UNIVERSIDADE FEDERAL DO CEARÁ
Centro de Ciências
Departamento de Estatística e Matemática Aplicada

Introdução ao R

Eduardo Janderson da Cunha Meneses
Francisco Hildemar Calixto de Alencar

Fortaleza, março de 2012

Conteúdo

Capítulo 1

Introdução

O software livre R é bastante utilizado durante todo o curso de graduação em Estatística da Universidade Federal do Ceará devido ser um programa didático e específico para o uso estatístico.

Com isso é de grande relevância que os alunos recém ingressos no curso de Estatística se familiarizem com este programa, isto facilitará muito o percurso dos alunos no decorrer do curso.

1.1 Objetivos do minicurso

- Objetivo Geral
  Familiarizar os alunos recém ingressos no curso de Estatística da Universidade Federal do Ceará com do software R no uso da matemática do ensino médio e no uso da Estatística básica.

- Objetivo Específico
  Estimular a busca de conhecimento sobre a utilização do software livre R.

1.2 O que é o R?

R é um ambiente de software livre para computação estatística e gráficos. Compila e corre em uma variedade larga de plataformas de UNIX, Windows e MacOS. Entre outras características permite:

- manipulação e armazenamento dos dados,

- operadores para cálculo, incluindo cálculo matricial,

- uma vasta, coerente e integrada coleção de ferramentas para análise de dados,
- capacidades gráficas para análise exploratória de dados.

R também é uma linguagem de programação orientada a objetos criada em 1996 por Ross Ihaka e Robert Gentleman que aliada a um ambiente integrado permite a manipulação de dados, realização de cáculos e geração de gráficos. Além dos procedimentos estatísticos o R permite operações matemáticas simples, e manipulação de vetores e matrizes. Assim como confecção de diversos tipos de gráficos.

Para as suas diversas aplicações e funções o programa R possui pacotes específicos para executalos. Os pacotes contêm um conjunto de funções que facilitam ou possibilitam a realização das análises estatísticas, além de possuírem ajuda para suas funções e, até mesmo, demonstrações de execução.
Quando se instala o R, apenas alguns pacotes vêm juntos, os quais são essenciais para o funcionamento do programa, além de poderem servir de base para outros pacotes. No R, existe uma grande diversidade de pacotes.

Os pacotes extras podem ser encontrados com base no próprio programa ou pelo site, sendo que em ambos os casos, o usuário deve estar conectado à internet.

1.3 Interface do R

Ao iniciar o R abrirá automaticamente o Console que é a janela onde os comandos são digitados. Internamente ao Console, se encontra o prompt, conforme Figura 1 abaixo, que é um sinal indicador de que o programa está pronto para receber comando.

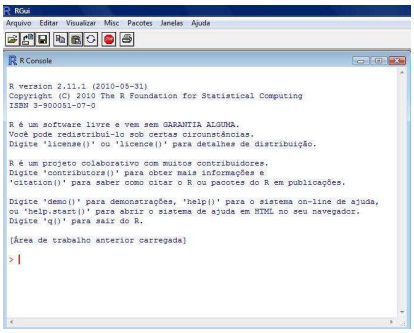


Figura 1.1 *Interface R*

1.4 Sintaxe do R

Tecnicamente o R é uma linguagem de expressões com regras e sintaxe muito simples. Faz distinção entre maiúsculas e minúsculas, de modo que os caracteres "A" e "a" são entendidos como sendo símbolos diferentes, referindo-se, portanto, a variáveis diferentes.

Os comandos ou ordens elementares consistem em expressões ou atribuições. Se uma ordem ou comando é uma expressão, o seu valor é calculado e visualizado sendo perdido em seguida. Uma atribuição, ao contrário, calcula a expressão e atribui o resultado que não é mostrado automaticamente, apenas é salvo no endereço de alguma variável que está sendo usada no R.

Os comandos são separados por ponto e vírgula (";") ou são inseridos em nova linha. Podem agrupar-se dentro de chaves ("...") vários comandos elementares numa expressão mais complexa. Se ao terminar uma linha, o comando não está sintaticamente completo o R mostra o símbolo "+" que é o comando de continuação do comando inicial.

1.5 Histórico do R

O R começou a ser desenvolvido por Robert Gentleman e Ross Ihaka do Departamento de Estatística da Universidade de Auckland em Nova Zelândia, mais conhecidos por "R & R", apelido do qual originou-se o nome R do programa. O objetivo inicial de "R & R", em 1991, era produzir um software para as suas aulas de laboratório baseado na já revolucionária linguagem S, utilizada

pelo software comercial S-Plus criado por Jonh M. Chambers da AT&T que atualmente vem contribuindo para o aperfeiçoamento e ampliação das análises estatísticas do R.

O primeiro relato da distribuição do R foi em 1993, quando algumas cópias foram disponibilizadas no StatLib, um sistema de distribuição de softwares estatísticos. Com o incentivo de um dos primeiros usuários deste programa, Martin Mächler do ETH Zürich (Instituto Federal de Tecnologia Zurique da Suíça), "R & R", em 1995, lançaram o código fonte do R, disponível por ftp (uma forma de se transferir dados pela internet), sobre os termos de Free Software Foundations GNU general license, que seria um tipo de "licença para softwares livres".

Em 1997 foi formado um grupo de profissionais que têm acesso ao código fonte do R, possibilitando assim a atualização mais rápida do software. Desde então o R vem ganhando cada vez mais adeptos em todo o mundo, em parte devido ao fato de ser totalmente gratuito e também por ser um programa que exigi do usuário o conhecimento das análises que está fazendo, diminuindo assim as chances de uma interpretação errada dos resultados.

Outro fato importante para a difusão do R é a sua compatibilidade com quase todos os sistemas operacionais. O R está disponível para a maior parte dos MacOS, Windows a partir do 95 e para UNIX e sistemas similares como Linux e FreeBSD.

Figura 1.2 *Robert Gentleman*

Figura 1.3 *Ross Ihaka*

1.6 Baixando e instalando o R

1.6.1 Baixando o programa R para o Windows

Para fazer o download do R, é necessário acessar o site www.r-project.org, clicar em CRAN (Figura 4) e escolher um servidor, de preferência no seu país e mais próximo da sua cidade(Figura 5).

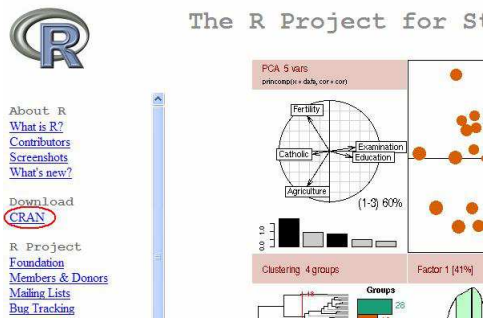


Figura 1.4 *CRAN*

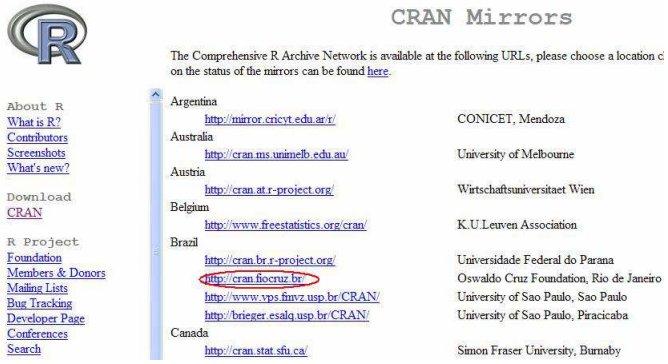


Figura 1.5 *Escolha do servidor*

Agora siga os seguintes passos:

1. Clique em Windows;
2. Clique em "base" para ter acesso à página de onde será feito o download;
3. Clique em "R- número da versão do R disponível naquele momento"(Figura 6).

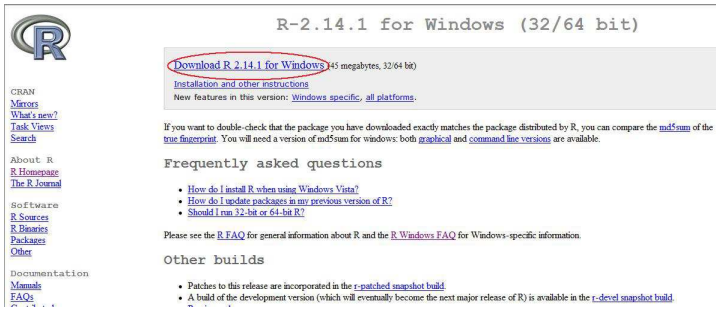


Figura 1.6 *R- número da versão do R disponível naquele momento*

Por último aparecerá uma janela, onde se pode executar ou salvar o programa (Figura 7). A forma mais segura de se fazer a instalação é salvar em uma pasta e depois executar o instalador a partir dela, já que o processo de download pode ser demorado.

Pronto seu programa R já foi baixado.

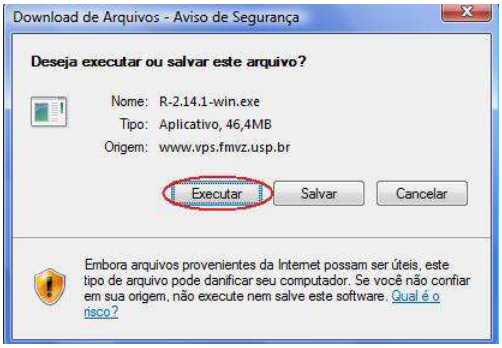


Figura 1.7 *Executar*

1.6.2 Instalando o R no Windows

Após o término do download, deve-se dar dois cliques com o botão esquerdo do mouse no arquivo baixado (instalador) para começar a instalação, abaixo estão os passos para a instalação:

1. Caso apareça um aviso de segurança, clique em executar.
2. Escolha a língua de sua preferência e clique em OK.
3. Na janela "R for Windows número da versão do programa R", clique em avançar até o fim.

4. Após o termino da instalação , aparacerá uma janela de finalização do instalador. Nela, clique em comcluir.

A partir daí, o R já pode ser usado.

Capítulo 2

Como Utilizar o R

2.1 Primeiro Contato

Logo após a instalação inicie o R, para isso basta clicar no ícone que foi criado na área de trabalho do seu computador.

Figura 2.1 *Ícone R criado na área de trabalho*

após clicar será aberta a seguinte janela:

Com o R iniciando você verá o símbolo > em vermelho, que é o *prompt* do R indicando que o sistema está pronto para receber seus comandos.

Acima do *prompt* em cor azul encontram-se algumas informações sobre o sistema e alguns comandos básicos. Para poder limpar o conteúdo que aparece na tela do console basta combinar as teclas "Ctrl"e "L".

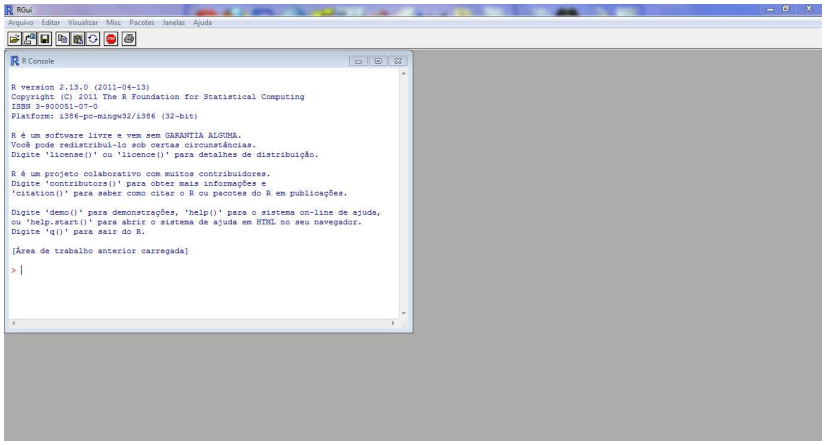


Figura 2.2 *Interface R*

Nesta janela é possível digitar e executar comandos, entretanto, a melhor forma de se editar comandos é usando o editor de *script*. Para ativar o editor, basta ir até a barra de menus e clicar sequencialmente em: **Arquivo** e **Novo script**.

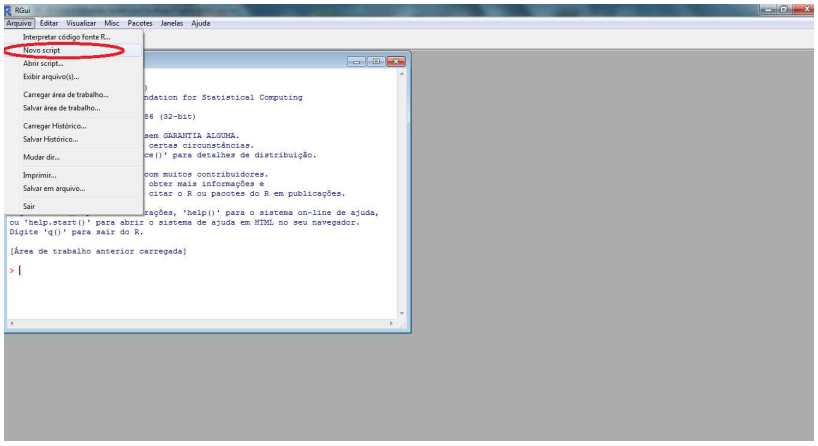


Figura 2.3 *Abrindo novo script*

No ambiente R as janelas podem ser organizadas por título (uma ao lado da outra) ou em forma de cascata. Para isso, basta ir até o menu Janelas e escolher uma dessas opções.

Para executar comandos no *script*, deixe o cursor na linha de comandos desejada e combine "Ctrl"e "R"; no caso de comandos ocupando várias linhas, selecione todas as desejadas e após isso faça a mesma combinação de teclas.

2.2 Pacotes

No R existe um grande diversiadade de pacotes. Estes pacotes contêm um conjunto de funções que permitem ou facilitam a realização de das análise estatísticas, além de possuírem ajuda para suas funções e demonstrações de execução.

Ao instalar o R, apenas alguns pacotes vêm juntos, os quais são essenciais para o funcionamento do programa e são denominados de módulo ou pacote básico, podemos citar pacotes como o *Stats* e o *base* que contêm funções básicas como plot(), mean(), matrix() entre outras, além de muitos poderem servir de base e pré-requisito para o funcionamento de outros pacotes.

Os pacotes extras podem ser encontrados com base no próprio programa ou pelo site, www.r-project.org, ressaltando que em ambos os caso, o usuário deve estar conectado a internet.

2.3 Tipos de Objetos

No R, a manipulação de dados é feita através de objetos, sob os quais estão definidas uma série de comandos que permitem, entre outras, operações de aritmética, armazenamento de dados, descrição e análise.

Os tipos básicos de objetos do R são:

- Vetores
- Matrizes e arrays
- data-frames
- funções

Os quatro primeiros tipos são objetos que armazenam dados e que diferem entre si na forma da armazenar e operar com os dados. O último (função) é um tipo objeto especial que recebe algum "input"e produz um "output".

2.3.1 Vetores

É o tipo de objeto mais básico e mais simples para armazenamento de dados no R.

```
> g1=5
> g1
[1] 5


> g1=5;g1
[1] 5
```

Analisaremos agora os seguintes comandos:

```
> g2=c(3,5,7);g2
[1] 3 5 7


> g3=1:10;g3
 [1]  1  2  3  4  5  6  7  8  9 10

> g4=rep(1,3);g4
[1] 1 1 1


> g5=rep(c(1,2),c(4,6));g5
 [1] 1 1 1 1 2 2 2 2 2 2
```

Podemos perceber que no R a atribuição de um valor ou conjunto de valores a um objeto é feita utilizando o símbolo "=".

2.3.2 Matrizes

Matrizes são objetos que possuem as mesmas propriedades dos vetores, diferindo apenas na dimensão.

```
matrix(um vetor, nrow=número de linhas, ncol=números de colunas, ...)
```

Esta função recebe um vetor e o organiza em uma matriz conforme indicado no argumento `byrow`. Se `byrow=FALSE` o vetor será organizados por colunas da matriz, senão `byrow=TRUE` o vetor será organizado por linhas.

```
> e1=matrix(1:9,nc=3);e1
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

> e2=matrix(1:9,nc=3,byrow=TRUE);e2
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

2.3.3 Arrays

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.

```
# array(vetor_de_dados, vetor_de_dimensões)

> ar1=array(1:20,c(3,3,2));ar1
, , 1
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

, , 2
     [,1] [,2] [,3]
[1,]   10   13   16
```

```
[2,]   11   14   17
[3,]   12   15   18
```

2.3.4 Data-Frames

Data-frames são objetos que podem armazenar elementos de mais de um tipo, diferentemente de vetores, matrizes e arrays, que forçam todos os seus elementos a serem de um mesmo tipo. sua estrutura é semelhante a de uma matriz, sendo que, aqui todas as colunas são tratadas individualmente.

2.3.5 Funções

As funções possuem uma lista de argumentos cujos valores, quando não padronizados, devem ser informados pelo usuário (input). O retorno (output) fornecido pelas funções podem ser um objeto de qualquer classe, ou um valor de qualquer natureza (cacter,numérico, etc.). No R, a maioria das funções são escritas com a própria linguagem R. para estas, é possível visualizar seu conteúdo digitando o nome da função desejada, por exemplo:

```
> matrix
function (data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
{
    if (is.object(data) || !is.atomic(data))
        data <- as.vector(data)
    .Internal(matrix(data, nrow, ncol, byrow, dimnames, missing(nrow),
        missing(ncol)))
}
<environment: namespace:base>
```

Podemos obter ajuda sobre qualquer função do R utilizando o símbolo "?". Por exemplo para obter informações sobre a função matrix, basta digitar `?matrix` e o help do R será disponibilizado.

2.4 Entrada de Dados

No R a entrada de dados pode ser feita na forma de vetores,matrizes,etc. Porém, é possível obter dados externos provenientes de outros programas.

Read.table()

Esta função permite importar dados nos formatos txt e csv (separado por vírgulas). sua sintaxe é:

```
read.table(file, header=False, sep= "", dec= ".",...)
```

- file: é onde deve ser informado o diretório onde está gravado o arquivo, também o nome e a extensão do mesmo (txt ou csv), tudo entre aspas. para definir um diretório padrão, basta seguir os seguintes passos: no console , vá até a barra de menus e clique sequencialmente em **Arquivo** e **Mudar dir...**. após isso, aparecerá uma caixa de diálogo pedindo para informar o diretório desejado. Se isso for feito, então basta informar o nome do arquivo com a extensão, pois o R já irá procurá-lo no diretório que foi definido.

- header: Se igual a TRUE a primeira linha do banco de dados será assumida como rótulo das variáveis. Se igual a FALSE os dados serão lidos a partir da primeira linha.

- sep: Informa ao R qual o separador de campos e valores que está sendo usado no arquivo a ser importando. Para arquivos csv é usado ponto e vírgula(;).

- dec: Informa ao R qual o separador de decimais que está sendo usado no arquivo. o *default* do R é dec=. portanto se no banco de dados a ser importado estiver sendo usado outro separador de decimais, este deverá ser informado aqui.

scan()

Esta função lê dados diretamente do console, ou seja, coloca o R em modo prompt onde o usuário deve digitar cada dado seguido da tecla ENTER. para

encerrar a entrada de dados basta digitar ENTER duas vezes consecutivas.

edit()

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

```
> ga=edit(data.frame())
```

será aberta uma planilha na qual os dados devem ser digitados. Quando terminar de entrar com os dados note que no canto superior direito da planilha existe um botão QUIT. Pressionando este botão a planilha será fechada e os dados serão gravados no objeto indicado (no exemplo acima no objeto ga).

Se você precisar abrir novamente planilha com os dados, para fazer correções e/ou inserir mais dados use o comando `fix()`. No exemplo acima voce digitaria `fix(ga)`.

Importando dados de outros programas

É possível ler dados diretamente de outros formatos que não seja texto ou csv. Isto em geral é mais eficiente e requer menos memória do que converter para formato texto. Há funções para importar dados diretamente de EpiInfo, Minitab, S-PLUS, SAS, SPSS, Stata, Systat e Octave. Além disto é comum surgir a necessidade de importar dados de planilhas eletrônicas. Muitas funções que permitem a importação de dados de outros programas são implementadas no pacote foreign.

```
> require(foreign)
```

A seguir algumas funções

- `read.spss()` Para dados do SPSS
- `read.dta()` para dados do STATA
- `read.S()` para arquivos do S-PLUS e `restore.data()` para "dumps"do S-PLUS

- `read.mtp()` para arquivos "Minitab Portable Worksheet"

Carregando dados já disponíveis no R

Para carregar conjuntos de dados que são já disponibilizados com o R use o comando `data()`. Por exemplo, abaixo mostramos como carregar o conjunto mtcars que está no pacote datasets.

```
>data(mtcars)
> head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

A função `data()` pode ainda ser usada para listar os conjutos de dados disponíveis. A primeira chamada a seguir lista os conjuntos de dados dos pacotes carregados. A segunda lista os conjuntos de dados de um pacote específico (no exemplo do pacote nlme).

```
> data()
> data(package = "nlme")
```

Capítulo 3

Matemática no R

O software livre R possui diversas funções e aplicações à matemática, desde operações básicas a coisas como funções trigonométricas e derivadas.
Faremos uma introdução a aplicação do programa R na matemática começando pelas operações básicas e no decorrer veremos algumas funções bem conhecidas da matemática elementar e faremos um breve passeio pelo cálculo, vendo como derivar usando o R.

3.1 Operações Básicas

3.2 Operações com Vetores

Há diversas operações númericas definidas sobre vetores. Vejamos alguns exemplos:

```
> z1=c(1,2,3);z1
[1] 1 2 3

> z1+3
[1] 4 5 6

> z1+4:6
[1] 5 7 9

> z1*4
[1]  4  8 12
```

```
> z1^(1:3)
[1]  1  4 27

> z2=rep(1,3);z2
[1] 1 1 1

> z2/z1
[1] 1.0000000 0.5000000 0.3333333

> round(z2/z1,2)
[1] 1.00 0.50 0.33
```

Pode-se perceber que no primeiro comando estamos criando um vetor com os valores (1,2,3), já no segundo comando estamos somando 3 a todos os elementos do vetor z1. em seguida, somamos cada elemento de z1 com cada elemento do vetor (3,4,5). No quarto comando, multiplicamos cada elemento de z1 por 4. Logo em seguida, elevamos o primeiro elemento de z1 a 1, o segundo a 2 e o terceiro a 3.

No sexto comando comando, através da função `rep()` estamos criando um vetor com os valores (1,1,1) e chamando o memso de z2. No comando seguinte estamos dividindo o vetor z2 por z1. No ultimo comando utilizamos a função `round()` para arredondarmos, no primeiro argumento desta função se coloca o valor ou vetor numérico, e o segundo argumento é o número de casas decimais que se deseja no valor ou valores.

3.3 Operações com Matrizes

Operações com matrizes Operações com matrizes são feitas diretamente assim como no caso de vetores. Existem diversas operações sobre matrizes e vamos apresentar apenas algumas aqui.

```
> m1=matrix((1:9),nc=3);m1
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

> m2=matrix(1:9,nc=3,byrow=TRUE);m2
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9

> m1+m2
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    6   10   14
[3,]   10   14   18

> m1*m2
     [,1] [,2] [,3]
[1,]    1    8   21
[2,]    8   25   48
[3,]   21   48   81
```

Nos dois primeiros comando criamos duas matrizes (m1 e m2) e logo após fizemos a soma de ambas, no quarto comando temos a multiplicação das duas matrizes elemento por elemento.

```
> m3=cbind(1:5,6:10,11:15);m3
     [,1] [,2] [,3]
[1,]    1    6   11
[2,]    2    7   12
[3,]    3    8   13
[4,]    4    9   14
[5,]    5   10   15

> t(m3)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    6    7    8    9   10
[3,]   11   12   13   14   15


> m3%*%m2
     [,1] [,2] [,3]
[1,]  102  120  138
[2,]  114  135  156
[3,]  126  150  174
[4,]  138  165  192
[5,]  150  180  210

> t(m2)%*%m2
     [,1] [,2] [,3]
[1,]   66   78   90
[2,]   78   93  108
[3,]   90  108  126
```

No primeiro comando criamos uma matriz utilizando a função `cbid()` que serve para concatenar vatores, podemos usar esta função para concatenarmos quantos vetores quisermos, e estes corresponderão às colunas da matriz que está sendo criada. No segundo comando foi utilizado a função `t()` que faz

transposição de matrizes.

No comando seguinte fizemos o produto matricial utilizando %*%. No exemplo temos o produto matricial de m3 e m2.por ultimo temos o produto matricial da transposta de m2 com m2. Esta operação é conhecida como produto cruzado e no R existe uma função específica para realizá-la.

```
> crossprod(m2)
     [,1] [,2] [,3]
[1,]   66   78   90
[2,]   78   93  108
[3,]   90  108  126
```

Podemos destacar também as funções `det()`, `solve()` e `eigen()` que retornam, respectivamente, o determinante, a inversa e os auto-valores e auto-vetores de uma matriz.

```
> m4=matrix(c(6,5,4,1,3,0,2,-1,1),nc=3);m4
     [,1] [,2] [,3]
[1,]    6    1    2
[2,]    5    3   -1
[3,]    4    0    1

> det(m4)
[1] -15

> solve(m4)
     [,1]        [,2]       [,3]
[1,] -0.2  0.06666667  0.4666667
[2,]  0.6  0.13333333 -1.0666667
[3,]  0.8 -0.26666667 -0.8666667
> eigen(m4)

$values
[1]  8.021597  2.676943 -0.698540
```

```
$vectors
           [,1]       [,2]       [,3]
[1,] -0.6895898  0.1176905 -0.3086003
[2,] -0.6083940 -0.9525448  0.6136865
[3,] -0.3928393  0.2807263  0.7267425
```

Seleção de Elementos: É possível selecionar elementos de uma matriz utilizando colchetes da seguinte forma: [,] onde antes da vírgula indica(m)-se a(s) linhas e depois a(s) colunas. Como exemplo utilizaremos a matriz m3 ateriormente citada.

```
> m3[1,2]
[1] 6

> m3[,2]
[1]  6  7  8  9 10

> m3[-2,]
     [,1] [,2] [,3]
[1,]    1    6   11
[2,]    3    8   13
[3,]    4    9   14
[4,]    5   10   15

> m3[4:5,1:2]
     [,1] [,2]
[1,]    4    9
[2,]    5   10
```

3.4 Funções

O R nos possibilita criar nossas próprias funções, que são genuinamente funções R, sendo armazenadas internamente e podendo ser utilizadas em novas expressões. O aprendizado em escrever funções úteis é uma das muitas

maneiras de fazer com que o uso do R seja confortável e produtivo.

Também vale ressaltar que o programa R possui além da capacidade de criar funções específicas conforme a necessidade do usuário possui muitas funções já definidas para atender algumas necessidades do usuário. Neste curso veremos algumas dessas funções conforme o nosso propósito.

3.5 Criando Funções no R

A sintaxe geral para criação de uma função é dada por:

```
> nome_da_função = function (argumento 1, ...) expressão
```

Temos do comando acima:

- `nome_da_função` é como o usuário chama a nova função criada e também o objeto que receberá a função.
- `(argumento 1, ...)` são as variáveis de trabalho da função.
- `expressão` é um grupo de comandos.

$Exemplo_1$: Criar uma função do segundo grau com nome $f1$, argumento igual a x e calcular seus valores para $2, 3$ e 10.
Criando a função:

```
> f1 = function ( x ) x^2 + 5*x + 6
```

Calculando seus valores:

```
> f1(2);f1(3);f(10)
[1] 20
[1] 30
[1] 156
```

Exemplo$_2$: Criar uma função com nome $f2$ que calcule as raízes da função $f1$.
Criando a função:

```
> f2 = function (a, b, c) {
+ delta = (b)^2 - 4*a*c
+ x1 = ((-b)+delta)/(2*a); x2 = ((-b)-delta)/(2*a)
+ print(c("x1"=x1,"x2"=x2))
}
```

Calculando as raízes:

```
>f2(1,5,6)
x1 x2
-2 -3
```

Neste último exemplo foram utilizados comandos que não serão abordados em nosso curso para demonstrar um pouco da extensão e poder do programa R. Também para estimular no aluno a busca por mais conhecimento neste vasto software.

3.6 Algumas Funções Específicas

Como citado anteriormente o R possui diversas funções específicas e nesta seção iremos conhecer algumas, e outras no decorrer deste curso.

3.6.1 Matemáticas

- Somatório:
  É uma notação simplificada de várias somas. Mesmo sendo uma notação simplis ela tem espaço no R.

Exemplo$_1$: Criar um vetor com os cinquenta primeiros números ímpares, armazenar o vetor em uma matriz e calcular usando o R a soma dos elementos.

```
> x = seq(1,99,2)  # Criando o vetor.

> a=matrix(x,ncol=10); a # Armazenando os valores.

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    1   11   21   31   41   51   61   71   81    91
[2,]    3   13   23   33   43   53   63   73   83    93
[3,]    5   15   25   35   45   55   65   75   85    95
[4,]    7   17   27   37   47   57   67   77   87    97
[5,]    9   19   29   39   49   59   69   79   89    99

> sum(a) # Somatório.
[1] 2500
```

- Produtório:
  Conceito similar ao somatório. É uma notação simplificada de várias multiplicações.

Exemplo$_2$: Calcular o produtório dos elementos das duas primeiras colunas da matriz do exemplo enterior.

```
> b=a[,1:2];b # Submatriz de a.

     [,1] [,2]
[1,]    1   11
[2,]    3   13
[3,]    5   15
[4,]    7   17
[5,]    9   19
```

```
> prod(b) # Produtório.
[1] 654729075
```

3.6.2 Logaritmica e Exponencial

A função logaritmica é dada por:

$$y = a\ln(bx).$$

Aqui a base do logaritmo é o e. A função exponencial em qual a resposta Y é o antilogaritmo da variável X, é determinada por:

$$y = ae^{bx}.$$

O programa R tem essas duas funções prontas para uso imediato, para calcularmos o logaritmo natural(ln) basta usarmos a função `log()`, e para a função exponencial `exp()`. Mais adiante aprendderemos a fazer gráficos de algumas funções vistas.

Exemplo$_3$: Selecione dois elementos da matriz "a" e calcule os valor das funções logaritma e exponencial para cada um deles.

```
> e1 = a[3,2];e1 # Primero elemento.
[1] 15

> log(e1);exp(e1)
[1] 2.70805
[1] 3269017

> e2 = a[4,1];e2 # Segundo elemento.
[1] 7
```

```
> log(e2);exp(e2)
[1] 1.94591
[1] 1096.633
```

3.6.3 Trigonométricas

Assim como as funções logaritmicas e exponencial o R também possui as funções trigonometricas prontas para serem usadas.Sabendo o os resultado é dado em radianos.
Assim para as seguintes funções $seno, cosseno, tangente$ temos respectivamente:

```
> sin()
> cos()
> tan()
```

Para $x = 0$, temos:

```
> sin(0)
[1] 0
> cos(0)
[1] 1
> tan(0)
[1] 0
```

3.6.4 Derivadas

A sintaxe da função do R que usamos para efetuar calculos com derivadas é dado da seguinte forma:

`> D(expression(),"x")` onde:

`D`, "Carrega" a função de derivação;
`expression()`, Expressão(função) a ser derivada;
`"x"`, Variável que "rege" a derivação.

Exemplo$_4$: Calcular as derivadas das função $x^2 + 5 * x + 6$, $\log(x)$, $cos(x)$ e $(3 * x)^4$ em relação a variável x.

```
> D(expression(x^2 + 5*x + 6),"x")
2 * x + 5

> D(expression(log(x)),"x")
1/x

> D(expression(cos(x)),"x")
-sin(x)

> D(expression((3*x)^4),"x")
4 * (3 * (3 * x)^3)
```

Função	Descrição
`abs(x)`	Valor absoluto de x
`round(x,digits = n)`	Arredonda x com n decimais
`ceiling(x)`	Arredondamento de x para o maior valor
`floor(x)`	Arredondamento de x para o menor valor
`length(x)`	Número de elementos do vetor x.
`max(x)`	Seleciona o maior elemento do vetor x.
`min(x)`	Seleciona o menor elemento do vetor x.
`range(x)`	Retorna o menor e o maior elemento do vetor x.

Tabela 3.1 *Mais algumas funções matemáticas básicas*

3.7 Gráficos de Funções

Agora vamos ver como criar gráficos de funções no R. E para isso utilizaremos alguns exemplos vistos anteriormente.

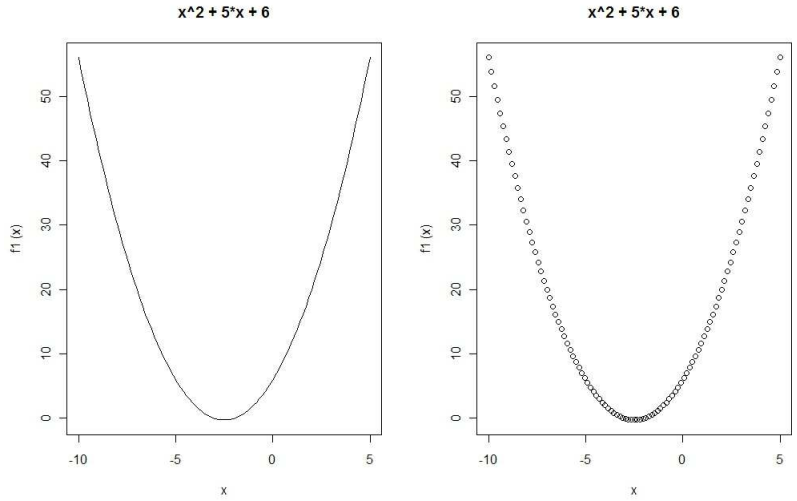
O comando abaixo é uma das maneiras de criarmos gráficos no R:

```
> plot(x,y,type="",col="",main="",xlab="",ylab="",
+ xlim=c(lim_inf,lim_sup),ylim=c(lim_inf,lim_sup))
```

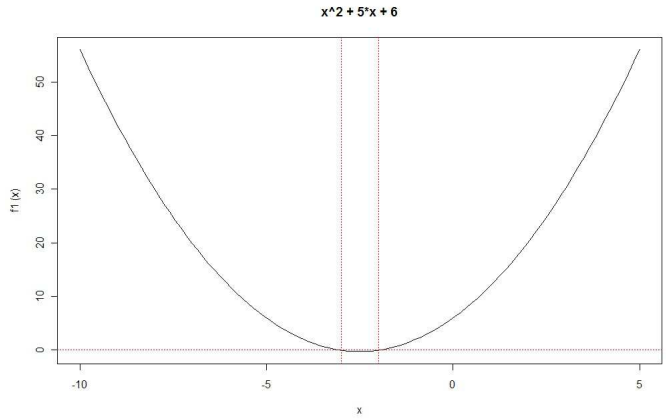
Onde:
`x,y`, são os argumentos;
`type=‘‘’’`, tipo de linha;
`col=‘‘’’`, cor da linha;
`main=‘‘’’`, título do gráfico;
`xlab=‘‘’’,ylab=‘‘’’`, títulos dos respectivos eixos;
`xlim=c(lim_inf,lim_sup),ylim=c(lim_inf,lim_sup)`, limites dos respectivos eixos.

Exemplo$_1$: Criar o gráfico da função $f1 = x^2 + 5x + 6$.

```
> par(mfrow=c(1,2))
> plot(f1,type="l",main="x^2 + 5*x + 6",xlim=c(-10,5))
> plot(f1,type="b",main="x^2 + 5*x + 6",xlim=c(-10,5))
```




```
> plot(f1,type="l",main="x^2 + 5*x + 6",xlim=c(-10,5))
> abline(h=0,col="red",lty=3)
> abline(v=-2,col="red",lty=3);abline(v=-3,col="red",lty=3)
```



Função abline:
`h=0`, valor horizontal;
`v=-2`, valor vertical;
`lty`, tipo de linha.

As funções logaritmica e exponencial são funções contínuas, assim para fazermos seus gráficos temos que criar um vetor com poucos espaços entre os valores.

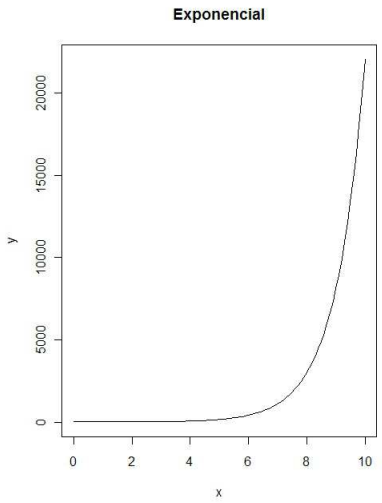
As funções trigonometricas têm sua variação entre 0 e 2π, assim para fazermos seus gráficos temos que fazer um vetor entre esses valores.
Abaixo temos os comandos e gráficos das funções citadas.

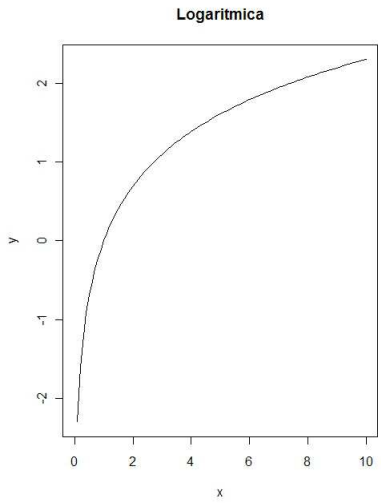
Funçõe Exponencial e Logaritmica.

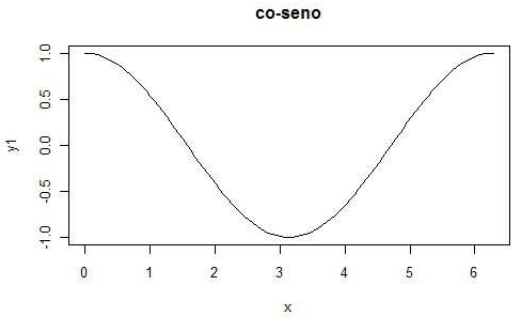
```
> par(mfrow=c(1,2))
> x=seq(0,10,0.1)
> y=exp(x)
> plot(y~x,type="l",main="Exponencial")
> y=log(x)
> plot(y~x,type="l",main="Logaritmica")
```

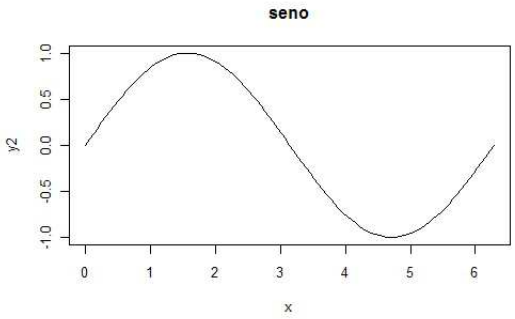
Funções Trigonometricas.

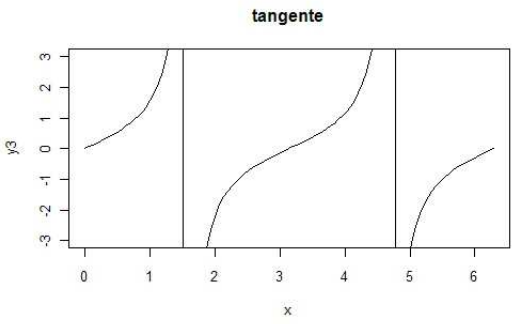
```
> par(mfrow=c(2,2))
> x= seq(0,2*pi,2*pi/100)
> y1=cos(x)
> y2=sin(x)
> y3=tan(x)
> plot(y1~x,type="l",main="co-seno")
> plot(y2~x,type="l",main="seno")
> plot(y3~x,type="l",main="tangente",ylim=c(-3,3))
```


Exponencial
y
x
0
5000
10000
15000
20000
0
2
4
6
8
10


Logaritmica
y
x
-2
-1
0
1
2
0
2
4
6
8
10


co-seno
y1
x
-1.0
-0.5
0.0
0.5
1.0
0
1
2
3
4
5
6


seno
y2
x
-1.0
-0.5
0.0
0.5
1.0
0
1
2
3
4
5
6


tangente
y3
x
-3
-2
-1
0
1
2
3
0
1
2
3
4
5
6

Capítulo 4

Estatística Básica

Neste capítulo trataresmos de algumas medidas de posição e medidas de dispersão, trasendo suas definições, formulas, como utuliza-las no programa R e alguns exemplos básicos para de cada uma das medidas.

Também faremos neste uma abordagem a importação de dados do Excel, para posteriormente trabalharmos nestes utilizando as medias estatísticas vistas neste capítulo.

4.1 Medidas de Posição

As medidas de posição podem se apresentar de várias formas, dependendo daquilo que se pretende conhecer a respeito dos dados estatísticos. As mais importantes são as medidas de tendência central, as quais são assim donominadas, em virtude da tendência de os dados observados se agruparem em torno desses valores centrais. A média, mediana e a moda são estas medidas de tendência central. Como medias de posição temos ainda além das citadas acima os quantis.

4.1.1 Média

A medida de tendência central mais usada para descrever um conjunto de dados é a média. A média de um conjunto de números é igual ao quociente entre a soma dos valores do conjunto e o número total de valores.

Símbolo: $\bar{x}$(lê-se "x traço" ou "x barra").

Quando queremos calcular a média de um comjunto de dados no R utilizamos

a função `mean(x)`, onde `x` representa o conjunto de dados.

Exemplo$_1$: Crie o seguinte dataframe,calcule a média das idades do indivíduos e faça o gráfico das alturas e marque a média.

```
  Indivíduos Altura
1    Antonio   1.85
2  Francisco   1.73
3        Ana   1.68
```

Calculando a média das alturas e fazendo o gráfico

```
> attach(df1)
> mean(Altura)
[1] 1.753333

> plot(Altura,ylab="Altura",xlab="Indivíduos")
> points(1.7533,col=3) # Adiciona um ponto no gráfico.
```

4.1.2 Mediana

A mediana pode ser definida como o valor que divide uma série ordeneda de tal forma que pelo menos a metade ou cinqüenta por cento dos itens sejam iguais ou menores que ela, e que haja pelo menos outra metade ou cinqüenta por cento de itens maiores que ela.

Símbolo: Md

No programa R assim como a média a mediana também tem uma função específica para seu cálculo, esta função é `median(x)`, onde `x` representa o conjunto de dados.

Exemplo$_2$: Usando a função `scan()` crie os seguintes conjuntos de dados, um com número par de elementos e o outro com número ímpar de dados. Depois calcule suas respectivas medianas.

```
> sc1=scan()
1: 1500
2: 950
3: 2000
4: 1800
5: 1750
6: 1900
7:
Read 6 items

> median(sc1)
[1] 1775

> sc2=scan()
1: 1
2: 3
3: 0
4: 2
5: 1
6:
Read 5 items

> median(sc2)
[1] 1
```

4.1.3 Moda

A moda é outra medida de tendência central, havendo outras denominações para designá-la: norma, valor dominante, valor típico. Genericamente, pode-se definir a moda como o valor mais freqüente, quando comparada sua freqüência com a dos valores contíguos de um conjunto ordenado.

Para calcularmos o valor mais freqüente de um conjunto de dados utilizando o preograma R, fazemos uso da função `Mode(x)`, onde "x" é um comjunto de

dados.

Símbolo: M_o

Exemplo$_3$: Utilizando a função `edit()`, acrescente no data-frame "df1" a variável salário e depois calcule sua moda.

```
> ed1
  Indivíduos Altura Salário
1    Antonio   1.85    2000
2  Francisco   1.73    2500
3        Ana   1.68    2000
4      Pedro   1.80    1800
5      Maria   1.65    2450

> moda<-function(d)
+ {
+ if ((is.vector(d) || is.matrix(d) || is.factor(d)==TRUE) &&
+ (is.list(d)==FALSE))
+ {
+ dd<-table(d)
+ valores<-which(dd==max(dd))
+ vmodal<-0
+ for(i in 1:(length(valores)))
+ if (i==1) vmodal<-as.numeric(names(valores[i]))
+ else
+ vmodal<-c(vmodal,as.numeric(names(valores[i])))
+ if (length(vmodal)==length(dd))
+ print("conjunto sem valor modal")
+ else return(vmodal)
+ }
+ else print("o parâmetro deve ser um vetor ou uma matriz")
+ }
```

```
> moda(Salário)
[1] 2000
```

4.1.4 Quartis-Decis-Percentis(ou Centis)

Há uma série de medidas de posição semelhantes na sua concepção à mediana, embora não sejam medidas de tendência central. Os quartis divide a distribuição em quatro partes iguais quanto ao número de elementos de cada uma; os decis em dez partes e os centis em cem partes iguais.
Para simbolizar cada uma dessas medidas separatrizes, faremos:

$Q_i = quartis \quad i = 1, 2, 3$
$D_i = decis \quad i = 1, 2, 3, ..., 9$
$C_i = centis \quad i = 1, 2, 3, ..., 99$

No programa R para trabalharmos com essas medidas de posição utilizamos a seguinte funçõe `quantile(x,probs=seq(0,1,?))`, onde "x" representa o conjunto de dados e "?" representa o porcentagem dos quantis.

$Exemplo_4$: Crie uma sequência de números e calcule seus quantis.

```
> f=seq(1,10,0.1)
> f
 [1]  1.0  1.1  1.2  1.3  1.4  1.5  1.6  1.7
 [9]  1.8  1.9  2.0  2.1  2.2  2.3  2.4  2.5
[17]  2.6  2.7  2.8  2.9  3.0  3.1  3.2  3.3
[25]  3.4  3.5  3.6  3.7  3.8  3.9  4.0  4.1
[33]  4.2  4.3  4.4  4.5  4.6  4.7  4.8  4.9
[41]  5.0  5.1  5.2  5.3  5.4  5.5  5.6  5.7
[49]  5.8  5.9  6.0  6.1  6.2  6.3  6.4  6.5
[57]  6.6  6.7  6.8  6.9  7.0  7.1  7.2  7.3
[65]  7.4  7.5  7.6  7.7  7.8  7.9  8.0  8.1
[73]  8.2  8.3  8.4  8.5  8.6  8.7  8.8  8.9
```

```
[81]  9.0  9.1  9.2  9.3  9.4  9.5  9.6  9.7
[89]  9.8  9.9 10.0

> quantile(f)
   0%   25%   50%   75%  100%
 1.00  3.25  5.50  7.75 10.00

> quantile(f,probs=seq(0,1,0.1))
  0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
 1.0  1.9  2.8  3.7  4.6  5.5  6.4  7.3  8.2  9.1 10.0

> quantile(f,probs=seq(0,1,0.01))
   0%    1%    2%    3%    4%    5%    6%    7%
 1.00  1.09  1.18  1.27  1.36  1.45  1.54  1.63
   8%    9%   10%   11%   12%   13%   14%   15%
 1.72  1.81  1.90  1.99  2.08  2.17  2.26  2.35
  16%   17%   18%   19%   20%   21%   22%   23%
 2.44  2.53  2.62  2.71  2.80  2.89  2.98  3.07
  24%   25%   26%   27%   28%   29%   30%   31%
 3.16  3.25  3.34  3.43  3.52  3.61  3.70  3.79
  32%   33%   34%   35%   36%   37%   38%   39%
 3.88  3.97  4.06  4.15  4.24  4.33  4.42  4.51
  40%   41%   42%   43%   44%   45%   46%   47%
 4.60  4.69  4.78  4.87  4.96  5.05  5.14  5.23
  48%   49%   50%   51%   52%   53%   54%   55%
 5.32  5.41  5.50  5.59  5.68  5.77  5.86  5.95
  56%   57%   58%   59%   60%   61%   62%   63%
 6.04  6.13  6.22  6.31  6.40  6.49  6.58  6.67
  64%   65%   66%   67%   68%   69%   70%   71%
 6.76  6.85  6.94  7.03  7.12  7.21  7.30  7.39
  72%   73%   74%   75%   76%   77%   78%   79%
 7.48  7.57  7.66  7.75  7.84  7.93  8.02  8.11
  80%   81%   82%   83%   84%   85%   86%   87%
```

```
 8.20  8.29  8.38  8.47  8.56  8.65  8.74  8.83
  88%   89%   90%   91%   92%   93%   94%   95%
 8.92  9.01  9.10  9.19  9.28  9.37  9.46  9.55
  96%   97%   98%   99%  100%
 9.64  9.73  9.82  9.91 10.00
```

4.1.5 Trabalhando dados do Excel

Para trabalharmos melhor as funções das medidas de posição vistas até aqui, importaremos uma tabela de dados do Excel na extensão "csv". Esses dados estão foram retirados da quinta edição do livro "Estatística Básica" escrito por Wilton de O.Bussab e Pedro A.Morettin.

Para importarmos estes dados utilizaremos a função abaixo, neste exemplo o diretorio do R foi mudado para não termos que digitar todo o endereço do arquivo.

```
read.table("nome_do_arquivo.csv",sep=";",header=T).
```

```
> dados1=read.table("dados1.csv",sep=";",header=T);dados1
   estadocivil    instrucao nfilhos salario idade procedencia
1     Solteiro  fundamental      NA    4.00    26    Interior
2       Casado  fundamental       1    4.56    32     Capital
3       Casado  fundamental       2    5.25    36     Capital
4     Solteiro ensino médio      NA    5.73    21       Outro
5     Solteiro  fundamental      NA    6.26    41       Outro
6       Casado  fundamental       0    6.66    28    Interior
7     Solteiro  fundamental      NA    6.86    41    Interior
8     Solteiro  fundamental      NA    7.39    43     Capital
9       Casado ensino médio       1    7.59    34     Capital
10    Solteiro ensino médio      NA    7.44    24       Outro
11      Casado ensino médio       2    8.12    34    Interior
12    Solteiro  fundamental      NA    8.46    28     Capital
13    Solteiro ensino médio      NA    8.74    37       Outro
14      Casado  fundamental       3    8.95    44       Outro
15      Casado ensino médio       0    9.13    30    Interior
16    Solteiro ensino médio      NA    9.35    39       Outro
17      Casado ensino médio       1    9.77    32     Capital
18      Casado  fundamental       2    9.80    40       Outro
19    Solteiro     superior      NA   10.53    26    Interior
20    Solteiro ensino médio      NA   10.76    37    Interior
21      Casado ensino médio       1   11.06    31       Outro
22    Solteiro ensino médio      NA   11.59    34     Capital
23    Solteiro  fundamental      NA   12.00    41       Outro
24      Casado     superior       0   12.79    26       Outro
25      Casado ensino médio       2   13.23    32    Interior
26      Casado  fundamental       2   13.60    35       Outro
27    Solteiro ensino médio      NA   13.85    47       Outro
28      Casado ensino médio       0   14.69    30    Interior
29      Casado ensino médio       5   14.71    41    Interior
30      Casado ensino médio       2   15.99    36     Capital
31    Solteiro     superior      NA   16.22    31       Outro
```

```
32      Casado ensino médio       1   16.61    36    Interior
33      Casado     superior       3   17.26    44     Capital
34    Solteiro     superior      NA   18.75    34     Capital
35      Casado ensino médio       2   19.40    49     Capital
36      Casado     superior       3   23.30    42    Interior
```

Nos exemplos seguintes faremos algumas manipulações com esses dados.

$Exemplo_5$: Obter o salário médio.

```
> attach(dados1)
> mean(salario)
[1] 11.12222
```

$Exemplo_6$: Obter o salário médio por região de procedência.

```
> x6=tapply(salario,procedencia,mean)
> x6
 Capital Interior    Outro
11.45545 11.55000 10.44538
```

$Exemplo_7$: Obter o salário médio por região de procedência e grau de instrução.

```
> x7=tapply(salario,list(procedencia,instrucao),mean)
> x7
         ensino fundamental ensino médio superior
Capital               6.415    12.868000   18.005
Interior              5.840    12.464286   16.915
Outro                10.122     9.361667   14.505
```

$Exemplo_8$: Obter número de funcionários por região de procedência e grau de instrução.

```
> x8=table(procedencia,instrucao)
> x8
```

```
           instrucao
procedencia ensino fundamental ensino médio superior
   Capital                   4            5        2
   Interior                  3            7        2
   Outro                     5            6        2
```

$Exemplo_9$: Obter a distribuição de frequência para o número de filhos.

```
> x9=table(nfilhos)
> x9
nfilhos
0 1 2 3 5
4 5 7 3 1
```

$Exemplo_{10}$: Criar um vetor contendo apenas as idades dos funcionários do interior.

```
> idade.i=dados1[dados1[,7]=="Interior",6]
> idade.i
 [1] 26 28 41 34 30 26 37 32 30 41 36 42
```

$Exemplo_{11}$: Obter os valores extremos, a média e os quartis da variável idade.

```
> summary(idade)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  21.00   30.75   34.50   35.06   41.00   49.00
```

4.2 Medidas de Dispersão

Abordaremos agora algumas medias de dispersão, essas medidas se referem a variabilidade dos valores dos dados observados. Vejamos um exemplo simples para melhor entendermos essas medidas.

Suponhamos que cinco grupos de alunos submeteram-se a um teste, obtendo-se as seguintes notas:

grupo A (variável X): 3,4,5,6,7
grupo B (variável Y): 1,3,5,7,9
grupo C (variável Z): 5,5,5,5,5
grupo D (variável W): 3,5,5,7
grupo E (variável V): 3,5,5,6,5

Vemos que $\bar{x} = \bar{y} = \bar{z} = \bar{w} = \bar{v} = 5,0$. A identificação de cada uma destas séries por sua média nada informa sobre suas diferentes variabilidades. Notamos, então, a conveniência de serem cridos medidas que sumarizem a varibilidade de um conjunto de observações.

4.2.1 Amplitude Total

A amplitude total de um conjunto de números é a diferença entre os valores extremos do conjunto.

Símbolo: A_t

O programa R não tem diretamente definida uma função para se calcular a amplitude total de um conjunto de dados, porém é muito simples obter esta medida de dispersão através de outras duas funções, veja o exemplo.

Exemplo$_1$: Um psicólogo deseja obter informações sobre o grau de dispersão de dados referentes à idade dos freqüentadores de um grupo de Alcoólicos Anônimos. Ele coletou os seguintes dados: 33 17 39 78 29 32 54 22 38 18. Pede-se calcular, a média, a mediana e a amplitude total.

```
> x=c(33,17,39,78,29,32,54,22,38,18);x
 [1] 33 17 39 78 29 32 54 22 38 18

> mean(x);median(x)
[1] 36
[1] 32.5

> at=max(x)-min(x);at
```

```
[1] 61
```

4.2.2 Desvio Padrão

O desvio-padrão é a medida de dispersão mais usada. O desvio-padrão não é senão uma média quadrática dos desvios em relação à média aritmética de um conjunto de números, ou seja, é a raiz quadrada da média aritmética dos quadrados dos desvios, estes tomados a partir da média aritmática.

Símbolo: S

Para calcularmos o desvio-padrão de um conjunto de dados utilizando o programa R usamos a função `sd(x)`, onde "x" representa o conjunto de dados.

$Exemplo_2$: Calcular o desvio-padrão, colocar os dados em ordem crescente e calcular a amplitude total dos dados do $exemplo_{10}$ da seção anterior.

```
> idade.i
 [1] 26 28 41 34 30 26 37 32 30 41 36 42

> sd(idade.i)
[1] 5.822501

> sort(idade.i)
 [1] 26 26 28 30 30 32 34 36 37 41 41 42

> at=max(idade.i)-min(idade.i);at
[1] 16
```

4.2.3 Variância

A variância é o quadrado do desvio-padrão, ou, se preferir, o desvio-padrão é a raiz quadrada do desvio-padrão.

Símbolo: S^2

No programa R a variância pode ser calculada pela função `var(x)`, onde "x" representa o conjunto de dados.

Exemplo$_3$: Criar um vetor contendo apenas os salário dos funcionários da capital. Obter o summary, os decis, o desvio-padrão e a variância.

```
> salario.c=dados1[dados1[,7]=="Capital",5]
> salario.c
 [1]  4.56  5.25  7.39  7.59  8.46  9.77 11.59 15.99 17.26 18.75 19.40

> summary(salario.c)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   4.56    7.49    9.77   11.46   16.62   19.40

> quantile(salario.c,probs=seq(0,1,0.1))
   0%   10%   20%   30%   40%   50%   60%   70%   80%   90%  100%
 4.56  5.25  7.39  7.59  8.46  9.77 11.59 15.99 17.26 18.75 19.40

> sd(salario.c)
[1] 5.476653

> var(salario)
[1] 21.04477
```

4.2.4 Coeficiante de Variação de Pearson

O coeficiente de variação de Pearson é igual ao quociente entre o desvio-padrão e a média aritmética.

Símbolo: CV_p

O programa R não tem uma função definida para calcular o coefiente de

variação de Pearson. Porém ele pode ser obtido a partir da definição dada.

Exemplo$_4$: Criar um vetor contendo apenas os salário dos funcionários da capital e do interior. Calcular o coefiente de variação de Pearson.

```
> salario.ci=c(salario.c,salario.i);salario.ci
 [1]  4.56  5.25  7.39  7.59  8.46  9.77 11.59 15.99 17.26 18.75 19.40
[12]  4.00  6.66  6.86  8.12  9.13 10.53 10.76 13.23 14.69 14.71 16.61
[23] 23.30

> cvp=sd(salario.ci)/mean(salario.ci)
> cvp
[1] 0.4571379
```

4.2.5 Trabalhando dados do Excel

Esta seção e destinada a prática da manipulação de dados e a prática das funções das medidas de dispersão vistas nesta apostila.

Exemplo: Criar 3 arquivos de dados (com todas as variáveis) segundo a região de procedência e calcular todas as medidas de disparsão para a variável salário em cada um dos arquivos.

```
> dados1.capital=dados1[dados1[,6]=="Capital",];dados1.capital
   estadocivil    instrucao nfilhos salario idade procedencia
2       Casado  fundamental       1    4.56    32     Capital
3       Casado  fundamental       2    5.25    36     Capital
8     Solteiro  fundamental      NA    7.39    43     Capital
9       Casado ensino médio       1    7.59    34     Capital
12    Solteiro  fundamental      NA    8.46    28     Capital
17      Casado ensino médio       1    9.77    32     Capital
22    Solteiro ensino médio      NA   11.59    34     Capital
30      Casado ensino médio       2   15.99    36     Capital
33      Casado     superior       3   17.26    44     Capital
```

```
34    Solteiro     superior      NA   18.75    34     Capital
35      Casado ensino médio       2   19.40    49     Capital

> at1;sd1;var1;cvp1
[1] 14.84
[1] 5.476653
[1] 29.99373
[1] 0.4780826

> dados1.Interior=dados1[dados1[,6]=="Interior",];dados1.Interior
   estadocivil    instrucao nfilhos salario idade procedencia
1     Solteiro  fundamental      NA    4.00    26    Interior
6       Casado  fundamental       0    6.66    28    Interior
7     Solteiro  fundamental      NA    6.86    41    Interior
11      Casado ensino médio       2    8.12    34    Interior
15      Casado ensino médio       0    9.13    30    Interior
19    Solteiro     superior      NA   10.53    26    Interior
20    Solteiro ensino médio      NA   10.76    37    Interior
25      Casado ensino médio       2   13.23    32    Interior
28      Casado ensino médio       0   14.69    30    Interior
29      Casado ensino médio       5   14.71    41    Interior
32      Casado ensino médio       1   16.61    36    Interior
36      Casado     superior       3   23.30    42    Interior

> at2;sd2;var2;cvp2
[1] 19.3
[1] 5.296055
[1] 28.0482
[1] 0.4585329

> dados1.Outro=dados1[dados1[,6]=="Outro",];dados1.Outro
   estadocivil    instrucao nfilhos salario idade procedencia
4     Solteiro ensino médio      NA    5.73    21       Outro
```

```
5     Solteiro  fundamental      NA    6.26    41       Outro
10    Solteiro ensino médio      NA    7.44    24       Outro
13    Solteiro ensino médio      NA    8.74    37       Outro
14      Casado  fundamental       3    8.95    44       Outro
16    Solteiro ensino médio      NA    9.35    39       Outro
18      Casado  fundamental       2    9.80    40       Outro
21      Casado ensino médio       1   11.06    31       Outro
23    Solteiro  fundamental      NA   12.00    41       Outro
24      Casado     superior       0   12.79    26       Outro
26      Casado  fundamental       2   13.60    35       Outro
27    Solteiro ensino médio      NA   13.85    47       Outro
31    Solteiro     superior      NA   16.22    31       Outro
>
```

```
> at3;sd3;var3;cvp3
[1] 10.49
[1] 3.145453
[1] 9.893877
[1] 0.3011333
```

4.3 Gráficos no R

As capacidades gráficas são uma componente muito importante e extremamente versátil do ambiente R. O R consegue plotar desde gráficos bidimensionais simples até graficos tridimensionais mais complexos por meio de comandos simples. Dá-se muita ênfase no R aos gráficos estatísticos, tais como histogramas, curvas de distribuições, gráfico de barras dentre outros.

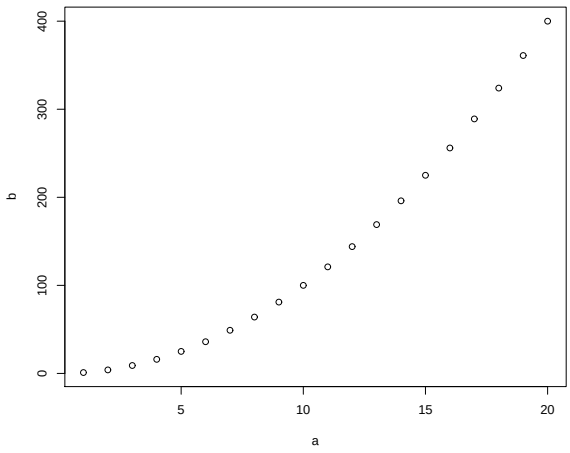
4.3.1 Comandos Básicos

O Comando básico para a criação gráfica é o `plot()`. A função `plot(dados)` gera um gráfico simples, atribuindo pontos em coordenadas cartesianas.

```
> a<- 1:20
```

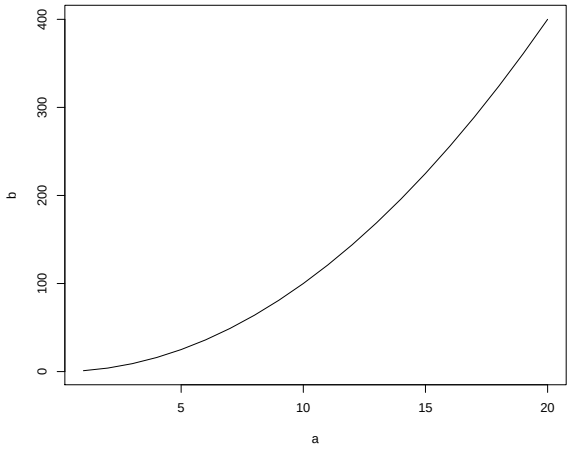
```
> b<- a^2
> plot(a,b)
```

Teremos como resposta do comando acima o seguinte gráfico:



Para tornar o gráfico acima contínuo, deve-se acrescentar o argumento `type="l"` na função `plot()`.

```
> plot(a,b,type="l")
```

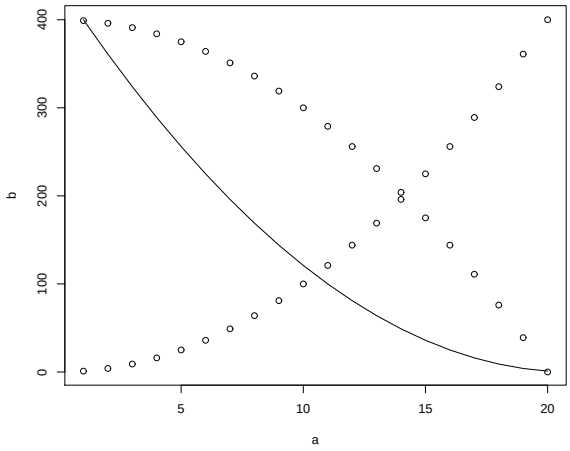


Além deste argumento, existem inúmeros outros argumentos para a configuração do gráfico que podem ser acessados com o comando `help(plot)`.

Através dos comandos `lines()` e `points()` é possível adicionar, após dado

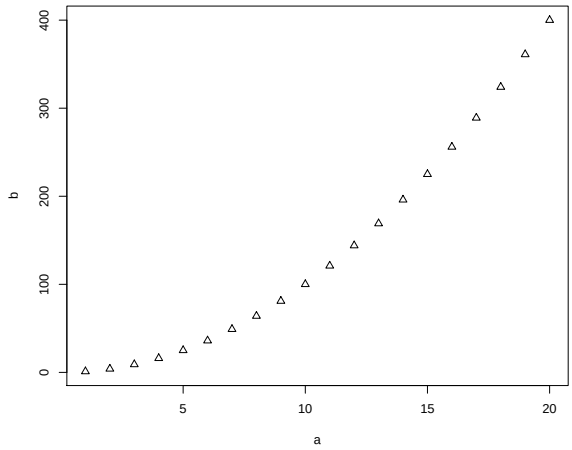
um comando de `plot()`, linhas e pontos, respectivamente, a um gráfico já existente. Veja o exemplo abaixo:

```
> b<- a^2
> plot(a,b)
> lines(rev(a),b) # adição de linhas
> points(a,400-b) # adição de pontos
```

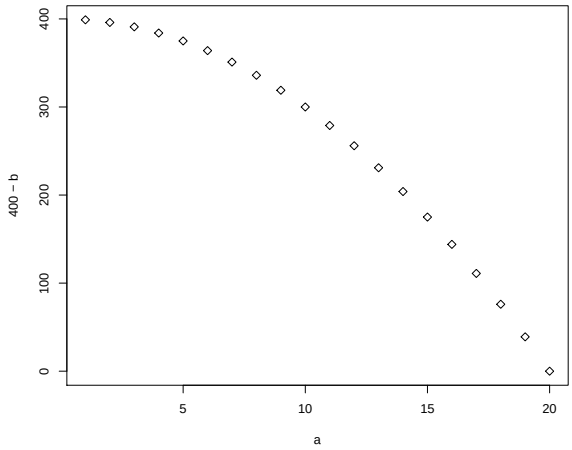


O R permite que sejam feitas mudanças na representação dos indicadores gráficos (pontos) através do parâmetro `pch=` nos comandos `plot()` e `points()`. Veja os exemplos abaixo:

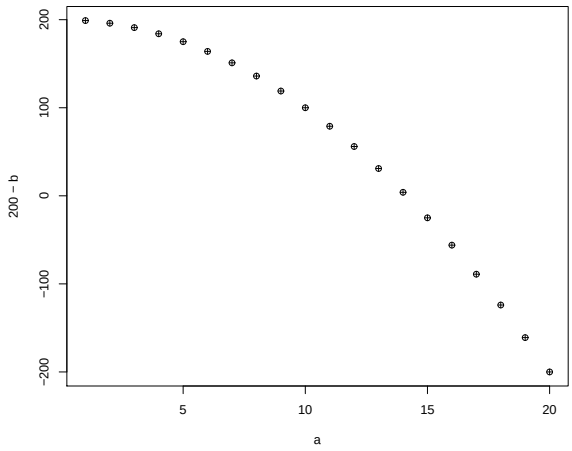
```
> a <- 1:20; b <- a^2
> plot(a,b,pch=2)
```



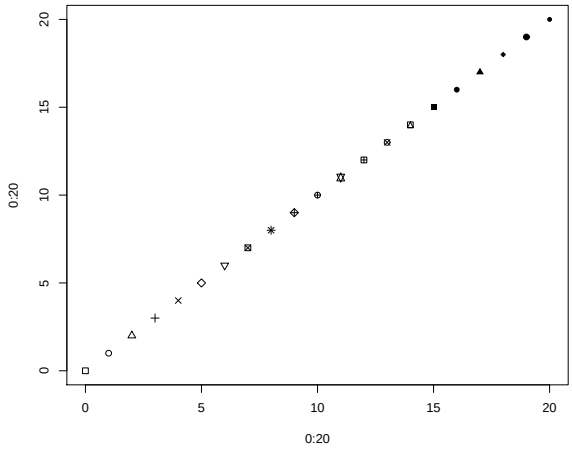

```
> plot(a,400-b, pch=5)
```




```
> plot(a,200-b, pch=10)
```




```
> plot(0:20,0:20,pch=0:20)
```

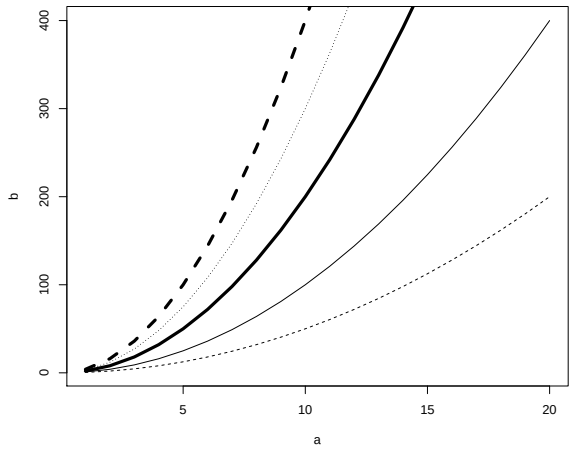


Ainda, é possível realizar mudanças nas caracteristicas das linhas. Para isso, basta utilizar os comandos `lwd=` e `lty=` que modificam, respectivamente, a largura e o estilo da linha. Veja o exemplo seguinte:

Exemplo$_1$:

```
>  a <- 1:20; b <- a^2
>  plot(a,b,type="l")
>  lines(a,2*b,lwd=4)
>  lines(a,0.5*b,lty=2)
>  lines(a,3*b,lty=3)
```

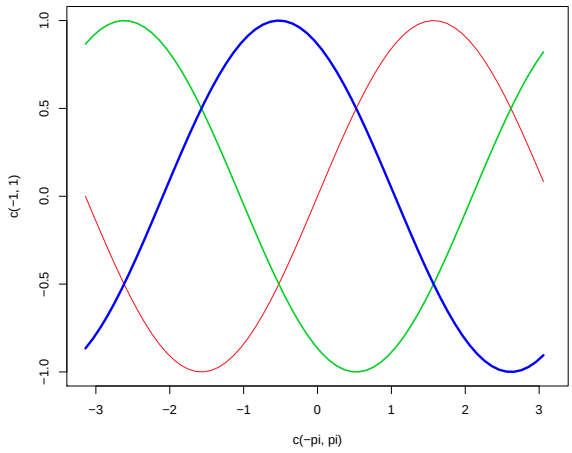
```
>  lines(a,4*b,lty=2,lwd=4)
```



Para alterar a dimensão dos intervalos, pode-se primeiro plotar um gráfico em branco, ajustando os limites da abscissa e da ordenada e depois gerar o gráfico desejado. Observe no exemplo como proceder:

$Exemplo_2$:

```
>  plot(c(-pi,pi),c(-1,1), type="n") #gerando um gráfico em branco
>  x<-seq(-pi,pi,0.1)
>  a <- sin(x)
>  b <- sin(x-2/3*pi)
>  c <- sin(x+2/3*pi)
>  lines(x,a,col=2,lwd=1)
>  lines(x,b,col=3,lwd=2)
>  lines(x,c,col=4,lwd=3)
```

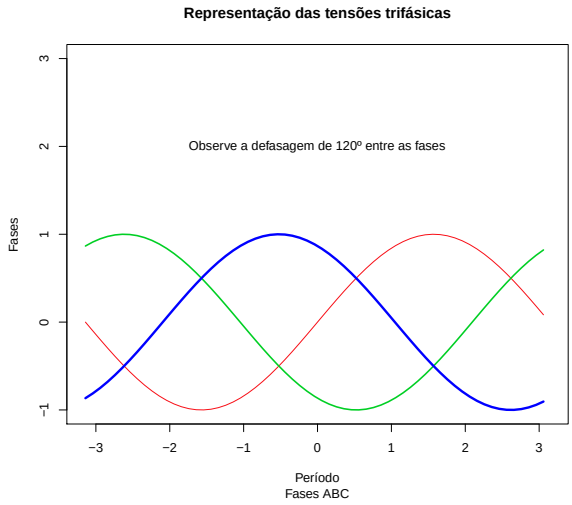


Pode-se ainda acrescentar o nome dos eixos através dos parâmetros `xlab=` e `ylab=` no comando `plot()`. O título do texto pode ser adicionado com o parâmetro `main=` no comando `plot()` ou através do comando `title("título","subtítulo"`

A legenda do gráfico pode ser acrescida através do comando text() que possui como argumentos as coordenadas do ponto em que se quer colocar a legenda e o texto desejado. Observe o exemplo:

Exenplo$_3$:

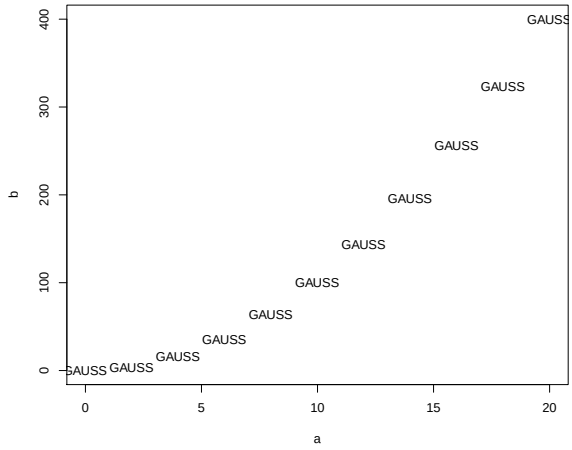
```
>  plot(c(-pi,pi),c(-1,3),xlab="Período", ylab="Fases", type="n")
>  title("Representação das tensões trifásicas","Fases ABC")
>  lines(x,a,col=2,lwd=1)
>  lines(x,b,col=3,lwd=2)
>  lines(x,c,col=4,lwd=3)
>  text(0,2,"Observe a defasagem de 120° entre as fases")
```



Outra utilidade do comando `text()` é acrescentar textos às coordenadas cartesianas `x` e `y`. Observe a sintaxe seguida do exemplo:

Exemplo$_4$

```
>  a <- seq(from=0, to=20, by=2); b <- a^2
>  plot(a,b,type="n") #plota um gráfico vazio
>  text(a,b,"GAUSS") #aplica a etiqueta no sítio dos pontos
```



4.3.2 Criando Novas Janelas Gráficas e Salvando Gráficos

Ao executarmos sucessivos comandos `plot()` os gráficos gerados são sobrepostos na mesma janela gráfica chamada de device ACTIVE. Para evitar este problema, podemos proceder de duas formas, conforme a conveniência:

- Os gráficos podem ser salvos imediatamente ao serem gerados. Existem vários formatos em que o R pode salvar imagens gráficas. Alguns deles são: JPEG, BMP, PDF, TIFF, PNG. Faremos abaixo um exemplo utilizando o formato JPEG, mas tenha em mente que a sintaxe para qualquer formato segue idêntica.

$Exemplo_1$:

```
>jpeg(file="figure.jpeg") #figure é o nome do arquivo imagem
>plot(rnorm(10)) #gráfico que estou salvando
>dev.off() #fecha a janela gráfica automaticamente
```

- Por vezes é necessário gerar várias janelas gráficas (devices). Para isso basta utilizar o comando `windows()` ou `X11` entre os sucessivos `plot()`. Confira o exemplo abaixo:

$Exemplo_2$:

```
> plot(rnorm(10)) #plotando o primeiro gráfico
> windows() #criação de uma nova janela gráfica
> plot(rnorm(20)) #plotando o segundo gráfico
```

Observe que o segundo gráfico vai surgir numa outra janela, o que permite ao utilizador ver os gráficos simultaneamente. Observe, também, que posteriores gráficos que venham a ser plotados serão sobrepostos ao gráfico do último device aberto (leia na parte superior da janela Device(ACTIVE)).

4.3.3 Outras Funcionalidades

Uma funcionalidade bastante útil do R consiste na utilização de identificadores gráficos quando se deseja identificar um ponto ou um conjunto de pontos em um gráfico. Para tanto, existem dois identificadores que podem ser utilizados:

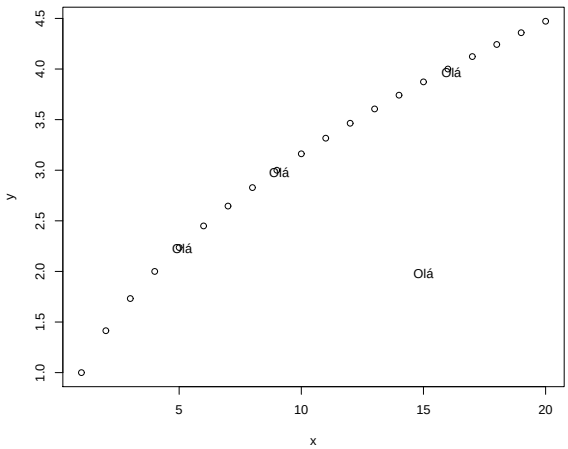
- `locator()`: permite que o utilizador selecione regiões do gráfico utilizando o botão esquerdo do mouse até que se tenha um número n de pontos selecionados ou até pressionar o botão direito do mouse. Cada clique que é dado com o botão esquerdo do mouse o R retorna na console as coordenadas do clique. Veja o exemplo:

$Exemplo_1$:

```
>  x=1:20
>  y=sqrt(x)
>  plot(x,y)
>text(locator(4),"Olá")#onde for dado o clique será escrita a mensagem

> #ou de outra forma:
>  plot(x,y)
>  locator(2)
$x
[1] 10.041331  5.974781

$y
[1] 3.169641 2.450349
```



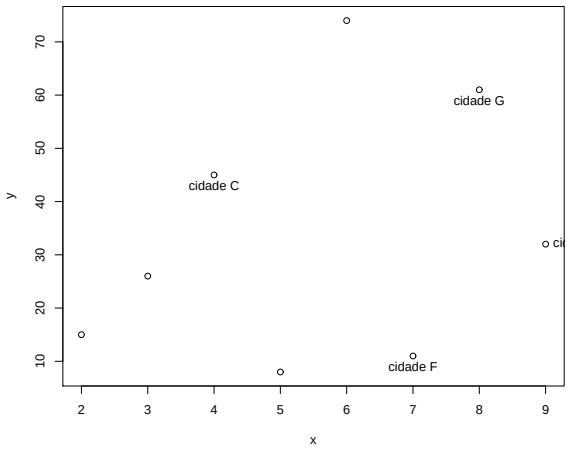
- `identify()`: comando semelhante ao `locator()`, porém apresenta a ca-

pacidade de identificar pontos particulares de um gráfico e não apenas sua posição.

Exemplo$_2$: Representar as coordenadas de oito diferentes cidades, nomeálas e identificá-las graficamente.

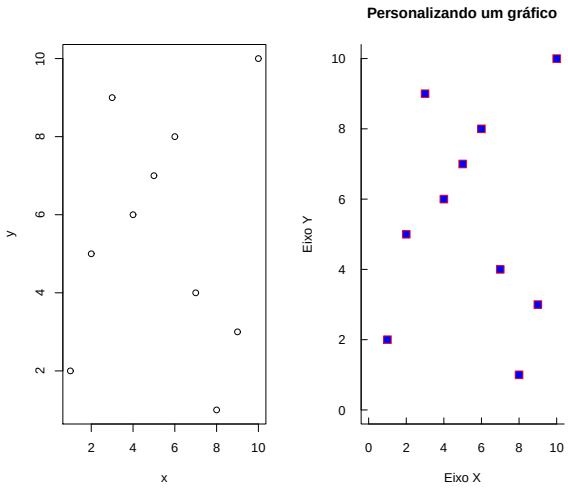
```
>  x <- c(2,3,4,5,6,7,8,9) #Representação das coordenadas
>  y <- c(15,26,45,8,74,11,61,32) #Coordenadas y das cidades
>  #Descrevendo o nome das cidades:
>  nomes <- paste("cidade", LETTERS[1:8], sep= " ")
>  cidades <- data.frame(x,y,row.names=nomes) #Juntando os dados
>  cidades #visualizando o objeto cidades
         x  y
cidade A 2 15
cidade B 3 26
cidade C 4 45
cidade D 5  8
cidade E 6 74
cidade F 7 11
cidade G 8 61
cidade H 9 32
>  #Visualizando graficamente os pontos que representam as cidades:
>  plot(cidades)
>  #será descrito e o número de pontos a serem identificados:
>  identify(x,y,nomes,n=4)
[1] 3 6 7 8
```

Depois de adicionado o comando `identify()` e definidos seus parâmetros, deve-se clicar nos pontos que se deseja identificar.



O R permite acrescentar gráficos múltiplos basta através dos comandos `par(mfrow=c(x,y))` e `par(mfcol=c(x,y))` que apresentam comportamentos idênticos. No vetor `c(x,y)`, `x` define o número de divisões horizontais (linhas) e `y` o número de divisões verticais (colunas). Os parâmetros dos gráficos podem ser encontrados no help através do comando `?par`. Proceda com o exemplo abaixo:

```
>  par(mfrow=c(1,2))
>  x<-1:10
>  y<- c(2,5,9,6,7,8,4,1,3,10)
>  x;y
 [1]  1  2  3  4  5  6  7  8  9 10
 [1]  2  5  9  6  7  8  4  1  3 10
>  plot(x,y)
>  plot (x,y, xlab="Eixo X", ylab="Eixo Y",
+  main="Personalizando um gráfico", xlim=c(0,10), ylim=c(0,10),
+  col="red", pch=22, bg="blue", tcl=0.4, las=1, cex=1.5, bty="l")
```



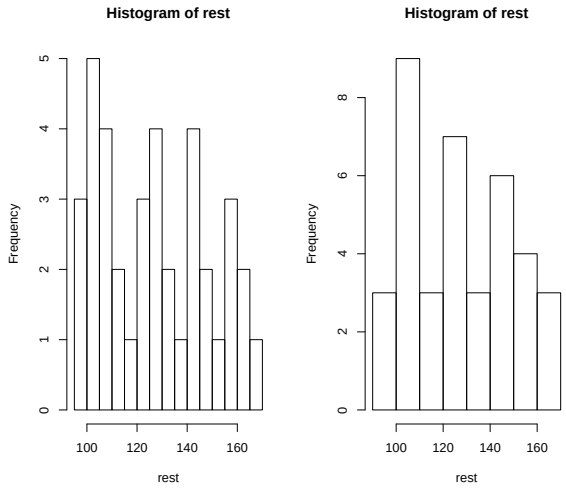
4.4 Gráficos de Análise Descritiva

Apresentaremos agora três gráficos fundamentais na análise descritiva dos dados: histograma, gráfico de barras e gráfico de caixas. São reconhecidos no R pelos nomes `hist`, `barplot` e `boxplot`.

4.4.1 Histograma

Um histograma divide uma série de dados em diferentes classes igualmente espaçadas e mostra a frequência de valores em cada classe. Em um gráfico, o histograma mostra diferentes barras, com bases iguais e amplitudes relativas às frequências dos dados em cada classe. O eixo das ordenadas, portanto, mostra a frequência relativa de cada classe e o eixo das abcissas os valores e intervalos das classes. Abaixo é apresentada a sintaxe do comando e um exemplo ilustrativo:

```
>  rest <- c(96,96,102,102,102,104,104,108,
+  126,126,128,128,140,156,160,160,164,170,
+  115,121,118,142,145,145,149,112,152,144,
+  122,121,133,134,109,108,107,148,162,96)
>  par(mfrow=c(1,2))
>  hist(rest,nclass=12)
>  hist(rest,nclass=6)
```



4.4.2 Barplot

A função `barplot(x, col=" ", legend.text=" ", xlab=" ",ylab=" ")` produz gráfico de barras, onde cada barra representa a medida de cada elemento de um vetor, ou seja, as barras são proporcionais com a "dimensão"do elemento. Onde temos:

x - é o vetor ou arquivo de dados;
col= - define-se a cor de exibição do gráfico de barras;
legend.text= - legenda do gráfico (o que representa a altura dos gráficos);
xlab= e ylab= - nome das grandezas expressas nos eixos x e y, respectivamente.

Faremos um exemplo simples utilizando um vetor qualquer e após utilizaremos o dataset euro que descreve as taxas de conversão entre as diversas moedas e o euro nos países da união européia. Observe os gráficos e veja os argumentos utilizados na descrição da função `barplot()`.

```
>  x <- c(1,2,3,4,5,6,7)
>  barplot(x)
>  barplot(euro,xlab="Euro conversions",col="red",
+  legend.text="Taxa")
```

Teremos como resultado os seguintes gráficos:

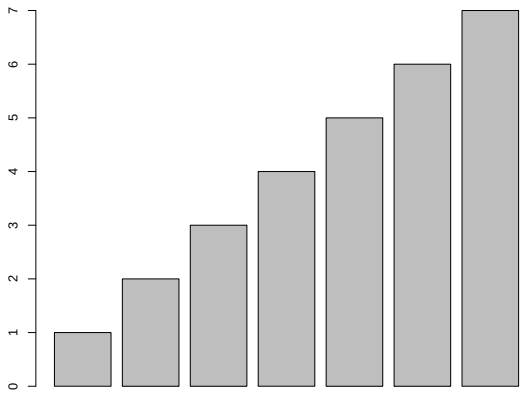


Figura 4.1 *Gráfico de Barras*

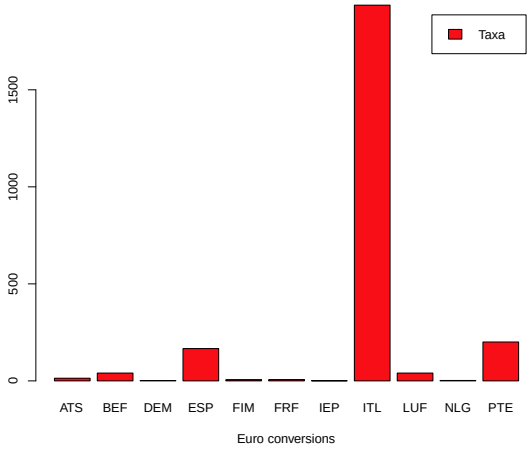


Figura 4.2 *Gráfico de Barras*

4.4.3 Boxplot

O boxplot é um gráfico que possibilita representar a distribuição de um conjunto de dados com base em alguns de seus parâmetros descritivos (mediana e os quartis). Ele permite avaliar a simetria dos dados e a sua dispersão. É especialmente recomendado para a comparação de dois ou mais conjuntos de dados correspondentes às categorias de uma variável qualitativa.

Podemos identificar em um boxplot os seguintes parâmetros:

- A linha central marca a mediana do conjunto de dados;

- A parte inferior da caixa é delimitada pelo primeiro quartil (Q_1) e a parte superior pelo terceiro quartil (Q_3);
- Podemos, com isso, verificar também o intervalo interquartil dado pela diferença entre o primeiro e o terceiro quartil ($IQR = Q_3 - Q_1$);
- As hastes inferiores e superiores se estendem, respectivamente, do quartil inferior até o menor valor não inferior a $Q_1 - 1.5 * IQR$ e do quartil superior até o maior valor não superior a $Q_3 + 1.5 * IQR$;
- Os valores inferiores a $Q_1 - 1.5 * IQR$ e superiores a $Q_3 + 1.5 * IQR$ são representados individualmente no gráfico sendo estes valores caracterizados como outliers, ou seja, que estão fora do intervalo $Q_1 - 1.5 * IQR < valor < Q_3 + 1.5 * IQR$;
- As quantidades $Q_1 - 1.5 * IQR$ e $Q_3 + 1.5 * IQR$ delimitam as cercas inferior e superior, respectivamente, e constituem limites para além dos quais, como visto, os dados passam a ser considerados outliers.

O comando utilizado no R é o `boxplot()`. Este comando possui vários argumentos. Utilize o comando `help(boxplot)` para maiores informações. $Exemplo_1$:

```
>  x = c(5,5,5,13,7,11,11,9,8,9)
>  y = c(11,8,4,5,9,5,10,5,4,10)
>  boxplot(x,y) #para plotar no mesmo gráfico (comparação)
>  boxplot(x);windows();boxplot(y) #para plotar em janelas diferentes
```

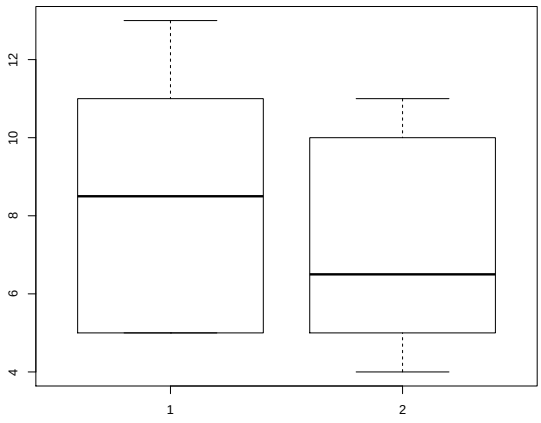


Figura 4.3 *Box plot dos dados x e y*

4.4.4 Gráfico de Ramo e Folhas

O gráfico de ramo e folhas é uma representação gráfica dos números que permite organizar os dados de forma a chamar a atenção para algumas características do conjunto de dados. São elas: forma da distribuição (simetria/assimetria), dispersão dos dados e existência de outliers.

O gráfico de ramo e folhas possui muita semelhança com o histograma, porém possui a vantagem de exibir o formato da distribuição sem que haja perda de informação. A desvantagem do ramo-e-folhas está no fato de ser um gráfico que deve ser utilizado com conjuntos dados de pequena dimensão.

Um gráfico de ramo e folhas é construído dispondo os dados em duas colunas: uma para os números inteiros (ramos) situada à esquerda, e outra à direita composta pelos números situados depois do ponto decimal dos dados (folhas). As colunas subsequentes à segunda coluna representam as diversas aparições de um mesmo ramo na série de dados em ordem crescente. As linhas apresentam os números dispostos em ordem crescente. Para a construção de um gráfico ramo e folhas no R basta utilizar o comando `stem()`. Veja o exemplo abaixo:

Exemplo$_1$:

```
> rf<-c(5.00,5.61,4.88,5.07,5.26,5.55,5.36,5.29,5.58,5.65,5.57,5.53,
+ 5.62,5.29,5.44,5.34,5.79,5.10,5.27,5.39,5.42,5.47,5.63,5.34,5.46)
>  stem(rf,scale = 2)

  The decimal point is 1 digit(s) to the left of the |

  48 | 8
  49 |
  50 | 07
  51 | 0
  52 | 6799
  53 | 4469
  54 | 2467
  55 | 3578
  56 | 1235
  57 | 9
```

Exemlo$_2$: Criar um vetor com notas de 19 alunos em uma prova da disciplina de Introdução à Probabilidade e à Estatística. E criar o gráfico de ramos e folhas.

```
> notas<-c(3.5,6.5,5.0,8.5,4.5,7.5,4.0,6.5,
+ 3.0,2.5,4.0,9.5,6.5,6.0,7.5,3.5,6.5,5.0,5.0)
> stem(notas,width = 19,scale = 2)

  The decimal point is at the |

  2 | 5
  3 | 055
  4 | 005
  5 | 000
```

```
6 | 05555
7 | 55
8 | 5
9 | 5
```

Observe que no exemplo acima as folhas contêm o último dígito decimal e os ramos se apresentam em sequencia. Se os números tiverem muitos algarismos significativos serão arredondados para a casa decimal que mais se aproximar do ramo.

4.4.5 Gráfico de Pizza

Gráficos de pizza exibem dados como proporção de um todo o que permite fazer comparações entre grupos. Este tipo de gráfico não apresenta nenhum eixo. Quando um dado é solto em um gráfico de pizza, o gráfico calcula a porcentagem de cada valor em relação a toda pizza. Para criarmos este gráfico utilizando o R fazemos uso da função `pie(dados, col = c(x)`.
Onde:
a: Vetor de dados;
x: Cores das partes do gráfico.

Exemplo$_1$: Criar um vetor de dados e criar seu gráfico de pizza.

```
a<-c(0.12, 0.3, 0.26, 0.16, 0.04, 0.12)
> names(a)<-c("a","b","c","d","e","f")
> pie(a,col = c("red","blue","green","gray", "brown", "black"))
```

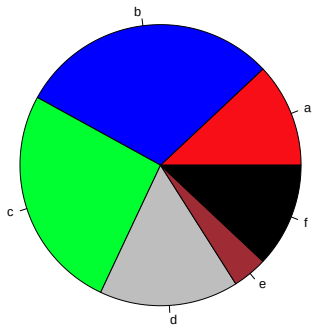


Figura 4.4 *Exemolo de um Gráfico de Pizza*