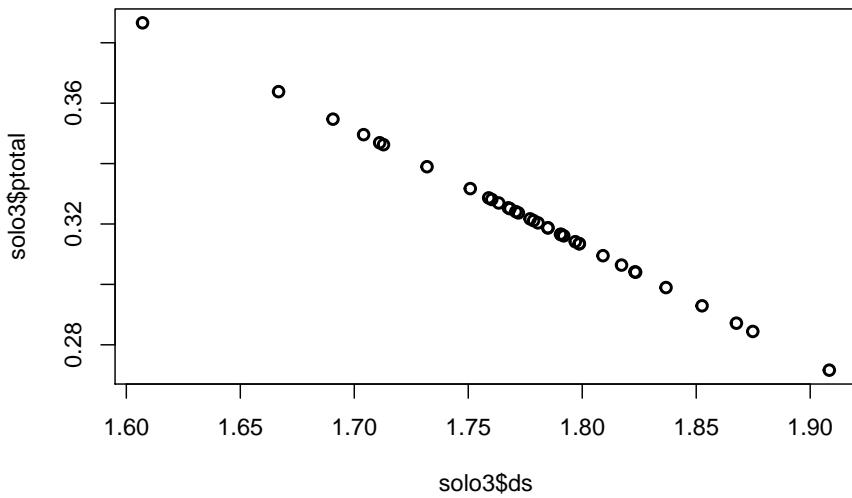
```
plot (0:20,                                     #coord. eixo X
      rep (0:21),                                #coord. eixo y
      pch = 0:20,                                 #padrão dos pontos variando
      cex = 2,                                    #tamanho dos pontos
      main = "Padrão dos pontos", #Titulo (note o \n)
      xlab = "pch = ",                            #texto do eixo de x
      ylab = "")                                 #texto do eixo de y
```



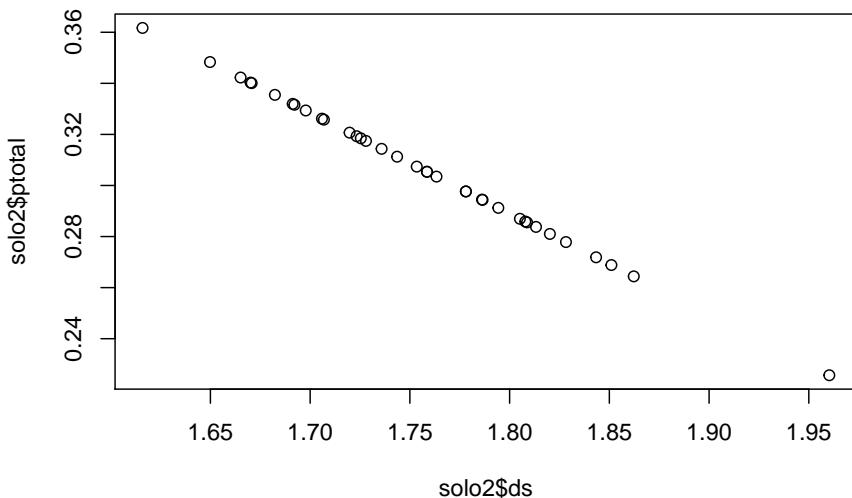
5.3.2 Mudando as linhas (*lwd* e *lty*)

Você pode alterar linhas usando as seguintes opções. Isso é particularmente útil para linhas de referência, eixos e linhas de ajuste. A largura das linhas pode ser mudada com o argumento *lwd*=, enquanto os estilos das linhas podem ser modificados com o argumento *lty*=:

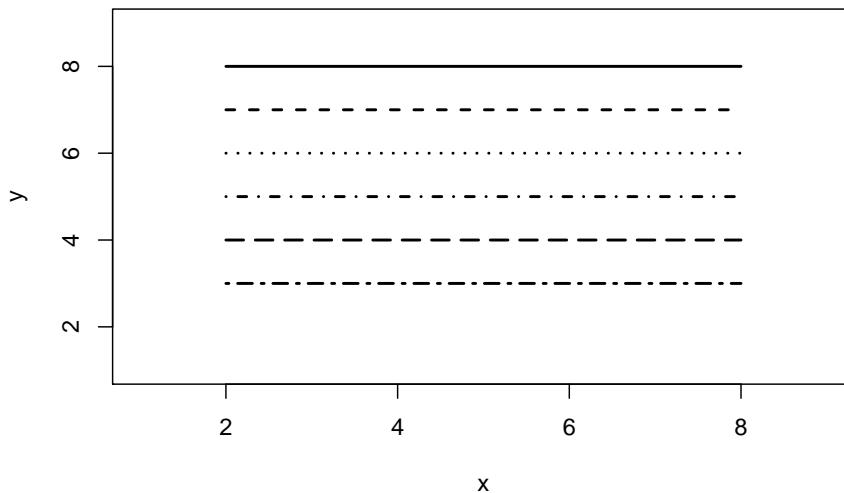
```
plot(solo3$ds,solo3$ptotal, lwd=2) # linha grossa
```



```
plot(solo2$ds,solo2$ptotal, lty=2) #linha interrompida
```



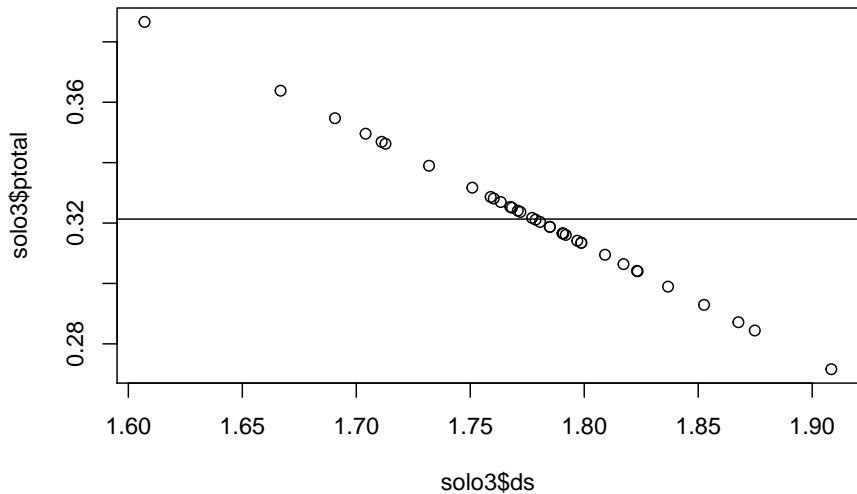
```
x <- 1:9
y <- 1:9
plot(x, y, type = "n")
lines(c(2, 8), c(8, 8), lwd = 2)
lines(c(2, 8), c(7, 7), lty = 2, lwd = 2)
lines(c(2, 8), c(6, 6), lty = 3, lwd = 2)
lines(c(2, 8), c(5, 5), lty = 4, lwd = 2)
lines(c(2, 8), c(4, 4), lty = 5, lwd = 2)
lines(c(2, 8), c(3, 3), lty = 6, lwd = 2)
```



5.3.3 Adicionando linhas a um grafico de pontos

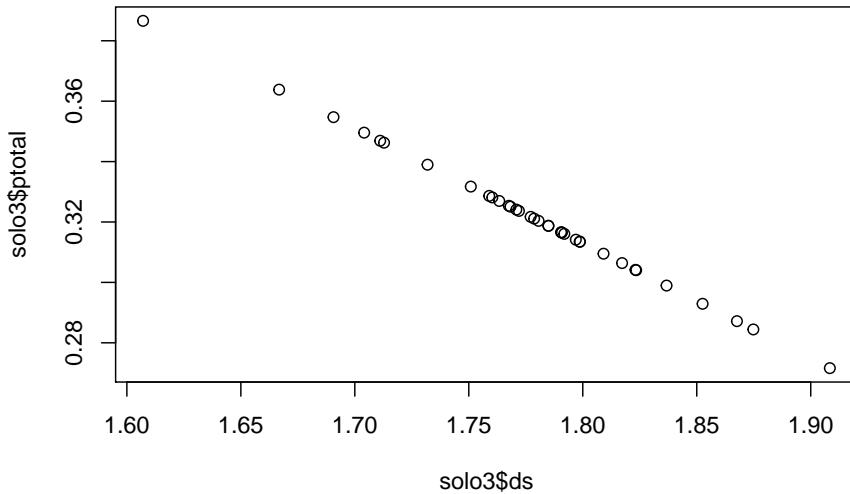
A função utilizada para inserir linhas é `abline()`. Vamos usar a função `abline` para inserir uma linha que mostra a média dos dados do eixo Y. o h é de linha horizontal. Fará uma linha na horizontal que passa pela média de y.

```
plot(solo3$ds,solo3$ptotal, abline(h=mean(solo3$ptotal)))
```

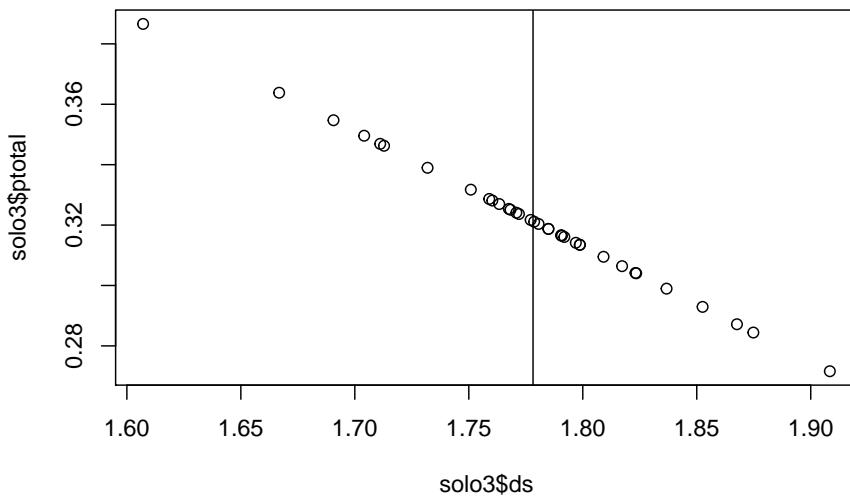


Para passar uma linha que passa pela média de x:

```
plot(solo3$ds,solo3$ptotal)
```

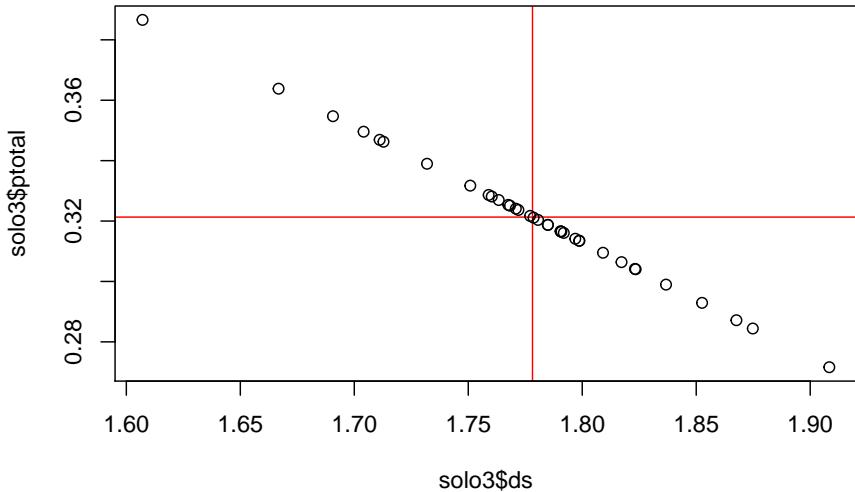


```
plot(solo3$ds,solo3$ptotal, abline(v=mean(solo3$ds))) ## o v é de vertical
```



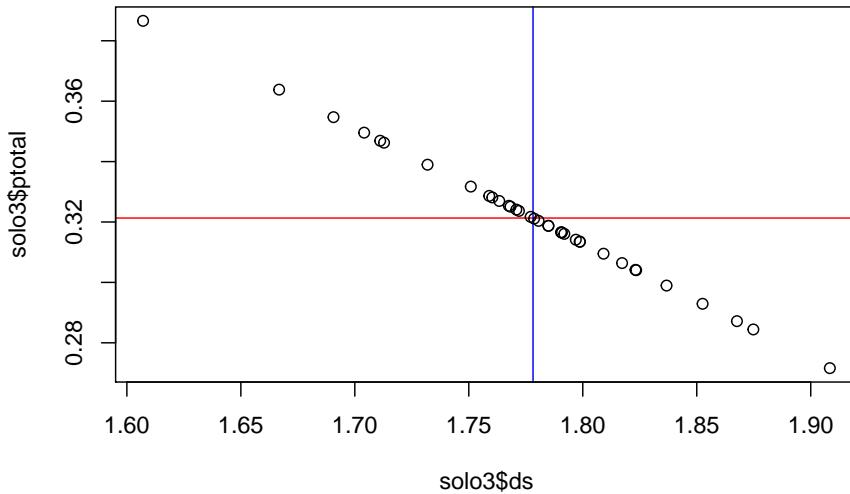
Também é possível inserir as duas linhas ao mesmo tempo:

```
plot(solo3$ds,solo3$ptotal, abline(h=mean(solo3$ptotal), v=mean(solo3$ds),col="red"))
```



Com cores diferentes:

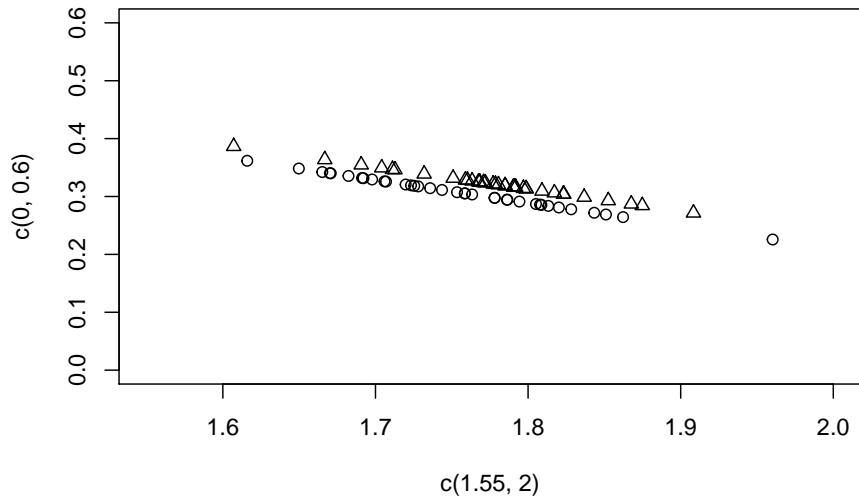
```
plot(solo3$ds,solo3$ptotal, abline(h=mean(solo3$ptotal), v=mean(solo3$ds),col=c(2,4)))
```



5.3.4 Definindo o intervalo dos eixos

Se você quiser preencher um mesmo gráfico com linhas e pontos que possuem diferentes amplitudes como nosso exemplo do solos, deve usar o argumento `type=n`. Com este argumento um gráfico em branco é criado.

```
plot(c(1.55,2),c(0,0.6),type='n')
points(solo3$ds,solo3$ptotal, pch=2)
points(solo2$ds,solo2$ptotal)
```

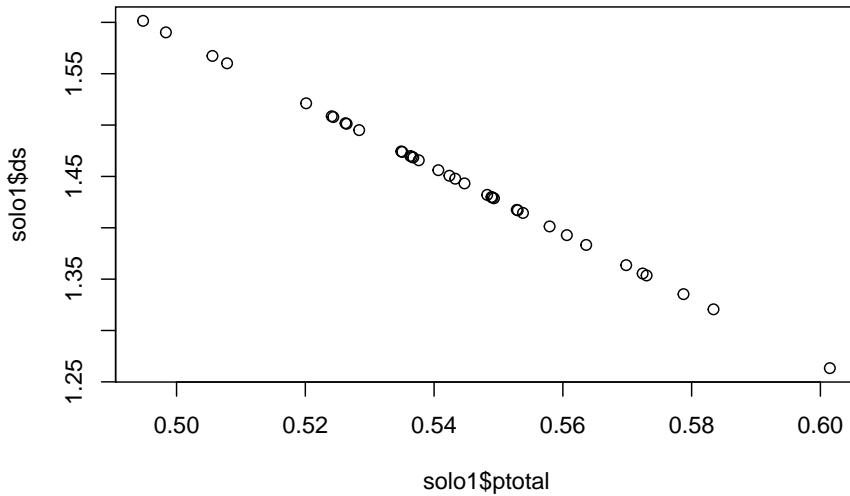


5.3.5 Personalizando os gráficos

Alguns parâmetros podem ser usados no intuito de personalizar um gráfico no R.

Exemplo:

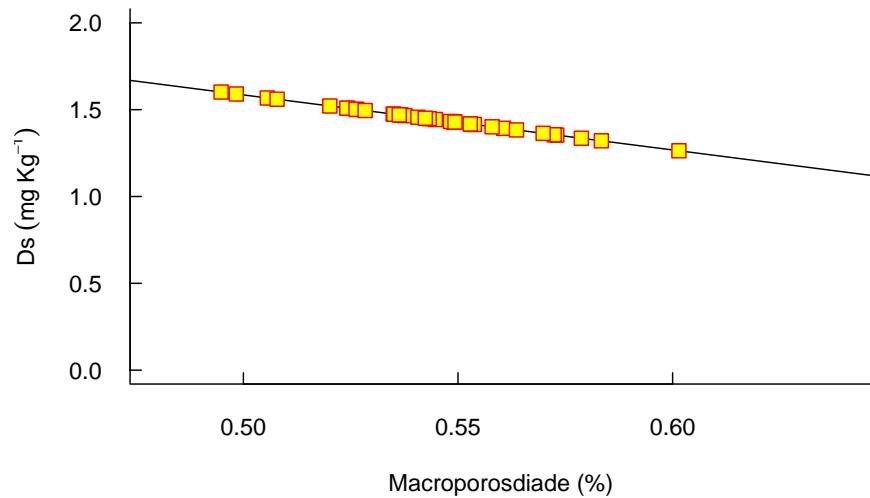
```
plot(solo1$ptotal,solo1$ds)
```



```

plot(solo1$ptotal,solo1$ds,           #plota ds e ptotal
      xlab="Macroporosdiade (%)",    #nomeia o eixo x
      ylab=expression(Ds~(mg~Kg^{-1})), #nomeia o eixo y
      main="Como personalizar um gráfico", #referente ao título
      xlim=c(0.48,0.64),             #limites do eixo x
      ylim=c(0,2), col="red",         #limites do eixo y
      pch=22,                         #padrão dos pontos
      bg="yellow",                    #cor de preenchimento
      tcl=0.4,                         #tamanho dos traços dos eixos
      las=1,                           #orientação do texto em y
      cex=1.5,                          #tamanho do objeto do ponto
      bty="l",                           #altera as bordas
      abline(lm(solo1$ds~solo1$ptotal))) #regressao dos pontos
  
```

Como personalizar um gráfico

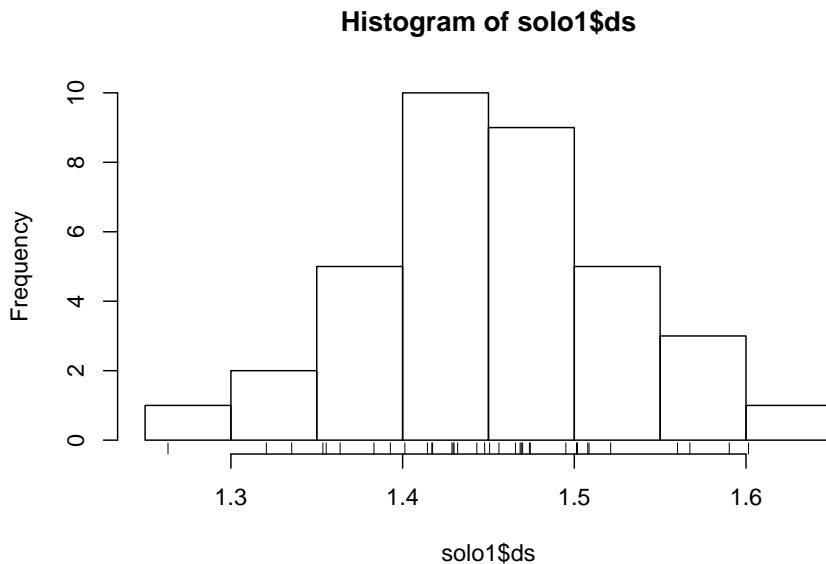


Veja o `demo(plotmath)` para saber mais sobre anotações em gráficos.

5.4 Histogramas

A função `hist()` produz um histograma dos dados informados em seu argumento enquanto a função `barplot()` produz um gráfico de barras:

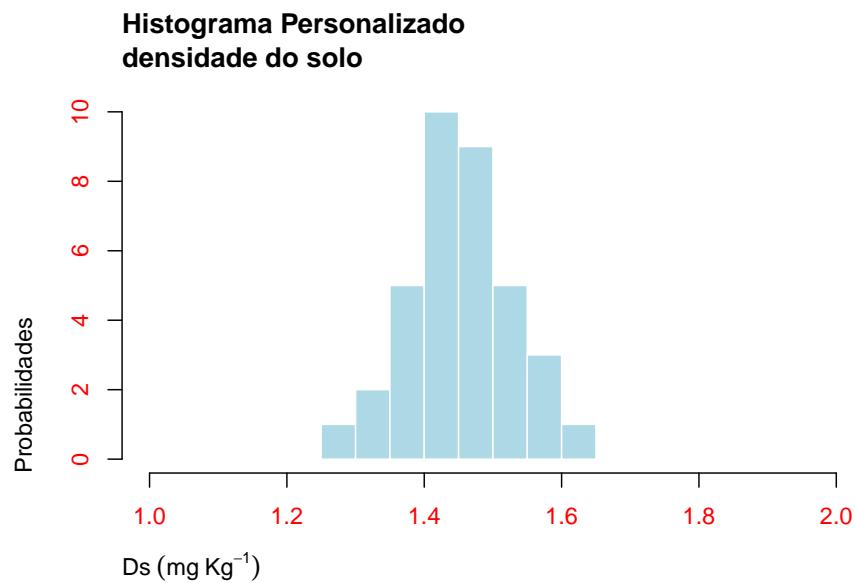
```
hist(solo1$ds)
rug(solo1$ds)
```



5.4.1 Personalizando gráficos

Os histogramas criados no R seguem um certo padrão (conhecido como parâmetros default) que podem ser alterados de acordo com a preferência do usuário. Você pode obter informações detalhadas desses parâmetros se usar os recursos de ajuda do R:

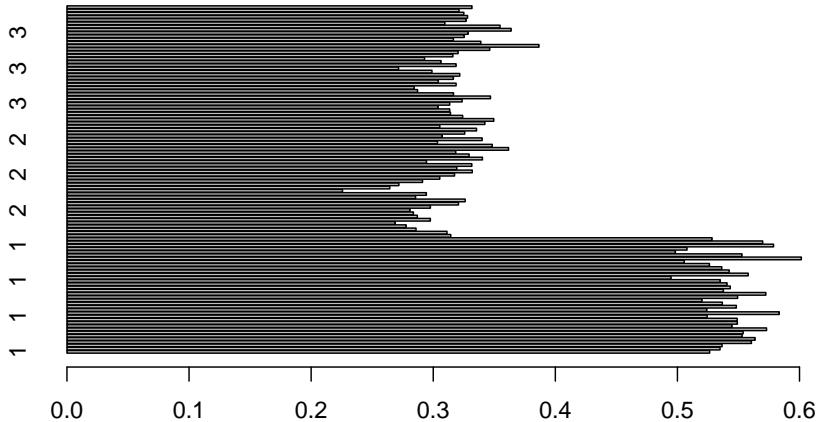
```
hist(solo1$ds, #histograma de ds
     main="Histograma Personalizado\ndensidade do solo",#título
     xlab=expression(Ds~(mg~Kg^{-1})), #texto do eixo das abscissas
     ylab="Probabilidades", #texto do eixo das ordenadas
     xlim=c(1,2), #limites do eixo de x
     ylim=c(0,10), #limites do eixo y
     col="lightblue", #cor das colunas
     border="white", #cor das bordas das colunas
     adj=0, #alinhamento dos textos 0, 0.5 e 1
     col.axis="red") #cor do texto nos eixos
```



5.5 Gráficos de Barras

Assemelha-se ao histograma, porém nesse caso os dados referem-se a categoria ou aos tratamentos:

```
barplot(solo$ptotal, names.arg=solo$z, horiz = T)
```



5.6 Boxplots

Dados de um experimento visando controle de pulgão (*Aphis gossypii Glover*) em cultura de pepino, instalado em *delineamento inteiramente casualizado* com 6 repetições. A resposta observada foi o número de pulgões após a aplicação de produtos indicados para seu controle.

```
dados <- read.table("https://www.dropbox.com/s/jjyo8dhyy0qt3ft/BanzattoQd3.2.1.txt?dl=1")
str(dados)
```

```
## 'data.frame':    30 obs. of  3 variables:
## $ trat   : Factor w/ 5 levels "Azinfos etilico",...: 5 1 3 4 2 5 1 3 4 2 ...
## $ rept   : int  1 1 1 1 1 2 2 2 2 2 ...
## $ pulgoes: int  2370 1282 562 173 193 1687 1527 321 127 71 ...
```

trat Fator de níveis nominais. Tratamento aplicado para controle do pulgão.

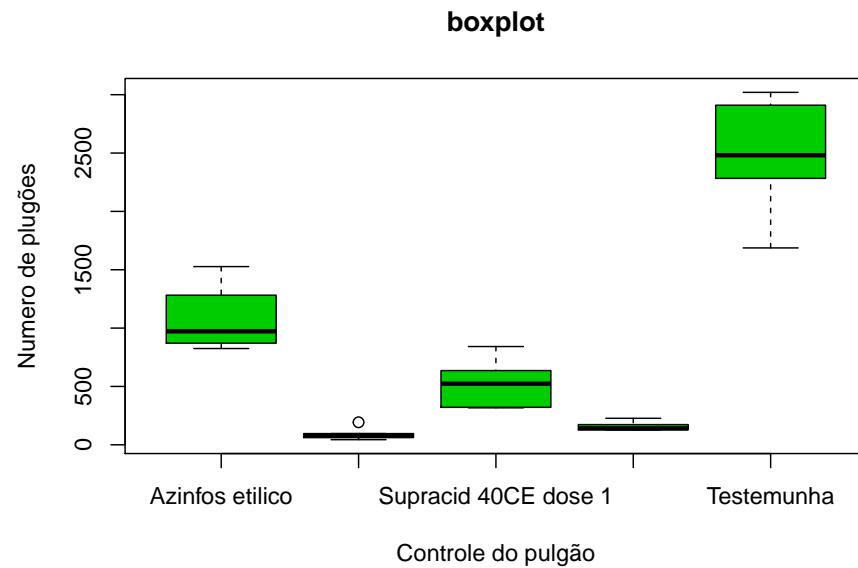
rept Número inteiro que identifica as repetições de cada tratamento.

pulgões Número de pulgões coletados 36 horas após a pulverização dos tratamentos.

Boxplots podem ser criados para variáveis individuais ou para variáveis por grupo. O formato é `boxplot (x , data =)`, em que `x` é uma fórmula e `data` = denota o quadro de dados que fornece os dados.

Um exemplo de uma fórmula é `y ~ group` onde um boxplot separado para a variável numérica é gerado para cada valor de `group`:

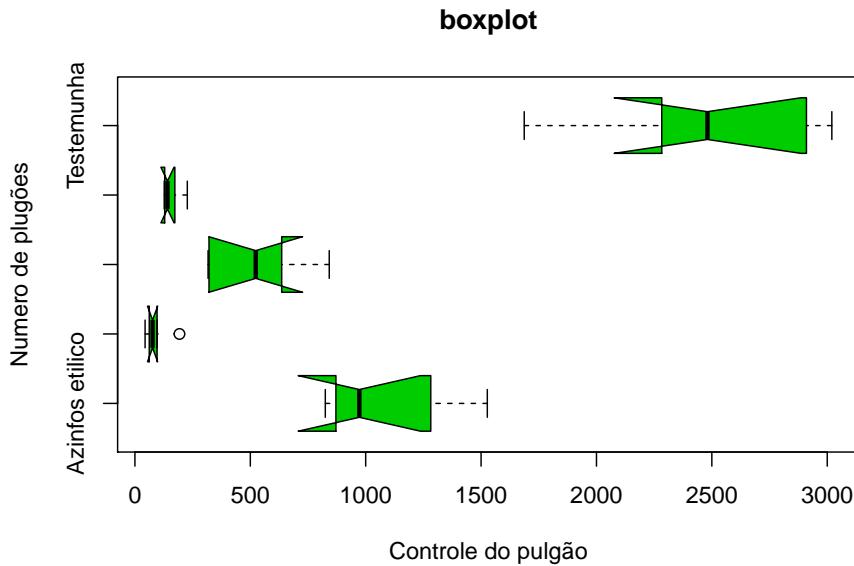
```
x11()
boxplot(pulgoes~trat,
         data = dados,
         main="boxplot",
         xlab="Controle do pulgão",
         ylab="Número de plugões",
         col=3)
```



Adicione `horizontal = TRUE` para inverter a orientação do eixo:

```
boxplot(pulgoes~trat,
        data = dados,
        main="boxplot",
        xlab="Controle do pulgão",
        ylab="Número de plugões",
        col=3, horizontal = T,
        notch=T)
```

```
## Warning in bxp(list(stats = structure(c(825, 871, 972.5, 1282, 1527, 44, :
## some
## notches went outside hinges ('box'): maybe set notch=FALSE
```



5.6.1 Boxplot com fatorial

Boxplot com 2 fatores, com caixas coloridas para facilitar a interpretação.

Efeito de Recipientes para duas Espécies de Eucalipto

Experimento em esquema fatorial 3x2 para estudar o efeito de 3 tipos de recipientes para a produção de mudas de duas espécies de Eucalipto. O experimento foi instalado em delineamento inteiramente casualizado.

recipie São os níveis de recipiente estudados: - SPP - saco plástico pequeno; - SPG - saco plástico grande; e - Lam - laminado.

especie São as espécies de Eucalipto: *Eucalyptus citriodora* e *Eucalyptus grandis*

rept Identifica as repetições de cada combinação dos fatores recipiente e espécie.

alt Altura das mudas aos 80 dias de idade (cm).

Baixar dados via web:

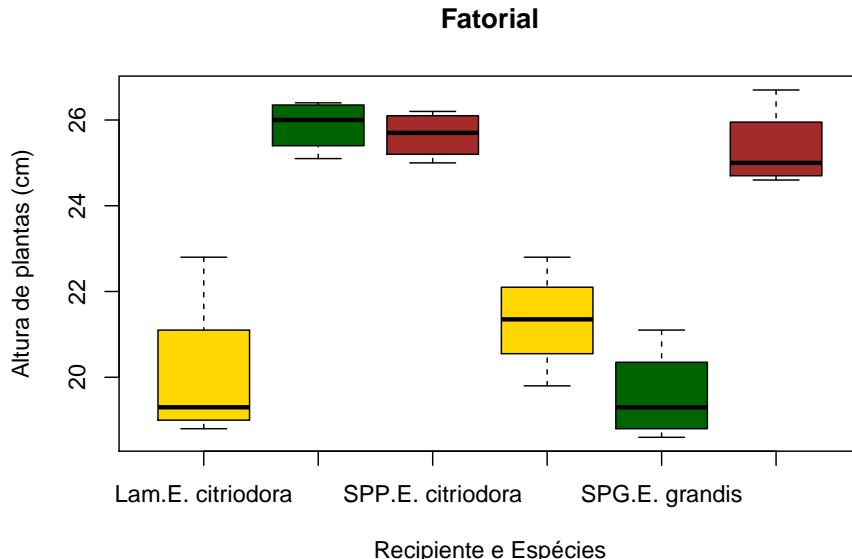
```
fat <- read.table("https://www.dropbox.com/s/sahc5n80rlkcf4/BanzattoQd5.2.4.txt?dl=1")
str(fat)
```

```
## 'data.frame':    24 obs. of  4 variables:
## $ recipie: Factor w/ 3 levels "Lam","SPG","SPP": 3 3 2 2 1 1 3 3 2 2 ...
## $ especie: Factor w/ 2 levels "E. citriodora",...: 1 2 1 2 1 2 1 2 1 2 ...
```

```
## $ rept   : int  1 1 1 1 1 1 2 2 2 2 ...
## $ alt    : num  26.2 24.8 25.7 19.6 22.8 19.8 26 24.6 26.3 21.1 ...
```

Gerar o gráfico boxplot com o comando abaixo:

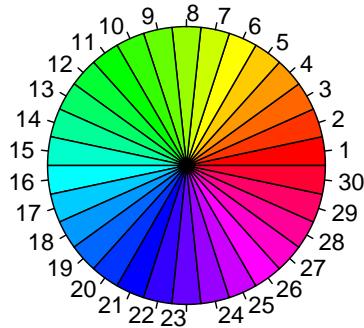
```
boxplot(fat$alt~fat$recipie*especie, data=fat, notch=F,
        col=c("gold","darkgreen","brown")),
        main="Fatorial", xlab="Recipiente e Espécies",
        ylab="Altura de plantas (cm)")
```



5.7 Cores

Gráficos em preto e branco são bons na maioria dos casos, mas cores podem ser mudadas usando `col="red"` (escrevendo o nome da cor) ou `col=2` (usando números). O comando abaixo mostra os números que especificam algumas cores:

```
pie(rep(1,30),col=rainbow(30))
```



Veja sua tabela de cores executando o script: paletedecores.R.

Podemos também criar cores personalizadas usando a função do `rgb()`, que recebe como argumentos as quantidades de vermelho (*red*), verde (*green*) e azul (*blue*) e, opcionalmente, o grau de opacidade (*alpha*). Os valores devem ser números reais entre 0 e 1.

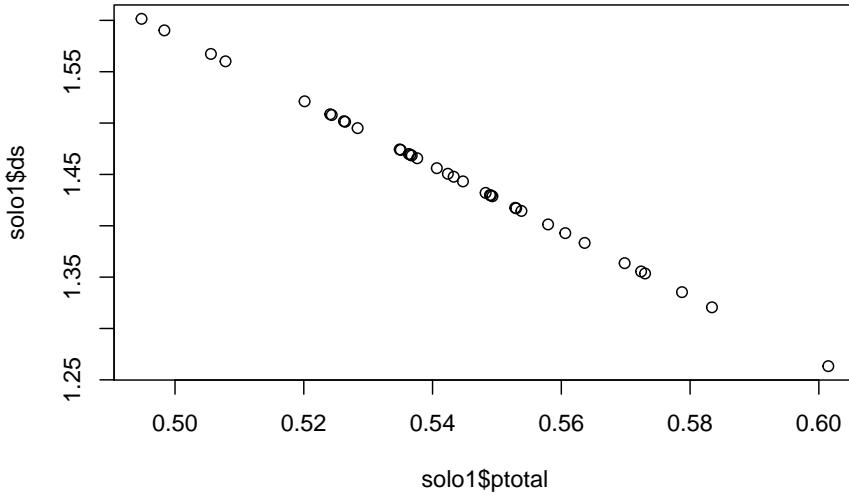
Exemplos:

```
goiaba <- rgb(0.94, 0.41, 0.40)
goiaba.semitrans <- rgb(0.94, 0.41, 0.40, alpha = 0.5)
vitamina <- rgb(red = c(0.87, 0.70), green = c(0.83, 0.77),
blue = c(0.71, 0.30), names = c("leite", "abacate"))
```

5.8 Interagindo com a Janela gráfica

Poderemos com o mouse marcar a ponte desejada usando a função `identify()`:

```
plot(solo1$ds~solo1$ptotal)
identify(solo1$ds,n=1)
```



```
## integer(0)
```

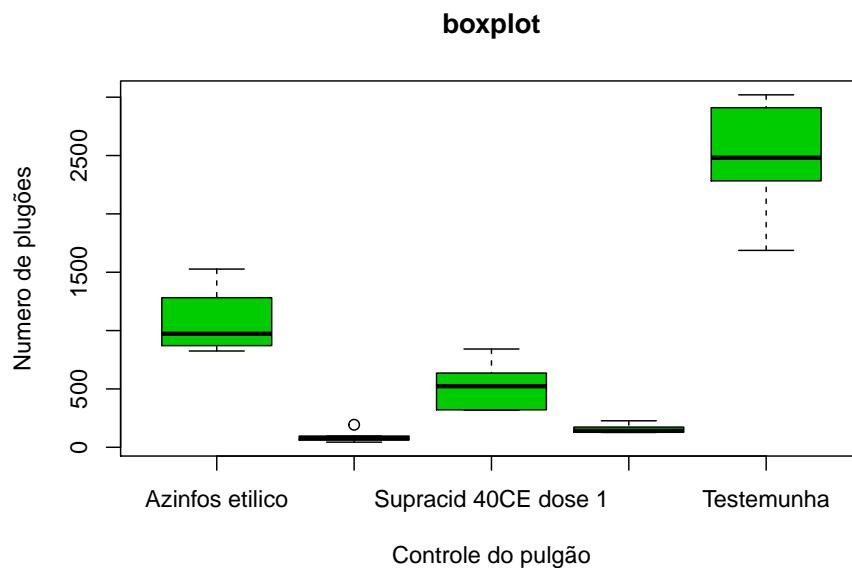
5.9 Texto e tamanho do símbolo

As seguintes opções podem ser usadas para controlar o tamanho do texto e do símbolo em gráficos.

`cex` número que indica o valor pelo qual o texto e os símbolos de plotagem devem ser dimensionados em relação ao padrão. $1 = \text{padrão}$, $1,5$ é 50% maior, $0,5$ é 50% menor, etc.

5.10 Visualizar vários gráficos

```
x11()
boxplot(pulgoes~trat,                      #formula do boxplot
         data = dados,                      #conjunto de dados
         main="boxplot",                   #título
         xlab="Controle do pulgão",       #texto do eixo x
         ylab="Número de plugões",        #texto do eixo y
         col=3,                           #cor verde
         notch=F)                        #teste para mediana
```

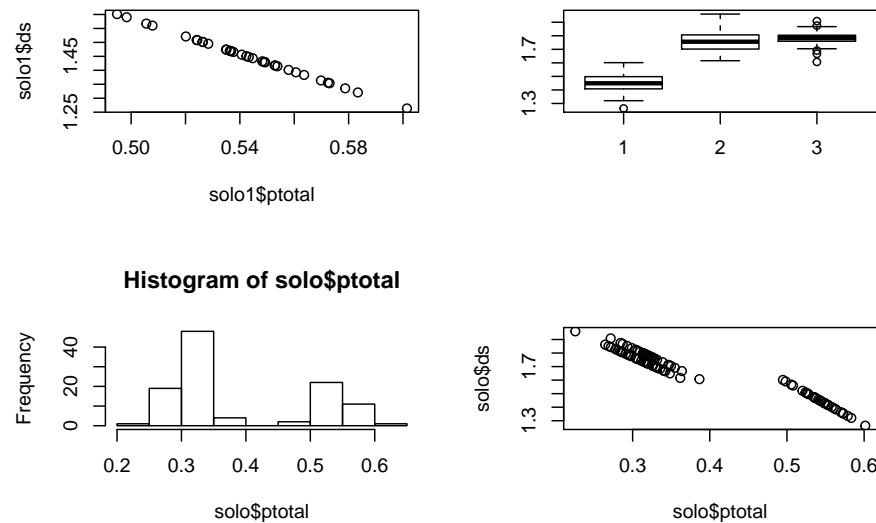


5.10.1 Varios gráficos na mesma janela gráfica

Você pode dar instruções para o programa mostrar diversos gráficos pequenos em uma mesma janela ao invés de um apenas. Para isto use a função `par()`.

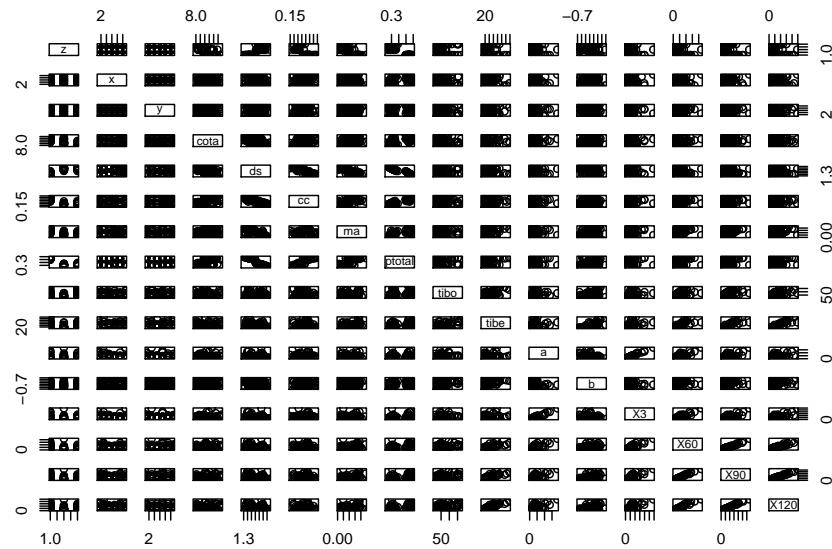
Exemplo 1

```
par(mfrow = c(2,2)) #2 linhas e 2 colunas
plot(solo1$ptotal,solo1$ds)
boxplot(solo1$ds,solo2$ds, solo3$ds)
hist(solo$ptotal)
plot(solo$ptotal,solo$ds)
```

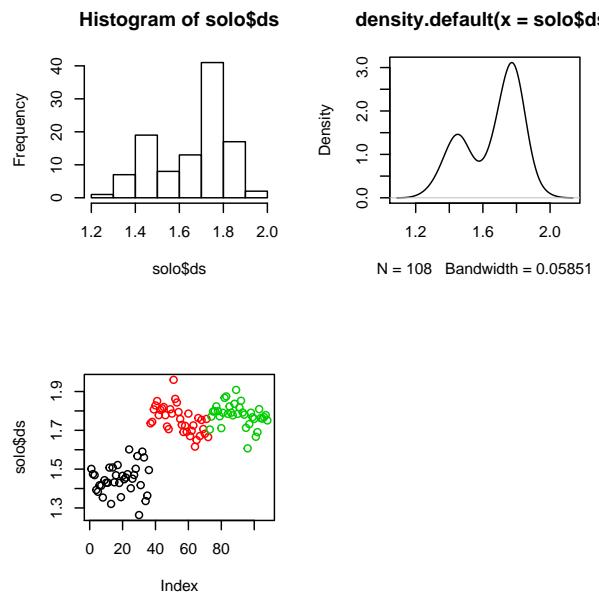


Exemplo 2

```
par(mfrow = c(2,3))  
pairs(solo)
```



```
hist(solo$ds)
plot(solo$ds, col=solo$z)
plot(density(solo$ds))
```



5.11 Salvando gráficos

Você pode salvar o gráfico em vários formatos no menu. *Arquivo -> Salvar como.*

Você também pode salvar o gráfico via código usando uma das seguintes funções:

```
pdf (file = "meugráfico.pdf") #ficheiro PDF
win.metafile ("meu grafico.wmf") #metarquivo do windows
png ("meu grafico.png") #arquivo png
jpeg ("meu grafico.jpg") #arquivo jpeg
bmp ("meu grafico.bmp") #arquivo bmp
postscript ("meu grafico.ps") #arquivo postscript
```


Chapter 6

Gráficos com ggplot2

Existem muitas maneiras de fazer gráficos com o R, cada um com suas vantagens e desvantagens. O foco aqui está no pacote `ggplot2`, que é baseado na *Grammar of Graphics* (Gramática dos Gráficos) para descrever os gráficos de dados.

Utilize o código abaixo para instalar o pacote `ggplot2`:

```
install.packages("ggplot2")
```

Sempre carregue o pacote antes de utilizá-lo:

```
library(ggplot2)
```

Utilizaremos o banco de dados: `dadosfisio`

Baixar os dados:

```
fisio <- read.csv2("https://www.dropbox.com/s/zg7fyg1iewtji49/dadosfisio.csv?dl=1")
```

Veja as primeiras linhas:

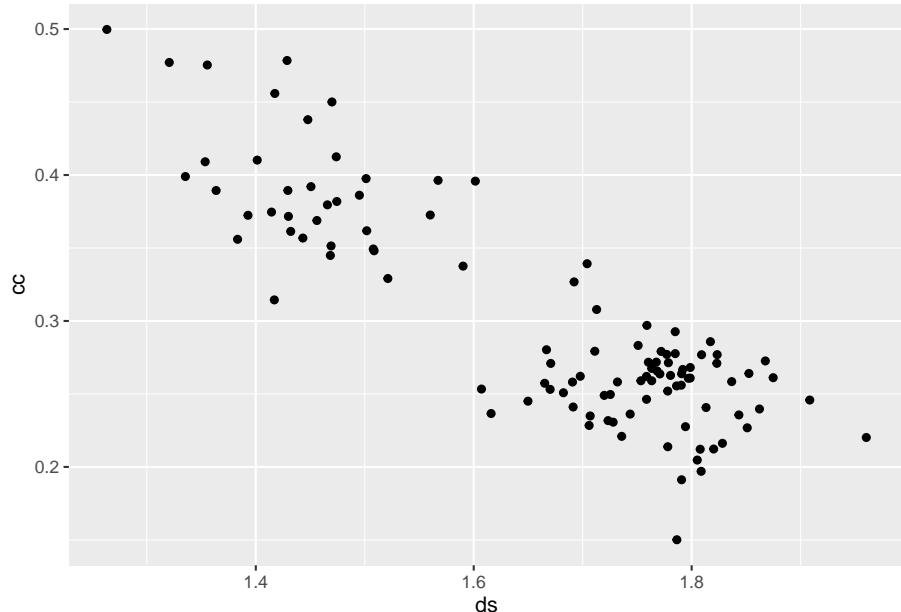
```
head(fisio)
```

```
##   z x  y cota      ds      cc      ma     ptotal     tibo     tibe      a
## 1 1 1  1 9.15 1.501258 0.3975615 0.1288555 0.5264170 46.08  26.78 926.3955
## 2 1 1  3 8.95 1.474362 0.3818767 0.1530250 0.5349017 19.20  26.10 383.8130
## 3 1 1  5 8.78 1.469118 0.3514075 0.1851484 0.5365559 172.80 113.92 275.3272
## 4 1 1  7 8.59 1.392845 0.3724094 0.1882073 0.5606167 96.00  74.83 1206.7585
```

```
## 5 1 1 9 8.48 1.383309 0.3559554 0.2076696 0.5636250 30.72 37.20 151.4032
## 6 1 1 11 8.41 1.417010 0.3144429 0.2385509 0.5529937 151.32 124.52 368.7992
##       b      X3      X60      X90      X120
## 1 -0.5290035 518.0810 153.24778 106.20581 73.60416
## 2 -0.4176008 242.5903 92.74121 69.43243 51.98188
## 3 -0.1307566 238.4857 176.48417 161.19222 147.22529
## 4 -0.3764298 798.0272 335.41915 258.38731 199.04648
## 5 -0.2270424 117.9800 69.94649 59.76121 51.05906
## 6 -0.1549382 311.0754 217.73464 195.56291 175.64890
```

O código abaixo é um exemplo de um gráfico bem simples, construído a partir das duas principais camadas. O eixo y representa a densidade do solo e ao eixo x a variável capacidade de campo:

```
ggplot(data = fisio, aes(x = ds, y = cc)) +
  geom_point()
```

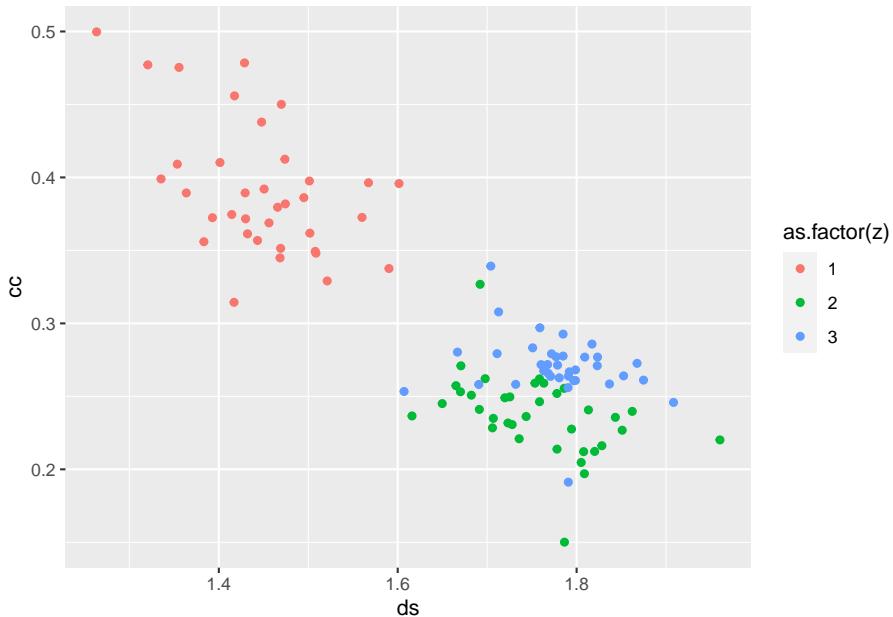


Aqui, essas formas geométricas são pontos, selecionados pela função `geom_point()`, gerando, assim, um gráfico de dispersão.

A função `aes()` vem da palavra *Aesthetics* define a relação entre os dados e cada aspecto visual do gráfico, como qual variável será representada no eixo x, qual será representada no eixo y, a cor e o tamanho dos componentes com a função `colour`.

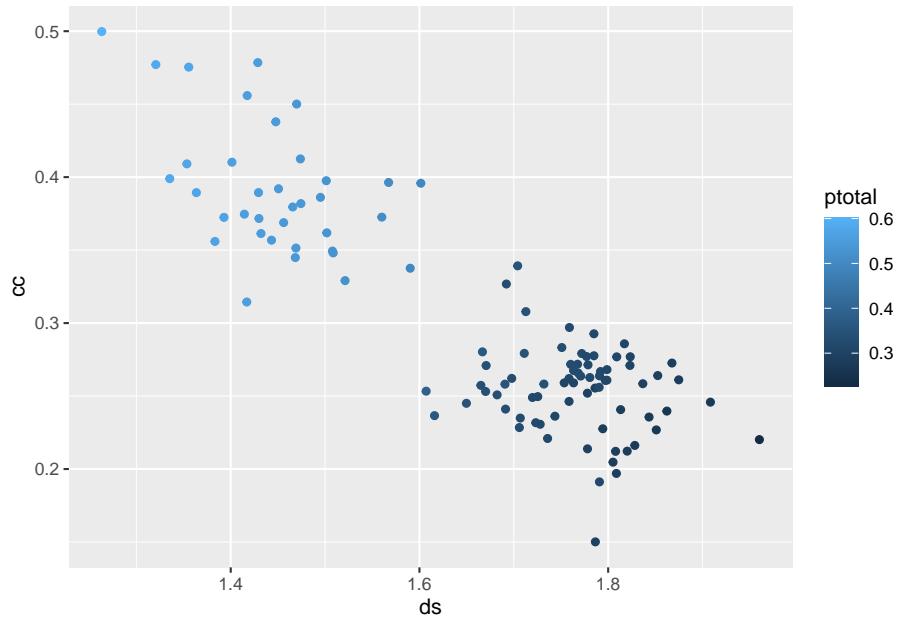
Outro aspecto que pode ser mapeado nesse gráfico é a cor dos pontos:

```
ggplot(data = fisio, aes(x = ds, y = cc, colour = as.factor(z))) +
  geom_point()
```



Agora, a variável z (classe de solo) foi mapeada a cor dos pontos, sendo que pontos vermelhos correspondem ao Nitossolo (valor 1) e pontos azuis e verdes, os Latossolos. Observe que inserimos a variável z como um fator, pois temos interesse apenas nos valores “1”, “2” e “3”. No entanto, também podemos mapear uma variável contínua, a cor dos pontos:

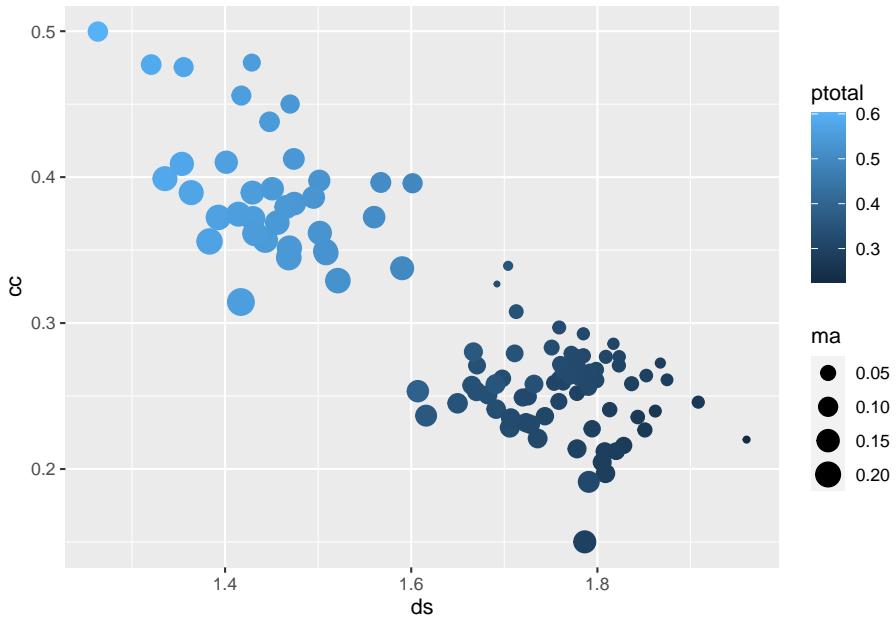
```
ggplot(data = fisio, aes(x = ds, y = cc, colour = ptotal)) +
  geom_point()
```



A porosidade do solo (ptotal), é representado pela tonalidade da cor azul.

Também podemos mapear o tamanho dos pontos a uma variável de interesse:

```
ggplot(data = fisio, aes(x = ds, y = cc, colour = ptotal, size = ma)) +  
  geom_point()
```

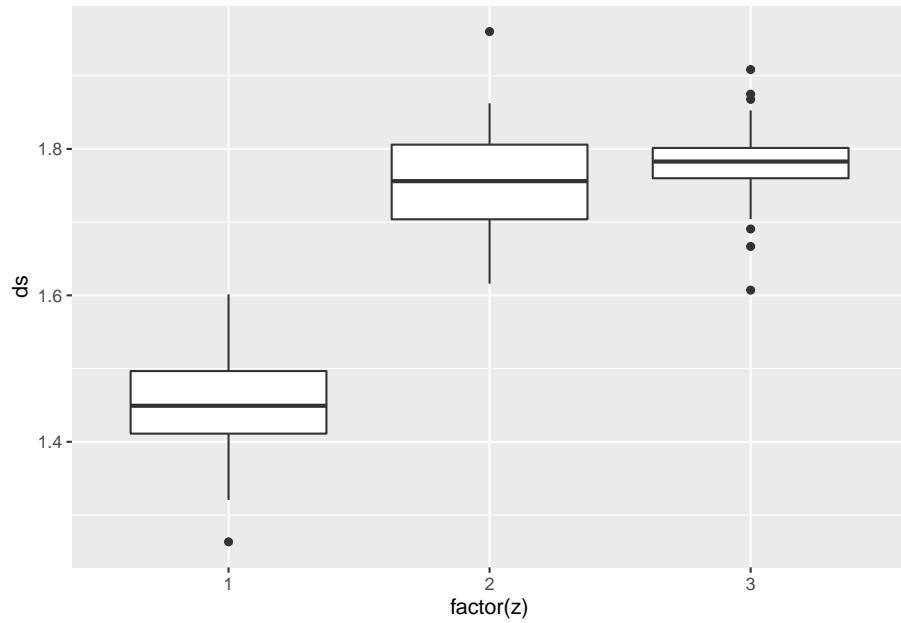


Outros `geoms` bastante utilizados:

- `geom_line`: para retas definidas por pares (x,y)
- `geom_abline`: para retas definidas por um intercepto e uma inclinação
- `geom_hline`: para retas horizontais
- `geom_boxplot`: para boxplots
- `geom_histogram`: para histogramas
- `geom_density`: para densidades
- `geom_area`: para áreas
- `geom_bar`: para barras

Veja a seguir como é fácil gerar diversos gráficos diferentes utilizando a mesma estrutura do gráfico de dispersão acima:

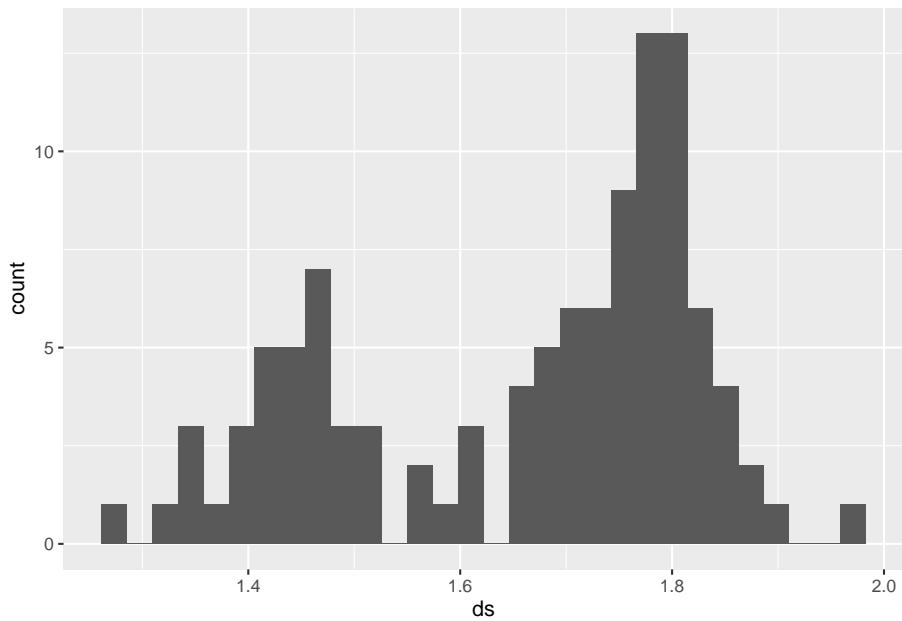
```
ggplot(data = fisio, aes(x = factor(z), y = ds)) +
  geom_boxplot()
```



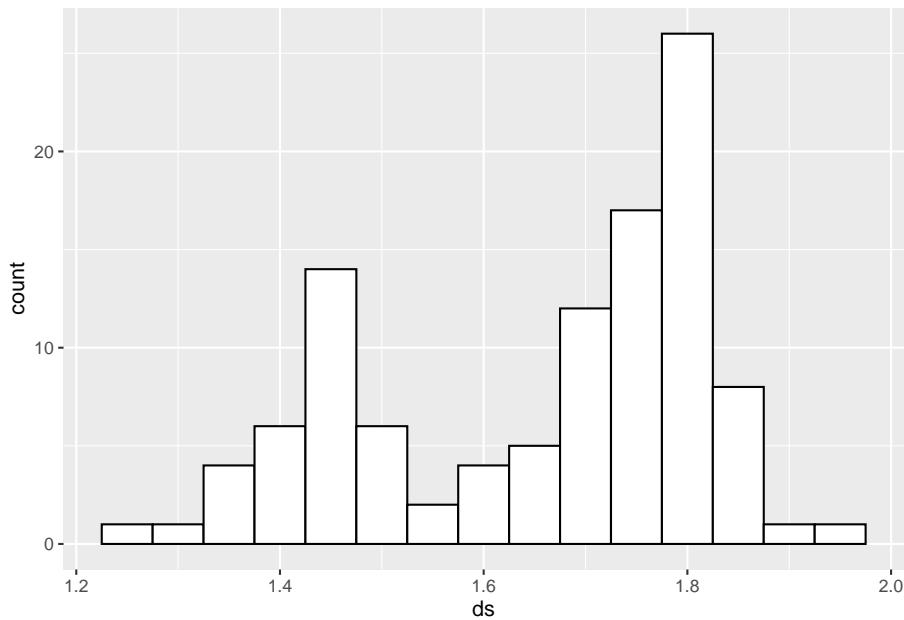
```
gra <- ggplot(data = fisio, aes(x = ds))
```

```
gra + geom_histogram()
```

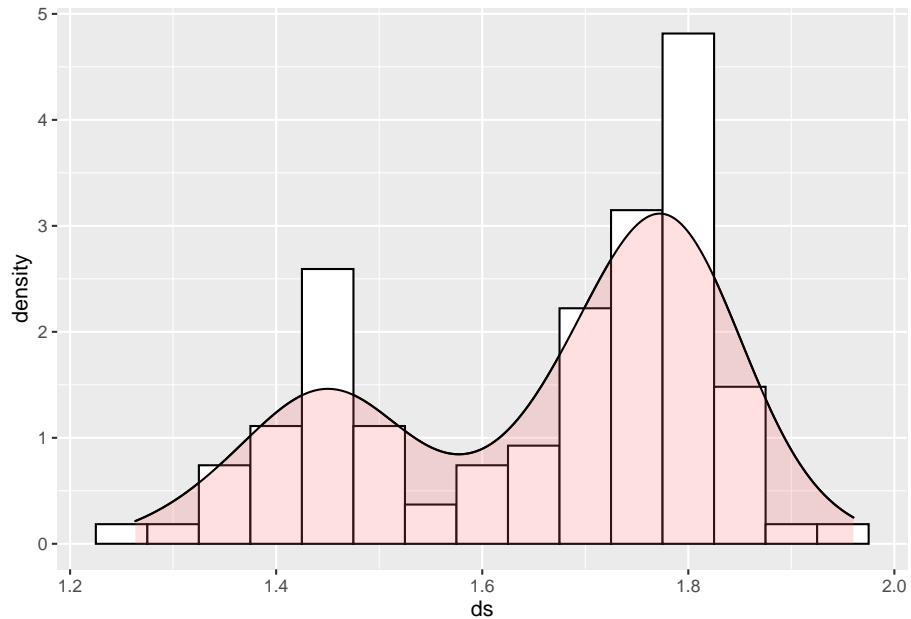
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
gra + geom_histogram(binwidth=.05, colour="black", fill="white")
```



```
gra + geom_density() +
  geom_histogram(aes(y=..density..), binwidth=.05,
    colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666")
```



Exemplo Baixar dados via web:

```
dados <- read.table("https://www.dropbox.com/s/9woiye3ce9twp78/BanzattoQd4.5.2.txt?dl=1")
```

Criar gráficos:

```
bar <- ggplot(data = dados, aes(y = peso, x = promalin, fill = factor(promalin)))
```

Nestes exemplos, a altura da barra representará o valor em uma coluna do quadro de dados. Isso é feito usando `stat="identity"` em vez do padrão `stat="bin"`:

```
bar + geom_bar(stat="identity")
```

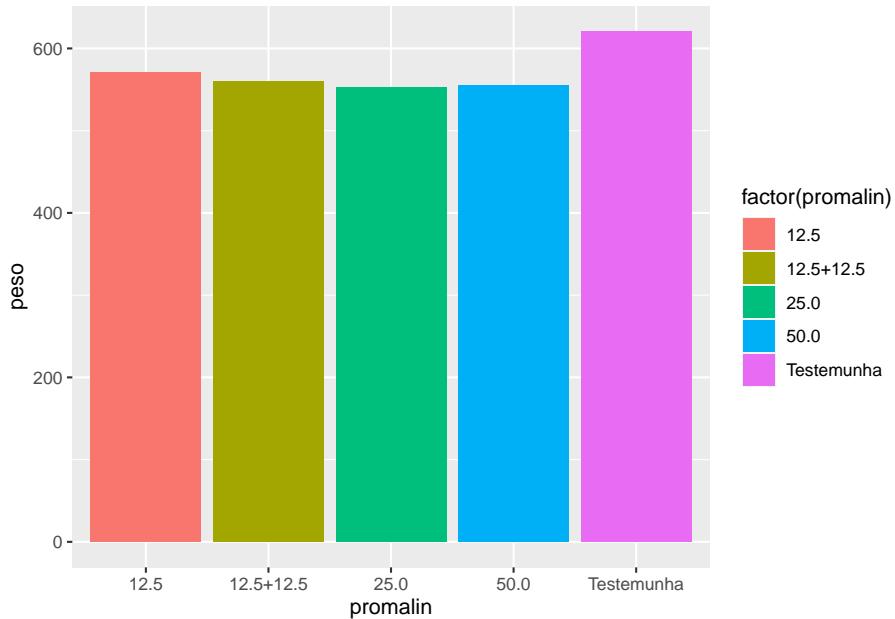
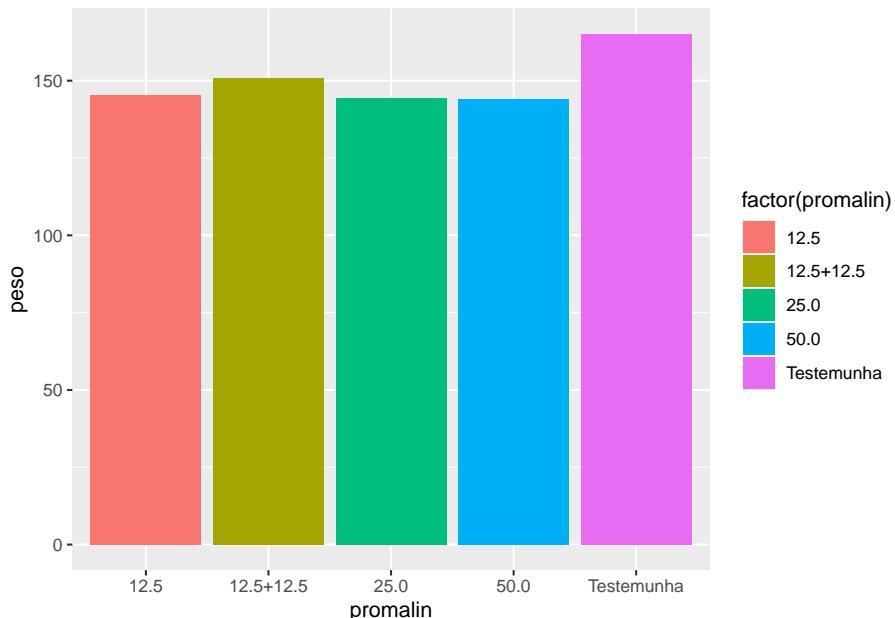


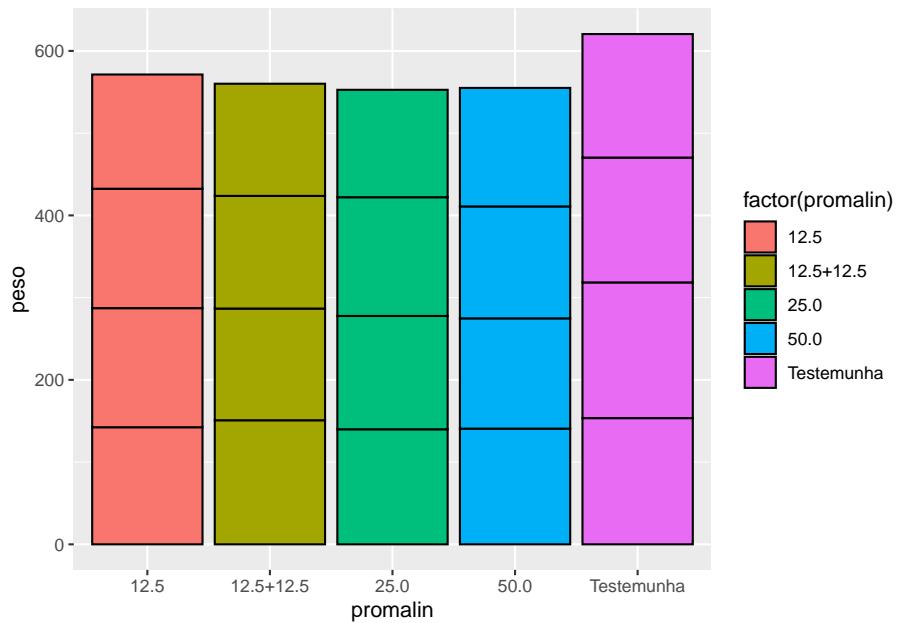
Gráfico de barras agrupados:

```
bar + geom_bar(stat="identity", position=position_dodge())
```



Empilhado:

```
bar + geom_bar(stat="identity", colour ="black")
```



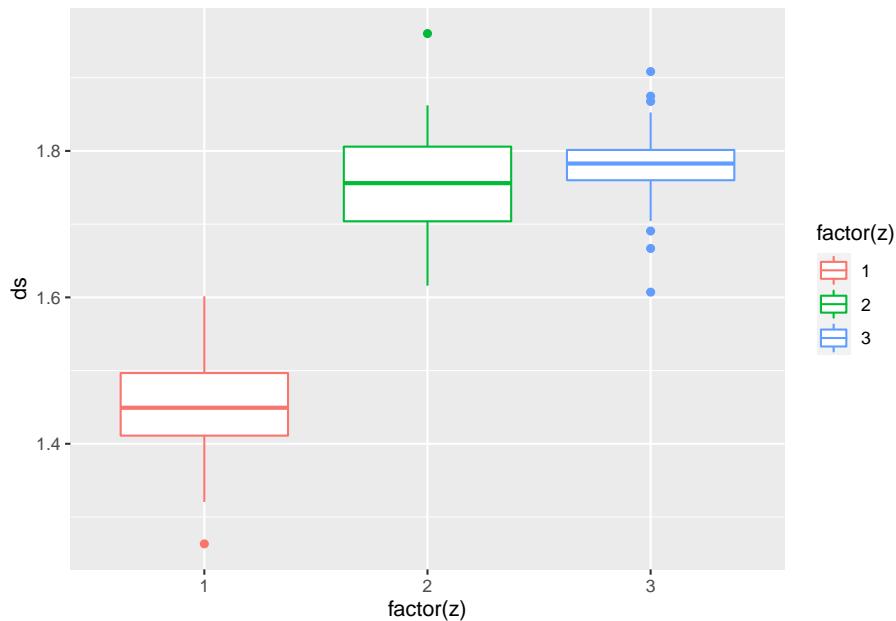
6.1 Personalizando os gráficos

6.1.1 Cores

O aspecto `colour` do boxplot, muda a cor do contorno. Para mudar o preenchimento, basta usar o `fill`.

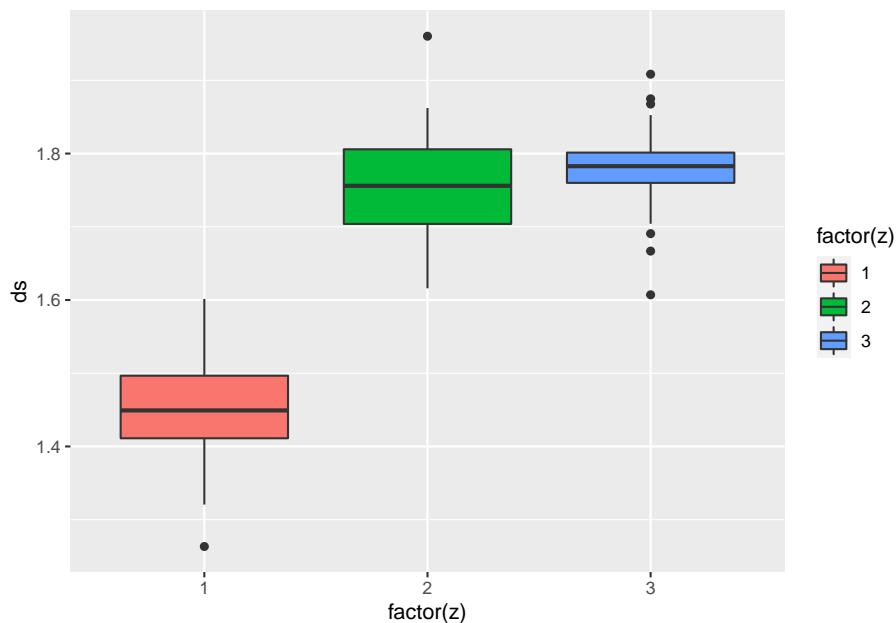
Usando `colour`:

```
ggplot(data = fisio, aes(x = factor(z), y = ds, colour = factor(z))) +
  geom_boxplot()
```



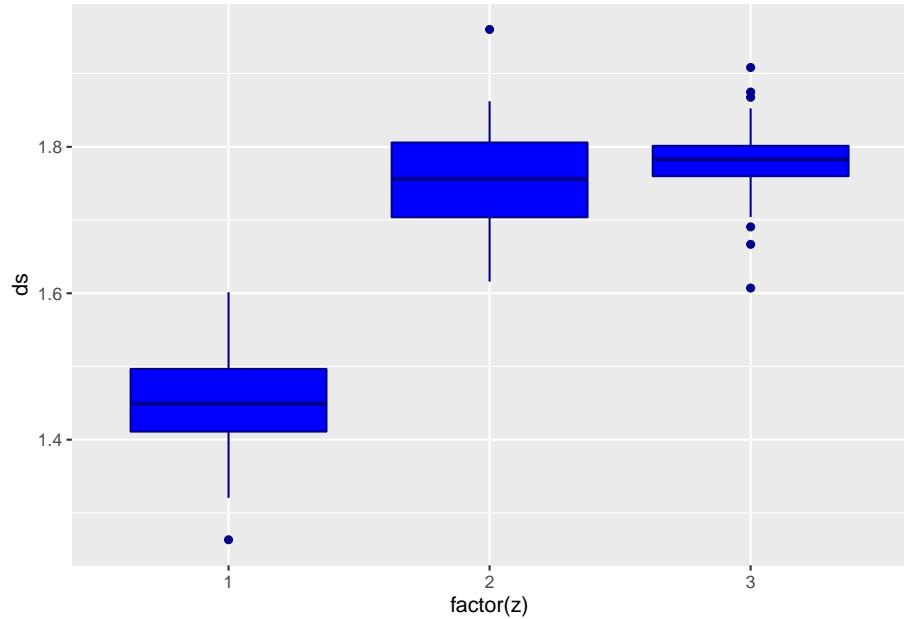
Usando `fill`:

```
ggplot(data = fisio, aes(x = factor(z), y = ds, fill = factor(z))) +  
  geom_boxplot()
```



Mude a cor dos objetos sem atribuir a uma variável. Para isso, observe que os aspectos `colour` e `fill` são especificados fora do `aes()`:

```
ggplot(data = fisio, aes(x = factor(z), y = ds)) +
  geom_boxplot(colour = "darkblue", fill= "blue")
```



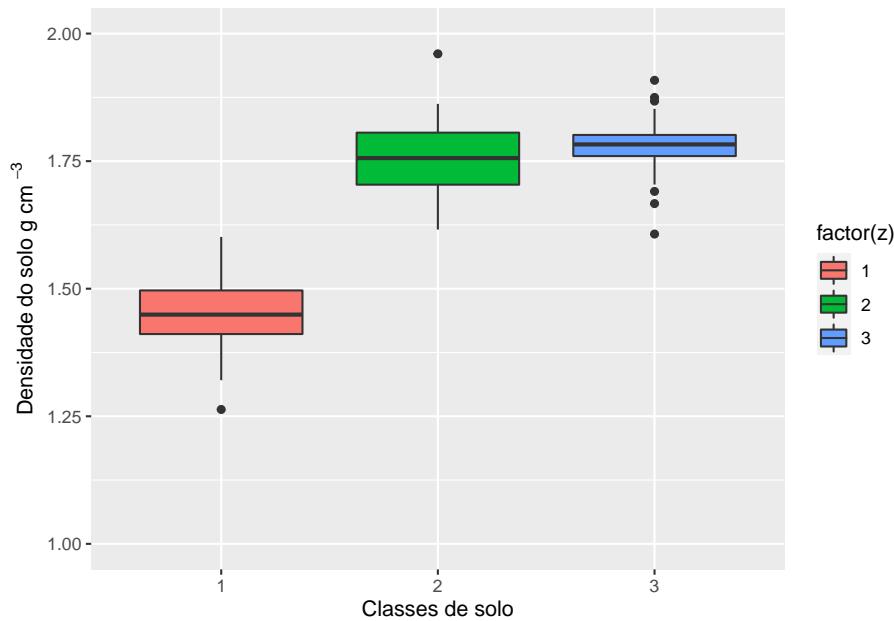
6.1.2 Eixos

Para alterar os rótulos dos eixos acrescentamos as funções `xlab()` ou `ylab()`:

```
box <- ggplot(data = fisio, aes(x = factor(z), y = ds, fill = factor(z))) +
  geom_boxplot() +
  xlab("Classes de solo") +
  ylab(expression(paste(Densidade~do~solo, " g cm ^{-3} )))
```

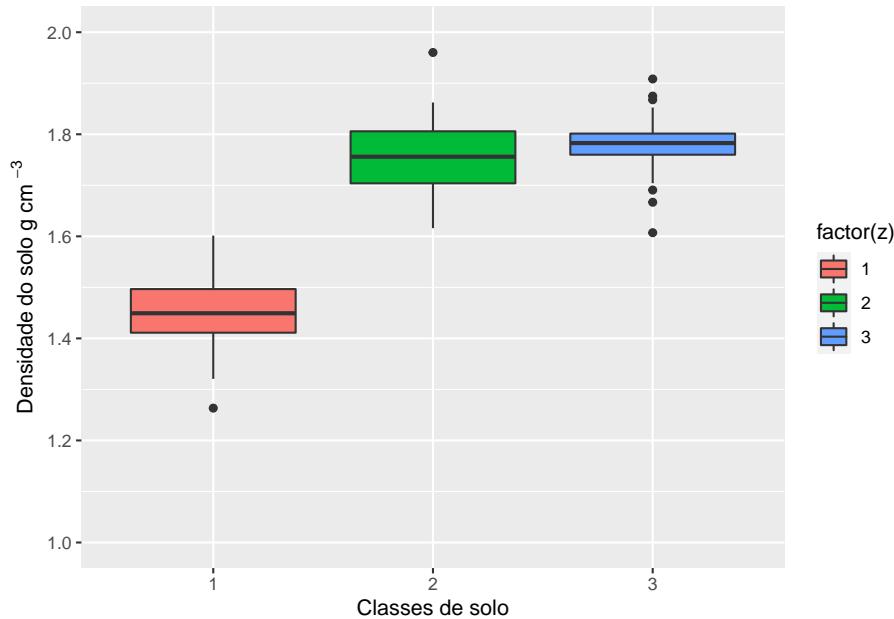
Alterar os limites dos gráficos usamos as funções `xlim()` e `ylim()`:

```
box + ylim (c(1.0,2.0))
```



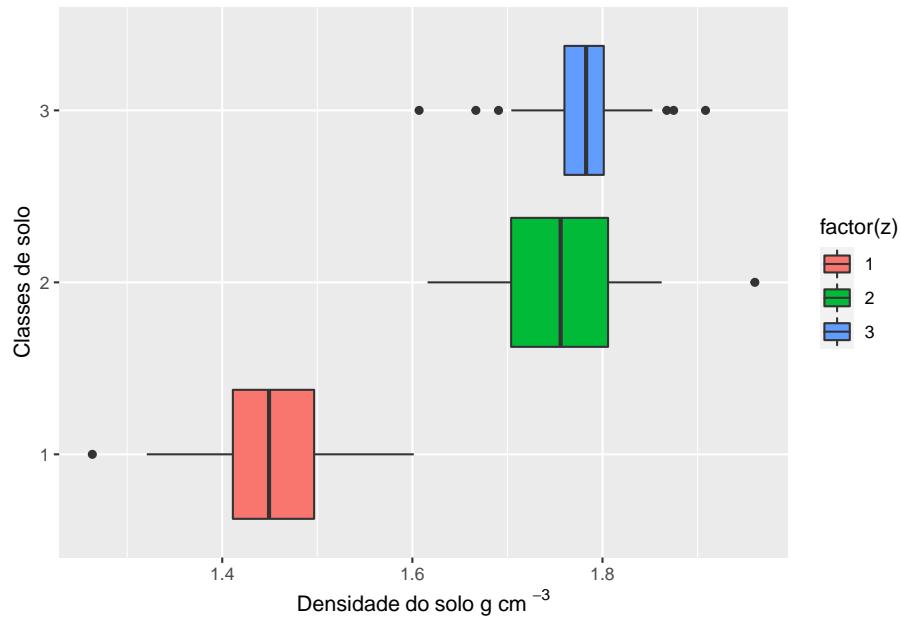
Especifique marcas de escala diretamente:

```
box + coord_cartesian(ylim=c(1, 2)) +
  scale_y_continuous(breaks=seq(0, 2, 0.20))
```



Troque os eixos x e y:

```
box +
  coord_flip()
```



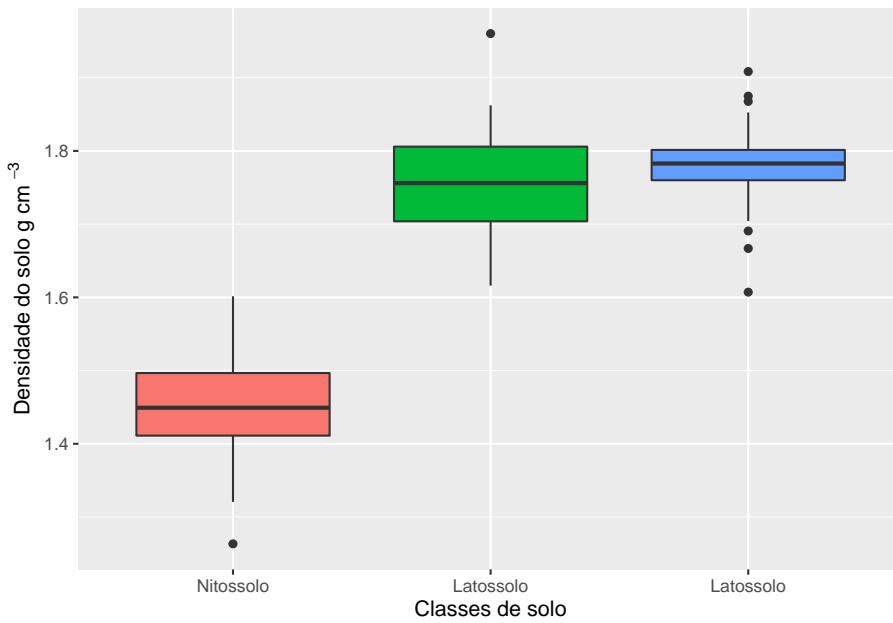
Definir rótulos de marca de escala:

```
box2 <- box +
  scale_x_discrete(breaks=c("1", "2", "3"),
    labels=c("Nitossolo", "Latossolo", "Latossolo"))
```

6.1.3 Legenda

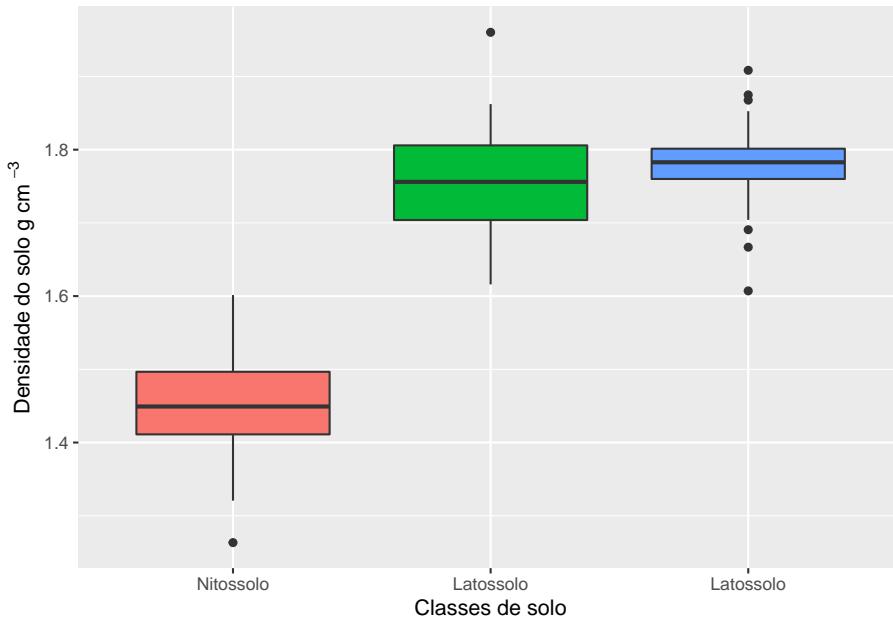
Remover a legenda para uma estética específica (`fill`):

```
box2 + guides(fill=FALSE)
```



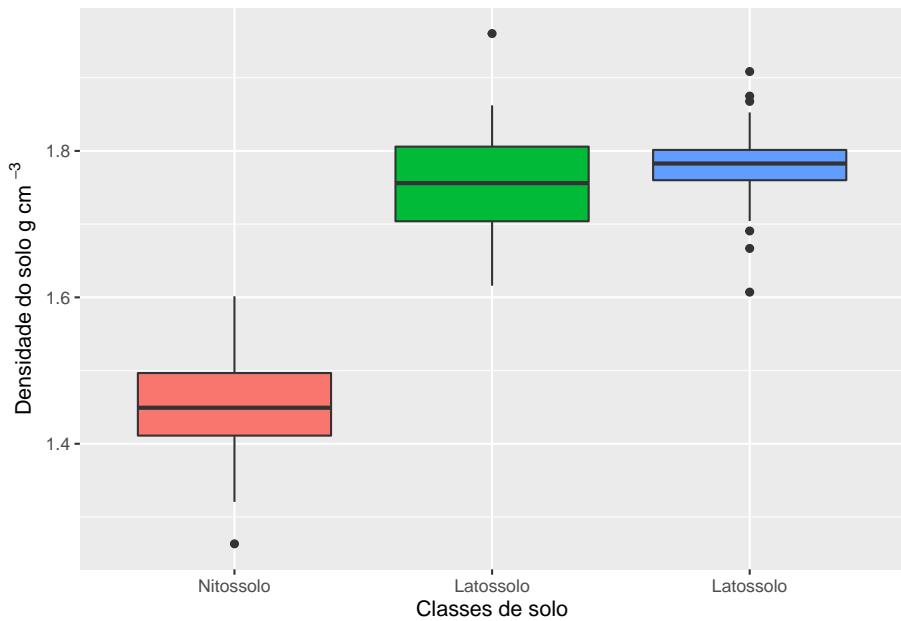
Também pode ser feito ao especificar a `scale`:

```
box2 + scale_fill_discrete(guide=FALSE)
```



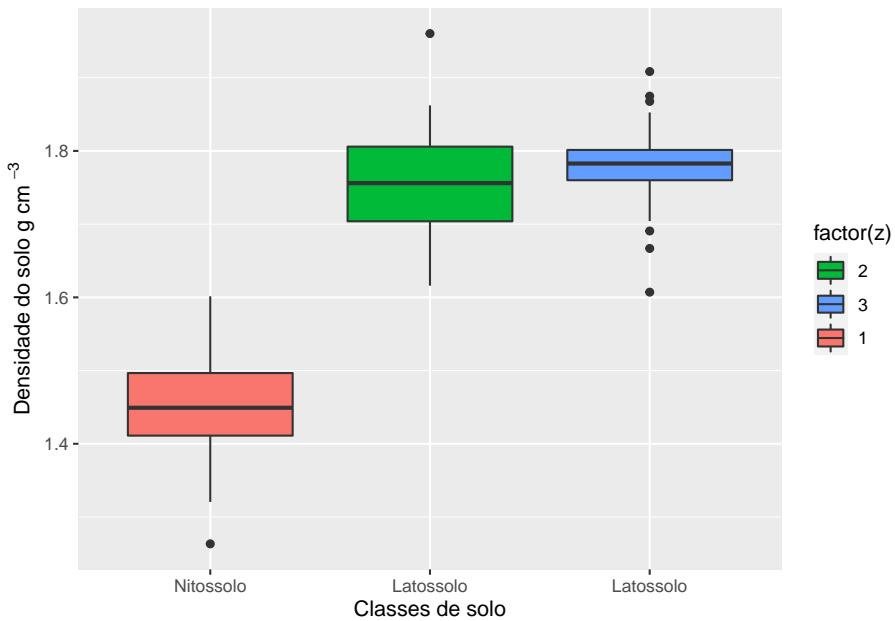
Isso remove todas as legendas:

```
box2 + theme(legend.position="none")
```



Alterando a ordem dos itens na legenda:

```
box2 + scale_fill_discrete(breaks=c("2", "3", "1"))
```

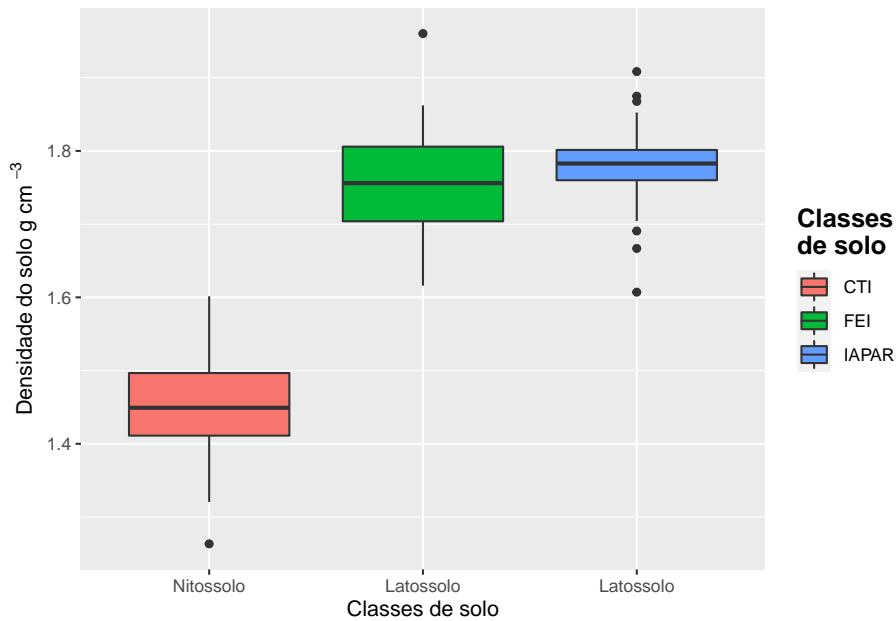


Modificando o texto de legenda de títulos e rótulos:

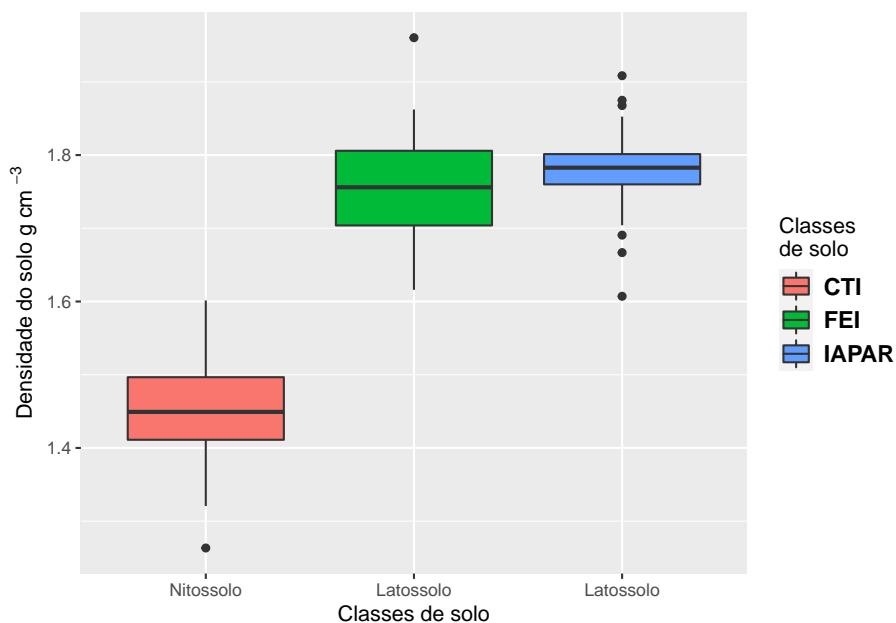
```
box3 <- box2 +
  scale_fill_discrete(name="Classes\nde solo",
    breaks=c("1", "2", "3"),
    labels=c("CTI", "FEI", "IAPAR"))
```

Modificando a aparência do título e dos rótulos da legenda:

```
# Título
box3 + theme(legend.title = element_text(colour="black", size=13, face="bold"))
```

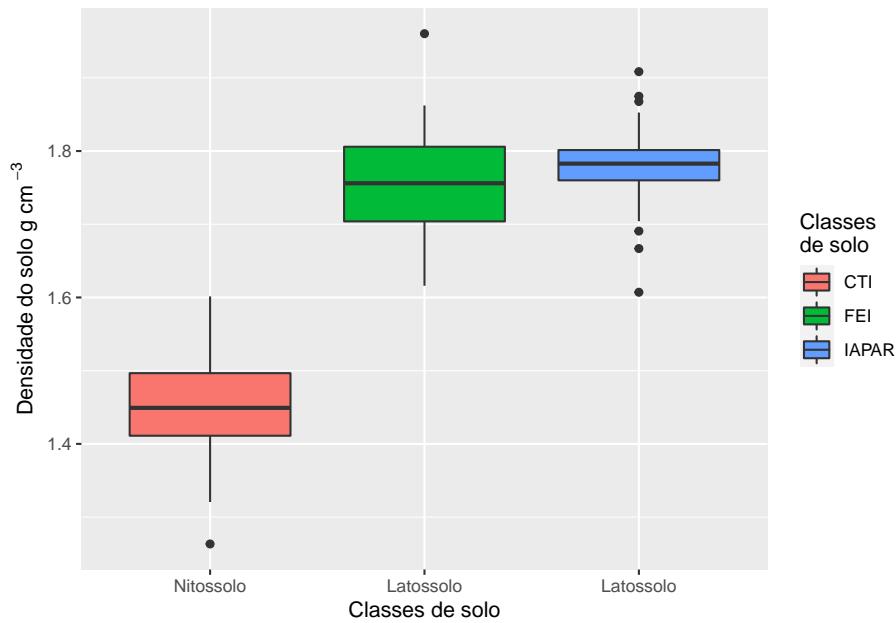


```
# Níveis
box3 + theme(legend.text = element_text(colour="black", size = 12, face = "bold"))
```

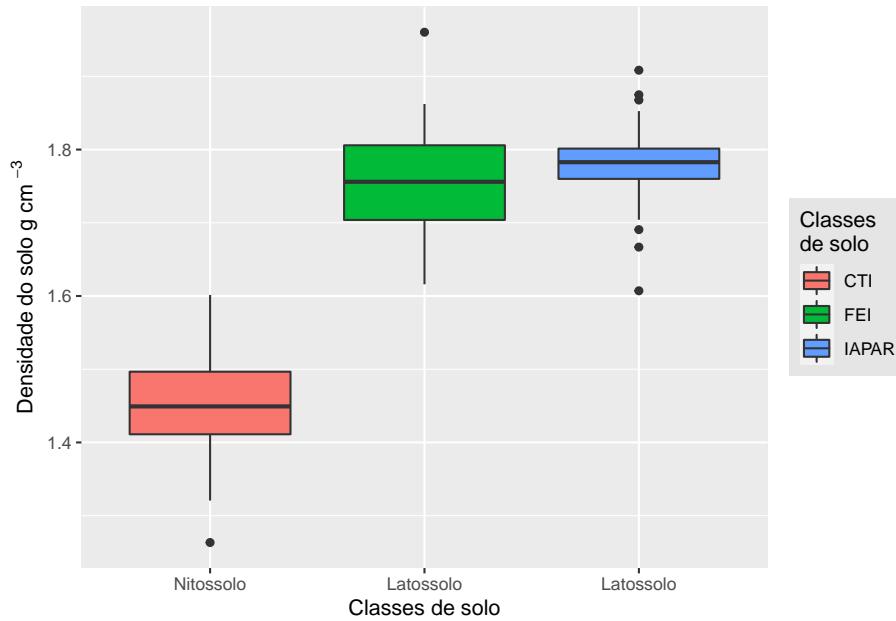


Modificando a caixa de legenda:

```
box3 + theme(legend.background = element_rect())
```

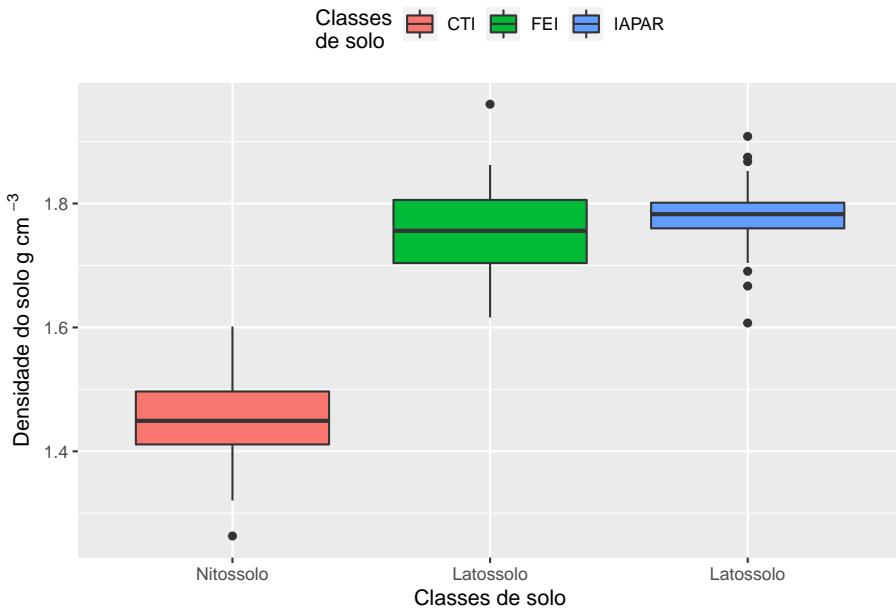


```
box3 + theme(legend.background = element_rect(fill="gray90"))
```



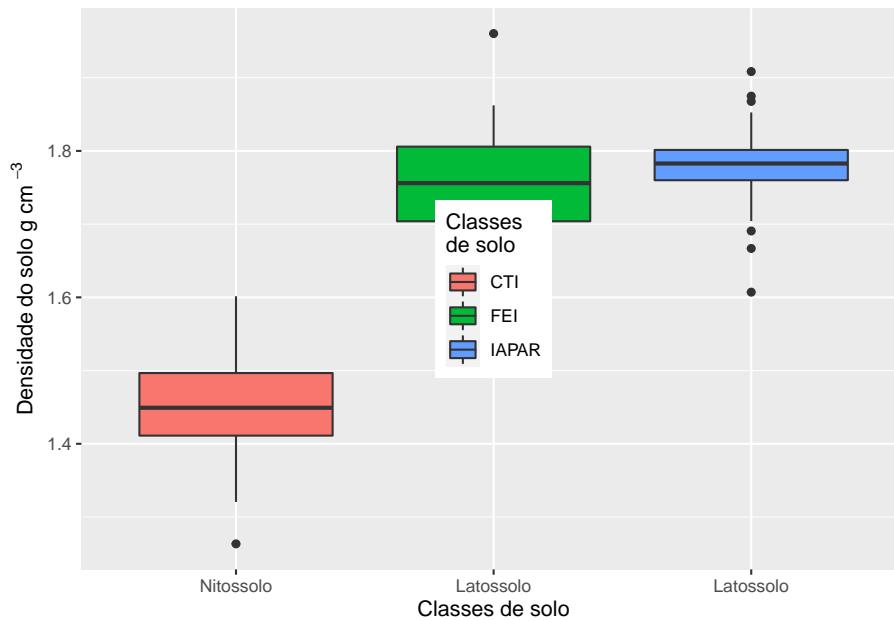
Mudando a posição da legenda:

```
box3 + theme(legend.position="top")
```



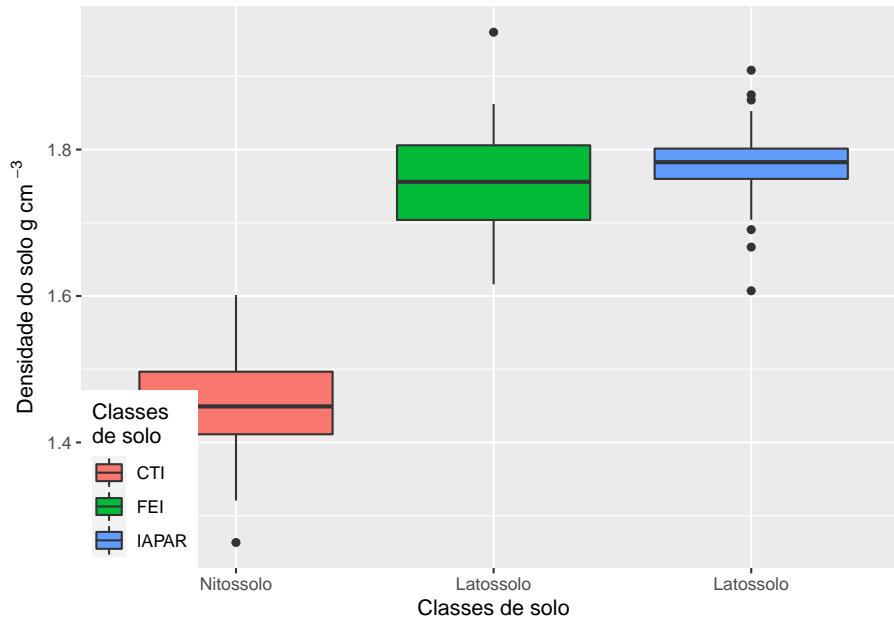
Posicione a legenda no gráfico, em que x, y é 0,0 (canto inferior esquerdo) a 1,1 (canto superior direito):

```
box3 + theme(legend.position=c(.5, .5))
```



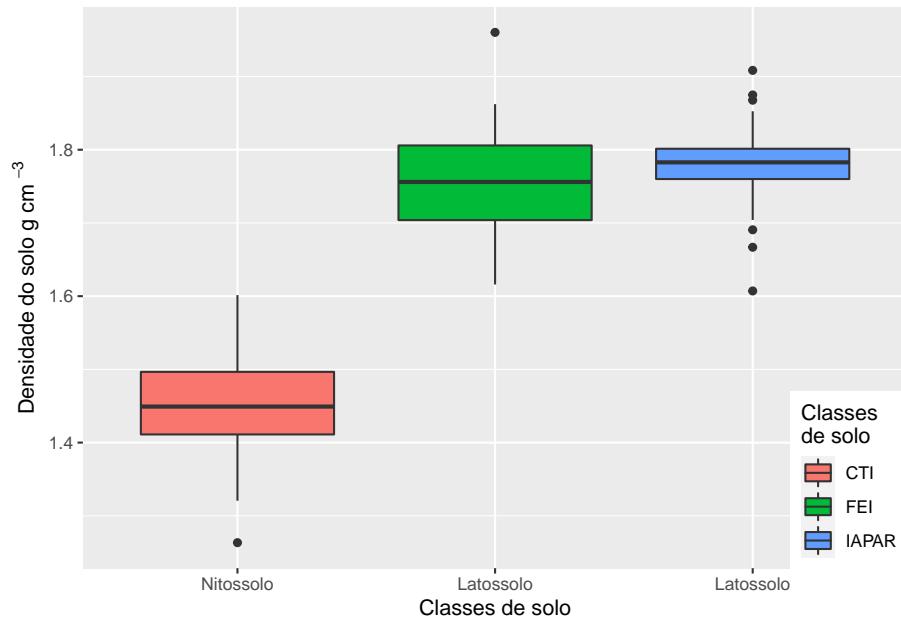
Defina o “ponto de ancoragem” da legenda (o canto inferior esquerdo é 0,0; o canto superior direito é 1,1):

```
box3 + theme(legend.justification=c(0,0), legend.position=c(0,0))
```



Coloque o canto inferior direito da caixa de legenda no canto inferior direito do gráfico:

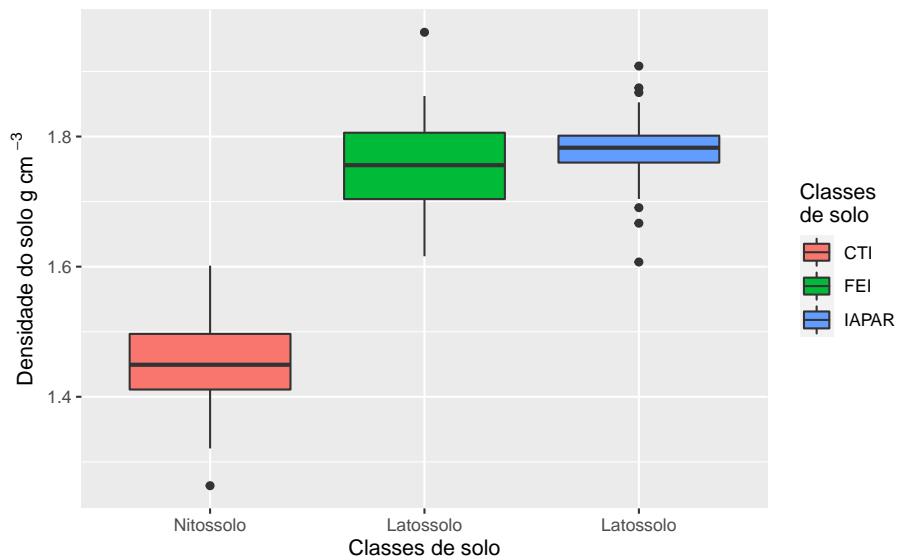
```
box3 + theme(legend.justification=c(1,0), legend.position=c(1,0))
```



6.1.4 Título

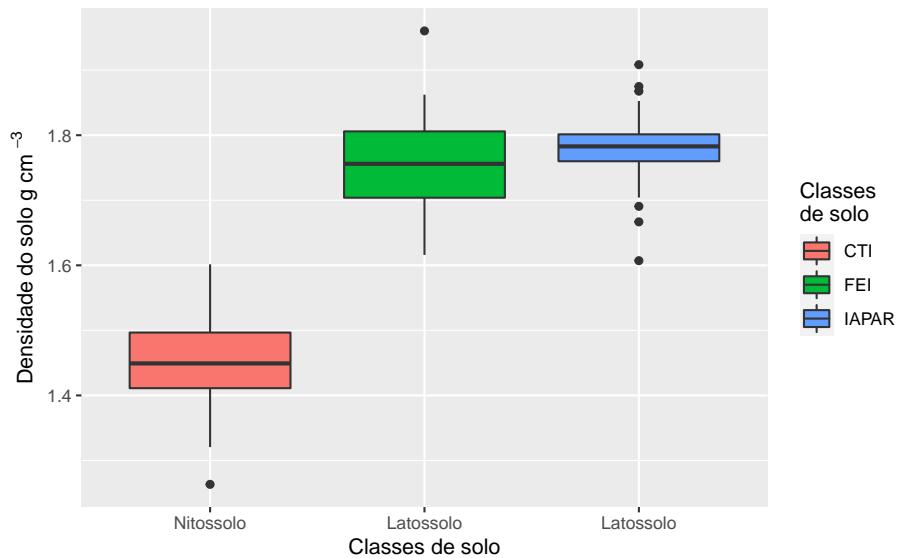
```
box3 + ggtitle("Variabilidade da densidade do solo\nem diferentes solos")
```

Variabilidade da densidade do solo
em diferentes solos



```
box3 + labs(title="Variabilidade da densidade do solo\n em diferentes solos")
```

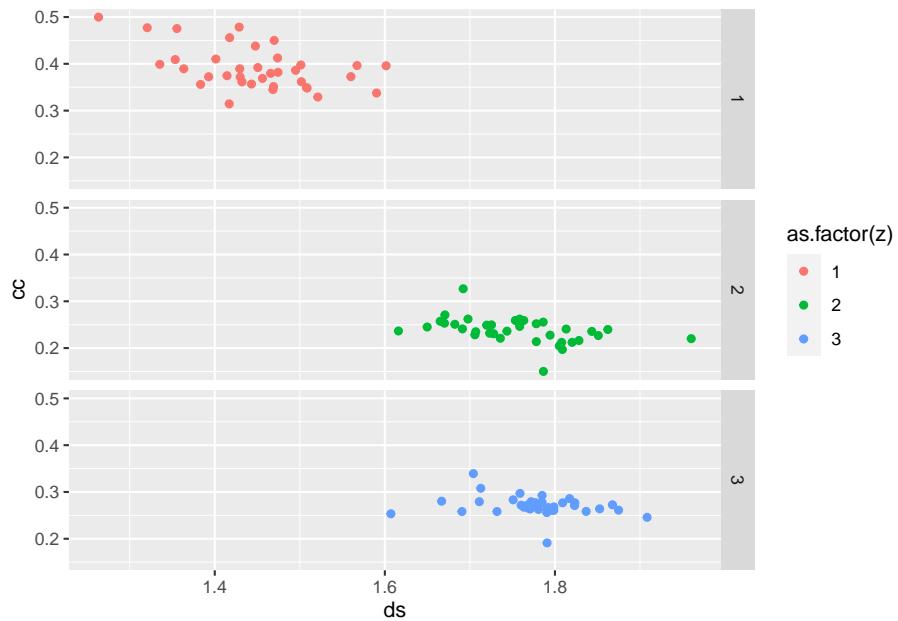
Variabilidade da densidade do solo
em diferentes solos



6.1.5 Facets

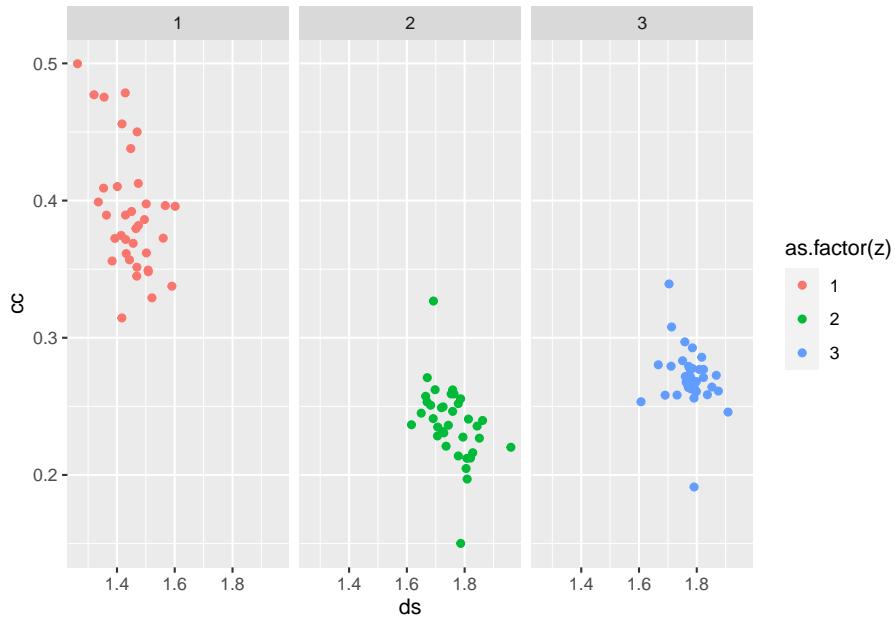
Outra funcionalidade muito importante do **ggplot2** é o uso de **facets**. Você quer dividir seus dados por uma ou mais variáveis e plotar os subconjuntos de dados juntos:

```
ggplot(data = fisio, aes(x = ds, y = cc, colour = as.factor(z))) +
  geom_point() +
  facet_grid(z ~ .)
```



Podemos colocar os gráficos lado a lado também:

```
ggplot(data = fisio, aes(x = ds, y = cc, colour = as.factor(z))) +
  geom_point() +
  facet_grid(. ~ z)
```



6.2 Exemplos

6.2.1 Regressão

Efeito do Gesso no Peso de grãos de feijão Estudo sobre o efeito do gesso no peso de grãos de feijo (*Phaseolus vulgaris* L.) feito por Ragazzi (1979). O experimento foi instalado em delineamento inteiramente casualizado e foram estudados 7 níveis de gesso, de 0 a 300, igualmente espados em 50 kg ha⁻¹.

Baixar dados:

```
dados <- read.table("https://www.dropbox.com/s/r6jz7mrktbgnbnx/BanzattoQd7.2.1.txt?dl=1")
```

Verificar a estrutura dos dados:

```
str(dados)
```

```
## 'data.frame': 28 obs. of 3 variables:
## $ gesso: int 0 0 0 50 50 50 50 100 100 ...
## $ rept : int 1 2 3 4 1 2 3 4 1 2 ...
## $ peso : num 135 140 148 132 162 ...
```

Análise de regressão:

```
model <- lm( gesso ~ peso, dados)

summary(model)

##
## Call:
## lm(formula = gesso ~ peso, data = dados)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -120.41 -70.79 -31.57  74.22 179.24
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -451.935    282.012 -1.603   0.121
## peso          3.849      1.799   2.139   0.042 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 95.7 on 26 degrees of freedom
## Multiple R-squared:  0.1496, Adjusted R-squared:  0.1169
## F-statistic: 4.575 on 1 and 26 DF,  p-value: 0.04201
```

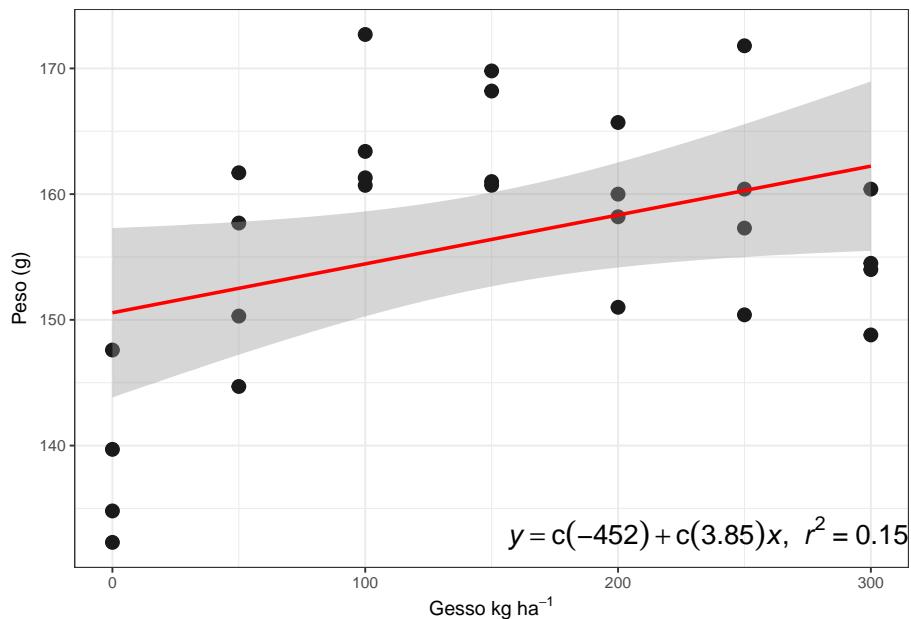
Extrair a equação do modelo:

```
eqn <- as.character(as.expression(substitute(italic(y) == a + b * italic(x) * ", " ~ i
```

Criando o gráfico:

```
ggplot(dados, aes(x=gesso,y=peso,color=peso)) +
  geom_point(size=2.9,shape=19, colour="grey10") +
  theme_bw(base_size = 10) +
  ylab(expression(paste( "Peso (g) " ))) +
  xlab(expression(paste(Gesso," kg ha"^{ -1} ))) +
  annotate("text", label=eqn, parse=TRUE, x=Inf, y=-Inf,
           hjust=1., vjust=-.5, size = 5) +
  stat_smooth(method = lm, se = T, colour="red", size=.85)

## `geom_smooth()` using formula 'y ~ x'
```



6.2.2 Delineamento em blocos casualizados- DBC

Efeito do Promalin sobre Furtos de Macieira

Resultados de um experimento instalado na Fazenda Chapadão, no município de Angatuba - SP. O delineamento experimental foi o de blocos casualizados, sendo as parcelas constituídas de 4 plantas espaçadas de 6 x 7 metros, com 12 anos de idade na época da instalação do experimento.

Baixar dados:

```
dados <- read.table("https://www.dropbox.com/s/9woiye3ce9twp78/BanzattoQd4.5.2.txt?dl=1")
```

Verificar estrutura dos dados:

```
str(dados)
```

```
## 'data.frame': 20 obs. of 3 variables:
## $ promalin: Factor w/ 5 levels "12.5","12.5+12.5",...: 1 3 4 2 5 1 3 4 2 5 ...
## $ bloco   : Factor w/ 4 levels "I","II","III",...: 1 1 1 1 1 2 2 2 2 2 ...
## $ peso    : num 142 140 141 151 154 ...
```

Transformação categorica:

```
dados$promalin = as.factor(dados$promalin)
dados$bloco= as.factor(dados$bloco)
```

Estatística descritiva:

```
summary(dados)
```

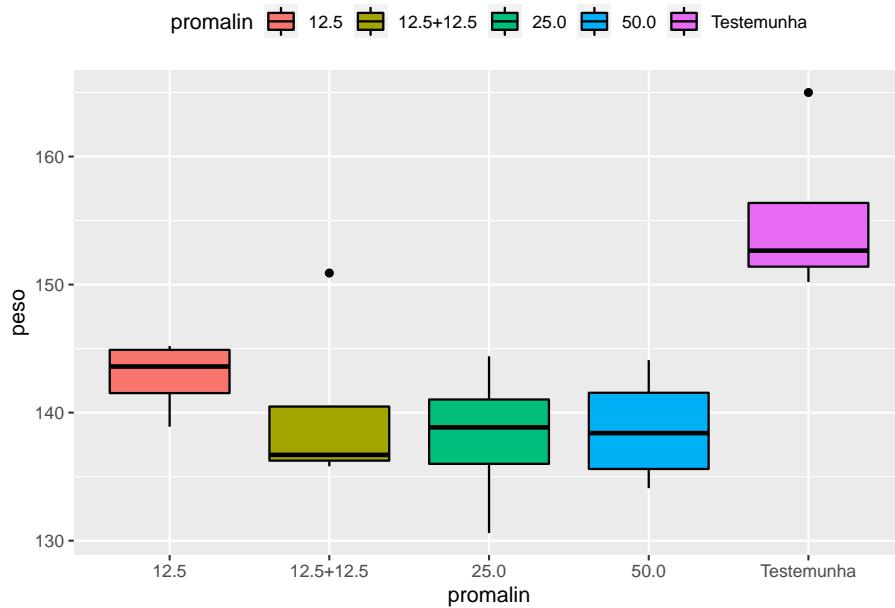
```
##          promalin bloco      peso
## 12.5      :4    I :5   Min.  :130.6
## 12.5+12.5 :4   II :5  1st Qu.:136.8
## 25.0      :4   III:5 Median :141.6
## 50.0      :4    IV :5  Mean  :143.0
## Testemunha:4                    3rd Qu.:146.4
##                               Max.  :165.0
```

Ativar o pacote ggplot:

```
library(ggplot2)
```

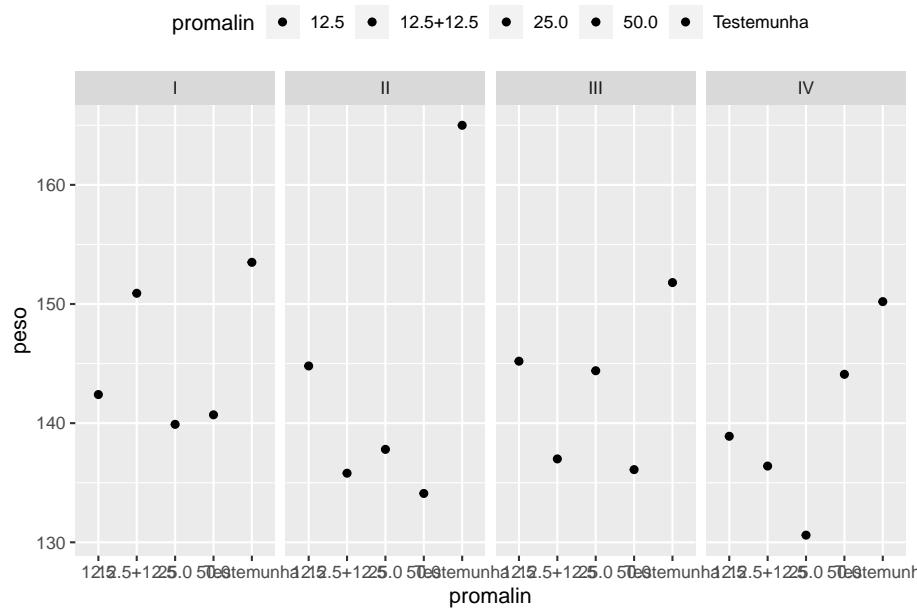
Fazer o gráfico:

```
ggplot(dados,aes(x=promalin ,y=peso, fill=promalin)) +
  geom_boxplot(size=0.55,shape=19, colour="black") +
  theme(legend.position="top")
```



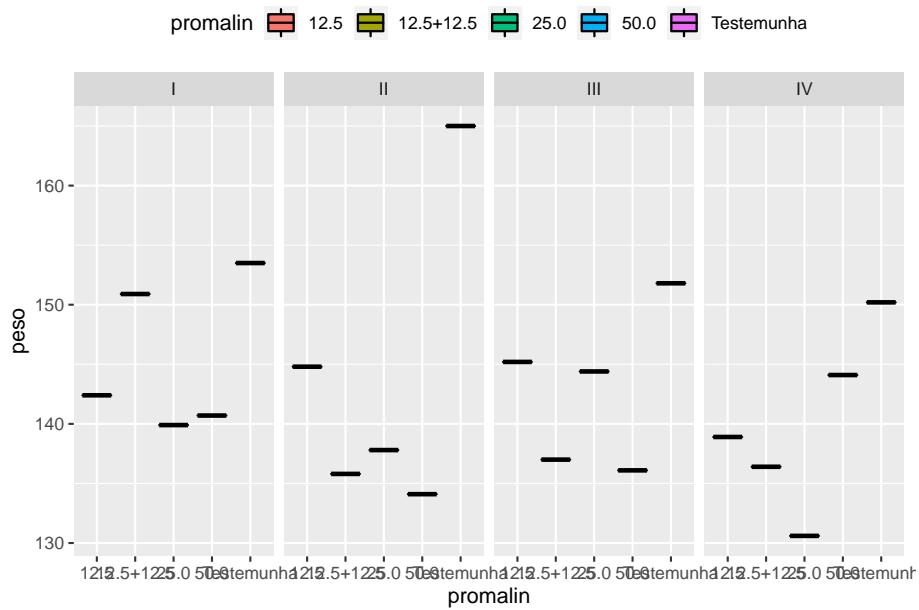
Analizando os blocos:

```
ggplot(dados,aes(x=promalin ,y=peso, fill=promalin)) +  
  geom_point() +  
  theme(legend.position="top") +  
  facet_wrap(~bloco,ncol=4)
```



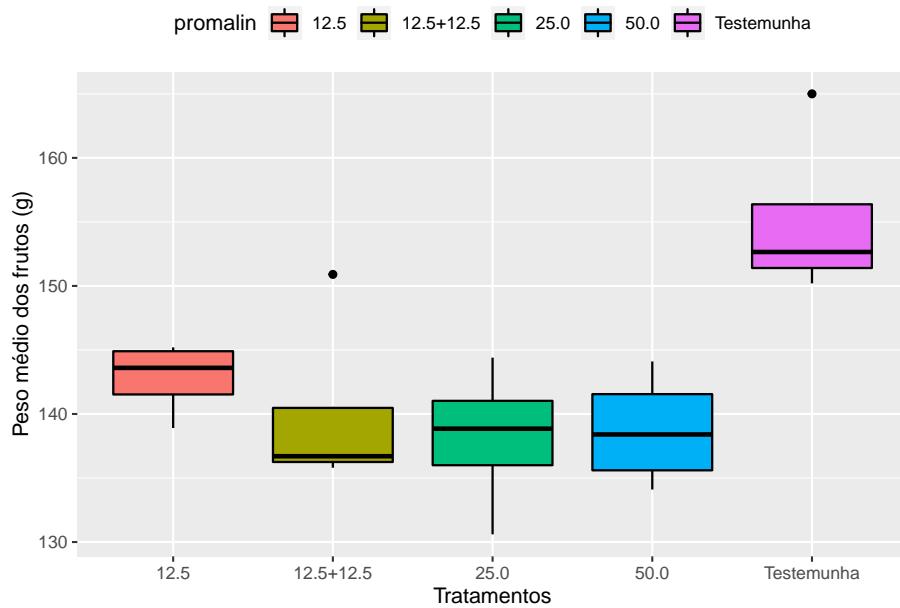
Inserindo médias:

```
ggplot(dados,aes(x=promalin ,y=peso, fill=promalin)) +  
  geom_boxplot(size=0.55,shape=19, colour="black") +  
  theme(legend.position="top") +  
  facet_wrap(~bloco,ncol=4)
```



Inserindo legenda nos eixos:

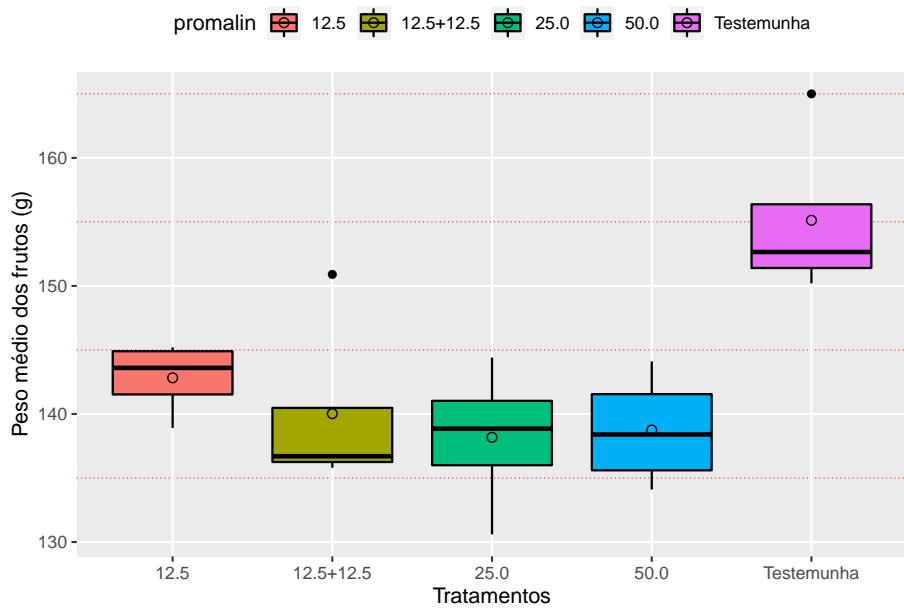
```
ggplot(dados,aes(x=promalin,y=peso, fill=promalin)) +
  geom_boxplot(size=0.55,shape=19, colour="black") +
  theme(legend.position="top") +
  xlab("Tratamentos") +
  ylab("Peso médio dos frutos (g)")
```



Inserindo legenda nos eixos:

```
ggplot(dados,aes(x=promalin ,y=peso, fill=promalin)) +
  geom_boxplot(size=0.55,shape=19, colour="black") +
  theme(legend.position="top") +
  stat_summary(fun.y=mean, geom="point",shape=1,size=2) +
  xlab("Tratamentos") +
  ylab("Peso médio dos frutos (g)") +
  theme(panel.grid.minor = element_line(colour = "red", linetype = "dotted"))
```

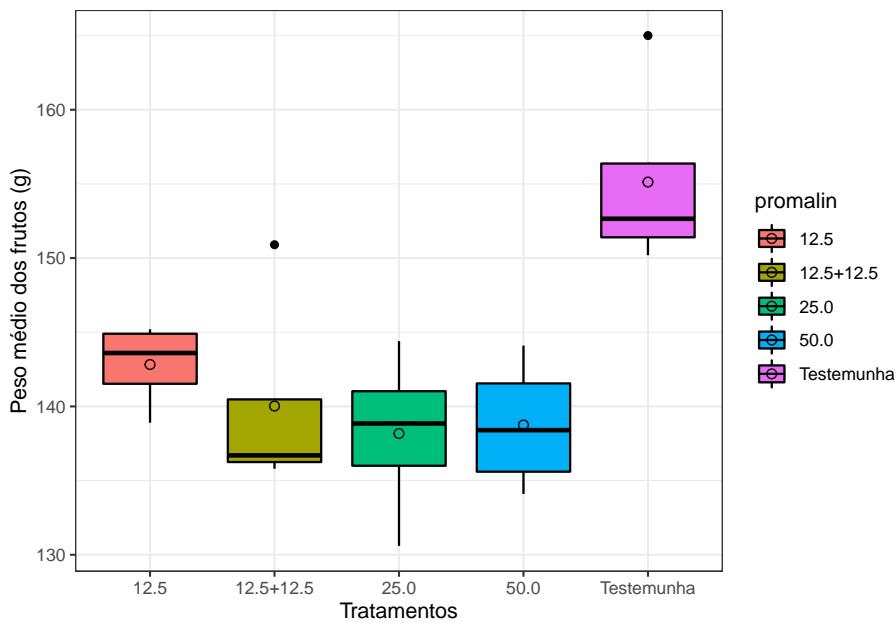
Warning: `fun.y` is deprecated. Use `fun` instead.



Inserindo `tema_bw` preto e branco:

```
ggplot(dados,aes(x=promalin ,y=peso, fill=promalin)) +
  geom_boxplot(size=0.55,shape=19, colour="black") +
  theme(legend.position="top") +
  stat_summary(fun.y=mean, geom="point",shape=1,size=2) +
  xlab("Tratamentos") +
  ylab("Peso médio dos frutos (g)") +
  theme_bw()
```

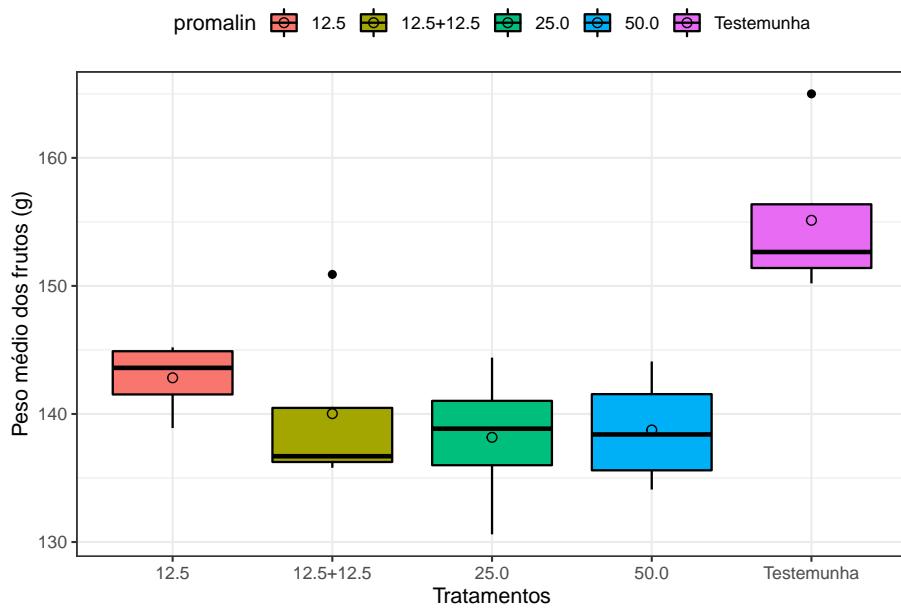
```
## Warning: `fun.y` is deprecated. Use `fun` instead.
```



Inserindo legenda no topo:

```
ggplot(dados,aes(x=promalin ,y=peso, fill=promalin)) +
  geom_boxplot(size=0.55,shape=19, colour="black") +
  theme(legend.position="top") +
  stat_summary(fun.y=mean, geom="point",shape=1,size=2) +
  xlab("Tratamentos") +
  ylab("Peso médio dos frutos (g)") +
  theme_bw() +
  theme(legend.position="top")
```

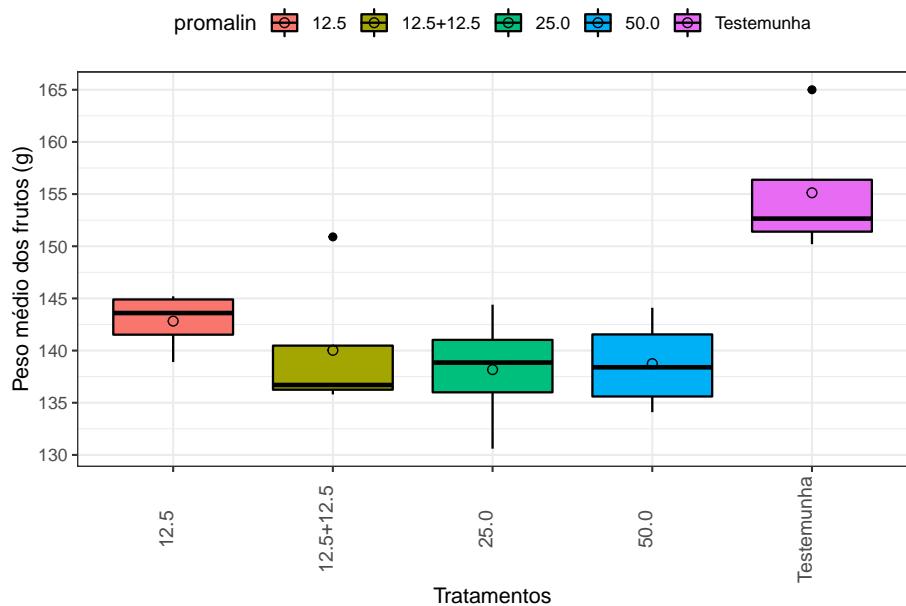
Warning: `fun.y` is deprecated. Use `fun` instead.



Mudando escala do eixo y:

```
ggplot(dados,aes(x=promalin ,y=peso, fill=promalin)) +
  geom_boxplot(size=0.55,shape=19, colour="black") +
  theme(legend.position="top") +
  stat_summary(fun.y=mean, geom="point",shape=1,size=2) +
  xlab("Tratamentos") +
  ylab("Peso médio dos frutos (g)") +
  theme_bw() +
  theme(legend.position="top") +
  scale_y_continuous(breaks=seq(0, 180, 5)) +
  theme( axis.text.x = element_text(angle=90, vjust=0, size=10))
```

```
## Warning: `fun.y` is deprecated. Use `fun` instead.
```



6.2.3 Dados Climáticos

Dados climáticos de Rondonópolis - MT

Baixar dados no banco de dados o arquivo: roo.xlsx

```
roo <- read.csv2("https://www.dropbox.com/s/1ajoi1c8pla3yk6/roo.csv?dl=1")
View(roo)

str(roo)
```

```

## 'data.frame': 4337 obs. of 11 variables:
## $ dd     : int 1 1 2 3 4 5 6 7 8 9 ...
## $ mm     : int 1 2 2 2 2 2 2 2 2 ...
## $ ano    : int 1998 1998 1998 1998 1998 1998 1998 1998 1998 1998 ...
## $ Prec   : num NA 8.2 51 0.6 0 0 0 2.4 NA 0.8 ...
## $ Tmax  : num 30 35.6 31.8 35.4 35.6 36.4 36.8 36.8 36.6 35.2 ...
## $ Tmin  : num 21.7 21.8 21.8 21.5 22.1 22.5 23.5 23.5 24.3 22.9 ...
## $ n     : num NA ...
## $ Tbs   : num NA NA 25.1 25 26.6 ...
## $ Tbu   : num NA NA 23.9 23 23.9 ...
## $ UR    : num NA NA 89.8 86.5 80 ...
## $ Vvento: num NA NA 0.125 0.125 0.275 0.325 0.2 0.175 0.15 0.25 ...

```

Boxplot para tempearatura mínima:

```
ggplot(data = roo, aes(x = factor(mm), y = (Tmin)))+
  geom_boxplot() +
  scale_x_discrete(breaks=c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12"),
                    labels=c("Jan", "Fev", "Mar", "Abr", "Mai", "Jun", "Jul", "Ago", "Set", "Out", "Nov", "Dez"))
## Warning: Removed 222 rows containing non-finite values (stat_boxplot).
```

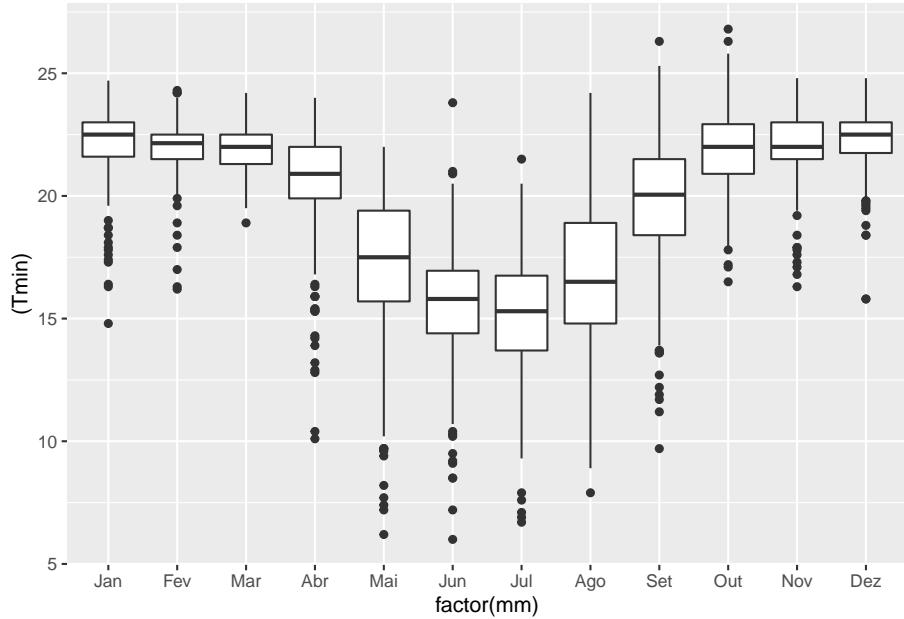


Gráfico de distribuição de temperatura mínima total:

```
ggplot(data = roo, aes(x = (Tmin)))+
  geom_density()
## Warning: Removed 222 rows containing non-finite values (stat_density).
```

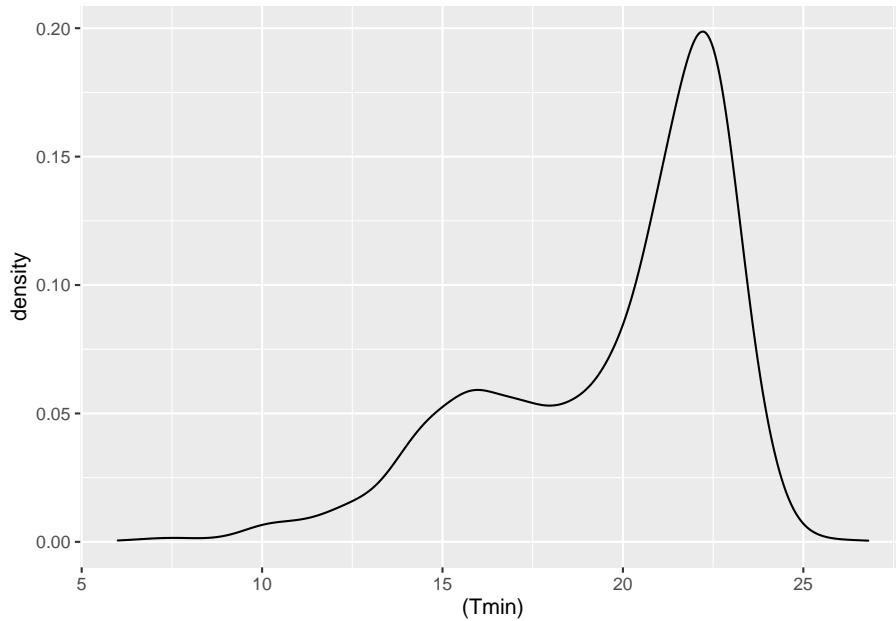
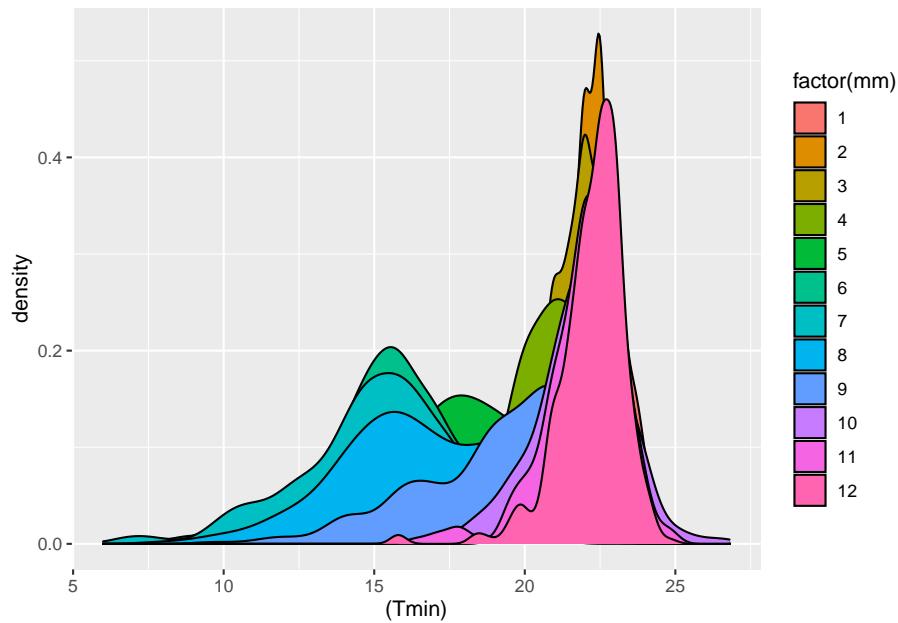


Gráfico de distribuição de temperatura mínima para cada mês:

```
ggplot(data = roo, aes(x = (Tmin), fill=factor(mm)))+  
  geom_density()
```

```
## Warning: Removed 222 rows containing non-finite values (stat_density).
```



6.3 Referência

GROLEMUND, G. WICKHAM, H. R for Data Science Site: <http://r4ds.had.co.nz/>

SITE: <https://www.statmethods.net/index.html>

CHANG, W. R Graphics Cookbook: Practical Recipes for Visualizing Data, Publisher: O'Reilly Media, 2002, 416 p. Site: <http://shop.oreilly.com/pesouct/0636920023135.do>

Este capítulo foi baseado no livro **Conhecendo o R: Um visão mais que estatística**, e na página do **Prof. Paulo Justiniano Ribeiro**

Chapter 7

Testes Estatísticos

O R inclui em sua gama de utilidades uma poderosa ferramenta da estatística contemporânea: os testes estatísticos. Dentre esses, podemos destacar os testes de média, amplamente usados em várias áreas do conhecimento.

7.1 Teste t de Student

O teste t é bastante usado em várias situações do cotidiano quando se deseja fazer comparações entre *uma ou mais médias*, sejam elas dependentes ou não. Abaixo estão exemplos de vários modos de realizarmos o teste t.

Dados referentes a temperatura média do ar em duas condições: dentro de uma casa de vegetação e no campo:

```
pira_tem <- read.csv2 ("https://www.dropbox.com/s/zvp5iftcpb6bdpe/pira_tem.csv?dl=1",
  dec=".")  
str(pira_tem)  
  
## 'data.frame':    768 obs. of  5 variables:  
## $ hora    : Factor w/ 96 levels "0:00","0:15",...: 1 2 3 4 5 6 7 8 49 50 ...  
## $ periodo: Factor w/ 4 levels "equi_out","equi_prim",...: 4 4 4 4 4 4 4 4 4 4 ...  
## $ local   : Factor w/ 2 levels "campo","estufa": 2 2 2 2 2 2 2 2 2 2 ...  
## $ temp    : num  23.1 22.9 22.7 22.6 22.5 ...  
## $ X       : logi NA NA NA NA NA NA ...
```

Apresentação dos dados em forma de gráfico:

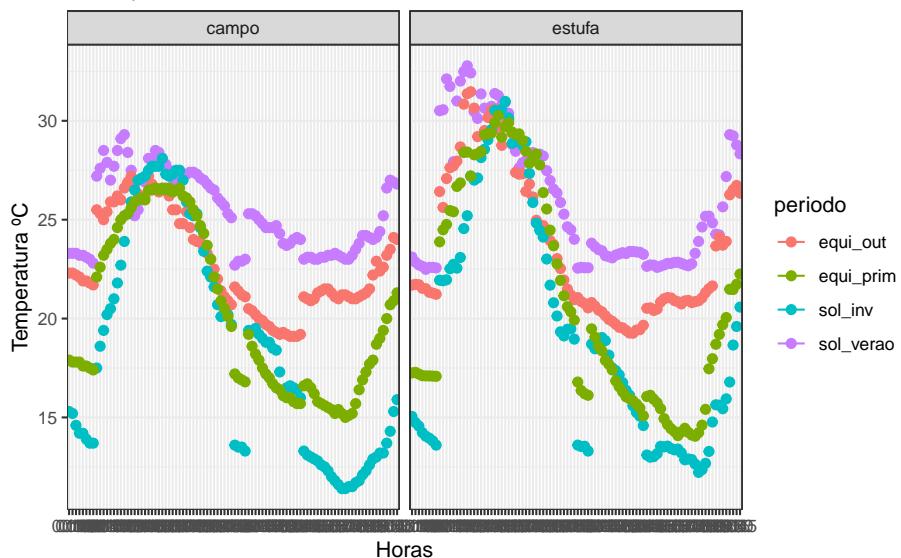
```

library(ggplot2)
ggplot(data= pira_tem, aes (x = hora, y = temp, colour = periodo)) +
  geom_point(size=2,shape=19) +
  geom_line() +
  facet_grid(.~local) +
  xlab("Horas") +
  ylab("Temperatura °C") +
  ggtitle("Variação da temperatura mediana\nnas quatro efemérides") +
  theme(plot.title=element_text(face="bold", size=12, hjust = 0.5)) +
  theme_bw()

## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?

```

Variação da temperatura mediana
nas quatro efemérides



`t.test()` Realiza o teste t-Student para uma ou duas amostras.

sintaxe: `t.test(amostra1, amostra2, opções)`

Parâmetros

amostra1: Vetor contendo a amostra da qual se quer testar a média populacional, ou comparar a média populacional com a média populacional da amostra 2.

amostra2: Vetor contendo a amostra 2 para comparação da média populacional com a média populacional da amostra 1.

Opções

alternative: string indicando a hipótese alternativa desejada. Valores possíveis: “two-sided”, “less” ou “greater”.

mu: valor indicando o verdadeiro valor da média populacional para o caso de uma amostra, ou a diferença entre as médias para o caso de duas amostras.

paired: - TRUE - realiza o teste t pareado. - FALSE - realiza o teste t não pareado.

var.equal: - TRUE - indica que a variância populacional é igual nas duas amostras. - FALSE - indica que a variância populacional de cada amostra é diferente.

conf.level: coeficiente de confiança do intervalo.

7.1.1 Para uma média

Vamos testar se a temperatura horária do solstício de verão no campo tem média igual ou maior que 21 °C na cidade de Piracicaba-SP.

H0: $\mu \geq 21$

IC 95 para mu

1.0 Passo filtrar os dados pelo fator “período” com o nível *sol_verao* (solstício de verão):

```
#Dividir os dados - subset()
sol_verao_amb <- subset(pira_tem, periodo == "sol_verao")
```

2.0 Passo filtrar os dados pelo fator “local” com o nível campo:

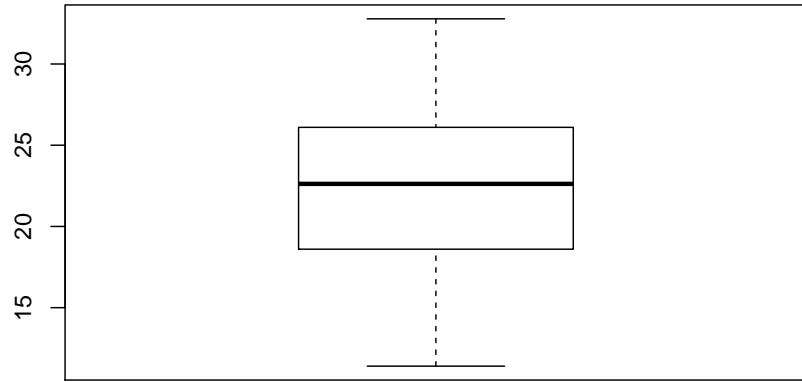
```
sol_verao_camp <- subset(sol_verao_amb, local == "campo")
```

3.0 Verificar dados graficamente:

```
attach(pira_tem)
```

```
## The following objects are masked from pira_tem (pos = 44):
##
##      hora, local, periodo, temp, X
```

```
boxplot(temp)
```



4.0 Usar o teste T:

```
t.test(sol_verao_camp$temp,
mu=21,
alternative="greater",
conf.level = 0.95 )
```

*#amostra a ser testada
#hipótese de nulidade
#teste unilateral pela direita
#Intervalo de confiancia de 95%*

```
##  

## One Sample t-test  

##  

## data: sol_verao_camp$temp  

## t = 21.648, df = 95, p-value < 2.2e-16  

## alternative hypothesis: true mean is greater than 21  

## 95 percent confidence interval:  

## 24.96332      Inf  

## sample estimates:  

## mean of x  

## 25.29271
```

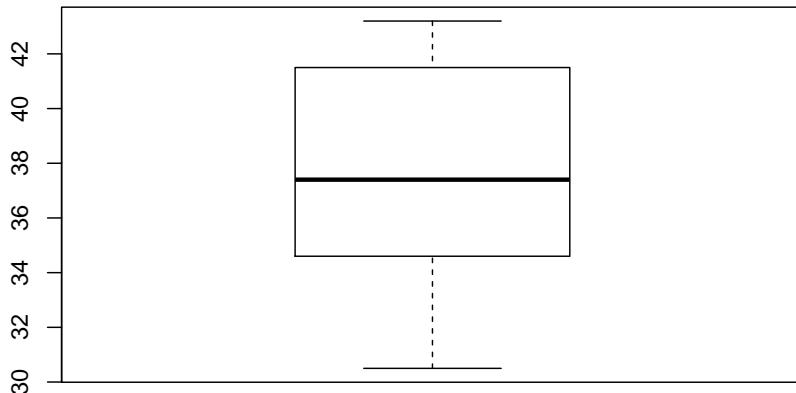
Agora basta fazer a interpretação correta da saída do R. Para saber qual hipótese foi aceita, basta verificar o valor do *p-value* e estipular um nível de significância. Se neste exemplo o nível de significância fosse de 5% a hipótese alternativa seria

aceita uma vez que o *p-value* foi menor ou igual a 0,05. Caso o *p-value* tivesse sido maior que 5% então aceitariamós a hipótese de nulidade. Como a hipótese alternativa foi a aceita isso implica que a temperatura do ar no solstício de verão possui média estatisticamente diferente do valor 21°C a um nível de significância de 5%.

Exercicio 1

Vamos testar se X tem média estatisticamente igual a 35 ou maior H₀: $\mu = > 35$:

```
x <- c (30.5,35.3,33.2,40.8,42.3,41.5,36.3,43.2,34.6,38.5)
boxplot(x)
```



Teste t:

```
t.test(x,
       mu=35,
       alternative = "greater")

##
##  One Sample t-test
##
## data: x
## t = 1.9323, df = 9, p-value = 0.04268
```

```
## alternative hypothesis: true mean is greater than 35
## 95 percent confidence interval:
## 35.13453      Inf
## sample estimates:
## mean of x
##      37.62
```

Como foi significativo, admitimos que a amostra x é oriunda de um população com média maior que o valor de 35, com nível de 5% de significância.

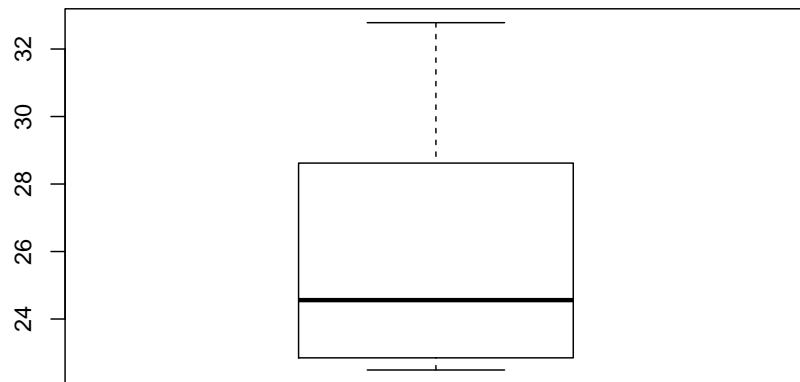
Exercício 2

Um pesquisador afirmou que a temperatura média de solstício de verão medido na casa de vegetação em Piracicaba-SP tem média **22,2 °C**. Desconfiando desse resultado, um outro pesquisador com dados provenientes da mesma estação climatológicas em períodos diferentes encontrou os seguintes resultados:

$H_0: \mu = 22,2$

```
sol_verao_amb <- subset(pira_tem, periodo == "sol_verao")
```

```
sol_verao_est <- subset(sol_verao_amb, local == "estufa")
boxplot(sol_verao_est$temp)
```



Essa afirmação é verdadeira?

```
t.test(sol_verao_est$temp,
      mu=22.2,
      alternative="two.sided",
      conf.level = 0.99) #amostra a ser testada
#hipótese de nulidade
#teste bilateral não considera se é maior ou menor
#significância de 1%
```

```
## 
## One Sample t-test
##
## data: sol_verao_est$temp
## t = 10.98, df = 95, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 22.2
## 99 percent confidence interval:
##  25.06976 26.87629
## sample estimates:
## mean of x
## 25.97302
```

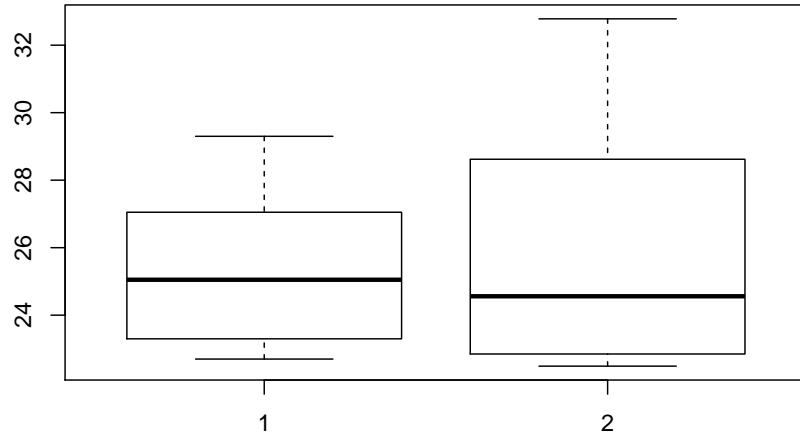
7.1.2 Para duas médias independentes

Para a realização do teste t pressupõe-se que as amostras possuem variâncias iguais além de seguirem distribuição normal.

Vamos a um exemplo:

Suponha dois conjuntos de dados de temperatura de média do ar de dois ambientes (casa de vegetação e campo). Verifique se as temperaturas dos dois ambientes são estatisticamente diferentes usando 5% de significância. $H_0: \mu_{da\ temp\ da\ casa\ de\ vegetação} = \mu_{da\ temp\ do\ campo}$

```
boxplot(sol_verao_camp$temp, sol_verao_est$temp)
```



```
t.test(sol_verao_camp$temp, sol_verao_est$temp, #amostras a serem testadas
       alternative = "greater", #unilateral a direita
       var.equal = T ) #variância homogênea
```

```
## 
##  Two Sample t-test
## 
## data: sol_verao_camp$temp and sol_verao_est$temp
## t = -1.7147, df = 190, p-value = 0.956
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
## -1.336097      Inf
## sample estimates:
## mean of x mean of y
## 25.29271  25.97302
```

Uma vez que o *p-value* foi maior que 0,05, podemos concluir que as médias de temperatura dos dois ambientes não são diferentes, estatisticamente, a 5% de significância. Veja que o resultado desta análise mostra o valor de t (estatística do teste), os graus de liberdade (df) e o valor de p (significância). Além disso, o resultado do teste ainda mostra as médias para cada grupo.

7.1.3 Para duas médias dependentes

Neste caso vamos usar o mesmo nível de significância do exemplo das amostras independentes. As hipóteses se mantêm. Agora basta adicionar o argumento `paired=T`, informando que as amostras são dependentes:

```
t.test(sol_verao_camp$temp, sol_verao_est$temp, #amostras a serem testadas
       conf.level=0.99, #nível de confiança
       paired=T, #indica dependência entre as amostras
       var.equal = T ) #variância homogênea

##
## Paired t-test
##
## data: sol_verao_camp$temp and sol_verao_est$temp
## t = -3.8433, df = 95, p-value = 0.0002193
## alternative hypothesis: true difference in means is not equal to 0
## 99 percent confidence interval:
## -1.1455999 -0.2150251
## sample estimates:
## mean of the differences
## -0.6803125
```

Note que a estatística do teste-t pareado não é baseada na média dos tratamentos, e sim na diferença entre os pares de tratamentos.

7.2 Teste de variância

7.2.1 Usando o teste de F

H_0 : as variâncias das amostras são homogêneas

```
var.test (sol_verao_camp$temp, sol_verao_est$temp)

##
## F test to compare two variances
##
## data: sol_verao_camp$temp and sol_verao_est$temp
## F = 0.33301, num df = 95, denom df = 95, p-value = 1.801e-07
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
## 0.2221988 0.4990825
```

```
## sample estimates:
## ratio of variances
##                 0.3330098
```

As variâncias não são homogeneas.

Vamos resolver novamente o exercicio anterior, modificando o argumento `var.equal`:

```
t.test(sol_verao_camp$temp, sol_verao_est$temp, #amostras a serem testadas
       conf.level=0.99,                                #nível de confiança
       paired=T,                                         #indica dependência entre as amostras
       var.equal = F )                                    #variância homogênea

##
## Paired t-test
##
## data: sol_verao_camp$temp and sol_verao_est$temp
## t = -3.8433, df = 95, p-value = 0.0002193
## alternative hypothesis: true difference in means is not equal to 0
## 99 percent confidence interval:
## -1.1455999 -0.2150251
## sample estimates:
## mean of the differences
##                  -0.6803125
```

7.3 Teste para a normalidade - `shapiro.test()`

Por vezes temos necessidade de identificar com certa confiança se uma amostra ou conjunto de dados segue a distribuição normal. Isso é possível, no R, com o uso do comando `shapiro.test()`

Verifique normalidade dos dados:

```
shapiro.test(sol_verao_camp$temp)
```

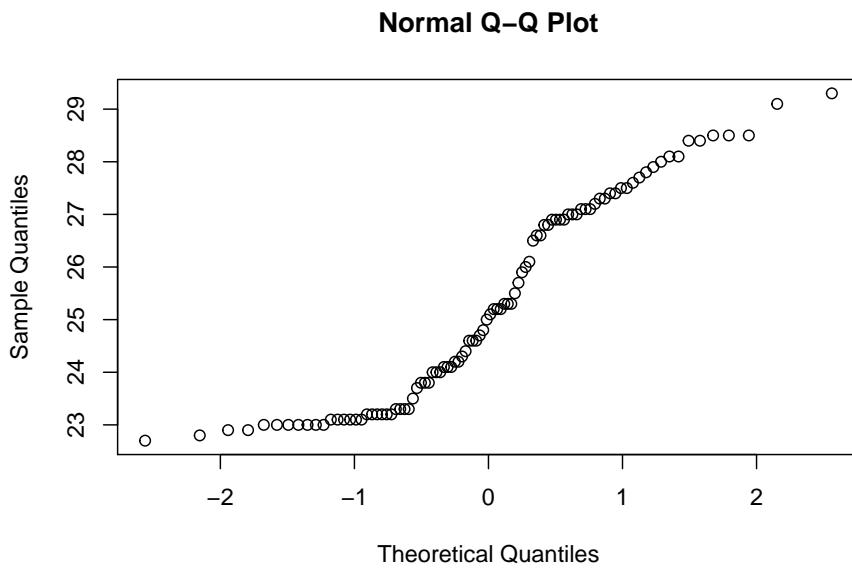
```
##
## Shapiro-Wilk normality test
##
## data: sol_verao_camp$temp
## W = 0.90789, p-value = 5.043e-06
```

```
shapiro.test(sol_verao_est$temp)
```

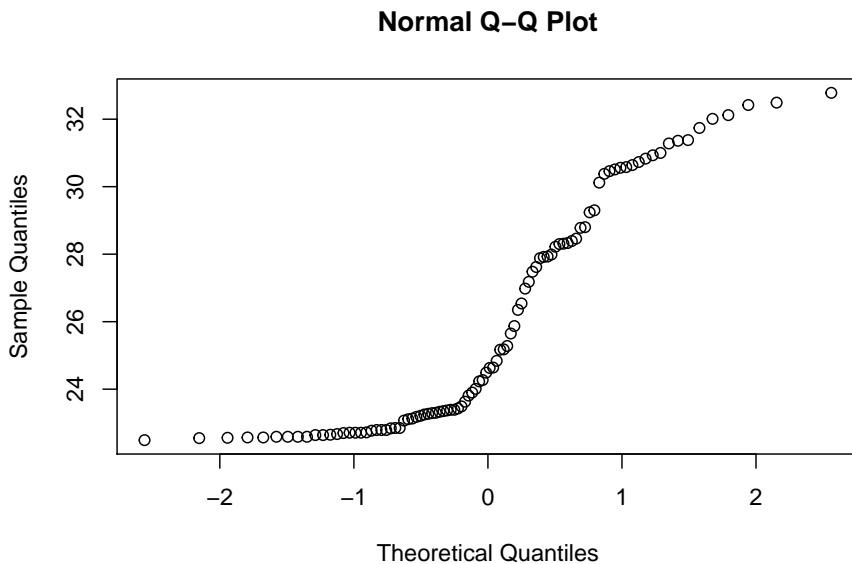
```
##  
## Shapiro-Wilk normality test  
##  
## data: sol_verao_est$temp  
## W = 0.85041, p-value = 2.027e-08
```

O comando `qqnorm()` nos fornece diretamente um gráfico da distribuição de percentagens acumuladas chamado de gráfico de probabilidade normal. Se os pontos deste gráfico seguem um padrão aproximado de uma reta, este fato evidencia que a variável aleatória em questão tem a distribuição aproximadamente normal:

```
qqnorm(sol_verao_camp$temp) #obtendo o normal probability plot só para comparação
```



```
qqnorm(sol_verao_est$temp)
```



7.4 Teste U de Mann-Whitney

H0: mu da temp da casa de vegetação = mu da temp do campo:

```
wilcox.test(sol_verao_camp$temp,sol_verao_est$temp,
            alternative = "two.side")

##
##  Wilcoxon rank sum test with continuity correction
##
##  data:  sol_verao_camp$temp and sol_verao_est$temp
##  W = 4579, p-value = 0.941
##  alternative hypothesis: true location shift is not equal to 0
```

7.5 Covariância e Correlação

A covariância e a correlação entre dois conjuntos de dados quaisquer podem ser obtidos pelos comandos `cov(x,y)` e `cor(x,y)`, respectivamente. São medidas utilizadas no estudo do comportamento conjunto de duas variáveis quantitativas distintas. Elas informam a variação conjunta (covariância) ou grau de associação (correlação) entre duas variáveis aleatórias X e Y.

A correlação de **Pearson** é uma medida paramétrica de associação linear entre duas variáveis.

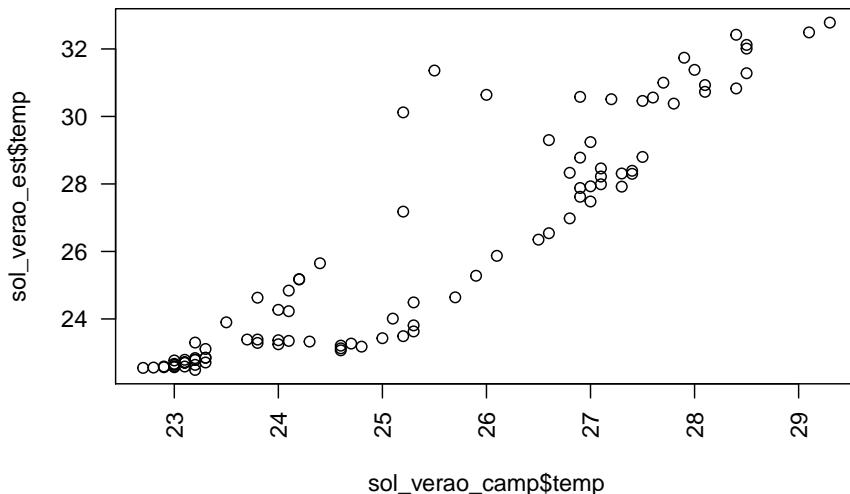
A correlação de ordem de **Sperman** é uma medida não paramétrica de associação entre duas variáveis

A correlação de ordem de **Kendall** é outra medida não paramétrica da associação, baseada na concordância ou discordância dos pares x-y:

```
help ("cor.test")
```

Plote os valores:

```
plot(sol_verao_camp$temp,sol_verao_est$temp, las=2)
```



Teste de correlação de Pearson:

```
cor(sol_verao_camp$temp,sol_verao_est$temp,
    method = "pearson"
    )
```

```
## [1] 0.9250728
```

Teste de correlação de Pearson (the default):

```
cor(sol_verao_camp$temp,sol_verao_est$temp)
```

```
## [1] 0.9250728
```

Teste de correlação de Pearson trocando o X e Y:

```
cor(sol_verao_est$temp, sol_verao_camp$temp)
```

```
## [1] 0.9250728
```

Teste de correlação de Spearman:

```
cor(sol_verao_camp$temp,sol_verao_est$temp,
    method = "spearman")
```

```
## [1] 0.9412321
```

Teste de correlação de Kendall:

```
cor(sol_verao_camp$temp,sol_verao_est$temp,
    method = "kendall")
```

```
## [1] 0.8113615
```

Teste de correlação de Pearson:

```
cor.test (sol_verao_camp$temp,sol_verao_est$temp,
          method = "pearson"
          )
```

```
##
## Pearson's product-moment correlation
##
## data: sol_verao_camp$temp and sol_verao_est$temp
## t = 23.615, df = 94, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8895702 0.9494668
## sample estimates:
##      cor
## 0.9250728
```

```

cor.test (sol_verao_camp$temp,sol_verao_est$temp,
          method = "spearman"
        )

## Warning in cor.test.default(sol_verao_camp$temp, sol_verao_est$temp, method =
## "spearman"): Cannot compute exact p-value with ties

## 
## Spearman's rank correlation rho
##
## data: sol_verao_camp$temp and sol_verao_est$temp
## S = 8664.7, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##      rho
## 0.9412321

cor.test (sol_verao_camp$temp,sol_verao_est$temp,
          method = "spearman", exact = F
        )

## 
## Spearman's rank correlation rho
##
## data: sol_verao_camp$temp and sol_verao_est$temp
## S = 8664.7, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##      rho
## 0.9412321

cov (sol_verao_camp$temp,sol_verao_est$temp)

## [1] 6.051517

```

7.6 Outros testes

Utilizaremos o banco de dados: dadosfisio

```

fisio <- read.csv2("https://www.dropbox.com/s/zg7fyg1iewtji49/dadosfisio.csv?dl=1")
attach(fisio)

```

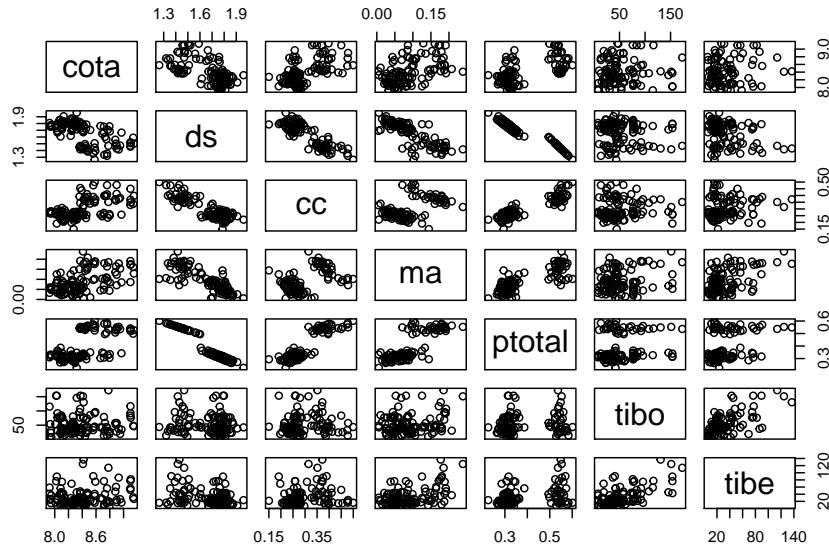
```

## The following objects are masked _by_ .GlobalEnv:
##
##      a, b, x, y, z

## The following objects are masked from fisio (pos = 44):
##
##      a, b, cc, cota, ds, ma, ptotal, tibe, tibo, x, X120, X3, X60, X90,
##      y, z

pairs(fisio[,4:10])

```



Teste de Spearman

```
cor(fisio[,3:8],method = "spearman")
```

```

##          y      cota      ds       cc      ma
## y 1.00000000 0.03913488 -0.007304133 -0.005043318 -0.02834692
## cota 0.039134875 1.00000000 -0.625860523 0.506913897 0.51222110
## ds -0.007304133 -0.62586052 1.000000000 -0.719094380 -0.80918958
## cc -0.005043318 0.50691390 -0.719094380 1.000000000 0.40193585
## ma -0.028346924 0.51222110 -0.809189577 0.401935847 1.000000000
## ptotal -0.001391263 0.54405131 -0.962597591 0.812104786 0.75424837
##          ptotal
## y      -0.001391263

```

```
## cota      0.544051305
## ds       -0.962597591
## cc       0.812104786
## ma       0.754248369
## ptotal   1.000000000
```

7.6.1 hydroGOF

Carregando a biblioteca hydroGOF, que contém dados e funções usadas nesta análise:

```
library(hydroGOF)
```

Cálculo das medidas numéricas de qualidade do ajuste para o “melhor” caso (inatingível):

```
gof(sim = fisio$ds, obs= fisio$cc)
```

```
##          [,1]
## ME        1.36
## MAE       1.36
## MSE       1.90
## RMSE      1.38
## NRMSE % 1866.00
## PBIAS % 453.80
## RSR        18.66
## rSD        2.24
## NSE       -350.47
## mNSE      -20.82
## rNSE      -460.60
## d          0.07
## md         0.04
## rd         -0.22
## cp        -911.57
## r          -0.87
## R2         0.75
## bR2        0.15
## KGE       -4.06
## VE        -3.54
```


Chapter 8

Analise de variância (ANOVA)

Estudos estatísticos contemporâneos contemplam a análise de variância tendo em vista que este procedimento permite identificar e quantificar as variações ocorridas em um experimento, discriminando a parte da variação associada ao modelo pelo qual o experimento foi procedido da variação que se dá devido ao acaso. No R encontram-se diversos procedimentos para se executar a ANOVA. A tabela abaixo mostra alguns modelos e suas usuais formulações:

Modelo	Fórmula	Comentário
DIC	$y \sim t$	t é um fator
DBC	$y \sim t + b$	t e b são fatores
DQL	$y \sim t + l + c$	t , l e c são fatores
Fatorial/ DIC	$y \sim N * P$	igual a $N + P + N:P$
Fatorial/ DBC	$y \sim b + N * P$	igual a $b + N + P + N:P$
Regressão linear simples	$y \sim x$	x é uma variável exploratória
Regressão quadrática	$y \sim x + x^2$	x^2 é um objeto $x^2 < -x^2$

Os modelos de análise de variância simples podem ser trabalhados também a partir da função `lm()` simplesmente indicando um fator como variável independente. No entanto, existem outras funções que também realizam análises de variância para casos específicos (como a `aov()` para amostras balanceadas) ou modelos lineares mais complexos como análise de modelos mistos ou hierárquicas função `lme()` do pacote `nlme` e modelos não lineares `nls()` e `glm()` para ANOVA, com estrutura de erros especificada.

8.1 Delineamento inteiramente casualizado

O delineamento inteiramente casualizado é o mais simples de todos os delineamentos experimentais.

Este delineamento apresenta as seguintes características:

- Utiliza apenas os princípios da *repetição* e da *casualização*, deixando de lado o princípio do *controle local*, e, portanto, as repetições não são organizadas em blocos;
- Os tratamentos são designados às parcelas de forma inteiramente casual, com números iguais ou diferentes repetições por tratamento.

Vantagens em relações a outros delineamentos

- é um delineamento bastante flexível, visto que o número de tratamentos e de repetições depende apenas do número de parcelas disponíveis;
- o número de repetições pode ser diferente de um tratamento para outro;
- a análise estatística é simples, mesmo quando o número de repetições por tratamento é variável;
- o número de grau de liberdade para o resíduo é o maior possível;

Desvantagens em relações a outros delineamentos

- exige homogeneidade total das condições experimentais;
- pode conduzir a uma estimativa de variância residual bastante alta, uma vez que, não se aplica o controle local;

Este delineamento é mais utilizado em experimentos de laboratório e nos ensaios com vasos, realizados dentro de casas de vegetação, nos quais as condições experimentais podem ser controladas. Nos experimentos realizados com vasos, estes devem ser cte mudados de posição, de forma interamente casual.

8.1.1 Análise de experimento em DIC

Dados de um experimento visando controle de pulgão (*Aphis gossypii Glover*) em cultura de pepino, instalado em delineamento inteiramente casualizado com 6 repetições. A resposta observada foi o número de pulgões após a aplicação de produtos indicados para seu controle. O primeiro passo é ler os dados.

Importando dados:

```
dados <- read.table("https://www.dropbox.com/s/jjyo8dhyy0qt3ft/BanzattoQd3.2.1.txt?dl=1")
```

Conferir se temos fatores para fazer a análise de variância:

```
str(dados)
```

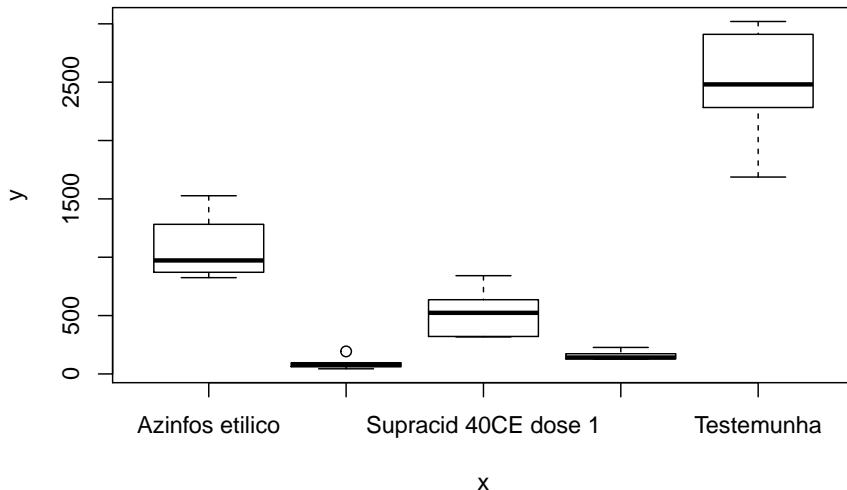
```
## 'data.frame':    30 obs. of  3 variables:
## $ trat   : Factor w/ 5 levels "Azinfos etilico",...: 5 1 3 4 2 5 1 3 4 2 ...
## $ rept   : int  1 1 1 1 1 2 2 2 2 2 ...
## $ pulgoes: int  2370 1282 562 173 193 1687 1527 321 127 71 ...
```

Lembramos que pulgões deve ter conteúdo numérico e trat deve ser fator:

```
dados$trat<-as.factor(dados$trat)
dados$pulgoes<-as.numeric(dados$pulgoes)
```

Verificação gráfica** É importante notar que as barras simplesmente refletem a variância dos dados dentro de cada tratamento e não são adequadas para detectar diferenças entre tratamentos:

```
plot(dados$trat, dados$pulgoes)
```



8.1.2 Análise de variância

Fazendo a análise de variância:

```
m0 <- lm (dados$pulgoes ~ dados$trat)
```

```
anova(m0)
```

```
## Analysis of Variance Table
##
## Response: dados$pulgoes
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## dados$trat  4 23145259 5786315   81.79 5.505e-14 ***
## Residuals  25 1768644   70746
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

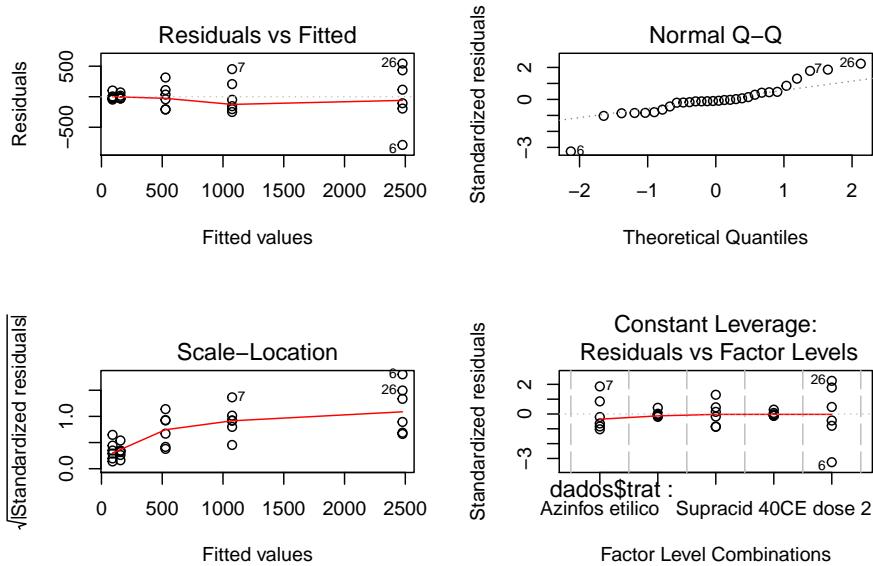
Extraindo o coeficiente de variação:

```
require(agricolae)
cv.model(m0)
```

```
## [1] 30.73619
```

Análise gráfica dos resíduos:

```
par(mfrow = c(2,2))
plot(m0)
```



Analizando a figura acima sugere que o principal problema deste conjunto de dados pode ser a heterosdasticidade. A variabilidade dos erros apresentam um forma de cone.

Teste das pressuposições da análise de variância**:

```
# Teste de Bartlett para homocedasticidade
bartlett.test(m0$res, dados$strat)
```

```
##
##  Bartlett test of homogeneity of variances
##
##  data: m0$res and dados$strat
##  Bartlett's K-squared = 29.586, df = 4, p-value = 5.942e-06
```

Como observamos uma significância estatística neste resultado, devemos rejeitar a hipótese nula de que as variâncias sejam as mesmas em todos os níveis do fator.

Teste de Shapiro-Wilk para Normalidade:

```
shapiro.test(m0$res)

##
##  Shapiro-Wilk normality test
##
##  data: m0$res
##  W = 0.89608, p-value = 0.006742
```

8.1.3 Transformação de dados

Transformação de dados é uma das possíveis formas de contornar o problema de dados que não obedecem os pressupostos da análise de variância. Vamos ver como isto poder ser feito com o programa R.

Nos resultados acima vemos que a homogeneidade de variâncias foi rejeitada. Para tentar contornar o problema podemos utilizar catálogos de transformação:

- Transformação de raiz quadrada: frequentemente utilizada para dados de contagens, que geralmente seguem a distribuição de Poisson, na qual a média é igual a variância. Quando ocorrem zeros ou valores baixos (menores que 10 ou 15), as transformações recomendadas são raiz quadrada + 0,5 ou 1,0.
- Transformação angular: recomendável para dados expressos em porcentagens, que geralmente segue distribuição binomial.
- Transformação logarítmica: utilizada quando é constatada certa proporcionalidade entre as médias e os desvios padrões dos diversos tratamentos:

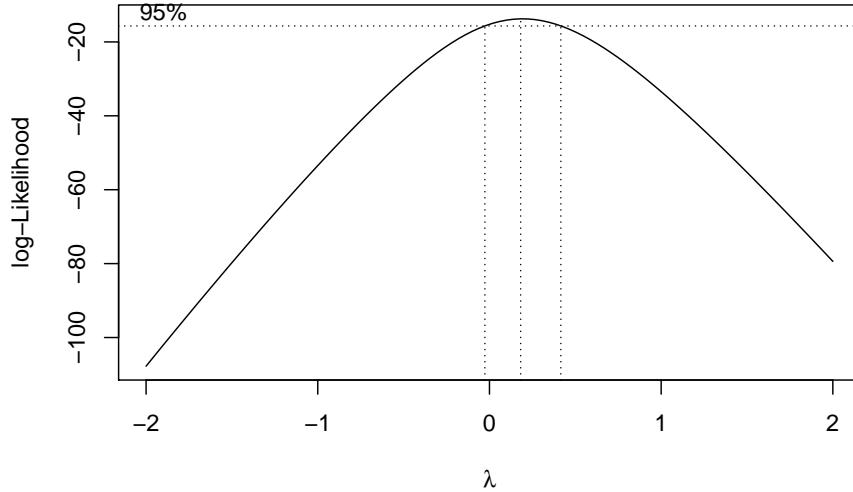
```
m1 <- lm (log(dados$pulgoes) ~ dados$trat)
```

8.1.3.1 Transformação de dados BOX-COX

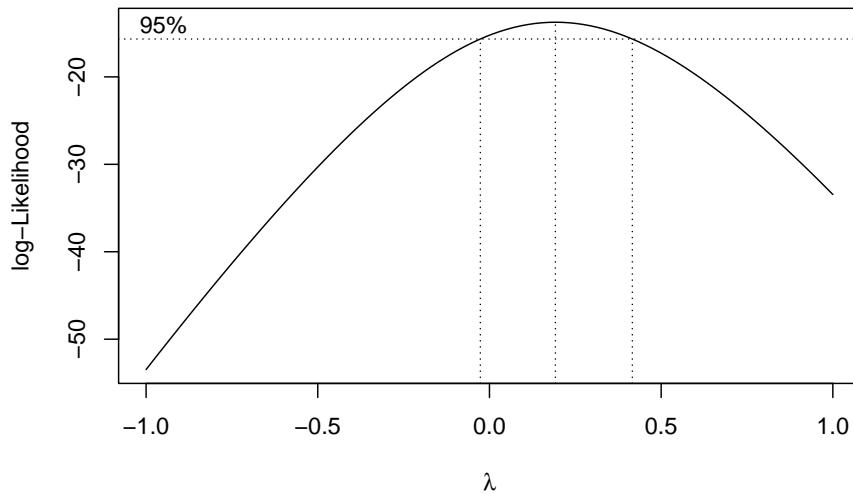
Para tentar contornar o problema vamos usar a transformação Box-Cox, que consiste em transformar os dados de acordo com uma expressão.

A função `boxcox()` do pacote MASS calcula a verossimilhança perfilhada do parâmetro. Devemos escolher o valor que maximiza esta função. Nos comandos a seguir começamos carregando o pacote MASS e depois obtemos o gráfico da verossimilhança perfilhada. Como estamos interessados no máximo fazermos um novo gráfico com um zoom na região de interesse:

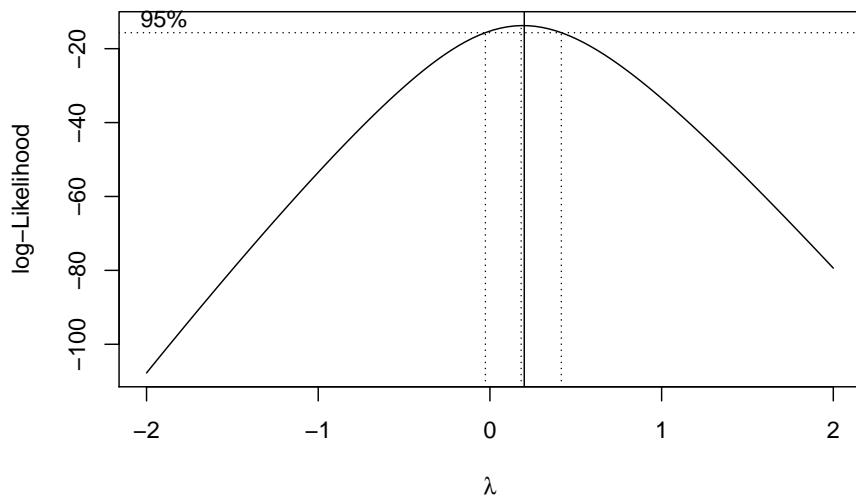
```
require(MASS)
boxcox(m0)
```



```
#Com zoom  
boxcox(m0, lam = seq(-1, 1, 1/10))
```



```
boxcox(m0)
abline(v=0.2)
locator()
```

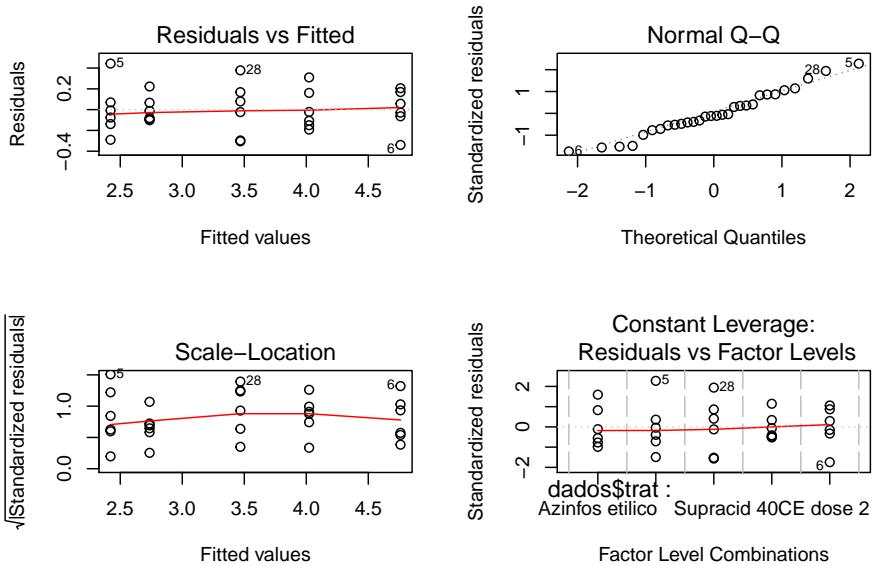


8.1.3.2 Análise de variância - Ajuste com a variável transformada.

```
m1 <- lm (dados$pulgoes^0.2 ~ dados$trat, data = dados)
```

análise gráfica dos resíduos

```
par(mfrow = c(2,2))
plot(m1)
```



Os pressupostos foram atendidos

Teste de Shapiro-Wilk para Normalidade:

```
shapiro.test(m1$res)
```

```
##
##  Shapiro-Wilk normality test
##
## data: m1$res
## W = 0.97288, p-value = 0.6207
```

Teste de Bartlett para homocedasticidade:

```
bartlett.test(m1$res, dados$strat)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data: m1$res and dados$strat
## Bartlett's K-squared = 2.7641, df = 4, p-value = 0.598
```

Resumo do modelo ajustado - Contraste :

```
summary(m1)

##
## Call:
## lm(formula = dados$pulgoes^0.2 ~ dados$trat, data = dados)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.33971 -0.10602 -0.02286  0.14069  0.44317
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                4.02268   0.08709  46.191 < 2e-16 ***
## dados$tratDiazinon 60CE    -1.60093   0.12316 -12.999 1.27e-12 ***
## dados$tratSupracid 40CE dose 1 -0.55418   0.12316 -4.500 0.000136 ***
## dados$tratSupracid 40CE dose 2 -1.28606   0.12316 -10.442 1.33e-10 ***
## dados$tratTestemunha        0.73703   0.12316   5.984 3.00e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2133 on 25 degrees of freedom
## Multiple R-squared:  0.95, Adjusted R-squared:  0.942
## F-statistic: 118.8 on 4 and 25 DF,  p-value: 6.978e-16
```

Resumo do modelo ajustado - Contraste em relação a testemunha:

```
dados$trat <- relevel(dados$trat, "Testemunha")

m1 <- lm (dados$pulgoes^0.2 ~ dados$trat, data = dados)

summary(m1)

##
## Call:
## lm(formula = dados$pulgoes^0.2 ~ dados$trat, data = dados)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.33971 -0.10602 -0.02286  0.14069  0.44317
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                4.75972   0.08709  54.654 < 2e-16 ***
```

```

## dados$tratAzinfos etilico      -0.73703   0.12316 -5.984 3.00e-06 ***
## dados$tratDiazinon 60CE       -2.33797   0.12316 -18.983 2.31e-16 ***
## dados$tratSupracid 40CE dose 1 -1.29122   0.12316 -10.484 1.23e-10 ***
## dados$tratSupracid 40CE dose 2 -2.02310   0.12316 -16.426 6.62e-15 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2133 on 25 degrees of freedom
## Multiple R-squared:  0.95, Adjusted R-squared:  0.942
## F-statistic: 118.8 on 4 and 25 DF, p-value: 6.978e-16

```

8.1.4 Aplicando teste de Tukey para comparar médias

No R, o teste de Tukey é apresentado através de intervalos de confiança. A interpretação é: se o intervalo de confiança para a diferença entre duas médias não incluir o valor zero, significa que se rejeita a hipótese nula, caso contrário, não se rejeita. O resultado pode ser visto através de uma tabela e/ou graficamente:

```

m1 <- aov (dados$pulgoes^0.2 ~ dados$trat, data = dados)

dados.tu <- TukeyHSD (m1)

print(dados.tu)

## Tukey multiple comparisons of means
##      95% family-wise confidence level
##
## Fit: aov(formula = dados$pulgoes^0.2 ~ dados$trat, data = dados)
##
## $`dados$trat`
##                                     diff      lwr      upr
## Azinfos etilico-Testemunha    -0.7370340 -1.09874544 -0.3753225
## Diazinon 60CE-Testemunha     -2.3379681 -2.69967961 -1.9762567
## Supracid 40CE dose 1-Testemunha -1.2912155 -1.65292701 -0.9295041
## Supracid 40CE dose 2-Testemunha -2.0230984 -2.38480982 -1.6613869
## Diazinon 60CE-Azinfos etilico -1.6009342 -1.96264562 -1.2392227
## Supracid 40CE dose 1-Azinfos etilico -0.5541816 -0.91589302 -0.1924701
## Supracid 40CE dose 2-Azinfos etilico -1.2860644 -1.64777584 -0.9243529
## Supracid 40CE dose 1-Diazinon 60CE    1.0467526  0.68504114  1.4084641
## Supracid 40CE dose 2-Diazinon 60CE    0.3148698 -0.04684168  0.6765812
## Supracid 40CE dose 2-Supracid 40CE dose 1 -0.7318828 -1.09359428 -0.3701714
##                                     p adj
## Azinfos etilico-Testemunha      0.0000277
## Diazinon 60CE-Testemunha       0.0000000

```

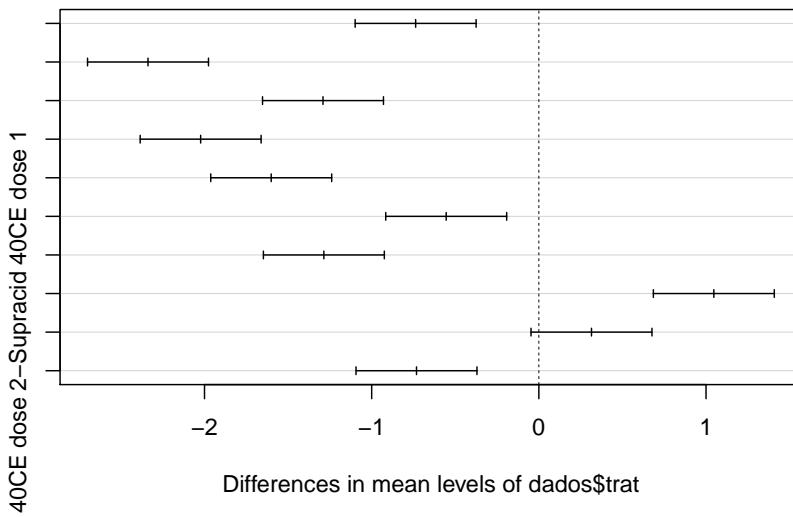
```

## Supracid 40CE dose 1-Testemunha          0.0000000
## Supracid 40CE dose 2-Testemunha          0.0000000
## Diazinon 60CE-Azinfos etilico           0.0000000
## Supracid 40CE dose 1-Azinfos etilico     0.0011809
## Supracid 40CE dose 2-Azinfos etilico     0.0000000
## Supracid 40CE dose 1-Diazinon 60CE       0.0000001
## Supracid 40CE dose 2-Diazinon 60CE       0.1099096
## Supracid 40CE dose 2-Supracid 40CE dose 1 0.0000307

```

```
plot(dados.tu)
```

95% family-wise confidence level



Teste Tukey com pacote Agricolae

```
library(agricolae)
```

```
summary(m1)
```

```

##             Df Sum Sq Mean Sq F value    Pr(>F)
## dados$trat   4 21.629   5.407   118.8 6.98e-16 ***
## Residuals   25  1.138   0.046
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

tu <- HSD.test(y = dados$pulgoes^0.2,
                 trt = dados$trat,
                 MSerror = deviance(m1)/df.residual(m1), #quadrado médio do residual
                 DFerror = df.residual(m1),
                 console = T)

## 
## Study: dados$pulgoes^0.2 ~ dados$trat
##
## HSD Test for dados$pulgoes^0.2
##
## Mean Square Error:  0.04550677
##
## dados$trat,  means
##
##          dados.pulgoes.0.2      std r      Min      Max
## Azinfos etilico        4.022682 0.1967278 6 3.830812 4.332792
## Diazinon 60CE          2.421748 0.2492525 6 2.131526 2.864913
## Supracid 40CE dose 1   3.468500 0.2678917 6 3.163821 3.846471
## Supracid 40CE dose 2   2.736617 0.1264052 6 2.634879 2.959410
## Testemunha            4.759716 0.1973855 6 4.420008 4.965939
##
## Alpha: 0.05 ; DF Error: 25
## Critical Value of Studentized Range: 4.153363
##
## Minimun Significant Difference: 0.3617115
##
## Treatments with the same letter are not significantly different.
##
##          dados$pulgoes^0.2 groups
## Testemunha           4.759716    a
## Azinfos etilico       4.022682    b
## Supracid 40CE dose 1  3.468500    c
## Supracid 40CE dose 2  2.736617    d
## Diazinon 60CE         2.421748    d

str(tu)

## List of 5
## $ statistics:'data.frame': 1 obs. of  5 variables:
##   ..$ MSerror: num 0.0455
##   ..$ Df     : int 25
##   ..$ Mean   : num 3.48
##   ..$ CV     : num 6.13

```

```

##   ..$ MSD      : num 0.362
## $ parameters:'data.frame': 1 obs. of 5 variables:
##   ..$ test       : Factor w/ 1 level "Tukey": 1
##   ..$ name.t    : Factor w/ 1 level "dados$trat": 1
##   ..$ ntr       : int 5
##   ..$ StudentizedRange: num 4.15
##   ..$ alpha     : num 0.05
## $ means      :'data.frame': 5 obs. of 8 variables:
##   ..$ dados$pulgoes^0.2: num [1:5] 4.02 2.42 3.47 2.74 4.76
##   ..$ std        : num [1:5] 0.197 0.249 0.268 0.126 0.197
##   ..$ r         : int [1:5] 6 6 6 6 6
##   ..$ Min       : num [1:5] 3.83 2.13 3.16 2.63 4.42
##   ..$ Max       : num [1:5] 4.33 2.86 3.85 2.96 4.97
##   ..$ Q25      : num [1:5] 3.88 2.3 3.24 2.65 4.7
##   ..$ Q50      : num [1:5] 3.96 2.38 3.5 2.69 4.77
##   ..$ Q75      : num [1:5] 4.14 2.47 3.61 2.78 4.9
## $ comparison: NULL
## $ groups     :'data.frame': 5 obs. of 2 variables:
##   ..$ dados$pulgoes^0.2: num [1:5] 4.76 4.02 3.47 2.74 2.42
##   ..$ groups      : Factor w/ 4 levels "a","b","c","d": 1 2 3 4 4
## - attr(*, "class")= chr "group"

```

`tu$groups`

```

##                               dados$pulgoes^0.2 groups
## Testemunha                  4.759716     a
## Azinfos etilico              4.022682     b
## Supracid 40CE dose 1         3.468500     c
## Supracid 40CE dose 2         2.736617     d
## Diazinon 60CE                2.421748     d

write.table(tu$groups,
            "tab.xls",
            sep = "\t",
            quote = F,
            row.names = F)

```

8.1.5 Aplicando teste para agrupar médias

carregando a biblioteca necessária

```

#install.packages("laercio")
require(laercio)

```

Teste de Duncan:

```
LDuncan (m1, "dados$trat")
```

```
##
## DUNCAN TEST TO COMPARE MEANS
##
## Confidence Level: 0.95
## Dependent Variable: dados$pulgoes^0.2
## Variation Coefficient: 6.126714 %
##
##
## Independent Variable: dados$trat
## Factors Means
## Testemunha 4.75971565312899 a
## Azinfos etilico 4.02268167030532 b
## Supracid 40CE dose 1 3.46850010815533 c
## Supracid 40CE dose 2 2.73661729164913 d
## Diazinon 60CE 2.42174750727391 e
```

```
LTukey (m1, "dados$trat")
```

```
##
## TUKEY TEST TO COMPARE MEANS
##
## Confidence level: 0.95
## Dependent variable: dados$pulgoes^0.2
## Variation Coefficient: 6.126714 %
##
## Independent variable: dados$trat
## Factors Means
## Testemunha 4.75971565312899 a
## Azinfos etilico 4.02268167030532 b
## Supracid 40CE dose 1 3.46850010815533 c
## Supracid 40CE dose 2 2.73661729164913 d
## Diazinon 60CE 2.42174750727391 d
##
##
```

Pacote para análise de experimentos:

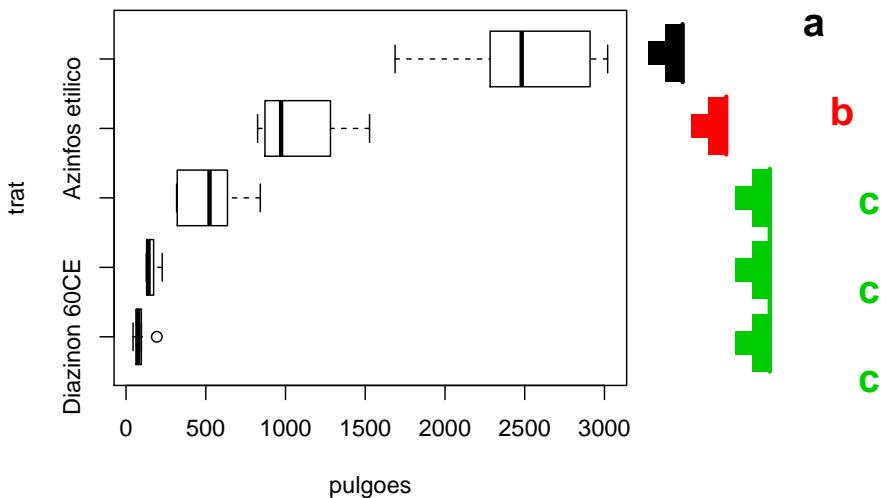
```
library(ExpDes.pt)
```

Recursos adicionais para comparações múltiplas

Outros procedimentos serão implementados em pacotes contribuídos do R. Entre estes encontra-se os pacotes multcomp e multcompView que implementam diversos outros procedimentos e gráficos para visualizações dos resultados. Vale notar que estes pacotes devem ser instalados com a opção `dependencies=TRUE` para garantir plena funcionalidade pois suas funções dependem de diversos outros pacotes:

```
#install.packages("multcompView", dep = TRUE)
require(multcomp)
require(multcompView)
```

```
multcompBoxplot(pulgoes ~ trat, data = dados, compFn = "TukeyHSD", decreasing = FALSE)
```



8.1.6 Referência

MELO, M. P.; PETERNELI, L. A. Conhecendo o R: Um visão mais que estatística. Viçosa, MG: UFV, 2013. 222p. Cap. 1.

BANZATTO, D. A; KRONKA, S. N. Experimentação agrícola. Jaboticabal, SP: FUNEP, 2006, 237p.

ZEVIANI, W. M. estatística Básica e Experimentação no R. 45p.

<http://www.leg.ufpr.br/~paulojus/>

8.2 Delineamento em bloco casualizado

O delineamento em blocos casualizados (DBC) tem três princípios básicos de experimentação:

- repetição
- casualização
- controle local

É o delineamento mais utilizado de todos os delineamentos. Ele é utilizado quando há heterogeneidade nas condições experimentais. Nesse caso divide-se o material experimental, ou amostra, em bloco homogêneos de forma a contemplar as diferenças entre grupos. A ANOVA associada a este modelo de experimento é também conhecida como *Two Way ANOVA*.

8.2.1 Análise de experimento DBC

Resultados de um experimento instalado na Fazenda Chapadão, no município de Angatuba - SP. O delineamento experimental foi o de blocos casualizados, sendo as parcelas constituídas de 4 plantas espaçadas de 6 x 7 metros, com 12 anos de idade na época da instalação do experimento.

Importando dados:

```
dados <- read.table("https://www.dropbox.com/s/9woiye3ce9twp78/BanzattoQd4.5.2.txt?dl=1")
```

conferir se temos fatores para fazer a análise de variância:

```
str(dados)
```

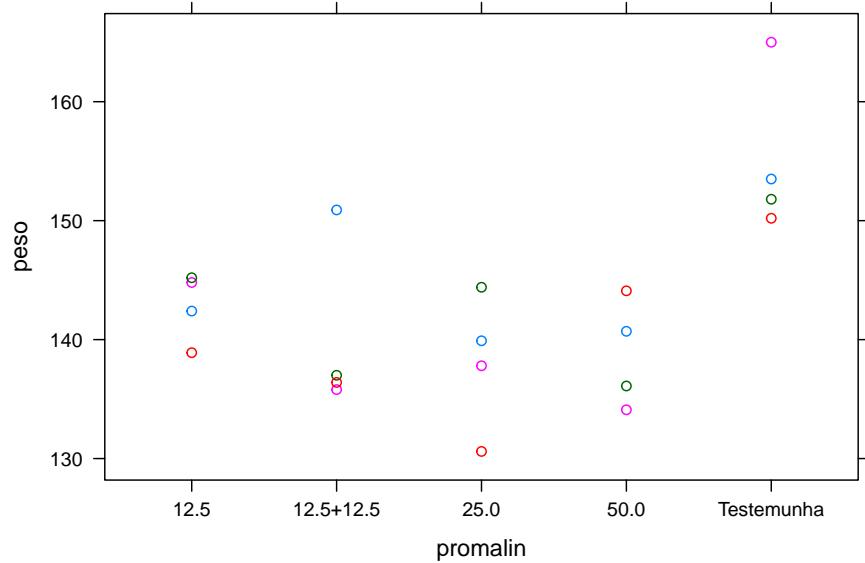
```
## 'data.frame': 20 obs. of 3 variables:
## $ promalin: Factor w/ 5 levels "12.5","12.5+12.5",...: 1 3 4 2 5 1 3 4 2 5 ...
## $ bloco   : Factor w/ 4 levels "I","II","III",...: 1 1 1 1 1 2 2 2 2 ...
## $ peso    : num 142 140 141 151 154 ...
```

Lembramos que o *peso* deve ter conteúdo numérico e o *promalin* e *bloco* deve ser fator:

```
dados$promalin<-as.factor(dados$promalin)
dados$bloco<-as.numeric(dados$bloco)
```

Verificação gráfica**:

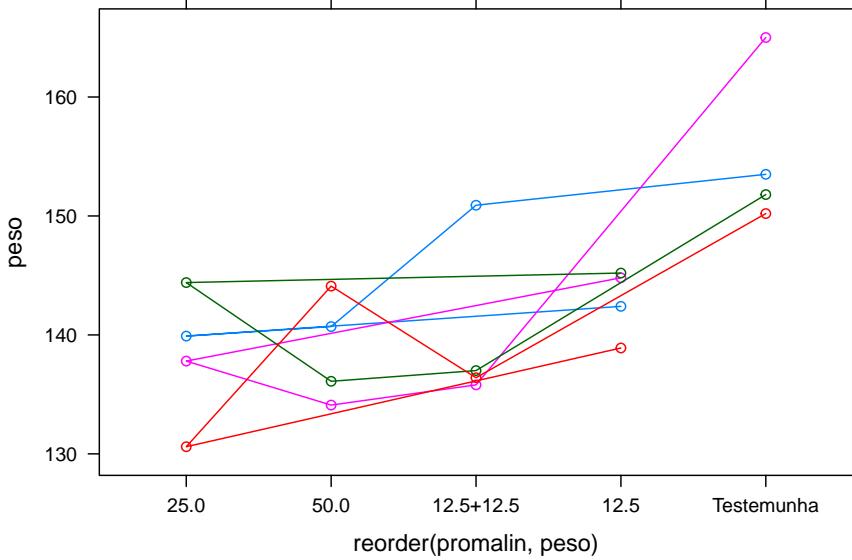
```
require(lattice)
xyplot(peso ~ promalin,
       groups = bloco,
       data= dados)
```



O efeito do bloco é aditivo?

Ligar as observações com o mesmo bloco com a função `type = "o"`:

```
xyplot(peso ~ reorder(promalin, peso),
       groups = bloco,
       data= dados,
       type = "o")
```

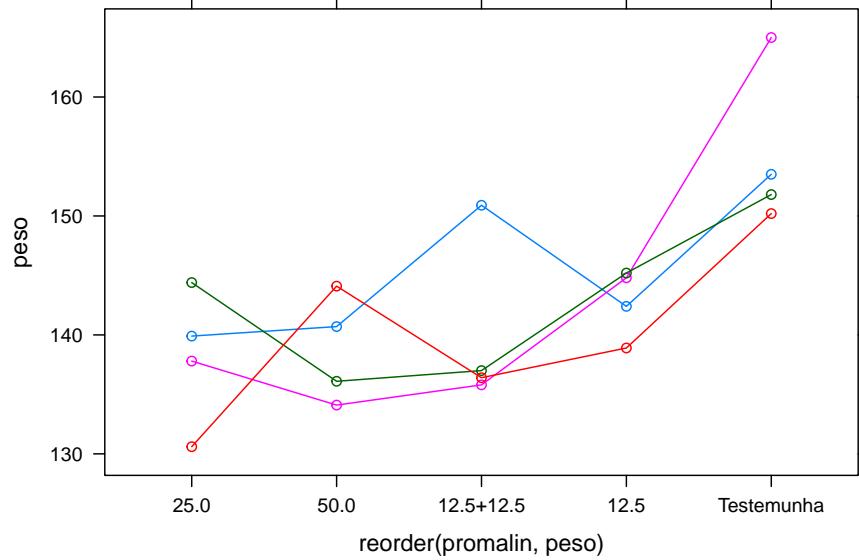


Reordenar os tratamentos:

```
require(plyr)
dados$promalin <- with(dados, reorder(promalin, peso))
dados <- arrange(dados, promalin, bloco)
```

Gráficos reordenados de menor média a maior média por tratamento:

```
xyplot(peso ~ reorder(promalin, peso),
       groups = bloco,
       data= dados,
       type = "o")
```



8.2.1.1 Análise de variância

Fazendo a análise de variância:

```
m0 <- lm (dados$peso ~ dados$bloco + dados$promalin, data = dados)
```

```
anova(m0)
```

```
## Analysis of Variance Table
##
## Response: dados$peso
##              Df Sum Sq Mean Sq F value    Pr(>F)
## dados$bloco     1  71.57  71.572  2.4574 0.139291
## dados$promalin  4 788.95 197.238  6.7721 0.002994 **
## Residuals      14 407.75  29.125
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

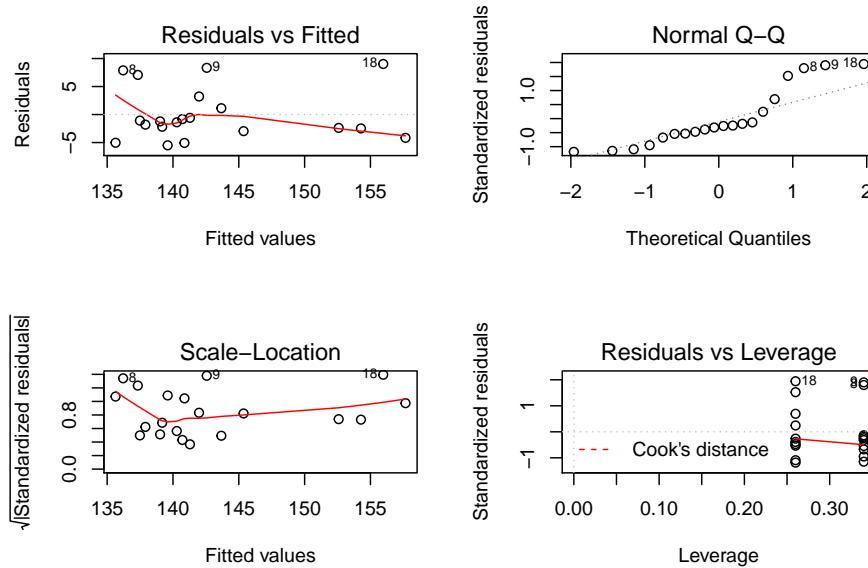
Extraíndo o coeficiente de variação:

```
require(agricolae)
cv.model(m0)
```

```
## [1] 3.774477
```

Análise gráfica dos resíduos:

```
par(mfrow= c(2,2))
plot(m0)
```



Analizando a figura acima sugere que o principal problema deste conjunto de dados pode ser a não normalidade.

8.2.1.1.1 Teste das pressuposições da análise de variância Teste de Bartlett para homocedasticidade:

```
bartlett.test(m0$res, dados$promalin)
```

```
##
##  Bartlett test of homogeneity of variances
##
##  data: m0$res and dados$promalin
##  Bartlett's K-squared = 1.7485, df = 4, p-value = 0.7819
```

Como observamos uma não significância estatística neste resultado ($p\text{-value} = 0.7819$), devemos aceitar a hipótese nula de que as variâncias sejam as mesmas em todos os níveis do fator.

Teste de Shapiro-Wilk para Normalidade:

```
shapiro.test(m0$res)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: m0$res  
## W = 0.855, p-value = 0.006472
```

Como observamos uma significância estatística neste resultado ($p\text{-value} = 0.006472$), devemos rejeitar a hipótese nula de que os resíduos têdem a distribuição normal.

8.2.1.2 Transformação de dados

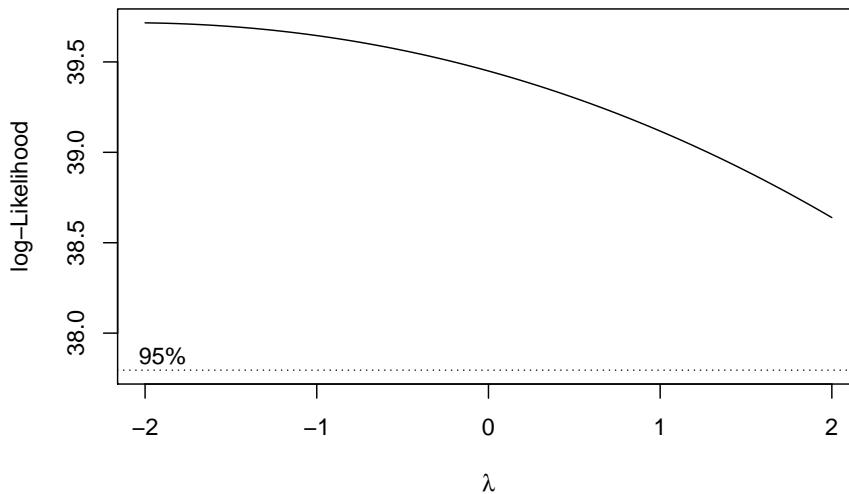
Transformação de dados é uma das possíveis formas de contornar o problema de dados que não obedecem os pressupostos da análise de variância. Vamos ver como isto poder ser feito com o programa R.

8.2.1.2.1 Transformação de dados com o BOX-COX

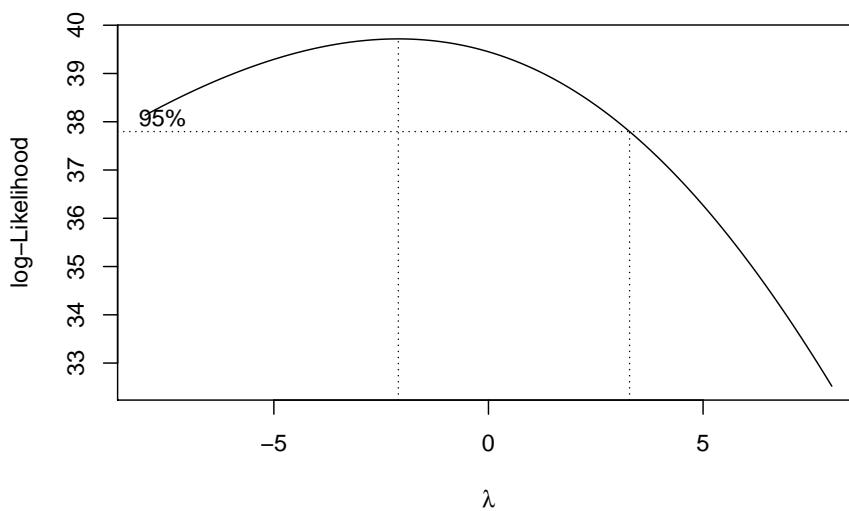
Para tentar tornar o problema vamos usar a transformação Box-Cox, que consiste em transformar os dados de acordo com uma expressão.

A função `boxcox()` do pacote MASS calcula a verossimilhança perfilhada do parâmetro lambda. Devemos escolher o valor que maximiza esta função. Nos comandos a seguir começamos carregando o pacote MASS e depois obtemos o gráfico da verossimilhança perfilhada. Como estamos interessados no máximo, vamos fazer um novo gráfico com um zoom na região de interesse:

```
require(MASS)  
boxcox(m0)
```



```
boxcox(m0, lam = seq(-8, 8, 1/10))
```



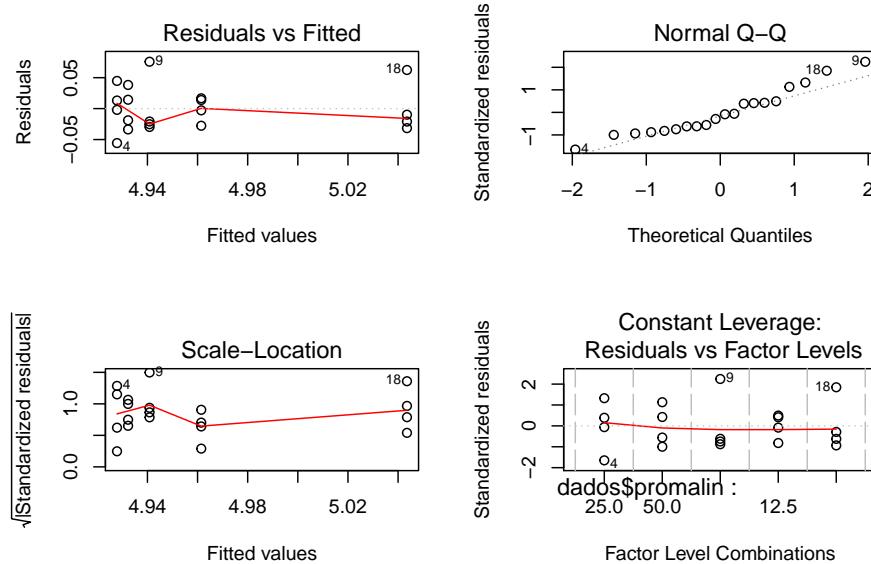
Localizando o ponto máximo:

```
m1 <- aov (log(dados$peso) ~ dados$promalin, data = dados)
```

8.2.1.2.2 Análise de variância - Ajuste com a variável transformada.

Anáise gráfica dos resíduos:

```
par(mfrow = c(2,2))
plot(m1)
```



Os pressupostos foram atendidos?

Teste de Shapiro-Wilk para Normalidade:

```
shapiro.test(m1$res)
```

```
##
##  Shapiro-Wilk normality test
##
##  data:  m1$res
##  W = 0.93909, p-value = 0.2305
```

Teste de Bartllet para homocedasticidade:

```

bartlett.test(m1$res, dados$promalin)

##
##  Bartlett test of homogeneity of variances
##
##  data:  m1$res and dados$promalin
##  Bartlett's K-squared = 2.1761, df = 4, p-value = 0.7034

anova(m1)

## Analysis of Variance Table
##
## Response: log(dados$peso)
##             Df  Sum Sq  Mean Sq F value    Pr(>F)
## dados$promalin  4 0.036571 0.0091428 6.0129 0.004296 ***
## Residuals      15 0.022808 0.0015205
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

8.2.2 Pacote para analise de experimentos

```
library(ExpDes.pt)
```

Conhecer o pacote ExpDes.pt:

```
ls("package:ExpDes.pt")
```

```

## [1] "anscombetukey"   "bartlett"        "ccboot"          "ccf"
## [5] "dbc"              "dic"              "dql"              "duncan"
## [9] "faixas"           "fat2.addbc"       "fat2.ad.dic"     "fat2dbc"
## [13] "fat2dic"          "fat3.addbc"       "fat3.ad.dic"     "fat3dbc"
## [17] "fat3dic"          "ginv"             "graficos"         "han"
## [21] "lastC"            "layard"           "levene"           "lsd"
## [25] "lsdb"              "oneilldbc"         "oneillmathews"   "order.group"
## [29] "order.stat.SNK"  "plotres"          "psub2dbc"         "psub2dic"
## [33] "reg.nl"            "reg.poly"         "samiuddin"        "scottknott"
## [37] "snk"               "tapply.stat"       "tukey"

```

Utilizando o exemplo anterior:

```

x <- dbc(trat = dados$promalin,
          bloco = dados$bloco,
          resp = log(dados$peso),
          quali = T,
          mcomp = "tukey")

## -----
## Quadro da analise de variancia
## -----
##      GL      SQ      QM      Fc    Pr>Fc
## Tratamento 4 0.036571 0.0091428 5.7552 0.00800
## Bloco      3 0.003745 0.0012483 0.7858 0.52459
## Residuo    12 0.019063 0.0015886
## Total     19 0.059379
## -----
## CV = 0.8 %
## 
## -----
## Teste de normalidade dos residuos
## valor-p: 0.005994506
## ATENCAO: a 5% de significancia, os residuos nao podem ser considerados normais!
## 
## -----
## Teste de homogeneidade de variancia
## valor-p: 0.8927087
## De acordo com o teste de oneillmathews a 5% de significancia, as variancias podem se
## 
## -----
## Teste de Tukey
## -----
## Grupos Tratamentos Medias
## a      Testemunha   5.043544
## ab    12.5    4.961465
## b    12.5+12.5  4.940843
## b    50.0    4.932278
## b    25.0    4.927864
## -----

```

Carregar pacotes:

```

library(ggplot2)
library(dplyr)

```

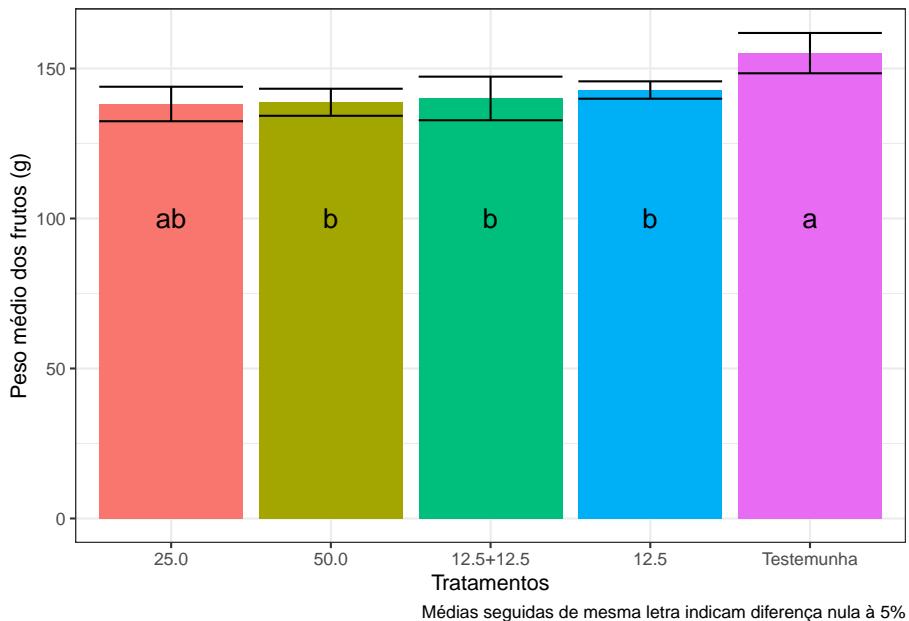
Cálculo do erro:

```
erro = summarise(group_by(dados, promalin),
                 avg = mean(peso), sd = sd(peso))
```

Gerando o gráfico:

```
ggplot(erro, aes(promalin, avg, fill=promalin))+  
  geom_bar(stat="identity") +  
  geom_errorbar(aes(ymin=avg-sd, ymax =avg+sd), width=0.1, col="black") +  
  xlab("Tratamentos") +  
  ylab("Peso médio dos frutos (g)") +  
  theme_bw() +  
  theme(legend.position="top") +  
  annotate("text", label="ab", x=1, y=100, size = 5) +  
  annotate("text", label="b", x=2, y=100, size = 5) +  
  annotate("text", label="b", x=3, y=100, size = 5) +  
  annotate("text", label="b", x=4, y=100, size = 5) +  
  annotate("text", label="a", x=5, y=100, size = 5) +  
  theme(legend.position="none") +  
  labs(caption = "Médias seguidas de mesma letra indicam diferença nula à 5%")
```

Warning: Ignoring unknown parameters: with



8.2.3 Teste não parametrico

As funções para comparações multiplas não-paramétricas incluídas no pacote agricolae são: **kruskal**, **waerden.test**, **friedman**, **durbin.test** e **Conover (1999)**. Os testes não-paramétricos post hoc (kruskal, friedman, durbin e waerden) estão usando o critério da diferença menos significativa de Fisher (LSD).

Carregar pacote:

```
library(agricolae)
```

A função **kruskal** é usada para N amostras ($N > 2$), populações ou dados provenientes de um experimento aleatório (populações = tratamentos):

```
woutKruskal<-with(dados,kruskal(promalin, y = peso
, p.adj="bon",group=T, console=T))

##
## Study: peso ~ promalin
## Kruskal-Wallis test's
## Ties or no Ties
##
## Critical Value: 10.41429
## Degrees of freedom: 4
## Pvalue Chisq : 0.03399839
##
## promalin, means of the ranks
##
##          peso r
## 12.5      12.00 4
## 12.5+12.5  7.75 4
## 25.0      7.50 4
## 50.0      7.00 4
## Testemunha 18.25 4
##
## Post Hoc Analysis
##
## P value adjustment method: bonferroni
## t-Student: 3.286039
## Alpha     : 0.05
## Minimum Significant Difference: 10.40002
##
## Treatments with the same letter are not significantly different.
##
##          peso groups
```

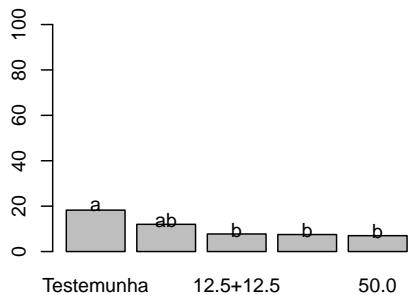
```
## Testemunha 18.25      a
## 12.5       12.00      ab
## 12.5+12.5  7.75      b
## 25.0       7.50      b
## 50.0       7.00      b
```

```
print(woutKruskal$group)
```

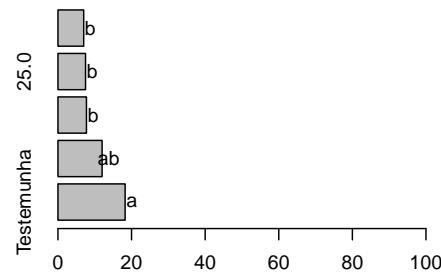
```
##           peso groups
## Testemunha 18.25      a
## 12.5       12.00      ab
## 12.5+12.5  7.75      b
## 25.0       7.50      b
## 50.0       7.00      b
```

Gráficos:

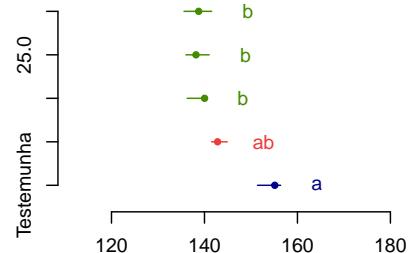
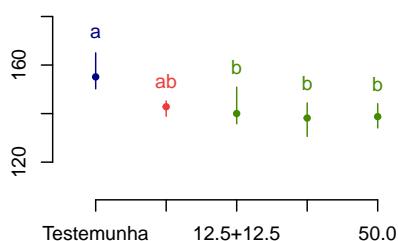
```
par(mfrow=c(2,2),mar=c(3,3,1,1),cex=0.8)
bar.group(woutKruskal$group,ylim=c(0,100), xlab ="promalin")
bar.group(woutKruskal$group,xlim=c(0,100),horiz = TRUE)
plot(woutKruskal)
plot(woutKruskal,variation="IQR",horiz = TRUE)
```



Groups and Range



Groups and Interquartile range



A função `friedman` é usada para análise de tratamentos do estudo randomizado de bloco completo, onde a resposta não pode ser tratada através da análise de variância:

```
woutfriedman <- out<-with(dados,friedman(bloco,promalin, peso,alpha=0.05, group=T,
  console=TRUE))

##  

## Study: peso ~ bloco + promalin  

##  

## promalin, Sum of the ranks  

##  

##          peso r  

## 12.5      8 4  

## 12.5+12.5 8 4  

## 25.0      10 4  

## 50.0      14 4  

## Testemunha 20 4  

##  

## Friedman's Test  

## =====  

## Adjusted for ties  

## Critical Value: 10.4  

## P.Value Chisq: 0.0342027  

## F Value: 5.571429  

## P.Value F: 0.009007502  

##  

## Post Hoc Analysis  

##  

## Alpha: 0.05 ; DF Error: 12  

## t-Student: 2.178813  

## LSD: 6.656383  

##  

## Treatments with the same letter are not significantly different.  

##  

##          Sum of ranks groups  

## Testemunha      20     a  

## 50.0          14    ab  

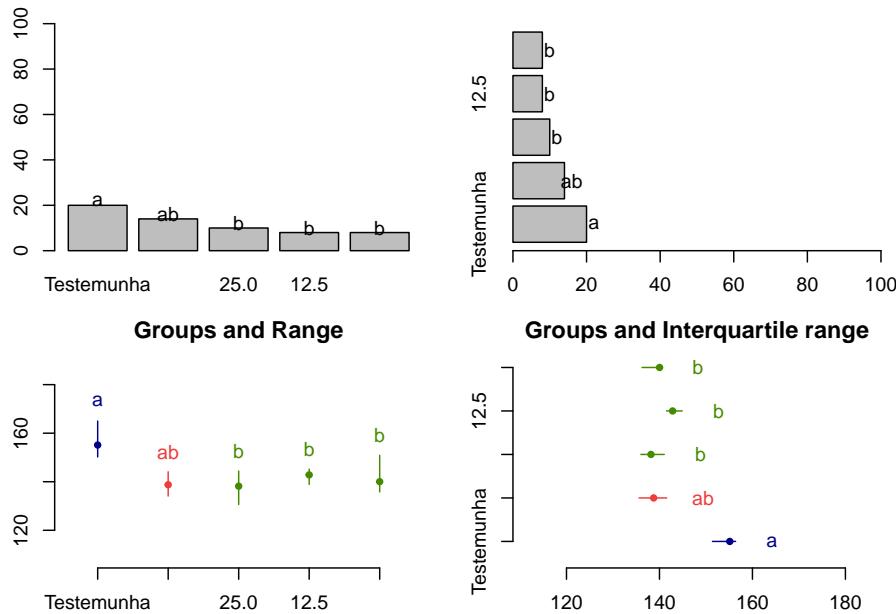
## 25.0          10     b  

## 12.5           8     b  

## 12.5+12.5     8     b
```

Gráfico:

```
par(mfrow=c(2,2),mar=c(3,3,1,1),cex=0.8)
bar.group(woutfriedman$group,ylim=c(0,100), xlab ="promalin")
bar.group(woutfriedman$group,xlim=c(0,100),horiz = TRUE)
plot(woutfriedman)
plot(woutfriedman,variation="IQR",horiz = TRUE)
```



8.2.4 Exercício 1

Obtenha: Análise exploratória, Análise de variância, teste de comparação múltipla, e recomendações.

Comparação de métodos de Semeadura do Mamoeiro

Estudo realizado em Jaboticabal - SP por Ruiz (1977) que comparou métodos de semeadura no mamoeiro. O experimento foi instalado em delineamento de blocos casualizados, com 4 repetições, avaliando 3 métodos de semeadura. Foram avaliadas duas unidades experimentais por método em cada bloco.

Importando dados:

```
dados <- read.table("https://www.dropbox.com/s/40m95attfw2fdh2/BanzattoQd4.7.1.txt?dl=1")
```

Conferir se temos fatores para fazer a análise de variância:

```
str(dados)
```

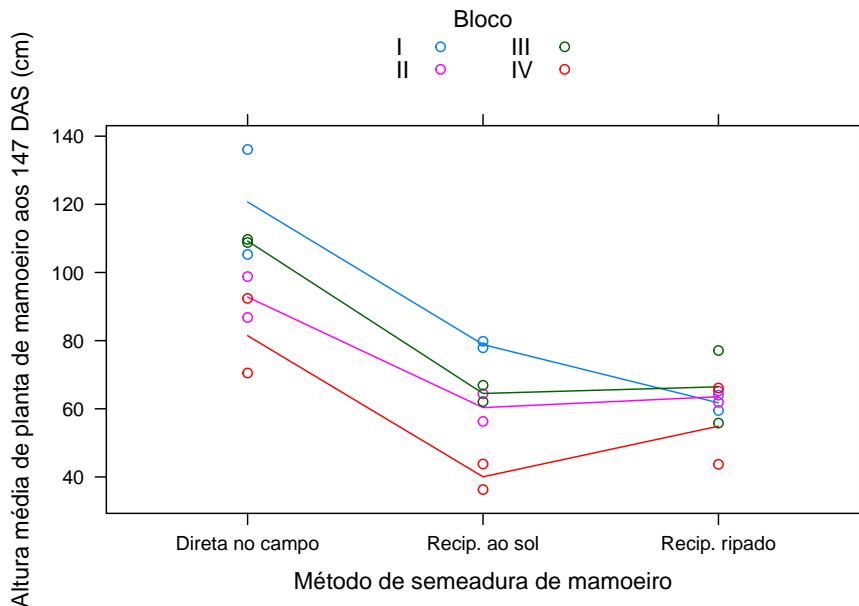
```
## 'data.frame': 24 obs. of 3 variables:
## $ bloco : Factor w/ 4 levels "I","II","III",...: 1 1 2 2 3 3 4 4 1 1 ...
## $ semead: Factor w/ 3 levels "Direta no campo",...: 1 1 1 1 1 1 1 1 2 2 ...
## $ altura: num 136.1 105.3 98.8 86.8 108.8 ...
```

Gráficos:

```
addmargins(with(dados,
  tapply(X = altura,
  INDEX = list(semead, bloco),
  FUN = sum)))
```

	I	II	III	IV	Sum
## Direta no campo	241.4	185.6	218.5	162.9	808.4
## Recip. ao sol	157.7	120.7	129.0	80.1	487.5
## Recip. ripado	123.5	127.1	132.9	109.8	493.3
## Sum	522.6	433.4	480.4	352.8	1789.2

```
xyplot(altura ~ semead, data = dados,
  groups = bloco, type = c("p", "a"),
  xlab = "Método de semeadura de mamoeiro",
  ylab = "Altura média de planta de mamoeiro aos 147 DAS (cm)",
  auto.key = list(title = "Bloco", cex.title = 1, columns = 2))
```



Análise de Variância:

```
m0 <- aov(altura~bloco+semead, data=dados)
class(m0)

## [1] "aov" "lm"

anova(m0)

## Analysis of Variance Table
##
## Response: altura
##              Df Sum Sq Mean Sq F value    Pr(>F)
## bloco       3 2648.2   882.7  7.2162 0.002219 ***
## semead      2 8429.1  4214.6 34.4535 7.014e-07 ***
## Residuals 18 2201.9   122.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

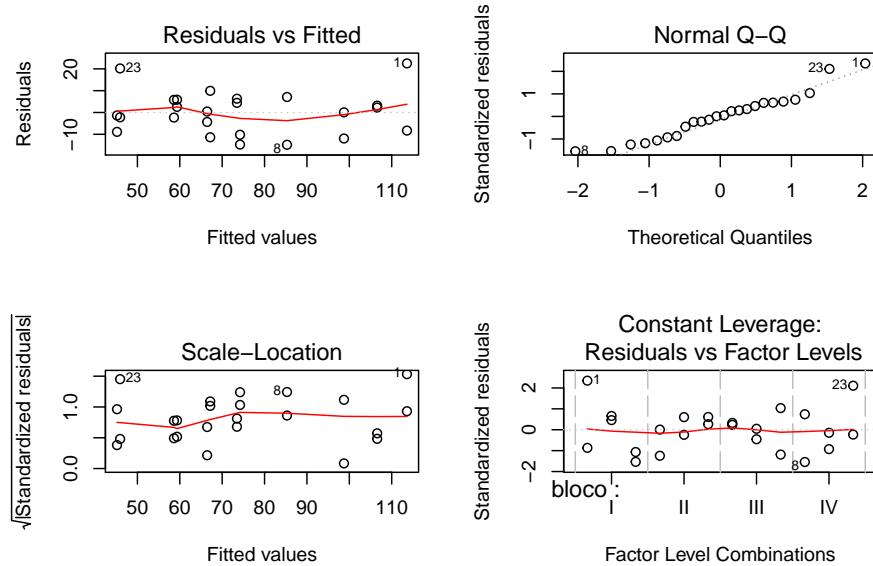
summary(m0)

##              Df Sum Sq Mean Sq F value    Pr(>F)
## bloco       3 2648     883     7.216  0.00222  **
##
```

```
## semead      2     8429     4215   34.453 7.01e-07 ***
## Residuals   18    2202      122
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Checagem gráfica:

```
par(mfrow=c(2,2))
plot(m0)
```



```
layout(1)
```

Teste das pressuposições de normalidade de homocedasticidade:

```
shapiro.test(residuals(m0))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(m0)
## W = 0.95197, p-value = 0.2988
```

```
bartlett.test(residuals(m0) ~ dados$semead)

##
##  Bartlett test of homogeneity of variances
##
##  data:  residuals(m0) by dados$semead
##  Bartlett's K-squared = 4.5219, df = 2, p-value = 0.1043
```

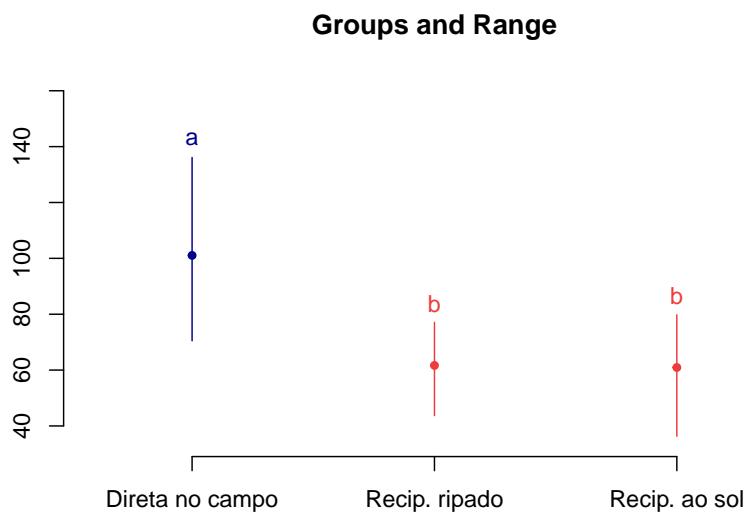
Teste de médias

Teste de Tukey

```
require(agricolae)

tu <- with(dados, HSD.test(altura, semead,
DFerror=df.residual(m0),
MSerror=deviance(m0)/df.residual(m0)))

plot(tu)
```



```
print(tu)
```

```

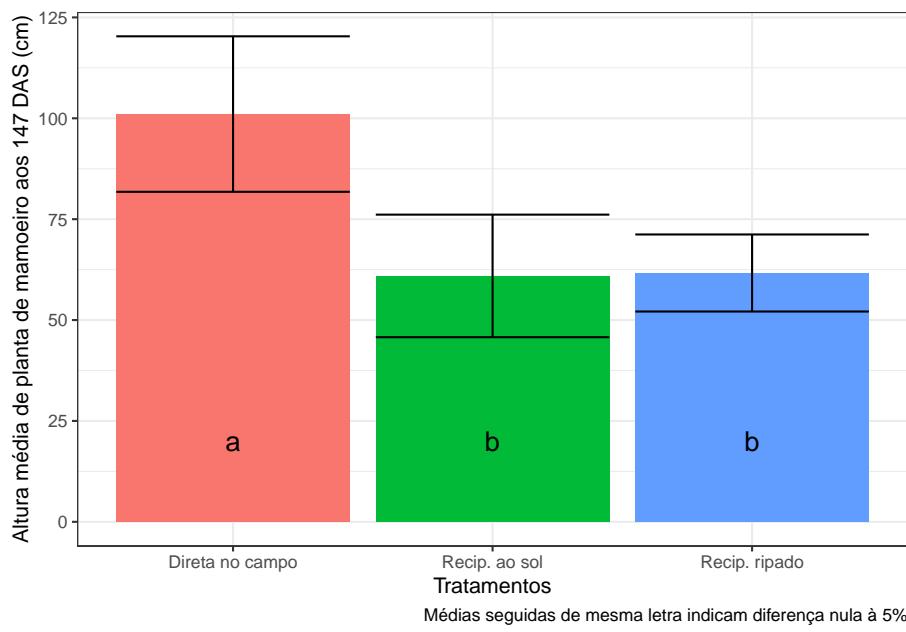
## $statistics
##      MSerror Df   Mean       CV      MSD
##    122.3258 18 74.55 14.83581 14.11359
##
## $parameters
##   test name.t ntr StudentizedRange alpha
##   Tukey semead  3        3.609304  0.05
##
## $means
##           altura     std r  Min   Max   Q25   Q50   Q75
## Direta no campo 101.0500 19.263956 8 70.5 136.1 91.000 102.05 109.025
## Recip. ao sol    60.9375 15.187489 8 36.3  79.8 53.175 63.25  69.650
## Recip. ripado    61.6625  9.544922 8 43.7  77.1 58.575 62.95  65.425
##
## $comparison
## NULL
##
## $groups
##           altura groups
## Direta no campo 101.0500      a
## Recip. ripado    61.6625      b
## Recip. ao sol    60.9375      b
##
## attr(,"class")
## [1] "group"

require(dplyr)
require(ggplot2)
erro = summarise(group_by(dados, semead),
                 avg = mean(altura), sd = sd(altura))

ggplot(erro, aes(semead, avg, fill=semead))+
  geom_bar(stat="identity")+
  geom_errorbar(aes(ymin=avg-sd, ymax =avg+sd), width=0.1, col="black") +
  xlab("Tratamentos") +
  ylab("Altura média de planta de mamoeiro aos 147 DAS (cm)") +
  theme_bw() +
  theme(legend.position="top") +
  annotate("text", label=tu$groups$groups[1], x=1, y=20, size = 5) +
  annotate("text", label=tu$groups$groups[2], x=2, y=20, size = 5) +
  annotate("text", label=tu$groups$groups[3], x=3, y=20, size = 5) +
  theme(legend.position="none") +
  labs(caption = "Médias seguidas de mesma letra indicam diferença nula à 5%")

```

```
## Warning: Ignoring unknown parameters: with
```



Teste de Scott-Knott

```
library(ScottKnott)
sk <- SK(x=dados, y=dados$altura, model="altura~bloco+semead", which="semead")
summary(sk)

##          Levels      Means SK(5%)
##  Direta no campo 101.0500     a
##  Recip. ripado   61.6625     b
##  Recip. ao sol    60.9375     b

print(sk)

## $av
## Call:
##   aov(formula = altura ~ bloco + semead, data = dat)
## 
## Terms:
##   bloco      semead Residuals
## Sum of Squares 2648.193 8429.103 2201.864
## Deg. of Freedom      3         2        18
```

```

## 
## Residual standard error: 11.0601
## Estimated effects may be unbalanced
##
## $groups
## [1] 1 2 2
##
## $nms
## [1] "Direta no campo" "Recip. ao sol"    "Recip. ripado"
##
## $ord
## [1] 1 3 2
##
## $m.inf
##                   mean   min   max
## Direta no campo 101.0500 70.5 136.1
## Recip. ripado    61.6625 43.7  77.1
## Recip. ao sol     60.9375 36.3  79.8
##
## $sig.level
## [1] 0.05
##
## attr(,"class")
## [1] "SK"   "list"

```

8.2.5 Referência

MELO, M. P.; PETERNELI, L. A. Conhecendo o R: Um visão mais que estatística. Viçosa, MG: UFV, 2013. 222p. Cap. 1.

BANZATTO, D. A; KRONKA, S. N. Experimentação agrícola. Jaboticabal, SP: FUNEP, 2006, 237p.

ZEVIANI, W. M. Estatística Básica e Experimentação no R. 45p.

Site: <http://www.leg.ufpr.br/~paulojus/>

8.3 Quadrado Latino

Esse material foi baseado nas aulas do Prof. Walmes Zeviani.

8.3.1 Banco de dados

Iremos instalar o pacote com banco de dados.

O labestData é um pacote para o software R de computação estatística que possui centenas de conjuntos de dados para o ensino e aprendizado de estatística. O pacote é desenvolvido pelo PET Estatística UFPR e conta com a participação de professores e colaboradores. O nome labest vem de LABoratório de ESTatística, que é o ambiente onde acontecem as aulas práticas do Curso de Estatística na UFPR.

```
library(devtools)
```

Instalação do repositório de desenvolvimento no GitLab:

```
install_git(url = "https://gitlab.c3sl.ufpr.br/pet-estatistica/labestData.git",
            branch = "master", build_vignettes = TRUE)
```

Do repositório de divulgação no GitHub:

```
install_github(repo = "labestData",
              username = "pet-estatistica",
              ref = "master", build_vignettes = TRUE)
```

Instalação por Arquivos Compactados:

```
install.packages("http://leg.ufpr.br/~walmes/pacotes/labestData_0.1.17.458.zip",
                 repos = NULL)
```

Lendo arquivos de dados:

Escolhendo os dados:

```
labestDataView()
```

Escolher o banco de dados

```
PimentelEg6.2
```

```
##      linha coluna varied prod
## 1        1      1      D  432
## 2        2      1      C  724
## 3        3      1      E  489
## 4        4      1      B  494
## 5        5      1      A  515
## 6        1      2      A  518
## 7        2      2      E  478
```

```

## 8    3    2    B  384
## 9    4    2    D  500
## 10   5    2    C  660
## 11   1    3    B  458
## 12   2    3    A  524
## 13   3    3    C  556
## 14   4    3    E  313
## 15   5    3    D  438
## 16   1    4    C  583
## 17   2    4    B  550
## 18   3    4    D  297
## 19   4    4    A  486
## 20   5    4    E  394
## 21   1    5    E  331
## 22   2    5    D  400
## 23   3    5    A  420
## 24   4    5    C  501
## 25   5    5    B  318

```

```
help("PimentelEg6.2")
```

Experimento de competição de variedades de cana-de-açúcar no qual foram usadas cinco variedades dispostas em um delineamento quadrado latino 5 times 5. Converte para nome de objeto mais simples:

```
pim <- PimentelEg6.2
help(pim)
```

```
## No documentation for 'pim' in specified packages and libraries:
## you could try '??pim'
```

Verificar estrutura dos dados

```
str(pim)
```

```
## 'data.frame': 25 obs. of 4 variables:
## $ linha : Factor w/ 5 levels "1","2","3","4",...: 1 2 3 4 5 1 2 3 4 5 ...
## $ coluna: Factor w/ 5 levels "1","2","3","4",...: 1 1 1 1 2 2 2 2 2 ...
## $ varied: Factor w/ 5 levels "A","B","C","D",...: 4 3 5 2 1 1 5 2 4 3 ...
## $ prod  : int 432 724 489 494 515 518 478 384 500 660 ...
```

Vendo o quadrado no plano:

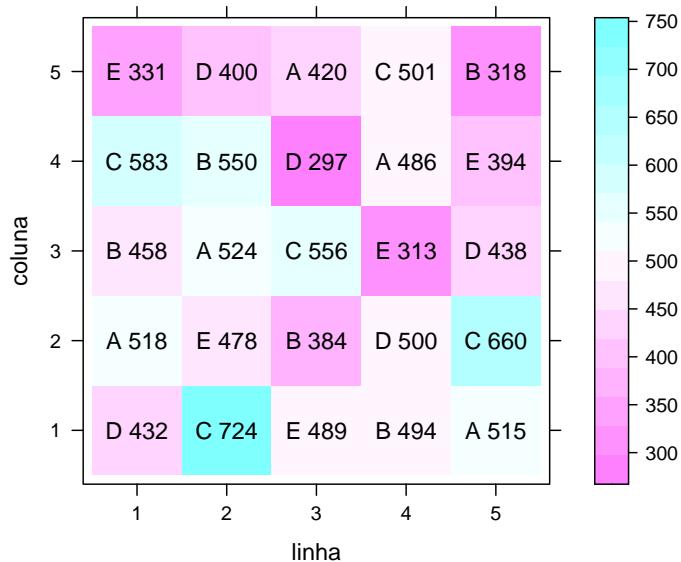
```
require(reshape)
```

Pacote gráfico:

```
require(latticeExtra)
```

Gerar gráfico:

```
levelplot(prod~linha+coluna, data=pim, aspect="iso") +
  layer(with(pim, panel.text(x=linha, y=coluna, label=paste(varied, prod))))
```

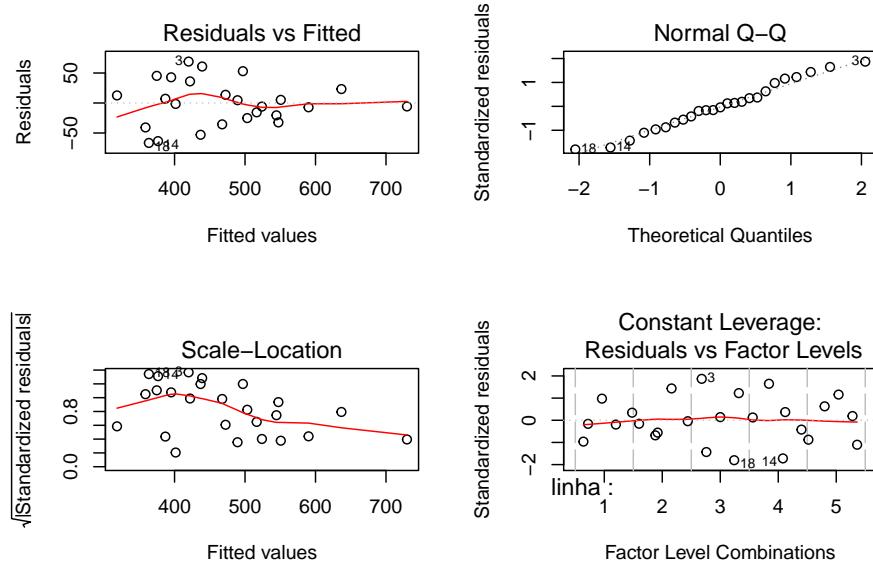


ANOVA:

```
m0 = lm (prod ~ linha + coluna + varied, data = pim)
```

Pressuposto seja atendidos?

```
par(mfrow = c(2,2))
plot(m0)
```



Teste de Bartlett para homocedasticidade:

```
bartlett.test(m0$residuals ~ pim$varied)
```

```
##
##  Bartlett test of homogeneity of variances
##
##  data:  m0$residuals by pim$varied
##  Bartlett's K-squared = 5.6629, df = 4, p-value = 0.2258
```

Teste de Shapiro-Wilk para Normalidade:

```
shapiro.test(m0$res)
```

```
##
##  Shapiro-Wilk normality test
##
##  data:  m0$res
##  W = 0.97701, p-value = 0.8202
```

Quadro de análise de variância:

```
anova(m0)
```

```
## Analysis of Variance Table
##
## Response: prod
##             Df Sum Sq Mean Sq F value    Pr(>F)
## linha        4 30481   7620  2.6804 0.0831343 .
## coluna       4 55641   13910  4.8930 0.0142293 *
## varied       4 137488   34372 12.0905 0.0003585 ***
## Residuals   12 34115    2843
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(m0)
```

```
##
## Call:
## lm(formula = prod ~ linha + coluna + varied, data = pim)
##
## Residuals:
##    Min     1Q Median     3Q    Max 
## -66.56 -25.16 -1.56  23.24  69.04 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 546.76     38.45 14.220 7.14e-09 ***
## linha2      70.80     33.72  2.100 0.05759 .  
## linha3     -35.20     33.72 -1.044 0.31713    
## linha4      -5.60     33.72 -0.166 0.87087    
## linha5       0.60     33.72  0.018 0.98610    
## coluna2     -22.80     33.72 -0.676 0.51178    
## coluna3     -73.00     33.72 -2.165 0.05127 .  
## coluna4     -68.80     33.72 -2.040 0.06397 .  
## coluna5    -136.80     33.72 -4.057 0.00159 ** 
## variedB     -51.80     33.72 -1.536 0.15045    
## variedC     112.20     33.72  3.327 0.00603 ** 
## variedD     -79.20     33.72 -2.349 0.03680 *  
## variedE     -91.60     33.72 -2.716 0.01873 * 
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 53.32 on 12 degrees of freedom
## Multiple R-squared:  0.8676, Adjusted R-squared:  0.7353 
## F-statistic: 6.555 on 12 and 12 DF,  p-value: 0.001366
```

Teste de média de comparações múltiplas:

```
library(doBy) # obter as médias ajustadas
library(multcomp)
p0 = LSmeans(m0, effect = "varied", level = 0.95)
```

Verificar estrutura:

```
str(p0)
```

```
## List of 3
## $ coef:'data.frame': 5 obs. of 5 variables:
##   ..$ estimate : num [1:5] 493 441 605 413 401
##   ..$ std.error: num [1:5] 23.8 23.8 23.8 23.8 23.8
##   ..$ statistic: num [1:5] 20.7 18.5 25.4 17.3 16.8
##   ..$ df      : num [1:5] 12 12 12 12 12
##   ..$ p.value  : num [1:5] 9.55e-11 3.49e-10 8.57e-12 7.34e-10 1.04e-09
## $ grid:'data.frame': 5 obs. of 1 variable:
##   ..$ varied: chr [1:5] "A" "B" "C" "D" ...
## $ L    : 'linest_matrix_class' num [1:5, 1:13] 1 1 1 1 1 0.2 0.2 0.2 0.2 0.2 ...
##   ..- attr(*, "dimnames")=List of 2
##   ... .$. : NULL
##   ... .$. : chr [1:13] "(Intercept)" "linha2" "linha3" "linha4" ...
##   ..- attr(*, "grid")='data.frame': 5 obs. of 1 variable:
##   ... .$. varied: chr [1:5] "A" "B" "C" "D" ...
##   - attr(*, "class")= chr "linest_class"
```

```
p0
```

```
## Coefficients:
##   estimate std.error statistic      df p.value
## [1,] 492.600   23.845   20.659  12.000      0
## [2,] 440.800   23.845   18.486  12.000      0
## [3,] 604.800   23.845   25.364  12.000      0
## [4,] 413.400   23.845   17.337  12.000      0
## [5,] 401.000   23.845   16.817  12.000      0
```

```
library(devtools)
install_github("walmes/wzRfun", ref = "master")
```

```
## Using github PAT from envvar GITHUB_PAT
```

```
## Skipping install of 'wzRfun' from a github remote, the SHA1 (a90b379a) has not changed
##   Use `force = TRUE` to force installation
```

```
require(wzRfun)
```

Comparações múltiplas, contrastes de Tukey. Método de correção de p-valor: single-step

```
tu <- summary(glht(m0, linfct=mcp(varied="Tukey")))
tu

## 
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: lm(formula = prod ~ linha + coluna + varied, data = pim)
##
## Linear Hypotheses:
##             Estimate Std. Error t value Pr(>|t|)    
## B - A == 0    -51.80     33.72  -1.536  0.56055  
## C - A == 0    112.20     33.72   3.327  0.03940 *  
## D - A == 0    -79.20     33.72  -2.349  0.19547  
## E - A == 0    -91.60     33.72  -2.716  0.10958  
## C - B == 0   164.00     33.72   4.863  0.00293 ** 
## D - B == 0    -27.40     33.72  -0.813  0.92176  
## E - B == 0    -39.80     33.72  -1.180  0.76211  
## D - C == 0   -191.40     33.72  -5.676 < 0.001 *** 
## E - C == 0   -203.80     33.72  -6.044 < 0.001 *** 
## E - D == 0    -12.40     33.72  -0.368  0.99555  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Resumo compacto com letras:

```
cld(tu)
```

```
##   A   B   C   D   E
## "a" "a" "b" "a" "a"

#cld(tu, decreasing=T)
```

Médias com intervalo de confiança:

```

ci <- confint(glht(m0, linfct= p0$L))
str(ci)

## List of 9
## $ model      :List of 13
##   ..$ coefficients : Named num [1:13] 546.8 70.8 -35.2 -5.6 0.6 ...
##   ... .- attr(*, "names")= chr [1:13] "(Intercept)" "linha2" "linha3" "linha4" ...
##   ..$ residuals    : Named num [1:25] -35.56 -5.76 69.04 4.64 -32.36 ...
##   ... .- attr(*, "names")= chr [1:25] "1" "2" "3" "4" ...
##   ..$ effects      : Named num [1:25] -2352.6 -161.7 64.937 10.772 -0.949 ...
##   ... .- attr(*, "names")= chr [1:25] "(Intercept)" "linha2" "linha3" "linha4" ...
##   ..$ rank         : int 13
##   ..$ fitted.values: Named num [1:25] 468 730 420 489 547 ...
##   ... .- attr(*, "names")= chr [1:25] "1" "2" "3" "4" ...
##   ..$ assign        : int [1:13] 0 1 1 1 1 2 2 2 2 3 ...
##   ..$ qr           :List of 5
##     ... $ qr     : num [1:25, 1:13] -5 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 ...
##     ... .- attr(*, "dimnames")=List of 2
##       ... ... $ : chr [1:25] "1" "2" "3" "4" ...
##       ... ... $ : chr [1:13] "(Intercept)" "linha2" "linha3" "linha4" ...
##     ... .- attr(*, "assign")= int [1:13] 0 1 1 1 1 2 2 2 2 3 ...
##     ... .- attr(*, "contrasts")=List of 3
##       ... ... $ linha : chr "contr.treatment"
##       ... ... $ coluna: chr "contr.treatment"
##       ... ... $ varied: chr "contr.treatment"
##     ... $ qraux: num [1:13] 1.2 1.42 1.41 1.4 1.38 ...
##     ... $ pivot: int [1:13] 1 2 3 4 5 6 7 8 9 10 ...
##     ... $ tol : num 1e-07
##     ... $ rank : int 13
##     ... - attr(*, "class")= chr "qr"
##     ..$ df.residual : int 12
##     ..$ contrasts   :List of 3
##       ... $ linha : chr "contr.treatment"
##       ... $ coluna: chr "contr.treatment"
##       ... $ varied: chr "contr.treatment"
##     ..$ xlevels     :List of 3
##       ... $ linha : chr [1:5] "1" "2" "3" "4" ...
##       ... $ coluna: chr [1:5] "1" "2" "3" "4" ...
##       ... $ varied: chr [1:5] "A" "B" "C" "D" ...
##     ..$ call         : language lm(formula = prod ~ linha + coluna + varied, data = p...
##     ..$ terms        :Classes 'terms', 'formula' language prod ~ linha + coluna + var...
##     ... .- attr(*, "variables")= language list(prod, linha, coluna, varied)
##     ... .- attr(*, "factors")= int [1:4, 1:3] 0 1 0 0 0 1 0 0 ...
##     ... .- attr(*, "dimnames")=List of 2
##       ... ... $ : chr [1:4] "prod" "linha" "coluna" "varied"

```

```

## ... . . . . . $ : chr [1:3] "linha" "coluna" "varied"
## ... . . . - attr(*, "term.labels")= chr [1:3] "linha" "coluna" "varied"
## ... . . . - attr(*, "order")= int [1:3] 1 1 1
## ... . . . - attr(*, "intercept")= int 1
## ... . . . - attr(*, "response")= int 1
## ... . . . - attr(*, ".Environment")=<environment: R_GlobalEnv>
## ... . . . - attr(*, "predvars")= language list(prod, linha, coluna, varied)
## ... . . . - attr(*, "dataClasses")= Named chr [1:4] "numeric" "factor" "factor" "factor"
## ... . . . - attr(*, "names")= chr [1:4] "prod" "linha" "coluna" "varied"
## ..$ model      : 'data.frame':   25 obs. of  4 variables:
## ...$ prod    : int [1:25] 432 724 489 494 515 518 478 384 500 660 ...
## ...$ linha   : Factor w/ 5 levels "1","2","3","4",...: 1 2 3 4 5 1 2 3 4 5 ...
## ...$ coluna  : Factor w/ 5 levels "1","2","3","4",...: 1 1 1 1 1 2 2 2 2 ...
## ...$ varied  : Factor w/ 5 levels "A","B","C","D",...: 4 3 5 2 1 1 5 2 4 3 ...
## ...- attr(*, "terms")=Classes 'terms', 'formula' language prod ~ linha + coluna + varied
## ... . . . - attr(*, "variables")= language list(prod, linha, coluna, varied)
## ... . . . - attr(*, "factors")= int [1:4, 1:3] 0 1 0 0 0 0 1 0 0 0 ...
## ... . . . . - attr(*, "dimnames")=List of 2
## ... . . . . . $ : chr [1:4] "prod" "linha" "coluna" "varied"
## ... . . . . . $ : chr [1:3] "linha" "coluna" "varied"
## ... . . . . - attr(*, "term.labels")= chr [1:3] "linha" "coluna" "varied"
## ... . . . . - attr(*, "order")= int [1:3] 1 1 1
## ... . . . . - attr(*, "intercept")= int 1
## ... . . . . - attr(*, "response")= int 1
## ... . . . . - attr(*, ".Environment")=<environment: R_GlobalEnv>
## ... . . . . - attr(*, "predvars")= language list(prod, linha, coluna, varied)
## ... . . . . - attr(*, "dataClasses")= Named chr [1:4] "numeric" "factor" "factor" "factor"
## ... . . . . - attr(*, "names")= chr [1:4] "prod" "linha" "coluna" "varied"
## ..- attr(*, "class")= chr "lm"
## $ linfct     : 'linest_matrix_class' num [1:5, 1:13] 1 1 1 1 1 0.2 0.2 0.2 0.2 0.2 ...
## ..- attr(*, "dimnames")=List of 2
## ... $ : chr [1:5] "1" "2" "3" "4" ...
## ... $ : chr [1:13] "(Intercept)" "linha2" "linha3" "linha4" ...
## ..- attr(*, "grid")='data.frame':  5 obs. of  1 variable:
## ... $ varied: chr [1:5] "A" "B" "C" "D" ...
## $ rhs        : num [1:5] 0 0 0 0 0
## $ coef       : Named num [1:13] 546.8 70.8 -35.2 -5.6 0.6 ...
## ..- attr(*, "names")= chr [1:13] "(Intercept)" "linha2" "linha3" "linha4" ...
## $ vcov       : num [1:13, 1:13] 1478 -569 -569 -569 -569 ...
## ..- attr(*, "dimnames")=List of 2
## ... $ : chr [1:13] "(Intercept)" "linha2" "linha3" "linha4" ...
## ... $ : chr [1:13] "(Intercept)" "linha2" "linha3" "linha4" ...
## $ df         : int 12
## $ alternative: chr "two.sided"
## $ type       : NULL
## $ confint    : num [1:5, 1:3] 493 441 605 413 401 ...

```

```

##  ..- attr(*, "dimnames")=List of 2
##    ...$ : chr [1:5] "1" "2" "3" "4" ...
##    ...$ : chr [1:3] "Estimate" "lwr" "upr"
##  ..- attr(*, "conf.level")= num 0.95
##  ..- attr(*, "calpha")= num 3.01
## - attr(*, "class")= chr [1:2] "confint.glht" "glht"
## - attr(*, "type")= chr "adjusted"

```

Juntar com os nomes dos tratamentos com os intervalos:

```
ci <- cbind(p0$grid, ci$confint)
```

Colocar as letras. onde estão as letras?

```
str(cld(tu))
```

```

## List of 10
## $ y      : int [1:25] 432 724 489 494 515 518 478 384 500 660 ...
## $ yname   : chr "prod"
## $ x      : Factor w/ 5 levels "A","B","C","D",...: 4 3 5 2 1 1 5 2 4 3 ...
## $ xname   : chr "varied"
## $ weights : NULL
## $ lp      : Named num [1:25] 468 730 420 489 547 ...
## ..- attr(*, "names")= chr [1:25] "1" "2" "3" "4" ...
## $ covar   : logi TRUE
## $ comps   : chr [1:10, 1:2] "B" "C" "D" "E" ...
## ..- attr(*, "dimnames")=List of 2
##   ...$ : chr [1:10] "B - A" "C - A" "D - A" "E - A" ...
##   ...$ : NULL
## $ signif  : logi [1:10] FALSE TRUE FALSE FALSE TRUE FALSE ...
## $ mcletters:List of 5
##   ...$ Letters      : Named chr [1:5] "a" "a" "b" "a" ...
##   ...- attr(*, "names")= chr [1:5] "A" "B" "C" "D" ...
##   ...$ monospacedLetters: Named chr [1:5] "a " "a " "b" "a " ...
##   ...- attr(*, "names")= chr [1:5] "A" "B" "C" "D" ...
##   ...$ LetterMatrix   : logi [1:5, 1:2] TRUE TRUE FALSE TRUE TRUE FALSE ...
##   ...- attr(*, "dimnames")=List of 2
##     ...$ : chr [1:5] "A" "B" "C" "D" ...
##     ...$ : chr [1:2] "a" "b"
##   ...$ aLetters       : chr [1:52] "a" "b" "c" "d" ...
##   ...$ aseparator     : chr "."
##   ..- attr(*, "class")= chr "multcompLetters"
## - attr(*, "class")= chr "cld"

```

```

ci$cld <- cld(tu)$mclatters$Letters
ci$cld

## [1] "a" "a" "b" "a" "a"

str(ci)

## 'data.frame':   5 obs. of  5 variables:
## $ varied : chr  "A" "B" "C" "D" ...
## $ Estimate: num  493 441 605 413 401
## $ lwr      : num  421 369 533 342 329
## $ upr      : num  564 512 676 485 473
## $ cld      : chr  "a" "a" "b" "a" ...

```

```

ci$varied <- factor(ci$varied)

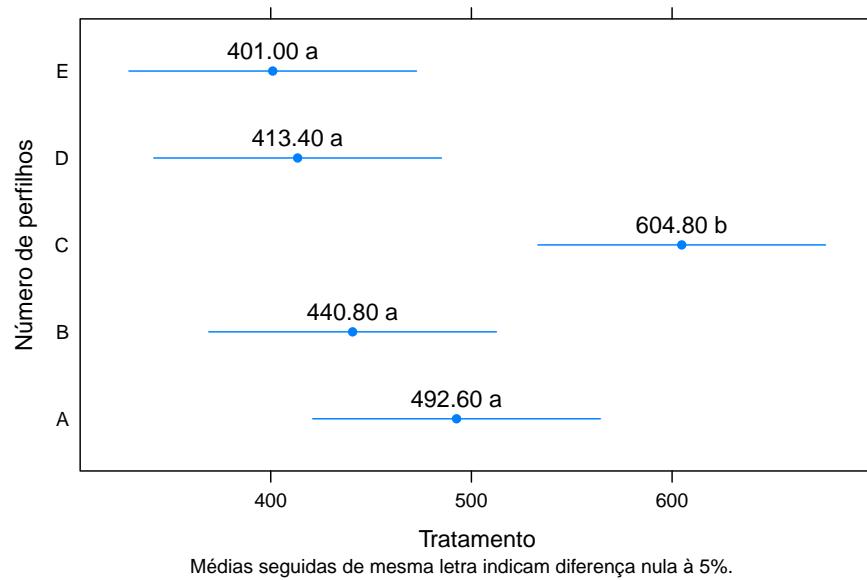
```

Representação gráfica dos resultados:

```

require(latticeExtra)
segplot(varied~lwr+upr,
        centers=Estimate,
        data=ci,
        draw=F, #desenhar segmentos
        xlab = expression("Tratamento"),
        ylab = "Número de perfilhos",
        sub = list("Médias seguidas de mesma letra indicam diferença nula à 5%.",
                  font=1, cex=0.8)) +
layer(panel.text( x = centers,
                  y = z,
                  labels = sprintf("%0.2f %s", centers, ci$cld),
                  pos = 3))

```



8.3.2 Exercícios

```
require(labestData)
```

```
ZimmermannTb5.15
```

```
##   linha coluna trat perf
## 1      1      1   1 155
## 2      2      1   5 198
## 3      3      1   4 187
## 4      4      1   2 204
## 5      5      1   6 209
## 6      6      1   3 216
## 7      1      2   5 191
## 8      2      2   3 199
## 9      3      2   2 213
## 10     4      2   6 225
## 11     5      2   4 218
## 12     6      2   1 181
## 13     1      3   6 203
## 14     2      3   4 205
## 15     3      3   3 195
```

```

## 16    4    3    1  135
## 17    5    3    5  196
## 18    6    3    2  178
## 19    1    4    3  187
## 20    2    4    1  186
## 21    3    4    6  177
## 22    4    4    4  190
## 23    5    4    2  209
## 24    6    4    5  215
## 25    1    5    2  162
## 26    2    5    6  181
## 27    3    5    5  188
## 28    4    5    3  204
## 29    5    5    1  175
## 30    6    5    4  203
## 31    1    6    4  167
## 32    2    6    2  174
## 33    3    6    1  161
## 34    4    6    5  181
## 35    5    6    3  210
## 36    6    6    6  206

```

```

help(ZimmermannTb5.15)
str(ZimmermannTb5.15)

```

```

## 'data.frame':   36 obs. of  4 variables:
## $ linha : Factor w/ 6 levels "1","2","3","4",...: 1 2 3 4 5 6 1 2 3 4 ...
## $ coluna: Factor w/ 6 levels "1","2","3","4",...: 1 1 1 1 1 1 2 2 2 2 ...
## $ trat  : Factor w/ 6 levels "1","2","3","4",...: 1 5 4 2 6 3 5 3 2 6 ...
## $ perf  : num  155 198 187 204 209 216 191 199 213 225 ...

```

Atribuir um nome menor ao conjunto de dados:

```

dados <- ZimmermannTb5.15

```

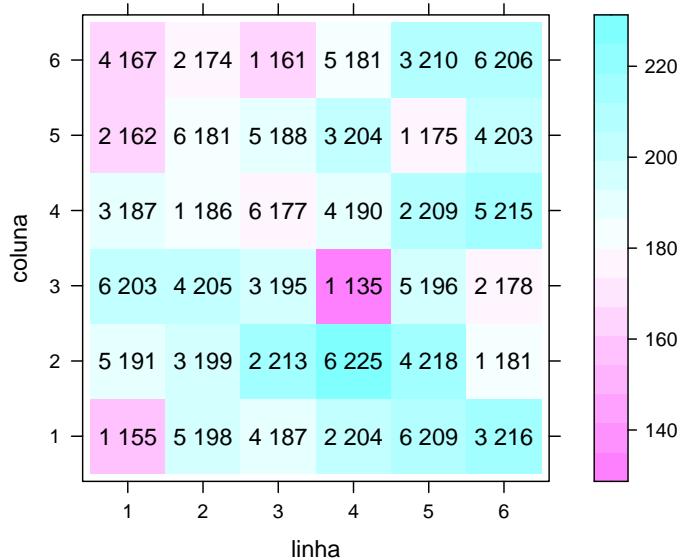
Gráfico do layout:

```

require(latticeExtra)

levelplot(perf~linha+coluna,
           data=dados, aspect="iso")+
  layer(with(dados,
             panel.text(x=linha, y=coluna,
                        label=paste(trat, perf))))

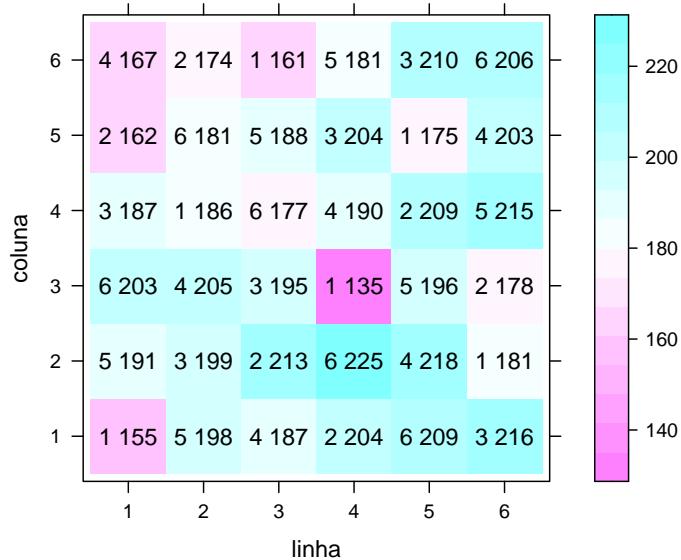
```



```
require(latticeExtra)
```

Gráfico do layout:

```
levelplot(perf~linha+coluna,
          data=dados, aspect="iso")+
  layer(with(dados,
             panel.text(x=linha, y=coluna,
                        label=paste(trat, perf))))
```

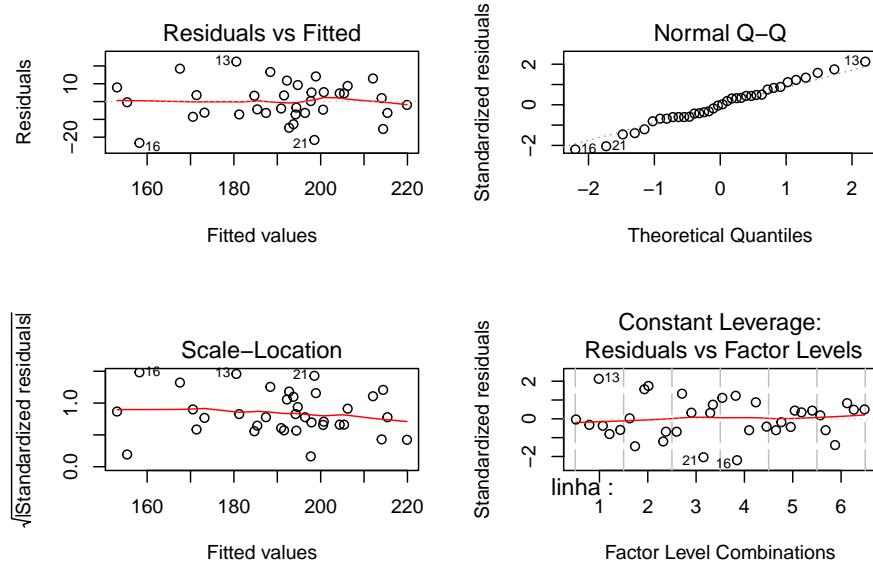


ANOVA:

```
m0 = lm (perf~linha + coluna + trat, data = dados)
```

Foram atendidos?

```
par(mfrow = c(2,2))
plot(m0)
```



Teste de Bartlett para homocedasticidade:

```
bartlett.test(m0$res, dados$trat)
```

```
##
##  Bartlett test of homogeneity of variances
##
##  data:  m0$res and dados$trat
##  Bartlett's K-squared = 6.5685, df = 5, p-value = 0.2548
```

Teste de Shapiro-Wilk para Normalidade:

```
shapiro.test(m0$res)
```

```
##
##  Shapiro-Wilk normality test
##
##  data:  m0$res
##  W = 0.98701, p-value = 0.9408
```

Quadro de análise de variância:

```
anova(m0)
```

```
## Analysis of Variance Table
##
## Response: perf
##             Df Sum Sq Mean Sq F value    Pr(>F)
## linha      5 2513.9  502.78  2.5039 0.06461 .
## coluna     5 1976.2  395.24  1.9684 0.12760
## trat       5 5298.2 1059.64  5.2773 0.00299 **
## Residuals 20 4015.9  200.79
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(m0)
```

```
##
## Call:
## lm(formula = perf ~ linha + coluna + trat, data = dados)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23.2222  -6.3889  -0.0556   5.9444  22.4444
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 155.3889    9.4468 16.449 4.34e-13 ***
## linha2      13.0000   8.1812  1.589 0.127741
## linha3      9.3333   8.1812  1.141 0.267417
## linha4     12.3333   8.1812  1.508 0.147310
## linha5     25.3333   8.1812  3.097 0.005689 **
## linha6     22.3333   8.1812  2.730 0.012905 *
## coluna2     9.6667   8.1812  1.182 0.251234
## coluna3    -9.5000   8.1812 -1.161 0.259231
## coluna4    -0.8333   8.1812 -0.102 0.919882
## coluna5    -9.3333   8.1812 -1.141 0.267417
## coluna6   -11.6667   8.1812 -1.426 0.169280
## trat2      24.5000   8.1812  2.995 0.007161 **
## trat3      36.3333   8.1812  4.441 0.000251 ***
## trat4      29.5000   8.1812  3.606 0.001764 **
## trat5      29.3333   8.1812  3.585 0.001850 **
## trat6      34.6667   8.1812  4.237 0.000404 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 14.17 on 20 degrees of freedom
## Multiple R-squared:  0.7091, Adjusted R-squared:  0.4909
## F-statistic:  3.25 on 15 and 20 DF,  p-value: 0.007591
```

Comparações múltiplas:

```
library(doBy)
library(multcomp)

p0 = LSmeans(m0, effect = "trat", level = 0.95)

str(p0)

## List of 3
## $ coef:'data.frame':   6 obs. of  5 variables:
##   ..$ estimate : num [1:6] 165 190 202 195 195 ...
##   ..$ std.error: num [1:6] 5.78 5.78 5.78 5.78 5.78 ...
##   ..$ statistic: num [1:6] 28.6 32.8 34.9 33.7 33.7 ...
##   ..$ df      : num [1:6] 20 20 20 20 20 20
##   ..$ p.value  : num [1:6] 1.06e-17 7.10e-19 2.16e-19 4.26e-19 4.33e-19 ...
## $ grid:'data.frame':   6 obs. of  1 variable:
##   ..$ trat: chr [1:6] "1" "2" "3" "4" ...
## $ L   : 'linest_matrix_class' num [1:6, 1:16] 1 1 1 1 1 ...
##   ..- attr(*, "dimnames")=List of 2
##   ... .$: NULL
##   ... .$: chr [1:16] "(Intercept)" "linha2" "linha3" "linha4" ...
##   ..- attr(*, "grid")='data.frame':   6 obs. of  1 variable:
##   ... .$: trat: chr [1:6] "1" "2" "3" "4" ...
##   - attr(*, "class")= chr "linest_class"
```

Criando a tabela com as estimativas:

```
library(devtools)
#install_github("walmes/wzRfun", ref = "master")
require(wzRfun)
```

Comparações múltiplas, contrastes de Tukey. Método de correção de p-valor: single-step.

```
tu <- summary(glht(m0, linfct=mcp(trat="Tukey")))
tu

##
```

```

##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: lm(formula = perf ~ linha + coluna + trat, data = dados)
##
## Linear Hypotheses:
##             Estimate Std. Error t value Pr(>|t|)
## 2 - 1 == 0  24.5000   8.1812  2.995  0.06733 .
## 3 - 1 == 0  36.3333   8.1812  4.441  0.00300 ** 
## 4 - 1 == 0  29.5000   8.1812  3.606  0.01889 *
## 5 - 1 == 0  29.3333   8.1812  3.585  0.01964 *
## 6 - 1 == 0  34.6667   8.1812  4.237  0.00481 ** 
## 3 - 2 == 0  11.8333   8.1812  1.446  0.69987
## 4 - 2 == 0   5.0000   8.1812  0.611  0.98893
## 5 - 2 == 0   4.8333   8.1812  0.591  0.99051
## 6 - 2 == 0  10.1667   8.1812  1.243  0.81105
## 4 - 3 == 0  -6.8333   8.1812 -0.835  0.95719
## 5 - 3 == 0  -7.0000   8.1812 -0.856  0.95272
## 6 - 3 == 0  -1.6667   8.1812 -0.204  0.99994
## 5 - 4 == 0  -0.1667   8.1812 -0.020  1.00000
## 6 - 4 == 0   5.1667   8.1812  0.632  0.98717
## 6 - 5 == 0   5.3333   8.1812  0.652  0.98522
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)

```

Resumo compacto com letras:

```
cld(tu)
```

```

##    1    2    3    4    5    6
##  "a" "ab" "b"  "b"  "b"  "b"

```

```
#cld(tu, decreasing=T)
```

Médias com intervalo de confiança:

```

ci <- confint(glht(m0, linfct= p0$L))
ci

```

```

## 
## Simultaneous Confidence Intervals

```

```

## 
## Fit: lm(formula = perf ~ linha + coluna + trat, data = dados)
## 
## Quantile = 2.8973
## 95% family-wise confidence level
## 
## 
## Linear Hypotheses:
##          Estimate lwr      upr
## 1 == 0 165.5000 148.7391 182.2609
## 2 == 0 190.0000 173.2391 206.7609
## 3 == 0 201.8333 185.0724 218.5942
## 4 == 0 195.0000 178.2391 211.7609
## 5 == 0 194.8333 178.0724 211.5942
## 6 == 0 200.1667 183.4058 216.9276

```

Juntar com os nomes dos tratamentos com os intervalos:

```

ci <- cbind(p0$grid, ci$confint)

ci$cld <- cld(tu)$mcletters$Letters

str(cld(tu))

```

```

## List of 10
## $ y      : num [1:36] 155 198 187 204 209 ...
## $ yname   : chr "perf"
## $ x      : Factor w/ 6 levels "1","2","3","4",...
## $ xname   : chr "trat"
## $ weights : NULL
## $ lp      : Named num [1:36] 155 198 194 192 215 ...
## ..- attr(*, "names")= chr [1:36] "1" "2" "3" "4" ...
## $ covar   : logi TRUE
## $ comps   : chr [1:15, 1:2] "2" "3" "4" "5" ...
## ..- attr(*, "dimnames")=List of 2
## ...$ : chr [1:15] "2 - 1" "3 - 1" "4 - 1" "5 - 1" ...
## ...$ : NULL
## $ signif  : logi [1:15] FALSE TRUE TRUE TRUE FALSE ...
## $ mcletters:List of 5
##   ..$ Letters      : Named chr [1:6] "a" "ab" "b" "b" ...
##   ...- attr(*, "names")= chr [1:6] "1" "2" "3" "4" ...
##   ..$ monospacedLetters: Named chr [1:6] "a" "ab" "b" "b" ...
##   ...- attr(*, "names")= chr [1:6] "1" "2" "3" "4" ...
##   ..$ LetterMatrix : logi [1:6, 1:2] TRUE TRUE FALSE FALSE FALSE ...
##   ...- attr(*, "dimnames")=List of 2

```

```

## ... . . . $ : chr [1:6] "1" "2" "3" "4" ...
## ... . . . $ : chr [1:2] "a" "b"
## ..$ aLetters : chr [1:52] "a" "b" "c" "d" ...
## ..$ aseparator : chr "."
## ..- attr(*, "class")= chr "multcompLetters"
## - attr(*, "class")= chr "cld"

```

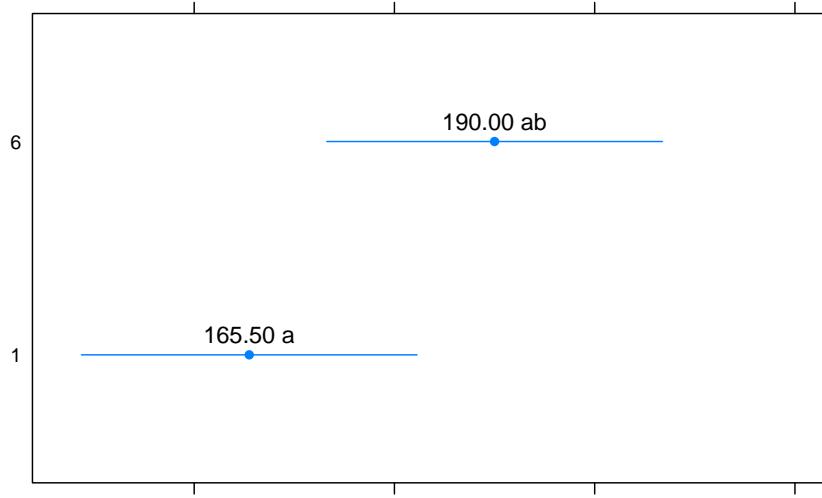
Representação gráfica dos resultados:

```

require(latticeExtra)

segplot(trat~lwr+upr,
        centers=Estimate,
        data=ci,
        draw=F,
        xlab = expression(" "),
        ylab = " ",
        sub = list("Médias seguidas de mesma letra indicam diferença nula à 5%.",
                  font=1, cex=0.8)) +
        layer(panel.text( x = centers,
                          y = z,
                          labels = sprintf("%0.2f %s", centers, ci$cld),
                          pos = 3))

```



Médias seguidas de mesma letra indicam diferença nula à 5%.

8.3.3 Exercício para entregar

DiasEg9.4 help("DiasEg9.4")

BarbinPg104 help("BarbinPg104")

ZimmermannTb12.27 help(ZimmermannTb12.27)

ZimmermannTb12.26 help(ZimmermannTb12.26)

StorckEg2.3.5 help(StorckEg2.3.5)

ZimmermannTb16.10 help(ZimmermannTb16.10)

8.4 Regressão Linear simples

Iremos utilizar o pacote tidyverse:

```
library(tidyverse)
```

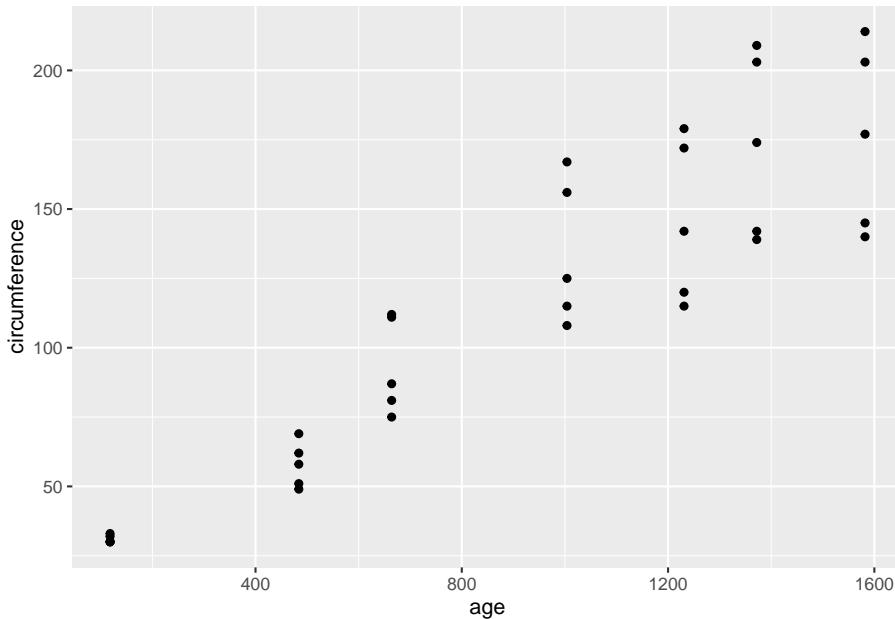
Regressão linear é uma equação para estimar a condicional (valor esperado) de uma variável y , dados os valores de algumas outras variáveis x .

$$y = a + bx$$

Vamos utilizar o exemplo do banco de dados do R:

```
View(Orange)
```

```
qplot(x=age,y=circumference,data=Orange)
```



```
cor.test(Orange$age, Orange$circumference)
```

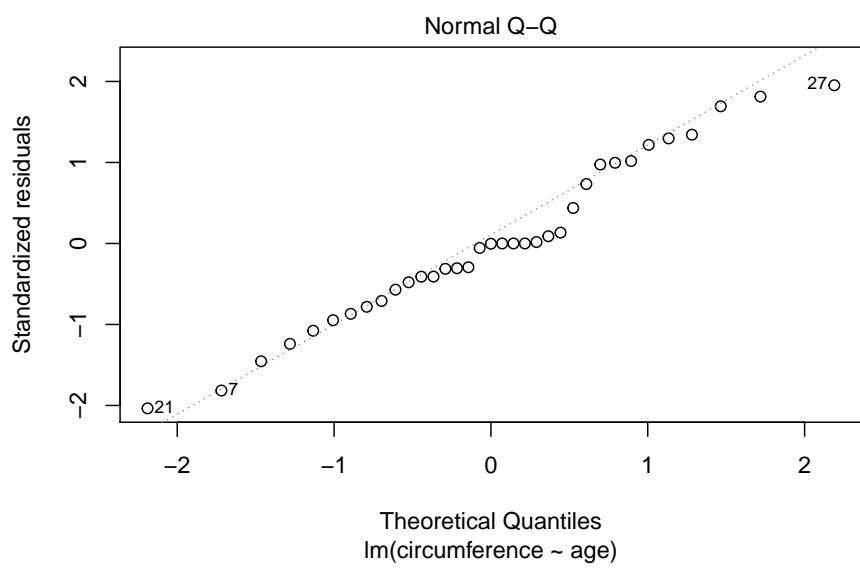
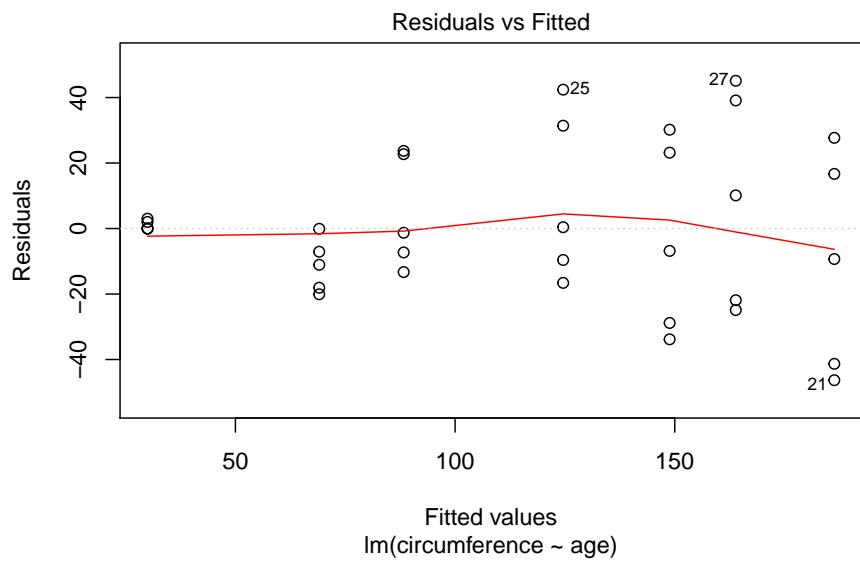
```
## 
## Pearson's product-moment correlation
## 
## data: Orange$age and Orange$circumference
## t = 12.9, df = 33, p-value = 1.931e-14
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8342364 0.9557955
## sample estimates:
##      cor
## 0.9135189
```

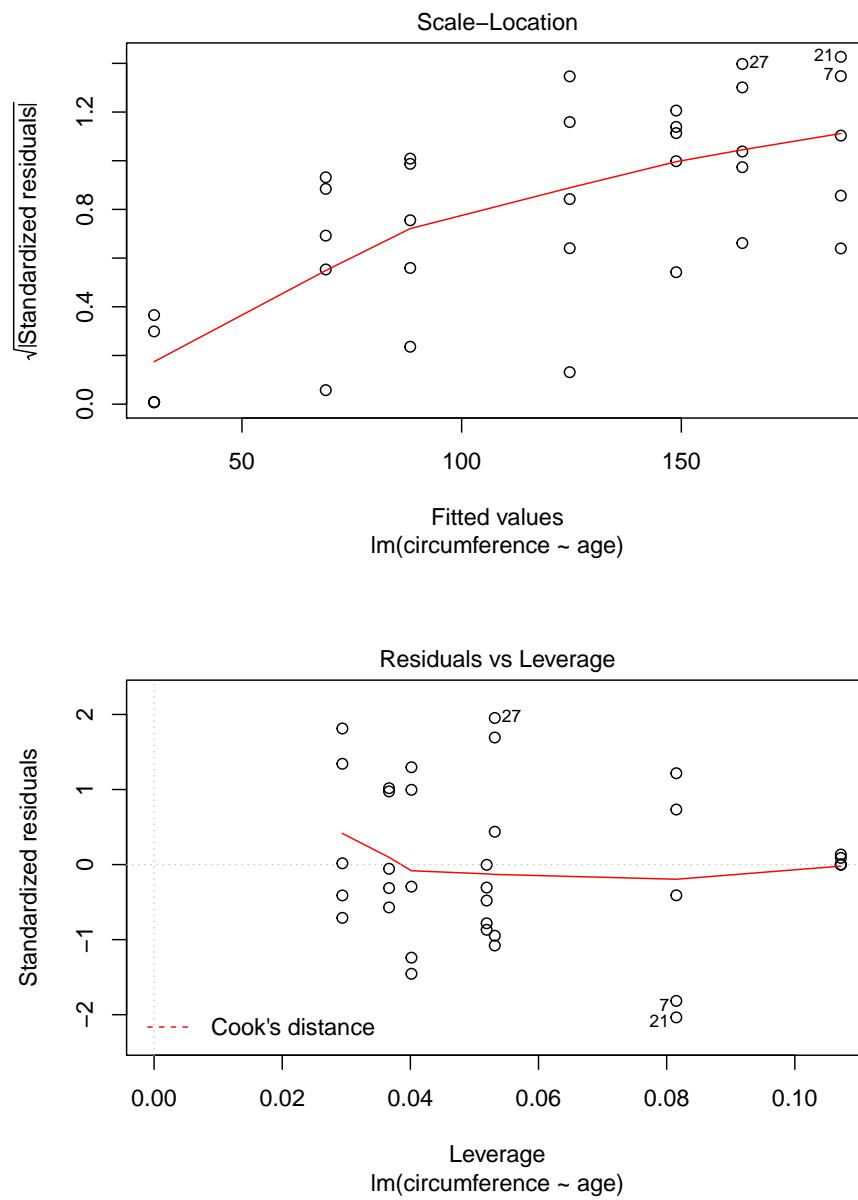
```
rl<- lm(circumference ~ age , data = Orange)
```

Zerando o intercepto

```
r10<- lm(circumference ~ 0+ age , data = Orange)
```

```
plot(rl) #plots diagnósticos
```





```
shapiro.test (residuals(r1))
```

```
##  
## Shapiro-Wilk normality test
```

```

## 
## data: residuals(rl)
## W = 0.97289, p-value = 0.5273

bartlett.test(rl$res, Orange$Tree, Orange$age)

## 
##  Bartlett test of homogeneity of variances
##
## data: rl$res and Orange$Tree
## Bartlett's K-squared = 2.7663, df = 4, p-value = 0.5977

```

Resultado da Regressão linear:

```

summary(rl)

## 
## Call:
## lm(formula = circumference ~ age, data = Orange)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max
## -46.310  -14.946   -0.076  19.697  45.111
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 17.399650  8.622660  2.018   0.0518 .
## age         0.106770  0.008277 12.900 1.93e-14 ***
## ---      
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 23.74 on 33 degrees of freedom
## Multiple R-squared:  0.8345, Adjusted R-squared:  0.8295 
## F-statistic: 166.4 on 1 and 33 DF,  p-value: 1.931e-14

```

Rejeição da hipótese nula (variáveis não são relacionadas) *valor p <0.005* R-square explica o quanto o modelo explica da variação

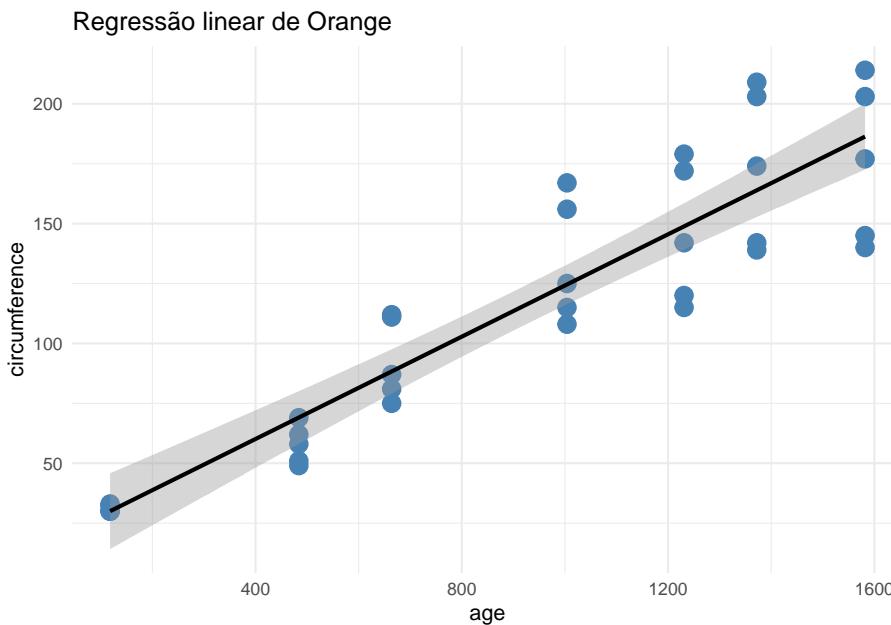
Gráfico:

```

ggplot(Orange,aes(x=age,y=circumference))+ geom_point(colour="steelblue",size=4)+ 
  geom_smooth(method="lm",colour="black")+labs(title="Regressão linear de Orange")+
  theme_minimal()

```

```
## `geom_smooth()` using formula 'y ~ x'
```

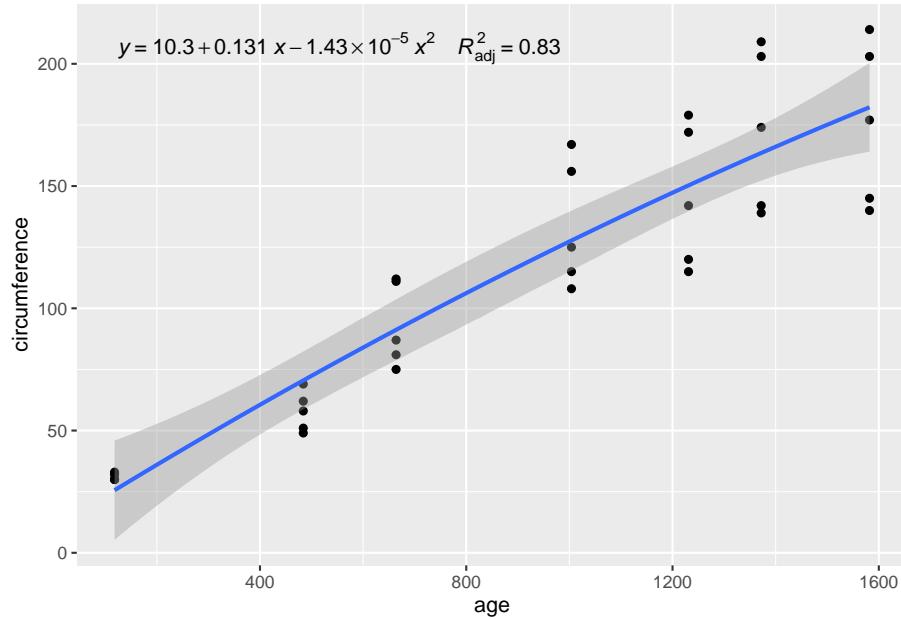


Gerar gráficos com o pacote ggpmisc:

```
library(ggpmisc)
```

Gráfico:

```
formula <- y ~ poly(x, 2, raw = TRUE)
ggplot(Orange, aes(x=age, y=circumference)) +
  geom_point() +
  geom_smooth(method = "lm", formula = formula) +
  stat_poly_eq(aes(label = paste(..eq.label.., ..adj.rr.label.., sep = "~~~~")),
               formula = formula, parse = TRUE)
```



Outras funções interessantes:

```
coefficients(rl) # coeficientes do modelo
```

```
## (Intercept)      age
## 17.3996502   0.1067703
```

```
confint(rl, level=0.95) # intervalo de confiança
```

```
##                  2.5 %    97.5 %
## (Intercept) -0.14328303 34.9425835
## age          0.08993141  0.1236092
```

```
fitted(rl) # valores previstos
```

```
##      1      2      3      4      5      6      7      8
## 29.99855 69.07649 88.29515 124.59706 148.83392 163.88854 186.31030 29.99855
##      9     10     11     12     13     14     15     16
## 69.07649 88.29515 124.59706 148.83392 163.88854 186.31030 29.99855 69.07649
##     17     18     19     20     21     22     23     24
## 88.29515 124.59706 148.83392 163.88854 186.31030 29.99855 69.07649 88.29515
##     25     26     27     28     29     30     31     32
## 124.59706 148.83392 163.88854 186.31030 29.99855 69.07649 88.29515 124.59706
##     33     34     35
## 148.83392 163.88854 186.31030
```

```
residuals(r1) # residuais
```

```
##          1          2          3          4          5
## 0.001451402 -11.076487573 -1.295146086 -9.597056609 -28.833920400
##          6          7          8          9         10
## -21.888536235 -41.310304499  3.001451402 -0.076487573 22.704853914
##          11         12         13         14         15
## 31.402943391 23.166079600 39.111463765 16.689695501  0.001451402
##          16         17         18         19         20
## -18.076487573 -13.295146086 -16.597056609 -33.833920400 -24.888536235
##          21         22         23         24         25
## -46.310304499  2.001451402 -7.076487573 23.704853914 42.402943391
##          26         27         28         29         30
## 30.166079600 45.111463765 27.689695501  0.001451402 -20.076487573
##          31         32         33         34         35
## -7.295146086  0.402943391 -6.833920400 10.111463765 -9.310304499
```

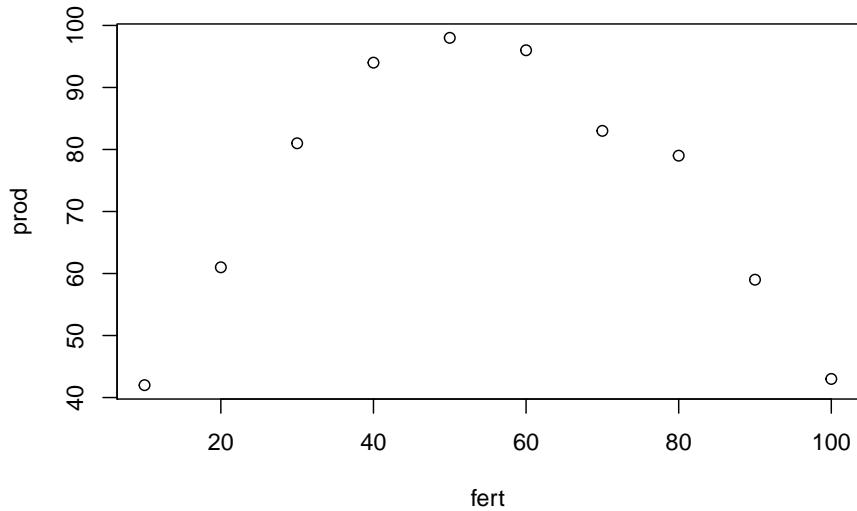
```
anova(r1) # tabela ANOVA
```

```
## Analysis of Variance Table
##
## Response: circumference
##           Df Sum Sq Mean Sq F value    Pr(>F)
## age        1  93772   93772  166.42 1.931e-14 ***
## Residuals 33  18595    563
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Qualquer modelo de regressão polinomial pode ser obtido com um comando simples: o `lm()` que vem do inglês “linear models”.

Veja o exemplo abaixo:

```
fert<-c(10,20,30,40,50,60,70,80,90,100)
prod<-c(42,61,81,94,98,96,83,79,59,43)
plot(fert,prod)
```



```
dados = data.frame(fert,prod)
```

Observe a necessidade do argumento `I()` para interações como x^2 :

```
reg <- lm ( #ajusta uma Regressão
            prod ~ fert + I (fert^2)) #modelo de Regressão quadrática
reg
```

```
##
## Call:
## lm(formula = prod ~ fert + I(fert^2))
##
## Coefficients:
## (Intercept)      fert      I(fert^2)
##     15.51667    2.95720   -0.02716
```

Para “desenhar” a curva ajustada:

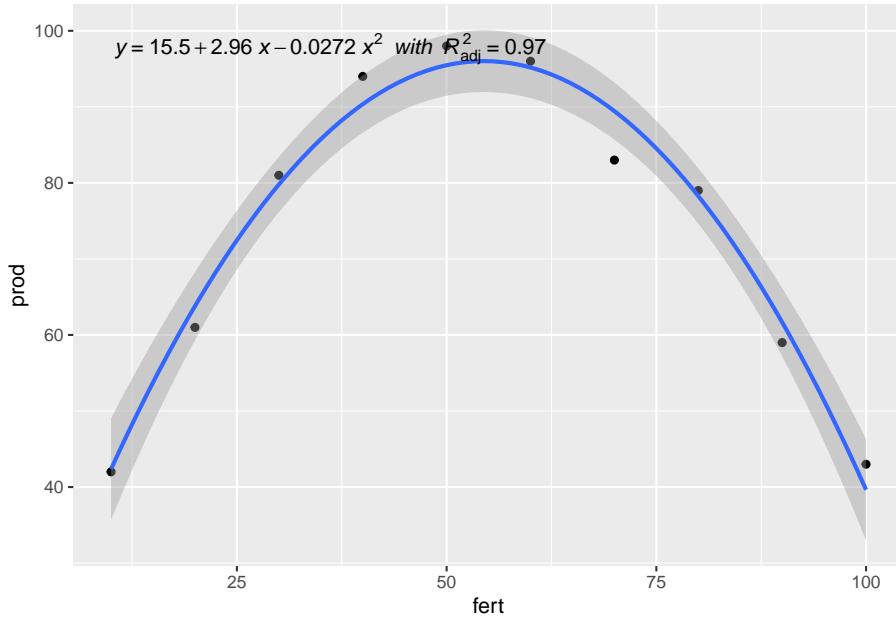
```
#curve(15.51667+2.95720*x-0.02716*x*x,0,100,add=T,col=2)
```

Várias outras análises podem ser feitas como anteriormente na regressão linear. Veja uma delas:

```
anova(reg)
```

```
## Analysis of Variance Table
##
## Response: prod
##             Df  Sum Sq Mean Sq  F value    Pr(>F)
## fert          1     7.6     7.6   0.5878   0.4683
## I(fert^2)    1 3894.6 3894.6 302.2072 5.126e-07 ***
## Residuals    7   90.2    12.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
formula <- y ~ poly(x, 2, raw = TRUE)
ggplot(dados, aes(x=fert, y=prod)) +
  geom_point() +
  geom_smooth(method = "lm", formula = formula) +
  stat_poly_eq(aes(label = paste(..eq.label.., ..adj.rr.label..,
    sep = "~~italic(\"with\")~~")),
    formula = formula, parse = TRUE)
```



8.5 Regressão Linear Multipla

Os modelos múltiplos são aqueles em que duas ou mais variáveis independentes influenciam na variação da variável dependente. Eles podem ser de grau 1,2 ou maior. O exemplo a seguir aborda uma regressão polinomial multipla de 2º grau, com duas variáveis independentes.

$$y = a + bx + b_1x_1 + b_2x_2 + b_3x_3\dots$$

Exemplo:

```
library(car)
```

```
data(iris)
head(iris)
```

```
##   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
```

VIF - variation inflation factor / fator de inflação da variância:

```
fit1<-lm(Sepal.Length ~ Sepal.Width + Petal.Length, data=iris)
```

```
vif(fit1) # ok, valores <5
```

```
##   Sepal.Width Petal.Length
##      1.224831     1.224831
```

```
summary(fit1)
```

```
##
## Call:
## lm(formula = Sepal.Length ~ Sepal.Width + Petal.Length, data = iris)
##
## Residuals:
##      Min      1Q  Median      3Q      Max
## -0.96159 -0.23489  0.00077  0.21453  0.78557
##
```

```

## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.24914   0.24797   9.07 7.04e-16 ***
## Sepal.Width  0.59552   0.06933   8.59 1.16e-14 ***
## Petal.Length 0.47192   0.01712  27.57 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3333 on 147 degrees of freedom
## Multiple R-squared:  0.8402, Adjusted R-squared:  0.838
## F-statistic: 386.4 on 2 and 147 DF,  p-value: < 2.2e-16

fit2 <- lm (Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width, data=iris)
vif(fit2) #problemático, valores>10

## Sepal.Width Petal.Length Petal.Width
##      1.270815     15.097572    14.234335

summary(fit2)

##
## Call:
## lm(formula = Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width,
##      data = iris)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.82816 -0.21989  0.01875  0.19709  0.84570
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.85600   0.25078   7.401 9.85e-12 ***
## Sepal.Width  0.65084   0.06665   9.765 < 2e-16 ***
## Petal.Length 0.70913   0.05672  12.502 < 2e-16 ***
## Petal.Width -0.55648   0.12755  -4.363 2.41e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3145 on 146 degrees of freedom
## Multiple R-squared:  0.8586, Adjusted R-squared:  0.8557
## F-statistic: 295.5 on 3 and 146 DF,  p-value: < 2.2e-16

```

```

fit3<-lm(Sepal.Length ~ Sepal.Width * Petal.Length, data=iris)
vif(fit3)# muito problemático, valores muito altos

##                               Sepal.Width          Petal.Length Sepal.Width:Petal.Length
##             6.457248                  81.811109                 69.223186

summary(fit3)

## 
## Call:
## lm(formula = Sepal.Length ~ Sepal.Width * Petal.Length, data = iris)
## 
## Residuals:
##   Min     1Q   Median     3Q    Max 
## -0.99594 -0.21165 -0.01652  0.21244  0.77249 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 1.40438   0.53253   2.637  0.00926 **  
## Sepal.Width  0.84996   0.15800   5.379 2.91e-07 *** 
## Petal.Length 0.71846   0.13886   5.174 7.45e-07 *** 
## Sepal.Width:Petal.Length -0.07701   0.04305  -1.789  0.07571 .  
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 0.3308 on 146 degrees of freedom
## Multiple R-squared:  0.8436, Adjusted R-squared:  0.8404 
## F-statistic: 262.5 on 3 and 146 DF,  p-value: < 2.2e-16

```

Outras funções interessantes:

```

coefficients(fit1) # coeficientes

## (Intercept) Sepal.Width Petal.Length
##      2.2491402    0.5955247    0.4719200

confint(fit1, level=0.95) # Intervalos de confiança

##                   2.5 %    97.5 %
## (Intercept) 1.7590943 2.7391860
## Sepal.Width  0.4585161 0.7325334
## Petal.Length 0.4380915 0.5057486

```

```
fitted(fit1) # valores previstos
```

```
##      1      2      3      4      5      6      7      8
## 4.994165 4.696402 4.768315 4.803147 5.053717 5.373951 4.934612 4.981804
##      9     10     11     12     13     14     15     16
## 4.636850 4.803147 5.160462 5.028996 4.696402 4.554826 5.197543 5.577329
##     17     18     19     20     21     22     23     24
## 5.185183 4.994165 5.314398 5.220014 5.076188 5.160462 4.864949 5.016636
##     25     26     27     28     29     30     31     32
## 5.170572 4.790786 5.028996 5.041357 4.934612 4.909891 4.850339 4.981804
##     33     34     35     36     37     38     39     40
## 5.398672 5.411032 4.803147 4.721123 4.946973 5.053717 4.649210 4.981804
##     41     42     43     44     45     46     47     48
## 4.946973 4.232343 4.768315 5.088549 5.408782 4.696402 5.267206 4.815507
##     49     50     51     52     53     54     55     56
## 5.160462 4.875060 6.372844 6.278460 6.407675 5.506527 6.087442 6.040250
##     57     58     59     60     61     62     63     64
## 6.432396 5.235736 6.146994 5.697545 5.091910 6.017779 5.446975 6.194186
##     65     66     67     68     69     70     71     72
## 5.675074 6.171715 6.159355 5.791929 5.682935 5.578440 6.420036 5.804290
##     73     74     75     76     77     78     79     80
## 6.050360 6.134634 6.005418 6.112163 6.181826 6.395315 6.099802 5.449225
##     81     82     83     84     85     86     87     88
## 5.471696 5.424504 5.697545 6.263849 6.159355 6.397564 6.313291 5.695295
##     89     90     91     92     93     94     95     96
## 5.970587 5.625632 5.873953 6.206547 5.685185 5.176183 5.839121 6.017779
##     97     98     99    100    101    102    103    104
## 5.958226 6.005418 5.153712 5.851482 7.045892 6.263849 6.820043 6.618914
##    105    106    107    108    109    110    111    112
## 6.772851 7.150387 5.861592 6.949258 6.475088 7.271741 6.561612 6.358233
##    113    114    115    116    117    118    119    120
## 6.631275 6.097552 6.323402 6.655996 6.631275 7.673998 7.053753 5.918895
##    121    122    123    124    125    126    127    128
## 6.844764 6.229018 7.078474 6.169465 6.904316 6.986340 6.181826 6.348123
##    129    130    131    132    133    134    135    136
## 6.559362 6.772851 6.795322 7.532422 6.559362 6.323402 6.440257 6.914427
##    137    138    139    140    141    142    143    144
## 6.916677 6.690827 6.300931 6.643635 6.738019 6.502059 6.263849 6.939148
##    145    146    147    148    149    150
## 6.904316 6.489699 6.097552 6.489699 6.822293 6.442507
```

```
residuals(fit1) # resíduos
```

##	1	2	3	4	5	6
----	---	---	---	---	---	---

```

##  0.105835164  0.203597538 -0.068315407 -0.203146940 -0.053717311  0.026049253
##    7           8           9           10          11          12
## -0.334612361  0.018195635 -0.236849987  0.096853060  0.239538210 -0.228996369
##   13          14          15          16          17          18
##  0.103597538 -0.254826450  0.602456797  0.122670886  0.214817268  0.105835164
##   19          20          21          22          23          24
##  0.385601728 -0.120014265  0.323811627 -0.060461790 -0.264949295  0.083364102
##   25          26          27          28          29          30
## -0.370572381  0.209213530 -0.028996369  0.158643160  0.265387639 -0.209891419
##   31          32          33          34          35          36
## -0.050338944  0.418195635 -0.198671689  0.088967840  0.096853060  0.278876596
##   37          38          39          40          41          42
##  0.553027168 -0.153717311 -0.249210458  0.118195635  0.053027168  0.267656866
##   43          44          45          46          47          48
## -0.368315407 -0.088548844 -0.308782280  0.103597538 -0.167206269 -0.215507411
##   49          50          51          52          53          54
##  0.139538210  0.124940114  0.627156459  0.121540467  0.492324926 -0.006527240
##   55          56          57          58          59          60
##  0.412558362 -0.340249634 -0.132396016 -0.335735687  0.453005887 -0.497545135
##   61          62          63          64          65          66
## -0.091909796 -0.117778572  0.553025235 -0.094186117 -0.075074073  0.528284945
##   67          68          69          70          71          72
## -0.559354584  0.008070857  0.517065215  0.021559814 -0.520035545  0.295710386
##   73          74          75          76          77          78
##  0.249639775 -0.034633642  0.394581899  0.487837420  0.618174354  0.304685397
##   79          80          81          82          83          84
## -0.099802109  0.250775355  0.028304293  0.075496297  0.102454865 -0.263849183
##   85          86          87          88          89          90
## -0.759354584 -0.397564483  0.386708934  0.604704744 -0.370586568 -0.125632190
##   91          92          93          94          95          96
## -0.373952680 -0.106546588  0.114815336 -0.176183212 -0.239121147 -0.317778572
##   97          98          99          100         101         102
## -0.258226097  0.194581899 -0.053712150 -0.151481618 -0.745892067 -0.463849183
##  103          104          105          106          107          108
##  0.279957361 -0.318914152 -0.272850635  0.449613334 -0.961592209  0.350741820
##  109          110          111          112          113          114
##  0.224911740 -0.071741496 -0.061611557  0.041766810  0.168725377 -0.397552229
##  115          116          117          118          119          120
## -0.523401657 -0.255995565 -0.131274623  0.026001531  0.646247222  0.081105196
##  121          122          123          124          125          126
##  0.055236419 -0.629017650  0.621526280  0.130534825 -0.204316055  0.213660408
##  127          128          129          130          131          132
##  0.018174354 -0.248122599 -0.159361677  0.427149365  0.604678303  0.367577543
##  133          134          135          136          137          138
## -0.159361677 -0.023401657 -0.340256727  0.785573353 -0.616676526 -0.290827098
##  139          140          141          142          143          144

```

```
## -0.300930595  0.256364906 -0.038019102  0.397940918 -0.463849183 -0.139147588
##          145           146           147           148           149           150
## -0.204316055  0.210301389  0.202447771  0.010301389 -0.622292518 -0.542506607
```

```
anova(fit1) # tabela anova
```

```
## Analysis of Variance Table
##
## Response: Sepal.Length
##             Df Sum Sq Mean Sq F value    Pr(>F)
## Sepal.Width     1  1.412   1.412  12.714 0.0004902 ***
## Petal.Length    1 84.427  84.427 760.059 < 2.2e-16 ***
## Residuals     147 16.329   0.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
vcov(fit1) # matriz de covariância
```

```
##              (Intercept) Sepal.Width Petal.Length
## (Intercept)  0.061488936 -0.0166054883 -0.0026556385
## Sepal.Width -0.016605488  0.0048063941  0.0005084458
## Petal.Length -0.002655638  0.0005084458  0.0002930149
```

8.5.1 Referência

MELO, M. P.; PETERNELI, L. A. Conhecendo o R: Um visão mais que estatística. Viçosa, MG: UFV, 2013. 222p. Cap. 1.

ZEVIANI, W. M. Estatística Básica e Experimentação no R. 45p.

<http://www.leg.ufpr.br/~paulojus/>

<https://docs.r4photobiology.info/ggpmisc/articles/user-guide-1.html>

8.6 Regressão não linear

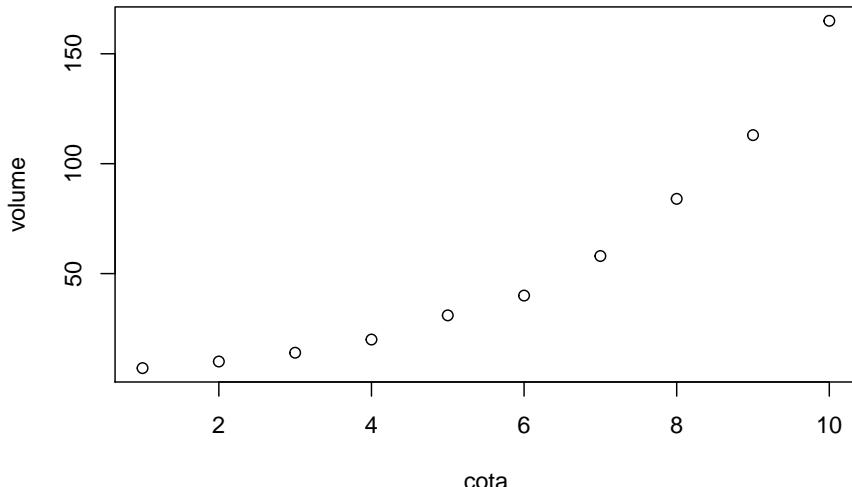
Para o ajuste de regressões não-lineares com o R, aconselhamos o uso da função **nls()** (modelo, dados e valores iniciais estimados dos parâmetros)

Exemplo:

Num projeto de construção de uma barragem é de grande interesse equacionar a relação entre a cota do nível d'água e o volume armazenado quando esta cota é atingida. Essa relação é obtida a partir de um diagrama cota-volume estimado

através do levantamento topográfico da região onde será construída a barragem e suas respectivas curvas de nível. Suponha os dados a seguir, com a cota dada em metros e o volume em quilômetros cúbicos:

```
cota<-c(1,2,3,4,5,6,7,8,9,10)
volume<-c(7,10,14,20,31,40,58,84,113,165)
dados<-data.frame(cota,volume)
plot(dados)
```



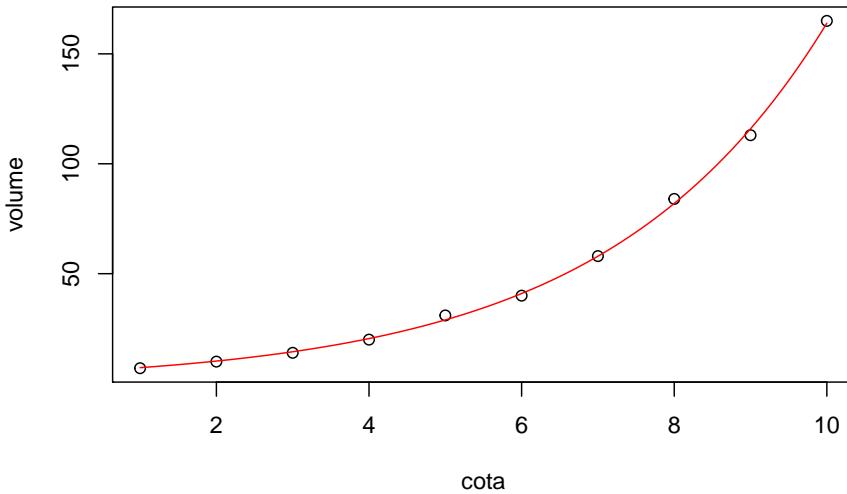
```
funcao <- volume~a*exp(b*cota)
exponencial <- nls(funcao, #modelo que se deseja ajustar($)
                     dados, #data.frame com o conjunto de dados
                     start=c(a=1,b=1)) #valores iniciais dos parâmetros($$)
summary(exponencial)
```

```
##
## Formula: volume ~ a * exp(b * cota)
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
##   a 5.116389  0.227553  22.48 1.62e-08 ***
##   b 0.346720  0.004879  71.06 1.71e-12 ***
##   ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##  
## Residual standard error: 1.559 on 8 degrees of freedom  
##  
## Number of iterations to convergence: 14  
## Achieved convergence tolerance: 2.694e-07
```

Desenhando a curva ajustada:

```
plot(dados)  
curve(5.1163887*exp(0.34672*x), 1, #limite inferior eixo das abscissas  
      10, #limite superior  
      add=T, #acrescentar no gráfico anterior  
      col=2) #cor da curva (2 = vermelha)
```



8.6.1 Curva de retenção de água no solo

O objetivo é determinar da curva de retenção de água no solo estimada segundo modelo de van Genutchen para cada uma das amostras.

Fonte: <http://www.leg.ufpr.br/~paulojus/embrapa/>

```
cra <- read.table("http://www.leg.ufpr.br/~paulojus/dados/cra.csv", head = T, sep = ",")  
head(cra)
```

```

##   am pot      u
## 1 30 10 0.3071
## 2 30 19 0.2931
## 3 30 30 0.2828
## 4 30 45 0.2753
## 5 30 63 0.2681
## 6 30 64 0.2628

cra <- transform(cra, am = as.factor(am))
summary(cra)

##   am          pot             u
## 30:15   Min.   : 10.0   Min.   :0.0636
## 41:13   1st Qu.: 58.5   1st Qu.:0.1199
##           Median :107.5   Median :0.1969
##           Mean   :2139.8   Mean   :0.1879
##           3rd Qu.:1550.0   3rd Qu.:0.2436
##           Max.   :26300.0  Max.   :0.3071

```

Isolar a amostra 30:

```

cra30 <- subset(cra, am == "41")
cra30

##   am pot      u
## 16 41 11 0.2407
## 17 41 14 0.2209
## 18 41 34 0.2100
## 19 41 65 0.2004
## 20 41 94 0.1935
## 21 41 110 0.1842
## 22 41 157 0.1706
## 23 41 1700 0.1023
## 24 41 1200 0.1180
## 25 41 1100 0.1261
## 26 41 1500 0.1101
## 27 41 2300 0.0917
## 28 41 12100 0.0636

```

Figura com o eixo horizontal com logarítmico (base 10), dados de umidade versus pressão aplicada na amostra:

```
with(cra30, plot(u ~ log10(pot), xlab = expression(log[10](Psi[m])), ylab = expression(theta), yl
```

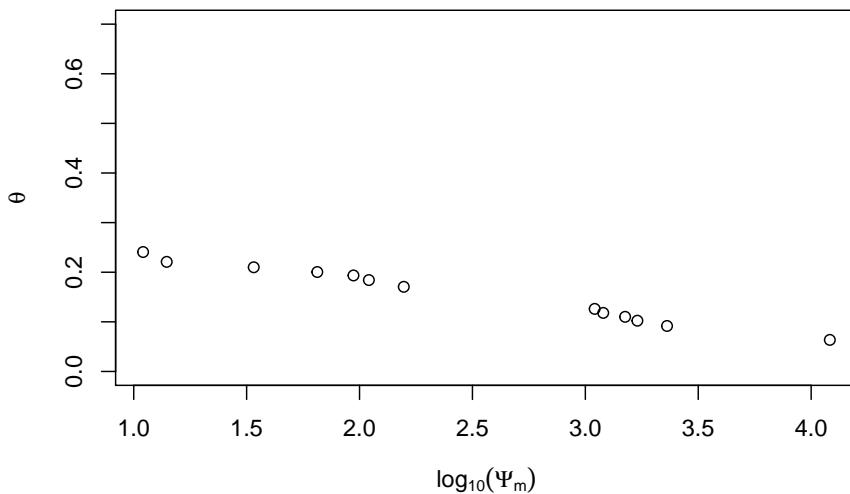
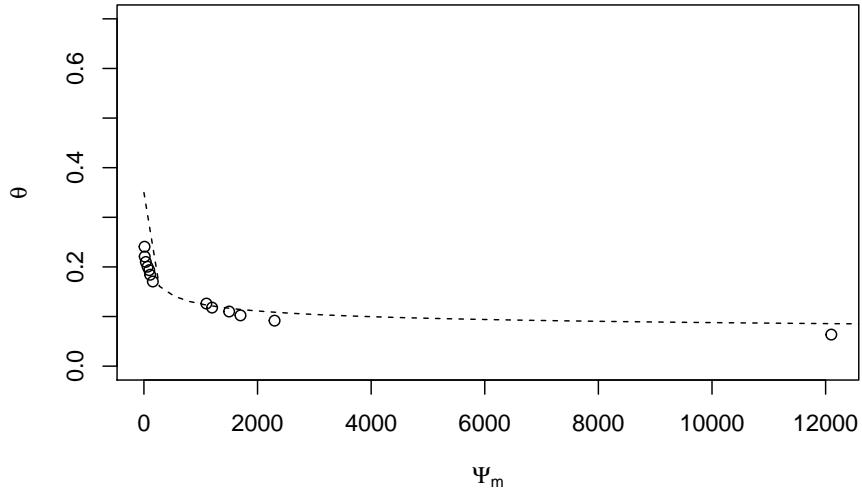


Figura dos dados de umidade versus pressão aplicada na amostra:

```
with(cra30, plot(u ~ pot, xlab = expression(Psi[m]),
                  ylab = expression(theta),
                  ylim = c(0,0.7)))
  curve(0.05 + (0.35 - 0.05)/((1 + (0.1 * x)^1.3)^(1 - 1/1.3)), from = 0,
         to = 27000, add = T, lty = 2)
```



Definidos os valores iniciais prossegue-se com o ajuste do modelo conforme os comandos a seguir:

```
fit30 = nls(u ~ ur + (us - ur)/((1 + (alpha * pot)^n)^(1 - 1/n)), data = cra30, start =
summary(fit30)

##
## Formula: u ~ ur + (us - ur)/((1 + (alpha * pot)^n)^(1 - 1/n))
##
## Parameters:
##             Estimate Std. Error t value Pr(>|t|)
## us      0.243148   0.009446 25.741 9.71e-10 ***
## ur     -0.122402   0.171614 -0.713   0.494
## alpha   0.035928   0.022324  1.609   0.142
## n       1.113320   0.079473 14.009 2.04e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.006207 on 9 degrees of freedom
##
## Number of iterations to convergence: 9
## Achieved convergence tolerance: 9.545e-06
```

Chapter 9

Multivariada

Entrada dos dados

Dados entre parênteses são os valores observados das respectivas variáveis, X1, X2, X3, X4 e X5.

```
X1<-c(0.0475,0.04715,0.0471,0.0469,0.0469,0.04635)
X2<-c(0.366,0.359,0.341,0.348,0.351,0.350)
X3<-c(0.388,0.345,0.367,0.353,0.355,0.426)
X4<-c(0.302,0.296,0.279,0.286,0.285,0.279)
X5<-c(1.29,1.16,1.31,1.23,1.24,1.52)
```

```
Y<-cbind(X1,X2,X3,X4,X5)
```

```
S<-cov(cbind(X1,X2,X3,X4,X5));
S
```

```
##          X1          X2          X3          X4          X5
## X1  1.446667e-07  1.76e-06 -5.603333e-06  2.726667e-06 -3.116667e-05
## X2  1.760000e-06  7.71e-05  1.480000e-05  7.610000e-05 -2.610000e-04
## X3 -5.603333e-06  1.48e-05  9.150667e-04 -6.873333e-05  3.563333e-03
## X4  2.726667e-06  7.61e-05 -6.873333e-05  8.696667e-05 -6.036667e-04
## X5 -3.116667e-05 -2.61e-04  3.563333e-03 -6.036667e-04  1.525667e-02
```

```
autoS<-eigen(S);
autoS
```

```
## eigen() decomposition
## $values
```

```

## [1] 1.612000e-02 2.134340e-04 2.489500e-06 1.859592e-08 1.729972e-09
##
## $vectors
## [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] -0.001968185 -0.0120528 -0.11495526  0.99242360 -0.04161186
## [2,] -0.015794397 -0.5786029  0.80786827  0.08928368  0.06592820
## [3,]  0.228128328 -0.5960180 -0.35869625 -0.07671633 -0.67688376
## [4,] -0.037679522 -0.5458098 -0.44622410 -0.02872057  0.70762359
## [5,]  0.972771496  0.1092141  0.07971945  0.02033617  0.18713406

S<-cov(cbind(X1,X2,X3,X4,X5))

S<-cov(cbind(X1,X2,X3,X4,X5));
S

##          X1          X2          X3          X4          X5
## X1  1.446667e-07  1.76e-06 -5.603333e-06  2.726667e-06 -3.116667e-05
## X2  1.760000e-06  7.71e-05  1.480000e-05  7.610000e-05 -2.610000e-04
## X3 -5.603333e-06  1.48e-05  9.150667e-04 -6.873333e-05  3.563333e-03
## X4  2.726667e-06  7.61e-05 -6.873333e-05  8.696667e-05 -6.036667e-04
## X5 -3.116667e-05 -2.61e-04  3.563333e-03 -6.036667e-04  1.525667e-02

autoS<-eigen(S);
autoS

## eigen() decomposition
## $values
## [1] 1.612000e-02 2.134340e-04 2.489500e-06 1.859592e-08 1.729972e-09
##
## $vectors
## [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] -0.001968185 -0.0120528 -0.11495526  0.99242360 -0.04161186
## [2,] -0.015794397 -0.5786029  0.80786827  0.08928368  0.06592820
## [3,]  0.228128328 -0.5960180 -0.35869625 -0.07671633 -0.67688376
## [4,] -0.037679522 -0.5458098 -0.44622410 -0.02872057  0.70762359
## [5,]  0.972771496  0.1092141  0.07971945  0.02033617  0.18713406

S.cov<-prcomp(cbind(X1,X2,X3,X4,X5))

summary(S.cov)

## Importance of components:
##                               PC1        PC2        PC3        PC4        PC5

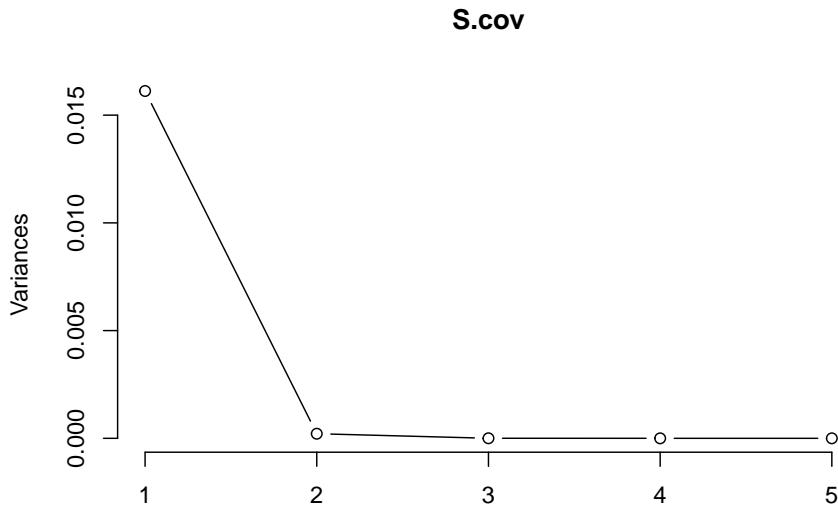
```

```

## Standard deviation      0.1270 0.01461 0.001578 0.0001364 4.159e-05
## Proportion of Variance 0.9868 0.01307 0.000150 0.0000000 0.000e+00
## Cumulative Proportion  0.9868 0.99985 1.000000 1.0000000 1.000e+00

screeplot(S.cov,type="lines")

```



Resultados a partir da matriz de covariância

```

R<-cor(cbind(X1,X2,X3,X4,X5));
R

##          X1          X2          X3          X4          X5
## X1  1.0000000  0.52698863 -0.48700773  0.7687257 -0.6634013
## X2  0.5269886  1.00000000  0.05571962  0.9293537 -0.2406487
## X3 -0.4870077  0.05571962  1.00000000 -0.2436490  0.9536747
## X4  0.7687257  0.92935375 -0.24364900  1.0000000 -0.5240720
## X5 -0.6634013 -0.24064868  0.95367468 -0.5240720  1.0000000

autoR<-eigen(R);
autoR

## eigen() decomposition
## $values
## [1] 3.163658e+00 1.525782e+00 3.025159e-01 8.042938e-03 1.732466e-06

```

```

## 
## $vectors
##          [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] -0.4975623 0.03740992 0.8416617 0.20648920 -0.0003848684
## [2,] -0.3778884 0.57085303 -0.3984395 0.61010372 0.0186158858
## [3,]  0.3684580 0.60263117  0.2311883 -0.16487762 -0.6484147684
## [4,] -0.4961633 0.37260806 -0.1269516 -0.74522471 0.2085872370
## [5,]  0.4771716 0.41319029  0.2515689  0.05090347 0.7319173132

SR<-prcomp(cbind(X1,X2,X3,X4,X5), cor=TRUE, scale=TRUE)

## Warning: In prcomp.default(cbind(X1, X2, X3, X4, X5), cor = TRUE, scale = TRUE) :
##   extra argument 'cor' will be disregarded

SR

## Standard deviations (1, ..., p=5):
## [1] 1.778667446 1.235225294 0.550014473 0.089682431 0.001316232
##
## Rotation (n x k) = (5 x 5):
##          PC1      PC2      PC3      PC4      PC5
## X1 -0.4975623 0.03740992 -0.8416617 0.20648920 -0.0003848684
## X2 -0.3778884 0.57085303  0.3984395 0.61010372 0.0186158858
## X3  0.3684580 0.60263117 -0.2311883 -0.16487762 -0.6484147684
## X4 -0.4961633 0.37260806  0.1269516 -0.74522471 0.2085872370
## X5  0.4771716 0.41319029 -0.2515689  0.05090347 0.7319173132

summary(SR)

## Importance of components:
##          PC1      PC2      PC3      PC4      PC5
## Standard deviation 1.7787 1.2352 0.5500 0.08968 0.001316
## Proportion of Variance 0.6327 0.3052 0.0605 0.00161 0.000000
## Cumulative Proportion 0.6327 0.9379 0.9984 1.00000 1.000000

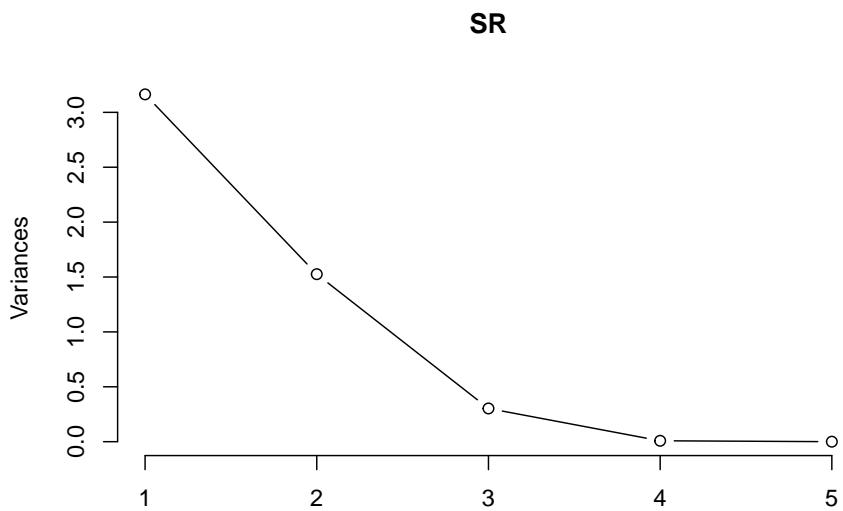
scores<-SR$x;
scores

##          PC1      PC2      PC3      PC4      PC5
## [1,] -1.82622087 1.8010525 -0.4542054 0.0003485361 -0.0007263451
## [2,] -1.77385089 -0.2196976 0.5143779 -0.0157753617 0.0019677049
## [3,]  0.81813538 -1.1340293 -0.8968309 0.0067988051 0.0008807578
## [4,] -0.07349764 -0.9654427 0.2286037 -0.1314456477 -0.0014616145
## [5,] -0.08640991 -0.7370654 0.3154692 0.1501343080 -0.0010828159
## [6,]  2.94184393 1.2551825 0.2925855 -0.0100606398 0.0004223128

```

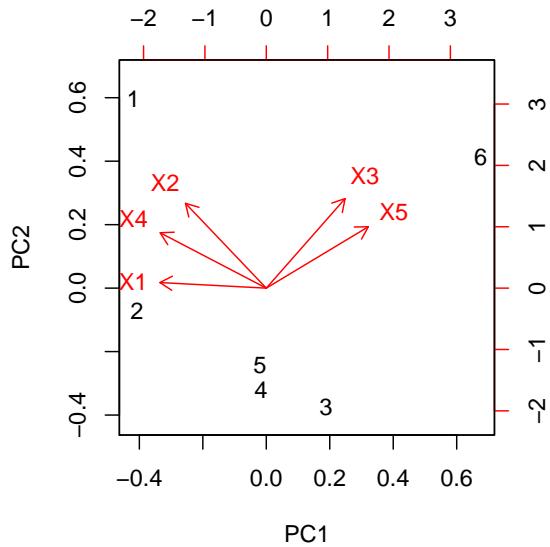
Resultados a partir da matriz de correlação R

```
screeplot(SR,type="lines")
```



Dispersão gráfica

```
biplot(SR)
```



Chapter 10

Dados climáticos

10.1 Precipitação pluvial

Tutorial do HydroTSM - <http://goo.gl/9u1PO>

Instalação:

```
install.packages("hydroTSM")
```

Carregar o pacote:

```
library(hydroTSM)
```

Carregando dados diários de Precipitação da estação de Rondonópolis-MT, com dados de 1 de janeiro de 1980 a 31 de dezembro de 2013:

```
roo<-read.zoo("https://www.dropbox.com/s/j1v3euj51dztv3r/1980-2013_pp.csv?dl=1",sep=";", dec=".")
```

Selecionar somente 5 anos da série:

```
x <- window(roo, start=as.Date("2009-01-01"))
```

Valores de Precipitação mensal:

```
( m <- daily2monthly(x, FUN=sum) )
```

```
## 2009-01-01 2009-02-01 2009-03-01 2009-04-01 2009-05-01 2009-06-01 2009-07-01  
##      200.17     340.88     312.69     161.13      32.76      66.03     19.83
```

```
## 2009-08-01 2009-09-01 2009-10-01 2009-11-01 2009-12-01 2010-01-01 2010-02-01
##    77.99     131.10     127.09     258.52     344.81     339.07     365.16
## 2010-03-01 2010-04-01 2010-05-01 2010-06-01 2010-07-01 2010-08-01 2010-09-01
##    247.46     129.92     14.29      0.22      0.49      0.00      0.62
## 2010-10-01 2010-11-01 2010-12-01 2011-01-01 2011-02-01 2011-03-01 2011-04-01
##    143.24     181.52     206.55     418.55     419.73     537.55     146.77
## 2011-05-01 2011-06-01 2011-07-01 2011-08-01 2011-09-01 2011-10-01 2011-11-01
##    10.42      8.89      0.00      4.34     10.00     212.64     105.84
## 2011-12-01 2012-01-01 2012-02-01 2012-03-01 2012-04-01 2012-05-01 2012-06-01
##    348.87     437.63     256.04     248.56     169.66     115.69     135.74
## 2012-07-01 2012-08-01 2012-09-01 2012-10-01 2012-11-01 2012-12-01 2013-01-01
##     0.15      0.00     82.56     93.41     310.81     158.77     349.01
## 2013-02-01 2013-03-01 2013-04-01 2013-05-01 2013-06-01 2013-07-01 2013-08-01
##    325.68     287.80     135.50     44.85     37.29     19.11      0.00
## 2013-09-01 2013-10-01 2013-11-01 2013-12-01
##    61.70     82.57     151.95     222.90
```

Datas dos valores diários de “x”:

```
dates <- time(x)
```

Quantidade de anos em ‘x’ (necessário para cálculos):

```
( nyears <- yip(from=start(x), to=end(x), out.type="nbr" ) )
```

```
## [1] 5
```

Análise exploratório do dados

Resumo estatístico:

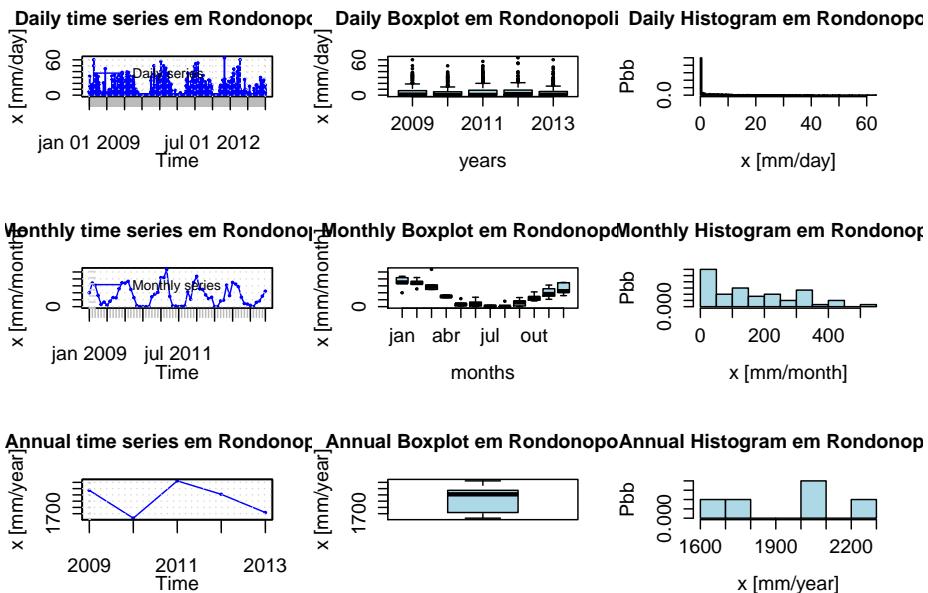
```
smry(x)
```

```
##                   Index          x
## Min.   2009-01-01 0.0000
## 1st Qu. 2010-04-02 0.0000
## Median  2011-07-02 0.5300
## Mean    2011-07-02 5.2860
## 3rd Qu. 2012-09-30 7.3950
## Max.   2013-12-31 63.3500
## IQR      <NA>       7.3950
## sd       <NA>       8.9907
## cv       <NA>       1.7008
## Skewness <NA>       2.5413
```

```
## Kurtosis      <NA>    7.6836
## NA's         <NA>    0.0000
## n            <NA> 1826.0000
```

Usando a função `hydroplot`, que (por padrão) representa 9 gráficos diferentes: gráficos de 3 ts, 3 gráficos de caixa e 3 histogramas com um resumo “x”. Para este exemplo, somente plotagens diárias e mensais são produzidas e apenas os dados iniciados em 1 de janeiro de 2009 são plotados:

```
hydroplot(x, var.type="Precipitation", main="em Rondonopolis",
          pfreq = "dma", from="2009-01-01")
```



Quantidade de dias com informação (não NA) por ano:

```
dwi(x)
```

```
## 2009 2010 2011 2012 2013
## 365 365 365 366 365
```

Quantidade de dias com informação (não NA) por mês por ano:

```
dwi(x, out.unit="mpy")
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 2009  31  28  31  30  31  30  31  31  30  31  30  31
## 2010  31  28  31  30  31  30  31  31  30  31  30  31
## 2011  31  28  31  30  31  30  31  31  30  31  30  31
## 2012  31  29  31  30  31  30  31  31  30  31  30  31
## 2013  31  28  31  30  31  30  31  31  30  31  30  31
```

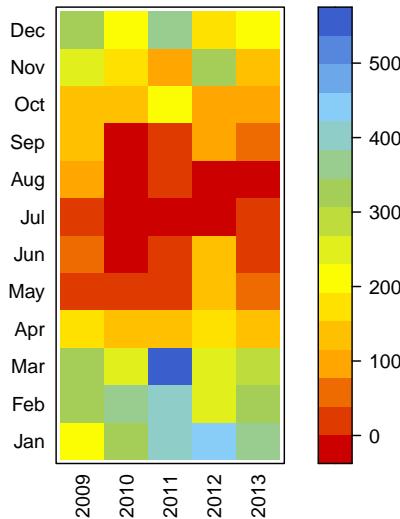
Plotar os valores mensais de precipitação para cada ano, para identificar meses secos/umidos:

```
# Daily zoo to monthly zoo
m <- daily2monthly(x, FUN=sum, na.rm=TRUE)

# Creating a matrix with monthly values per year in each column
M <- matrix(m, ncol=12, byrow=TRUE)
colnames(M) <- month.abb
rownames(M) <- unique(format(time(m), "%Y"))

# Plotting the monthly precipitation values
require(lattice)
## Loading required package: lattice
print(matrixplot(M, ColorRamp="Precipitation",
main="Precipitação mensal de Rondonópolis-MT (mm/mês)"))
```

Precipitação mensal de Rondonópolis-MT (mm/mês)



10.1.1 Análise anual dos dados

Valores anuais de precipitação:

```
daily2annual(x, FUN=sum, na.rm=TRUE)

## 2009-01-01 2010-01-01 2011-01-01 2012-01-01 2013-01-01
##     2073.00     1628.54     2223.60     2009.02     1718.36
```

Precipitação média anual:

```
mean( daily2annual(x, FUN=sum, na.rm=TRUE) )

## [1] 1930.504
```

Outra forma (mais útil para ‘streamflows’, onde $\text{FUN} = \text{mean}$): A função anual aplica FUN duas vezes sobre x :

- (i) primeiramente, sobre todos os elementos de x pertencentes ao mesmo ano, para obter os correspondentes valores anuais, e
- (ii) em segundo lugar, acima de todos os valores anuais de x obtidos anteriormente, a fim de obter um único valor anual.

```
annualfunction(x, FUN=sum, na.rm=TRUE) / nyears

##      value
## 1930.504
```

10.1.2 Análise mensal dos dados - BOXPLOT

Mediana dos valores mensais na estação de Rondonópolis-MT:

```
monthlyfunction(m, FUN=median, na.rm=TRUE)
```

```
##      Jan      Feb      Mar      Apr      May      Jun      Jul      Aug      Sep      Oct      Nov
## 349.01 340.88 287.80 146.77  32.76   37.29    0.49    0.00   61.70 127.09 181.52
##      Dec
## 222.90
```

Vetor com as abreviaturas de três letras para os nomes dos meses:

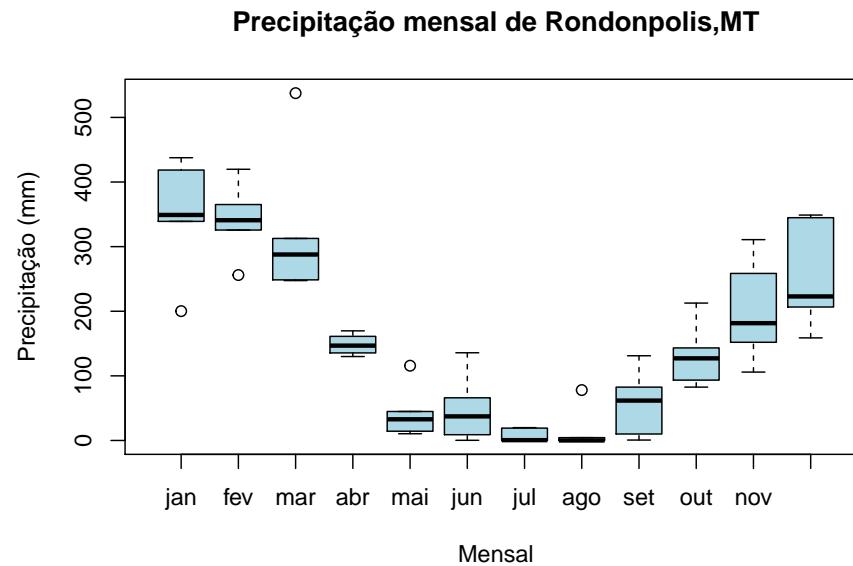
```
cmonth <- format(time(m), "%b")
```

Criando fatores mensais ordenados:

```
months <- factor(cmonth, levels=unique(cmonth), ordered=TRUE)
```

Boxplot dos valores mensais:

```
boxplot( coredata(m) ~ months,
         col="lightblue",
         main="Precipitação mensal de Rondonpolis,MT",
         ylab="Precipitação (mm)", xlab="Mensal")
```



10.1.3 Análise sazonal

Valores sazonais médios de precipitação:

```
seasonalfunction(x, FUN=sum, na.rm=TRUE) / nyyears
```

```
##      DJF      MAM      JJA      SON
## 946.764 519.010  74.016 390.714
```

Extraindo os valores sazonais para cada ano

Dezembro, Janeiro e Fevereiro:

```
( DJF <- dm2seasonal(x, season="DJF", FUN=sum) )
```

```
## 2009 2010 2011 2012 2013
## 541.05 1049.04 1044.83 1042.54 833.46
```

Março, Abril e Maio:

```
( MAM <- dm2seasonal(m, season="MAM", FUN=sum) )
```

```
## 2009 2010 2011 2012 2013
## 506.58 391.67 694.74 533.91 468.15
```

Junho, Julho e Agosto:

```
( JJA <- dm2seasonal(m, season="JJA", FUN=sum) )
```

```
## 2009 2010 2011 2012 2013
## 163.85 0.71 13.23 135.89 56.40
```

Setembro, Outubro e Novembro:

```
( SON <- dm2seasonal(m, season="SON", FUN=sum) )
```

```
## 2009 2010 2011 2012 2013
## 516.71 325.38 328.48 486.78 296.22
```

Plotar a evolução temporal dos valores da precipitação sazonal:

```
x11()
hydroplot(x, pfreq="seasonal", FUN=sum, stype="default")
```

10.1.4 Alguns índices extremos

Etapas comuns para a análise desta série:

Carregando dados diários de precipitação da estação Rondonópolis-MT, com dados de 01/Jan/1980 a 31/Dez/2013:

```
data(roo)
```

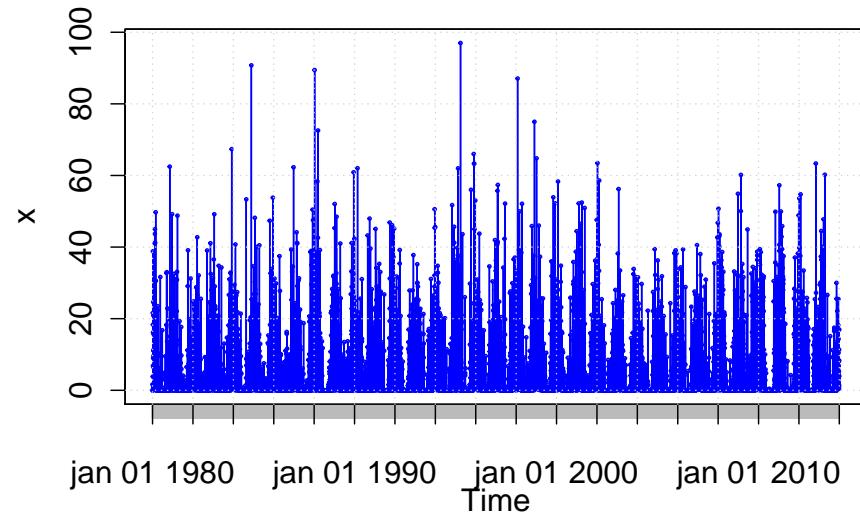
```
## Warning in data(roo): data set 'roo' not found
```

Selecionar a data inicial:

```
x <- window(roo, start=as.Date("1980-01-01"))
```

Plotar a série temporal selecionada:

```
hydroplot(x, ptype="ts", pfreq="o", var.unit="mm")
```



Contagem e plotagem do número de dias no período em que a precipitação foi maior que 80 mm:

```
( R10mm <- length( x[x>80] ) )
```

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

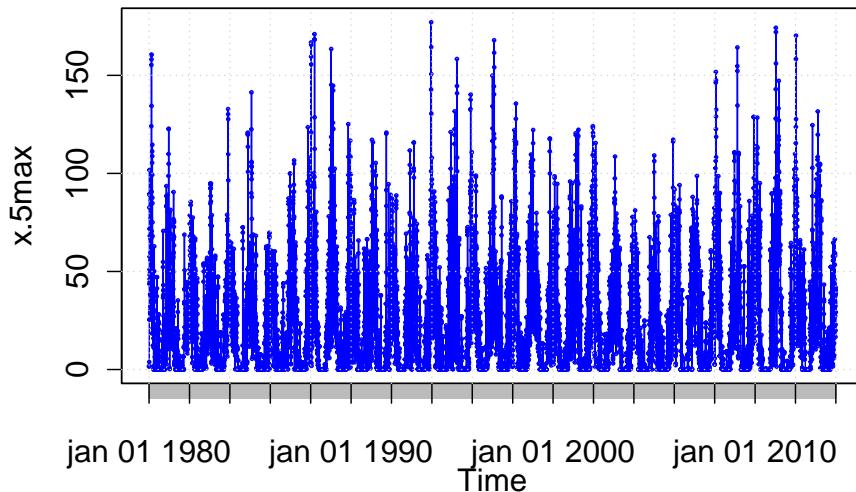
10.1.5 Precipitação total chuva

Calculando a Precipitação total (acumulada) de 5 dias:

```
x.5max <- rollapply(data=x, width=5, FUN=sum, fill=NA, partial= TRUE,
align="center")

hydroplot(x.5max, ptype="ts+boxplot", pfreq="o", var.unit="mm")

## [Note: pfreq='o' => ptype has been changed to 'ts']
```



Valor anual máximo de precipitação total em 5 dias:

```
(x.5max.annual <- daily2annual(x.5max, FUN=max, na.rm=TRUE))

## 1980-01-01 1981-01-01 1982-01-01 1983-01-01 1984-01-01 1985-01-01 1986-01-01
##      160.54      90.51     85.49    132.81    120.78    141.33     99.98
## 1987-01-01 1988-01-01 1989-01-01 1990-01-01 1991-01-01 1992-01-01 1993-01-01
##      123.58     171.03    163.45     86.39    120.82    111.70    177.09
## 1994-01-01 1995-01-01 1996-01-01 1997-01-01 1998-01-01 1999-01-01 2000-01-01
##      121.09     158.35    149.88    167.91    135.63    122.16     98.53
## 2001-01-01 2002-01-01 2003-01-01 2004-01-01 2005-01-01 2006-01-01 2007-01-01
##      124.10     115.52    108.61     81.14    117.21     94.05    98.69
## 2008-01-01 2009-01-01 2010-01-01 2011-01-01 2012-01-01 2013-01-01
##      151.72     164.22    128.46    174.28    170.24    131.61
```

- *Nota 1:* para este cálculo, é utilizada uma janela móvel centrada no dia atual. Se o usuário quiser precipitação total de 5 dias acumulada nos 4

dias anteriores ao dia atual + a precipitação no corrente dia, o usuário tem que modificar a janela móvel.

- *Nota 2:* Para os dois primeiros e os últimos dois valores, a largura da janela é adaptada para ignorar valores a série temporal

10.2 Evapotranspiração

A evapotranspiração é a forma pela qual a água da superfície terrestre passa para a atmosfera no estado de vapor, tendo papel importante no Ciclo Hidrológico. Esse processo envolve a evaporação da água de superfícies de água livre (rios, lagos, represas, oceano, etc), dos solos e da vegetação úmida (que foi interceptada durante uma chuva) e a transpiração dos vegetais

Apagar objetos antigos:

```
rm(list = ls())
```

Baixar dados

[a.txt] (<https://www.dropbox.com/s/qixbu9tqnnfsuyt/a.txt?dl=0>)

```
a = read.table("https://www.dropbox.com/s/qixbu9tqnnfsuyt/a.txt?dl=1", sep=";", head=1)
head(a)

##           Data      Rs Tmax Tmin RHmax      u2 RHmin
## 1 1980-01-01 18.51 31.1 21.8 82.00 2.000000 82.00
## 2 1980-01-02  9.65 26.5 21.8 91.50 1.000000 91.50
## 3 1980-01-03 13.70 29.9 21.0 86.00 1.000000 86.00
## 4 1980-01-04 13.86 30.3 22.0 82.75 1.666667 82.75
## 5 1980-01-05  9.66 25.1 21.6 90.97 1.070000 90.97
## 6 1980-01-06 16.10 31.2 20.8 80.67 1.380000 80.67
```

Transformar em Data e Dia Juliano:

```
a$Data = as.Date(a$Data)
```

Dia Juliano: calendário de dias corridos, trata-se de uma sequência de números inteiros uma para cada dia.

```
a$DiaJuliano = as.numeric(format(a$Data, trim = T, '%j'))
```

Verificar

```
head(a)
```

```
##          Data      Rs Tmax Tmin RHmax      u2 RHmin DiaJuliano
## 1 1980-01-01 18.51 31.1 21.8 82.00 2.000000 82.00           1
## 2 1980-01-02  9.65 26.5 21.8 91.50 1.000000 91.50           2
## 3 1980-01-03 13.70 29.9 21.0 86.00 1.000000 86.00           3
## 4 1980-01-04 13.86 30.3 22.0 82.75 1.666667 82.75           4
## 5 1980-01-05  9.66 25.1 21.6 90.97 1.070000 90.97           5
## 6 1980-01-06 16.10 31.2 20.8 80.67 1.380000 80.67           6
```

Constantes utilizadas no calculo da ET.

Altura acima do nível do mar (m):

```
altitude = 259.38
```

Pressão atmosférica local, calculada com base na altitude (kPa):

```
Patm = 101.3*((293-0.0065*altitude)/293)**5.26
```

Coeficiente psicrométrico (kPa/°C):

```
gama = 0.665*(10**-3)*Patm
```

Latitude:

```
lat = -7.53
```

Transformar em radianos:

```
corr = pi/180
```

Declinação solar (rad):

```
decl = 23.45*sin(corr*((a$DiaJuliano-80)*360/365))
```

Ângulo horário do nascer ao por do Sol em (rad):

```
hn = 1/corr*acos(-tan(corr*lat)*tan(corr*decl))
```

Número máximo de horas de luz solar em um dia (h):

```
a$N = 2*hn/15
```

Distância relativa da Terra ao Sol:

```
s = (1+0.033*cos(a$DiaJuliano*360/365))
```

Ângulo de horário (ângulo da radiação do Sol)

```
t = corr*hn*sin(corr*lat)*sin(corr*decl)
u = cos(corr*lat)*cos(corr*decl)*sin(corr*hn)
```

Radiação Solar Extraterrestre diária (mm/dia):

```
a$Qo = 37.6*s*(t+u)
```

Verificar dados:

```
head(a)
```

	Data	Rs	Tmax	Tmin	RHmax	u2	RHmin	DiaJuliano	N	Qo
## 1	1980-01-01	18.51	31.1	21.8	82.00	2.000000	82.00	1	12.42742	38.08036
## 2	1980-01-02	9.65	26.5	21.8	91.50	1.000000	91.50	2	12.42559	36.92744
## 3	1980-01-03	13.70	29.9	21.0	86.00	1.000000	86.00	3	12.42363	36.20749
## 4	1980-01-04	13.86	30.3	22.0	82.75	1.666667	82.75	4	12.42153	36.57674
## 5	1980-01-05	9.66	25.1	21.6	90.97	1.070000	90.97	5	12.41929	37.71584
## 6	1980-01-06	16.10	31.2	20.8	80.67	1.380000	80.67	6	12.41691	38.61551

coeficientes para climas médios:

```
x = 0.25
```

coeficientes para climas médios:

```
b = 0.50
```

Insolação (h):

```
n.est = (a$Rs/a$Qo-x)*a$N/b;
n.est
```

```

## [1] 5.86766463 0.28139699 3.18975978 3.20299747 0.15215691 4.14553255
## [7] 5.60035036 5.06395663 7.18184567 2.09257628 0.53824724 0.64483607
## [13] 0.09397131 2.98306242 0.32106628 5.31197003 1.55169394 6.69674498
## [19] 10.16813689 4.16127733 10.54409264 9.24240792 3.41110542 5.80650115
## [25] 10.70381619 3.60636454 0.98119361 2.67209406 0.38143136 2.40521504
## [31] 0.53702168 3.75374177 0.07795675 4.45483117 1.44021931 0.28772836
## [37] 2.21671242 1.93485423 1.36729211 1.01774308 1.72033923 3.69529721
## [43] 4.14231118 1.67223428 0.15737776 0.20630007 5.52833622 0.34193989
## [49] 2.48012912 0.75247275 5.56125285 2.59193472 0.21090938 4.23009235
## [55] 7.76188966 3.90007633 -0.07825419 1.53604448 0.73062629 0.28627197
## [61] 3.74291244 0.93373793 6.37959798 4.52414664 0.32849030 1.14635463
## [67] 1.29131529 6.61890534 8.15047464 8.49811811 8.97091813 9.94053880
## [73] 9.44949986 10.98174653 7.13356426 7.41701734 1.54833193 4.07230851
## [79] 3.41619370 8.48249316 6.66760853 3.04373022 9.86407506 8.56286582
## [85] 4.32520887 9.76421082 10.03180647 9.03721092 3.70543300 9.18394511
## [91] 9.92391786 10.50703138 7.14183985 1.28389723 6.33318031 5.72306781
## [97] 7.30087096 0.13124950 1.20092477 4.16558620 9.61028738 7.67576890
## [103] 7.59496295 5.90212016 9.08590266 6.04905897 8.83996841 7.83990441
## [109] 9.65408987 10.10170546 10.31310431 7.13424894 3.22173587 5.56185216
## [115] 7.50816824 9.86266468 4.89610980 0.11336880 1.39687752 3.40063272
## [121] 4.87255758 8.66962106 9.52590567 8.83271541 3.66755934 5.03999524
## [127] 8.01288975 5.66111562 9.61699600 9.32391609 10.27015221 9.96814903
## [133] 9.50157897 9.75293994 9.37719921 6.86038459 6.73699676 6.72963430
## [139] 7.88250607 9.51965406 8.71589259 9.93849275 9.90901897 9.91558375
## [145] 7.42776604 6.51108973 8.77100750 6.81059617 10.06493288 10.49397618
## [151] 10.30962305 9.15275116 9.26531025 7.37018172 8.25292179 8.36140983
## [157] 9.07288604 8.80222833 8.78612455 9.27743393 9.95701271 6.09990897
## [163] 8.97363706 9.48858286 6.28684357 8.40138349 7.38485201 6.70157700
## [169] 9.08947709 9.77468308 8.76095370 7.98434983 8.46622506 6.65901306
## [175] 5.00749157 7.03847412 7.82785395 5.56887581 3.16221413 9.02821274
## [181] 9.73336958 10.46578828 6.65380168 4.59386522 8.36465402 9.71717569
## [187] 9.81209472 10.31912883 8.37033054 8.98482853 9.39074916 7.00793845
## [193] 10.04340692 10.40321950 7.51979368 9.35825878 8.31884070 8.51874409
## [199] 3.87039200 10.42642447 10.41549548 10.04726883 8.53360398 9.51449499
## [205] 8.45862032 6.84052568 10.46719939 10.04538037 8.69820236 8.50833848
## [211] 7.67529125 10.03994207 9.03896451 7.00616409 9.86652912 9.45004608
## [217] 9.12688892 9.88147530 10.36568410 10.51295681 6.73076163 6.35183982
## [223] 7.32452934 9.06882290 7.66328465 10.15378781 10.28542064 9.90383257
## [229] 9.83920510 9.35101802 9.39745443 6.29537190 8.93342087 8.06616460
## [235] 9.39395129 9.40282684 9.06737979 9.44152433 10.24252320 9.46122369
## [241] 8.41563522 9.33220222 10.10603023 9.41980487 10.47651291 9.95991830
## [247] 9.21215152 4.92074148 3.50477203 0.01497428 9.12472357 9.08954618
## [253] 7.94823180 10.15390649 10.01104322 8.86795144 10.75451387 11.01623044
## [259] 10.47037204 6.49787918 5.73297800 -0.17383864 7.93356817 9.05545292
## [265] 10.36697738 7.03342520 3.98239779 0.70491280 1.54087460 5.25489023
## [271] 0.18712705 2.38346030 7.52255206 7.39577177 5.62080813 10.16413562

```

```
## [277] 9.02223949 11.30647194 8.81929614 10.14960963 10.05633799 10.02817957
## [283] 9.46672028 10.67594128 7.61870809 2.33816206 -0.14322615 5.99203987
## [289] 0.19058306 3.28460358 8.13151398 1.44710799 3.77716617 8.87772468
## [295] 10.48289162 2.68521761 4.68611335 7.09412047 6.19895081 6.09308102
## [301] 8.70913498 11.85685826 2.27768644 6.72837917 0.35594482 1.58286295
## [307] 0.03345685 3.44852019 0.43307681 0.64789908 0.93504367 1.20648613
## [313] -0.01266910 5.11739394 6.61735866 0.50992101 7.66705580 2.21216257
## [319] 4.42960878 0.11544199 6.60639255 8.92100750 3.82313363 1.02554593
## [325] 8.51706070 10.40291579 4.95079409 7.91251366 4.06338538 3.46886581
## [331] 1.96149651 0.01682332 2.46924185 3.33296647 2.44961566 5.22646603
## [337] 8.19301484 8.77287654 0.95197678 7.38713104 10.74919683 8.51524732
## [343] 2.99462140 4.98927685 0.88014912 2.07615109 4.13163448 2.71136022
## [349] 2.24779638 3.60486122 3.70745035 0.20424064 1.11668432 0.78727216
## [355] 0.22948294 2.43078136 4.69146007 3.46890950 3.56860367 4.31694335
## [361] 3.03367449 0.30778199 3.51145603 2.18611115 5.57723917 0.51138680
```

Importante quando se trabalha com datas e fatores, retorna um valor com a mesma forma:

```
a$n<- ifelse((n.est)<0,1, n.est);
head(a)
```

```
##           Data      Rs Tmax Tmin RHmax      u2 RHmin DiaJuliano      N      Qo
## 1 1980-01-01 18.51 31.1 21.8 82.00 2.000000 82.00          1 12.42742 38.08036
## 2 1980-01-02  9.65 26.5 21.8 91.50 1.000000 91.50          2 12.42559 36.92744
## 3 1980-01-03 13.70 29.9 21.0 86.00 1.000000 86.00          3 12.42363 36.20749
## 4 1980-01-04 13.86 30.3 22.0 82.75 1.666667 82.75          4 12.42153 36.57674
## 5 1980-01-05  9.66 25.1 21.6 90.97 1.070000 90.97          5 12.41929 37.71584
## 6 1980-01-06 16.10 31.2 20.8 80.67 1.380000 80.67          6 12.41691 38.61551
##           n
## 1 5.8676646
## 2 0.2813970
## 3 3.1897598
## 4 3.2029975
## 5 0.1521569
## 6 4.1455326
```

Nebulosidade ou fração de luz (h):

```
a$n.N = a$n/a$N;
head(a)
```

```
##           Data      Rs Tmax Tmin RHmax      u2 RHmin DiaJuliano      N      Qo
## 1 1980-01-01 18.51 31.1 21.8 82.00 2.000000 82.00          1 12.42742 38.08036
```

```

## 2 1980-01-02 9.65 26.5 21.8 91.50 1.000000 91.50
## 3 1980-01-03 13.70 29.9 21.0 86.00 1.000000 86.00
## 4 1980-01-04 13.86 30.3 22.0 82.75 1.666667 82.75
## 5 1980-01-05 9.66 25.1 21.6 90.97 1.070000 90.97
## 6 1980-01-06 16.10 31.2 20.8 80.67 1.380000 80.67
##           n      n.N
## 1 5.8676646 0.47215460
## 2 0.2813970 0.02264656
## 3 3.1897598 0.25674945
## 4 3.2029975 0.25785862
## 5 0.1521569 0.01225166
## 6 4.1455326 0.33386175

```

Albedo, para solo gramado=0,23:

```
r = 0.23
```

Temperatura média do ar a 2 metros acima da superfície do solo (°C):

```

a$Tmed = (a$Tmax+a$Tmin)/2
head(a)

```

```

##           Data     Rs Tmax Tmin RHmax      u2 RHmin DiaJuliano      N      Qo
## 1 1980-01-01 18.51 31.1 21.8 82.00 2.000000 82.00
## 2 1980-01-02 9.65 26.5 21.8 91.50 1.000000 91.50
## 3 1980-01-03 13.70 29.9 21.0 86.00 1.000000 86.00
## 4 1980-01-04 13.86 30.3 22.0 82.75 1.666667 82.75
## 5 1980-01-05 9.66 25.1 21.6 90.97 1.070000 90.97
## 6 1980-01-06 16.10 31.2 20.8 80.67 1.380000 80.67
##           n      n.N Tmed
## 1 5.8676646 0.47215460 26.45
## 2 0.2813970 0.02264656 24.15
## 3 3.1897598 0.25674945 25.45
## 4 3.2029975 0.25785862 26.15
## 5 0.1521569 0.01225166 23.35
## 6 4.1455326 0.33386175 26.00

```

Pressão de saturação de vapor máxima (kPa):

```
a$esMax = 0.6100*exp((17.3*a$Tmax)/(237.3+a$Tmax))
```

Pressão de saturação de vapor mínima (kPa):

$$a\$esMin = 0.6100 * \exp((17.3 * a\$Tmin) / (237.3 + a\$Tmin))$$

Pressão de saturação de vapor média (kPa):

$$a\$esMed = (a\$esMax + a\$esMin) / 2$$

Pressão atual de vapor (kPa) :

$$a\$ea = (a\$esMed * a\$RHmax) / 100$$

Déficit de vapor de pressão de saturação:

$$a\$DPV = a\$esMed - a\$ea$$

É a declividade da curva de pressão de vapor em relação a temperatura (kPa/°C):

$$a\$Delta = (4098 * a\$esMed) / (a\$Tmed + 237.3)^{**2}$$

Balanço de ondas curtas:

$$a\$BOC = a\$Rs * (1 - r)$$

Balanço de ondas longas:

$$a\$BOL = -(0.903 * (10^{**-9}) * ((a\$Tmed + 273)^{**4}) * (0.34 - 0.14 * (\sqrt{a\$ea})) * (0.1 + (0.9 * a\$n.N)))$$

Saldo de radiação é superfície da cultura:

$$a\$Rn = (a\$BOC + a\$BOL)$$

Estimativa de ETo pelo método de Penman-Monteith-FAO

$$a\$ETP.Penman = ((0.408 * a\$Delta * a\$Rn) + (gama * (900 / (a\$Tmed + 273)) * a\$u2 * a\$DPV)) / (a\$Delta + (gama * (1 - a\$Rn)))$$

##	Data	Rs	Tmax	Tmin	RHmax	u2	RHmin	DiaJuliano	N	Qo
## 1	1980-01-01	18.51	31.1	21.8	82.00	2.000000	82.00	1	12.42742	38.08036
## 2	1980-01-02	9.65	26.5	21.8	91.50	1.000000	91.50	2	12.42559	36.92744
## 3	1980-01-03	13.70	29.9	21.0	86.00	1.000000	86.00	3	12.42363	36.20749
## 4	1980-01-04	13.86	30.3	22.0	82.75	1.666667	82.75	4	12.42153	36.57674

```

## 5 1980-01-05 9.66 25.1 21.6 90.97 1.070000 90.97      5 12.41929 37.71584
## 6 1980-01-06 16.10 31.2 20.8 80.67 1.380000 80.67      6 12.41691 38.61551
##           n       n.N Tmed    esMax    esMin    esMed     ea      DPV
## 1 5.8676646 0.47215460 26.45 4.528027 2.615043 3.571535 2.928659 0.6428764
## 2 0.2813970 0.02264656 24.15 3.467983 2.615043 3.041513 2.782985 0.2585286
## 3 3.1897598 0.25674945 25.45 4.227431 2.489813 3.358622 2.888415 0.4702071
## 4 3.2029975 0.25785862 26.15 4.325640 2.647198 3.486419 2.885012 0.6014073
## 5 0.1521569 0.01225166 23.35 3.191668 2.583231 2.887449 2.626713 0.2607367
## 6 4.1455326 0.33386175 26.00 4.553895 2.459337 3.506616 2.828787 0.6778290
##           Delta      BOC      BOL      Rn ETP.Penman
## 1 0.2103982 14.2527 -0.38272413 13.869976 4.507272
## 2 0.1823410  7.4305 -0.09021728  7.340283 2.212778
## 3 0.1993644 10.5490 -0.24209055 10.306909 3.244769
## 4 0.2058520 10.6722 -0.24544334 10.426757 3.480430
## 5 0.1741690  7.4382 -0.08745750  7.350742 2.194204
## 6 0.2072805 12.3970 -0.30213854 12.094861 3.979211

```

Este método, além de procurar representar, de maneira consistente, o fenômeno biofísico da evapotranspiração, é alimentado por quase todos os elementos meteorológicos observados em estações meteorológicas de superfície.

Método de Priestley-Taylor

O método de Priestley-Taylor é uma simplificação das equações de Penman e de Penman-Monteith. Apresenta a vantagem de se exigir menos dados:

```

a$W = ifelse(a$Tmed<16,(0.407+0.0145*a$Tmed),(0.483+0.01*a$Tmed))

a$ETP.Priestley = (1.26*a$W*a$Rn)/2.45

head (a)

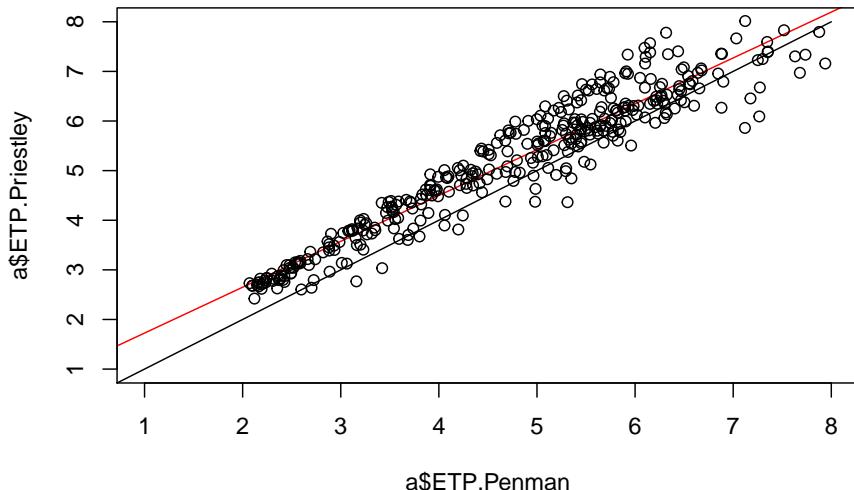
##           Data     Rs Tmax Tmin RHmax      u2 RHmin DiaJuliano      N      Qo
## 1 1980-01-01 18.51 31.1 21.8 82.00 2.000000 82.00      1 12.42742 38.08036
## 2 1980-01-02  9.65 26.5 21.8 91.50 1.000000 91.50      2 12.42559 36.92744
## 3 1980-01-03 13.70 29.9 21.0 86.00 1.000000 86.00      3 12.42363 36.20749
## 4 1980-01-04 13.86 30.3 22.0 82.75 1.666667 82.75      4 12.42153 36.57674
## 5 1980-01-05  9.66 25.1 21.6 90.97 1.070000 90.97      5 12.41929 37.71584
## 6 1980-01-06 16.10 31.2 20.8 80.67 1.380000 80.67      6 12.41691 38.61551
##           n       n.N Tmed    esMax    esMin    esMed     ea      DPV
## 1 5.8676646 0.47215460 26.45 4.528027 2.615043 3.571535 2.928659 0.6428764
## 2 0.2813970 0.02264656 24.15 3.467983 2.615043 3.041513 2.782985 0.2585286
## 3 3.1897598 0.25674945 25.45 4.227431 2.489813 3.358622 2.888415 0.4702071
## 4 3.2029975 0.25785862 26.15 4.325640 2.647198 3.486419 2.885012 0.6014073
## 5 0.1521569 0.01225166 23.35 3.191668 2.583231 2.887449 2.626713 0.2607367

```

```
## 6 4.1455326 0.33386175 26.00 4.553895 2.459337 3.506616 2.828787 0.6778290
##       Delta      BOC      BOL      Rn ETP.Penman      W ETP.Priestley
## 1 0.2103982 14.2527 -0.38272413 13.869976 4.507272 0.7475 5.332015
## 2 0.1823410  7.4305 -0.09021728  7.340283 2.212778 0.7245 2.734989
## 3 0.1993644 10.5490 -0.24209055 10.306909 3.244769 0.7375 3.909264
## 4 0.2058520 10.6722 -0.24544334 10.426757 3.480430 0.7445 3.992256
## 5 0.1741690  7.4382 -0.08745750  7.350742 2.194204 0.7165 2.708644
## 6 0.2072805 12.3970 -0.30213854 12.094861 3.979211 0.7430 4.621619
```

Gráficos correlacionando método de Penman-Monteith e Método de Priestley-Taylor:

```
plot(a$ETP.Penman,a$ETP.Priestley,
      xlim=c(1,8),
      ylim=c(1,8),
      abline(lm(a$ETP.Priestley~a$ETP.Penman),col="red"))
lines(c(0, 8), c(0, 8), col = "black")
```



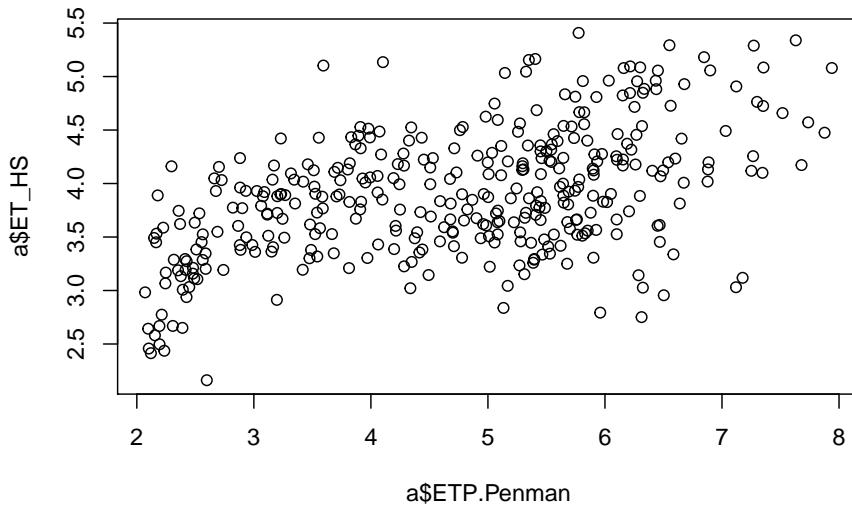
Salvar os dados:

```
write.table(a, "ETP.txt")
```

Método Hargreaves-Samani (1985)

```
a$ET_HS = (0.408*(0.0023*a$Qo*((a$Tmax-a$Tmin)**0.5)*a$Qo))
```

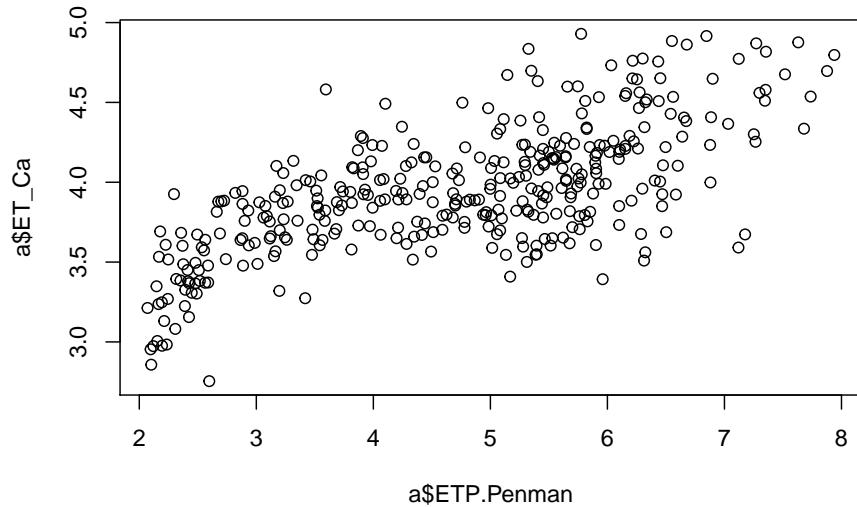
```
plot(a$ETP.Penman,a$ET_HS)
```



Método Carmago

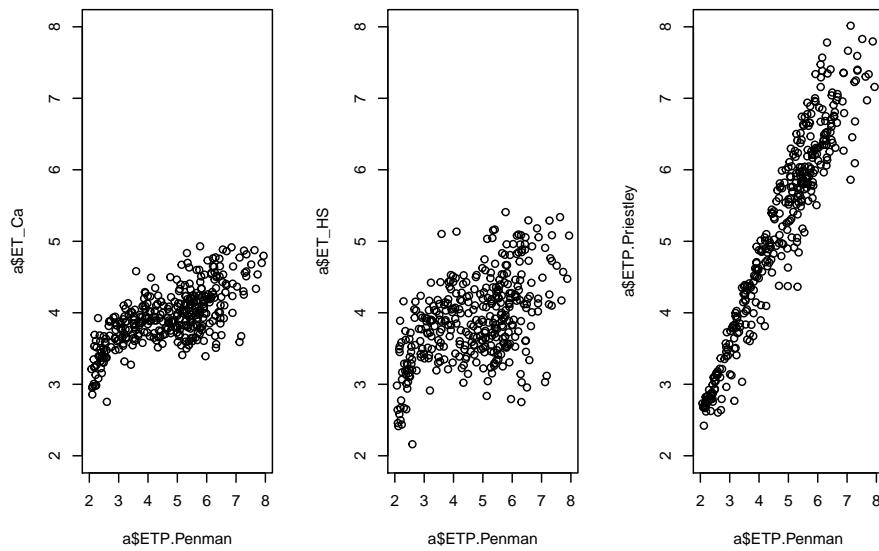
```
a$ET_Ca = 0.01*(a$Qo/2.45)*((1.08*a$Tmax-(0.36*a$Tmin)))*1
```

```
plot(a$ETP.Penman,a$ET_Ca)
```



Gráficos correlacionando Penman-Monteith com método Camargo, Hargreaves-Samani e Priestley-Taylor:

```
par(mfrow=c(1,3))
fplot(a$ETP.Penman,a$ET_Ca,
      xlim=c(2,8),
      ylim=c(2,8))
plot(a$ETP.Penman,a$ET_HS,
      xlim=c(2,8),
      ylim=c(2,8))
plot(a$ETP.Penman,a$ETP.Priestley,
      xlim=c(2,8),
      ylim=c(2,8))}
```



```
dev.off() #Encerra todos os dispositivos gráficos abertos
```

```
## null device
##          1
```

Comparação numérica e gráfica de séries temporais simuladas e observadas, focadas principalmente na modelagem hidrológica:

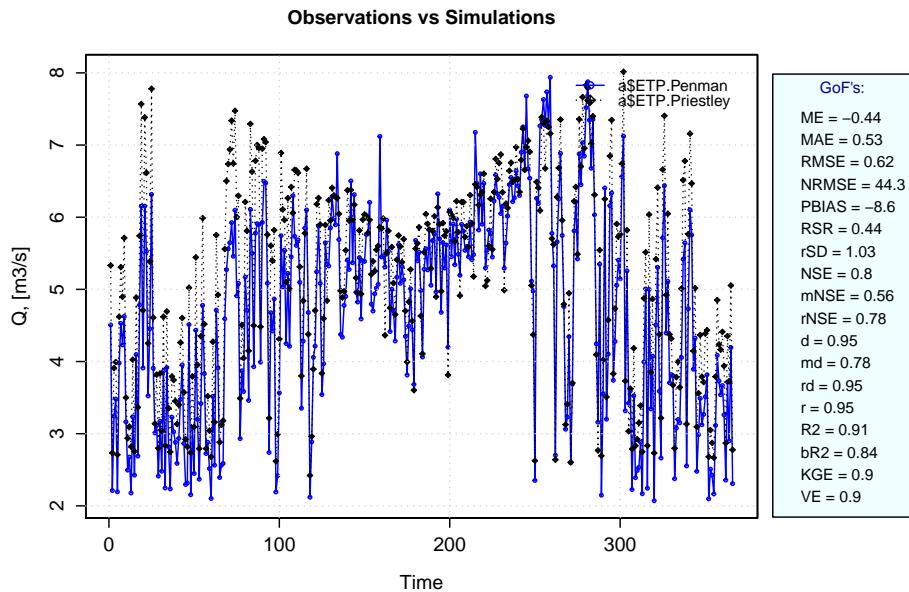
```
hydroGOF::br2(sim=a$ETP.Penman, obs=a$ETP.Priestley)
```

```
## [1] 0.8372423
```

Comparação gráfica entre dois vetores:

```
hydroGOF::ggoft(sim=a$ETP.Penman, obs=a$ETP.Priestley)
```

```
## [ Note: You did not provide dates, so only a numeric index will be used in the time axis ]
```



10.2.1 Pacote Evapotranspiration

Apagar objetos antigos:

```
rm(list=ls())
```

Baixar Pacote Evapotranspiration:

```
require(Evapotranspiration)
```

Baixar dados:

```
df <- read.table("https://www.dropbox.com/s/qixbu9tqnnfsuyt/a.txt?dl=1", header = T, sep = "")
```

Transformar em Data:

```
df$data <- as.Date(df$data)
climate <- lapply(as.list(df)[2:7], zoo, df$data)
J       <- as.numeric(format(df$data, "%j"))
data    <- c(list(Date.daily=df$data, J=J), climate)
```

Atribuindo nomes:

```
names(data)
```

```
## [1] "Date.daily" "J"           "Rs"          "Tmax"        "Tmin"
## [6] "RHmax"      "u2"         "RHmin"
```

Ver a estrutura do objeto com todas as variáveis da tabela de dados:

```
str(data)
```

```
## List of 8
## $ Date.daily: Date[1:366], format: "1980-01-01" "1980-01-02" ...
## $ J      : num [1:366] 1 2 3 4 5 6 7 8 9 10 ...
## $ Rs     : 'zoo' series from 1980-01-01 to 1980-12-31
##   Data: num [1:366] 18.51 9.65 13.7 13.86 9.66 ...
##   Index: Date[1:366], format: "1980-01-01" "1980-01-02" ...
## $ Tmax   : 'zoo' series from 1980-01-01 to 1980-12-31
##   Data: num [1:366] 31.1 26.5 29.9 30.3 25.1 31.2 31.6 31.5 31.3 29.6 ...
##   Index: Date[1:366], format: "1980-01-01" "1980-01-02" ...
## $ Tmin   : 'zoo' series from 1980-01-01 to 1980-12-31
##   Data: num [1:366] 21.8 21.8 21 22 21.6 20.8 22.3 21.4 23 22.1 ...
##   Index: Date[1:366], format: "1980-01-01" "1980-01-02" ...
## $ RHmax  : 'zoo' series from 1980-01-01 to 1980-12-31
##   Data: num [1:366] 82 91.5 86 82.8 91 ...
##   Index: Date[1:366], format: "1980-01-01" "1980-01-02" ...
## $ u2     : 'zoo' series from 1980-01-01 to 1980-12-31
##   Data: num [1:366] 2 1 1 1.67 1.07 ...
##   Index: Date[1:366], format: "1980-01-01" "1980-01-02" ...
## $ RHmin  : 'zoo' series from 1980-01-01 to 1980-12-31
##   Data: num [1:366] 82 91.5 86 82.8 91 ...
##   Index: Date[1:366], format: "1980-01-01" "1980-01-02" ...
```

Converter latitude para radianos:

```
pi/180*-23.45 # [1] -0.4092797
```

```
## [1] -0.4092797
```

Editar adequadamente as constantes a serem utilizadas. As constantes a serem utilizadas variam com o modelo escolhido, Ex:ET.PenmanMonteith e ET.PristleyTaylor.

Constantes dos modelos

Este conjunto de dados contém os dados climáticos brutos, incluindo as variáveis necessárias para o cálculo da evapotranspiração:

```
myConst <- list(lambda = 2.45, sigma = 4.903e-09, Gsc = 0.082,
                  lat = -23.45, lat_rad = -0.40928, as = 0.25,
                  bs = 0.55, Elev = 480, z = 2, Roua = 1.2,
                  Ca = 0.001013, G = 0,
                  alphaA = 0.14, alphaPT = 1.26,
                  ap = 2.4, fz = 28, b0 = 1,
                  a_0 = 11.9, b_0 = -0.15,
                  c_0 = -0.25, d_0 = -0.0107,
                  e0 = 0.81917,
                  e1 = -0.0040922, e2 = 1.0705, e3 = 0.065649,
                  e4 = -0.0059684,
                  e5 = -0.0005967, gammaps = 0.66,
                  epsilonMo = 0.92, PA = 285.8,
                  alphaMo = 17.27, betaMo = 237.3,
                  sigmaMo = 5.67e-08, lambdaMo = 28.5,
                  b1 = 14, b2 = 1.2)
```

Métodos de estimativa de ETO

O método Penman-Monteith (FAO) é considerado, internacionalmente, o mais apropriado para a estimativa da ETo.

Penman-Monteith

```
res1 <- ET.PenmanMonteith(data, myConst, ts="daily", solar="data", wind="yes", crop =
## Penman-Monteith FA056 Reference Crop ET

## Evaporative surface: FAO-56 hypothetical short grass, albedo = 0.23 ; surface resist

## Solar radiation data have been used directly for calculating evapotranspiration

## Wind data have been used for calculating the reference crop evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats
```

```
## Mean: 3.91
```

```
## Max: 6.72
```

```
## Min: 1.93
```

Constantes: Altitude, lambda-calor latente da vaporização, latitude em radianos, Gsc-constante solar, sigma-constante de Stefan-Boltzmann, fluxo de calor no solo.

- ts: Dados diários
- solar: data, indica que os dados da radiação solar devem ser utilizados diretamente para calcular a evapotranspiração
- wind: yes, indica que o cálculo utilizará dados reais da velocidade do vento
- crop: short, indica que o método para grama curta hipotética FAO-56 será aplicado

Priestley Taylor:

```
res2 <- ET.PriestleyTaylor(data, myConst, ts="daily", solar="data", alpha=.23)

## Priestley-Taylor Potential ET

## Evaporative surface: user-defined, albedo = 0.23

## Solar radiation data have been used directly for calculating evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 4.15

## Max: 7

## Min: 2.19
```

```
res3 <- ET.Romanenko(data, myConst= NULL, ts="daily")
```

```
## Romanenko Actual ET  
## Timestep: daily  
## Units: mm  
## Time duration: 1980-01-01 to 1980-12-31  
## 366 ET estimates obtained  
## Basic stats  
## Mean: 5.57  
## Max: 12.31  
## Min: 0.8
```

```
res4 <- ET.HargreavesSamani(data, myConst, ts="daily")
```

```
## Hargreaves-Samani Reference Crop ET  
## Evaporative surface: reference crop  
## Timestep: daily  
## Units: mm  
## Time duration: 1980-01-01 to 1980-12-31  
## 366 ET estimates obtained  
## Basic stats  
## Mean: 4.45  
## Max: 7.44  
## Min: 2.74
```

```
res5 <- ET.Abtew(data, myConst, ts="daily", solar="data")

## Abtew Actual ET

## Solar radiation data have been used directly for calculating evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 3.81

## Max: 5.76

## Min: 1.79

res6 <- ET.BrutsaertStrickler(data, myConst, ts="daily", solar="data", alpha=0.23)

## Brutsaert-Strickler Actual Areal ET

## Evaporative surface: user-defined, albedo = 0.23

## Solar radiation data have been used directly for calculating evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 3.99

## Max: 7.88

## Min: 1.1
```

```

res7 <- ET.ChapmanAustralian(data, myConst, ts="daily", PenPan= T,
                               solar="data", alpha=0.23)

## Chapman Potential ET

## Evaporative surface: user-defined, albedo = 0.23

## Solar radiation data have been used for calculating evapotranspiration

## PenPan formulation has been used to estimate Class-A pan evaporation for the calcula

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 2.64

## Max: 4.69

## Min: 1.17

res8 <- ET.GrangerGray(data, myConst, ts="daily",
                        solar="data", windfunction_ver=1948, alpha=0.23)

## Granger-Gray Actual Areal ET

## Evaporative surface: user-defined, albedo = 0.23

## Solar radiation data have been used directly for calculating evapotranspiration

## Wind data have been used for the calculation of the drying power of air, using Penma

## Timestep: daily

```

```
## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 2.86

## Max: 5.39

## Min: 1.45

res9 <- ET.JensenHaise(data, myConst, ts="daily", solar="data")

## Jensen-Haise Potential ET

## Solar radiation data have been used for calculating evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 5.48

## Max: 9.01

## Min: 2.42

res10 <- ET.Makkink(data, myConst, ts="daily", solar="data")

## Makkink Reference crop ET
```

```
## Solar radiation data have been used directly for calculating evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 3.3

## Max: 5.19

## Min: 1.45

res11 <- ET.MattShuttleworth(data, myConst, ts="daily",
                               solar="data", alpha=0.23, r_s=70, CH=0.12)

## Matt-Shuttleworth Reference Crop ET

## Evaporative surface: user-defined, albedo = 0.23 ; surface resistance = 70 sm^-1; cr

## Solar radiation data have been used directly for calculating evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 3.07

## Max: 5.18

## Min: 1.62
```

```
res12 <- ET.McGuinnessBordne(data, myConst, ts="daily")

## McGuinness-Bordne Potential ET

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 6.37

## Max: 8.69

## Min: 3.99

res13 <- ET.PenPan(data, myConst, ts="daily",
                     solar="data", alpha=0.23,
                     est="potential ET", pan_coeff=0.71, overest= FALSE)

## PenPan potential ET

## Evaporative surface: user-defined, albedo = 0.23

## Pan coefficient: 0.71

## Solar radiation data have been used directly for calculating evapotranspiration

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained
```

```
## Basic stats

## Mean: 6.18

## Max: 10.97

## Min: 2.73

res14 <- ET.Penman(data, myConst, ts="daily",
                     solar="data", wind="yes",
                     windfunction_ver = "1948", alpha = 0.08, z0 = 0.001)

## Penman Open-water Evaporation

## Evaporative surface: water, albedo = 0.08 ; roughness height = 0.001 m

## Solar radiation data have been used directly for calculating evapotranspiration

## Wind data have been used for calculating the Penman evaporation. Penman 1948 wind f

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## Basic stats

## Mean: 5.27

## Max: 8.69

## Min: 2.58

res16 <- ET.Turc(data, myConst, ts="daily", solar="data", humid= FALSE)

## Turc Reference Crop ET

## Evaporative surface: reference crop
```

```

## Solar radiation data have been used directly for calculating evapotranspiration

## No adjustment for non-humid conditions has been applied to calculated Turc reference crop evap

## Timestep: daily

## Units: mm

## Time duration: 1980-01-01 to 1980-12-31

## 366 ET estimates obtained

## Basic stats

## Mean: 3.99

## Max: 6

## Min: 2.05

```

Tabela de Resultados

```

res <- cbind(PM=res1$ET.Daily,
              PT=res2$ET.Daily, RM=res3$ET.Daily, HS=res4$ET.Daily,
              AB=res5$ET.Daily, BS=res6$ET.Daily, CA=res7$ET.Daily, GG=res8$ET.Daily,
              JH=res9$ET.Daily, MK=res10$ET.Daily, MS=res11$ET.Daily, MG=res12$ET.Daily,
              PP=res13$ET.Daily, PN=res14$ET.Daily, Tu=res16$ET.Daily)

head(res)

##          PM        PT        RM        HS        AB        BS        CA
## 1980-01-01 4.156318 4.929036 3.430645 5.082573 3.928653 5.277834 2.730982
## 1980-01-02 2.188232 2.745827 1.478422 5.141016 2.048163 3.129764 1.306937
## 1980-01-03 3.060989 3.741211 2.565564 4.972826 2.907755 4.165446 1.890737
## 1980-01-04 3.296640 3.807941 3.249473 5.083175 2.941714 4.000580 2.116993
## 1980-01-05 2.167625 2.712797 1.519894 4.957608 2.050286 3.071238 1.294748
## 1980-01-06 3.685204 4.314276 3.619968 5.056544 3.417143 4.597076 2.351236
##          GG        JH        MK        MS        MG        PP        PN
## 1980-01-01 3.670318 5.562444 3.389243 3.522360 8.084051 6.381423 5.599644
## 1980-01-02 2.164547 2.673444 1.655891 2.054957 7.489878 3.053828 2.885753
## 1980-01-03 2.891066 3.977194 2.444971 2.807769 7.820466 4.418005 4.066475
## 1980-01-04 2.795805 4.122643 2.497964 2.764389 7.996387 4.946702 4.379952

```

```
## 1980-01-05 2.137567 2.597357 1.638069 2.018321 7.273783 3.025346 2.872788
## 1980-01-06 3.209531 4.764286 2.915389 3.177262 7.949163 5.494064 4.914844
## Tu
## 1980-01-01 4.081560
## 1980-01-02 2.248908
## 1980-01-03 3.084847
## 1980-01-04 3.147341
## 1980-01-05 2.221660
## 1980-01-06 3.581712
```

Salvar Tabela com Evapotranspiração a partir de todos os métodos:

```
write.table(res, "ETo5.csv", sep = ";", dec = ".")
```

Plotar a evapotranspiração estimada com variáveis climáticas:

```
ETForcings(data, res1, forcing = "RHmin")
```

lm é usado para ajustar modelos lineares. Ele pode ser usado para realizar regressão e análise de covariância:

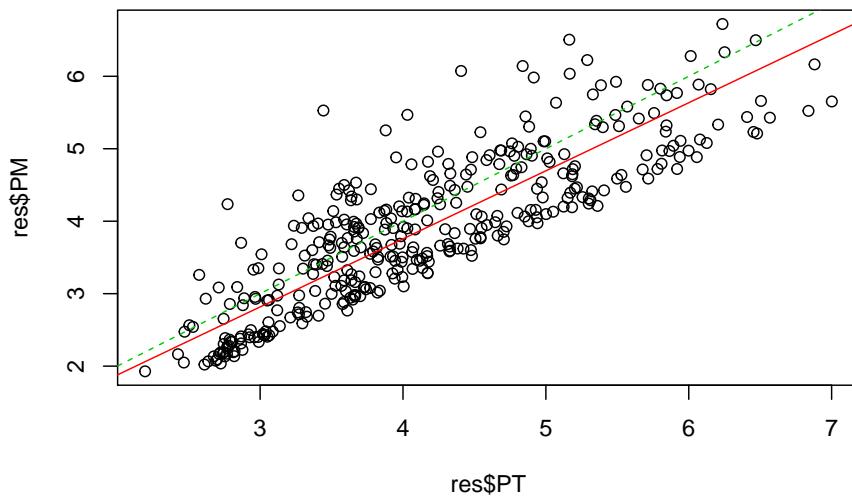
```
fit <- lm(res$PM ~ res$PT - 1)

summary(fit)

##
## Call:
## lm(formula = res$PM ~ res$PT - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -0.9230 -0.4409 -0.1374  0.4106  2.2943 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## res$PT    0.939128   0.006854    137   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5595 on 365 degrees of freedom
## Multiple R-squared:  0.9809, Adjusted R-squared:  0.9809 
## F-statistic: 1.878e+04 on 1 and 365 DF,  p-value: < 2.2e-16
```

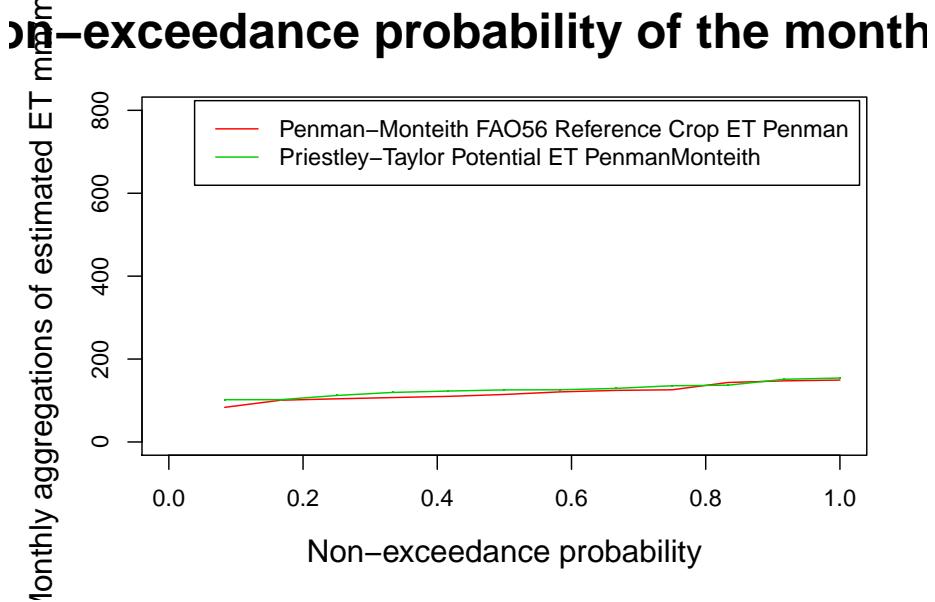
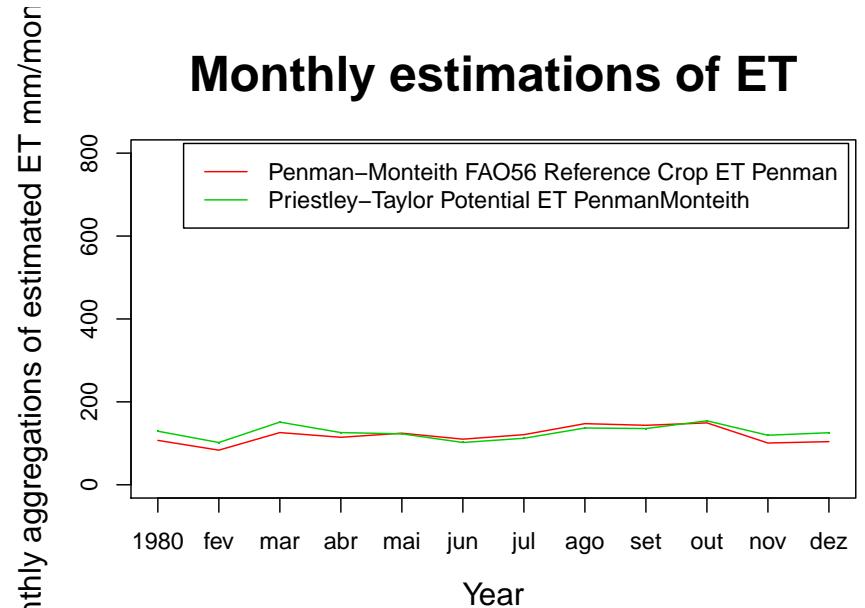
Apresenta similaridade entre os métodos, análise significativa:

```
plot(res$PM~res$PT);
abline(fit, col=2);
abline(a=0, b=1, col=3, lty=2)
```

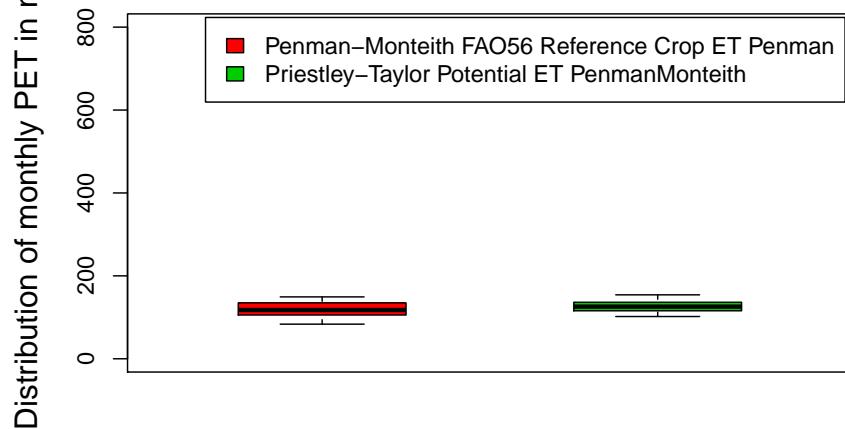


Comparar a evapotranspiração estimada entre vários conjuntos de resultados:

```
ETComparison(res1, res2, type = "Monthly", ylim=c(0,800),
             labs=c("Penman", "PenmanMonteith"))
```



Box plot of the monthly estimations of



10.2.2 Referência

<http://www.gis-blog.com/download-srtm-for-an-entire-country/>
<http://www.gis-blog.com/r-raster-data-acquisition/>
https://cmerow.github.io/RDataScience/05_Raster.html#1_setup
<https://ropensci.org/blog/2017/03/07/hddtools/>

10.3 Dados temporais no R

Vamos baixar os dados climáticos de Rondonópolis-MT:

```
roo <- read.csv2("https://www.dropbox.com/s/1ajoi1c8pla3yk6/roo.csv?dl=1")
```

Podemos verificar o cabeçario do conjunto de dados:

```
head(roo)
```

```
##   dd mm  ano Prec Tmax Tmin  n    Tbs    Tbu     UR Vvento
## 1  1  1 1998  NA 30.0 21.7 NA    NA    NA    NA    NA
## 2  1  2 1998  8.2 35.6 21.8 NA    NA    NA    NA    NA
```

```
## 3 2 1998 51.0 31.8 21.8 NA 25.075 23.85 89.75 0.125
## 4 3 2 1998 0.6 35.4 21.5 NA 24.975 23.00 86.50 0.125
## 5 4 2 1998 0.0 35.6 22.1 NA 26.650 23.90 80.00 0.275
## 6 5 2 1998 0.0 36.4 22.5 NA 26.950 24.00 78.50 0.325
```

Baixando os pacotes necessários:

```
library(dplyr)
```

Selecionando algumas colunas pelo nome com a função `select`:

```
selecionar <- select(roo,ano,Tmax)
head(selecionar)
```

```
##     ano Tmax
## 1 1998 30.0
## 2 1998 35.6
## 3 1998 31.8
## 4 1998 35.4
## 5 1998 35.6
## 6 1998 36.4
```

Retirar colunas com a função `select`:

```
dados <- select (roo,-UR)
head(dados)
```

```
##   dd mm ano Prec Tmax Tmin n    Tbs    Tbu Vvento
## 1 1  1 1998 NA 30.0 21.7 NA     NA     NA     NA
## 2 1  2 1998 8.2 35.6 21.8 NA     NA     NA     NA
## 3 2  2 1998 51.0 31.8 21.8 NA 25.075 23.85 0.125
## 4 3  2 1998 0.6 35.4 21.5 NA 24.975 23.00 0.125
## 5 4  2 1998 0.0 35.6 22.1 NA 26.650 23.90 0.275
## 6 5  2 1998 0.0 36.4 22.5 NA 26.950 24.00 0.325
```

Filtrar os dados com a função `filter` Apresentar somente os dias com temperatura mínima do ar menor que 20°C:

```
filter(roo, Tmin < 20)
```

```
##      dd mm ano  Prec Tmax Tmin   n    Tbs    Tbu      UR Vvento
## 1      1  5 1998 0.0 31.2 16.0  NA     NA     NA     NA     NA
## 2      2  5 1998 0.0 32.4 16.8  NA 24.300 20.550 71.25 0.20000
```

```

## 3   3 5 1998   0.0 32.8 18.0 NA 25.250 21.350 70.00 0.22500
## 4   4 5 1998 14.8 24.0 17.2 NA 21.300 20.100 88.50 0.25000
## 5   5 5 1998   1.0 26.8 17.2 NA 20.275 19.450 92.25 0.17500
## 6   6 5 1998   0.0 30.2 17.4 NA 22.900 21.000 85.00 0.05000
## 7   7 5 1998   0.0 30.6 16.6 NA 23.900 20.550 73.75 0.22500
## 8   8 5 1998   0.0 32.0 15.6 NA 25.000 21.300 72.00 0.25000
## 9   9 5 1998   0.0 34.4 14.8 NA 24.900 21.550 75.50 0.17500
## 10 10 5 1998   0.0 34.2 17.0 NA 26.150 22.050 70.25 0.22500
## 11 11 5 1998   0.0 33.4 17.4 NA 26.000 21.750 70.25 0.25000
## 12 12 5 1998   0.0 33.8 16.2 NA 25.550 21.200 68.50 0.07500
## 13 13 5 1998   0.0 33.6 17.6 NA 25.850 22.500 74.75 0.12500
## 14 14 5 1998   0.0 35.0 19.7 NA 25.900 22.200 73.00 0.22500
## 15 16 5 1998   1.0 23.4 16.6 NA 20.425 18.750 84.25 0.30000
## 16 17 5 1998   0.0 24.4 11.8 NA 18.875 15.800 72.00 0.30000
## 17 18 5 1998   0.0 27.0 11.6 NA 18.325 15.200 71.75 0.20000
## 18 19 5 1998   0.0 31.2 13.2 NA 21.075 17.350 69.00 0.12500
## 19 20 5 1998   0.0 32.0 15.6 NA 23.950 20.450 74.25 0.17500
## 20 21 5 1998   0.0 34.0 16.8 NA 24.950 21.450 74.75 0.20000
## 21 22 5 1998   0.0 34.4 15.6 NA 30.200 20.650 42.75 0.22500
## 22 23 5 1998   0.0 34.4 14.0 NA 25.225 20.500 67.00 0.25000
## 23 24 5 1998   0.0 34.4 16.6 NA 25.400 20.875 67.75 0.12500
## 24 25 5 1998   0.0 34.8 16.6 NA 25.750 21.500 70.75 0.17500
## 25 26 5 1998   0.0 33.8 19.5 NA 25.500 21.550 70.25 0.17500
## 26 28 5 1998   0.0 33.6 18.1 NA 24.650 21.250 77.00 0.10000
## 27 29 5 1998   0.0 30.2 19.5 NA 24.750 22.850 85.00 0.12500
## 28 30 5 1998   4.3 23.2 17.0 NA 21.300 19.900 86.50 0.52500
## 29 31 5 1998   0.0 24.0 16.6 NA 19.575 18.550 90.50 0.22500
## 30  1 6 1998   NA 23.8 18.3 NA   NA   NA   NA   NA
## 31  1 7 1998   0.0 35.4 15.6 NA   NA   NA   NA   NA
## 32  2 7 1998   0.0 35.2 15.6 NA 25.050 19.350 60.00 0.35000
## 33  3 7 1998   0.0 34.8 14.0 NA 24.175 19.400 65.00 0.15000
## 34  4 7 1998   0.0 35.6 13.8 NA 24.825 18.600 54.75 0.22500
## 35  5 7 1998   0.0 35.8 14.4 NA 25.500 19.100 54.75 0.30000
## 36  6 7 1998   0.0 35.8 15.2 NA 26.400 19.350 52.50 0.27500
## 37  7 7 1998   0.0 35.4 15.4 NA 26.100 18.800 49.75 0.35000
## 38  8 7 1998   0.0 36.0 14.2 NA 24.900 18.400 53.75 0.22500
## 39  9 7 1998   0.0 29.8 16.4 NA 23.250 19.200 66.50 0.32500
## 40 10 7 1998   0.0 29.8 16.8 NA 21.225 17.950 73.25 0.35000
## 41 11 7 1998   0.0 31.0 12.4 NA 21.625 16.550 58.75 0.52500
## 42 12 7 1998   0.0 33.8 12.8 NA 21.125 16.450 64.75 0.22500
## 43 13 7 1998   0.0 35.0 15.6 NA 24.875 19.200 59.50 0.15000
## 44 14 7 1998   NA 31.4 17.0 NA 24.650 19.700 61.50 0.42500
## 45 15 7 1998   NA 34.0 15.4 NA 24.625 20.300 68.50 0.15000
## 46 16 7 1998   0.0 35.8 17.6 NA 25.600 20.850 67.75 0.27500
## 47 17 7 1998   0.0 35.6 15.6 NA 26.825 19.950 53.75 0.30000
## 48 18 7 1998   0.0 36.6 15.0 NA 26.950 18.850 44.75 0.42500

```

```

## 49   19  7 1998  0.0 31.8 15.0  NA 28.200 20.100  46.75 0.30000
## 50   20  7 1998  0.0 36.8 16.0  NA 27.500 19.000  44.75 0.10000
## 51   21  7 1998  0.0 37.4 15.8  NA 27.550 19.025  45.25 0.22500
## 52   22  7 1998  0.0 37.0 16.6  NA 26.100 19.550  56.25 0.20000
## 53   23  7 1998  0.0 37.2 19.7  NA 27.400 20.700  56.25 0.25000
## 54   24  7 1998  0.0 36.2 14.4  NA 26.850 18.550  44.75 0.20000
## 55   25  7 1998  0.0 36.4 14.2  NA 26.650 18.550  45.25 0.27500
## 56   26  7 1998  0.0 33.4 17.0  NA 25.525 19.350  56.50 0.37500
## 57   27  7 1998  0.0 35.4 14.8  NA 24.125 19.300  68.25 0.22500
## 58   28  7 1998  0.0 35.6 13.8  NA 25.875 18.300  47.75 0.25000
## 59   29  7 1998  0.0 36.4 14.4  NA 25.850 18.200  47.50 0.05000
## 60   30  7 1998  0.0 36.4 14.4  NA 26.350 18.300  45.75 0.30000
## 61   31  7 1998  0.0 36.8 13.8  NA 31.125 18.050  28.75 0.05000
## 62    1  8 1998  0.0 36.6 18.9  NA 26.800 18.850  45.25 0.30000
## 63    2  8 1998  0.0 33.0 17.0  NA 25.025 20.300  65.25 0.30000
## 64    3  8 1998  0.0 36.4 14.8  NA 25.850 19.800  59.00 0.12500
## 65    4  8 1998  0.0 38.6 19.3  NA 29.250 19.750  40.50 0.42500
## 66    5  8 1998  11.0 27.4 16.0  NA 22.550 18.300  68.75 0.27500
## 67    6  8 1998  0.0 28.2 15.2  NA 19.825 18.900  91.50 0.12500
## 68    7  8 1998  0.0 33.4 17.2  NA 25.050 21.900  77.50 0.20000
## 69    8  8 1998  0.0 36.8 19.7  NA 28.200 22.450  62.00 0.25000
## 70   15  8 1998  0.0 36.4 19.9  NA 26.800 23.600  79.00 0.20000
## 71   16  8 1998  0.0 38.8 19.9  NA 29.750 22.950  58.00 0.25000
## 72   17  8 1998  0.0 39.2 18.1  NA 28.800 21.250  53.50 0.20000
## 73   18  8 1998  0.0 37.4 19.5  NA 28.100 20.350  48.50 0.17500
## 74   19  8 1998  0.0 39.2 19.9  NA 28.550 21.350  53.50 0.12500
## 75   20  8 1998  0.0 38.4 19.3  NA 28.450 20.300  48.50 0.15000
## 76   21  8 1998  0.0 39.4 15.6  NA 29.400 21.100  49.75 0.22500
## 77   22  8 1998  0.0 37.2 19.9  NA 28.450 20.450  50.00 0.35000
## 78   23  8 1998  0.0 37.6 16.4  NA 26.850 20.800  63.25 0.25000
## 79   24  8 1998  NA 37.6 15.4  NA 26.975 17.700  41.00 0.20000
## 80   25  8 1998  0.0 37.4 15.6  NA 27.625 17.350  35.50 0.25000
## 81   26  8 1998  0.0 27.0 18.0  NA 23.075 18.700  65.50 0.27500
## 82   27  8 1998  0.0 26.4 16.0  NA 19.825 17.200  76.75 0.45000
## 83   28  8 1998  0.0 34.2 13.8  NA 22.325 18.250  69.25 0.25000
## 84   29  8 1998  0.0 34.0 15.6  NA 24.650 19.400  59.75 0.30000
## 85   30  8 1998  0.0 35.6 15.6  NA 26.100 19.350  52.50 0.37500
## 86   31  8 1998  NA 38.2 16.6  NA 26.900 19.400  49.75 0.32500
## 87    1  9 1998  0.0 38.6 19.9  NA 29.700 20.650  43.25 0.40000
## 88    2  9 1998  0.0 38.4 19.3  NA 28.950 20.150  44.25 0.27500
## 89    3  9 1998  0.0 38.6 17.2  NA 28.400 19.600  43.50 0.30000
## 90    4  9 1998  0.0 37.2 19.3  NA 29.150 20.050  40.50 0.27500
## 91    9  9 1998  6.0 31.8 18.9  NA 23.025 20.800  83.75 0.32500
## 92   11  9 1998  0.0 39.4 18.3  NA 27.550 21.650  62.25 0.22500
## 93   19  9 1998  1.6 29.4 14.2  NA 24.675 19.500  60.50 0.35000
## 94   20  9 1998  0.0 19.0 13.6  NA 15.400 14.250  87.25 0.57500

```

```

## 95 21 9 1998 0.0 25.0 14.4 NA 18.025 15.400 75.00 0.45000
## 96 22 9 1998 0.0 34.6 13.6 NA 22.900 17.050 55.75 0.22500
## 97 23 9 1998 0.0 38.0 15.4 NA 26.950 19.050 47.00 0.27500
## 98 24 9 1998 0.0 38.8 18.9 NA 30.350 21.300 43.00 0.32500
## 99 27 9 1998 7.0 35.4 19.4 NA 24.400 22.500 84.75 0.15000
## 100 28 9 1998 31.0 27.4 19.0 NA 24.100 22.800 89.00 0.37500
## 101 29 9 1998 5.0 28.0 19.1 NA 21.750 21.300 95.00 0.22500
## 102 1 10 1998 NA 36.2 18.9 NA 26.050 22.900 78.75 0.35000
## 103 2 10 1998 0.0 37.4 19.7 NA 27.500 22.200 63.50 0.27500
## 104 12 10 1998 37.6 29.4 19.1 NA 23.125 21.500 86.25 0.35000
## 105 13 10 1998 6.8 32.4 19.3 NA 23.450 22.250 90.00 0.20000
## 106 18 10 1998 0.0 34.6 17.2 NA 25.750 20.800 63.75 0.27500
## 107 19 10 1998 0.0 35.6 18.5 NA 27.250 21.900 62.25 0.22500
## 108 20 10 1998 0.0 37.2 19.3 NA 28.550 21.900 55.00 0.30000
## 109 28 10 1998 75.0 29.0 19.7 NA 25.500 23.900 86.50 0.17500
## 110 13 12 1998 10.0 33.4 19.8 NA 26.450 24.300 83.75 0.27500
## 111 24 12 1998 12.8 32.0 19.6 NA 24.900 23.400 88.25 0.27500
## 112 25 12 1998 0.0 34.0 19.7 NA 24.700 23.300 89.25 0.15000
## 113 30 12 1998 17.5 32.6 19.5 NA 23.900 23.100 93.00 0.10000
## 114 31 12 1998 0.0 33.6 18.4 NA 25.300 23.900 88.25 0.12500
## 115 1 1 1999 12.0 33.8 17.9 NA 26.150 24.700 89.75 2.50000
## 116 2 1 1999 0.0 31.8 17.4 NA 26.800 24.600 84.50 0.20000
## 117 3 1 1999 16.7 28.4 18.7 NA 24.000 23.700 97.50 0.10000
## 118 4 1 1999 16.4 32.4 18.7 NA 24.350 23.300 91.00 0.10000
## 119 15 1 1999 50.0 25.8 18.1 NA 24.050 23.850 97.00 0.10000
## 120 16 1 1999 103.0 28.4 16.4 NA 23.450 23.750 100.00 0.00000
## 121 31 1 1999 105.4 32.0 19.0 NA 24.900 24.200 94.00 0.22500
## 122 2 4 1999 0.0 35.2 18.5 NA 27.400 23.600 72.75 0.10000
## 123 7 4 1999 0.0 35.8 19.3 NA 26.350 23.100 77.00 0.05000
## 124 13 4 1999 7.5 32.8 19.7 NA 24.200 22.900 89.25 0.00000
## 125 14 4 1999 NA 34.4 19.9 NA 24.700 23.200 88.75 0.05000
## 126 15 4 1999 0.0 34.2 19.8 NA 25.000 23.300 86.00 0.30000
## 127 16 4 1999 13.0 32.6 18.0 NA 24.150 23.200 92.00 0.22500
## 128 17 4 1999 0.0 23.4 12.8 NA 18.525 16.100 78.25 0.37500
## 129 18 4 1999 0.0 26.2 10.1 NA 17.175 13.350 64.75 0.40000
## 130 19 4 1999 0.0 30.6 10.4 NA 19.375 15.150 64.00 0.35000
## 131 20 4 1999 0.0 32.4 14.2 NA 22.425 18.050 65.00 0.22500
## 132 21 4 1999 0.0 35.8 13.2 NA 25.050 21.150 71.50 0.15000
## 133 22 4 1999 0.0 35.6 19.3 NA 27.100 23.250 73.75 0.20000
## 134 24 4 1999 0.0 35.0 19.5 NA 27.500 23.250 69.50 0.15000
## 135 25 4 1999 0.0 35.2 17.0 NA 25.800 21.650 70.75 0.27500
## 136 26 4 1999 0.0 35.6 19.1 NA 26.350 22.500 72.75 0.05000
## 137 27 4 1999 0.0 31.6 19.9 NA 25.650 22.450 75.50 0.15000
## 138 28 4 1999 0.0 33.4 19.1 NA 25.000 22.000 78.00 0.25000
## 139 29 4 1999 0.0 34.0 18.1 NA 26.050 22.050 71.75 0.10000
## 140 30 4 1999 0.0 34.6 17.4 NA 24.450 21.450 80.00 0.00000

```

```

## 141   1 5 1999  0.0 35.0 18.3 NA 26.150 21.750 69.25 0.05000
## 142   2 5 1999  0.0 34.8 19.1 NA 26.750 22.150 67.75 0.20000
## 143   3 5 1999  0.0 35.0 16.8 NA 25.950 21.850 72.25 0.12500
## 144   4 5 1999  0.0 34.4 17.0 NA 25.350 21.750 73.25 0.17500
## 145   5 5 1999  0.0 35.6 18.1 NA 26.250 22.750 76.00 0.37500
## 146   6 5 1999  0.0 36.0 19.7 NA 26.200 22.950 75.50 0.07500
## 147   8 5 1999  0.0 30.6 16.8 NA 22.500 21.100 88.25 0.10000
## 148   9 5 1999  0.0 31.2 14.2 NA 23.450 19.650 70.25 0.22500
## 149  10 5 1999  0.0 34.0 14.4 NA 22.975 19.950 78.25 0.15000
## 150  11 5 1999  0.0 32.8 14.6 NA 24.950 21.350 71.50 0.15000
## 151  12 5 1999  0.0 33.0 16.0 NA 27.400 23.350 70.75 0.17500
## 152  13 5 1999  0.0 33.2 16.0 NA 24.900 19.700 61.25 0.07500
## 153  14 5 1999  0.0 32.4 14.6 NA 23.425 19.150 67.75 0.15000
## 154  15 5 1999  0.0 31.0 18.9 NA 23.500 19.200 66.50 0.37500
## 155  16 5 1999  0.0 32.6 13.8 NA 23.075 18.750 66.75 0.27500
## 156  17 5 1999  0.0 33.0 14.2 NA 22.625 18.500 68.00 0.22500
## 157  18 5 1999  0.0 33.2 13.2 NA 23.900 18.375 59.25 0.30000
## 158  19 5 1999  0.0 32.8 13.6 NA 23.475 18.950 66.00 0.15000
## 159  20 5 1999  0.0 29.6 15.2 NA 24.000 18.850 61.00 0.30000
## 160  21 5 1999  0.0 28.4 10.6 NA 19.275 15.400 66.75 0.20000
## 161  22 5 1999  0.0 29.2 7.4  NA 18.125 13.150 57.00 0.30000
## 162  23 5 1999  0.0 30.2 8.2  NA 19.425 14.750 61.75 0.17500
## 163  24 5 1999  0.0 30.2 10.2 NA 20.725 16.100 65.25 0.17500
## 164  25 5 1999  0.0 34.0 12.6 NA 21.675 17.050 65.25 0.12500
## 165  26 5 1999  0.0 31.6 13.2 NA 23.125 19.500 72.00 0.20000
## 166  27 5 1999  0.0 33.4 18.1 NA 25.650 21.750 70.75 0.15000
## 167  28 5 1999  0.0 34.2 17.0 NA 25.400 21.750 74.75 0.12500
## 168  29 5 1999  0.0 35.6 16.4 NA 26.850 21.900 65.50 0.25000
## 169  30 5 1999  0.0 26.2 12.2 NA 23.200 20.600 77.50 0.25000
## 170  31 5 1999  0.0 27.6 9.6  NA 18.575 14.650 65.50 0.37500
## 171   1 6 1999  0.0 30.4 6.0  NA 17.425 13.550 66.50 0.12500
## 172   2 6 1999  0.0 31.8 10.2 NA 21.125 16.700 65.50 0.20000
## 173   3 6 1999  0.0 30.8 11.8 NA 21.725 17.850 68.25 0.05000
## 174   4 6 1999  0.0 34.0 15.0 NA 24.475 20.825 70.25 0.12500
## 175   5 6 1999  0.0 33.2 16.0 NA 24.175 18.950 61.00 0.22500
## 176   6 6 1999  0.0 31.2 15.6 NA 23.825 19.550 66.50 0.22500
## 177   7 6 1999  0.0 32.4 14.0 NA 22.325 18.750 72.50 0.22500
## 178   8 6 1999  0.0 34.0 15.6 NA 24.300 19.850 67.50 0.10000
## 179   9 6 1999  0.0 34.0 14.8 NA 24.425 19.650 66.75 0.07500
## 180  10 6 1999  0.0 29.0 17.4 NA 23.750 19.350 65.75 0.12500
## 181  11 6 1999  0.0 31.4 13.4 NA 20.625 18.550 82.75 0.17500
## 182  12 6 1999  0.0 33.8 14.8 NA 24.175 20.950 78.25 0.15000
## 183  13 6 1999  0.0 33.4 17.0 NA 25.000 21.875 74.75 0.00000
## 184  14 6 1999  0.0 33.8 16.8 NA 24.850 20.800 71.25 0.15000
## 185  15 6 1999  0.0 34.8 17.2 NA 25.250 21.200 70.75 0.15000
## 186  16 6 1999  0.0 34.8 17.9 NA 25.500 21.250 71.50 0.10000

```

```

## 187 17 6 1999 0.0 35.0 14.8 NA 24.475 19.650 66.75 0.17500
## 188 18 6 1999 0.0 34.4 14.6 NA 25.550 19.550 57.00 0.15000
## 189 19 6 1999 NA 29.4 17.6 NA 23.375 19.850 70.75 0.12500
## 190 20 6 1999 0.0 28.0 16.6 NA 21.325 18.000 71.00 0.47500
## 191 21 6 1999 0.0 31.4 15.2 NA 22.625 19.350 74.25 0.20000
## 192 22 6 1999 0.0 34.4 16.0 NA 24.700 20.700 71.25 0.05000
## 193 23 6 1999 0.0 35.4 16.8 NA 25.500 20.650 65.50 0.20000
## 194 24 6 1999 0.0 35.8 17.3 NA 25.950 20.950 66.00 0.15000
## 195 25 6 1999 0.0 35.6 16.7 NA 26.175 20.400 60.25 0.22500
## 196 26 6 1999 0.0 35.0 16.9 NA 26.125 19.650 57.00 0.17500
## 197 27 6 1999 0.0 35.0 13.9 NA 25.850 18.800 51.25 0.25000
## 198 28 6 1999 0.0 35.8 15.7 NA 24.725 19.000 59.50 0.22500
## 199 29 6 1999 1.4 26.4 19.2 NA 24.150 21.400 76.50 0.10000
## 200 30 6 1999 0.0 32.4 19.4 NA 23.850 21.600 82.50 0.27500
## 201 1 7 1999 0.0 33.6 17.7 NA 25.250 21.800 76.25 0.22500
## 202 2 7 1999 0.0 34.4 17.1 NA 25.850 20.700 64.75 0.17500
## 203 3 7 1999 0.0 35.6 16.9 NA 26.450 19.650 52.75 0.12500
## 204 4 7 1999 0.0 24.7 15.6 NA 22.825 18.600 64.75 0.40000
## 205 5 7 1999 0.0 19.6 13.5 NA 16.050 14.750 85.75 0.57500
## 206 6 7 1999 0.0 24.8 13.1 NA 17.325 15.150 78.75 0.40000
## 207 7 7 1999 0.0 32.0 12.9 NA 19.825 18.550 88.50 0.15000
## 208 8 7 1999 0.0 33.2 15.7 NA 23.925 19.500 67.50 0.12500
## 209 9 7 1999 0.0 34.4 14.1 NA 24.275 18.450 58.00 0.17500
## 210 10 7 1999 0.0 35.0 14.9 NA 26.450 18.850 48.00 0.15000
## 211 11 7 1999 0.0 35.0 14.7 NA 24.800 19.300 58.00 0.12500
## 212 12 7 1999 0.0 34.4 14.7 NA 27.000 19.500 47.75 0.17500
## 213 13 7 1999 0.0 33.4 15.7 NA 25.800 18.100 46.50 0.20000
## 214 14 7 1999 0.0 33.2 14.7 NA 24.525 17.700 51.75 0.22500
## 215 15 7 1999 0.0 33.6 13.1 NA 24.075 17.000 48.25 0.02500
## 216 16 7 1999 0.0 34.2 11.9 NA 23.275 16.600 52.00 0.10000
## 217 17 7 1999 0.0 25.4 16.1 NA 21.975 17.450 62.00 0.27500
## 218 18 7 1999 0.0 28.8 13.3 NA 19.425 17.050 79.00 0.42500
## 219 19 7 1999 0.0 34.4 14.5 NA 21.975 18.000 71.50 0.32500
## 220 20 7 1999 0.0 34.8 16.1 NA 24.675 19.000 59.50 0.25000
## 221 21 7 1999 0.0 34.8 15.7 NA 25.225 19.850 62.00 0.32500
## 222 22 7 1999 0.0 34.2 17.9 NA 26.000 20.650 61.75 0.32500
## 223 23 7 1999 0.0 35.0 15.9 NA 25.325 19.950 63.00 0.22500
## 224 24 7 1999 0.0 35.2 17.3 NA 26.050 20.050 59.00 0.25000
## 225 25 7 1999 0.0 35.2 15.1 NA 25.800 18.350 49.50 0.17500
## 226 26 7 1999 0.0 35.8 15.3 NA 27.075 18.350 43.25 0.32500
## 227 27 7 1999 0.0 36.0 15.7 NA 27.900 19.100 42.50 0.20000
## 228 28 7 1999 0.0 36.0 14.9 NA 26.375 18.450 47.00 0.22500
## 229 29 7 1999 0.0 36.0 14.1 NA 27.350 18.500 42.00 0.17500
## 230 30 7 1999 NA 35.2 15.1 NA 24.025 18.000 55.75 0.22500
## 231 31 7 1999 0.0 34.6 17.3 NA 25.375 18.950 55.50 0.25000
## 232 1 8 1999 0.0 35.4 16.9 NA 26.900 19.450 50.50 0.17500

```

```

## 233   2   8 1999   0.0 34.6 15.5   NA 26.350 18.150   43.00 0.07500
## 234   3   8 1999   0.0 34.8 13.9   NA 26.075 18.000   44.75 0.20000
## 235   4   8 1999   0.0 35.4 14.1   NA 25.375 17.350   44.25 0.07500
## 236   5   8 1999   0.0 36.0 15.3   NA 25.875 17.250   41.00 0.12500
## 237   6   8 1999   0.0 35.2 15.3   NA 26.950 17.050   35.00 0.50000
## 238   7   8 1999   0.0 35.4 15.3   NA 26.050 18.250   45.25 0.10000
## 239   8   8 1999   0.0 32.6 16.7   NA 23.425 18.400   60.50 0.15000
## 240   9   8 1999   0.0 35.2 15.3   NA 25.175 18.000   48.25 0.30000
## 241  10   8 1999   0.0 36.2 15.7   NA 27.125 18.550   44.75 0.15000
## 242  11   8 1999   NA 37.2 15.0   NA 26.175 18.050   46.00 0.07500
## 243  12   8 1999   0.0 32.8 14.9   NA 32.575 18.100   NA 0.05000
## 244  13   8 1999   0.0 36.0 15.7   NA 27.075 18.950   46.25 0.15000
## 245  14   8 1999   0.0 33.4 13.5   NA 20.625 16.600   64.75 0.50000
## 246  15   8 1999   0.0 24.3  7.9   NA 21.025 13.600   38.50   NA
## 247  16   8 1999   NA 24.3  9.7   NA 15.425 12.075   65.00 0.47500
## 248   1   9 1999   0.0 38.6 17.5   NA 29.925 22.850   52.75 0.25000
## 249   2   9 1999   0.0 38.4 15.3   NA 29.650 19.300   35.50 0.12500
## 250   3   9 1999   0.0 38.4 17.9   NA 28.900 20.825   45.00 0.05000
## 251   4   9 1999   0.0 38.2 18.9   NA 29.250 20.350   42.50 0.15000
## 252   5   9 1999   0.0 39.8 19.4   NA 30.050 22.775   52.25 0.27500
## 253   6   9 1999   0.0 37.6 19.6   NA 29.950 22.525   51.25 0.07500
## 254  15   9 1999  10.8 26.0 19.8   NA 22.100 21.200   91.25 0.35000
## 255  16   9 1999   0.0 31.8 18.9   NA 24.050 21.750   82.50 0.10000
## 256  25   9 1999   0.0 36.2 17.9   NA 29.450 22.900   58.00 0.45000
## 257   3  10 1999   0.0 27.4 17.1   NA 23.975 19.150   62.00 0.95000
## 258   4  10 1999   0.0 31.6 18.3   NA 23.550 18.750   62.25 0.75000
## 259   3  11 1999   3.8 29.2 19.2   NA 23.600 23.900  100.00 0.25000
## 260   4  11 1999   2.9 30.0 19.4   NA 23.900 23.450   94.50 0.52500
## 261   6  11 1999   0.0 30.0 19.6   NA 22.650 19.800   78.25 0.37500
## 262   9  11 1999   0.9 28.6 17.6   NA 22.050 21.600   95.00 0.27500
## 263  10  11 1999   0.0 25.9 17.1   NA 20.725 20.050   91.50 0.57500
## 264  22  11 1999   0.0 36.0 19.6   NA 25.700 22.000   75.00 0.32500
## 265  23  11 1999   0.2 34.8 19.8   NA 26.800 22.300   68.75 0.50000
## 266   2  12 1999   0.0 35.8 18.8   NA 31.950 24.550   53.25 0.27500
## 267  26  12 1999   2.2 31.2 19.8   NA 24.200 22.100   83.75 0.65000
## 268   9   2 2000   2.5 26.9 16.2   NA 23.700 23.350   96.00 0.32500
## 269  10   2 2000   2.4 29.8 16.3   NA 24.350 24.100   97.25 0.27500
## 270  21   4 2000   0.0 30.6 17.9   NA 23.900 21.450   80.50 0.30000
## 271  22   4 2000   0.0 31.6 15.9   NA 23.775 20.400   74.00 0.37500
## 272  23   4 2000   0.0 31.0 12.9   NA 22.500 19.100   72.00 0.42500
## 273  24   4 2000   0.0 32.8 15.9   NA 25.925 21.600   69.25 0.30000
## 274  25   4 2000   0.0 32.4 17.9   NA 26.050 22.350   72.25 0.25000
## 275  26   4 2000   0.0 33.8 19.4   NA 26.300 23.350   77.00 0.22500
## 276  27   4 2000   1.0 32.0 19.4   NA 26.000 23.000   78.50 0.25000
## 277  30   4 2000   NA 32.0 19.6   NA 24.150 21.975   84.00 0.17500
## 278   1   5 2000   0.0 33.2 18.7   NA 26.650 23.200   75.25 0.22500

```

```

## 279  2  5 2000  0.0 33.2 19.4 NA 27.600 23.350 70.50 0.32500
## 280  6  5 2000  0.0 27.4 18.8 NA 23.750 22.050 86.00 0.25000
## 281  7  5 2000  0.0 29.2 18.9 NA 21.600 19.950 85.75 0.27500
## 282  8  5 2000  NA 31.4 18.7 NA 23.200 21.600 87.25 0.15000
## 283  9  5 2000  0.0 31.8 18.7 NA 25.250 22.650 81.00 0.20000
## 284 10  5 2000  0.0 33.8 19.8 NA 25.500 22.850 81.75 0.07500
## 285 12  5 2000  0.0 33.6 18.3 NA 25.400 21.750 74.75     NA
## 286 13  5 2000  0.0 33.0 17.3 NA 25.250 21.700 74.75 0.32500
## 287 14  5 2000  0.0 33.4 17.5 NA 24.850 21.250 74.75 0.20000
## 288 15  5 2000  0.0 33.8 16.7 NA 25.050 20.950 70.50 0.45000
## 289 16  5 2000  0.0 33.8 16.3 NA 24.450 20.650 72.75 0.30000
## 290 17  5 2000  5.0 28.7 18.4 NA 23.000 21.150 84.75 0.27500
## 291 18  5 2000  0.0 26.0 15.5 NA 23.050 20.600 79.75 0.55000
## 292 19  5 2000  0.0 26.2 13.5 NA 19.425 16.850 78.50 0.32500
## 293 20  5 2000  0.0 26.8 13.9 NA 19.275 17.050 80.75 0.25000
## 294 21  5 2000  0.0 31.0 14.7 NA 21.375 18.250 75.50 0.35000
## 295 22  5 2000  0.0 26.8 15.7 NA 23.550 20.050 73.00 0.30000
## 296 23  5 2000  0.0 33.8 16.5 NA 23.100 19.350 72.25 0.27500
## 297 24  5 2000  0.0 33.0 15.7 NA 25.150 20.500 68.00 0.30000
## 298 25  5 2000  0.0 34.0 16.1 NA 24.950 21.250 74.00 0.25000
## 299 26  5 2000  0.0 34.0 17.7 NA 26.750 22.550 70.00 0.32500
## 300 27  5 2000  0.0 34.2 16.9 NA 26.100 22.400 74.25 0.15000
## 301 28  5 2000  0.0 34.4 18.1 NA 25.950 22.100 72.75 0.17500
## 302 29  5 2000  0.0 33.8 17.7 NA 25.950 22.300 73.75 0.25000
## 303 30  5 2000  0.0 33.8 18.3 NA 26.600 22.550 73.00 0.15000
## 304 31  5 2000  0.0 33.8 17.1 NA 25.250 20.800 68.50 0.30000
## 305  1  6 2000  0.0 31.8 19.4 NA 23.950 20.850 77.00     NA
## 306  2  6 2000  0.0 31.8 19.4 NA 23.825 21.150 80.50     NA
## 307  3  6 2000  0.0 33.2 16.3 NA 25.850 21.100 79.00 0.17500
## 308  4  6 2000  0.0 34.8 18.3 NA 26.200 22.100 70.00 0.22500
## 309  5  6 2000  0.0 33.8 16.5 NA 26.650 20.400 56.00 0.27500
## 310  6  6 2000  0.0 33.4 15.5 NA 26.550 20.850 58.00 0.45000
## 311  7  6 2000  0.0 33.2 14.5 NA 24.325 19.700 67.50 0.35000
## 312  8  6 2000  0.0 33.2 15.5 NA 24.125 19.650 68.25 0.15000
## 313  9  6 2000  0.0 32.8 14.3 NA 25.500 20.400 64.50 0.27500
## 314 10  6 2000  0.0 33.4 13.9 NA 26.775 20.400 57.75 0.22500
## 315 11  6 2000  0.0 33.2 14.5 NA     NA     NA     NA     NA
## 316 13  6 2000  0.0 34.0 13.5 NA     NA     NA     NA     NA
## 317 14  6 2000  NA 34.1 19.2 NA     NA     NA     NA     NA
## 318 15  6 2000  0.0 35.0 15.5 NA     NA     NA     NA     NA
## 319 17  6 2000  0.0 34.5 17.9 NA     NA     NA     NA     NA
## 320 18  6 2000  NA 34.3 18.9 NA     NA     NA     NA     NA
## 321 19  6 2000  0.0 26.5 15.5 NA     NA     NA     NA     NA
## 322 20  6 2000  0.0 24.6 14.7 NA 22.775 17.800 63.75 0.45000
## 323 21  6 2000  0.0 24.8 15.3 NA 17.275 15.650 85.50 0.47500
## 324 22  6 2000  0.0 30.0 12.1 NA 19.575 17.050 81.75 0.22500

```

```

## 325 23 6 2000 0.0 34.2 13.3 NA 23.075 18.800 69.25 0.20000
## 326 24 6 2000 0.0 34.6 15.1 NA 26.500 20.200 59.50 0.30000
## 327 25 6 2000 NA 34.6 18.1 NA 27.000 21.050 58.00 0.22500
## 328 26 6 2000 0.0 34.8 16.2 NA 25.350 20.550 65.50 0.20000
## 329 27 6 2000 NA 34.0 15.3 NA 25.450 19.550 60.25 0.20000
## 330 28 6 2000 0.0 33.4 13.9 NA 25.875 18.400 48.75 0.37500
## 331 29 6 2000 NA 34.4 15.3 NA 25.775 19.750 58.25 0.30000
## 332 30 6 2000 0.0 34.6 18.9 NA 27.000 20.400 54.50 0.30000
## 333 1 7 2000 NA 34.0 17.9 NA 27.500 20.350 53.25 0.32500
## 334 2 7 2000 NA 35.4 15.8 NA 26.750 21.100 61.50 0.37500
## 335 3 7 2000 NA 34.2 15.9 NA 25.750 20.750 64.25 0.27500
## 336 4 7 2000 NA 34.6 17.5 NA 24.800 19.150 60.00 0.37500
## 337 5 7 2000 NA 34.2 16.5 NA 24.925 19.700 63.25 0.27500
## 338 6 7 2000 0.0 35.2 14.9 NA 24.325 17.900 54.00 0.47500
## 339 7 7 2000 0.0 34.0 13.7 NA 24.675 19.650 64.25 2.17500
## 340 8 7 2000 0.0 34.4 15.7 NA 25.600 19.300 55.75 0.32500
## 341 9 7 2000 0.0 34.8 16.3 NA 25.950 19.350 55.00 0.22500
## 342 10 7 2000 0.0 34.8 17.1 NA 25.850 19.500 56.50 0.30000
## 343 11 7 2000 0.0 26.0 15.7 NA 21.200 17.800 72.00 0.62500
## 344 12 7 2000 NA 26.6 9.7 NA 15.225 12.325 71.75 0.75000
## 345 13 7 2000 0.0 25.0 9.3 NA 14.150 10.975 68.75 0.65000
## 346 14 7 2000 NA 28.6 6.7 NA 16.300 11.525 56.75 0.35000
## 347 15 7 2000 0.0 32.8 11.1 NA 21.475 16.300 60.25 0.37500
## 348 16 7 2000 0.0 24.2 14.4 NA 22.375 17.000 58.75 0.60000
## 349 17 7 2000 0.0 26.8 6.9 NA 16.450 11.675 58.25 0.27500
## 350 18 7 2000 0.0 31.6 7.6 NA 19.725 13.350 49.00 0.22500
## 351 19 7 2000 0.0 29.0 7.9 NA 21.725 15.950 51.50 0.42500
## 352 20 7 2000 0.0 28.8 10.1 NA 20.225 14.550 53.75 0.42500
## 353 21 7 2000 NA 34.8 11.1 NA 21.975 16.700 61.25 0.37500
## 354 22 7 2000 0.0 33.0 15.3 NA 23.800 17.550 51.50 0.42500
## 355 23 7 2000 2.0 20.9 12.6 NA 22.625 17.250 62.00 0.52500
## 356 24 7 2000 NA 24.4 7.1 NA 14.650 12.425 81.50 0.22500
## 357 25 7 2000 0.0 29.8 10.7 NA 18.075 13.800 64.50 0.37500
## 358 26 7 2000 0.0 32.8 12.9 NA 21.575 16.600 61.25 0.27500
## 359 27 7 2000 0.0 34.8 13.7 NA 23.875 18.250 60.50 0.22500
## 360 28 7 2000 0.0 35.6 16.1 NA 26.650 19.750 52.75 0.25000
## 361 29 7 2000 0.0 35.2 16.1 NA 27.400 19.500 48.00 0.17500
## 362 30 7 2000 0.0 35.6 14.7 NA 25.275 18.500 55.75 0.22500
## 363 31 7 2000 0.0 35.2 15.9 NA 25.225 18.550 54.50 0.12500
## 364 1 8 2000 0.0 35.8 14.5 NA 24.500 17.400 50.50 0.20000
## 365 2 8 2000 0.0 37.0 14.1 NA 26.950 17.800 40.25 0.42500
## 366 3 8 2000 0.0 31.8 18.5 NA 24.900 19.400 59.75 0.10000
## 367 4 8 2000 0.0 33.2 16.5 NA 24.250 20.100 70.25 0.22500
## 368 5 8 2000 0.0 36.2 15.5 NA 25.275 19.450 61.25 0.32500
## 369 6 8 2000 0.0 38.2 19.4 NA 26.500 19.550 52.00 0.15000
## 370 7 8 2000 0.0 37.8 18.1 NA 28.600 20.150 45.75 0.12500

```

```

## 371   8   8 2000   0.0 37.4 17.5   NA 28.650 20.250   45.50 0.32500
## 372   9   8 2000   0.0 38.6 18.1   NA 28.650 19.850   44.50 0.40000
## 373  10   8 2000   0.0 36.6 17.1   NA 27.550 20.450   52.50 0.30000
## 374  11   8 2000   NA 29.6 17.2   NA 25.050 20.650   67.00 0.75000
## 375  12   8 2000   0.0 26.6 12.5   NA 23.250 17.500   83.00 0.67500
## 376  13   8 2000   0.0 34.4 14.0   NA 23.950 19.700   64.50 0.30000
## 377  14   8 2000   0.0 37.8 13.1   NA 26.850 19.950   53.50 0.30000
## 378  16   8 2000   0.0 35.0 19.4   NA 27.050 22.250   66.50 0.20000
## 379  17   8 2000   0.0 37.6 18.7   NA 27.600 21.850   62.00 0.27500
## 380  19   8 2000   0.0 37.4 18.5   NA 27.900 19.950   48.50 0.30000
## 381  20   8 2000   0.0 38.8 18.3   NA 29.650 20.250   44.00 0.22500
## 382  21   8 2000   0.0 38.8 15.9   NA 29.250 20.000   42.00 0.30000
## 383  25   8 2000   0.0 38.4 18.5   NA 29.800 19.950   39.00 0.27500
## 384  26   8 2000   0.0 37.6 18.9   NA 30.600 21.100   43.00 0.30000
## 385  27   8 2000   0.0 38.2 18.5   NA 29.450 21.300   48.00 0.25000
## 386  29   8 2000   0.0 24.8 18.5   NA 21.750 19.900   84.00 0.45000
## 387  30   8 2000   0.0 31.8 18.7   NA 23.450 19.850   74.25 0.42500
## 388  31   8 2000   0.6 38.4 18.7   NA 28.550 21.900   57.00 0.25000
## 389   2   9 2000   0.0 29.0 19.8   0.0 23.750 21.550   82.25 0.25000
## 390   3   9 2000   0.0 27.6 16.8   0.0 22.450 20.450   84.00 0.47500
## 391   4   9 2000   0.0 33.4 15.7   0.0 21.750 18.300   75.50 0.45000
## 392   5   9 2000   0.0 36.4 19.4   0.0 27.550 21.150   57.00 0.37500
## 393   9   9 2000   2.0 32.6 18.8   0.0 21.925 21.250   94.00 0.40000
## 394  10   9 2000   4.4 36.2 18.9   0.0 25.500 22.550   81.00 0.17500
## 395  15   9 2000   0.0 28.2 18.9   0.0 23.750 21.900   85.25 0.22500
## 396  16   9 2000   0.0 27.6 16.3   0.0 21.225 18.600   78.50 0.52500
## 397  17   9 2000   0.0 34.4 16.9   0.0 24.000 20.200   72.00 0.25000
## 398  21   9 2000   0.0 38.4 18.9   0.0 30.550 20.700   40.00 0.32500
## 399  22   9 2000   0.0 37.8 18.9   0.0 30.900 20.450   38.25 0.40000
## 400  24   9 2000   0.0 36.0 15.9   0.0 28.950 22.000   55.25 0.37500
## 401  25   9 2000   1.3 31.4 18.4   0.0 21.975 19.900   82.50 0.50000
## 402  26   9 2000   0.0 27.0 15.7   0.0 21.625 16.650   59.50 0.42500
## 403  27   9 2000   0.0 27.2 15.7   0.0 22.300 18.250   67.00 0.62500
## 404  28   9 2000   0.0 32.2 18.7   0.0 23.500 20.350   76.75 0.42500
## 405  29   9 2000   0.0 32.8 19.4   0.0 26.100 21.650   68.00 0.27500
## 406  30   9 2000   NA 33.6 18.1   0.0 23.700 21.350   84.50 0.30000
## 407   1  10 2000   0.0 36.6 16.5   NA 26.200 21.300   64.00 0.30000
## 408   2  10 2000   0.0 36.4 18.9   NA 28.300 22.750   63.25 0.27500
## 409   6  10 2000   0.0 32.8 19.8   NA 24.550 22.200   82.75 0.30000
## 410  17  10 2000   0.0 37.6 18.6   NA 26.850 23.550   76.00 0.57500
## 411  18  10 2000   0.4 36.4 18.7   NA 25.600 22.600   80.50 0.37500
## 412  24  10 2000   0.0 31.4 19.8   NA 24.700 23.000   87.75 0.20000
## 413  25  10 2000   2.0 31.4 19.8   NA 23.600 22.250   90.00 0.17500
## 414  15   1 2001  30.8 33.1 19.8   NA 24.550 22.350      NA 0.52500
## 415  17   1 2001   6.8 33.9 19.6   NA 24.850 22.900      NA 0.32500
## 416  27   2 2001   0.0 32.3 19.6   NA 24.500 22.750   88.00 0.30000

```

```

## 417   2   4 2001  46.5 30.5 19.6 NA 24.400 22.650  87.00 0.22500
## 418   15  4 2001   0.0 34.7 18.3 NA 26.850 22.700      NA 0.27500
## 419   16  4 2001   0.0 32.3 18.9 NA 25.750 22.350      NA 0.35000
## 420   17  4 2001   0.0 34.3 19.8 NA 27.050 23.550  76.00 0.07500
## 421   18  4 2001   0.0 34.3 19.6 NA 27.300 23.800  76.50 0.25000
## 422   20  4 2001   0.0 34.5 19.4 NA 27.300 23.850  78.00 0.32500
## 423   27  4 2001   0.0 33.3 19.8 NA 25.600 23.450      NA 0.20000
## 424   28  4 2001   0.0 33.7 19.4 NA 27.050 23.600  75.50 0.27500
## 425   29  4 2001   0.0 29.0 17.9 NA 25.750 23.350      NA 0.20000
## 426   30  4 2001   0.0 31.1 18.1 NA 23.750 22.100  87.75 0.22500
## 427    1  5 2001   0.0 31.7 17.7 NA 23.375 21.550      NA 0.30000
## 428    2  5 2001   0.0 34.5 18.5 NA 26.550 22.650  72.75 0.30000
## 429    3  5 2001   0.0 34.1 18.7 NA 26.350 22.950      NA 0.25000
## 430    5  5 2001   0.0 25.2 16.3 NA 20.525 18.800  85.25 0.52500
## 431    6  5 2001   0.0 29.7 14.1 NA 20.475 18.300      NA 0.22500
## 432    7  5 2001   0.0 30.3 16.9 NA 22.950 20.350  79.50 0.22500
## 433    8  5 2001   0.0 29.9 16.1 NA 23.700 21.100  80.25 0.25000
## 434    9  5 2001   0.0 31.3 15.3 NA 23.725 20.850      NA 0.15000
## 435   12  5 2001   0.4 29.1 18.5 NA 22.675 21.200  88.00 0.30000
## 436   13  5 2001   0.0 26.0 17.3 NA 21.925 19.850  81.75 0.37500
## 437   14  5 2001   0.0 25.8 18.3 NA 22.200 20.600  86.00 0.25000
## 438   15  5 2001   0.0 31.7 16.7 NA 22.600 20.700      NA 0.25000
## 439   16  5 2001   0.0 31.5 19.4 NA 25.500 23.000      NA 0.40000
## 440   17  5 2001   0.0 27.0 18.1 NA 23.175 20.700  80.25 0.55000
## 441   18  5 2001   0.0 29.1 14.5 NA 20.725 18.500  82.25 0.20000
## 442   19  5 2001   0.0 31.5 16.1 NA 23.750 20.700  77.50 0.25000
## 443   20  5 2001   0.0 33.5 17.5 NA 25.250 21.900  77.00 0.12500
## 444   21  5 2001   0.0 33.5 17.5 NA 25.700 21.900      NA 0.22500
## 445   22  5 2001   0.0 34.5 17.5 NA 26.000 22.100      NA 0.25000
## 446   23  5 2001   0.0 22.1 19.2 NA 22.550 21.050  87.25 0.40000
## 447   24  5 2001  28.0 27.6 17.5 NA 21.575 20.750  93.50 0.37500
## 448   25  5 2001   0.0 30.9 18.1 NA 24.400 22.250  83.75 0.25000
## 449   26  5 2001   0.0 26.8 19.8 NA 22.900 22.300  94.50 0.07500
## 450   27  5 2001  13.2 29.0 19.6 NA 22.950 21.600  89.50 0.35000
## 451   28  5 2001   0.0 31.7 18.5 NA 25.400 22.950      NA 0.35000
## 452   29  5 2001   6.8 32.3 18.5 NA 24.050 21.850  83.75 0.60000
## 453   31  5 2001   0.0 34.1 19.2 NA 26.100 23.550  83.25 0.27500
## 454    1  6 2001   0.0 33.1 18.3 NA 25.900 22.750      NA 0.30000
## 455    2  6 2001   0.0 32.7 17.5 NA 25.250 22.050  78.50 0.30000
## 456    3  6 2001   0.0 32.1 14.7 NA 24.300 20.250  70.75 0.30000
## 457    4  6 2001   0.0 31.7 16.3 NA 24.450 21.500      NA 0.10000
## 458    5  6 2001   0.0 32.7 17.3 NA 24.800 21.450      NA 0.20000
## 459    6  6 2001   0.0 33.1 18.1 NA 26.350 23.250      NA 0.20000
## 460    7  6 2001   0.0 32.5 17.3 NA 25.800 22.000  73.50 0.27500
## 461    8  6 2001   0.0 32.1 16.3 NA 24.750 20.950  72.50 0.30000
## 462    9  6 2001      NA 32.5 15.1 NA 24.300 21.150  78.25 0.15000

```

```

## 463 10 6 2001 0.0 32.9 15.3 NA 24.275 20.500 73.75 0.22500
## 464 11 6 2001 0.0 31.1 14.9 NA 24.050 20.400 73.50 0.25000
## 465 12 6 2001 0.0 31.7 13.5 NA 23.075 19.700 76.00 0.25000
## 466 13 6 2001 0.0 31.9 16.9 NA 24.200 20.400 70.75 0.12500
## 467 14 6 2001 0.0 31.9 16.1 NA 24.400 20.700 73.00 0.22500
## 468 15 6 2001 0.0 32.7 16.1 NA 24.400 20.450 71.50 0.17500
## 469 16 6 2001 0.0 32.5 15.7 NA 24.450 20.150 NA 0.17500
## 470 17 6 2001 0.0 24.7 13.9 NA 20.750 18.750 82.25 0.60000
## 471 18 6 2001 0.0 NA 11.7 NA 14.200 12.950 86.25 0.72500
## 472 19 6 2001 0.0 NA 10.9 NA 12.625 NA NA 0.60000
## 473 20 6 2001 0.0 NA 10.7 NA 13.250 NA 84.75 0.47500
## 474 21 6 2001 0.0 22.7 8.5 NA 15.675 13.400 78.25 0.42500
## 475 22 6 2001 0.0 NA 8.5 NA NA NA 70.75 0.30000
## 476 23 6 2001 0.0 29.9 9.5 NA 18.575 NA 72.00 0.30000
## 477 24 6 2001 0.0 31.3 13.0 NA 22.325 18.850 NA 0.17500
## 478 25 6 2001 NA 31.3 17.9 NA 24.200 20.900 75.25 0.25000
## 479 26 6 2001 0.8 26.6 14.1 NA 21.175 NA 76.00 0.45000
## 480 27 6 2001 0.0 27.8 10.3 NA 18.925 NA 65.50 0.22500
## 481 28 6 2001 0.0 28.8 9.1 NA 19.175 15.500 NA 0.30000
## 482 29 6 2001 0.0 30.5 11.3 NA 21.325 17.600 NA 0.20000
## 483 30 6 2001 0.0 32.7 12.7 NA 22.775 18.750 70.50 0.07500
## 484 1 7 2001 0.0 31.3 13.9 NA 23.475 19.050 67.25 0.20000
## 485 2 7 2001 0.0 32.7 14.1 NA 24.175 19.650 69.00 0.35000
## 486 3 7 2001 0.0 32.9 13.5 NA 23.475 19.150 70.50 0.10000
## 487 4 7 2001 0.0 33.5 12.5 NA 24.175 18.650 61.00 0.25000
## 488 5 7 2001 0.0 33.3 14.1 NA 24.375 19.300 63.75 0.07500
## 489 6 7 2001 0.0 33.5 14.5 NA 25.550 19.400 58.25 0.30000
## 490 7 7 2001 0.0 32.9 10.7 NA 22.775 17.300 61.00 0.15000
## 491 8 7 2001 0.0 33.7 14.9 NA 24.825 19.700 64.25 0.37500
## 492 9 7 2001 0.0 33.3 15.3 NA 25.975 19.450 55.25 0.62500
## 493 10 7 2001 0.0 34.1 14.7 NA 25.400 20.000 62.75 0.25000
## 494 11 7 2001 0.0 34.3 13.5 NA 24.425 18.750 59.00 0.12500
## 495 12 7 2001 0.0 27.2 18.9 NA 24.350 21.550 78.00 0.22500
## 496 13 7 2001 0.0 30.7 14.9 NA 21.575 19.700 81.50 0.22500
## 497 14 7 2001 0.0 34.1 15.7 NA 24.750 20.700 71.50 0.22500
## 498 15 7 2001 0.0 34.5 18.3 NA 26.850 21.150 61.25 0.20000
## 499 16 7 2001 0.0 33.9 16.7 NA 26.000 20.300 61.50 0.35000
## 500 17 7 2001 0.0 34.3 16.7 NA 26.350 20.250 58.75 0.35000
## 501 18 7 2001 0.0 34.9 14.9 NA 25.600 19.050 55.00 0.17500
## 502 19 7 2001 0.0 35.7 15.7 NA 26.400 19.850 55.75 0.20000
## 503 20 7 2001 0.0 35.7 16.7 NA 26.700 20.250 55.50 0.32500
## 504 21 7 2001 0.0 35.7 17.5 NA 27.000 21.300 61.25 0.37500
## 505 22 7 2001 0.0 35.7 15.5 NA 27.200 21.400 61.00 0.25000
## 506 23 7 2001 0.0 34.3 19.2 NA 26.750 21.900 66.50 0.42500
## 507 25 7 2001 0.0 34.5 18.7 NA 24.600 21.400 74.00 0.27500
## 508 26 7 2001 0.2 35.3 18.3 NA 25.900 22.100 74.00 0.20000

```

```

## 509 27 7 2001 0.0 25.6 12.9 NA 20.450 NA 78.50 NA
## 510 28 7 2001 0.0 24.3 10.3 NA 15.100 12.575 NA NA
## 511 29 7 2001 0.0 32.3 10.3 NA 19.625 15.600 NA 0.25000
## 512 30 7 2001 0.0 34.1 14.3 NA 24.650 19.500 64.25 0.37500
## 513 31 7 2001 0.0 34.3 14.3 NA 26.250 19.700 NA 0.35000
## 514 1 8 2001 0.0 35.1 14.9 NA 26.525 18.700 NA 0.30000
## 515 2 8 2001 0.0 34.1 14.9 NA 26.050 19.050 52.50 0.35000
## 516 3 8 2001 0.0 34.1 13.3 NA 25.650 18.650 53.00 0.30000
## 517 4 8 2001 0.0 33.7 14.3 NA 25.200 18.150 51.25 0.15000
## 518 5 8 2001 0.0 34.1 14.9 NA 26.950 19.200 48.75 0.27500
## 519 6 8 2001 0.0 34.3 13.7 NA 25.475 17.950 48.00 0.35000
## 520 7 8 2001 0.0 34.9 14.5 NA 25.900 18.350 47.75 0.27500
## 521 8 8 2001 0.0 34.3 13.9 NA 25.450 18.350 52.25 0.20000
## 522 9 8 2001 0.0 34.5 11.9 NA 25.025 17.050 45.75 0.22500
## 523 10 8 2001 0.0 35.1 12.5 NA 24.125 16.650 47.50 0.27500
## 524 11 8 2001 0.0 35.3 13.3 NA 26.325 17.700 42.75 0.25000
## 525 12 8 2001 0.0 35.5 13.7 NA 26.500 18.400 46.00 0.22500
## 526 13 8 2001 0.0 34.7 14.7 NA 27.400 17.600 37.25 0.27500
## 527 14 8 2001 0.0 33.5 14.9 NA 25.900 17.700 43.25 0.40000
## 528 15 8 2001 0.0 34.7 14.9 NA 26.350 18.600 48.25 0.25000
## 529 16 8 2001 0.0 35.3 14.7 NA 26.800 18.250 43.25 0.35000
## 530 17 8 2001 0.0 36.1 15.5 NA 27.800 19.050 43.25 0.20000
## 531 18 8 2001 0.0 36.3 15.3 NA 27.500 18.350 40.75 0.35000
## 532 19 8 2001 0.0 35.9 15.7 NA 28.500 18.750 38.25 0.45000
## 533 20 8 2001 0.0 36.9 14.9 NA 28.450 19.800 44.75 0.20000
## 534 21 8 2001 0.0 35.5 17.5 NA 28.300 19.100 41.00 0.42500
## 535 22 8 2001 0.0 36.9 17.5 NA 28.200 19.450 43.75 0.32500
## 536 23 8 2001 0.0 37.7 17.1 NA 27.850 19.350 45.25 0.25000
## 537 24 8 2001 0.0 37.9 18.7 NA 28.900 20.200 45.00 0.30000
## 538 25 8 2001 0.0 36.5 19.4 NA 28.900 21.150 50.25 0.22500
## 539 26 8 2001 5.5 32.3 19.6 NA 24.250 20.750 74.75 0.67500
## 540 27 8 2001 1.1 35.5 19.4 NA 27.000 22.200 67.25 0.30000
## 541 28 8 2001 2.0 35.5 17.3 NA 25.550 21.400 70.75 0.30000
## 542 29 8 2001 0.0 34.5 19.4 NA 25.450 21.150 69.50 0.32500
## 543 30 8 2001 0.0 36.5 18.7 NA 28.300 22.150 59.50 0.25000
## 544 31 8 2001 0.0 36.9 19.2 NA 27.650 21.850 62.25 0.22500
## 545 19 9 2001 0.0 NA 15.9 NA NA NA NA NA
## 546 20 9 2001 0.0 NA 19.8 NA NA NA NA NA
## 547 21 9 2001 0.0 NA 18.1 NA NA NA NA NA
## 548 23 9 2001 0.0 NA 19.2 NA NA NA NA NA
## 549 3 10 2001 0.0 35.3 19.8 NA NA NA NA NA
## 550 9 10 2001 17.7 31.3 19.4 NA 24.850 NA 86.00 0.35000
## 551 19 10 2001 17.4 NA 19.8 NA 23.350 NA 86.00 0.42500
## 552 22 10 2001 0.0 NA 18.1 NA 24.350 NA 79.75 0.22500
## 553 23 10 2001 0.0 26.7 19.2 NA NA NA NA NA
## 554 19 1 2002 0.0 33.1 19.6 NA 26.350 NA 74.75 1.68350

```

## 555	15	2	2002	3.0	30.5	17.0	NA	NA	NA	NA	NA
## 556	22	4	2002	0.0	34.1	19.9	NA	27.900	NA	76.50	0.51444
## 557	24	4	2002	0.0	33.9	18.9	NA	27.000	NA	74.75	0.90027
## 558	25	4	2002	0.0	34.1	19.4	NA	26.850	NA	74.00	0.38583
## 559	26	4	2002	0.0	33.7	19.4	NA	26.300	NA	78.25	0.90027
## 560	9	5	2002	0.0	32.5	19.9	NA	25.050	NA	80.50	2.18637
## 561	10	5	2002	0.0	32.3	19.9	NA	26.200	NA	79.50	1.67193
## 562	11	5	2002	0.0	33.5	19.9	NA	26.350	NA	81.50	1.02888
## 563	12	5	2002	0.0	34.1	19.9	NA	NA	NA	NA	NA
## 564	14	5	2002	0.0	35.1	19.9	NA	NA	NA	NA	NA
## 565	18	5	2002	0.0	35.5	19.4	NA	27.300	NA	73.00	0.90027
## 566	21	5	2002	6.0	23.7	19.9	NA	22.150	NA	95.75	0.77166
## 567	22	5	2002	2.0	27.0	17.8	NA	21.200	NA	89.25	2.44359
## 568	23	5	2002	0.0	28.6	17.8	NA	21.825	NA	88.75	1.54332
## 569	24	5	2002	0.0	NA	15.3	NA	23.575	NA	77.25	1.67193
## 570	25	5	2002	0.0	31.7	16.3	NA	NA	NA	NA	NA
## 571	26	5	2002	0.0	31.9	15.8	NA	24.600	NA	71.75	0.77166
## 572	28	5	2002	0.0	31.9	13.7	NA	24.900	NA	59.00	1.54332
## 573	29	5	2002	0.0	31.5	12.7	NA	23.125	NA	63.50	0.38583
## 574	30	5	2002	0.0	31.5	13.2	NA	23.150	NA	66.25	0.00000
## 575	31	5	2002	0.0	34.1	12.7	NA	22.725	NA	73.50	NA
## 576	1	6	2002	0.0	33.3	12.7	NA	25.250	NA	67.25	1.28610
## 577	2	6	2002	0.0	33.1	15.8	NA	25.150	NA	69.75	0.51444
## 578	3	6	2002	0.0	33.9	16.8	NA	25.600	NA	68.25	0.64305
## 579	4	6	2002	0.0	33.5	15.3	NA	25.475	NA	65.25	1.67193
## 580	5	6	2002	0.0	33.3	15.3	NA	24.825	NA	62.00	0.64305
## 581	6	6	2002	0.0	33.1	13.7	NA	24.725	NA	65.75	0.12861
## 582	7	6	2002	0.0	33.9	14.0	NA	24.525	NA	69.25	0.51444
## 583	8	6	2002	0.0	33.9	14.3	NA	25.150	NA	62.50	0.90027
## 584	9	6	2002	0.0	34.3	12.2	NA	23.875	NA	65.75	1.28610
## 585	10	6	2002	0.0	33.9	12.8	NA	24.825	NA	69.50	0.38583
## 586	11	6	2002	0.0	33.9	15.8	NA	25.650	NA	61.50	0.90027
## 587	13	6	2002	0.0	26.8	16.9	NA	20.925	NA	85.25	2.18637
## 588	14	6	2002	0.0	24.9	15.3	NA	18.525	NA	91.00	2.82942
## 589	15	6	2002	0.0	26.4	16.3	NA	19.925	NA	85.75	3.60108
## 590	16	6	2002	0.0	31.9	17.3	NA	21.425	NA	82.00	NA
## 591	17	6	2002	0.0	32.9	15.8	NA	NA	NA	NA	NA
## 592	18	6	2002	0.0	33.1	16.3	NA	25.750	NA	63.00	2.05776
## 593	19	6	2002	0.0	32.9	17.3	NA	24.850	NA	68.75	2.70081
## 594	21	6	2002	0.0	31.9	14.8	NA	23.525	NA	68.50	2.31498
## 595	22	6	2002	0.0	31.7	14.3	NA	24.225	NA	61.00	2.05776
## 596	23	6	2002	0.0	26.2	14.3	NA	25.575	NA	56.50	1.67193
## 597	24	6	2002	0.0	29.3	12.7	NA	19.425	NA	82.75	1.67193
## 598	25	6	2002	0.0	32.1	13.7	NA	22.275	NA	74.00	1.67193
## 599	26	6	2002	0.0	33.1	16.8	NA	24.900	NA	67.00	1.67193
## 600	27	6	2002	0.0	34.5	14.3	NA	25.300	NA	57.00	1.67193

```

## 601 28 6 2002 0.0 34.3 14.3 NA 26.550 NA 62.75 0.90027
## 602 29 6 2002 0.0 33.9 19.9 NA 26.800 NA 59.25 2.31498
## 603 30 6 2002 0.0 36.3 16.8 NA 26.450 NA 59.25 1.41471
## 604 2 7 2002 0.0 33.3 15.3 NA 25.400 NA 60.00 0.12861
## 605 3 7 2002 0.0 33.3 13.7 NA 24.325 NA 63.25 1.41471
## 606 4 7 2002 0.0 NA 13.2 NA 26.400 NA 48.75 0.77166
## 607 5 7 2002 0.0 30.7 15.8 NA NA NA NA NA NA
## 608 7 7 2002 0.0 NA 15.3 NA NA NA NA NA NA
## 609 9 7 2002 0.0 30.3 12.2 NA 20.925 NA 59.00 0.38583
## 610 10 7 2002 0.0 NA 17.8 NA 25.100 NA 57.00 NA
## 611 12 7 2002 0.0 33.5 19.4 NA 26.650 NA 60.00 1.02888
## 612 14 7 2002 0.0 31.5 16.8 NA 23.450 NA 69.50 0.77166
## 613 15 7 2002 0.0 33.9 16.7 NA 26.750 NA 55.25 0.90027
## 614 16 7 2002 0.0 36.9 15.3 NA 25.900 NA 60.00 1.80054
## 615 17 7 2002 0.0 34.7 16.3 NA 26.750 NA 58.50 1.02888
## 616 18 7 2002 0.0 34.7 16.3 NA 26.750 NA 57.25 0.64305
## 617 19 7 2002 0.0 34.9 17.3 NA 26.150 NA 60.25 0.64305
## 618 20 7 2002 0.0 34.5 18.9 NA 26.100 NA 60.50 0.25722
## 619 22 7 2002 0.0 35.1 19.9 NA 25.900 NA 71.75 1.15749
## 620 23 7 2002 0.0 NA 18.4 NA 26.250 NA 68.50 1.67193
## 621 26 7 2002 0.0 35.3 18.9 NA 27.225 NA 60.25 1.54332
## 622 27 7 2002 0.0 34.1 16.3 NA 25.125 NA 70.00 2.31498
## 623 28 7 2002 0.0 28.8 18.4 NA 25.000 NA 70.50 2.82942
## 624 29 7 2002 0.0 32.7 17.3 NA 23.750 NA 75.00 1.80054
## 625 30 7 2002 0.0 37.5 17.3 NA 27.150 NA 59.50 1.02888
## 626 2 8 2002 0.0 23.9 18.4 NA 21.500 NA 90.50 3.60108
## 627 3 8 2002 0.0 28.0 15.8 NA NA NA NA NA NA
## 628 4 8 2002 0.0 35.5 16.5 NA 23.950 NA 77.50 0.51444
## 629 6 8 2002 0.0 36.1 17.8 NA 27.550 NA 59.25 0.77166
## 630 10 8 2002 0.0 36.7 16.3 NA 27.550 NA 52.50 0.77166
## 631 11 8 2002 0.0 36.5 16.3 NA NA NA NA NA NA
## 632 12 8 2002 0.0 37.5 16.8 NA 28.500 NA 46.75 0.51444
## 633 13 8 2002 0.0 36.9 16.3 NA 28.550 NA 43.25 1.67193
## 634 14 8 2002 0.0 36.1 16.3 NA 27.350 NA 47.50 1.28610
## 635 15 8 2002 0.0 36.9 16.2 NA 28.150 NA 48.00 0.90027
## 636 16 8 2002 0.0 37.5 17.3 NA NA NA NA NA NA
## 637 17 8 2002 0.0 37.3 18.4 NA 30.050 NA 42.00 0.77166
## 638 18 8 2002 0.0 37.1 19.4 NA 27.200 NA 51.25 0.64305
## 639 19 8 2002 0.0 36.7 19.9 NA 30.700 NA 40.00 2.05776
## 640 20 8 2002 0.0 36.9 19.7 NA 30.000 NA 43.50 1.02888
## 641 21 8 2002 0.0 34.1 19.9 NA 28.700 NA 44.25 1.28610
## 642 22 8 2002 0.0 36.3 19.9 NA 27.150 NA 58.75 2.05776
## 643 23 8 2002 0.0 36.7 18.9 NA 28.800 NA 46.25 1.15749
## 644 24 8 2002 0.0 37.5 19.4 NA 29.800 NA NA 0.90027
## 645 25 8 2002 0.0 NA 19.4 NA 30.600 NA 51.50 0.77166
## 646 26 8 2002 0.0 37.9 18.4 NA NA NA NA NA NA

```

## 647	27	8	2002	0.0	37.3	18.3	NA	29.400	NA	42.00	1.28610
## 648	30	8	2002	10.0	26.4	19.9	NA	24.825	NA	81.25	1.28610
## 649	31	8	2002	0.0	NA	19.9	NA	22.300	NA	89.75	1.41471
## 650	2	9	2002	0.0	NA	11.7	NA	NA	NA	NA	NA
## 651	4	9	2002	0.0	NA	15.5	NA	23.150	NA	47.00	0.12861
## 652	5	9	2002	0.0	37.9	18.4	NA	NA	NA	NA	NA
## 653	8	9	2002	5.0	33.7	19.9	NA	26.100	NA	76.00	0.90027
## 654	21	9	2002	0.0	NA	19.4	NA	24.400	NA	78.50	0.38583
## 655	22	9	2002	0.0	36.1	19.4	NA	NA	NA	NA	NA
## 656	25	9	2002	0.0	33.5	18.4	NA	NA	NA	NA	NA
## 657	26	9	2002	0.0	NA	17.3	NA	NA	NA	NA	NA
## 658	27	9	2002	0.0	NA	18.4	NA	NA	NA	NA	NA
## 659	29	9	2002	0.0	38.9	19.4	NA	29.500	NA	46.75	0.64305
## 660	22	2	2003	25.5	25.4	17.9	NA	NA	NA	95.00	0.45000
## 661	13	4	2003	0.0	29.0	15.4	NA	NA	NA	NA	NA
## 662	14	4	2003	0.0	31.5	16.9	NA	23.900	20.750	75.25	0.07500
## 663	15	4	2003	0.0	31.9	19.5	NA	25.850	23.100	78.75	0.22500
## 664	24	4	2003	0.0	32.5	18.9	NA	NA	NA	NA	NA
## 665	30	4	2003	0.0	34.1	18.9	NA	26.650	23.300	76.75	0.12500
## 666	1	5	2003	0.0	34.1	18.4	NA	26.100	23.100	79.50	0.15000
## 667	2	5	2003	0.0	33.7	18.9	NA	26.750	23.000	81.25	0.32500
## 668	4	5	2003	0.0	31.3	18.9	NA	23.750	21.750	85.25	0.45000
## 669	7	5	2003	0.0	27.2	14.4	NA	21.150	17.450	69.50	0.42500
## 670	8	5	2003	0.0	27.6	10.9	NA	19.500	15.700	66.25	0.47500
## 671	9	5	2003	0.0	26.8	9.4	NA	18.850	15.150	67.75	0.37500
## 672	10	5	2003	0.0	29.5	11.4	NA	NA	NA	NA	NA
## 673	11	5	2003	0.0	31.5	14.9	NA	NA	NA	NA	NA
## 674	12	5	2003	0.0	31.1	17.9	NA	24.350	21.250	78.25	0.25000
## 675	13	5	2003	0.0	30.3	16.4	NA	23.350	19.900	73.00	0.25000
## 676	14	5	2003	0.0	32.1	14.4	NA	22.450	19.850	80.25	0.27500
## 677	15	5	2003	0.0	33.1	17.9	NA	25.600	21.850	73.25	0.27500
## 678	16	5	2003	0.0	33.1	16.9	NA	25.550	21.600	72.50	0.22500
## 679	17	5	2003	0.0	32.9	17.4	NA	25.350	NA	73.50	0.25000
## 680	18	5	2003	0.0	32.9	17.4	NA	25.550	21.700	73.75	0.22500
## 681	19	5	2003	0.0	31.3	15.4	NA	24.100	20.150	70.50	0.25000
## 682	20	5	2003	0.0	32.9	14.9	NA	23.600	19.750	70.25	0.20000
## 683	21	5	2003	0.0	32.9	17.4	NA	25.100	21.250	71.75	0.25000
## 684	22	5	2003	0.0	33.7	18.9	NA	25.750	22.000	69.75	0.17500
## 685	23	5	2003	0.0	33.3	18.9	NA	25.450	22.750	78.25	0.42500
## 686	25	5	2003	6.7	27.0	17.9	NA	22.150	21.275	92.75	0.15000
## 687	26	5	2003	0.0	29.9	17.4	NA	23.375	22.050	90.00	0.20000
## 688	27	5	2003	0.0	31.3	18.4	NA	24.750	22.100	80.00	0.27500
## 689	28	5	2003	0.0	30.9	18.9	NA	24.400	21.700	79.25	0.27500
## 690	29	5	2003	0.0	30.3	17.4	NA	24.100	21.150	78.00	0.20000
## 691	30	5	2003	0.0	31.5	16.4	NA	NA	NA	NA	NA
## 692	31	5	2003	0.0	31.9	16.4	NA	24.550	20.800	73.00	NA

```

## 693   1   6 2003   0.0 31.7 17.4   NA    NA    NA    NA    NA
## 694   2   6 2003   0.0 32.7 18.4   NA 25.500 22.000  75.25 0.15000
## 695   3   6 2003   0.0 32.7 18.9   NA 26.400 22.450  71.50 0.17500
## 696   4   6 2003   0.0 28.6 19.5   NA 24.900 22.550  81.00 0.40000
## 697   5   6 2003   0.0 27.8 15.9   NA 21.475 19.950  87.75 0.37500
## 698   6   6 2003   0.0 32.5 18.4   NA 24.650 22.450  84.25 0.30000
## 699  10   6 2003   0.0 31.9 16.4   NA 24.250 21.150  77.50 0.40000
## 700  11   6 2003   0.0 32.1 17.4   NA 24.650 21.000  72.50 0.20000
## 701  12   6 2003   0.0 31.3 16.9   NA 24.300 20.850  75.50 0.27500
## 702  13   6 2003   0.0 31.3 14.4   NA 23.025 19.500  73.00 0.10000
## 703  14   6 2003   0.0 32.5 15.4   NA 24.500 20.400  71.00 0.30000
## 704  15   6 2003   0.0 32.7 14.9   NA 23.975 19.650  70.00 0.30000
## 705  16   6 2003   0.0 31.7 16.4   NA 24.275 20.050  70.00 0.22500
## 706  17   6 2003   0.0 31.7 13.9   NA 23.525 19.400  70.00 0.20000
## 707  18   6 2003   0.0 31.9 13.9   NA 22.875 18.900  71.50 0.22500
## 708  19   6 2003   0.0 31.5 13.9   NA 22.875 18.600  68.25 0.32500
## 709  20   6 2003   0.0 31.7 13.9   NA    NA    NA    NA    NA
## 710  21   6 2003   0.0 29.9 13.4   NA    NA    NA    NA    NA
## 711  22   6 2003   0.0 31.3 13.9   NA    NA    NA    NA    NA
## 712  23   6 2003   0.0 31.5 12.9   NA    NA    NA    NA    NA
## 713  24   6 2003   0.0 31.1 12.4   NA    NA    NA    NA    NA
## 714  25   6 2003   0.0 31.3 11.4   NA    NA    NA    NA    NA
## 715  26   6 2003   0.0 31.3 11.9   NA 22.675 18.250  67.75 0.35000
## 716  27   6 2003   0.0 30.9 13.4   NA 22.675 18.250  66.50 0.32500
## 717  28   6 2003   0.0 31.9 14.4   NA    NA    NA    NA    NA
## 718  29   6 2003   0.0 32.7 16.4   NA    NA    NA    NA    NA
## 719  30   6 2003   0.0 31.3 13.9   NA 23.175 18.150  63.00 0.30000
## 720  10   7 2003   0.0 27.4 17.8   NA 23.625      NA 70.50 0.90027
## 721  11   7 2003   0.0 20.7 15.8   NA 18.850      NA 82.75 2.05776
## 722  13   7 2003   0.0 30.5 13.2   NA    NA    NA    NA    NA
## 723  14   7 2003   0.0 32.1 13.7   NA    NA    NA    NA    NA
## 724  15   7 2003   0.0 32.3 13.7   NA    NA    NA    NA    NA
## 725  16   7 2003   0.0 31.9 12.7   NA 23.125      NA 54.00 0.51444
## 726  18   7 2003   0.0 32.5 14.4   NA 22.475      NA 73.50 0.12861
## 727  19   7 2003   0.0 33.5 19.4   5.7 24.825      NA 68.50 0.38583
## 728  21   7 2003   0.0 35.1 13.2   NA 24.775      NA 62.50 0.38583
## 729  22   7 2003   0.0 34.1 14.1   NA 24.875      NA 47.75 0.38583
## 730  24   7 2003   0.0 33.7 12.7   NA 26.825      NA 38.00 0.38583
## 731  25   7 2003   0.0 32.9 14.5   NA 25.275      NA 64.75 0.38583
## 732  26   7 2003   0.0 33.7 14.9   NA 24.925      NA 56.00 0.38583
## 733  27   7 2003   0.0 32.3 17.6   NA 24.125      NA 62.25 0.51444
## 734  30   7 2003   0.0 32.7 14.8   NA 25.750      NA 51.75 0.38583
## 735  31   7 2003   0.0 33.2 14.8   NA 26.750      NA 62.75 0.38583
## 736   1   8 2003   0.0 34.9 14.3   NA 24.075      NA 62.50 0.64305
## 737   2   8 2003   0.0 35.1 15.9   NA    NA    NA    NA    NA
## 738   4   8 2003   0.0 34.9 17.8   NA 27.400      NA 52.75 0.38583

```

```

## 739 5 8 2003 0.0 36.7 19.9 NA 28.500 NA 45.50 1.02888
## 740 10 8 2003 0.0 28.0 16.0 NA 23.125 NA 73.00 1.92915
## 741 11 8 2003 0.0 32.9 13.7 NA 22.675 NA 60.75 0.90027
## 742 12 8 2003 0.0 35.7 15.8 NA 25.675 NA 61.25 0.25722
## 743 13 8 2003 0.0 36.9 18.4 NA 28.950 NA 59.75 1.28610
## 744 14 8 2003 0.0 NA 17.1 NA 28.050 NA 61.50 0.64305
## 745 16 8 2003 0.0 31.9 15.8 NA NA NA NA NA NA
## 746 18 8 2003 0.0 32.5 10.3 NA 21.925 NA 62.25 0.90027
## 747 19 8 2003 0.0 NA 9.7 NA 25.225 NA 54.75 0.38583
## 748 21 8 2003 0.0 NA 13.7 NA NA NA NA NA NA
## 749 22 8 2003 0.0 NA 18.6 NA NA NA NA NA NA
## 750 24 8 2003 0.0 NA 19.7 NA NA NA NA NA NA
## 751 25 8 2003 0.0 NA 14.9 NA NA NA NA NA NA
## 752 26 8 2003 0.0 NA 11.7 NA NA NA NA NA NA
## 753 27 8 2003 0.0 NA 13.7 NA NA NA NA NA NA
## 754 29 8 2003 0.0 NA 11.7 NA NA NA NA NA NA
## 755 1 9 2003 0.0 NA 16.2 NA 27.150 NA 75.50 0.38583
## 756 3 9 2003 0.0 NA 15.9 NA NA NA NA NA NA
## 757 4 9 2003 0.0 NA 19.9 NA NA NA NA NA NA
## 758 6 9 2003 0.0 NA 18.9 NA NA NA NA NA NA
## 759 11 9 2003 0.0 NA 13.7 NA 18.325 NA 82.25 3.85830
## 760 12 9 2003 0.0 NA 14.3 NA NA NA NA NA NA
## 761 13 9 2003 0.0 NA 17.1 NA NA NA NA NA NA
## 762 15 9 2003 0.0 NA 18.9 NA 28.975 NA 86.25 0.51444
## 763 17 9 2003 0.0 NA 16.8 NA NA NA NA NA NA
## 764 22 3 2004 0.0 33.1 18.9 NA 25.800 22.500 75.00 0.60000
## 765 26 3 2004 0.0 34.1 19.5 NA 27.850 23.700 71.50 0.45000
## 766 5 5 2004 0.0 26.0 18.9 NA 23.400 22.050 88.25 0.70000
## 767 6 5 2004 0.0 23.1 17.4 NA NA NA 87.00 0.85000
## 768 7 5 2004 0.0 23.3 16.9 NA 19.525 17.950 85.25 0.85000
## 769 8 5 2004 0.0 26.0 16.9 NA 20.025 18.300 84.25 0.57500
## 770 9 5 2004 0.0 26.0 16.4 NA 20.325 18.950 87.75 0.32500
## 771 10 5 2004 0.0 29.9 17.4 NA 23.600 21.500 84.00 0.20000
## 772 11 5 2004 0.0 32.3 18.9 NA 25.350 23.000 83.00 0.50000
## 773 13 5 2004 0.0 31.1 19.5 NA 24.500 NA 93.00 0.22500
## 774 15 5 2004 0.0 22.3 17.9 NA 19.775 18.500 88.25 0.42500
## 775 16 5 2004 0.0 28.4 15.4 NA 21.025 19.000 83.50 0.72500
## 776 17 5 2004 0.0 29.7 16.9 NA 22.775 20.550 88.00 0.30000
## 777 18 5 2004 0.0 30.7 17.9 NA 24.100 21.450 79.50 0.47500
## 778 19 5 2004 0.0 31.9 17.9 NA 24.750 21.550 75.75 0.40000
## 779 20 5 2004 0.0 32.3 18.9 NA 24.600 21.900 79.50 0.37500
## 780 21 5 2004 0.0 32.4 19.5 NA 25.350 22.600 80.00 0.27500
## 781 25 5 2004 0.0 24.5 17.9 NA 22.350 NA 88.00 0.67500
## 782 26 5 2004 0.0 26.4 16.4 NA 19.525 17.500 82.25 0.75000
## 783 27 5 2004 0.0 26.8 14.9 NA 19.575 16.650 74.00 0.72500
## 784 28 5 2004 0.0 28.6 13.4 NA 20.425 17.350 74.00 0.55000

```

```

## 785 29 5 2004 0.0 32.1 14.9 NA 22.900 19.500 73.25 0.30000
## 786 30 5 2004 0.0 32.7 17.4 NA 25.350 21.750 72.25 0.25000
## 787 31 5 2004 0.0 32.5 17.4 NA 25.400 21.650 72.25 0.35000
## 788 1 6 2004 0.0 31.5 19.5 NA 25.200 21.700 73.75 0.45000
## 789 2 6 2004 0.0 29.7 18.9 NA 23.100 NA 81.25 0.57500
## 790 3 6 2004 0.0 28.0 18.4 NA 23.850 21.250 79.50 0.57500
## 791 4 6 2004 0.0 28.6 19.5 NA 22.350 19.900 80.50 0.60000
## 792 5 6 2004 0.0 29.0 14.4 NA 21.925 18.950 77.00 0.37500
## 793 6 6 2004 0.0 31.1 12.9 NA 21.925 17.550 67.25 0.50000
## 794 7 6 2004 0.0 30.7 12.9 NA 23.275 17.600 57.75 0.70000
## 795 8 6 2004 0.0 31.1 12.4 NA 22.675 17.750 61.75 0.35000
## 796 9 6 2004 0.0 32.7 16.4 NA 25.050 20.850 69.25 0.47500
## 797 10 6 2004 0.0 32.1 17.4 NA 25.750 21.300 67.00 0.45000
## 798 11 6 2004 0.0 31.9 18.4 NA 25.100 21.550 72.75 0.27500
## 799 12 6 2004 0.0 NA 15.4 NA 19.600 18.400 88.25 1.00000
## 800 13 6 2004 0.0 25.8 10.4 NA NA 14.500 73.75 0.67500
## 801 14 6 2004 0.0 29.9 11.4 NA 20.025 16.250 68.25 0.52500
## 802 15 6 2004 0.0 31.3 14.9 NA 23.025 19.150 71.00 0.42500
## 803 16 6 2004 0.0 31.7 13.9 NA 23.725 20.000 73.25 0.17500
## 804 17 6 2004 0.0 31.5 14.4 NA 23.850 18.900 63.50 0.27500
## 805 18 6 2004 0.0 31.7 13.9 NA 23.525 18.500 62.50 0.45000
## 806 19 6 2004 0.0 32.3 13.9 NA 23.225 18.100 62.50 0.30000
## 807 20 6 2004 0.0 31.7 13.9 NA 24.775 19.200 60.25 0.40000
## 808 21 6 2004 0.0 31.5 14.9 NA 23.475 19.250 67.75 0.37500
## 809 22 6 2004 0.0 32.9 16.4 NA 25.500 20.600 64.25 0.37500
## 810 23 6 2004 0.0 32.7 15.9 NA 25.575 NA NA 0.40000
## 811 24 6 2004 0.0 32.7 14.4 NA 24.175 18.650 59.75 0.27500
## 812 25 6 2004 0.0 32.5 14.9 NA 25.125 18.750 55.00 0.45000
## 813 26 6 2004 0.0 33.3 16.4 NA NA 20.200 61.00 0.35000
## 814 27 6 2004 0.0 32.9 16.9 NA 25.250 20.000 61.75 0.42500
## 815 28 6 2004 0.0 32.7 15.9 NA 24.950 20.000 64.75 0.35000
## 816 29 6 2004 0.0 33.5 16.9 NA 26.200 20.450 59.75 0.37500
## 817 30 6 2004 0.0 34.1 16.9 NA 26.550 NA NA 0.27500
## 818 1 7 2004 0.0 33.7 17.4 NA 26.350 NA 65.50 0.37500
## 819 2 7 2004 0.0 33.9 15.9 NA 26.650 20.400 57.50 0.35000
## 820 3 7 2004 0.0 33.1 14.9 NA 24.850 19.350 60.00 0.52500
## 821 4 7 2004 0.0 33.7 14.4 NA 25.050 19.700 62.50 0.27500
## 822 5 7 2004 0.0 33.7 14.9 NA 24.275 19.200 64.00 0.45000
## 823 6 7 2004 0.0 33.5 16.9 NA 25.800 19.500 56.00 0.47500
## 824 7 7 2004 0.0 34.3 13.9 NA 24.675 18.700 57.75 0.32500
## 825 8 7 2004 0.0 24.9 17.9 NA 22.300 19.550 77.25 0.40000
## 826 9 7 2004 23.4 29.3 17.9 NA 21.125 20.600 90.50 0.65000
## 827 10 7 2004 0.0 31.7 18.4 NA 23.100 22.000 90.75 0.32500
## 828 11 7 2004 15.8 23.5 17.4 NA 21.225 19.800 87.00 0.95000
## 829 12 7 2004 0.0 30.7 14.9 NA 20.275 18.450 86.25 0.42500
## 830 13 7 2004 0.0 32.7 16.4 NA 25.250 22.000 76.50 0.32500

```

```

## 831 15 7 2004 0.0 32.9 17.9 NA 26.300 21.900 68.75 0.37500
## 832 16 7 2004 0.0 33.3 18.4 NA 25.950 21.900 71.50 0.42500
## 833 17 7 2004 0.0 25.1 16.9 NA 22.375 19.950 79.50 0.90000
## 834 18 7 2004 0.0 28.2 15.9 NA 19.675 17.750 84.25 0.92500
## 835 19 7 2004 0.0 24.9 16.4 NA 19.225 17.800 87.25 0.42500
## 836 20 7 2004 0.0 29.3 16.4 NA 20.225 18.300 84.75 0.60000
## 837 21 7 2004 0.0 30.9 15.9 NA 23.225 19.300 NA 0.52500
## 838 22 7 2004 0.0 30.1 12.4 NA 23.325 17.250 NA 0.37500
## 839 23 7 2004 0.0 31.1 11.9 NA 22.225 16.800 58.00 0.27500
## 840 24 7 2004 0.0 30.3 12.4 NA 22.575 16.900 56.75 0.45000
## 841 25 7 2004 0.0 29.9 9.9 NA 21.825 16.450 58.00 0.47500
## 842 26 7 2004 0.0 30.3 10.9 NA 22.525 15.900 49.50 0.40000
## 843 27 7 2004 0.0 30.1 11.9 NA 22.525 16.600 54.25 0.55000
## 844 28 7 2004 0.0 31.1 9.9 NA 23.125 16.550 51.50 0.55000
## 845 29 7 2004 0.0 31.7 11.9 NA 23.575 16.350 46.75 0.37500
## 846 30 7 2004 0.0 23.7 16.4 NA 20.975 16.900 65.00 0.67500
## 847 31 7 2004 0.0 27.6 14.9 NA 17.000 NA 76.75 0.97500
## 848 1 8 2004 0.0 31.9 13.9 NA 21.925 17.950 70.50 0.50000
## 849 2 8 2004 0.0 34.1 12.9 NA 25.250 19.350 58.25 0.40000
## 850 3 8 2004 0.0 35.9 16.9 NA 26.700 20.250 57.00 0.50000
## 851 4 8 2004 0.0 35.9 16.4 NA 27.050 19.400 47.25 0.12500
## 852 5 8 2004 0.0 34.1 16.9 NA 27.200 19.600 49.25 0.45000
## 853 6 8 2004 0.0 32.5 13.9 NA 25.000 17.200 44.00 0.42500
## 854 7 8 2004 0.0 27.0 13.9 NA 23.600 17.600 53.25 0.87500
## 855 8 8 2004 0.0 28.6 12.4 NA 19.325 15.450 68.50 0.97500
## 856 9 8 2004 0.0 30.3 10.9 NA 21.175 15.300 53.50 0.52500
## 857 10 8 2004 0.0 33.3 13.9 NA 24.400 17.200 47.50 0.52500
## 858 11 8 2004 0.0 34.7 13.9 NA 25.400 18.000 48.00 0.37500
## 859 12 8 2004 0.0 33.7 13.9 NA 25.100 18.250 51.25 0.15000
## 860 13 8 2004 0.0 35.1 13.7 NA 25.800 17.800 45.75 0.12500
## 861 14 8 2004 0.0 33.1 13.4 NA 25.400 17.050 31.25 0.40000
## 862 15 8 2004 0.0 33.3 12.9 NA 25.700 17.400 41.00 0.52500
## 863 16 8 2004 0.0 34.9 13.9 NA 25.750 17.100 39.50 0.52500
## 864 17 8 2004 0.0 35.1 14.4 NA 26.750 17.600 39.75 0.20000
## 865 18 8 2004 0.0 35.7 13.9 NA 26.100 18.050 45.25 0.15000
## 866 19 8 2004 0.0 36.7 15.4 NA 26.300 18.150 45.50 0.27500
## 867 20 8 2004 0.0 37.1 14.9 NA 26.850 17.900 39.50 0.42500
## 868 21 8 2004 0.0 37.7 16.9 NA 27.650 19.150 45.00 0.22500
## 869 22 8 2004 0.0 33.3 15.7 NA 26.450 20.500 58.25 0.57500
## 870 23 8 2004 0.0 35.9 17.9 NA 25.900 20.100 59.75 0.42500
## 871 24 8 2004 0.0 37.3 18.9 NA 27.800 20.800 54.50 0.20000
## 872 27 8 2004 0.0 33.7 18.9 NA 26.250 21.300 64.50 0.42500
## 873 29 8 2004 0.0 37.1 19.5 NA 29.550 21.200 47.75 0.35000
## 874 30 8 2004 0.0 37.3 15.4 NA 27.150 17.000 33.00 0.45000
## 875 31 8 2004 0.0 38.5 14.9 NA 28.350 17.850 33.25 0.25000
## 876 1 9 2004 0.0 38.1 14.9 NA 28.800 18.800 34.75 0.40000

```

```

## 877   3   9 2004   0.0 37.1 17.9  NA 30.300 19.450  33.50 0.35000
## 878   4   9 2004   0.0 38.1 15.9  NA 29.100 18.650  35.25 0.25000
## 879   5   9 2004   0.0 38.7 16.9  NA 29.600 17.950  29.00 0.45000
## 880   6   9 2004   0.0 39.3 15.9  NA 29.100 17.750  30.25 0.20000
## 881   7   9 2004   0.0 38.7 16.9  NA 29.850 18.450  31.00 0.10000
## 882   8   9 2004   0.0 39.1 18.4  NA 29.950 18.900  32.00 0.17500
## 883  10   9 2004   0.0 36.1 18.9  NA 29.350 20.400  43.50 0.45000
## 884  11   9 2004   0.0 33.9 17.9  NA 26.950 20.150  52.50 0.65000
## 885  12   9 2004   0.0 24.1 15.9  NA 21.025 17.300  68.25 1.52500
## 886  13   9 2004   0.0 28.6 13.9  NA 19.275 15.000  64.25 1.15000
## 887  14   9 2004   0.0 35.1 13.9  NA 23.625 16.150  44.75 0.52500
## 888   4  10 2004   0.0 35.9 19.5  NA 26.900 22.100  67.50 0.32500
## 889   5  10 2004   0.0 37.1 19.5  NA 29.350      NA 40.75 0.37500
## 890   7  10 2004   0.0 38.9 17.9  NA 31.050 19.000  28.25 0.35000
## 891   2  11 2004   0.0 37.1 17.9  0.0 30.350 24.600  62.25 0.25000
## 892   3  11 2004  42.9 36.3 17.9  0.0 28.400 24.000  71.25 0.57500
## 893  11  11 2004  47.2     NA 19.5  0.0 22.075 21.550  95.75 0.37500
## 894  12  11 2004  7.3 29.5 19.5  0.0 23.650 21.950  87.00 0.45000
## 895  21  11 2004   0.0 33.7 19.5  0.0 26.550      NA 63.25 0.25000
## 896  22  11 2004   0.0 35.3 19.5  0.0 28.000 22.650  63.25 0.22500
## 897  13  12 2004   0.0 35.1 18.4  NA 27.500 21.350  57.25 0.25000
## 898  14   2 2005   0.0 33.5 18.9  NA 26.700 22.300  68.75 0.27500
## 899  15   2 2005   0.0 33.9 18.4  NA 27.000 22.050  64.50 0.30000
## 900  21   4 2005   0.6 32.5 19.5  NA 25.150 22.700  82.50 0.42500
## 901  26   4 2005   5.5 21.9 15.9  NA 19.850 18.700  89.00 1.05000
## 902  27   4 2005   0.0 24.3 13.9  NA 17.675 16.150  85.50 0.77500
## 903  28   4 2005   0.0 25.6 15.4  NA 19.925 18.300  85.50 0.72500
## 904  29   4 2005   0.0 28.4 16.4  NA 21.500 19.900  87.00 0.52500
## 905  30   4 2005   0.0 28.6 18.4  NA 23.250 20.750  79.50 0.62500
## 906   1   5 2005   0.0 30.1 17.4  NA 23.000 20.450  79.75 0.32500
## 907   2   5 2005   0.0 31.7 16.4  NA 24.950 20.900  69.50 0.42500
## 908   3   5 2005   0.0 32.7 17.4  NA 25.700 21.550  70.00 0.32500
## 909   4   5 2005   0.0 33.7 17.4  NA 26.050 21.650  68.25 0.17500
## 910   5   5 2005   0.0 32.9 19.5  NA 28.000 22.200  59.75 0.47500
## 911   6   5 2005   0.0 32.5 19.5  NA 25.700 20.700  64.00 0.17500
## 912   7   5 2005   0.0 32.1 14.9  NA 23.950 19.050  64.00 0.52500
## 913   8   5 2005   0.0 30.5 17.9  0.0 23.575 19.550      NA 0.37500
## 914   9   5 2005   0.0 31.3 17.9  0.0 24.850 21.500  74.50 0.22500
## 915  11   5 2005   0.0 34.5 19.5  0.0 27.800 23.100  67.00 0.35000
## 916  12   5 2005   0.0 34.5 18.9  0.0 27.600 23.000  68.75 0.25000
## 917  13   5 2005   0.0 34.5 18.4  0.0 26.900 22.150  68.00 0.50000
## 918  14   5 2005   0.0 34.3 17.9  0.0 27.550 21.800  61.50 0.32500
## 919  15   5 2005   0.0 34.5 17.9  0.0 27.850 21.750  57.75 0.42500
## 920  16   5 2005   0.0 34.1 18.9  0.0 27.950 22.400  61.25 0.25000
## 921  17   5 2005   0.0 34.1 17.9  0.0 26.950 21.650  64.25 0.25000
## 922  18   5 2005   0.0 33.7 16.4  0.0      NA 20.600      NA 0.22500

```

```

## 923 19 5 2005 0.0 34.1 17.9 0.0 25.950 21.250 67.25 0.15000
## 924 20 5 2005 4.6 34.1 17.4 0.0 25.700 21.550 71.25 0.32500
## 925 21 5 2005 0.0 34.9 17.4 0.0 26.650 22.000 67.75 0.17500
## 926 26 5 2005 11.4 29.3 18.9 0.0 22.200 20.700 88.50 0.65000
## 927 27 5 2005 0.0 31.5 18.4 0.0 24.850 22.200 79.25 0.27500
## 928 28 5 2005 0.0 33.3 18.9 0.0 26.150 22.500 73.50 0.32500
## 929 29 5 2005 0.0 33.1 19.5 0.0 26.900 22.500 69.00 0.35000
## 930 30 5 2005 0.0 33.1 18.9 0.0 27.150 22.100 64.00 0.47500
## 931 31 5 2005 0.0 32.7 17.9 0.0 26.550 21.500 66.00 0.47500
## 932 2 6 2005 0.0 33.1 17.4 0.0 25.650 21.450 70.00 0.35000
## 933 3 6 2005 0.0 32.7 17.4 0.0 26.550 21.250 62.50 0.32500
## 934 4 6 2005 0.0 32.5 16.9 0.0 26.000 21.300 66.75 0.22500
## 935 5 6 2005 0.0 32.9 15.9 0.0 26.850 20.350 54.75 0.50000
## 936 6 6 2005 0.0 33.1 15.9 NA 24.750 20.050 66.25 0.17500
## 937 7 6 2005 0.0 32.9 15.9 0.0 25.100 19.950 63.25 0.17500
## 938 8 6 2005 0.0 32.9 16.9 0.0 24.550 NA 60.75 0.22500
## 939 9 6 2005 0.0 33.3 14.9 0.0 24.600 18.900 59.25 0.15000
## 940 10 6 2005 0.0 33.1 14.9 0.0 24.800 19.400 60.75 0.22500
## 941 11 6 2005 0.0 33.5 16.9 0.0 26.700 20.750 58.75 0.20000
## 942 12 6 2005 0.0 33.7 16.9 0.0 27.300 NA 51.50 0.20000
## 943 13 6 2005 0.0 33.5 15.4 0.0 26.050 20.050 57.75 0.30000
## 944 14 6 2005 0.0 34.7 16.9 0.0 NA 20.650 57.00 0.30000
## 945 15 6 2005 0.0 34.1 16.4 0.0 NA 20.700 56.50 NA
## 946 16 6 2005 0.0 33.7 15.9 0.0 25.350 19.850 60.25 0.17500
## 947 17 6 2005 0.0 34.9 15.4 0.0 25.650 19.550 58.00 0.17500
## 948 18 6 2005 5.0 32.7 17.9 0.0 25.250 NA NA 0.30000
## 949 19 6 2005 0.0 33.5 18.4 0.0 26.000 22.600 75.75 0.17500
## 950 20 6 2005 0.0 33.9 18.9 0.0 25.650 22.250 74.50 0.30000
## 951 21 6 2005 0.0 28.4 18.8 0.0 23.650 21.100 79.75 0.67500
## 952 22 6 2005 0.0 29.0 18.9 0.0 22.700 20.300 NA 0.50000
## 953 23 6 2005 0.0 30.1 16.9 0.0 22.575 20.150 80.25 0.50000
## 954 24 6 2005 0.0 32.9 16.9 0.0 24.550 20.750 73.50 0.10000
## 955 25 6 2005 0.0 32.3 14.9 0.0 24.700 18.700 56.50 0.37500
## 956 26 6 2005 0.0 32.1 11.9 0.0 24.075 NA NA 0.20000
## 957 27 6 2005 0.0 32.7 15.4 0.0 23.625 18.350 59.25 0.40000
## 958 28 6 2005 0.5 31.9 18.9 0.0 24.550 20.950 73.00 0.30000
## 959 29 6 2005 23.2 33.5 19.5 0.0 25.450 22.050 76.50 0.22500
## 960 30 6 2005 0.0 34.1 18.4 0.0 27.000 22.200 67.50 0.35000
## 961 1 7 2005 0.0 32.5 16.9 0.0 26.300 20.350 57.75 0.47500
## 962 2 7 2005 0.0 32.9 14.4 0.0 25.050 19.550 60.75 0.07500
## 963 3 7 2005 0.0 33.3 15.4 NA 25.700 19.450 55.50 0.22500
## 964 4 7 2005 0.0 33.7 15.9 0.0 25.150 20.250 64.50 0.20000
## 965 5 7 2005 NA 32.9 17.4 0.0 NA 20.550 NA 0.35000
## 966 6 7 2005 0.0 25.8 18.4 NA 22.225 20.000 81.50 1.12500
## 967 7 7 2005 0.0 26.6 11.4 NA 17.150 15.650 NA 1.05000
## 968 8 7 2005 0.0 30.3 12.9 0.0 21.575 17.050 63.00 0.70000

```

```

## 969   9   7 2005   0.0 31.9 16.4   NA 24.325 19.000   61.00 0.55000
## 970  10   7 2005   0.0 29.7 15.4   NA 24.100 17.750   52.75 0.50000
## 971  11   7 2005   0.0 30.1 13.9   NA 23.625 16.150   44.00 0.35000
## 972  12   7 2005   0.0 30.3  9.9   NA 21.425 14.650   47.25 0.27500
## 973  13   7 2005   0.0 32.1 10.9   NA 22.575 16.300   51.75 0.20000
## 974  15   7 2005   0.0 33.5 15.4   NA 26.850 18.650   43.00 0.42500
## 975  16   7 2005   0.0 34.1 16.4   NA 26.750 19.450   49.75 0.50000
## 976  17   7 2005   0.0 28.6 17.4   NA 24.650 19.100   57.75 0.62500
## 977  18   7 2005   0.0 20.7 13.4   NA 18.250 15.550   74.25 1.30000
## 978  19   7 2005   0.0 25.2 11.9   NA 17.025 14.100   72.50 1.22500
## 979  20   7 2005   0.0 31.5 10.9   NA 20.325 15.800   62.50 0.57500
## 980  21   7 2005   0.0 34.9 15.9   NA 21.150 19.700      NA 0.30000
## 981  22   7 2005   0.0    NA 17.9   NA 26.950 20.150   52.50 0.37500
## 982  23   7 2005   0.0    NA 17.4  0.0 26.200 20.000   56.00 0.62500
## 983  24   7 2005   0.0    NA 14.9  0.0 22.925 18.600   66.75 0.65000
## 984  25   7 2005   0.0    NA 14.4  0.0 23.025 18.050   61.50 0.45000
## 985  26   7 2005   0.0    NA 14.9  0.0 23.975      NA 57.25 0.45000
## 986  27   7 2005   0.0    NA 16.4  0.0 26.250 20.000   56.50 0.22500
## 987  28   7 2005   0.0    NA 17.4  0.0 27.100 20.400   54.00 0.20000
## 988  29   7 2005   0.0    NA 17.4  0.0 27.450 19.300   46.50 0.25000
## 989  30   7 2005   0.0    NA 15.4  0.0 27.800 18.650   40.50 0.30000
## 990  31   7 2005   0.0    NA 16.4  0.0 27.800 18.450   37.75 0.20000
## 991   1   8 2005   0.0    NA 15.4  0.0 26.950 18.500   43.25 0.20000
## 992   2   8 2005   0.0    NA 15.9  0.0 27.250 18.250   40.00 0.32500
## 993   3   8 2005   0.0    NA 15.9  0.0 27.400 18.450   40.00 0.32500
## 994   4   8 2005   0.0    NA 15.9  0.0 27.450 18.150   38.50 0.37500
## 995   5   8 2005   0.0    NA 16.4  0.0 28.050 18.400   37.00 0.27500
## 996   6   8 2005   0.0    NA 16.9  0.0 29.000 19.050      NA 0.32500
## 997   7   8 2005   0.0    NA 14.4  0.0 27.050 17.900   38.50 0.27500
## 998   8   8 2005   0.0    NA 15.9  0.0 24.200      NA 46.50 0.57500
## 999   9   8 2005   0.0    NA 12.9  0.0 16.325 13.350   71.25 1.10000
## 1000  10   8 2005   0.0    NA  8.9  0.0 18.125 12.325   49.25 0.60000
## 1001  11   8 2005   0.0    NA  9.9  0.0 19.925 13.250   44.75 0.52500
## 1002  12   8 2005   0.0    NA 10.9  0.0 21.775 15.100   48.00 0.32500
## 1003  13   8 2005   0.0    NA 11.4  0.0 23.925 15.850   42.25 0.37500
## 1004  14   8 2005   0.0    NA 11.9  0.0 25.225 16.600   40.50 0.12500
## 1005  15   8 2005   0.0    NA 14.4  0.0 27.950 17.000   29.75 0.15000
## 1006  16   8 2005   0.0    NA 16.4  0.0 28.250 18.000      NA 0.35000
## 1007  17   8 2005   0.0    NA 18.9  0.0 30.500 19.250      NA 0.35000
## 1008  19   8 2005   0.0    NA 18.4  0.0 27.300 20.250   40.00 0.35000
## 1009  20   8 2005   0.0    NA 19.5  0.0 29.150 19.750   39.00 0.25000
## 1010  21   8 2005   0.0    NA 17.9  0.0 29.600 19.050   34.00 0.45000
## 1011  23   8 2005   0.0    NA 18.9  0.0 29.550 19.200   35.75 0.27500
## 1012  25   8 2005   0.0    NA 19.5  0.0 27.500 21.150   56.00 0.60000
## 1013  28   8 2005   0.0    NA 19.5  0.0 30.950 20.650   36.75 0.20000
## 1014   1   9 2005   0.0    NA 15.9  0.0 21.825 18.200   69.25 1.53249

```

```

## 1015 2 9 2005 0.0 NA 11.9 0.0 21.150 14.700 65.00 1.05000
## 1016 3 9 2005 0.0 NA 14.4 0.0 23.350 17.200 54.50 0.30000
## 1017 4 9 2005 0.0 NA 19.5 0.0 28.900 21.100 50.50 0.17500
## 1018 7 9 2005 0.0 NA 18.9 0.0 29.100 21.600 29.75 0.20000
## 1019 12 9 2005 0.0 NA 14.4 0.0 20.100 18.750 87.50 1.10000
## 1020 13 9 2005 0.0 NA 13.9 0.0 16.000 14.600 85.50 1.45000
## 1021 14 9 2005 0.0 27.3 13.9 0.0 17.825 14.900 73.00 1.17500
## 1022 15 9 2005 0.0 34.7 14.4 0.0 24.150 19.150 63.25 0.52500
## 1023 16 9 2005 0.0 31.9 16.4 0.0 24.900 19.400 59.00 0.47500
## 1024 17 9 2005 0.0 30.9 16.9 0.0 24.750 18.600 53.50 0.57500
## 1025 18 9 2005 0.0 33.3 16.9 0.0 25.550 18.450 48.25 0.37500
## 1026 19 9 2005 0.0 35.9 17.9 0.0 27.100 19.600 48.00 0.27500
## 1027 6 10 2005 0.0 32.9 19.9 0.0 25.700 22.450 75.75 0.50000
## 1028 5 11 2005 3.9 34.1 19.9 0.0 26.100 23.200 79.50 0.40000
## 1029 7 11 2005 0.0 33.3 19.9 0.0 28.050 24.450 73.25 0.20000
## 1030 8 11 2005 25.5 33.9 19.9 0.0 25.850 23.800 NA 0.35000
## 1031 7 12 2005 51.8 NA 19.9 NA 22.950 21.750 90.00 0.30000
## 1032 13 1 2006 3.9 NA 19.9 0.0 22.950 NA 96.25 0.32500
## 1033 14 1 2006 1.4 NA 19.9 NA 25.500 23.650 86.75 0.20000
## 1034 13 2 2006 0.0 NA 19.9 0.0 26.850 23.250 75.25 0.17500
## 1035 18 4 2006 0.0 28.0 19.9 0.0 22.600 21.200 NA 0.57500
## 1036 19 4 2006 0.0 31.9 19.4 0.0 23.850 22.100 NA 0.25000
## 1037 20 4 2006 0.0 32.7 18.9 0.0 26.400 23.800 80.75 0.40000
## 1038 25 4 2006 0.0 33.7 19.9 0.0 27.600 24.600 78.75 0.17500
## 1039 26 4 2006 0.0 34.1 19.9 0.0 26.700 22.950 73.50 0.22500
## 1040 27 4 2006 0.0 33.5 18.9 0.0 26.950 23.600 78.00 0.22500
## 1041 28 4 2006 0.0 33.7 19.9 0.0 27.300 23.000 70.00 0.20000
## 1042 29 4 2006 0.0 33.5 17.8 0.0 25.550 21.950 74.50 0.20000
## 1043 30 4 2006 0.0 35.3 17.3 0.0 25.900 NA 73.25 0.12500
## 1044 1 5 2006 0.0 32.5 15.8 NA 25.300 NA NA 0.27500
## 1045 2 5 2006 0.0 29.9 19.9 0.0 24.925 21.150 70.00 0.52500
## 1046 3 5 2006 0.0 29.0 13.2 0.0 21.725 17.950 68.50 0.55000
## 1047 4 5 2006 0.0 29.1 10.7 0.0 20.475 16.100 63.75 0.47500
## 1048 5 5 2006 0.0 29.9 10.2 0.0 21.175 16.850 65.50 0.37500
## 1049 6 5 2006 0.0 30.5 13.2 0.0 22.625 18.350 66.50 0.27500
## 1050 7 5 2006 0.0 30.7 14.8 0.0 23.350 18.850 64.50 0.37500
## 1051 8 5 2006 0.0 29.1 15.8 0.0 23.000 18.600 65.00 0.27500
## 1052 9 5 2006 0.0 29.1 15.3 0.0 22.450 18.050 64.75 0.45000
## 1053 10 5 2006 0.0 28.4 13.7 0.0 21.575 17.450 66.00 0.42500
## 1054 11 5 2006 0.0 28.0 12.2 0.0 20.725 16.750 67.25 0.32500
## 1055 12 5 2006 0.0 27.8 9.7 0.0 19.675 15.450 65.00 0.55000
## 1056 13 5 2006 0.0 28.6 9.7 0.0 20.075 15.500 62.00 0.37500
## 1057 14 5 2006 0.0 30.7 10.7 0.0 21.025 17.450 72.25 0.27500
## 1058 15 5 2006 0.0 31.7 13.2 0.0 22.600 18.600 68.75 0.30000
## 1059 16 5 2006 0.0 33.9 17.3 0.0 24.950 21.400 73.75 0.22500
## 1060 17 5 2006 0.0 33.9 16.8 0.0 25.850 22.000 73.25 0.20000

```

```

## 1061 18 5 2006 0.0 33.9 13.7 NA 24.950 NA 62.25 0.15000
## 1062 19 5 2006 0.0 32.7 12.7 0.0 22.675 18.750 71.50 0.15000
## 1063 20 5 2006 15.2 26.0 17.3 0.0 22.025 20.450 89.25 0.35000
## 1064 21 5 2006 2.1 25.2 17.8 0.0 20.925 20.450 95.50 0.30000
## 1065 22 5 2006 1.6 25.1 18.4 0.0 21.175 20.100 90.50 0.47500
## 1066 23 5 2006 0.0 25.6 17.8 0.0 21.250 19.750 87.00 0.60000
## 1067 24 5 2006 0.0 27.8 16.8 0.0 21.575 19.800 85.25 0.42500
## 1068 25 5 2006 0.0 31.7 17.8 0.0 23.175 21.100 84.25 0.27500
## 1069 26 5 2006 0.0 32.9 18.4 0.0 25.000 22.050 78.75 0.20000
## 1070 27 5 2006 0.0 32.5 17.3 0.0 25.250 22.250 78.25 0.12500
## 1071 28 5 2006 0.0 33.7 16.8 0.0 25.100 21.450 74.25 0.15000
## 1072 29 5 2006 0.0 32.7 15.8 0.0 24.500 20.900 73.75 0.17500
## 1073 30 5 2006 0.0 32.5 15.8 0.0 24.750 21.250 74.50 0.12500
## 1074 31 5 2006 0.0 33.9 18.4 0.0 26.000 22.450 73.75 0.15000
## 1075 1 6 2006 0.0 32.5 17.8 0.0 25.550 22.000 74.25 0.22500
## 1076 2 6 2006 0.0 33.9 17.8 0.0 25.700 NA 72.75 0.27500
## 1077 3 6 2006 0.0 32.7 16.3 0.0 25.550 19.850 59.25 0.25000
## 1078 4 6 2006 0.0 32.5 15.8 0.0 24.050 19.250 64.25 0.15000
## 1079 5 6 2006 0.0 32.1 15.3 0.0 24.450 19.850 66.00 0.12500
## 1080 6 6 2006 0.0 32.5 15.8 0.0 24.050 20.300 72.50 0.22500
## 1081 7 6 2006 0.0 32.3 16.3 0.0 25.500 20.450 63.25 0.32500
## 1082 8 6 2006 0.0 32.7 15.8 0.0 25.300 20.100 62.25 0.25000
## 1083 9 6 2006 0.0 32.5 14.8 0.0 25.100 20.350 65.50 0.22500
## 1084 10 6 2006 0.0 32.7 15.8 0.0 25.200 20.550 66.50 0.25000
## 1085 11 6 2006 0.0 30.5 14.8 0.0 24.450 20.400 68.75 0.07500
## 1086 12 6 2006 0.0 31.7 18.9 0.0 25.000 21.100 71.50 0.40000
## 1087 13 6 2006 0.0 32.9 15.3 0.0 24.350 20.150 71.25 0.20000
## 1088 14 6 2006 0.0 32.5 15.3 0.0 24.300 19.850 67.50 0.12500
## 1089 15 6 2006 0.0 32.1 15.3 0.0 24.050 19.700 65.25 0.20000
## 1090 16 6 2006 0.0 32.5 15.8 0.0 23.900 19.250 66.00 0.10000
## 1091 17 6 2006 0.0 32.5 18.4 0.0 25.100 19.850 62.25 0.07500
## 1092 18 6 2006 0.0 32.9 15.3 0.0 25.000 19.100 58.75 0.27500
## 1093 19 6 2006 0.0 33.5 15.8 0.0 24.750 19.500 61.75 0.17500
## 1094 20 6 2006 0.0 31.5 17.8 0.0 25.300 20.150 NA 0.32500
## 1095 21 6 2006 0.0 32.1 15.8 0.0 25.900 19.950 57.00 0.30000
## 1096 22 6 2006 0.0 32.5 14.8 0.0 24.650 19.650 63.75 0.12500
## 1097 23 6 2006 0.0 31.7 13.7 0.0 23.650 19.200 67.00 0.27500
## 1098 24 6 2006 0.0 NA 13.7 0.0 23.350 18.700 65.50 0.22500
## 1099 25 6 2006 0.0 32.9 14.8 0.0 26.050 19.500 53.50 0.35000
## 1100 26 6 2006 0.0 32.9 14.3 0.0 25.150 20.150 63.00 0.37500
## 1101 27 6 2006 0.0 30.5 18.4 0.0 24.850 21.300 72.50 0.22500
## 1102 28 6 2006 0.0 31.7 14.8 0.0 24.450 19.550 63.00 0.32500
## 1103 29 6 2006 0.0 32.7 15.8 0.0 24.850 19.650 62.50 0.22500
## 1104 30 6 2006 0.0 33.5 15.8 0.0 24.850 19.750 63.50 0.27500
## 1105 1 7 2006 0.0 34.5 16.3 0.0 25.900 20.400 61.50 0.27500
## 1106 2 7 2006 2.7 22.7 17.3 0.0 21.625 19.750 84.50 0.20000

```

```

## 1107 3 7 2006 2.3 28.0 16.8 0.0 21.725 20.200 87.00 0.30000
## 1108 4 7 2006 0.0 31.1 14.8 0.0 22.725 19.650 77.75 0.27500
## 1109 5 7 2006 0.0 31.5 14.3 0.0 23.450 19.300 69.25 0.10000
## 1110 6 7 2006 0.0 32.5 13.7 0.0 23.600 18.700 64.25 0.32500
## 1111 7 7 2006 0.0 33.3 14.3 0.0 23.850 19.250 66.50 0.50000
## 1112 8 7 2006 0.0 33.5 16.8 0.0 26.050 20.450 60.25 0.35000
## 1113 9 7 2006 0.0 32.9 18.4 0.0 26.750 21.400 61.25 0.27500
## 1114 10 7 2006 0.0 34.5 17.3 0.0 26.400 NA 69.50 0.15000
## 1115 31 7 2006 0.0 28.6 11.2 0.0 23.225 NA 72.25 0.32500
## 1116 1 8 2006 0.0 30.7 11.7 0.0 21.675 16.700 60.00 0.70000
## 1117 2 8 2006 0.0 32.9 14.3 0.0 24.300 18.350 56.00 0.42500
## 1118 3 8 2006 0.0 35.7 15.8 0.0 26.200 19.400 53.25 0.35000
## 1119 4 8 2006 0.0 37.1 16.3 0.0 27.850 20.100 49.00 0.32500
## 1120 5 8 2006 0.0 35.9 17.8 0.0 29.300 21.350 47.50 0.17500
## 1121 6 8 2006 0.0 35.9 18.1 0.0 28.900 19.750 41.00 0.40000
## 1122 7 8 2006 0.0 35.7 15.8 0.0 28.600 19.500 41.00 0.30000
## 1123 13 8 2006 0.0 37.9 19.4 0.0 30.200 20.250 38.75 0.32500
## 1124 14 8 2006 0.0 37.9 17.3 0.0 29.650 19.550 37.25 0.25000
## 1125 15 8 2006 0.0 38.7 17.3 0.0 29.900 20.150 39.25 0.17500
## 1126 17 8 2006 0.0 35.9 18.9 0.0 28.750 20.650 47.50 0.32500
## 1127 18 8 2006 0.0 30.1 16.8 0.0 25.700 20.600 62.00 0.52500
## 1128 19 8 2006 0.0 32.5 16.3 0.0 24.550 19.750 64.25 0.60000
## 1129 20 8 2006 0.0 34.1 17.8 0.0 26.050 20.250 59.25 0.42500
## 1130 21 8 2006 0.0 NA 16.8 0.0 26.250 19.000 49.00 0.37500
## 1131 22 8 2006 0.0 NA 12.7 0.0 23.900 15.600 40.00 0.25000
## 1132 23 8 2006 0.0 NA 13.2 0.0 25.450 15.800 33.25 0.15000
## 1133 24 8 2006 0.0 NA 15.8 0.0 28.150 17.450 31.25 0.35000
## 1134 25 8 2006 0.0 NA 17.3 0.0 27.900 20.050 48.50 0.25000
## 1135 28 8 2006 4.6 NA 18.9 0.0 24.200 22.225 87.00 0.12500
## 1136 30 8 2006 0.0 NA 16.8 0.0 26.650 20.300 56.75 0.07500
## 1137 2 9 2006 0.0 NA 18.9 0.0 24.800 21.850 76.75 0.45000
## 1138 3 9 2006 4.0 NA 17.3 0.0 23.700 20.350 74.00 0.50000
## 1139 4 9 2006 0.0 NA 18.4 0.0 22.900 18.950 68.00 0.95000
## 1140 5 9 2006 0.0 NA 12.7 0.0 19.775 NA 39.75 0.67500
## 1141 6 9 2006 0.0 NA 11.2 0.0 21.525 NA 34.25 0.35000
## 1142 7 9 2006 0.0 NA 14.8 0.0 26.100 17.250 39.25 0.22500
## 1143 9 9 2006 NA NA 19.4 0.0 23.250 21.250 83.50 0.17500
## 1144 10 9 2006 0.0 NA 19.9 0.0 23.975 21.350 80.75 0.27500
## 1145 11 9 2006 0.0 NA 19.9 0.0 26.900 21.800 64.75 0.30000
## 1146 12 9 2006 0.0 NA 18.4 0.0 29.200 21.350 50.00 0.10000
## 1147 13 9 2006 0.0 38.2 18.9 0.0 29.350 NA 46.50 0.22500
## 1148 11 11 2006 0.0 31.1 18.4 0.0 25.150 21.050 69.75 0.40000
## 1149 12 11 2006 0.0 32.3 16.3 0.0 26.100 19.400 51.75 0.32500
## 1150 13 11 2006 0.0 33.7 16.8 0.0 26.450 20.450 56.25 0.30000
## 1151 14 11 2006 0.0 34.1 17.8 0.0 27.200 20.300 52.75 0.20000
## 1152 6 3 2007 0.0 35.3 19.9 0.0 27.200 23.450 74.25 0.12500

```

```

## 1153 9 3 2007 1.0 33.3 19.9 0.0 NA NA NA NA
## 1154 26 3 2007 0.0 34.7 19.9 0.0 27.900 24.450 76.00 0.12500
## 1155 10 4 2007 0.0 32.5 19.9 0.0 25.650 23.050 80.50 0.22500
## 1156 17 4 2007 0.0 35.7 19.9 NA NA NA NA NA
## 1157 21 4 2007 5.4 33.3 19.9 NA NA NA NA NA
## 1158 22 4 2007 0.0 33.9 19.9 0.0 NA NA NA NA
## 1159 28 4 2007 0.0 31.7 19.4 0.0 NA NA NA NA
## 1160 30 4 2007 0.0 34.3 19.9 0.0 NA NA NA NA
## 1161 3 5 2007 0.0 33.7 17.3 0.0 NA NA NA NA
## 1162 4 5 2007 0.0 33.5 15.8 0.0 NA NA NA NA
## 1163 5 5 2007 0.0 33.5 19.9 0.0 NA NA NA NA
## 1164 6 5 2007 0.0 33.7 18.4 0.0 NA NA NA NA
## 1165 7 5 2007 0.0 33.9 19.4 0.0 NA NA NA NA
## 1166 8 5 2007 0.0 30.7 19.4 0.0 NA NA NA NA
## 1167 9 5 2007 0.0 NA 14.8 0.0 NA NA NA NA
## 1168 10 5 2007 0.0 27.8 10.2 0.0 NA NA NA NA
## 1169 11 5 2007 0.0 31.3 14.8 0.0 NA NA NA NA
## 1170 12 5 2007 0.0 33.3 18.4 0.0 NA NA NA NA
## 1171 13 5 2007 0.0 33.1 18.4 0.0 NA NA NA NA
## 1172 14 5 2007 0.0 33.8 19.9 NA NA NA NA NA
## 1173 15 5 2007 0.0 34.3 19.9 0.0 NA NA NA NA
## 1174 17 5 2007 0.0 33.9 16.8 0.0 NA NA NA NA
## 1175 18 5 2007 0.0 34.3 19.4 0.0 NA NA NA NA
## 1176 20 5 2007 0.0 31.5 17.8 NA NA NA NA NA
## 1177 21 5 2007 0.0 34.3 18.4 0.0 NA NA NA NA
## 1178 23 5 2007 19.2 23.1 16.3 0.0 NA NA NA NA
## 1179 24 5 2007 8.4 22.3 13.2 0.0 NA NA NA NA
## 1180 25 5 2007 0.0 24.5 7.7 0.0 NA NA NA NA
## 1181 26 5 2007 0.0 29.1 10.2 0.0 NA NA NA NA
## 1182 27 5 2007 0.0 31.3 14.8 0.0 NA NA NA NA
## 1183 28 5 2007 0.0 30.9 17.8 0.0 NA NA NA NA
## 1184 29 5 2007 0.0 25.4 16.3 0.0 NA NA NA NA
## 1185 30 5 2007 0.0 26.6 11.2 0.0 NA NA NA NA
## 1186 31 5 2007 0.0 29.9 11.2 0.0 NA NA NA NA
## 1187 1 6 2007 0.0 33.9 14.8 NA NA NA NA NA
## 1188 2 6 2007 0.0 29.9 19.4 0.0 NA NA NA NA
## 1189 3 6 2007 0.0 26.8 17.3 0.0 NA NA NA NA
## 1190 4 6 2007 0.0 27.2 14.8 NA NA NA NA NA
## 1191 5 6 2007 0.0 31.3 13.2 NA NA NA NA NA
## 1192 6 6 2007 0.0 33.3 15.8 0.0 NA NA NA NA
## 1193 7 6 2007 0.0 33.5 15.3 0.0 NA NA NA NA
## 1194 8 6 2007 0.0 33.5 16.3 NA NA NA NA NA
## 1195 9 6 2007 0.0 33.9 16.3 NA NA NA NA NA
## 1196 10 6 2007 0.0 32.9 13.7 0.0 NA NA NA NA
## 1197 11 6 2007 0.0 33.1 12.2 NA NA NA NA NA
## 1198 15 6 2007 0.0 34.7 11.7 0.0 NA NA NA NA

```

```

## 1199 16 6 2007 0.0 32.1 18.4 0.0 NA NA NA NA
## 1200 17 6 2007 0.0 31.3 18.4 0.0 NA NA NA NA
## 1201 18 6 2007 0.0 34.5 15.3 0.0 NA NA NA NA
## 1202 19 6 2007 0.0 34.5 15.8 0.0 NA NA NA NA
## 1203 20 6 2007 0.0 34.1 14.8 0.0 NA NA NA NA
## 1204 21 6 2007 0.0 34.1 14.3 0.0 NA NA NA NA
## 1205 22 6 2007 0.0 33.5 13.2 0.0 NA NA NA NA
## 1206 23 6 2007 0.0 33.3 12.7 0.0 NA NA NA NA
## 1207 24 6 2007 0.0 32.1 14.3 0.0 NA NA NA NA
## 1208 25 6 2007 0.0 32.1 15.8 0.0 NA NA NA NA
## 1209 26 6 2007 0.0 30.1 15.3 0.0 NA NA NA NA
## 1210 27 6 2007 0.0 32.3 14.8 0.0 NA NA NA NA
## 1211 28 6 2007 0.0 32.5 15.8 0.0 NA NA NA NA
## 1212 29 6 2007 0.0 33.5 16.8 0.0 NA NA NA NA
## 1213 30 6 2007 0.0 33.5 16.8 0.0 NA NA NA NA
## 1214 22 7 2007 0.0 35.9 17.8 NA NA NA NA NA
## 1215 24 7 2007 18.9 33.7 18.9 NA NA NA NA NA
## 1216 25 7 2007 12.1 22.1 15.3 NA NA NA NA NA
## 1217 26 7 2007 2.4 23.1 11.2 0.0 NA NA NA NA
## 1218 27 7 2007 0.0 25.1 10.7 NA NA NA NA NA
## 1219 28 7 2007 0.0 30.9 12.2 NA NA NA NA NA
## 1220 29 7 2007 0.0 30.9 12.7 NA NA NA NA NA
## 1221 30 7 2007 0.0 33.3 12.2 NA NA NA NA NA
## 1222 31 7 2007 0.0 24.7 19.7 NA NA NA NA NA
## 1223 1 8 2007 0.0 34.3 17.3 0.0 NA NA NA NA
## 1224 2 8 2007 0.0 33.7 13.2 0.0 NA NA NA NA
## 1225 3 8 2007 0.0 33.7 11.2 0.0 NA NA NA NA
## 1226 4 8 2007 0.0 34.5 13.2 0.0 NA NA NA NA
## 1227 5 8 2007 0.0 28.8 10.2 0.0 NA NA NA NA
## 1228 6 8 2007 0.0 34.5 11.7 0.0 NA NA NA NA
## 1229 7 8 2007 0.0 33.6 15.3 NA NA NA NA NA
## 1230 8 8 2007 0.0 35.7 14.3 0.0 NA NA NA NA
## 1231 9 8 2007 0.0 35.9 14.8 0.0 NA NA NA NA
## 1232 10 8 2007 0.0 35.3 14.3 0.0 NA NA NA NA
## 1233 11 8 2007 0.0 34.1 14.8 0.0 NA NA NA NA
## 1234 12 8 2007 0.0 34.5 13.2 0.0 NA NA NA NA
## 1235 13 8 2007 0.0 36.3 14.3 0.0 NA NA NA NA
## 1236 14 8 2007 0.0 35.5 16.3 0.0 NA NA NA NA
## 1237 15 8 2007 0.0 34.7 14.8 0.0 NA NA NA NA
## 1238 16 8 2007 0.0 34.3 12.2 0.0 NA NA NA NA
## 1239 17 8 2007 0.0 34.7 12.7 0.0 NA NA NA NA
## 1240 18 8 2007 0.0 31.3 14.3 0.0 NA NA NA NA
## 1241 19 8 2007 0.0 32.9 14.3 0.0 NA NA NA NA
## 1242 20 8 2007 0.0 28.8 12.7 0.0 NA NA NA NA
## 1243 21 8 2007 0.0 35.5 12.2 0.0 NA NA NA NA
## 1244 22 8 2007 0.0 36.9 14.8 0.0 NA NA NA NA

```

```

## 1245 23 8 2007 0.0 37.5 14.8 0.0 NA NA NA NA
## 1246 24 8 2007 0.0 37.5 15.8 0.0 NA NA NA NA
## 1247 25 8 2007 0.0 38.1 14.3 0.0 NA NA NA NA
## 1248 26 8 2007 0.0 35.9 16.3 0.0 NA NA NA NA
## 1249 27 8 2007 0.0 24.9 15.3 0.0 NA NA NA NA
## 1250 28 8 2007 0.0 29.0 12.2 0.0 NA NA NA NA
## 1251 29 8 2007 0.0 32.5 14.8 0.0 NA NA NA NA
## 1252 30 8 2007 0.0 34.7 16.3 0.0 NA NA NA NA
## 1253 31 8 2007 0.0 36.5 17.8 0.0 NA NA NA NA
## 1254 1 9 2007 0.0 35.7 14.3 0.0 NA NA NA NA
## 1255 2 9 2007 0.0 37.1 15.8 0.0 NA NA NA NA
## 1256 3 9 2007 0.0 38.1 15.8 NA NA NA NA NA
## 1257 4 9 2007 0.0 37.5 15.8 0.0 NA NA NA NA
## 1258 5 9 2007 0.0 36.7 18.9 0.0 NA NA NA NA
## 1259 6 9 2007 0.0 37.1 17.3 0.0 NA NA NA NA
## 1260 7 9 2007 0.0 37.3 16.8 0.0 NA NA NA NA
## 1261 8 9 2007 0.0 37.9 16.3 0.0 NA NA NA NA
## 1262 9 9 2007 0.0 38.5 15.8 NA NA NA NA NA
## 1263 10 9 2007 0.0 38.5 18.4 0.0 NA NA NA NA
## 1264 11 9 2007 0.0 38.8 18.4 NA NA NA NA NA
## 1265 12 9 2007 0.0 39.9 18.9 NA NA NA NA NA
## 1266 13 9 2007 0.0 39.5 19.9 0.0 NA NA NA NA
## 1267 14 9 2007 0.0 39.1 19.9 0.0 NA NA NA NA
## 1268 25 9 2007 0.0 30.9 17.3 NA NA NA NA NA
## 1269 26 9 2007 0.0 35.5 16.8 0.0 NA NA NA NA
## 1270 24 10 2007 46.5 24.9 19.9 0.0 NA NA NA NA
## 1271 25 10 2007 53.4 NA 19.4 0.0 NA NA NA NA
## 1272 26 10 2007 9.8 34.1 19.9 0.0 NA NA NA NA
## 1273 25 12 2007 3.6 32.1 19.9 NA NA NA NA NA
## 1274 16 3 2008 0.0 31.5 19.9 NA NA NA NA NA
## 1275 5 4 2008 3.3 28.0 19.4 NA NA NA NA NA
## 1276 6 4 2008 0.0 31.3 19.4 NA NA NA NA NA
## 1277 15 4 2008 16.1 26.4 17.8 NA NA NA NA NA
## 1278 16 4 2008 0.0 29.3 18.9 NA NA NA NA NA
## 1279 21 4 2008 35.3 30.1 19.4 NA NA NA NA NA
## 1280 22 4 2008 0.0 32.1 19.9 NA NA NA NA NA
## 1281 23 4 2008 0.0 31.9 19.9 NA NA NA NA NA
## 1282 24 4 2008 0.0 32.9 18.9 NA NA NA NA NA
## 1283 25 4 2008 0.0 32.5 19.9 NA NA NA NA NA
## 1284 26 4 2008 0.0 33.5 19.9 NA NA NA NA NA
## 1285 30 4 2008 68.0 27.8 19.9 NA NA NA NA NA
## 1286 1 5 2008 0.0 27.6 19.4 NA NA NA NA NA
## 1287 2 5 2008 0.0 22.5 15.8 NA NA NA NA NA
## 1288 3 5 2008 0.0 23.3 14.3 NA NA NA NA NA
## 1289 4 5 2008 0.0 25.4 12.2 NA NA NA NA NA
## 1290 5 5 2008 0.0 26.8 14.3 NA NA NA NA NA

```

```

## 1291 6 5 2008 0.0 29.9 15.3 NA NA NA NA
## 1292 7 5 2008 0.0 32.5 16.8 NA NA NA NA
## 1293 8 5 2008 0.0 32.7 18.9 NA NA NA NA
## 1294 10 5 2008 8.6 30.7 19.4 NA NA NA NA
## 1295 11 5 2008 0.0 31.3 19.9 NA NA NA NA
## 1296 13 5 2008 0.0 30.3 19.9 NA NA NA NA
## 1297 15 5 2008 0.0 31.7 18.9 NA NA NA NA
## 1298 16 5 2008 0.0 30.3 18.4 NA NA NA NA
## 1299 17 5 2008 18.5 30.7 15.8 NA NA NA NA
## 1300 18 5 2008 0.0 32.5 16.8 NA NA NA NA
## 1301 19 5 2008 0.0 32.7 19.4 NA NA NA NA
## 1302 20 5 2008 0.0 32.7 18.4 NA NA NA NA
## 1303 21 5 2008 0.0 32.3 16.8 NA NA NA NA
## 1304 22 5 2008 0.0 32.3 14.8 NA NA NA NA
## 1305 23 5 2008 0.0 31.5 14.8 NA NA NA NA
## 1306 24 5 2008 0.0 31.5 15.8 NA NA NA NA
## 1307 25 5 2008 0.0 31.3 14.6 NA NA NA NA
## 1308 26 5 2008 0.0 32.5 16.8 NA NA NA NA
## 1309 27 5 2008 0.0 32.3 16.3 NA NA NA NA
## 1310 28 5 2008 0.0 33.3 18.9 NA NA NA NA
## 1311 30 5 2008 15.8 21.7 14.8 NA NA NA NA
## 1312 31 5 2008 0.0 24.1 14.3 NA NA NA NA
## 1313 1 6 2008 0.0 23.9 14.8 NA NA NA NA
## 1314 2 6 2008 0.0 29.1 14.8 NA NA NA NA
## 1315 3 6 2008 0.0 29.5 14.8 NA NA NA NA
## 1316 4 6 2008 0.0 29.5 18.9 NA NA NA NA
## 1317 5 6 2008 0.0 31.3 15.8 NA NA NA NA
## 1318 6 6 2008 0.0 31.9 15.3 NA NA NA NA
## 1319 7 6 2008 0.0 31.7 14.8 NA NA NA NA
## 1320 8 6 2008 0.0 32.7 15.8 NA NA NA NA
## 1321 9 6 2008 0.0 32.7 16.8 NA NA NA NA
## 1322 10 6 2008 0.0 30.3 16.8 NA NA NA NA
## 1323 11 6 2008 0.0 31.3 19.9 0.0 NA NA NA NA
## 1324 12 6 2008 0.0 32.7 16.3 NA NA NA NA
## 1325 13 6 2008 0.0 32.5 15.8 NA NA NA NA
## 1326 14 6 2008 0.0 32.3 17.3 NA NA NA NA
## 1327 15 6 2008 0.0 32.9 16.3 NA NA NA NA
## 1328 16 6 2008 0.0 30.3 18.9 NA NA NA NA
## 1329 17 6 2008 0.0 30.1 11.7 NA NA NA NA
## 1330 18 6 2008 0.0 32.3 14.3 NA NA NA NA
## 1331 19 6 2008 0.0 32.5 14.8 NA NA NA NA
## 1332 20 6 2008 0.0 32.9 16.8 NA NA NA NA
## 1333 21 6 2008 0.0 29.5 18.4 NA NA NA NA
## 1334 22 6 2008 0.0 21.9 17.8 NA NA NA NA
## 1335 23 6 2008 0.0 27.6 17.3 NA NA NA NA
## 1336 24 6 2008 0.0 27.6 16.3 NA NA NA NA

```

```

## 1337 25 6 2008 0.0 30.9 15.8 NA NA NA NA NA
## 1338 26 6 2008 0.0 31.5 15.8 NA NA NA NA NA
## 1339 27 6 2008 0.0 32.5 15.8 NA NA NA NA NA
## 1340 28 6 2008 0.0 31.3 17.8 NA NA NA NA NA
## 1341 29 6 2008 0.0 31.5 15.3 NA NA NA NA NA
## 1342 30 6 2008 0.0 31.7 15.3 NA NA NA NA NA
## 1343 31 7 2008 0.0 35.4 12.7 NA NA NA NA NA
## 1344 1 8 2008 0.0 35.5 13.7 NA NA NA NA NA
## 1345 3 8 2008 0.0 28.8 19.4 NA NA NA NA NA
## 1346 4 8 2008 0.0 31.1 18.9 NA NA NA NA NA
## 1347 5 8 2008 0.0 36.3 18.9 NA NA NA NA NA
## 1348 7 8 2008 0.0 37.5 17.8 NA NA NA NA NA
## 1349 8 8 2008 0.0 36.7 17.8 NA NA NA NA NA
## 1350 9 8 2008 0.0 36.9 19.9 NA NA NA NA NA
## 1351 10 8 2008 0.0 36.3 18.9 NA NA NA NA NA
## 1352 11 8 2008 0.0 37.9 17.3 NA NA NA NA NA
## 1353 12 8 2008 0.0 37.9 17.8 NA NA NA NA NA
## 1354 13 8 2008 0.0 37.3 16.8 NA NA NA NA NA
## 1355 14 8 2008 0.0 37.3 15.8 NA NA NA NA NA
## 1356 15 8 2008 0.0 37.3 14.3 NA NA NA NA NA
## 1357 16 8 2008 0.0 36.7 14.8 NA NA NA NA NA
## 1358 17 8 2008 0.0 36.9 15.8 NA NA NA NA NA
## 1359 18 8 2008 0.0 36.5 15.8 NA NA NA NA NA
## 1360 19 8 2008 0.0 36.3 15.3 NA NA NA NA NA
## 1361 20 8 2008 0.0 36.1 14.8 NA NA NA NA NA
## 1362 21 8 2008 0.0 35.5 14.8 NA NA NA NA NA
## 1363 22 8 2008 0.0 35.7 14.8 NA NA NA NA NA
## 1364 23 8 2008 0.0 36.1 16.8 NA NA NA NA NA
## 1365 24 8 2008 0.0 36.9 15.3 NA NA NA NA NA
## 1366 25 8 2008 0.0 37.3 16.8 NA NA NA NA NA
## 1367 26 8 2008 0.0 37.3 16.8 NA NA NA NA NA
## 1368 27 8 2008 0.0 37.5 15.8 NA NA NA NA NA
## 1369 28 8 2008 0.0 38.3 15.8 NA NA NA NA NA
## 1370 30 8 2008 0.0 35.5 17.8 NA NA NA NA NA
## 1371 31 8 2008 0.0 36.3 17.3 NA NA NA NA NA
## 1372 1 9 2008 0.0 38.5 18.9 NA NA NA NA NA
## 1373 2 9 2008 0.0 39.5 16.3 NA NA NA NA NA
## 1374 3 9 2008 0.0 38.9 16.3 NA NA NA NA NA
## 1375 4 9 2008 0.0 39.5 18.9 NA NA NA NA NA
## 1376 6 9 2008 0.0 27.6 17.3 NA NA NA NA NA
## 1377 7 9 2008 0.0 32.3 13.7 NA NA NA NA NA
## 1378 8 9 2008 0.0 38.1 15.8 NA NA NA NA NA
## 1379 9 9 2008 0.0 40.3 19.4 NA NA NA NA NA
## 1380 10 9 2008 0.0 40.7 18.9 NA NA NA NA NA
## 1381 11 9 2008 0.0 39.9 15.8 NA NA NA NA NA
## 1382 13 9 2008 0.0 33.3 19.4 NA NA NA NA NA

```

```

## 1383 14 9 2008 0.0 34.7 19.4 NA NA NA NA NA
## 1384 15 9 2008 0.0 35.5 19.9 NA NA NA NA NA
## 1385 17 9 2008 0.0 36.9 18.9 NA NA NA NA NA
## 1386 19 9 2008 0.0 38.5 19.9 NA NA NA NA NA
## 1387 21 9 2008 12.7 21.1 16.8 NA NA NA NA NA
## 1388 22 9 2008 0.0 26.4 14.3 NA NA NA NA NA
## 1389 23 9 2008 0.0 33.3 9.7 NA NA NA NA NA
## 1390 24 9 2008 0.0 36.5 12.2 NA NA NA NA NA
## 1391 25 9 2008 0.0 39.5 16.3 NA NA NA NA NA
## 1392 26 9 2008 0.0 38.5 19.9 NA NA NA NA NA
## 1393 27 9 2008 0.0 38.9 19.4 NA NA NA NA NA
## 1394 28 9 2008 0.0 38.9 19.9 NA NA NA NA NA
## 1395 7 10 2008 0.0 33.3 19.9 NA NA NA NA NA
## 1396 8 10 2008 0.0 37.5 19.9 NA NA NA NA NA
## 1397 4 12 2008 0.0 31.5 15.8 NA NA NA NA NA
## 1398 5 12 2008 0.0 33.3 15.8 NA NA NA NA NA
## 1399 6 12 2008 0.0 35.5 19.4 NA NA NA NA NA
## 1400 5 1 2009 0.0 30.1 17.6 NA NA NA NA NA
## 1401 6 1 2009 0.0 31.5 14.8 NA NA NA NA NA
## 1402 7 1 2009 0.0 33.5 16.3 NA NA NA NA NA
## 1403 8 1 2009 0.0 35.3 18.4 NA NA NA NA NA
## 1404 21 1 2009 0.0 31.3 17.8 NA NA NA NA NA
## 1405 22 1 2009 0.0 33.3 17.3 NA NA NA NA NA
## 1406 12 4 2009 0.0 32.8 19.9 NA NA NA NA NA
## 1407 16 4 2009 0.0 33.2 19.9 NA NA NA NA NA
## 1408 17 4 2009 0.0 33.4 19.9 NA NA NA NA NA
## 1409 21 4 2009 0.0 32.6 17.8 NA NA NA NA NA
## 1410 22 4 2009 0.0 32.8 17.8 NA NA NA NA NA
## 1411 23 4 2009 0.0 33.4 17.8 NA NA NA NA NA
## 1412 24 4 2009 0.0 32.1 18.9 NA NA NA NA NA
## 1413 25 4 2009 0.0 32.4 19.4 NA NA NA NA NA
## 1414 26 4 2009 0.0 33.4 18.9 NA NA NA NA NA
## 1415 27 4 2009 0.0 32.8 19.9 NA NA NA NA NA
## 1416 28 4 2009 0.0 32.6 19.4 NA NA NA NA NA
## 1417 29 4 2009 0.0 32.1 16.8 NA NA NA NA NA
## 1418 30 4 2009 0.0 32.6 15.3 NA NA NA NA NA
## 1419 1 5 2009 0.0 32.4 15.3 NA NA NA NA NA
## 1420 2 5 2009 0.0 32.8 15.8 NA NA NA NA NA
## 1421 3 5 2009 0.0 32.6 14.8 NA NA NA NA NA
## 1422 4 5 2009 0.0 31.7 15.3 NA NA NA NA NA
## 1423 5 5 2009 28.0 30.1 17.8 NA NA NA NA NA
## 1424 6 5 2009 0.0 32.1 17.3 NA NA NA NA NA
## 1425 7 5 2009 0.0 32.4 18.9 NA NA NA NA NA
## 1426 8 5 2009 0.0 33.6 17.3 NA NA NA NA NA
## 1427 9 5 2009 0.0 33.6 17.3 NA NA NA NA NA
## 1428 10 5 2009 0.0 33.6 18.4 NA NA NA NA NA

```

```
## 1429 12 5 2009 0.0 33.6 19.9 NA NA NA NA NA
## 1430 15 5 2009 14.4 23.7 19.4 NA NA NA NA NA
## 1431 16 5 2009 0.0 26.7 18.4 NA NA NA NA NA
## 1432 17 5 2009 0.0 30.3 18.4 NA NA NA NA NA
## 1433 18 5 2009 0.0 31.5 18.9 NA NA NA NA NA
## 1434 19 5 2009 0.0 32.4 18.9 NA NA NA NA NA
## 1435 20 5 2009 0.0 32.2 17.3 NA NA NA NA NA
## 1436 21 5 2009 0.0 31.5 17.8 NA NA NA NA NA
## 1437 22 5 2009 0.0 31.9 18.4 NA NA NA NA NA
## 1438 23 5 2009 0.0 33.0 18.4 NA NA NA NA NA
## 1439 24 5 2009 0.0 33.0 18.9 NA NA NA NA NA
## 1440 25 5 2009 0.0 33.4 19.9 NA NA NA NA NA
## 1441 26 5 2009 9.5 31.9 18.9 NA NA NA NA NA
## 1442 27 5 2009 0.0 33.8 19.4 NA NA NA NA NA
## 1443 28 5 2009 0.0 34.0 19.4 NA NA NA NA NA
## 1444 29 5 2009 0.0 32.8 19.4 NA NA NA NA NA
## 1445 30 5 2009 0.0 28.5 18.9 NA NA NA NA NA
## 1446 31 5 2009 0.0 24.9 17.8 NA NA NA NA NA
## 1447 1 6 2009 0.0 21.0 15.3 NA NA NA NA NA
## 1448 2 6 2009 0.0 25.1 14.8 NA NA NA NA NA
## 1449 3 6 2009 0.0 25.7 9.2 NA NA NA NA NA
## 1450 4 6 2009 0.0 30.3 7.2 NA NA NA NA NA
## 1451 5 6 2009 0.0 32.6 11.2 NA NA NA NA NA
## 1452 6 6 2009 0.0 32.1 16.3 NA NA NA NA NA
## 1453 7 6 2009 0.0 32.6 14.8 NA NA NA NA NA
## 1454 8 6 2009 0.0 32.4 12.7 NA NA NA NA NA
## 1455 9 6 2009 0.0 33.0 15.3 NA NA NA NA NA
## 1456 10 6 2009 0.0 32.8 17.3 NA NA NA NA NA
## 1457 11 6 2009 0.0 30.7 18.9 NA NA NA NA NA
## 1458 12 6 2009 0.0 27.5 19.9 NA NA NA NA NA
## 1459 13 6 2009 0.0 27.5 16.8 NA NA NA NA NA
## 1460 14 6 2009 0.0 31.9 14.2 NA NA NA NA NA
## 1461 15 6 2009 0.0 33.4 16.3 NA NA NA NA NA
## 1462 17 6 2009 8.7 27.5 18.4 NA NA NA NA NA
## 1463 18 6 2009 0.0 30.3 15.3 NA NA NA NA NA
## 1464 19 6 2009 0.0 32.4 15.8 NA NA NA NA NA
## 1465 20 6 2009 0.0 31.9 13.7 NA NA NA NA NA
## 1466 21 6 2009 0.0 32.6 17.3 NA NA NA NA NA
## 1467 22 6 2009 0.0 31.5 17.3 NA NA NA NA NA
## 1468 23 6 2009 0.0 32.2 15.8 NA NA NA NA NA
## 1469 24 6 2009 0.0 25.7 17.3 NA NA NA NA NA
## 1470 25 6 2009 0.0 27.7 17.3 NA NA NA NA NA
## 1471 26 6 2009 0.0 29.7 16.8 NA NA NA NA NA
## 1472 27 6 2009 0.0 31.7 18.9 NA NA NA NA NA
## 1473 28 6 2009 0.0 32.8 17.8 NA NA NA NA NA
## 1474 29 6 2009 0.0 33.6 17.3 NA NA NA NA NA
```

```

## 1475 30 6 2009 0.0 33.2 16.8 NA NA NA NA NA
## 1476 1 7 2009 0.0 33.6 17.3 NA NA NA NA NA
## 1477 2 7 2009 0.0 33.8 16.3 NA NA NA NA NA
## 1478 3 7 2009 0.0 31.9 14.8 NA NA NA NA NA
## 1479 4 7 2009 0.0 31.9 14.3 NA NA NA NA NA
## 1480 5 7 2009 0.0 32.2 14.3 NA NA NA NA NA
## 1481 6 7 2009 0.0 33.4 16.3 NA NA NA NA NA
## 1482 7 7 2009 0.0 32.6 14.8 NA NA NA NA NA
## 1483 8 7 2009 0.0 32.4 15.3 NA NA NA NA NA
## 1484 9 7 2009 0.0 33.6 15.8 NA NA NA NA NA
## 1485 10 7 2009 0.0 34.4 18.9 NA NA NA NA NA
## 1486 11 7 2009 0.0 30.0 17.8 NA NA NA NA NA
## 1487 12 7 2009 5.3 26.1 14.3 NA NA NA NA NA
## 1488 13 7 2009 0.0 32.6 13.2 NA NA NA NA NA
## 1489 14 7 2009 0.0 33.4 17.8 NA NA NA NA NA
## 1490 15 7 2009 0.0 34.6 18.4 NA NA NA NA NA
## 1491 16 7 2009 0.0 34.6 18.4 NA NA NA NA NA
## 1492 17 7 2009 0.0 33.8 18.9 NA NA NA NA NA
## 1493 18 7 2009 0.0 33.2 15.8 NA NA NA NA NA
## 1494 19 7 2009 0.0 33.6 15.8 NA NA NA NA NA
## 1495 20 7 2009 0.0 34.0 14.8 NA NA NA NA NA
## 1496 21 7 2009 0.0 35.0 16.3 NA NA NA NA NA
## 1497 22 7 2009 0.0 35.4 18.9 NA NA NA NA NA
## 1498 23 7 2009 0.0 29.5 18.9 NA NA NA NA NA
## 1499 24 7 2009 0.7 22.9 12.2 NA NA NA NA NA
## 1500 25 7 2009 0.0 23.5 10.7 NA NA NA NA NA
## 1501 26 7 2009 0.0 31.7 10.9 NA NA NA NA NA
## 1502 27 7 2009 0.0 33.0 15.3 NA NA NA NA NA
## 1503 28 7 2009 0.0 36.4 17.3 NA NA NA NA NA
## 1504 29 7 2009 0.0 35.0 17.8 NA NA NA NA NA
## 1505 30 7 2009 0.0 34.8 17.8 NA NA NA NA NA
## 1506 31 7 2009 0.0 35.6 16.8 NA NA NA NA NA
## 1507 1 8 2009 0.0 36.0 15.3 NA NA NA NA NA
## 1508 2 8 2009 0.0 36.4 14.8 NA NA NA NA NA
## 1509 3 8 2009 0.0 33.6 16.6 NA NA NA NA NA
## 1510 4 8 2009 0.0 35.8 17.3 NA NA NA NA NA
## 1511 5 8 2009 0.0 37.2 16.3 NA NA NA NA NA
## 1512 6 8 2009 0.0 37.2 16.3 NA NA NA NA NA
## 1513 7 8 2009 0.0 36.8 15.8 NA NA NA NA NA
## 1514 8 8 2009 0.0 35.6 14.8 NA NA NA NA NA
## 1515 9 8 2009 0.0 31.5 14.3 NA NA NA NA NA
## 1516 10 8 2009 0.0 27.1 12.2 NA NA NA NA NA
## 1517 11 8 2009 0.0 31.7 11.2 NA NA NA NA NA
## 1518 12 8 2009 0.0 35.4 12.2 NA NA NA NA NA
## 1519 13 8 2009 0.0 36.6 15.3 NA NA NA NA NA
## 1520 14 8 2009 0.0 36.4 16.8 NA NA NA NA NA

```

```
## 1521 15 8 2009 0.0 36.2 16.3 NA NA NA NA NA
## 1522 16 8 2009 0.0 38.4 17.3 NA NA NA NA NA
## 1523 17 8 2009 0.0 38.4 17.8 NA NA NA NA NA
## 1524 18 8 2009 0.0 38.4 19.4 NA NA NA NA NA
## 1525 20 8 2009 6.3 29.9 19.9 NA NA NA NA NA
## 1526 21 8 2009 5.6 28.3 18.4 NA NA NA NA NA
## 1527 22 8 2009 6.3 34.6 16.3 NA NA NA NA NA
## 1528 23 8 2009 0.0 35.6 19.4 NA NA NA NA NA
## 1529 25 8 2009 0.6 26.7 19.9 NA NA NA NA NA
## 1530 26 8 2009 0.0 34.6 18.4 NA NA NA NA NA
## 1531 27 8 2009 0.0 35.4 18.9 NA NA NA NA NA
## 1532 28 8 2009 0.0 35.6 18.4 NA NA NA NA NA
## 1533 29 8 2009 0.0 36.4 18.8 NA NA NA NA NA
## 1534 30 8 2009 0.0 37.2 17.3 NA NA NA NA NA
## 1535 1 9 2009 0.0 35.8 19.9 NA NA NA NA NA
## 1536 11 9 2009 0.0 28.5 15.8 NA NA NA NA NA
## 1537 12 9 2009 0.0 31.9 17.8 NA NA NA NA NA
## 1538 13 9 2009 0.0 36.8 18.4 NA NA NA NA NA
## 1539 23 9 2009 0.0 30.3 18.4 NA NA NA NA NA
## 1540 24 9 2009 1.7 26.9 16.8 NA NA NA NA NA
## 1541 25 9 2009 0.0 34.6 16.8 NA NA NA NA NA
## 1542 26 9 2009 0.0 38.2 18.4 NA NA NA NA NA
## 1543 29 9 2009 0.8 27.7 16.8 NA NA NA NA NA
## 1544 30 9 2009 0.0 33.4 16.3 NA NA NA NA NA
## 1545 1 10 2009 0.0 37.4 17.8 NA NA NA NA NA
## 1546 8 3 2010 0.0 36.8 19.9 NA NA NA NA NA
## 1547 7 4 2010 0.0 29.5 17.3 NA NA NA NA NA
## 1548 8 4 2010 0.0 29.3 15.3 NA NA NA NA NA
## 1549 9 4 2010 0.0 29.1 15.3 NA NA NA NA NA
## 1550 10 4 2010 0.0 31.1 14.3 NA NA NA NA NA
## 1551 11 4 2010 0.0 32.6 16.3 NA NA NA NA NA
## 1552 12 4 2010 0.0 33.2 16.8 NA NA NA NA NA
## 1553 13 4 2010 0.0 33.0 17.8 NA NA NA NA NA
## 1554 14 4 2010 0.0 33.0 17.8 NA NA NA NA NA
## 1555 15 4 2010 0.0 33.6 17.3 NA NA NA NA NA
## 1556 16 4 2010 0.0 34.0 18.4 NA NA NA NA NA
## 1557 17 4 2010 0.0 34.6 19.9 NA NA NA NA NA
## 1558 19 4 2010 0.0 35.8 19.9 NA NA NA NA NA
## 1559 9 5 2010 0.0 24.3 18.4 NA NA NA NA NA
## 1560 10 5 2010 0.0 26.7 14.3 NA NA NA NA NA
## 1561 11 5 2010 0.0 26.1 10.2 NA NA NA NA NA
## 1562 12 5 2010 0.0 25.3 10.2 NA NA NA NA NA
## 1563 13 5 2010 0.0 26.1 6.2 NA NA NA NA NA
## 1564 14 5 2010 0.0 29.5 7.2 NA NA NA NA NA
## 1565 15 5 2010 0.0 34.0 9.7 NA NA NA NA NA
## 1566 16 5 2010 0.0 36.0 15.8 NA NA NA NA NA
```

```

## 1567 17 5 2010 0.0 35.4 19.4 NA NA NA NA
## 1568 18 5 2010 0.0 26.7 16.8 NA NA NA NA
## 1569 19 5 2010 0.0 26.5 14.8 NA NA NA NA
## 1570 20 5 2010 0.0 32.1 15.3 NA NA NA NA
## 1571 21 5 2010 0.0 31.7 14.3 NA NA NA NA
## 1572 22 5 2010 0.0 31.5 13.7 NA NA NA NA
## 1573 23 5 2010 0.0 33.8 15.3 NA NA NA NA
## 1574 24 5 2010 0.0 33.0 16.8 NA NA NA NA
## 1575 27 5 2010 0.0 35.0 19.9 NA NA NA NA
## 1576 28 5 2010 0.0 34.0 14.8 NA NA NA NA
## 1577 29 5 2010 0.0 34.0 14.3 NA NA NA NA
## 1578 30 5 2010 0.0 34.8 15.3 NA NA NA NA
## 1579 31 5 2010 0.0 27.9 17.8 NA NA NA NA
## 1580 1 6 2010 0.0 28.3 12.2 NA NA NA NA
## 1581 2 6 2010 0.0 32.1 15.8 NA NA NA NA
## 1582 3 6 2010 0.0 33.8 17.8 NA NA NA NA
## 1583 4 6 2010 0.0 36.0 18.4 NA NA NA NA
## 1584 6 6 2010 0.0 29.1 14.8 NA NA NA NA
## 1585 7 6 2010 0.0 33.8 15.8 NA NA NA NA
## 1586 8 6 2010 0.0 34.4 17.8 NA NA NA NA
## 1587 9 6 2010 0.0 35.4 16.8 NA NA NA NA
## 1588 10 6 2010 0.0 34.0 17.8 NA NA NA NA
## 1589 11 6 2010 0.0 34.0 17.8 NA NA NA NA
## 1590 12 6 2010 0.0 33.6 16.3 NA NA NA NA
## 1591 13 6 2010 0.0 33.2 12.7 NA NA NA NA
## 1592 14 6 2010 0.0 34.6 12.7 NA NA NA NA
## 1593 15 6 2010 0.0 34.4 13.7 NA NA NA NA
## 1594 16 6 2010 0.0 34.6 14.8 NA NA NA NA
## 1595 17 6 2010 0.0 34.8 13.7 NA NA NA NA
## 1596 18 6 2010 0.0 35.6 14.3 NA NA NA NA
## 1597 19 6 2010 0.0 35.4 13.7 NA NA NA NA
## 1598 20 6 2010 0.0 35.6 14.3 NA NA NA NA
## 1599 21 6 2010 0.0 36.0 16.3 NA NA NA NA
## 1600 22 6 2010 0.0 35.8 16.3 NA NA NA NA
## 1601 23 6 2010 0.0 35.4 15.8 NA NA NA NA
## 1602 24 6 2010 0.0 36.6 14.8 NA NA NA NA
## 1603 25 6 2010 0.0 35.4 15.8 NA NA NA NA
## 1604 26 6 2010 0.0 34.4 15.8 NA NA NA NA
## 1605 27 6 2010 0.0 35.0 15.3 NA NA NA NA
## 1606 28 6 2010 0.0 34.0 15.3 NA NA NA NA
## 1607 29 6 2010 0.0 32.1 14.8 NA NA NA NA
## 1608 30 6 2010 0.0 33.4 14.3 NA NA NA NA
## 1609 1 7 2010 0.0 34.0 13.7 NA NA NA NA
## 1610 2 7 2010 0.0 33.8 14.8 NA NA NA NA
## 1611 3 7 2010 0.0 34.0 14.3 NA NA NA NA
## 1612 4 7 2010 0.0 34.6 13.7 NA NA NA NA

```

```
## 1613 5 7 2010 0.0 34.8 14.3 NA NA NA NA NA
## 1614 6 7 2010 0.0 35.4 16.3 NA NA NA NA NA
## 1615 7 7 2010 0.0 35.6 16.8 NA NA NA NA NA
## 1616 8 7 2010 0.0 35.0 16.3 NA NA NA NA NA
## 1617 9 7 2010 0.0 35.6 16.3 NA NA NA NA NA
## 1618 10 7 2010 0.0 35.4 15.8 NA NA NA NA NA
## 1619 11 7 2010 0.0 35.8 15.3 NA NA NA NA NA
## 1620 12 7 2010 0.0 35.8 16.3 NA NA NA NA NA
## 1621 13 7 2010 1.2 23.5 16.3 NA NA NA NA NA
## 1622 14 7 2010 0.0 26.3 11.7 NA NA NA NA NA
## 1623 15 7 2010 0.0 27.9 12.3 NA NA NA NA NA
## 1624 16 7 2010 0.0 20.6 12.2 NA NA NA NA NA
## 1625 17 7 2010 0.0 15.8 11.2 NA NA NA NA NA
## 1626 18 7 2010 0.0 20.8 10.2 NA NA NA NA NA
## 1627 19 7 2010 0.0 23.1 10.2 NA NA NA NA NA
## 1628 20 7 2010 0.0 33.8 10.2 NA NA NA NA NA
## 1629 21 7 2010 0.0 35.4 14.8 NA NA NA NA NA
## 1630 22 7 2010 0.0 36.0 15.8 NA NA NA NA NA
## 1631 23 7 2010 0.0 35.8 16.3 NA NA NA NA NA
## 1632 24 7 2010 0.0 35.0 16.3 NA NA NA NA NA
## 1633 25 7 2010 0.0 34.6 12.7 NA NA NA NA NA
## 1634 26 7 2010 0.0 33.8 12.2 NA NA NA NA NA
## 1635 27 7 2010 0.0 32.8 14.3 NA NA NA NA NA
## 1636 28 7 2010 0.0 35.0 12.2 NA NA NA NA NA
## 1637 29 7 2010 0.0 35.8 14.8 NA NA NA NA NA
## 1638 30 7 2010 0.0 35.0 15.8 NA NA NA NA NA
## 1639 31 7 2010 0.0 36.2 16.8 NA NA NA NA NA
## 1640 1 8 2010 0.0 36.5 17.3 NA NA NA NA NA
## 1641 22 8 2010 0.0 37.6 17.8 NA NA NA NA NA
## 1642 23 8 2010 0.0 37.6 15.8 NA NA NA NA NA
## 1643 24 8 2010 0.0 37.8 15.8 NA NA NA NA NA
## 1644 25 8 2010 0.0 38.0 17.3 NA NA NA NA NA
## 1645 26 8 2010 0.0 37.8 16.3 NA NA NA NA NA
## 1646 27 8 2010 0.0 38.6 15.3 NA NA NA NA NA
## 1647 28 8 2010 0.0 38.6 16.8 NA NA NA NA NA
## 1648 29 8 2010 0.0 31.1 19.9 NA NA NA NA NA
## 1649 31 8 2010 0.0 38.6 19.4 NA NA NA NA NA
## 1650 2 9 2010 0.0 39.8 19.4 NA NA NA NA NA
## 1651 3 9 2010 0.0 38.0 18.4 NA NA NA NA NA
## 1652 5 9 2010 0.0 33.6 17.3 NA NA NA NA NA
## 1653 6 9 2010 0.0 36.4 17.3 NA NA NA NA NA
## 1654 7 9 2010 0.0 38.0 18.9 NA NA NA NA NA
## 1655 8 9 2010 0.0 39.0 19.8 NA NA NA NA NA
## 1656 10 9 2010 0.0 40.0 18.9 NA NA NA NA NA
## 1657 13 9 2010 0.0 39.8 16.8 NA NA NA NA NA
## 1658 14 9 2010 0.0 36.6 19.4 NA NA NA NA NA
```

```

## 1659 15 9 2010 0.0 35.8 18.4 NA NA NA NA
## 1660 16 9 2010 0.0 40.8 18.9 NA NA NA NA
## 1661 21 9 2010 0.0 39.8 18.4 NA NA NA NA
## 1662 22 9 2010 0.0 39.4 19.4 NA NA NA NA
## 1663 23 9 2010 0.0 39.0 18.9 NA NA NA NA
## 1664 4 10 2010 0.0 36.8 19.4 NA NA NA NA
## 1665 8 10 2010 26.2 34.0 19.4 NA NA NA NA
## 1666 9 10 2010 0.0 37.0 18.4 NA NA NA NA
## 1667 10 10 2010 0.0 37.6 18.4 NA NA NA NA
## 1668 27 10 2010 0.0 36.8 19.4 NA NA NA NA
## 1669 28 10 2010 0.0 37.8 18.9 NA NA NA NA
## 1670 29 10 2010 0.0 36.8 18.9 NA NA NA NA
## 1671 2 11 2010 0.4 32.6 19.9 NA NA NA NA
## 1672 3 11 2010 0.0 NA 17.3 NA NA NA NA

```

Apresentar somente os dias com temperatura menor que 20°C e umidade relativa do ar menor que 30%:

```

dados_criticos <- filter(roo,Tmin < 20, UR < 30 )
head(dados_criticos)

```

```

## dd mm ano Prec Tmax Tmin n Tbs Tbu UR Vvento
## 1 31 7 1998 0 36.8 13.8 NA 31.125 18.05 28.75 0.05
## 2 5 9 2004 0 38.7 16.9 NA 29.600 17.95 29.00 0.45
## 3 7 10 2004 0 38.9 17.9 NA 31.050 19.00 28.25 0.35
## 4 15 8 2005 0 NA 14.4 0 27.950 17.00 29.75 0.15
## 5 7 9 2005 0 NA 18.9 0 29.100 21.60 29.75 0.20

```

Quando precisamos adicionar uma coluna usamos a função `mutate`:

```

dados <- mutate(roo, Tmd = ((Tmax + Tmin )/2))
head(dados)

```

```

## dd mm ano Prec Tmax Tmin n Tbs Tbu UR Vvento Tmd
## 1 1 1 1998 NA 30.0 21.7 NA NA NA NA 25.85
## 2 1 2 1998 8.2 35.6 21.8 NA NA NA NA 28.70
## 3 2 2 1998 51.0 31.8 21.8 NA 25.075 23.85 89.75 0.125 26.80
## 4 3 2 1998 0.6 35.4 21.5 NA 24.975 23.00 86.50 0.125 28.45
## 5 4 2 1998 0.0 35.6 22.1 NA 26.650 23.90 80.00 0.275 28.85
## 6 5 2 1998 0.0 36.4 22.5 NA 26.950 24.00 78.50 0.325 29.45

```

Resumir e agrupar os dados usando a função `summarise`

Qual o valor mínimo de umidade relativa?

```
summarise(roo, min(UR, na.rm = TRUE))
```

```
##   min(UR, na.rm = TRUE)
## 1          27.25
```

Qual o valor médio de velocidade do vento por mês?

```
summarise(group_by(roo, mm), mean(Vvento, na.rm = TRUE))
```

```
## # A tibble: 12 x 2
##       mm `mean(Vvento, na.rm = TRUE)` 
##   <int>            <dbl>
## 1     1             0.432
## 2     2             0.385
## 3     3             0.386
## 4     4             0.322
## 5     5             0.378
## 6     6             0.443
## 7     7             0.450
## 8     8             0.432
## 9     9             0.430
## 10   10            0.427
## 11   11            0.485
## 12   12            0.417
```

Vamos aprender a usar o operador **pipe**

A idéia do operador ‘%>%’ (pipe) é bem simples: usar o valor resultante da expressão do lado esquerdo como primeiro argumento da função do lado direito.

Iremos filtrar os valores do banco de dados *roo* com as variáveis UR e mês com apresentação do cabeçalho (‘head()’):

```
head(select(roo, UR, mm))
```

```
##      UR mm
## 1    NA  1
## 2    NA  2
## 3 89.75  2
## 4 86.50  2
## 5 80.00  2
## 6 78.50  2
```

Usando o piper:

```
roo %>% select(UR, mm) %>% head
```

```
##      UR mm
## 1    NA 1
## 2    NA 2
## 3 89.75 2
## 4 86.50 2
## 5 80.00 2
## 6 78.50 2
```

Exploração de dados: gerar um resumo estatístico com o comando pipe:

```
roo %>%
  summarise(avg = mean(UR,na.rm=TRUE),
            min = min(UR,na.rm=TRUE),
            max = max(UR,na.rm=TRUE),
            total = n())
```



```
##      avg   min max total
## 1 74.41532 27.25 100  4337
```

Exploração de dados gerar um resumo estatístico com o comando pipe agrupado por mês:

```
roo %>%
  group_by(mm) %>%
  summarise(mean(UR, na.rm = TRUE))
```

```
## # A tibble: 12 x 2
##       mm `mean(UR, na.rm = TRUE)` 
##   <int>          <dbl>
## 1     1             86.4
## 2     2             85.8
## 3     3             85.6
## 4     4             79.7
## 5     5             75.8
## 6     6             68.3
## 7     7             61.3
## 8     8             51.4
## 9     9             61.0
## 10   10            71.4
## 11   11            77.4
## 12   12            84.4
```

Caso necessitamos retirar o mês de outubro:

```
roo %>%
  filter(mm != 10) %>%
  group_by(mm) %>%
  summarise(mean(UR, na.rm = TRUE))

## # A tibble: 11 x 2
##       mm `mean(UR, na.rm = TRUE)` 
##   <int>                <dbl>
## 1     1                  86.4
## 2     2                  85.8
## 3     3                  85.6
## 4     4                  79.7
## 5     5                  75.8
## 6     6                  68.3
## 7     7                  61.3
## 8     8                  51.4
## 9     9                  61.0
## 10    11                 77.4
## 11    12                 84.4
```

Com os dados roo vamos filtrar os valores mensais da UR e Tmax, com uma condição de apresentar somente os valores de Tmax maiores que 40. Apresentação com a função head:

```
roo %>%
  select(mm, UR, Tmax) %>%
  filter(Tmax > 40) %>% head

## #> #> #> #> #>
```

	mm	UR	Tmax
## 1	9	61.0	40.4
## 2	10	56.5	40.1
## 3	9	NA	40.5
## 4	9	NA	40.1
## 5	9	NA	40.1
## 6	10	NA	40.5

Explorar os dados e visualizar com pacote dplyr e ggplot2

Vamos utilizar um novo banco de dados:

```
trmm <- read.csv2(file ="https://www.dropbox.com/s/hf80ptt7lm6kbdb/roo_trmm.csv?dl=1",
head(trmm)
```

```

##          TIME MONTH_01 MONTH_02 MONTH_03 MONTH_04 MONTH_05 MONTH_06 MONTH_07
## 2003    317.6  161.086  412.182  147.819   67.488  0.0207795  1.81712  4.11907
## 2004    312.005 208.448  87.4705  133.678   42.0127  15.5672  103.441     0.0
## 2005    379.356  93.9756  246.675   59.383   19.4379  26.0076      0.0  2.70203
## 2006    290.128 225.015  248.064  113.462   31.0722  0.252401  41.6723  4.64863
## 2007    244.462  378.149  97.8047  56.5509  33.4926  0.0622512  28.4546  0.139021
## 2008    450.736  200.676  169.109  207.288  60.3384  0.0730931  0.00526014  1.45909
##          MONTH_08 MONTH_09 MONTH_10 MONTH_11 MONTH_12
## 2003    137.034  97.0541  209.735  268.363      NA
## 2004    23.4719  162.749  174.922  157.4      NA
## 2005    48.2405  90.2749  106.92   327.281      NA
## 2006    83.8976  228.045  119.292  356.136      NA
## 2007    5.64493  124.682  176.562  199.727      NA
## 2008    45.046   84.4426  224.299  248.747      NA

```

Com o pacote tidyverse podemos reorganizar a tabela:

```
library(tidyverse)
```

Com a função `gather` iremos unir todos valores de chuva mensal em uma coluna separada pelo meses:

```
dados = gather(trmm, TIME, ppt, MONTH_01:MONTH_12, na.rm = TRUE)
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```

```
head(dados)
```

```

##          TIME      ppt
## 1 MONTH_01 161.086
## 2 MONTH_01 208.448
## 3 MONTH_01  93.9756
## 4 MONTH_01 225.015
## 5 MONTH_01 378.149
## 6 MONTH_01 200.676

```

Visualizar dados temporais no R

Vamos baixar os dados climáticos de Rondonpolis-MT:

```
roo <- read.csv2("https://www.dropbox.com/s/1ajoi1c8pla3yk6/roo.csv?dl=1")
head(roo)
```

```
##   dd mm ano Prec Tmax Tmin n    Tbs    Tbu    UR Vvento
## 1 1 1 1998 NA 30.0 21.7 NA     NA     NA     NA     NA
## 2 1 2 1998 8.2 35.6 21.8 NA     NA     NA     NA     NA
## 3 2 2 1998 51.0 31.8 21.8 NA 25.075 23.85 89.75 0.125
## 4 3 2 1998 0.6 35.4 21.5 NA 24.975 23.00 86.50 0.125
## 5 4 2 1998 0.0 35.6 22.1 NA 26.650 23.90 80.00 0.275
## 6 5 2 1998 0.0 36.4 22.5 NA 26.950 24.00 78.50 0.325
```

```
tail(roo)
```

```
##      dd mm ano Prec Tmax Tmin n Tbs Tbu UR Vvento
## 4332 29 10 2010 0.0 36.8 18.9 NA NA NA NA     NA
## 4333 30 10 2010 0.0 36.4 24.8 NA NA NA NA     NA
## 4334 31 10 2010 49.0 33.6 22.5 NA NA NA NA     NA
## 4335  1 11 2010 21.6 30.7 21.5 NA NA NA NA     NA
## 4336  2 11 2010 0.4 32.6 19.9 NA NA NA NA     NA
## 4337  3 11 2010 0.0    NA 17.3 NA NA NA NA     NA
```

Os dados de Roo inicia de janeiro de 1998 a novembro de 2010.

Vamos retirar as colunas das datas:

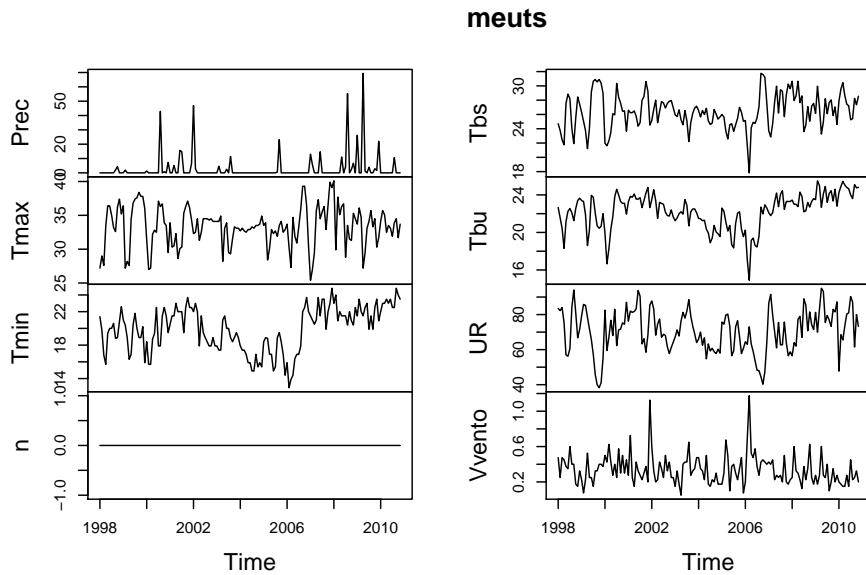
```
roo2=roo[,4:11]
head(roo2)
```

```
##   Prec Tmax Tmin n    Tbs    Tbu    UR Vvento
## 1  NA 30.0 21.7 NA     NA     NA     NA
## 2  8.2 35.6 21.8 NA     NA     NA     NA
## 3 51.0 31.8 21.8 NA 25.075 23.85 89.75 0.125
## 4  0.6 35.4 21.5 NA 24.975 23.00 86.50 0.125
## 5  0.0 35.6 22.1 NA 26.650 23.90 80.00 0.275
## 6  0.0 36.4 22.5 NA 26.950 24.00 78.50 0.325
```

```
roo2=na.omit(roo2)
```

Converter os dados em um objeto de série temporal (ts):

```
meuts = ts(roo2, start=c(1998, 1), end=c(2010, 11), frequency=12)
plot(meuts)
```



Verificar os dados:

```
start(meuts)
```

```
## [1] 1998     1
```

```
end(meuts)
```

```
## [1] 2010     11
```

```
frequency(meuts)
```

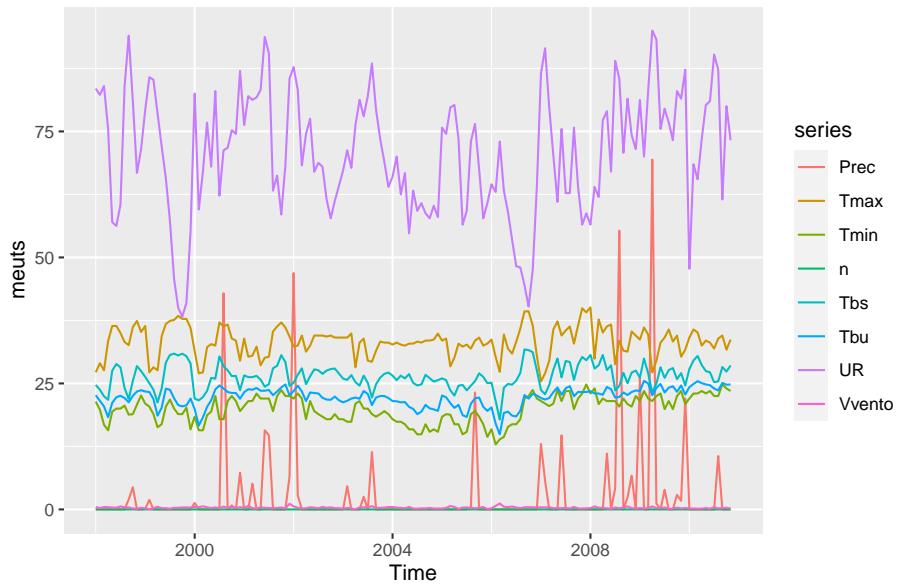
```
## [1] 12
```

Apresentação gráfica:

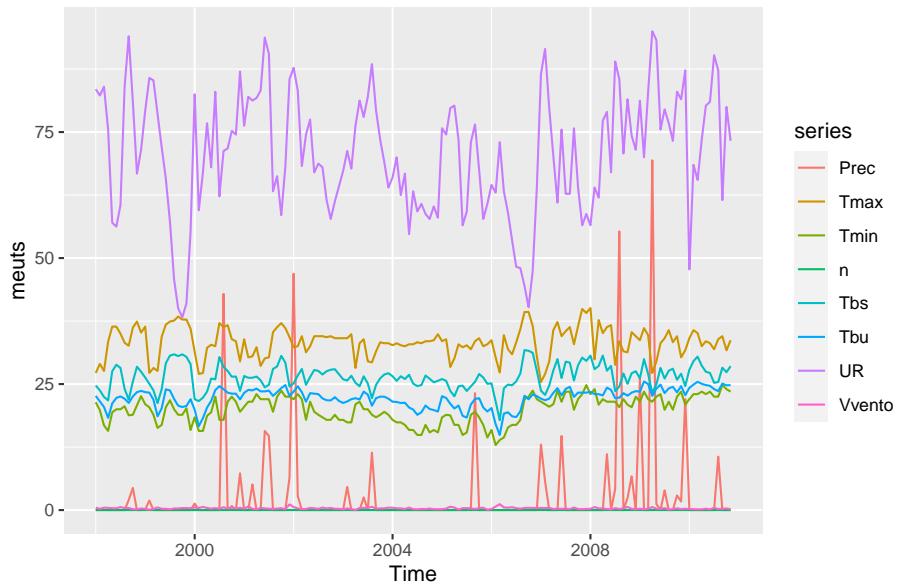
```
library(devtools)

require(ggfortify) #will plot the ts objects as ggplot2

autoplot(meuts) # time series on facets
```



```
autoplot(meuts, facets = F) #time series of stocks on one plot
```



10.4 Datas no R

O Pacote Lubridate fornece ferramentas que facilitam a análise e manipulação de datas. Essas ferramentas são agrupadas abaixo por um propósito comum. Mais informações sobre cada função podem ser encontradas em sua documentação de ajuda:

```
#install.packages("lubridate")
library(lubridate)
```

Qual o dia de hoje?

```
today()
```

```
## [1] "2020-05-13"
```

Qual o dia e horário?

```
now()
```

```
## [1] "2020-05-13 11:24:22 -05"
```

Atribuir data em um objeto:

```
x="1983-10-28"
```

YYYY-MM-DD: corresponde ao ano, mês e dia.

Qual a estrutura desse objeto?

```
str(x)
```

```
## chr "1983-10-28"
```

O objeto *x* é um caracter, precisamos transformar para data com a função `as.Date()`:

```
dia1 <- as.Date(x)
str(dia1)
```

```
## Date[1:1], format: "1983-10-28"
```

Convertendo datas não padronizadas para padrão:

```

dia2 = as.Date("12/27/2015", format = "%m/%d/%Y")
str(dia2)

## Date[1:1], format: "2015-12-27"

dia3 = as.Date("Novembro 22, 1998", format = "%B %d, %Y")
str(dia3)

## Date[1:1], format: "1998-11-22"

```

Cálculos com as datas:

```

dia2-dia3

## Time difference of 6244 days

dia2>dia3

## [1] TRUE

```

10.5 Teste Mann-Kendall para Tendência

O teste de tendência Mann Kendall é usado para analisar dados coletados ao longo do tempo para aumentar ou diminuir consistentemente tendências (“tendências monotônicas”) em valores de Y.

É um teste não paramétrico, o que significa que funciona para todas as distribuições (ou seja, seus dados não precisam atender à suposição de normalidade), mas seus dados não devem ter correlação serial. Se seus dados seguem uma distribuição normal, você pode executar uma regressão linear simples.

Para realizar o teste de tendência Mann-Kendall em R, você pode usar o pacote Kendall ou trend.

Para realizar o teste de tendência Mann-Kendall em R, você pode usar o pacote Kendall ou trend:

```

library(Kendall)
require(trend)

```

O pacote Kendall tem uma função chamada `MannKendall()` que implementa o teste não paramétrico para detecção de tendência monotônica, conhecido como

teste de Mann-Kendall. (Uma tendência monótona pode ser uma tendência ascendente ou uma tendência descendente).

No pacote trend a função é chamada de `mk.test` () Para ilustrar as funções dos pacotes em R, usaremos os conjuntos de dados climáticos de Rondonópolis-MT (`roo.csv`):

```
roo <- read.csv2 ("https://www.dropbox.com/s/1ajoi1c8pla3yk6/roo.csv?dl=1")

head (roo)

##   dd mm  ano Prec Tmax Tmin  n    Tbs    Tbu     UR Vvento
## 1  1  1 1998   NA 30.0 21.7 NA    NA    NA    NA    NA
## 2  1  2 1998   8.2 35.6 21.8 NA    NA    NA    NA    NA
## 3  2  2 1998  51.0 31.8 21.8 NA 25.075 23.85 89.75  0.125
## 4  3  2 1998   0.6 35.4 21.5 NA 24.975 23.00 86.50  0.125
## 5  4  2 1998   0.0 35.6 22.1 NA 26.650 23.90 80.00  0.275
## 6  5  2 1998   0.0 36.4 22.5 NA 26.950 24.00 78.50  0.325

tail(roo)

##      dd mm  ano Prec Tmax Tmin  n    Tbs    Tbu     UR Vvento
## 4332 29 10 2010   0.0 36.8 18.9 NA    NA    NA    NA    NA
## 4333 30 10 2010   0.0 36.4 24.8 NA    NA    NA    NA    NA
## 4334 31 10 2010  49.0 33.6 22.5 NA    NA    NA    NA    NA
## 4335  1 11 2010  21.6 30.7 21.5 NA    NA    NA    NA    NA
## 4336  2 11 2010   0.4 32.6 19.9 NA    NA    NA    NA    NA
## 4337  3 11 2010   0.0    NA 17.3 NA    NA    NA    NA    NA

str (roo)

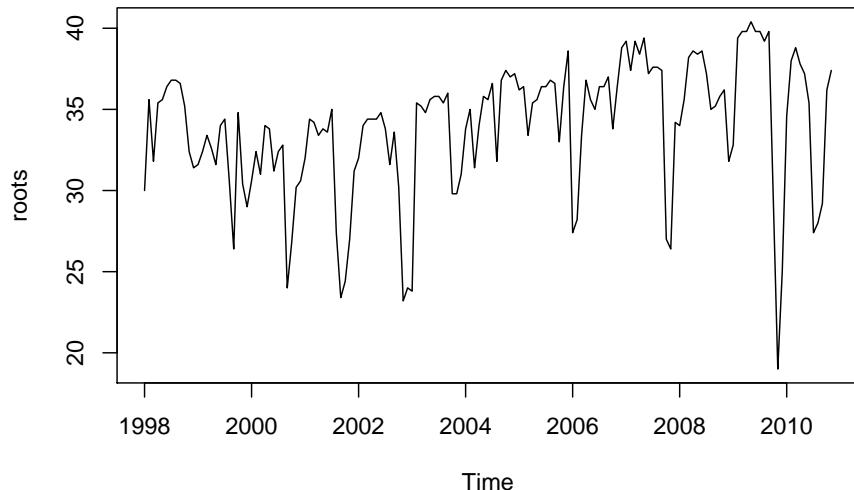
## 'data.frame':  4337 obs. of  11 variables:
## $ dd : int  1 1 2 3 4 5 6 7 8 9 ...
## $ mm : int  1 2 2 2 2 2 2 2 2 ...
## $ ano : int  1998 1998 1998 1998 1998 1998 1998 1998 1998 ...
## $ Prec : num  NA 8.2 51 0.6 0 0 2.4 NA 0.8 ...
## $ Tmax : num  30 35.6 31.8 35.4 35.6 36.4 36.8 36.8 36.6 35.2 ...
## $ Tmin : num  21.7 21.8 21.8 21.5 22.1 22.5 23.5 23.5 24.3 22.9 ...
## $ n : num  NA NA NA NA NA NA NA NA NA ...
## $ Tbs : num  NA NA 25.1 25 26.6 ...
## $ Tbu : num  NA NA 23.9 23 23.9 ...
## $ UR : num  NA NA 89.8 86.5 80 ...
## $ Vvento: num  NA NA 0.125 0.125 0.275 0.325 0.2 0.175 0.15 0.25 ...
```

A saída produzida por `str()` indica que o R não visualiza esse conjunto de dados como um objeto de série temporal. Precisamos transformar para uma série temporal. Vamos converter os dados para uma série temporal fixando apenas a temperatura do ar máxima. A função `ts` significa série temporal:

```
roots=ts(roo$Tmax,c(1998,1),c(2010,11),12)
```

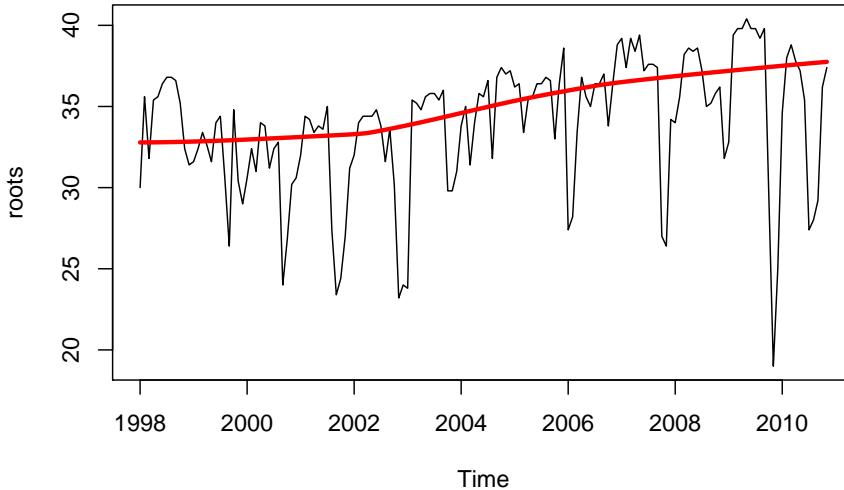
Antes de implementar o teste Mann-Kendall de tendência para a série temporal da Temperatura do ar máxima, devemos visualizar esta série usando o comando R abaixo:

```
plot(roots)
```



O gráfico de séries temporais produzido por R revela a presença de uma tendência ascendente nos níveis de temperatura do ar máxima diária para Rondonópolis ao longo do período de interesse. Para ver melhor essa tendência, vamos ajustar uma curva não paramétrica aos dados usando a função `lowess()` em R:

```
{plot(roots)
lines(lowess(time(roots),roots),lwd=3, col=2)}
```



Iremos aplicar o teste de Mann-Kendall “como está” (ou seja, sem quaisquer correções para autocorrelação) usando o comando R:

```
res <- MannKendall(roots)
summary(res)

## Score = 3821 , Var(Score) = 417409.7
## denominator = 11838.61
## tau = 0.323, 2-sided pvalue =< 2.22e-16

mk.test(roots)

##
##  Mann-Kendall trend test
##
##  data: roots
##  z = 5.9126, n = 155, p-value = 3.366e-09
##  alternative hypothesis: true S is not equal to 0
##  sample estimates:
##          S      varS      tau
##  3.821000e+03 4.174097e+05 3.227575e-01
```

A saída deste teste produzido por R é concisa e relata apenas o valor de tau (ie, estatística tau de Kendall) e o valor p para testar as hipóteses “ H_0 : sem tendência” versus “ H_a : tendência monotônica (para cima ou para baixo)”: Para o nosso

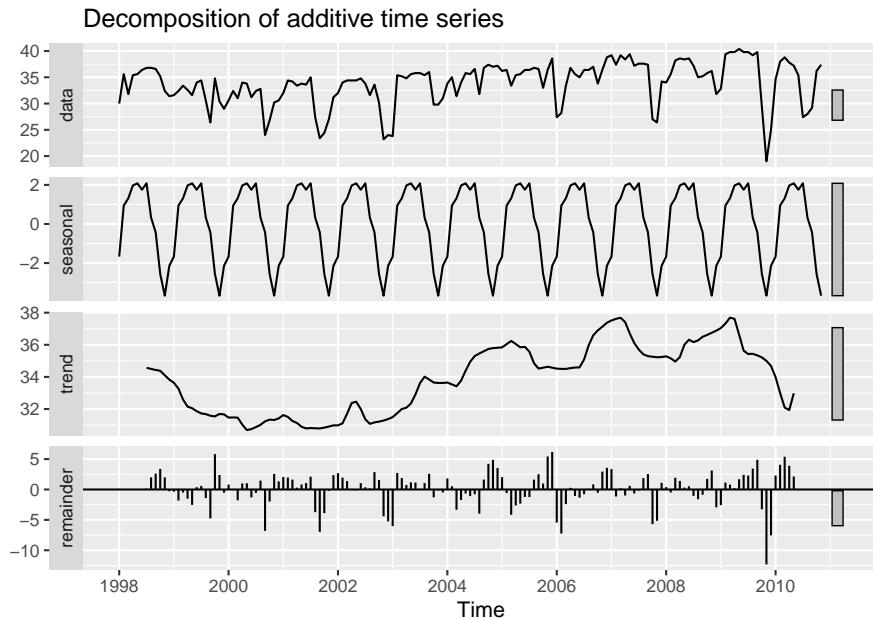
exemplo, o valor de p associado ao teste de Mann-Kendall é estatisticamente significativo, sugerindo a presença de uma tendência ascendente estatisticamente significativa na série temporal de temperatura máxima diária. Vamos calcular a inclinação da reta (isto é, taxa de variação linear) que intercepta de acordo com o método de Sen:

```
sens.slope(roots)
```

```
##  
## Sen's slope  
##  
## data: roots  
## z = 5.9126, n = 155, p-value = 3.366e-09  
## alternative hypothesis: true z is not equal to 0  
## 95 percent confidence interval:  
## 0.02714286 0.04883721  
## sample estimates:  
## Sen's slope  
## 0.03870968
```

Podemos plotar a composição da série de dados:

```
library(ggfortify)  
  
autoplot(decompose(roots))
```



Organizar os dados com o ano e a temperatura do ar máxima:

```
library(forecast)
library(dplyr)

s=select(roo, ano, Tmax)
head(s)

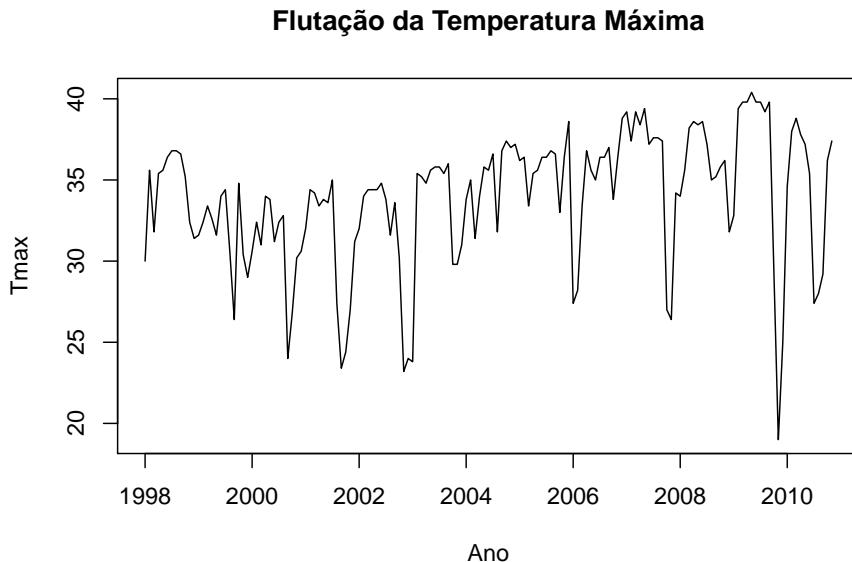
##      ano Tmax
## 1 1998 30.0
## 2 1998 35.6
## 3 1998 31.8
## 4 1998 35.4
## 5 1998 35.6
## 6 1998 36.4
```

Converter os dados em série temporal:

```
roots=ts(s,c(1998,1),c(2010,11),12)
```

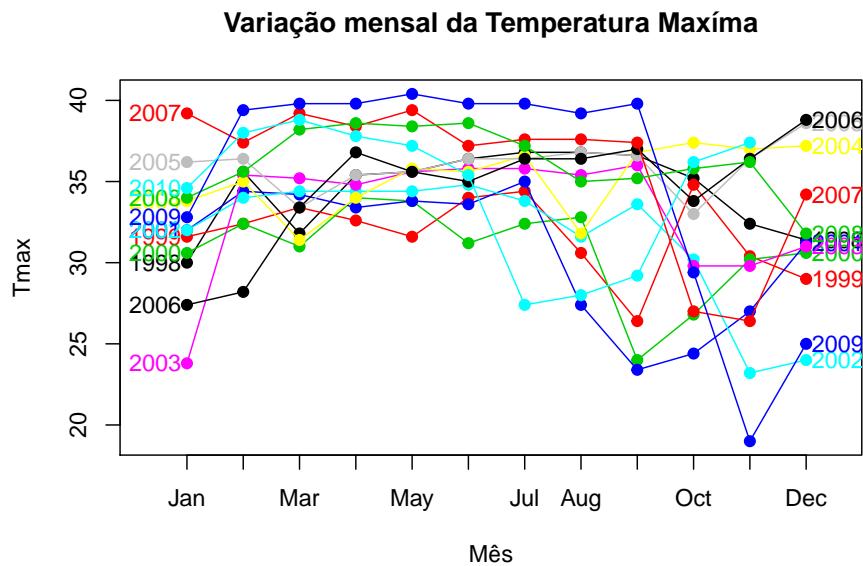
Flutuação da Temperatura Máxima:

```
plot.ts(roots[,2], main = "Flutuação da Temperatura Máxima", xlab = "Ano", ylab = "Tmax")
```



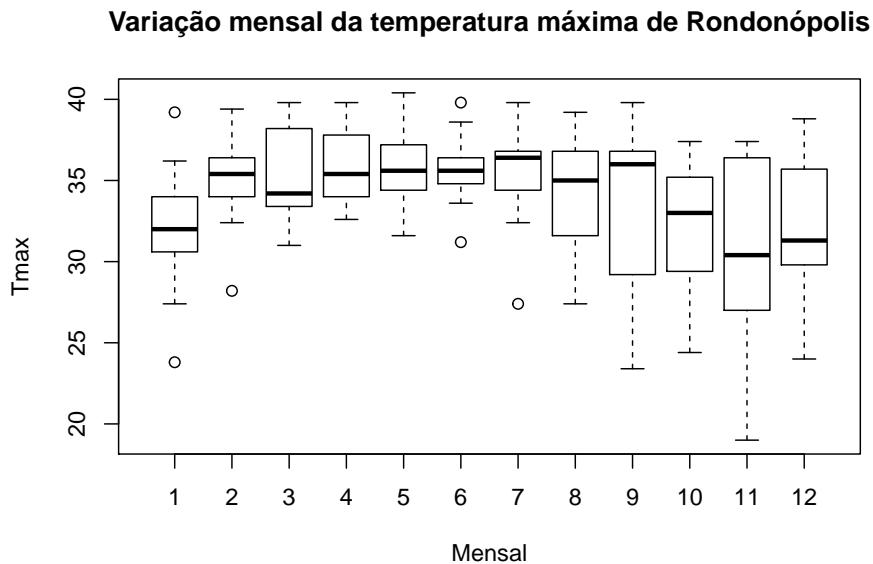
Variação mensal da Temperatura Maxíma:

```
seasonplot(roots[,2], year.labels = TRUE, year.labels.left=TRUE, col=1:40, pch=19, main=
```



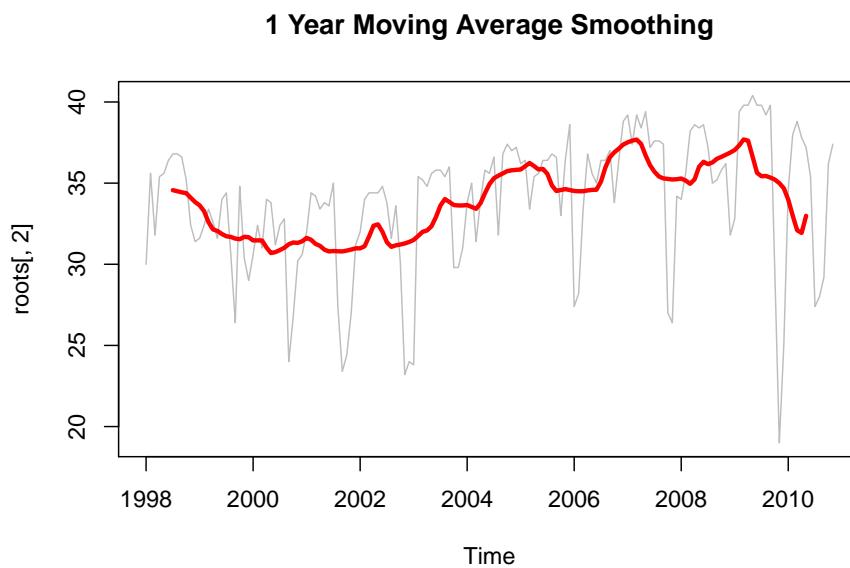
Variação mensal da temperatura máxima de Rondonópolis – boxplot:

```
boxplot(roots[,2] ~ cycle(roots[,2]), xlab = "Mensal", ylab = "Tmax", main = "Variação
```

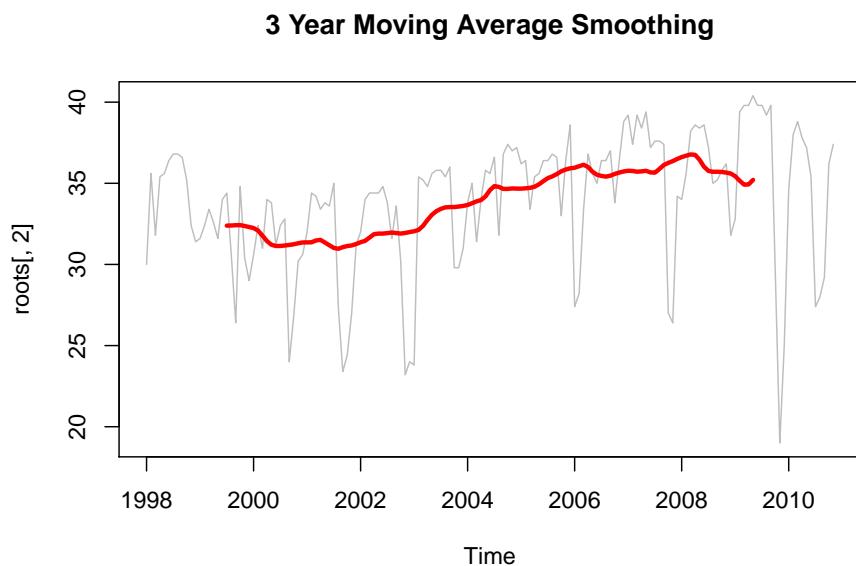


Movendo suavização média para ver a tendência:

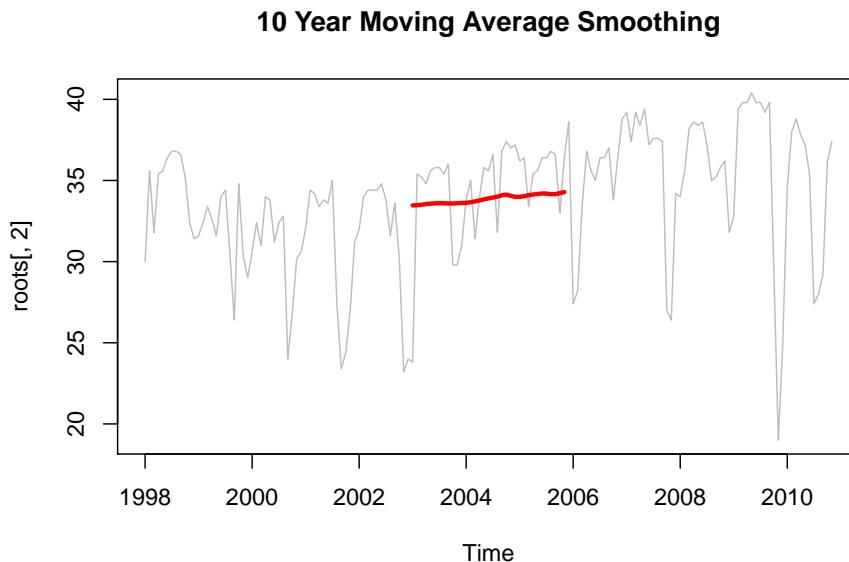
```
{plot(roots[,2], col="gray", main = "1 Year Moving Average Smoothing")
lines(ma(roots[,2], order = 12), col = "red", lwd=3)}
```



```
{plot(roots[,2], col="gray", main = "3 Year Moving Average Smoothing")
lines(ma(roots[,2], order = 36), col = "red", lwd=3)}
```



```
{plot(roots[,2], col="gray", main = "10 Year Moving Average Smoothing")
lines(ma(roots[,2], order = 120), col = "red", lwd=3)}
```



Previsão de dados:

```

mediamovel = ma(roots[,2], order = 2)
print(mediamovel)

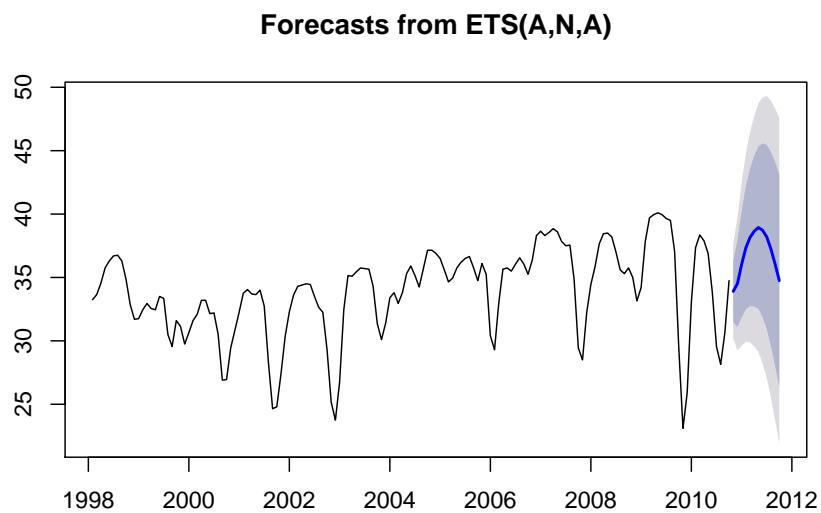
##           Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 1998      NA 33.25 33.65 34.55 35.75 36.30 36.70 36.75 36.30 34.85 32.85 31.70
## 1999 31.75 32.45 32.95 32.55 32.45 33.50 33.35 30.50 29.55 31.60 31.15 29.75
## 2000 30.65 31.60 32.10 33.20 33.20 32.15 32.20 30.50 26.90 26.95 29.45 30.85
## 2001 32.25 33.75 34.05 33.70 33.65 34.00 32.75 28.30 24.65 24.80 27.40 30.35
## 2002 32.30 33.60 34.30 34.40 34.50 34.45 33.50 32.65 32.25 29.30 25.15 23.75
## 2003 26.75 32.45 35.15 35.10 35.45 35.75 35.70 35.65 34.30 31.35 30.10 31.40
## 2004 33.40 33.80 32.95 33.80 35.30 35.90 35.15 34.25 35.70 37.15 37.15 36.90
## 2005 36.50 35.60 34.65 34.95 35.75 36.20 36.50 36.65 35.75 34.75 36.10 35.25
## 2006 30.40 29.30 32.95 35.65 35.75 35.50 36.05 36.55 36.05 35.25 36.35 38.30
## 2007 38.65 38.30 38.55 38.85 38.60 37.85 37.50 37.55 34.85 29.45 28.50 32.20
## 2008 34.45 35.85 37.65 38.45 38.50 38.20 37.00 35.60 35.30 35.75 35.00 33.15
## 2009 34.20 37.85 39.70 39.95 40.10 39.95 39.65 39.50 37.05 29.40 23.10 25.90
## 2010 33.05 37.35 38.35 37.90 36.90 33.85 29.55 28.15 30.65 34.75      NA

previsao = forecast(mediamovel, h=12)

## Warning in ets(object, lambda = lambda, biasadj = biasadj,
## allow.multiplicative.trend = allow.multiplicative.trend, : Missing values
## encountered. Using longest contiguous portion of time series

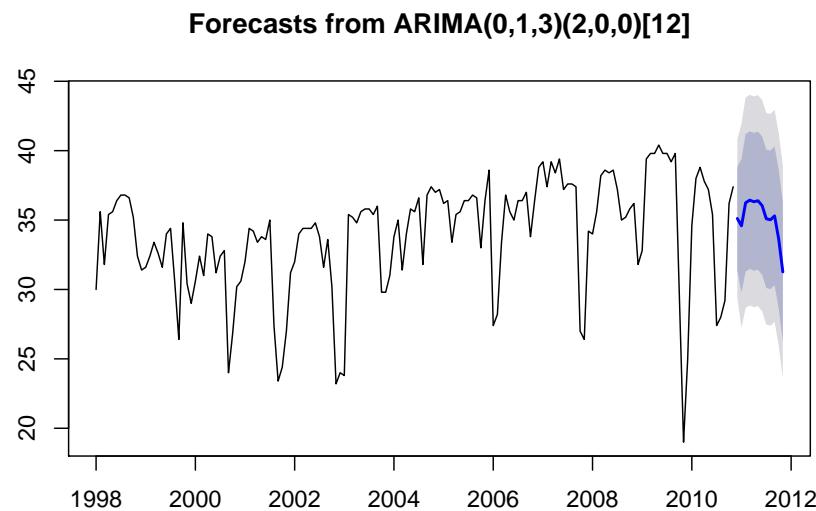
```

```
plot(previsao)
```



Previsão ARIMA:

```
arima = auto.arima(roots[,2])
previsao = forecast(arima, h=12)
plot(previsao)
```



10.5.1 Referência

<http://www.ghement.ca/Mann-Kendall%20Trend%20Test%20in%20R.doc>
<https://cran.r-project.org/web/packages/trend/trend.pdf>

Chapter 11

Sensoriamento remoto

11.1 Imagens de RPAs

Este capítulo oferece uma introdução ao processamento e classificação de imagens de RPAs no ambiente R, usando algoritmos de aprendizado capazes de realizar diversos processamentos. Ele também fornece um tutorial de referência conciso e prático, que oferece aos leitores uma visão geral do que é possível realizar no Sistema R com o processamento e classificação de imagens de RPAs.

É necessário instalar alguns pacotes necessários para aplicar esta tarefa:

```
install.packages("devtools", dependency=TRUE)
library(devtools)
install_github("filipematiass23/FIELDimageR")
install.packages("sp", dependency=TRUE)
install.packages("raster", dependency=TRUE)
install.packages("rgdal", dependency=TRUE)
```

Carregar os pacote que foram baixados:

```
library(FIELDimageR)
library(raster)
```

Baixar a imagem ortomosaicada:

```
EX1<-stack("https://www.dropbox.com/s/qxg6fn9ljiy35ssm/EX1_RGB.tif?dl=1")
```

Plotar a imagem nas bandas RGB:

```
plotRGB(EX1, r = 1, g = 2, b = 3)
```

Remover o solo e trabalhar apenas com a vegetação para aplicar os índices de vegetação RGB:

```
EX1.RemSoil<- fieldMask(mosaic = EX1, Red = 1, Green = 2, Blue = 3, index = "HUE")
```

Aplicado os índices de vegetação. Aplicaremos o índice NGRDI, BGI e podemos criar um índice usando as bandas disponíveis. Criaremos como exemplo `myIndex` com a fórmula `Red-Blue/Green`:

```
EX1.Indices<- indices(mosaic = EX1.RemSoil$newMosaic, Red = 1, Green = 2, Blue = 3,
                        index = c("NGRDI", "BGI"), myIndex = c("(Red-Blue)/Green"))
```

11.2 Imagens de satélites

Esse procedimento será realizado com imagens de satélite (Sentinel 2), porém pode ser aplicado com imagens de RPA, desde que sejam multiespectrais.

Carregar pacotes necessários para trabalhar com os dados raster. Caso não tenha algum dos pacotes, realize a sua instalação:

```
library(raster)
library(knitr)
library(sp)
library(rgdal)
library(ggplot2)
library(viridis)
library(rasterVis)
library(LSRS)
```

Baixar o arquivo . sentinel2.tif.

```
imagesentinel=raster('D:/livro/TudodoRa/sentinel2.tif')
```

```
imagesentinel=raster('sentinel2.tif')
```

Visualizar os dados:

```
imagesentinel
```

É necessário criar camadas individuais para cada uma das bandas espetrais:

```
b1 <- raster('sentinel2.tif', band=1)
b2 <- raster('sentinel2.tif', band=2)
b3 <- raster('sentinel2.tif', band=3)
b4 <- raster('sentinel2.tif', band=4)
b5 <- raster('sentinel2.tif', band=5)
b6 <- raster('sentinel2.tif', band=6)
b7 <- raster('sentinel2.tif', band=7)
b8 <- raster('sentinel2.tif', band=8)
b9 <- raster('sentinel2.tif', band=9)
b10 <- raster('sentinel2.tif', band=10)
b11 <- raster('sentinel2.tif', band=11)
b12 <- raster('sentinel2.tif', band=12)
```

Comparar duas bandas para ver se elas possuem a mesma extensão:

```
compareRaster(b2, b3)
```

Plotar a banda 4 para pré-visualização:

```
plot(b4)
```

```
image(b4)
```

Visualizar a imagem nas bandas do RGB:

```
RGB <- stack(list(b4, b3, b2))
plotRGB(RGB, axes = TRUE, stretch = "lin", main = "Sentinel RGB colour composite")
```

Juntar todas as bandas num só arquivo:

```
t <- stack(b1,b2, b3, b4, b5, b6, b7, b8, b9, b10, b11, b12)
st <- brick('sentinel2.tif')
plot(st)
```

Aplicar o índice de vegetação NDVI, para o Sentinel 2 com: NIR = 8, red = 4.

Criar a VI (vegetation index) por meio de função:

```
VI <- function(img, k, i) {
  bk <- img[[k]]
  bi <- img[[i]]
  vi <- (bk - bi) / (bk + bi)
  return(vi)
}
```

NDVI:

```
ndvi <- VI(st, 8, 4)
plot(ndvi, col = rev(terrain.colors(10)), main = "Sentinel2-NDVI")
```

Outras fórmula de aplicar o NDVI

```
vi2 <- function(x, y) {
  (x - y) / (x + y)
}
ndvi2 <- overlay(st[[8]], st[[4]], fun=vi2)
plot(ndvi2, col=rev(terrain.colors(10)), main="Sentinel2-NDVI")
```

Visualizar o NDVI em histograma

```
hist(ndvi,
      main = "Distribuição dos valores de NDVI",
      xlab = "NDVI",
      ylab= "Frequência",
      col = "wheat",
      xlim = c(-0.5, 1),
      breaks = 30,
      xaxt = 'n')
axis(side=1, at = seq(-0.5,1, 0.05), labels = seq(-0.5,1, 0.05))
```

Visualizar apenas a vegetação com NDVI acima de 0.4:

```
veg <- reclassify(ndvi, cbind(-Inf, 0.4, NA))
plot(veg, main='Vegetação')
```

Reclassificar o NDVI e defini-lo por classes numéricas:

```
vegc <- reclassify(ndvi, c(-Inf,0.25,1, 0.25,0.3,2, 0.3,0.4,3, 0.4,0.5,4, 0.5,Inf, 5))
plot(vegc,col = rev(terrain.colors(4)), main = 'NDVI reclassificado')
```

Criar uma classificação não supervisionada a partir do NDVI:

Converter o raster (NDVI) a um vetor/matriz:

```
nr <-getValues(ndvi)
str(nr)
```

É importante definir o gerador de pontos, porque o “*kmeans*” inicia os centros em locais aleatórios:

```
set.seed(99)
```

Criar 10 clusters, permitir 500 iterações, comece com 5 conjuntos aleatórios usando o método Lloyd:

```
kmncluster <- kmeans(na.omit(nr), centers = 10, iter.max = 500,
                      nstart = 5, algorithm = "Lloyd")
```

Ver o vetor/matriz:

```
str(kmncluster)
```

Crie uma cópia do NDVI para não perder os dados:

```
knr <- ndvi
```

Agora substitua os valores das células de varredura pelo `kmncluster$cluster`:

```
knr[] <- kmncluster$cluster
```

Realize o plot do NDVI e do kmeans:

```
par(mfrow = c(1, 2))
plot(ndvi, col = rev(terrain.colors(10)), main = "NDVI")
plot(knr, main = "Kmeans", col = viridis_pal(option = "D")(10))
```

Se quiser traçar a classificação kmeans ao lado da renderização do RGB para verificar a qualidade da classificação e identificação das classes:

```
par(mfrow = c(1, 2))
plotRGB(RGB, axes = FALSE, stretch = "lin", main = "RGB")
plot(knr, main = "Kmeans", yaxt = 'n', col = viridis_pal(option = "D")(10))
```

Aplicar outros índices de vegetação com o pacote LSRS:

```
NDVI=NDVI(b8,b4)
SAVI=SAVI(b8,b4)
TGSI=TGSI(b4,b2,b3)
MSAVI=MSAVI(b8,b4, Pixel.Depth=1)
EVI=EVI(b8,b4,b2,Pixel.Depth=1)
NBR=NBR(b8,b11)
```

```
par(mfrow = c(3, 2))
plot(NDVI,lwd=4,main="NDVI",xlab="easting", ylab="northing")
plot(SAVI,lwd=4,main="SAVI",xlab="easting", ylab="northing")
plot(TGSI,lwd=4,main="TGSI",xlab="easting", ylab="northing")
plot(MSAVI,lwd=4,main="MSAVI",xlab="easting", ylab="northing")
plot(EVI,lwd=4,main="EVI",xlab="easting", ylab="northing")
plot(NBR,lwd=4,main="NBR",xlab="easting", ylab="northing")
```

11.3 Curvas de nível e modelo 3D a partir do Modelo Digital de Elevação

Primeiro carregar os pacotes necessários:

```
library(raster)
library(plot3D)
```

Carregar o dado raster do pacote Raster (Volcano) para ser usado como exemplo:

```
filled.contour(volcano, color.palette = terrain.colors)
```

Criar as curvas de nível:

```
cont <- contourLines(volcano)
fun <- function(x) x$level
LEVS <- sort(unique(unlist(lapply(cont, fun))))
COLS <- terrain.colors(length(LEVS))
```

Plotar somente as curvas de nível:

```
contour(volcano)
```

Plotar o modelo 3D com curvas de nível:

```
x <- seq(1, nrow(volcano), by = 3)
y <- seq(1, ncol(volcano), by = 3)
Volcano <- volcano [x, y]
```

Exemplo 1:

```
ribbon3D(z = Volcano, contour = TRUE, zlim= c(-100, 200),image = TRUE)
```

Exemplo 2:

```
persp3D(z = Volcano, contour = TRUE, zlim= c(-200, 200), image = FALSE)
```

Exemplo 3:

```
persp3D(z = Volcano, x = x, y = y, scale = FALSE,contour = list(nlevels = 20, col = "red"),zlim =
```

Exemplo 4:

```
persp3D(z = Volcano, contour = list(side = c("zmin", "z", "350")), zlim = c(-100, 400), phi = 20,
```

Exemplo 5:

```
persp3D(z = volcano, shade = 0.3, col = gg.col(100))
```

11.4 LIDAR

O exemplo a seguir é um processamento de imagens LIDAR, por meio da qual será segmentado árvores individuais e métricas.

Carregar pacote:

```
library(lidR)
library(raster)
```

Baixar dados Example.las

Carregar uma área florestal de exemplo a partir de imagens LIDAR que será trabalhada:

```
las = readLAS('C:/Users/Jefferson/Dropbox/Livro/Example.las')
plot(las)
```

O lasground fornece vários algoritmos para classificar os pontos de referência. Essa função é conveniente para gráficos de pequeno a médio porte, como o que estamos processando.

Classificar pontos do solo (pontos de referência):

```
las = lasground(las, csf())
plot(las, color = "Classification")
```

É necessário definir o terreno em 0 metros. Deve-se subtrair o MDT para obter pontos de aterrramento em 0, mas aqui não será usado um MDT, mas vamos interpolar exatamente cada ponto.

Definir altura normalizada:

```
las = lasnormalize(las, tin())
plot(las)
```

Na próxima etapa, será usado um algoritmo que requer um modelo de altura da copa a partir da nuvem de pontos.

Calcular um modelo de altura das copas:

```
algo = pitfree(thresholds = c(0,10,20,30,40,50), subcircle = 0.2)
chm = grid_canopy(las, 0.5, algo)
```

Plotar o CHM:

```
plot(chm, col = height.colors(50))
```

A segmentação pode ser alcançada com `lastrees`. Aqui foi escolhido o algoritmo de bacia com um limiar de 4 metros. A nuvem de pontos foi atualizada e cada ponto agora tem um número que se refere a uma árvore individual (treeID). Pontos que não são árvores recebem o valor de ID NA.

Realizar a segmentação das árvores:

```
algo = watershed(chm, th = 4)
las = lastrees(las, algo)
```

Remova os pontos que não estão atribuídos a uma árvore:

```
trees = lasfilter(las, !is.na(treeID))
```

Plotar a segmentação:

```
plot(trees, color = "treeID", colorPalette = pastel.colors(100))
```

Calcular algumas métricas:

```
hulls = tree_hulls(las, func = .stdmetrics)
spplot(hulls, "zmax")
```

No exemplo anterior, mesmo se a segmentação for feita usando um modelo de altura do dossel, a classificação foi feita na nuvem de pontos. Isso ocorre porque `lidR` é uma biblioteca orientada à nuvem de pontos. Mas pode-se querer que o raster trabalhe com rasters. Nesse caso, a função divisor de águas pode ser usada de forma independente:

```
crowns = watershed(chm, th = 4)()
plot(crowns, col = pastel.colors(100))
```

Criar polígonos de contornos a partir da copa:

```
contour = rasterToPolygons(crowns, dissolve = TRUE)
```

Plotar o CHM e contornos:

```
plot(chm, col = height.colors(50))
plot(contour, add = T)
```

11.5 Modelo Digital de Elevação MDE

Carregar pacotes:

```
library(raster)
```

Carregar dados e plotar:

```
MDE=raster("https://www.dropbox.com/s/b2rzimq500rmj5o/MDE.tif?dl=1")
plot(MDE)
```

Usar a função `terrain()` para calcular/extrair algumas informações topográficas:

```
Declividade <- terrain(MDE, "slope")
Aspecto <- terrain(MDE, "aspect")
TPI <- terrain(MDE, "TPI") # Topographic Position Index (Índice de posição topográfica)
TRI <- terrain(MDE, "TRI") # Terrain Ruggedness Index (Índice de robustez do terreno)
Rugosidade <- terrain(MDE, "roughness")
Escoamento <- terrain(MDE, "flowdir")
Hillshade <- hillShade(Declividade, Aspecto, angle=45, direction=0, filename='', normalize=FALSE)
```

Juntar todos os dados com a função `stack()`:

```
topo <- stack(MDE, Declividade, Aspecto, TPI, TRI, Rugosidade, Escoamento, Hillshade)
```

Renomear os dados para aparecerem no plot:

```
names(topo) <- c("MDE", "Declividade", "Aspecto", "TPI", "TRI", "Rugosidade", "Escoamento")
```

Plotar os dados:

```
plot(topo)
```