# Econometria com aplicações em R e C

Daniel Francisco Neyra Castañeda

2015

# Sumário

# Dedicatória

A mis padres Segundo y Maria.

# Agradecimentos

Este livro é fruto das notas de aula da disciplina de Métodos Quantitativos em Econometria, no curso de Estatística, na Universidade Federal de Sergipe, durante os últimos anos. Uma contribuição adicional a este livro são as disciplinas de estatística aplicada e de Séries Temporais, cursadas na pós-graduação da UFPE. Especial agradecimento ao professor Francisco Cribari, por ter norteado as disciplinas que atualmente ministro, agradecimentos também são estendidos ao professor Marcelo Gama, colega e integrante do curso de pós-graduação que participou da elaboração dos algoritmos na plataforma C.

O curso de Estatística na Universidade Federal de Sergipe, oferece atualmente duas disciplinas de Econometria ( I e II), e nos últimos anos abordou-se os modelos econométricos em algumas plataformas, tendo definido como padrão o **projeto R**, que alem de ser livre, segue uma linguagem S, as suas funções já estão implementadas nos pacotes do projeto R, estes pacotes e a plataforma sempre estão em constante atualização. O projeto R é uma linguagem de programação matemática orientada a objetos.
Este trabalho tenta contornar a necessidade de aplicar os métodos estatísticos na medida econômica tanto na área de programação do projeto R, e na programação C.

# Prefácio

Muitos programas de computador auxiliam as análises de dados de pesquisas, estas análises recaem em pacotes estatísticos que no mercado são pagos, ou alugados com licenças que expiram periodicamente. No entanto o programa R, supera estes impasses, proporcionando um programa livre (o seu código fonte esta disponível para alteração pelo usuário) e gratuito (sem custo monetário) usando a linguagem de programação S, do SPLUS. Este programa é confiável pela qualidade dos seus resultados, e pela estabilidade em ambientes como Linux e Windows, usaremos este último ambiente neste livro. Uma outra linguagem como o C é proposta para construção de algoritmo e programas no decorrer do livro. Esta linguagem de programação de aplicações é a mais comum no sistema operativo Windows. Nesta primeira edição, o objetivo inicial, é apresentar um livro funcional e simples que atenda o aluno que cursam a disciplina de Econometria. Para contornar a totalidade dos conteúdos do livro, sugerimos dividi-lo em duas partes ou em duas disciplinas.

## 0.1 Parte I

Na introdução definimos a metodologia adotada em todas as unidades, e aplicamos um exemplo motivador para despertar a curiosidade da econometria.

No capítulo 2, apresentamos uma breve introdução aos comandos do R, as bibliotecas utilizadas, manipulação de números, vetores e matrizes, multiplicação e inversão de matrizes, leitura de dados de arquivos em vários formatos, assim como construção de gráficos como histogramas, boxplot, funções acumuladas, plot de ajuste dos modelos de regressão alem dos testes estatísticos mais usados como: Qui-quadrado, t-student, F.

No capítulo 3 abordamos o modelo de regressão linear simples, estendemos ao modelo de regressão múltipla. O tratamento matricial do modelo de regressão, e a estimativa de parâmetros e seus respectivos intervalos de confiança, são sugeridos nestas regressões. As perturbações ou resíduos estocásticos, alem disso método de mínimos quadrados ordinários e de máxima verosimilhança, para as estimativas dos parâmetros, e suas propriedades assim como o teorema de Gaus - Markov, coeficiente de determinação, os experimentos de Montecarlo com e sem violação de pressupostos básicos, Análise de variância (ANOVA).

No capítulo 4 damos atenção a linguagem de programação C, construindo

replicas de amostras com os experimentos de Montecarlo para a valiar a normalidade dos erros no modelo de regressão linear simples utilizando diferentes erros do tipo I.

No capítulo 5 é tratado casos especias de regressão como ANOVA em delineamentos, como inteiramente causalizado e bloco causalizado, assim como modelos de efeitos fixos e aleatórios.

No capítulo 6 atenção para a superação dos pressupostos básicos como Heterocedasticidade, Multicolinearidade, Autocorrelação, considerando a natureza, identificação e correção.

## 0.2 Parte II

Na segunda parte, iniciando o capítulo 7, com modelos não lineares entre estes os modelos exponenciais e logísticos, também aborda-se modelos de regressão para respostas binárias, em particular usamos os modelos não lineares como logit, probit, clog-log.

O capítulo 8 correspondente a séries temporais, identificamos uma série usando a decomposição dos seus elementos, como sazonalidade e tendência. Abordamos métodos de suavização, iniciando com os algoritmos de Holt e de Holt Winters, modelos autorregressivos (AR), médias moveis (MA) e integrados ARIMA$(p, d, q)$, acrescentamos também modelos sazonais como o SARIMA, e modelos com presença de memoria longo ou ARFIMA. Para séries de alta volatilidade os modelos não lineares estruturais como ARCH e GARCH. Finalizamos este capítulo com raízes unitárias.

No capítulo 9 apresentamos simulações no ambiente R, como o desempenho de modelos AR e MA no R, Simulações para determinar o verdadeiro nível de significância entre outras.

O último capítulo é destinado a geração de números aleatórios, apresentamos a bateria de testes Diehard, útil para o processo de obtenção de dados em simulações, desenvolvido por George Marsaglia. Dois testes Diehard foram usados e propomos um novo gerador de números aleatórios, utilizando os dígitos do número $\pi$ como processo gerador. Estes testes são implementados no código de programação C.

# Capítulo 1

# Introdução

## 1.1 Como funciona a Econometria

Dados econômicos, precisam de uma atenção matemática para dar fundamento a suas teorias. Atualmente pelo avanço computacional não há barreiras para serem aplicadas, e torna-se até divertida se definimos a plataforma de trabalho. A econometria requer de uma abordagem diferenciada, pois faz uso de da teoria econômica, economia matemática, estatística econômica, estatística matemática e inferência estatística. Esta disciplina aproxima a Estatística da Economia, onde a teoria econômica formula as hipóteses. Por exemplo: uma redução de preços de uma mercadoria devera aumentar a quantidade demandada dessa mercadoria, considerando uma relação inversa entre preço e quantidade demandada, ou a despesa de consumo per capita depende da renda per capita, considerando uma relação direta, ou seja: a medida que a renda aumenta a despesa tende a aumentar. Outro exemplo onde consideremos a área microeconômica: consumo de água de uma residência e o valor pago, é natural imaginar que há uma relação direta entre consumo e valor pago pelo consumo. Porem precisamos saber qual e quanto é esta força de relação, por tanto o econometrista fornece esta informação usando a inferência estatística e a estatística econômica. As estimativas de esta relação fica por conta de estatística matemática. A econometria surge como uma disciplina autônoma em virtude de aglutinar todas estas áreas do conhecimento merecendo um estudo próprio.

## 1.2 Tipos de Dados

Os tipos de dados são:

1. Dados de corte transversal (cross section), são dados de uma ou mais variáveis coletadas no mesmo ponto do tempo. Por exemplo, os censos no Brasil são coletados num instante do tempo. No censo, variáveis demográficas são exploradas como: contagem da população, número de indivíduos em cada família, idade dos integrantes da família ou variáveis

econômicas, como: renda, consumo, tipo de moradia (alugada ou própria) etc., estes dados são adquiridas a cada década pelo IBGE.

2. Dados longitudinais ou de séries temporais, são caracterizadas por observações ao longo do tempo. No Brasil um site que disponibiliza dados desta natureza é o IPEADATA.

3. Dados de painel, é uma mistura de dados de corte transversal e longitudinais, ou ao longo do tempo estudamos o comportamento de unidades individuais. Por exemplo, séries mensal do índice de preços, série anuais do produto interno bruto (PIB), ou série anual da balança comercial dos municípios do Brasil, considerando cada município como unidades individuais.

## 1.3 Metodologia

Os passos da metodologia econométrica tradicional ou clássica se inicia desde a formulação da hipótese até as conclusões, em geral são:

### 1.3.1 Hipótese

Como referência usaremos a hipótese nula $H0$, a qual nulifica qualquer diferença, e a hipóteses alternativa $Ha$, aquela que contradiz, total ou parcialmente a hipótese nula. Exemplo:

1. $H0$: não existe uma relação entre consumo de água e valor pago pela mesma.

2. $Ha$: existe uma relação entre consumo de água e valor pago pela mesma.

A hipótese alternativa pode ser dividida em três possibilidades, no exemplo do consumo de água pode ser dividida em: existe alguma relação, ou a relação é negativa ou a relação é positiva. A primeira denomina-se de hipótese alternativa com teste bicaudal, o segundo de teste unicaudal á esquerda, e finalmente o último com teste unicaudal a direita. Nosso exemplo considere $Ha$: existe alguma relação (teste bicaudal)

### 1.3.2 Modelo matemático

Usaremos uma função de consumo simples

$$\hat{Y} = \beta_1 + \beta_2 X \tag{1.1}$$

A variável dependente $Y$, valor pago pelo consumo é estimado por $\hat{Y}$. A variável independente é $X$, consumo de água em $m^3$. $\beta_1$, representa o valor pago quando não há consumo de água. Nosso interesse é determinar a tangente ou a variação pelo consumo $\beta_2$.

### 1.3.3 Modelo econométrico do valor pago

O modelo matemático fornece a relação determinística entre valor pago em reais e consumo de água, porem as relações entre as variáveis econômicas são em geral inexatas. Assim ao longo do tempo no podemos esperar que o valor pago e o consumo em $m^3$ se situem exatamente numa linha reta, isto por que alem do consumo, outras variáveis afetam o consumo, por exemplo o tamanho da família, a reforma da casa, ou a inflação, podendo alterar o valor pago pelo consumo.

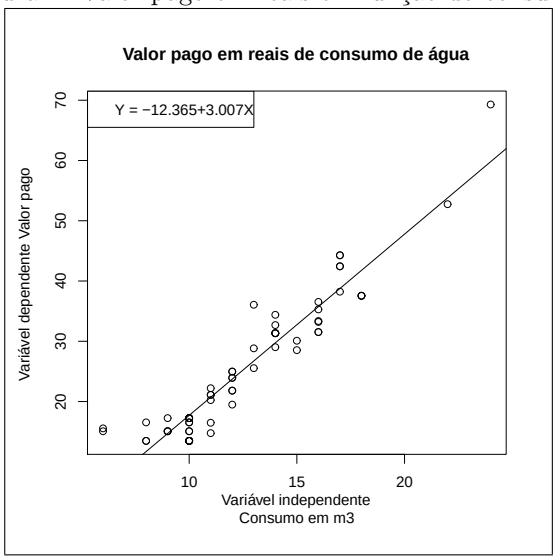
### 1.3.4 Nível de significância

Denominado de $\alpha$, frequentemente usa-se $\alpha = 0.05$; em situações de risco, usa-se $\alpha = 0.01$. Em testes bicaudais divide-se $\alpha$ em duas partes, e localizam-se nos extremos da distribuição de prova. Para testes unicaudais localiza-se no extremo da distribuição indicada pela hipótese alternativa. Nosso exemplo $\alpha = 0.05$ e por tanto $\alpha/2 = 0.025$.

### 1.3.5 Obtenção de Dados

Considere 65 observações (mês a mês) do consumo de água e valor pago em reais em uma residência, durante o período de 12/2005 a 04/2011. Ver dados no apêndice A. Um plot destes dados é dado a seguir:

figura 1. Valor pago em reais em função do consumo




Podemos observar visualmente que há uma relação positiva entre valor pago e consumo de água, onde a variação pelo consumo (tangente) $\beta_2$, cresce positivamente. Precisamos verificar se esta variação $\beta_2$ é estatisticamente diferente de zero (significativa).

### 1.3.6 Estatística de prova

Denominada de função pivô ou estatística de teste. No âmbito paramétrico usa-se as estatísticas: t student (amostras pequenas), quando $n$, é grande é indiferente o uso da t student ou normal. No exemplo para testar qual a inclinação da variação de consumo de água, usamos a seguinte estatística:

$$t_0 = \frac{\hat{\beta}_2}{dp(\hat{\beta}_2)} \approx t_{n-2gl} \tag{1.2}$$

onde: $t_0$ é o valor calculado da estatística $t$, $\hat{\beta}_2$ é a estimativa do parâmetro (variação de consumo) $\beta_2$, $dp(\hat{\beta}_2)$ é o desvio padrão da estimativa do parâmetro $\beta_2$, o valor $t_0$ segue uma distribuição t-student com $n-2$ graus de liberdade($gl$). Os graus de liberdade é calculado diferenciando o tamanho da amostra com o número de parâmetro do modelo (n-p). O valor quantilico $t_{n-2gl}(0.025)$, corresponde a distribuição t-student com 0.025 de significância com $n-2$ graus de liberdade, e $n$ é o número total de observações. Nosso exemplo $\beta_2 = 3.007$, e $dp(\beta_2) = 0.13$, por tanto $t_0 = 23.13$ é comparado com o valor da tabela da t-student com 63 $gl$ é com $\alpha/2 = 0.025$, sendo este valor igual 1.9983.

### 1.3.7 Tipos de erro

Ocorre o erro tipo I ($\alpha$) ao se rejeitar a hipótese de nulidade quando deveria aceitá-la. Por outro lado quando se aceita a hipótese de nulidade quando deveria rejeitá-la comete-se o erro tipo II ($\beta$). Nas duas situações tomou-se uma decisão errada. O interesse é a minimização dos erros tipo I e II que na prática só é possível aumentando-se o tamanho da amostra. Por razões diversas o aumento do tamanho da amostra muitas vezes torna-se impraticável. A gravidade do erro tipo I é distinta da do erro tipo II, quando se consegue identificar o erro mais grave a opção é a minimização deste, porem quando a probabilidade de ocorrência de um tipo de erro diminui a do outro aumenta e vice versa. Podemos observar este critério na seguinte tabela:

| | Não rejeita | rejeita |
|---|---|---|
| H0 verdadeira | Correto $(1-\alpha)$ | Erro do tipo I $(\alpha)$ |
| H0 falsa | Erro do tipo II $(\beta)$ | Correto $(1-\beta)$ |

### 1.3.8 Regra de Decisão

Os pacotes ou plataformas estatísticas apresentam o valor observado da função pivô e o $p-valor$ (Área de probabilidade está relacionada como nível de significância), assim a função pivô apresenta o valor numérico na função de densidade da Estatística, exemplo dentro da densidade da normal. Quando o $pvalor$ é menor do que $\alpha = 0.05$ no caso do teste unicaudal (e $\alpha = 0.025$ no caso bicaudal), concluímos que não há evidencia suficiente para aceitar H0. Caso contrario se o $p-valor$ for maior que $\alpha = 0.05$, podemos concluir que não há evidencia suficiente para rejeitar H0. Nosso exemplo: $t_0 = 23.13 > 1.998$,

(1.998 é valor da distribuição t com 63 gl e nível de significância bicaudal de 0.05), este valor localiza-se na cauda positiva da distribuição com 63 graus de liberdade($n-2=65-2$). Por outro lado o $p-valor < 0.025$, por tanto não há evidencias para aceitar H0, em favor de aceitar que o coeficiente de inclinação (ou tangente) é positiva. No recorrer do livro usaremos códigos para as medidas do $p-valor$, que em geral aparece em todo comando **summary()**. Como consequência a linha que aparece a seguir será excluída das saídas do R:

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

será interpretada da seguinte maneira:

- 0 *** significância menor que 0.001
- 0.001 ** significância menor que 0.01
- 0.01 * significância menor que 0.05
- 0.05 . significância menor que 0.1
- 0.1 significância menor que 1

### 1.3.9 Conclusões

Esta parte do teste de hipóteses é explicada em conjunto: O Estatístico e o Economista, que são os profissionais da área em que as observações foram coletadas, decidem sobre as providencias futuras. No exemplo podemos afirmar que a cada metro cubico a mais de consumo de água será pago em média 3 reais a mais. No momento este valor pago pode parecer alto, porem na falta de este liquido elementar, este valor pode aumentar e por tanto sofrerá mudanças.

### 1.3.10 Previsões

Muitas vezes estamos interessados em determinar o que acontece quando uma quantidade de consumo de água em $m^3$ é utilizada e qual o valor pago em reais. Por exemplo se nosso interesse é saber quanto sera pago por $20m^3$ de consumo de água em uma residência. A conta a realizar é:
Valor pago $= -12.365 + 3.007 * 20 = 47.775$ reais
A interpretação deste valor é: em média o valor pago em reais por $20m^3$ de água consumida é de aproximadamente 47.8 reais

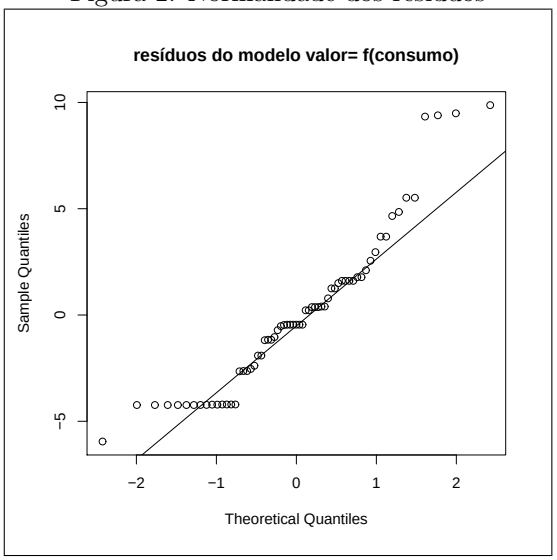
### 1.3.11 Resíduos

Uma questão fundamental em econometria é identificar o comportamento dos resíduos de um modelo. Considere $Y$, o valor real pago e $\hat{Y}$ valor pago estimado a partir de um modelo, então este erro é aleatório e definido por:

$$\epsilon_i = Y_i - \hat{Y}_i \tag{1.3}$$

A abordagem mais formal dos erros é feita nos próximos capítulos. Uma questão importante é determinar qual o comportamento da estimativa destes erros aos quais chamamos de resíduos. Para determinar a normalidade dos resíduos realizamos o teste de **Jarque Bera**, com a função **jarque.bera.test()** da biblioteca **tseries**, com o seguinte gráfico:

Figura 2. Normalidade dos resíduos




## 1.4 Exercícios

1. Por que a econometria precisa ser uma disciplina autônoma?
2. O que entende por Econometria?
3. Pesquise dados de corte transversal, corte longitudinal e de painel.
4. Construa uma metodologia econométrica para o consumo familiar em função da renda familiar (linear).
5. No item anterior é possível encontrar uma relação quadrática?
6. Quais são os pressupostos básicos de um modelo de regressão linear simples?
7. Como é chamado o paramêtro de inclinação ou tangente num modelo de regressão linear simples em econometria?
8. Que é o projeto R?
9. R é um programa econométrico?

# Capítulo 2

# Alguns comandos do R

R é um sistema de estatística computacional e gráfica, desenvolvido por um grupo de referis, que podem ser identificados no site, http://www.r-project.org. Este projeto de software segue as orientações do GNU Geral Public License(GPL). Para instalar o programa do R temos que estar conectado a internet e baixar o programa no site, http://cran.r-project.org

## 2.1 Iniciando o R.

Para iniciar o programa R clicamos no ícone que aparece na área de trabalho. A apresentação inicial do R e dada a seguir:

```
R version 2.13.1 (2011-07-08)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)

R é um software livre e vem sem GARANTIA ALGUMA.
Você pode redistribuí-lo sob certas circunstâncias.
Digite 'license()' ou 'licence()' para detalhes de distribuição.

R é um projeto colaborativo com muitos contribuidores.
Digite 'contributors()' para obter mais informações e
'citation()' para saber como citar o R ou pacotes do R em publicações.

Digite 'demo()' para demonstrações, 'help()' para o sistema on-line de ajuda,
ou 'help.start()' para abrir o sistema de ajuda em HTML no seu navegador.
Digite 'q()' para sair do R.

[Área de trabalho anterior carregada]
```

## 2.2 Pacotes do R

Atualmente o R disponibiliza mais de 1000 pacotes para serem instaladas na sua biblioteca, porem cada área utiliza alguns destes, por exemplo, em econometria, podemos esta interessados inicialmente nos pacotes: "lmtest", "quantreg", "KernSmooth", "car", "strucchange", "AER", alem do pacote padrão "stats". Para séries temporais, podemos acrescentar os pacotes: "tseries", "forecast", "fracdiff", "vars", todos estes pacotes podem ser facilmente baixados (download) via internet, alem disso os seus tutoriais no formato htlm, pdf, ou ser procurados na ajuda (help) do R. para instalar pacotes do R uma lista completa esta disponível com o comando:

```
> install.packages()
```

Uma lista de pacotes que iniciam o R padrão é dada por:

```
> search()
[1] ".GlobalEnv"        "package:stats"     "package:graphics"
[4] "package:grDevices" "package:utils"     "package:datasets"
[7] "package:methods"   "Autoloads"         "package:base"
```

## 2.3 Dados permanentes e removendo objetos

Durante uma sessão no R, objetos são criados e guardados por nomes, para verificar estes objetos usamos o comando ou função

```
> objects( ) # ou >ls( )

[1] "a1"               "a2"             "aa"          "acei"
[5] "ads"             "aju"           "aju1"        "ajuste"
[9] "ajuste1"      ajuste2"       "ajustehw"           "ara"
```

Para remover objetos antigos ou sem uso utilizamos o comando rm:

```
>rm(a1, a2, aa, acei, ads, aju)
```

Para detectar os pacotes na biblioteca local

```
> library()
```

Para instalar o pacote **urca** disponível na biblioteca:

```
> install.packages("urca")
```

Um pacote instalado tem que ser chamado no R para realizar a tarefa. Para chamar o pacote **urca** já instalado.

```
> library(urca)
```

Para verificar os pacotes na biblioteca externa.

```
> update.packages()
```

Para ver a lista de pacotes instalados na biblioteca local:

```
> installed.packages()
```

## 2.4 Manipulação de números e vetores

Para declarar números podemos usar o seguinte comando

```
> v=10
> v
[1] 10
```

Alertamos que no R o símbolo $<-$ é equivalente ao símbolo $=$.
Para criar um vetor com os números 1, 3, 3.4, 6, 99, 3, com o nome q, usamos:

```
> q=c(1,3,3.4,6,99,3)
```

Outra forma alternativa é usando o comando:

```
> assign("q",c(1,3,3.4,6,99,3))
```

Comandos de soma diferença, multiplicação e divisão são usados com os sinais de +, -, *, /, para inverter q, podemos usar:

```
> 1/q
[1] 1.0000000 0.3333333 0.29411765 0.1666667 0.0101010 0.3333333
```

Para determinar como verdadeiro ou falso para os valores de q maiores do que 3 podemos realizar:

```
> q>3
[1] FALSE FALSE TRUE TRUE TRUE FALSE
```

para restringir valores de q maiores do que 3:

```
> q[q>3]
[1] 3.4 6 99
```

Podemos converter números em caracteres ou dígitos assim:

```
z2=c(0:9) # equivalente a z2=0:9
> digit=as.character(z2)
> digit
 [1] "0" "1" "2" "3" "4" "5" "6" "7" "8" "9"
```

Para retornar dígitos em números com o seguinte comando:

```
> d=as.integer(digit)
> d
 [1] 0 1 2 3 4 5 6 7 8 9
```

Para preparar uma variável para ingressar seus valores por exemplo

```
> e=numeric()
> e[3]=17
```

Onde o terceiro elemento de e é 17 e os dois primeiros ainda não definidos:

```
> e
[1] NA NA 17
```

É possível a escolha de elementos de 2 em 2 ou de 3 em 3 a seguir:

```
> alpha=z[2*1:5]
> alpha
[1]  2  4  0 NA NA
> z
[1] 1 2 3 4 5 0
> alpha=z[3*1:5]
> alpha
[1]  3  0 NA NA NA
```

Construimos um vetor qualitativo com 10 homens e 5 mulheres asi:

```
w=c(rep("homem",10), rep("mulher",5))
> w
 [1] "homem"  "homem"  "homem"  "homem"  "homem"  "homem"  "homem"  "homem"
 [9] "homem"  "homem"  "mulher" "mulher" "mulher" "mulher" "mulher"
```

Considere o tamanho do calzado no pé de cada individuo em w:

```
> pé=c(40,42,39,41,44,40,42,41,44,40,35,35,37,36,37)
```

Podemos identificar a distribuição de frequências de w e pé da seguinte forma:

```
> table(z,pé)
       pé
z       35 36 37 39 40 41 42 44
  homem  0  0  0  1  3  2  2  2
  mulher 2  1  2  0  0  0  0  0
```

Para construir gráficos para as tabelas usamos as funções:

```
> plot(table(z,pé))
> hist(table(z,pé))
> barplot(table(z,pé))
```

## 2.5 Funções matemáticas

Para utlizar as funções matemáticas, usamos os comandos:

```
> history()   # comandos usados previamente
> length(x)   # número de elemetos do vetor ou lista x
> log(x)      # log na base e de x
> exp(x)      # antilogaritmo de x
> log(x,n)    # log na base n de x
> log10(x)    # log na base 10 de x
> sqrt(x)     # raiz quadrada de x
> factorial(x)# factorial de x(x!)
> choose(n,x) # combinatoria:  n!/(x!(n-x)!)
> gamma(x)    # x (x-1)!, x inteiro
> lgamma(x)   # log natural da gamma em x
> floor(x)    # menor inteiro de x
> trunc(x)    # escolha do menor inteiro
> abs(x)      # valor absoluto de x
> cos(x)      # coseno de x
> sin(x)      # seno de x
> tan(x)      # tangente de x
> acos(x) # inversa coseno
> asin(x) # inversa seno
> atan(x)     # inversa tangente
> acosh(x)    # inversa coseno hiperbólica
> asinh(x)    # inversa seno hiperbólica
> atanh(x)    #inversa tangente hiperbólica
> round(x, digits=2)  # aproximação a duas casas decimais
> signif(x, digits=6) # x com notação científica de 6 dígitos
```

## 2.6 Funções estatísticas

```
> max(x)     # valor máximo de x
> min(x)     # valor mínimo de x
> sum(x)     # soma total dos valores de x
> mean(x)    # média aritmética de x
> median(x)  # mediana de x
> range(x)   # vetor de minimo e máximo de x
> var(x)     # variância de x
> cor(x,y)   # correlação de x e y
> sort(x)    # ordena em forma crescente os elementos de x
> rank(x)    # vector de filas em x
> order(x)   # valores originais de x
> quantile(x)# vetor contendo os quartis de x
> cumsum(x)  # vetor contendo a suma total elemento a elemento
> cumprod(x) # vetor contendo o produto elemento a elemento
```

```
> cummax(x)  # vetor não decrescente de x, substituindo valores
> cummin(x)  # vetor não crescente de x, substitui os valores
               pelo mínimo
> pmax(x,y,z)# vetor contendo o vetor maior entre x y ou z, e
               substituindo o máximo em cada posição.
> pmin(x,y,z)# vetor contendo o vetor maior entre x y ou z,
               e substituindo o mínimo em cada posição.
> colMeans(x)# calcula a média por colunas na matriz x
> colSums(x) # calcula os totais por coluna na matriz x
> rowMeans(x)# calcula a média por linhas na matriz x
> rowSums(x) # calcula os totais por linhas na matriz x
```

## 2.7 Cores

O R disponibiliza cores usando a função *palette()* com 8 cores, outra função é *colours()*, e que também podem ser identificados por um número já especificado por exemplo:

```
> teta=(1:20)*0.92
> plot(teta,sin(teta), col="red", pch=16,cex=2) #ou
> plot(teta,sin(teta), col=2, pch=16,cex=2)
```

Onde cex, representa o caracter de expansão da bolinha (quanto maior o número maior a bolinha). Para incrementar várias cores no mesmo gráfico podemos realizar o seguinte comando:

```
> plot(teta,sin(teta), col=1:20, pch=16,cex=2)
```

Para modificar a bolinha apenas modificamos pch:

```
> plot(teta,sin(teta), col=1:20, pch="+",cex=2)
```

Cores podem também ser incrementadas fora do corpo do gráfico

```
> plot(teta,sin(teta), main="seno", col.main=4, col=1:20, pch="+",cex=2)
```

Outras alterações de cores podem ser construídas com:

```
col.main    # cor para o título do gráfico
col.axis    # cor para os números dos eixos
col.lab     # cor para os títulos dos eixos
col.sub     # cor para o subtítulo do gráfico
```

A fonte pode ser alterada da seguinte maneira:

```
font.main    # fonte para o título do gráfico
font.axis    # fonte para os números dos eixos
font.lab     # fonte para os títulos dos eixos
font.sub     # fonte para o subtítulo do gráfico
```

Onde: 1 é texto normal, 2 preto, 3 itálico, 4 preto itálico para simbolo fonte. A orientação dos números pode ser alterada com o comando *las*=; onde: 1 horizontal a x, 2 perpendicular a x, 3 perpendicular a ambos eixos).

## 2.8 Fatores ordenados e desordenados

exemplo

```
estados=c("PE","AL","SE","BA","PB","BA","PB","AL","SE","BA","PB",
+ "SE", "BA","SE","BA")
> estadosf=factor(estados)
> estadosf
 [1] PE AL SE BA PB BA PB AL SE BA PB SE BA SE BA
Levels: AL BA PB PE SE
> levels(estadosf)
[1] "AL" "BA" "PB" "PE" "SE"
> salarios=c(60,49,59,56,49,48,56,59,60,69,66,58,47,58,68)
```

A função tapply é usada para reordenar vetores, calculando uma característica dos níveis, no exemplo suponha que os salários médio e o desvio padrão por cada trabalhador por dia nos estados seja dado a seguir:

```
> mestados=tapply(salarios,estadosf,mean)  #média
> mestados
   AL    BA    PB    PE    SE
54.00 57.60 57.00 60.00 58.75
> destados=tapply(salarios,estadosf,sd)   #desvio padrão
> destados
        AL         BA         PB         PE         SE
 7.0710678 10.5498815  8.5440037         NA  0.9574271
```

Construímos a função do coeficiente de variação de duas formas distintas, e incluímos no comando tapply:

```
> cv=function(x) sd(x)*100/mean(x)
> cvestados=tapply(salarios,estadosf,cv)
> cvestados
       AL        BA        PB        PE        SE
13.094570 18.315767 14.989480        NA  1.629663

> cv=function(x) sqrt(var(x))*100/mean(x)
> cvestados=tapply(salarios,estadosf,cv)
> cvestados
       AL        BA        PB        PE        SE
13.094570 18.315767 14.989480        NA  1.629663
```

## 2.9 Vetores e Matrizes

Um vetor pode ser considerado como a entrada de uma coleção de dados, sendo estes numéricos ou não. R facilita a manipulação e construção de vetores, em particular casos como matrizes. Considere a construção de 30 números aleatórios Uniformes,

```
> z=runif(30)
> dim(z)=c(3,5,2)
> z
, , 1

          [,1]      [,2]       [,3]       [,4]      [,5]
[1,] 0.9500783 0.9262226 0.05847740 0.34257312 0.3718370
[2,] 0.8591887 0.7057183 0.20592054 0.03569694 0.6709746
[3,] 0.6806468 0.6270572 0.01570158 0.21650180 0.5170189

, , 2

          [,1]      [,2]       [,3]      [,4]      [,5]
[1,] 0.3539723 0.5918548 0.37104534 0.2148440 0.4169542
[2,] 0.8895178 0.5751237 0.69399721 0.3808024 0.3872492
[3,] 0.8036340 0.7065130 0.03400331 0.1593949 0.2306890
```

### 2.9.1 Indexando matrizes

```
> x=array(1:20,dim=c(4,5))
> x
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    5    9   13   17
[2,]    2    6   10   14   18
[3,]    3    7   11   15   19
[4,]    4    8   12   16   20
> i=array(c(1:3,3:1),dim=c(3,2))
> i
     [,1] [,2]
[1,]    1    3
[2,]    2    2
[3,]    3    1
> x[i]
[1] 9 6 3

# x[1,3]=9; x[2,2]=6; x[3,1]=3
```

O índice i indica os valores da matriz x a serem extraídos, temos

```
> x[i]=0  # Substituímos x[i] por zero.
> x
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    5    0   13   17
[2,]    2    0   10   14   18
[3,]    0    7   11   15   19
[4,]    4    8   12   16   20
```

Construindo uma matriz $3 \times 3$ com elementos de 1 ao 9.

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

### 2.9.2 Sequências

Sequencias são construídas, ingressando o primeiro elemento, o ultimo elemento, e o espaçamento.

```
> b=seq(10,14, by=.5)
> b
[1] 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5 14.0
> a=seq(1,9)
> ab=a%*%b # multiplicando vetores
> ab
     [,1]
[1,]  570
> ab=outer(a,b,"*")
> ab
      [,1] [,2] [,3]  [,4] [,5]  [,6] [,7]  [,8] [,9]
 [1,]   10 10.5   11  11.5   12  12.5   13  13.5   14
 [2,]   20 21.0   22  23.0   24  25.0   26  27.0   28
 [3,]   30 31.5   33  34.5   36  37.5   39  40.5   42
 [4,]   40 42.0   44  46.0   48  50.0   52  54.0   56
 [5,]   50 52.5   55  57.5   60  62.5   65  67.5   70
 [6,]   60 63.0   66  69.0   72  75.0   78  81.0   84
 [7,]   70 73.5   77  80.5   84  87.5   91  94.5   98
 [8,]   80 84.0   88  92.0   96 100.0  104 108.0  112
 [9,]   90 94.5   99 103.5  108 112.5  117 121.5  126

> ab=a%*%t(b)
> ab
      [,1] [,2] [,3]  [,4] [,5]  [,6] [,7]  [,8] [,9]
 [1,]   10 10.5   11  11.5   12  12.5   13  13.5   14
 [2,]   20 21.0   22  23.0   24  25.0   26  27.0   28
 [3,]   30 31.5   33  34.5   36  37.5   39  40.5   42
 [4,]   40 42.0   44  46.0   48  50.0   52  54.0   56
 [5,]   50 52.5   55  57.5   60  62.5   65  67.5   70
 [6,]   60 63.0   66  69.0   72  75.0   78  81.0   84
 [7,]   70 73.5   77  80.5   84  87.5   91  94.5   98
 [8,]   80 84.0   88  92.0   96 100.0  104 108.0  112
 [9,]   90 94.5   99 103.5  108 112.5  117 121.5  126
```

```
> d=outer(0:9,0:9)
> d
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0    0    0    0    0    0    0    0    0     0
 [2,]    0    1    2    3    4    5    6    7    8     9
 [3,]    0    2    4    6    8   10   12   14   16    18
 [4,]    0    3    6    9   12   15   18   21   24    27
 [5,]    0    4    8   12   16   20   24   28   32    36
 [6,]    0    5   10   15   20   25   30   35   40    45
 [7,]    0    6   12   18   24   30   36   42   48    54
 [8,]    0    7   14   21   28   35   42   49   56    63
 [9,]    0    8   16   24   32   40   48   56   64    72
[10,]    0    9   18   27   36   45   54   63   72    81
```

### 2.9.3 Multiplicando matrizes

```
>a= matrix(c(1:9),3,3)
> a
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> b= aperm(A,c(2,1))
>b
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
> a*b              # multiplicação elemento a elemento
     [,1] [,2] [,3]
[1,]    1    8   21
[2,]    8   25   48
[3,]   21   48   81
> a%*%b            # multiplicação das matrizes a e b
     [,1] [,2] [,3]
[1,]   66   78   90
[2,]   78   93  108
[3,]   90  108  126

> crossprod(a,b)  # produto da transposta de a por b

     [,1] [,2] [,3]
[1,]   30   36   42
[2,]   66   81   96
```

```
[3,]  102  126  150
> t(a)%*%b
     [,1] [,2] [,3]
[1,]   30   36   42
[2,]   66   81   96
[3,]  102  126  150
```

### 2.9.4 Inversão de matrizes

```
> c=matrix(c(3,2,4,5,7,2,1,5,9),3,3)
> d=matrix(c(2,2,3,5,8,2,8,5,8),3,3)
> c
     [,1] [,2] [,3]
[1,]    3    5    1
[2,]    2    7    5
[3,]    4    2    9
> solve(c)
            [,1]        [,2]        [,3]
[1,]  0.36551724 -0.29655172  0.12413793
[2,]  0.01379310  0.15862069 -0.08965517
[3,] -0.16551724  0.09655172  0.07586207

> solve(c,d)  # inverte a matriz c e multiplica por d:
           [,1]        [,2]       [,3]
[1,] 0.51034483 -0.29655172  2.4344828
[2,] 0.07586207  1.15862069  0.1862069
[3,] 0.08965517  0.09655172 -0.2344828

> solve(c)%*%d # forma alternativa:
           [,1]        [,2]       [,3]
[1,] 0.51034483 -0.29655172  2.4344828
[2,] 0.07586207  1.15862069  0.1862069
[3,] 0.08965517  0.09655172 -0.2344828
```

### 2.9.5 Comandos cbind() e rbind()

É possível formar novas matrizes com as já existentes, usando a função **cbind()** e **rbind()**, a primeira construí matrizes em colunas a segunda em linhas.

```
> cbind(c,d)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    3    5    1    2    5    8
[2,]    2    7    5    2    8    5
[3,]    4    2    9    3    2    8
```

```
> rbind(c,d)
     [,1] [,2] [,3]
[1,]    3    5    1
[2,]    2    7    5
[3,]    4    2    9
[4,]    2    5    8
[5,]    2    8    5
[6,]    3    2    8

> cbind(1,c)
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    1
[2,]    1    2    7    5
[3,]    1    4    2    9

> rbind(1,d)
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    2    5    8
[3,]    2    8    5
[4,]    3    2    8
```

### 2.9.6 Autovalores e autovetores

```
> ev=eigen(c)  # calcula autovalores e autovetores.
> ev
$values
[1] 13.265694+0.000000i  2.867153+1.646172i  2.867153-1.646172i

$vectors
             [,1]                   [,2]                   [,3]
[1,] 0.3799264+0i  0.7884141+0.0000000i  0.7884141+0.0000000i
[2,] 0.6480231+0i  0.0745376+0.3051027i  0.0745376-0.3051027i
[3,] 0.6600924+0i -0.4774265-0.2276482i -0.4774265+0.2276482

> det(c)
[1] 145
> det(d)
[1] -57
```

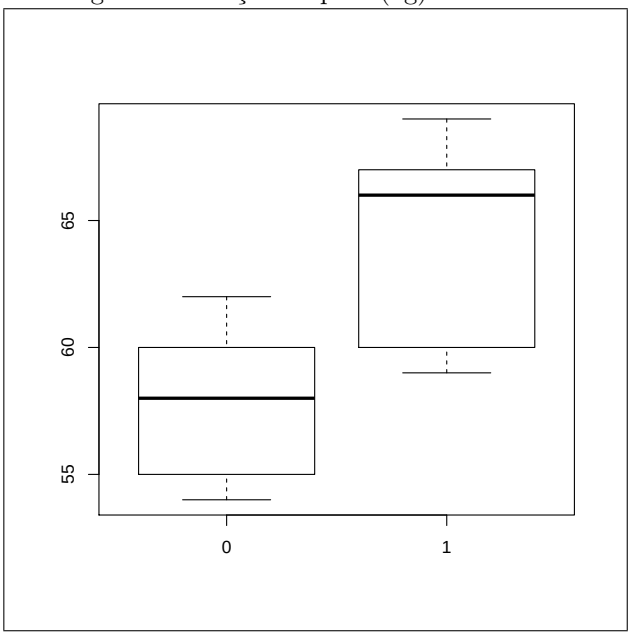
## 2.10 Data frame ou construtor de dados

Suponha um conjunto de dados com três variáveis: sexo (1=homem, 0=mulher), peso em quilogramas e salário em reais por dia de 12 indivíduos. Para calcular

a média, o resumo de estatísticas descritivas e um gráfico (plot) realizamos os seguintes comandos.

```
> ind=data.frame(sex=c(1,0,0,1,0,0,0,1,1,1,1,0),
+ peso=c(67,60,62,69,54,59,57,67,60,65,59,55),
+ salario=c(59,40,45,65,55,45,48,67,65,68,62,45))
> apply(ind,2,mean)
     sex     peso  salario
 0.50000 61.16667 55.33333
> summary(ind)
      sex           peso          salario
 Min.   :0.0   Min.   :54.00   Min.   :40.00
 1st Qu.:0.0   1st Qu.:58.50   1st Qu.:45.00
 Median :0.5   Median :60.00   Median :57.00
 Mean   :0.5   Mean   :61.17   Mean   :55.33
 3rd Qu.:1.0   3rd Qu.:65.50   3rd Qu.:65.00
 Max.   :1.0   Max.   :69.00   Max.   :68.00
>library(fields)
> bplot(ind$peso,by=ind$sex, xlab="homem=1  mulher=0",
+ ylab="peso", col=2)
> boxplot(peso~sexo) #forma alternativa
```

Figura 3. Relação do peso (kg) com o sexo



## 2.11 Funções attach() e detach()

Usamos estas funções para disponibilizar (attach()) ou remover (detach()) nas analises as variáveis internas de um conjunto de dados, pois os comandos:

```
> ind$peso
> ind$sex
```

Nem sempre é conveniente e de fácil manipulação. Veja no comando a seguir:

```
> attach(ind)
> peso
 [1] 67 60 62 69 54 59 57 67 60 65 59 55
> salário
 [1] 59 40 45 65 55 45 48 67 65 68 62 45
> sex
 [1] 1 0 0 1 0 0 0 1 1 1 1 0

> detach(ind)
> salário
Erro: objeto 'salário' não encontrado
```

A função attach e detach, pode ser usada no somente em arquivo de dados dos pacotes do R, se não também para listas e data frame

## 2.12 Função read.table()

Um amplo conjunto de arquivos com diferentes extensões pode ser lido por esta função.

#### Arquivos com extensão txt

Considere as seguintes variáveis $x1$: idade dos funcionários, $x2$: sexo (1 homem; 0 mulher), $x3$: anos de experiência na função, $x4$: estado civil, $x5$: número de filhos, $x6$: nota de $0-10$ de um teste de avaliação de conhecimento na sua área, $x7$: teste psicotécnico de 0 a 50, $x8$: satisfação com a vida pessoal. estes dados em bloco de notas com extensão txt, obtidos de Mingoti, 2005 [26]. Ver dados na integra no apêndice.

```
> funcao=read.table("func.txt",header=T)
> funcao

    grupo idade sexo ecivil nfilhos aexper tpsico tconhec satisf
1       0    24    1      1       0      2    7.7    36.0      1
2       0    29    1      1       0      4    7.9    36.0      1
3       0    38    1      0       1      6    8.6    40.8      0
```

```
.............................................................
107     0    24    1       1       0       2    7.3    36.0       0
108     0    27    1       0       0       3    7.4    36.0       1
109     1    42    1       0       2      14    9.9    44.0       1
```

Considere os dados de consumo de água de uma residência durante o período de 12/05 a 04/11. Com as seguintes variáveis: valor em reais, consumo em $m^3$, dias de consumo, e construindo o imóvel (1:sim, 2:não). Dados próprios do autor. Ver dados na integra no apêndice.

```
> ca=read.table("consumoagua2.txt",header=T)
> ca
   dado valor consumo diasconsumo Construindo
1     1 14.76      11          30           0
2     2 13.47      10          31           0
.........................................
64   64 34.39      14          31           1
65   65 22.21      11          32           0
```

Observação: use o comando choose(), para escolher um conjunto de dados de um diretório especifico no computador, podemos usar:

```
> data<-read.table(file.choose(),header=T)
```

### 2.12.1 Script com extensão .R

Exemplo com extensão R, lida por read.table com dados do própio R

```
> exemplo=read.table("exemplo.R",header=TRUE)
> exemplo
  g Year  NSW Vic.  Qld   SA   WA Tas.  NT ACT Aust.
1 1 1917 1904 1409  683  440  306  193   5   3  4941
2 2 1927 2402 1727  873  565  392  211   4   8  6182
3 3 1937 2693 1853  993  589  457  233   6  11  6836
4 4 1947 2985 2055 1106  646  502  257  11  17  7579
5 5 1957 3625 2656 1413  873  688  326  21  38  9640
6 6 1967 4295 3274 1700 1110  879  375  62 103 11799
7 7 1977 5002 3837 2130 1286 1204  415 104 214 14192
8 8 1987 5617 4210 2675 1393 1496  449 158 265 16264
9 9 1997 6274 4605 3401 1480 1798  474 187 310 18532
```

## 2.13 Função scan()

A função scan também pode importar dados com várias extensões, exemplo:

**Dados com extensão dat**

Considere um conjunto de dados já disponíveis pelo autor na pasta padrão do R.

```
inp=scan("input.dat")
Read 8 items
> inp
[1] 34 12 23 34 34 30 25 23
```

**Dados com extensão .R**

```
> arroz=scan("arroz.R")
Read 17 items
> arroz
[1] 5.992090 5.623515 5.452086 6.243138 6.101772 6.024657 5.108250
[8] 5.351473 4.754692 5.584979 5.979792 5.959100 5.250149 3.946691
[15]4.121597 4.687022 4.411315
```

Use a função scan(), para ingressar dados diretamente na plataforma R.

## 2.14 Função sample

Usa-se para escolher aleatoriamente uma amostra de um conjunto de observações ou uma base de dados. Como exemplo considere uma amostra de 8 observações dos primeiros 20, do banco de dados de consumo de água (ca)

```
> ca20=ca[sample(1:20,8),]
> ca20
   dado valor consumo diasconsumo Construindo
20   20 38.23      17          30           0
17   17 33.36      16          30           0
13   13 36.06      13          31           1
16   16 37.55      18          30           0
12   12 37.55      18          32           1
10   10 19.49      12          30           1
1     1 14.76      11          30           0
3     3 13.47      10          32           0
```

## 2.15 Rcmdr

Para importar dados do SPSS, STATISTICA, MINITAB, EXCEL, TXT, etc, podemos usar o pacote livre **Rcmdr**, já disponível na biblioteca.

```
> library(Rcmdr)
```

este pacote trabalha via janelas para maior comodidade.

## 2.16 Dados disponíveis no R

R possuí mais de 100 conjunto de dados alocados no pacote **dataset** que podem ser chamados diretamente ( sem extensão) com a função data.

```
> data()
```

No R pode-se acessar a conjunto de dados de outros pacotes disponíveis na sua biblioteca. Se por acaso esta instalado o pacote **lmtest**, o conjunto de dados ativo neste pacote pode ser chamado da seguinte maneira:

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

## 2.17 Modificando e editando dados

Considere o conjunto de dados anterior. Para modificar este conjunto será disponibilizado uma tela de editor de dados a seguir:

```
> xnovo=edit(wages,package="lmtest")
```

Para editar um conjunto de dados novo, podemos usar a função **edit** da seguinte forma:

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

## 2.18 Exportando dados do R para o Excel

Para exportar uma coluna de dados, por exemplo o valor pago no consumo de água, executamos no R

```
> writeClipboard(as.character(valor))
```

Na tela do Excel, usar o comando colar. Suponha que esta interessado em exportar o conjunto de dados **ca**, consumo de água numa residência, disponíveis no R, para transferir este conjunto de dados, use o seguinte comando

```
> write.table(ca,"clipboard",sep="\t",col.names=NA)
```

Na tela do Excel faça Ctrl+V ou um click em colar.

## 2.19 Funções: apply, tapply, sapply e lapply

A função apply, calcula medidas estatísticas por linhas ou por colunas, e a função sapply calcula apenas por coluna. A diferença da função apply da função tapply é que esta última calcula medidas estatísticas por estratos. Considere os dados de consumo de água de uma residência:

```
> apply(ca,2,mean)
       dado       valor     consumo diasconsumo Construindo
 33.0000000  25.5690769  12.6153846  30.7076923   0.2615385
> tapply(valor,Construindo,mean)
       0        1
21.15271 38.03882
> sapply(ca,mean)
       dado       valor     consumo diasconsumo Construindo
 33.0000000  25.5690769  12.6153846  30.7076923   0.2615385
> lapply(ca,mean) # verificar
```

## 2.20 Distribuições de probabilidade

R proporciona uma serie de funções para variáveis aleatórias estatísticas conhecidas, dentro destas funções temos distribuição, aleatorização, probabilidade e quantidade.

| **Distribuição** | **nome no R** | **argumentos adicionais.** |
|---|---|---|
| beta | beta | shape1,shap2,npc |
| binomial | binom | size,prob |
| cauchy | cauchy | location,scale |
| chi-quadrado | chisq | df,npc |
| exponencial | exp | rate |
| F | f | df1,df2,npc |
| gamma | gamma | shape,scale |
| geometric | geom | prob |
| hypergeometric | hyper | m,n,k |
| log-normal | lnorm | meanlog,sdlog |
| logistic | logis | location,scale |
| negativa binomial | nbinom | size,prob |
| normal | norm | mean,sd |
| poisson | pois | lambda |
| t-student | t | df,ncp |
| uniforme | unif | min,max |
| weibull | weibull | shape,scale |
| wilcoxon | wilcoxon | m,n |

Nomes iniciando com "d"calculara a densidade da função, "p"para calcular a probabilidade ou função acumulada, "q"para o valor quantilico e "r"para simulação de números aleatórios.
Exemplos:

```
> rpois(10,5) # 10 números aleatórios da dist. poisson com parâmetro 5
 [1] 3 2 3 1 8 6 1 3 2 3
> rnorm(5)    # 5 números aleatórios da normal padrão
[1]  0.04659011 -0.05019082 -1.28753167  1.15411245 -1.37573809
```

```
> qt(0.95,50) # valor quantilico da t, com 50 gl
[1] 1.675905
> qchisq(0.95,1) # valor da qui-quadrado
# com 95% de confiança e 1 gl
[1] 3.841459
> dpois(4,5) # densidade da dist poisson, x=4, parâmetro=5.
[1] 0.1754674
> ppois(4,5) # P(x<=4), parâmetro=5.
[1] 0.4404933
```

## 2.21 Algoritmos de programação no R

R é uma linguagem de programação orientada a objetos, que alem de analisar dados, têm sua própria programação, aqui usaremos esta programação para analises de estatísticas e econometria. Para maiores detalhes podemos pesquisar em Venables and Ripley (2000 e 2002). Voce pode estender a funcionalidade do R, criando novas funções ou programando por exemplo:

```
> mediana.media.razao <- function(x)
{
return(median(x)/mean(x))
}
> set.seed(20)
> z = rexp(10000)
> mediana.media.razao(z)
[1] 0.6925193
```

Construindo a mediana

```
> mediana=function(x) {
odd.even=length(x)%%2
if (odd.even == 0) (sort(x)[length(x)/2]+sort(x)[1+ length(x)/2])/2
else sort(x)[ceiling(length(x)/2)]
}
```

Construindo a média geométrica

```
> mediageometrica = function (x) exp(mean(log(x)))
```

Construindo a média harmonica

```
> mediaharmonica=function (x) 1/mean(1/x)
```

Um algoritmo que envolva as medidas de tendência central

```
> medidascentrais = function(y, medida) {
switch(medida,
média = mean(y),
```

```
mgeometrica = exp(mean(log(y))),
mharmonica = 1/mean(1/y),
mediana = median(y),
stop("Medida não inclusa"))
}
> medidascentrais(y,"média")
```

Gerando uma distribuição uniforme para o lançamento de um dado

```
> ndado=function(x){ return(round(runif(x)*6+0.5))}
> hist(ndado(100000)
```

Com o comando **sample**, podemos realizar este mesmo experimento da seguinte maneira:

```
> gerando=sample(1:6, 100000, replace=TRUE)
> hist(gerando)
```

Considere o seguinte objeto c

```
> c=c(1,2.2,7,15,3,6.5)
```

Para determinar que classe de objeto é o vetor c, podemos identificar usando os seguintes comandos:

```
> is.character(c)
[1] FALSE
> is.numeric(c)
[1] TRUE
```

A este objeto c, podemos a signar nome a cada valor definido da seguinte forma:

```
> names(c)=c("a","b","c","d","e","f")
> c
   a    b    c    d    e    f
 1.0  2.2  7.0 15.0  3.0  6.5
```

Podemos associar uma comparação logica com o vetor c, da seguinte forma:

```
> c>3
    a     b     c     d     e     f
FALSE FALSE  TRUE  TRUE FALSE  TRUE
```

Quando estamos diante de uma matriz, podemos definir os nomes das colunas com **colnames()**, e os nomes das linhas com **rownames()**, veja o exemplo:

```
> d=matrix(c(2,2,3,5,8,2,8,5,8),3,3)
> colnames(d)=c("c1","c2","c3")
> rownames(d)=c("1°","2°","3°")
> d
```

```
   c1 c2 c3
1°  2  5  8
2°  2  8  5
3°  3  2  8
```

A matriz d é entendida como um conjunto de dados que podem ser associados as linhas e as colunas. Para determinar se há associação entre linhas e colunas o teste utilizado pode se o qui-quadrado com a função **chisq.test()**

## 2.22 Laços no R

Em algumas situações variáveis contínuas podem ser transformadas em variáveis ordinais ou dummy, para realizar estas operações o R proporciona vários laços como else if ou ifelse:
Exemplo, para construir variáveis binárias, considere o consumo de água, onde consumo menor ou igual a 10 $m^3$ é zero e acima de 10 $m^3$ é 1. O valor de consumo sendo zero se o valor é menor ou igual a 20 reais, e 1 caso contrario.

```
consumoc=numeric(length(consumo))
for(i in 1:length(consumo)){if (consumo[i]<=10) consumoc[i]=0 else consumoc[i]=1}

valorc=numeric(length(valor))
for(i in 1:length(valor)){if (valor[i]<=20) valorc[i]=0 else valorc[i]=1}
```

Nota: Podemos construir os laços considerando a condição como texto, exemplo: consumoc[i]="$[0-10]$". Comando alternativo de construção:

```
consumoc2=ifelse(consumo<=10,"[0-10]","Acima de 10")
valorc2=ifelse(valor<=20,"[0-20]","Acima de 20")
```

limitação deste laço é que utilizado apenas para duas condições. Para laços com mais de duas condições podemos usar os seguintes comandos:

```
valorc3=numeric(length(valor))
for(i in 1:length(valor)){if (valor[i]<20) valorc3[i]="[0-20)" else if (valor[i]<30)
valorc3[i]="[20-30)" else valorc3[i]="[30 a +"}
table(valorc3)
```

### 2.22.1 A função cut()

Alguma funções usam laços para criar novas variáveis categóricas, no exemplo de valor pago em reais pelo consumo de água, podemos construir categorias por quartis de $.25, .50, .75, 1$, da seguinte maneira:

```
> q=cut(valor,quantile(valor), include.lowest=T)
> table(q)
q
[13.5,16.5] (16.5,22.2] (22.2,32.7] (32.7,69.3]
```

```
        17          16          16          16
```

Outras formas podem ser usando intervalos a critério pessoal, no exemplo a seguir, consideramos com valor mínimo 13 e valor máximo 70, com amplitude do intervalo de 19, e com classes fechadas no mínimo e aberto no máximo.

```
> valr=cut(valm,seq(13,70,19),right=F,include.upper=T)
> table(valr)
valr
[13,32) [32,51) [51,70)
     48      15       2
```

## 2.23 Criando a suas próprias funções

### 2.23.1 Teste que compara duas médias

Como exemplo implementamos a estatística de comparação de duas amostras independentes.

```
> testeduas <- function(y1, y2) {
+ n1 <- length(y1); n2 <- length(y2)
+ yb1 <- mean(y1); yb2 <- mean(y2)
+ s1 <- var(y1); s2 <- var(y2)
+ s <- ((n1-1)*s1 + (n2-1)*s2)/(n1+n2-2)
+ tst <- (yb1 - yb2)/sqrt(s*(1/n1 + 1/n2))
+ tst
+ }
> x=runif(20)
> y=rnorm(20)
> testeduas(x,y)
[1] 2.874636
```

Uma forma alternativa é usando a função **t.test()**

```
> t.test(x,y)

        Welch Two Sample t-test

data:  x and y
t = 2.8746, df = 21.784, p-value = 0.008858
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.1846283 1.1429380
sample estimates:
 mean of x  mean of y
 0.5539270 -0.1098562
```

### 2.23.2 Coeficiente de regressão

Implementando o calculo da tangente (inclinação) do coeficiente no modelo de regressão linear simples:

```
> mq1<- function(X, y) {
+ X <- qr(X)
+ qr.coef(X, y)
+ }
> mq1(x,y)
[1] -0.08755885
```

## 2.24 Teste Qui- Quadrado

Útil para associar linhas e colunas, considere os dados em d, com as seguintes hipóteses:
H0: não há associação entre linhas e colunas.
Ha: há associação entre linhas e colunas.

```
> chisq.test(d)

        Pearson's Chi-squared test

data:  d
X-squared = 4.6497, df = 4, p-value = 0.3252

Warning message:
In chisq.test(d) : Aproximação Qui-quadrado pode estar incorreta
```

Para um nível $\alpha = 0,05$ de significância, podemos afirmar que não há evidencias para afirmar que a associação entre linhas e colunas (linhas e colunas são independentes). Um outro exemplo do uso da distribuição Qui-quadrado é verificar se um conjunto de dados provem de uma determinada distribuição teórica:

```
> chisq.test(c)

        Chi-squared test for given probabilities

data:  c
X-squared = 22.549, df = 5, p-value = 0.0004116
```

Pelo p-value, podemos afirmar que a $0,05$ de significância não há evidencias para afirmar que a distribuição dos números em c sejam uniformes.
Para implementar este teste de associação no exemplo de consumo de água, considerando a classificação mediante laços (ver laços no R) entre consumoc e valorc. temos:

```
> table(consumoc,valorc)
        valorc
consumoc  0  1
       0 24  0
       1  3 38
> chisq.test(consumoc,valorc)$expected
> chisq.test(consumoc,valorc)$observed
> chisq.test(consumoc,valorc)

        Pearson's Chi-squared test with Yates' continuity correction

data:  consumoc and valorc
X-squared = 49.8015, df = 1, p-value = 1.701e-12
>barplot(table(consumoc,valorc), beside=T)
```

Concluimos que há associação entre consumo de agua e o valor pago em reais.

## 2.25 Teste t - student

O teste mais usado pelos estatísticos, é útil para comparar médias de duas amostras tanto independente como emparelhadas.
Exemplo, considere as exportações mensais (FOB-US) do Brasil durante os anos de $2009 - 2010$. Após digitação das observações, temos:

```
> exp2009
 [1]  9781.920  9586.406 11809.225 12321.617 11984.585 14467.785 14141.930
 [8] 13840.850 13863.222 14081.686 12652.892 14462.624
> exp2010
 [1] 11305.07 12197.24 15727.50 15161.21 17702.50 17093.91 17672.92 19236.25
 [9] 18832.79 18380.42 17687.33 20918.14
```

H0: A médias anuais das exportações durante 2009 e 2010 é a mesma.
Há: C.C.

```
> t.test(exp2009,exp2010,paired = FALSE)

        Welch Two Sample t-test

data:  exp2009 and exp2010
t = -4.2813, df = 18.202, p-value = 0.0004394
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -6075.666 -2077.757
sample estimates:
mean of x mean of y
 12749.56  16826.27
```

Conclusão: não há evidencias para aceitar que a média das exportações de 2009 e 2010 é a mesma.

Quando um conjunto de dados é dado em formato de matriz (data.frame), podemos realizar o teste t de uma coluna em função da outra, por exemplo no consumo de água, queremos testar a hipótese nula H0: o valor de consumo de água é o mesmo construindo ou não versus a hipótese alternativa Ha: Caso contrario.

```
> attach(ca)
> t.test(valor~Construindo)

        Welch Two Sample t-test

data:  valor by Construindo
t = -5.8383, df = 21.731, p-value = 7.465e-06
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -22.88867 -10.88356
sample estimates:
mean in group 0 mean in group 1
       21.15271        38.03882
```

Concluímos que não há evidencias para aceitar que o valor de consumo de água é o mesmo quando estamos construindo ( $p - valor < 0.05$)

Exemplos adicionais podem ser comparando entre as variáveis classificadas (ver seção de laços) da seguinte forma:

```
t.test(valor~consumoc)
t.test(consumo~valorc)
boxplot(consumo~valorc2, main="Consumo e valor pago em Reais")
boxplot(consumo~valorc3, main="Consumo e valor pago em Reais")
```

## 2.26 Teste F

Para determinar se há a mesma variabilidade no valor de consumo, quando estamos ou não construindo, utilizamos o teste F. A hipótese nula é, H0: O valor de consumo de água tem a mesma variabilidade quando construímos que quando não construímos. No R, temos:

```
> var.test(valor~Construindo)
        F test to compare two variances
data:  valor by Construindo
F = 0.4834, num df = 47, denom df = 16, p-value = 0.05478
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
```

```
 0.194815 1.014916
sample estimates:
ratio of variances
         0.4834048
```

Concluímos que não há evidencias para rejeitar a hipótese nula, por tanto a variabilidade no valor de consumo é o mesmo.
O correspondente gráfico (Box-plot) é apresentado a seguir:

```
> boxplot(valor~Construindo, col=8, names=c("não construção", "construção"))
```

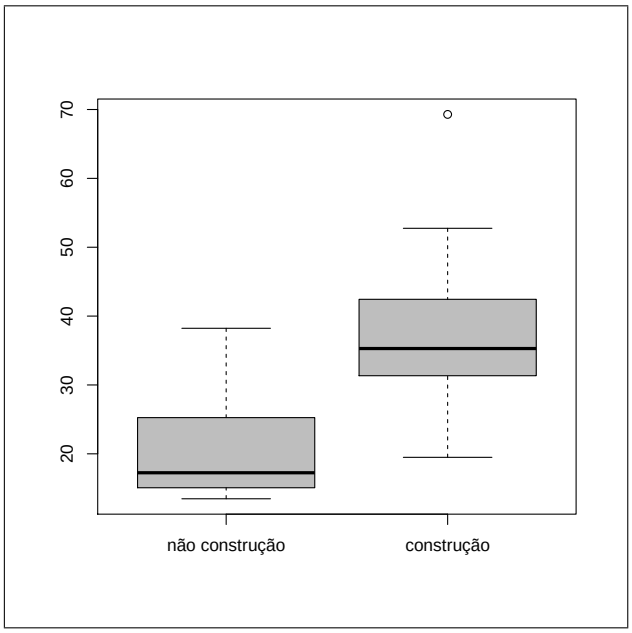



Figura 4. Valor pago em reais no consumo de água

Em alguns casos há necessidade de identificar se a variabilidade das exportações é a mesma comparando 2009 e 2010.
H0: As exportações de 2009 e 2010 tem a mesma variabilidade.
Há: C.C.

```
> var.test(exp2009,exp2010)
        F test to compare two variances
data:  exp2009 and exp2010
F = 0.3728, num df = 11, denom df = 11, p-value = 0.1166
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.1073346 1.2951624
sample estimates:
ratio of variances
         0.3728482
```

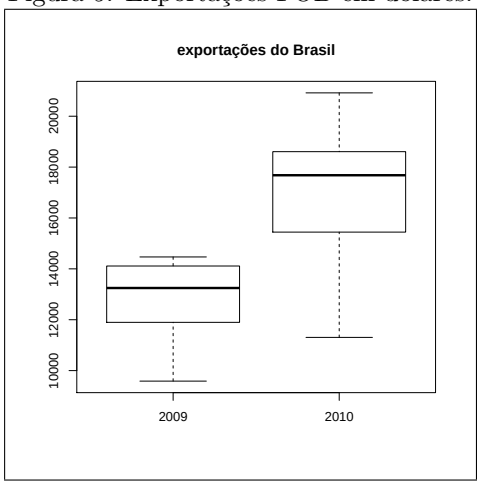
Conclusão: Não houve mudanças na variabilidade das exportações de 2009 e 2010.

## 2.27 Gráficos

Considere os dados mensais das exportações do Brasil (FOB) durante 2009 e 2010, para realizar um gráfico de box-plot, fazemos:

```
> boxplot(exp2009,exp2010, main="exportações do Brasil",
names=c(2009,2010))
```

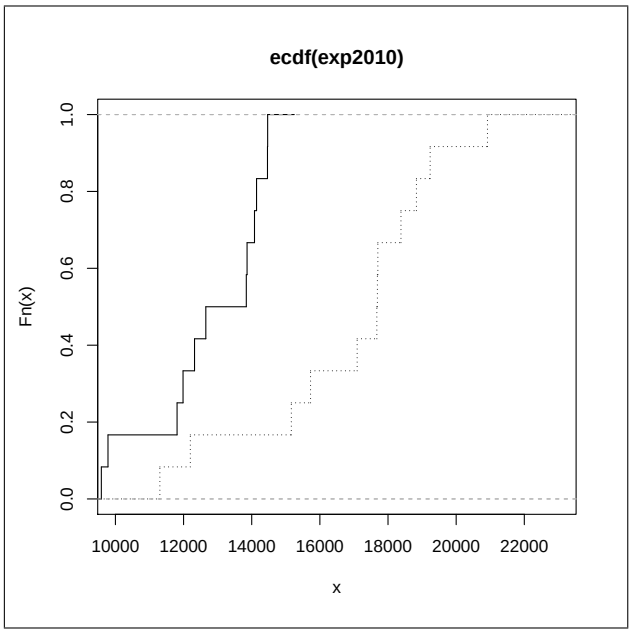
Figura 5. Exportações FOB em dólares.




Para o gráfico da função acumulativa das exportações:

```
> plot(ecdf(exp2010), do.point=FALSE, verticals=TRUE,
lty=3, xlim=c(10000,23000))
> lines(ecdf(exp2009), do.point=FALSE, verticals=TRUE,
col=2)
```

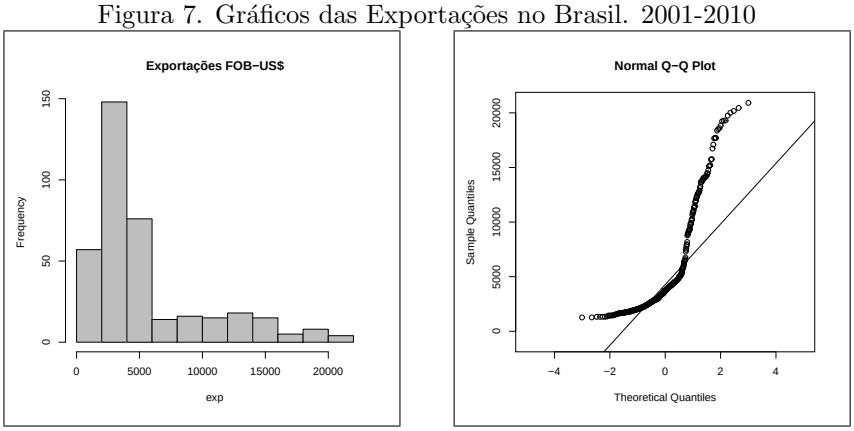
Figura 6. Função acumulada das exportações 2009-2010




Considere as exportações do Brasil, um histograma e Q-Q plot será:

```
> hist(exp, main="Exportações FOB-US", col=8)
> qqnorm(exp, xlim=c(-5,5),ylim=c(-1000,21000))
> qqline(exp)
```

Figura 7. Gráficos das Exportações no Brasil. 2001-2010

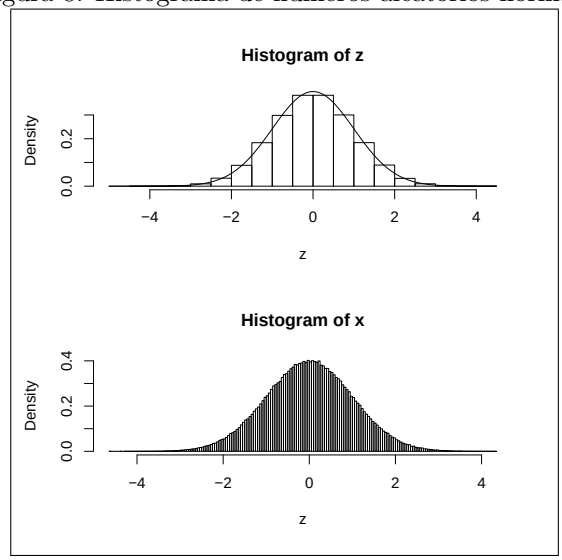



Para ativar alguns gráficos, pacotes são necessários, exemplo:

```
>library(MASS)
> par(mfrow=c(2,1))
> z = rnorm(500000)
> hist(z, prob=T)
```

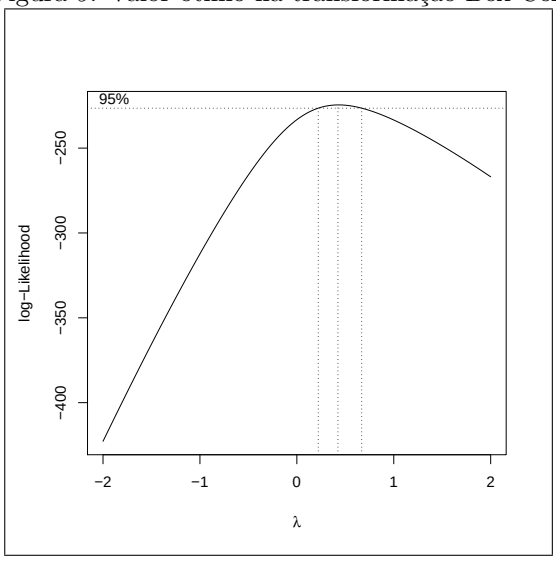
```
> curve(dnorm(x),add=T)
> hist.scott(z, prob=T)
```

Figura 8. Histograma de números aleatórios normais




```
> boxcox(dist~speed, data=cars) #transformação Box-Cox
```

Figura 9. Valor ótimo na transformação Box-Cox




Para ter um alcance maior sobre a capacidade gráfica do R, podemos usar as seguintes funções:

```
> demo(graphics)
```

```
> demo(image)
> demo(persp)
> demo(recursion)
> demo(plotmath)
> demo(package = .packages(all.available = TRUE))
```

## 2.28 Tabelas

Algumas tabelas estatísticas podem ser construídas com a função *table()*.

```
>estado=c("PB","PE","AL","SE","BA","MG","ES","RJ","PB","PE","AL",
"SE","BA","MG","ES","RJ","PB","PE","AL","SE","BA","MG","ES","RJ",
"PB","PE","AL","SE","BA","MG","ES","RJ")

> estadof=factor(estado)
> estadof
[1] PB PE AL SE BA MG ES RJ PB PE AL SE BA MG ES RJ PB PE AL SE BA
[22] MG ES RJ PB PE AL SE BA MG ES RJ
Levels: AL BA ES MG PB PE RJ SE
>set.seed(10)
>salarios=c(round(runif(32)*100))
> salarios
[1] 51 31 43 69  9 23 27 27 62 43 65 57 11 60 36 43  5 26 40 84 86 62 78 36 41
[26] 71 84 24 77 36 54  9
>salariom=tapply(salarios,estadof,mean)
> salariom
   AL    BA    ES    MG    PB    PE    RJ    SE
58.00 45.75 48.75 45.25 39.75 42.75 28.75 58.50
>salariof=factor(cut(salarios,breaks=0+15*(0:7)))
> salariof
 [1] (45,60] (30,45] (30,45] (60,75] (0,15]  (15,30] (15,30] (15,30] (60,75]
[10] (30,45] (60,75] (45,60] (0,15]  (45,60] (30,45] (30,45] (0,15]  (15,30]
[19] (30,45] (75,90] (75,90] (60,75] (75,90] (30,45] (30,45] (60,75] (75,90]
[28] (15,30] (75,90] (30,45] (45,60] (0,15]
Levels: (0,15] (15,30] (30,45] (45,60] (60,75] (75,90]
> table(salariof,estadof)
         estadof
salariof  AL BA ES MG PB PE RJ SE
  (0,15]   0  2  0  0  1  0  1  0
  (15,30]  0  0  1  1  0  1  1  1
  (30,45]  2  0  1  1  1  2  2  0
  (45,60]  0  0  1  1  1  0  0  1
  (60,75]  1  0  0  1  1  1  0  1
  (75,90]  1  2  1  0  0  0  0  1

> table(salariof,estadof)
```

```
          estadof
salariof   AL BA ES MG PB PE RJ SE
   (0,15]  1  1  0  0  1  0  1  2
  (15,30]  0  0  1  1  0  0  1  1
  (30,45]  0  0  2  0  0  1  0  0
  (45,60]  2  0  1  0  0  1  0  0
  (60,75]  0  1  0  2  3  0  1  0
  (75,90]  0  1  0  1  0  2  0  1
 (90,105]  0  1  0  0  0  0  1  0
```

## 2.29 Comandos adicionais

Em situações onde a ausência de informação ou dados faltantes(missing). as funções do R precisam declarar esta ausência com o comando na.rm=T, indicando que a informação apresenta dados faltantes. Exemplo

```
df=c(2,NA,5,8,NA) # O último elemento esta ausente
mean(f)        #Comando padrão para calcular a média
[1] NA
mean(f,na.rm=T) #Comando declarando a ausência de elementos
[1] 5
```

## 2.30 Exercicios

1. Construa uma sequência que inicia em $-5$, e finaliza em 5, separada por intervalos de 0.01.

2. Construa uma matriz de números aleatórios normais de $10 \times 10$, e outra matriz de números aleatórios uniformes de $10 \times 10$, e determine, o produto de matrizes, a inversão de cada matriz, o determinante das matrizes, os autovalores e autovetores do produto das matrizes.

3. Salve um conjunto de dados em formato **txt**, e importe estes dados no R, usando os comandos **read.table()** e **scan()**

4. Quais são os efeitos dos seguintes comandos:
   $qt(0.975, 3)$ $qnorm(0.975)$ $qt(0.975, 300)$

5. Qual a diferença dos comandos $log(16, 2)$ e $log(16)$, explique.

6. Considere os seguintes dados: salario=680,540,2680,3402,706,4009,1130,1480,1268, e escolaridade=1,1,3,3,1,3,2,2,2. Identifique detalhadamente a execução e os resultados no R.

7. Pesquise o que esta sendo calculado no seguinte algoritmo:

```
for(i in 1:5) print(1:i)
for(n in c(2,5,10,20,50)) {
x <- stats::rnorm(n)
cat(n,":", sum(x^2),"\n")
}
f = factor(sample(letters[1:5], 10, replace=TRUE))
for( i in unique(f) ) print(i)
```

8. No comando **if else** a seguir identifique o que esta sendo calculado

```
> if(rnorm(1)>0) 1 else 0
```

9. No comando **for** a seguir identifique o que esta sendo calculado

```
> x=c(1:10)
> for(i in 2:10){ x[i]=x[i]-x[i-1]}
```

10. use a função while() com o exemplo anterior.

11. Considere y=3,5,4,4,5,6,7,8,5,6, qual o efeito dos seguintes comandos:

```
> y>3 & y<5
> which(y>4 & y<=6)
> all(y>4)
> any(y<5)
> y+TRUE
```

12. num gráfico com o comando **plot(x)**, qual o efeito do comando **rug(x)**

13. pesquise os comandos: **scan() set.seed()**, **barplot(x)** e **pie(x)**

14. Quais são os dados disponíveis no pacote **tseries**.

15. Determine o efeito dos seguintes comandos:

```
> dbinom(c(2,3,4,5),10,0.5)
> pbinom(c(2,3,4,5),10,0.5)
> binom.test(c(682,243),p=3/4)
> chisq.test(c(20,23,15),p=c(1/3,1/3,1/3))
> chisq.test(c(20,23,15),p=c(1/3,1/3,1/3))$expected
> chisq.test(c(20,23,15),p=c(1/3,1/3,1/3))$observed
```

16. Gere $x$ uma amostra de 21 elementos considerando desde 1970 a 1990, gere $y = 2 * x + rnorm(21, 4, 2)$, e calcule um modelo de regressão.

17. qual o efeito dos seguintes comandos:

```
> g=matrix(rnorm(100),10,10)
> colnames(g)=variáveis
> boxplot(g)
```

18. Implemente a distribuição normal bivariada, como os seguintes comandos:

```
> x = seq(-4, 4, length=100)
> y = x
> normal.bi = function(x, y, rho=0.5){
z = 1/(2*pi*sqrt(1-rho^2))*exp(-1/(2*(1-rho^2))*
(x^2-2*rho*x*y+y^2))}
> z = outer(x,y,normal.bi,rho=0.8)
> persp(x,y,z)
```

19. Use as funções **floor()**, **ceiling()**, **rounded()**, para arredondar números

20. Identifique o que esta sendo executado no seguinte comando:

```
vendas<-c(4:6,NA,NA,4:7)
ifelse(is.na(vendas),0,vendas)
```

21. Considere uma tabela com números aleatórios Poisson, realize a seguinte sentencia e explique a execução:

```
st<-table(rpois(2000,2.3))
as.vector(st)
[1] 205 455 510 431 233 102 43 13 7 1
```

22. Identifique cada função no seguinte comando

```
y<-c(8,3,5,7,6,6,8,9,2,3,9,4,10,4,11)
sum(rev(sort(y))[1:3])
> xv<-rnorm(10,100,10)
> xv[seq(4,length(xv),2)]
```

23. Explique em palavras a seguinte execução e o resultado

```
> which(abs(xv-108)==min(abs(xv-108)))
[1] 6
> closest<-function(xv,sv){
 xv[which(abs(xv-sv)==min(abs(xv-sv)))] }
> closest(xv,108)
[1] 103.8979
```

24. Se um banco de dados denominado popbrasiles.txt, esta localizado num acessório externo f, este pode ser chamado da seguinte maneira:

```
> dados<-read.table("f:\\dados\\popbrasiles.txt",header=T)
```

25. Execute os comandos para os dados do consumo de água

```
> attach(ca)
> ranks<-rank(valor)
> sorted<-sort(valor)
> orderd=order(valor)
> view<-data.frame(valor,ranks,sorted,orderd)
```

26. use a função **sample()**, para sortear aleatoriedade 8 observações para os 20 últimos elementos dos dados de consumo de água.

27. hipoteticamente considere o número de pneus que as construtoras da formula 1 usam nos seus testes, usando as seguintes funções:

```
X<-matrix(rpois(20,1.5),nrow=4)
rownames(X)<-rownames(X,do.NULL=FALSE,prefix="Pneu.")
drug.names<-c("ferrari", "renault", "mercedes", "honda", "williams")
colnames(X)<-drug.names
```

Interprete a saída dos resultados.

28. Execute os seguintes comandos, e explique o resultado

```
lista1<-c(5,8,3,5,3,6,4,4,2,8,8,8,4,4,6)
lista2<-c(8,6,4,2)
match(lista1,lista2)
```

29. pesquise as seguintes funções: sweep(), dim(), ts(), sapply(), lapply(),paste().

30. use este comando para salvar a coluna do valor de consumo de água em formato txt

```
> write(valor,"c:\\valor.txt",1)
```

31. use este comando para salvar uma matriz de dados, exemplo **ca** (consumo de água) em formato txt

```
> write.table(ca,"c:\\consumodeagua.txt",col.names=T,row.names=T)
```

32. Construa a distribuição normal com valores críticos em $[-1.96, 1.96]$, com os comandos

```
> x<-seq(-3,3,0.01)
> y<-dnorm(x)
> polygon(c(x[x>=1.96],1.96),c(y[x>=1.96],y[x==3]),col=2)
> polygon(c(x[x<=-1.96],-1.96),c(y[x<=-1.96],y[x==-3]),col=2)
> legend("center","95%")
```

33. Use o seguinte algoritmo para determinar a distribuição da média de amostras uniformes, $U[0, 10]$

```
> medias=numeric(10000)
for (i in 1:10000){
medias[i]<-mean(runif(5)*10)
}
> hist(medias,ylim=c(0,1600))
```

34. Use Bootstrap, para determinar um intervalo de confiança para a média do valor em reais no consumo de água em uma residência

```
> attach(ca)
> a<-numeric(10000)
> for(i in 1:10000) a[i]<-mean(sample(valor,replace=T))
> hist(a,main="")
```

35. Use o comando ca[order(valor),] para ordenar ca em função do valor de consumo.

36. Implemente os intervalos de confiança da Normal e t com 3 gl com os seguintes comandos:

```
curve(dnorm(x), ylab="densidade",xlim=c(-5,5),
main="Duas distribuições e \n nível de confiança")
curve(dt(x,3),lty=3, add=T)
x=seq(-5,-1.96,length=100)
y=dnorm(x)
polygon(c(-5,x,-1.96),c(0,y,0),col="gray")
x2=seq(1.96,5,length=100)
y=dnorm(x2)
polygon(c(1.96,x2,5),c(0,y,0),col="gray")
x3=seq(-5,-3.18,length=100)
y=dt(x3,3)
polygon(c(-5,x3,-3.18),c(0,y,0),col=1)
x4=seq(3.18,5,length=100)
y=dt(x4,3)
polygon(c(3.18,x4,5),c(0,y,0), col=1)
text(-2,0.3,"Normal")
text(3,0.05,"t  3gl")
text(0,0.15,"95% ")
```

# Capítulo 3

# Regressão linear simples

Neste capítulo apresentaremos o modelo de regressão linear simples, e o modelo de regressão não linear (nos parâmetros), as equações dos modelos, suas propriedades e fundamentos, os pressupostos básicos para as variáveis independentes e para a variável dependente, assim como para a perturbação estocástica ou erro.

As variáveis são fatores de qualquer fenômeno, podemos classificar-las em dependentes e independentes, as dependentes são aquelas que recebem influência de outras variáveis, no área econométrica chamadas também de variáveis endógenas ou variáveis efeito. As variáveis independentes chamadas de exógenas, são aquelas que afetam o controlam a variável dependente.

## 3.1 Objetivo

De uma maneira geral, a partir dos dados (observações) realizar inferências sobre uma população. Em particular estudar o comportamento de uma variável (dependente) quando esta se relaciona com o comportamento de uma ou outras variáveis (independente).

## 3.2 Definição

Quando falamos de modelo linear estamos tratando de modelos que são lineares nos parâmetros [14], assim um modelo de regressão linear simples ou análise de regressão de duas variáveis considere como variável independente $X$ e como variável dependente $Y$ de tal forma que, admitindo que $E(Y|X)$ seja linear em $X$, possamos escrever $Y_i = E(Y|X_i) + \epsilon_i$ ou

$$Y_i = \beta_1 + \beta_2 X_i + \epsilon_i, \tag{3.1}$$

onde $\epsilon_i$ é definido como termo de erro estocástico ou perturbação estocástica com distribuição normal com média 0 e variância $\sigma^2$, $\beta_1$ definido como parâmetro

do intercepto, e $\beta_2$ como o parâmetro de inclinação (ou tangente). Com esta definição, são relevantes:

1. Linearidade de impacto: X deve ser a média de Y.
2. A variância é constante ou homoscedástica.

Para que o modelo seja tratável é necessário algumas suposições:

1. $Y_i = \beta_1 + \beta_2 X_i$ linearidade do modelo
2. $v(Y) = \sigma^2$ $(0 < \sigma^2 < \infty)$, $\quad \forall X$
3. Cada Y é não correlacionado com os demais
4. X deve assumir pelo menos dois valores distintos
5. Y possuí distribuição normal (nem sempre necessário)
6. O erro aleatório é dado por $\epsilon = Y_i - \beta_1 - \beta_2 X_i$

Até aqui temos definido o modelo populacional, porém quando nos limitamos a questões práticas surge a ideia de usar uma amostragem de valores de $Y$ correspondentes a alguns valores fixos de $X$ e nossa equação (3.1) será dada pela reta de regressão amostral e pode ser escrita como:

$$y_i = \widehat{\beta}_1 + \widehat{\beta}_2 x_i + \widehat{\epsilon}_i \tag{3.2}$$

A estimativa $\widehat{y}$ de $Y$ é definida como:

$$\widehat{y}_i = \widehat{\beta}_1 + \widehat{\beta}_2 x_i$$

. Onde $\widehat{\beta}_1$, $\widehat{\beta}_2$, são estimativas dos parâmetros acima descritos, $\widehat{\epsilon}_i$ é a estimativa do erro (ou perturbação) denominado de resíduo aleatório.

Inicialmente são exigidos os seguintes pressupostos básicos:

1. **O modelo é linear**, nos parâmetros.
2. **Normalidade**, Os $\epsilon_i$'s têm distribuição normal.
3. **Média Zero**, $E(\epsilon_i) = 0$.
4. **Homoscedasticidade**, $V(\epsilon_i) = \sigma^2$, a variabilidade dos erros é constante.
5. **Independência das perturbações ou ausência de autocorrelação**, $E(\epsilon_i, \epsilon_j) = 0$.
6. $x$ **é determinística**.
7. O número de observações tem que ser maior que o número de parâmetros.
8. **Covariância zero** entre $\epsilon_i$ e $x_i$, ou $E(\epsilon_i x_i) = 0$.

9. O modelo de regressão está corretamente especificado.

10. Nenhum erro de medida nos $x$'s. As variáveis explicativas são medidas sem erro. Para maiores detalhes ver Gujarati [14].

Nossa tarefa agora é estimar a função de regressão. Aqui, apresentaremos dois métodos que são: *o método dos mínimos quadrados ordinários (MQO)* e o *método de máxima verossimilhança (MMV).*

## 3.3 Mínimos Quadrados Ordinários

Para estimar os parâmetros de regressão, inicialmente expressamos a perturbação como

$$\widehat{\epsilon}_i = y_i - \widehat{y}_i = y_i - \widehat{\beta}_1 - \widehat{\beta}_2 x_i. \tag{3.3}$$

Desta forma queremos determinar uma reta que esteja tão próxima quanto possível de $Y$ real. Para tanto, teremos que minimizar a soma dos erros quadráticos

$$\sum \widehat{\epsilon}_i^2 = \sum (y_i - \widehat{\beta}_1 - \widehat{\beta}_2 x_i)^2 \tag{3.4}$$

de modo a torná-la menor possível. Diferenciando em relação aos parâmetros $\beta_1$ e $\beta_2$ do modelo, obtemos as seguintes equações:

$$\sum y_i = n\widehat{\beta}_1 + \widehat{\beta}_2 \sum x_i, \tag{3.5}$$

$$\sum y_i x_i = \widehat{\beta}_1 \sum x_i + \widehat{\beta}_2 \sum x_i^2 \tag{3.6}$$

Resolvendo essas equações simultaneamente, obtemos

$$\widehat{\beta}_2 = \frac{n\sum y_i x_i - \sum y_i \sum x_i}{n\sum x_i^2 - (\sum x_i)^2}, \tag{3.7}$$

$$\widehat{\beta}_1 = \bar{y} - \widehat{\beta}_2 \bar{x} \tag{3.8}$$

Como $y$ é variável aleatória, se tiver distribuição normal, $\hat{\beta}_i$'s, são variáveis aleatórias e tem distribuição normal também. Se Y não tiver distribuição normal, $\hat{\beta}_i$'s serão aproximadamente normal, se o tamanho da amostra for grande.

## 3.4 Máxima Verosimilhança

Um outro método apresentado na literatura é o *método de máxima verossimilhança* (ver Carneiro [4]). Considere o modelo em (3.1). De acordo com os pressupostos de validade de um modelo econométrico, $y$ é um variável aleatória com distribuição normal, média $(\beta_1 + \beta_2 x)$ e variância $\sigma^2$. Considerando que os valores observados $y_1, y_2, \cdots, y_n$ são independentes, e $n$ é o total de observações,

então a função de probabilidade conjunta a partir da distribuição normal é dada pela função de verossimilhança:

$$L = p(y_1)p(y_2)\cdots p(y_n) = L(y_1, y_2, \cdots, y_n, \beta_1, \beta_2, \sigma^2)$$
$$= \prod_{i=1}^{n} 1/(2\pi\sigma^2)^{1/2} \exp[-1/2\sigma^2(y_i - \beta_1 - \beta_2 x_i)^2] \quad (3.9)$$

Para estimar os parâmetros $\beta_1$ e $\beta_2$ minimizamos a função de verossimilhança. Para facilitar os cálculos utilizamos a expressão

$$LogL = -\frac{n}{2}\log 2\pi - n\log\sigma - \frac{1}{2\sigma^2}\sum(y_i - \beta_1 - \beta_2 x_i)^2. \quad (3.10)$$

Desse modo, quando as derivadas parciais de LogL em relação aos parâmetros a serem estimados, $\widehat{\beta}_1$, $\widehat{\beta}_2$ e $\widehat{\sigma}^2$, são igualadas a zero, obtemos as equações (3.7) e (3.8).
Observação: para estima os $\beta_i$'s usando máxima verosimilhança precisa,os uma distribuição para os erros aleatórios, no caso uma normal, já mínimos quadrados esta restrição não é necessária. Assim os EMV dos $\beta_i$'s sob normalidade são exatamente os $\hat{\beta}_i$'s em MQO.

## 3.5 Teorema de Gauss - Markov

Considerando certas, as primeiras 5 suposições, $\widehat{\beta}$ é o melhor estimador linear não tendencioso (ou não-viesado) de $\beta$. Assim $\widehat{\beta}$ é o estimador de mínimos quadrados de $\beta$.

## 3.6 Covariância de $\beta$

A matriz de variâncias e covariâncias dos parâmetros estimados é dado por:

$$cov(\hat{\beta}) = \begin{bmatrix} \frac{\sigma^2\sum x_i^2}{\sum(x_i-\bar{x})^2} & \frac{-\sigma^2\bar{x}}{\sum(x_i-\bar{x})^2} \\ \frac{-\sigma^2\bar{x}}{\sum(x_i-\bar{x})^2} & \frac{\sigma^2}{\sum(x_i-\bar{x})^2} \end{bmatrix} \quad (3.11)$$

## 3.7 Previsões

Suponha que o interesse é prever o valor da variável dependente correspondente a um dado valor do regressor, digamos $x_0$, então:

$$y_0 = \beta_1 + \beta_2 x_0 + \epsilon_0 \quad (3.12)$$

cuja previsão é:

$$\hat{y}_0 = \hat{\beta}_1 + \hat{\beta}_2 x_0 \quad (3.13)$$

Onde o erro e previsão é dado por $y_0 - \hat{y}_0$, e o calculo da esperança é:

$$E(\beta - 1 - \beta_2 x_0 + \epsilon_0 - \hat{\beta}_1 - \hat{\beta}_2 x_0) = (\beta_1 - E(\hat{\beta}_1)) + x_0(\beta_2 - E(\hat{\beta}_2)) = 0 \quad (3.14)$$

Então a previsão não possui viés, ainda a variância do erro de previsão é dada por:

$$V(y_0 - \hat{y}_0) = \sigma[1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum(x_i - \bar{x})^2}] \tag{3.15}$$

Esta variância pode ser estimada por $\hat{V}(y_0 - \hat{y}_0)$, substituindo na equação: $\sigma$ por $\hat{\sigma}$.
$V(y_0 - \hat{y}_0)$ é mínima em $x_0 = \bar{x}$. Um intervalo de precisão de nível $1 - \alpha$ para $y_0$ é:

$$\hat{y}_0 \pm t_{\alpha/2,n-2}\sqrt{V(y_0 - \hat{y}_0)} \tag{3.16}$$

A qual é a previsão da variável aleatória $y_0$.
Suponha que desejamos inferir o valor médio da variável dependente correspondente a um dado valor do regressor, digamos $x_0$. Temos:

$$\eta_0 = E(y_0) = \beta_1 + \beta_2 x_0 \tag{3.17}$$

então, a esperança do erro é

$$E(erro) = E(\eta_0 - \hat{y}_0) = E(\beta_1 - \hat{\beta}_1 + x_0(\beta_2 - \hat{\beta}_2)) = 0 \tag{3.18}$$

e a variância é dada por:

$$V(erro) = V(\eta_0 - \hat{y}_0) = V(\hat{y}_0) = \sigma^2[\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum(x_i - \bar{x})^2}] \tag{3.19}$$

que pode ser estimado por:

$$\hat{V}(\hat{y}_0) = \hat{\sigma}^2[\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum(x_i - \bar{x})^2}] \tag{3.20}$$

Um intervalo de precisão de nível $1 - \alpha$ para $y_0$ é:

$$\hat{y}_0 \pm t_{\alpha/2,n-2}\sqrt{V(\hat{y}_0)} \tag{3.21}$$

cujo valor é a previsão para a média $\eta_0$.

## 3.8 Coeficiente de correlação e de determinação

Regressão e correlação, estão relacionados da seguinte maneira, no caso da regressão estamos interessados nas estimativas dos parâmetros do modelo, já na correlação estamos interessados no grau (e em que direção) de intensidade na relação entre a variável dependente com a independente.

### 3.8.1 Coeficiente de correlação

$$r = \frac{\sum X_i Y_i}{\sqrt{(\sum X_i^2)(\sum Y_i^2)}} \qquad X_i = X_i - \bar{X}, \qquad y_i = Y_i - \bar{Y} \tag{3.22}$$

O coeficiente de correlação é um número entre -1 e 1, e de fácil interpretação indica que quando se aproxima de -1 a relação é forte e inversamente proporcional (Quando a variável independente aumenta a dependente diminui) e no caso de se aproximar a 1, a relação é forte e diretamente proporcional. Quando este coeficiente se aproxima de zero esta relação é fraca. Esta correlação é denominada de correlação de Pearson, e no R, é usada com o seguinte comando:

```
>cor(x,y)  #valor da relação entre x e y
>cor.test(x,y) # teste para verificar a significância da correlação.
```

Tudo isto é valido sobre o pressuposto de conhecer a distribuição das variáveis( por exemplo normalidade). Contudo existe uma variante não paramétrica quando não dependemos da distribuição das variáveis. Há duas variantes não paramétricas como a correlação de Spearman ($\rho$), baseada nos ranks de correlação; a outra de Kendall($\tau$) é baseada na contagem de número de pares concordantes e discordantes. No R é implementado da seguinte maneira:

```
>cor(x,y) , method="spearman") #correlaão de spearman
>cor(x,y) , method="kendall") #correlação de kendall
>cor.test(x,y, method="spearman") #significância da correlação de spearman
>cor.test(x,y, method="kendall") #significância da correlação de kendall
```

Exemplo, no consumo de água em uma residência, para encontrar a relação entre valor e consumo:

```
> cor(valor,consumo)
[1] 0.9458655
> cor(valor,consumo,method="spearman")
[1] 0.9270707
> cor(valor,consumo,method="kendall")
[1] 0.8078577
```

### 3.8.2 Coeficiente de determinação

É o quadrado do coeficiente de correlação, e pode ser interpretado em percentual, assim mede em percentual quanto $X$ explica $Y$

$$R^2 = \left\{ \frac{\sum x_i y_i}{\sqrt{(\sum x_i^2)(\sum y_i^2)}} \right\}^2 \tag{3.23}$$

**Definição 1** *Um estimador $\hat{\theta}$ num espaço paramétrico ($\theta \in \Theta$), é dito ser não-viesado, se $E(\hat{\theta}) - \theta = 0, \ \forall \theta \in \Theta$*

## 3.9 Estimação de $\sigma^2$

O estimador de mínimos quadrados ordinários (EMQO) de $\sigma^2$ é dado por:

$$\hat{\sigma}^2 = \sum \hat{\epsilon}^2/n-2 \tag{3.24}$$

onde $\hat{\epsilon}$, são os resíduos dos modelo.
O estimador de máxima verosimilhança (EMV), assumindo normalidade dos erros, e dado por:

$$\tilde{\sigma}^2 = \sum \hat{\epsilon}^2/n \tag{3.25}$$

que é diferente de $\hat{\sigma}^2$, o EMQO. Como o EMV de $\sigma^2$ é viesado, quando n é grande, este estimador é assintoticamente não-viesado. Os estimadores de MQO é não viesado, pois

$$E(\hat{\sigma}^2) = \sigma^2, \qquad \forall\sigma^2 \tag{3.26}$$

Sobre normalidade temos que

$$\hat{\sigma}^2 \approx \frac{\sigma^2}{n-2} X^2_{n-2} \tag{3.27}$$

Asim:

$$E(\hat{\sigma}^2) = E(\frac{\sigma^2}{n-2} X^2_{n-2}) = \frac{\sigma^2}{n-2} E(X^2_{n-2}) = \sigma^2 \tag{3.28}$$

A variância é dada por:

$$V(\hat{\sigma}^2) = V(\frac{\sigma^2}{n-2} X^2_{n-2}) = \frac{\sigma^4}{(n-2)^2} V(X^2_{n-2}) = \frac{\sigma^4}{(n-2)^2} 2*(n-2) = \frac{2\sigma^4}{n-2} \tag{3.29}$$

Note que se $\sigma^2$ é grande, maiores serão os estimadores, e a variância dos estimadores decresce quando n cresce.

## 3.10 Intervalo de confiança

Com a suposição de normalidade dos erros, os parâmetros do modelo tem distribuição normal:

$$\widehat{\beta}_1 \approx N(\beta_1, var(\widehat{\beta}_1)) \qquad e \qquad \widehat{\beta}_2 \approx N(\beta_2, var(\widehat{\beta}_2)) \tag{3.30}$$

Seja $z_c$, um número real, onde $P(Z > z_c) = P(Z < -z_c) = \alpha/2$, $0 < \alpha < 1$, e $Z$, tem distribuição normal padrão. Caso $\sigma^2$ ser conhecido podemos determinar o intervalo de confiança de $\beta_i$, para $i = 1, 2$ como:

$$P[-z_c \leq \frac{\widehat{\beta}_i - \beta_i}{\sqrt{var(\widehat{\beta}_i)}} \leq z_c] = 1 - \alpha \tag{3.31}$$

ou:

$$P[\widehat{\beta}_i - z_c\sqrt{var(\widehat{\beta}_i)} \leq \beta_i \leq \widehat{\beta}_i + z_c\sqrt{var(\widehat{\beta}_i)}] = 1 - \alpha \tag{3.32}$$

Em geral desconhecemos $\sigma^2$, por tanto substituímos o estimador de $\sigma^2$, por:

$$\widehat{\sigma}^2 = \frac{\sum(\hat{\epsilon}_i^2)}{n-2} \tag{3.33}$$

## 3.11 Análise de variância

| Fontes de variação | $Gl$ | $SQ$ | $QM$ | $F$ | $p - valor$ |
|---|---|---|---|---|---|
| Regressão | 1 | $\sum \hat{y}_i^2$ | $\sum \hat{y}_i^2$ | $\frac{SQReg}{SQRes}$ | |
| Resíduo | $n-2$ | $\sum \hat{\epsilon}_i^2$ | $\sum \hat{\epsilon}_i^2/n-2 = \hat{\sigma}^2$ | | |
| Total | $n-1$ | $\sum y_i^2$ | | | |

$SQReg$: significa soma de quadrados da regressão.
$SQRes$: significa soma de quadrados dos resíduos.
$SQTotal = SQReg + SQRes$
Uma forma alternativa de calcular $R^2$ usando o ANOVA é dado por:

$$R^2 = \frac{SQRes}{SQTotal} = 1 - \frac{\sum \hat{\epsilon}^2}{\sum y_i^2 - n\bar{y}^2} \qquad 0 \leq R^2 \leq 1 \tag{3.34}$$

Observação: Se o modelo não tiver um intercepto, então $SQTotal \neq SQReg + SQRes$, pois $\sum \hat{\epsilon}^2 \neq 0$, e

$$SQTotal = SQReg + vies + SQRes \tag{3.35}$$

Para modelos sem intercepto use a medida $R^2$ não centrado, dado por:

$$R^2_{nc} = 1 - \frac{\sum \hat{\epsilon}_i^2}{\sum y_i^2} \tag{3.36}$$

onde: $y_i$ é uma variável centrada.

## 3.12 Exemplo 1.

Considere $x$ uma variável independente não estocástica com valores iniciados em 1 e finalizados em 20 com espaçamento de 0.1, a variável dependente $y$, sera gerada dos valores de $x$ acrescidas de um termo aleatório proveniente de uma distribuição normal padrão.

### 3.12.1 O modelo

$$Considere: \qquad x = 1, 1.1, 1.2, \ldots, 19.9, 20; \qquad y = x + N(0,1) \tag{3.37}$$

o modelo é dado por:

$$\widehat{Y}_i = \widehat{\beta}_1 + \widehat{\beta}_2 x_i$$

No R.

```
> set.seed(20)

> x=seq(1,20, by=0.1)

> y=x+rnorm(length(x))

> m1=lm(y~x)
```

### 3.12.2 A função lm()

É composta por vários comandos, para exemplificar considere u objeto m1:

1. O comando print

   Este comando apresenta o modelo proposto e as estimativas dos parâmetros do modelo

   ```
   > print(m1)      # ou simplesmente  >m1

   Call:
   lm(formula = y ~ x)

   Coefficients:

   (Intercept)            x
       -0.1749       1.0237
   ```

2. O comando summary
   Este comando apresenta um resumo de estatísticas do modelo de regressão.

   ```
   > summary(m1)

   Call:
   lm(formula = y ~ x)

   Residuals:
        Min       1Q   Median       3Q      Max
   -2.75273 -0.67023  0.05422  0.75149  2.14659

   Coefficients:
               Estimate Std. Error t value Pr(>|t|)
   ```

```
(Intercept) -0.17487    0.15254  -1.146    0.253
x            1.02367    0.01286  79.589   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.9801 on 189 degrees of freedom
Multiple R-squared: 0.971,     Adjusted R-squared: 0.9709
F-statistic:  6334 on 1 and 189 DF,  p-value: < 2.2e-16
```

3. O comando coef
   Este comando apresenta simplesmente as estimativas dos parâmetros do modelo de regressão

```
> coef(m1)

(Intercept)           x
 -0.1748678   1.0236721
```

4. O comando residuals
   Apresenta os resíduos do modelo de regressão

```
> residuals(m1)

           1            2   ...          190          191
 1.313880999 -0.437095965   ... -1.501733127  0.381029987
```

5. O comando fitted
   Este comando calcula os valores estimados da função dependente ($\hat{y}$)

```
> fitted(m1)

         1          2  ...        190        191
 0.8488043  0.9511715  ... 20.1962067 20.2985739
```

6. O comando predict
   Para este comando precisamos identificar os valores que realizaremos as previsões, considere que os valores para previsão de $x$, sejam $1, 2$ e $3$, o qual denominamos de **novo**:

```
> novo=data.frame(x=c(1,2,3))

> predict(lm(y ~ x), novo, se.fit = TRUE)

$fit
        1         2         3
0.8488043 1.8724764 2.8961485
```

```
$se.fit
        1         2         3
0.1412777 0.1303135 0.1197276

$df
[1] 189

$residual.scale
[1] 0.9800842
```

7. anova
   Este comando apresenta o análise de variância do modelo de regressão

```
> anova(m1)

Analysis of Variance Table

Response: y
           Df Sum Sq Mean Sq F value    Pr(>F)
x           1 6084.6  6084.6  6334.3 < 2.2e-16 ***
Residuals 189  181.5     1.0
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

8. O comando plot
   Este comando apresenta o gráfico das variáveis de interesse, considere o modelo m1, este objeto tem quatro gráficos internamente construídos, por tanto precisamos preparar a saída como o comando **par(mfrow)**, da seguinte maneira:

```
> par(mfrow=c(2,2))

> plot(m1, main="m1=lm(y~x)")
```

Figura 10. Gráficos do objeto m1

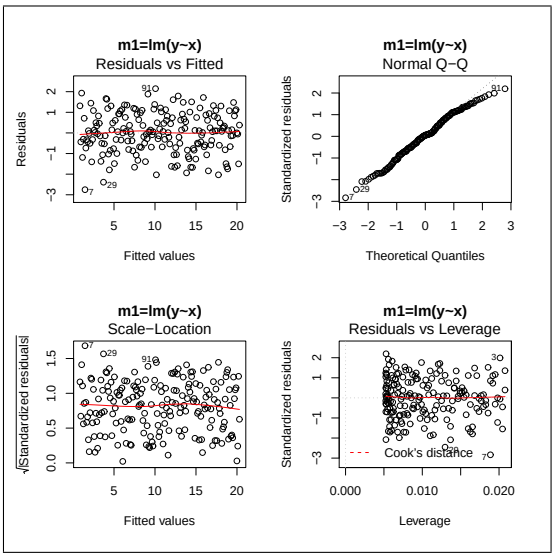
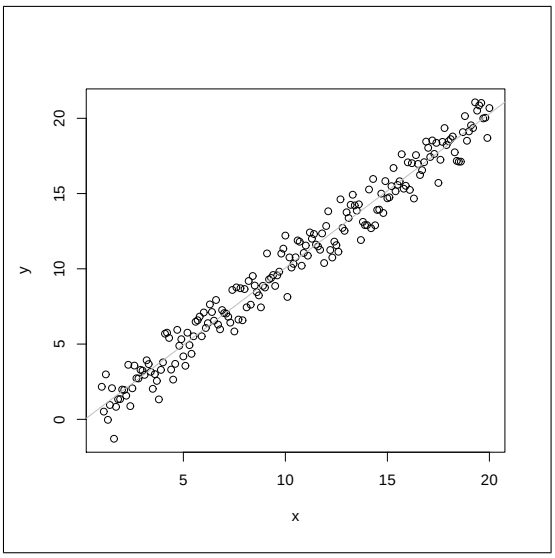



Figura 10. Gráficos do objeto m1

Outro gráfico de interesse é o ajuste do modelo, dado a seguir:

```
> plot(x,y)
> abline(m1$coef, col=8)   #$ Ajusta a reta de regressão
```

Figura 11. Gráfico do modelo de regressão linear simples do exemplo 1



9. O comando confint
   Determina o intervalos de 0.95 de confiança para os verdadeiros valores dos parâmetros do modelo de regressão.

```
> confint(m1)
                 2.5 %    97.5 %
(Intercept) -0.4757647 0.1260291
x            0.9983005 1.0490437
```

10. O comando vcov
    Determina a matriz de variâncias e covariâncias das estimativas dos parâmetros do modelo de regressão:

```
> vcov(m1)
             (Intercept)             x
(Intercept)  0.023268029 -0.0017370374
x           -0.001737037  0.0001654321
```

11. Comandos adicionais
    A soma de quadrados dos resíduos, a log- verosimilhança do modelo assumindo normalidade dos resíduos e os critérios de informação AIC e BIC (estes critérios são discutidos no capítulo de séries temporais) pode ser calculada como:

```
> deviance(m1)
[1] 181.5468
> logLik(m1) #função log-verosimilhança
log Lik. -266.1697 (df=3)
> AIC(m1)
[1] 538.3394
> BIC(m1)
[1] 548.0962
```

Comandos adicionais com graus de liberdade, ajuste de y, resíduos, e coeficientes estimados, podem também se calculados com os comandos:

```
m1$df;m1$fitted.values;m1$residuals;m1$coeff
```

## 3.13 Exemplo 2

No exemplo apresentado na introdução do livro referente ao consumo de água em uma residência, são observações de corte longitudinal ou uma série ao longo do tempo, usaremos a função de consumo simples dado em 3.1: Para esclarecimento usaremos os nomes das variáveis:

$$valorpago = \beta_1 + \beta_2 consumo \tag{3.38}$$

### 3.13.1 O modelo

```
> attach(ca)
> md=lm(valor~consumo)
> summary(md)

Call:
lm(formula = valor ~ consumo)

Residuals:
    Min      1Q  Median      3Q     Max
-5.9517 -2.6347 -0.4547  1.6074  9.8732

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -12.365      1.704  -7.258 7.09e-10 ***
consumo        3.007      0.130  23.132  < 2e-16 ***

Residual standard error: 3.721 on 63 degrees of freedom
Multiple R-squared: 0.8947,     Adjusted R-squared: 0.893
F-statistic: 535.1 on 1 and 63 DF,  p-value: < 2.2e-16
```

### 3.13.2 Resumo dos resultados

Para fortalecer a interpretação do exemplo motivador dado na introdução do livro, usaremos a saída do comando **summary()**. Inicialmente é apresentado o modelo de regressão proposto, posteriormente as estatísticas dos resíduos como valor mínimo, valor máximo, e os valores do primeiro segundo e terceiro quartil. Na parte dos coeficientes identificamos as estatísticas das estimativas dos parâmetros do modelo. O valor do intercepto $\hat{\beta}_1$ que corta o eixo vertical do valor de consumo é $-12,37$, com erro padrão de 1.704, o quociente de estes dois valores é $t_0 = -7,258$, num teste de hipótese bicaudal, se comparado com a estatística $t = t_{64}(0.025) = -1.99773$, temos que o valor observado $t_0 < t$, por tanto o parâmetro do intercepto é significativo (não há evidencias para aceitar a hipóteses nula H0, de que o intercepto é zero), isto pode ser confirmado por outra estatística chamada de $p - valor = 7.09e - 10$, o qual é menor que $\alpha/2 = 0.025$, concluindo que a localização do $p - valor$ está fora da área de aceitação da hipótese nula H0.
O valor de variação de consumo $\hat{\beta}_2$ é 3.007, indicando que a uma unidade de consumo em $m^3$, á um acréscimo de 3 reais em média no valor pago. O erro padrão de esta estimativa é 0.13, o quociente é $t_1 = 23.132$, num teste de hipótese bicaudal, $t = t_{64}(0.975) = 1.99773$, (valor observado maior que o tabular), por tanto o parâmetro do valor de consumo é significativo, isto pode ser confirmado por outra estatística chamada de $p - valor = 2e - 16$ o qual é menor que $\alpha/2 = 0.025$, o qual está fora da área de aceitação da hipótese nula H0.

### 3.13.3 Os gráficos

Região de aceitação de H0, num teste bicaudal.

Figura 12. Gráfico da distribuição t para um teste de hipótese.

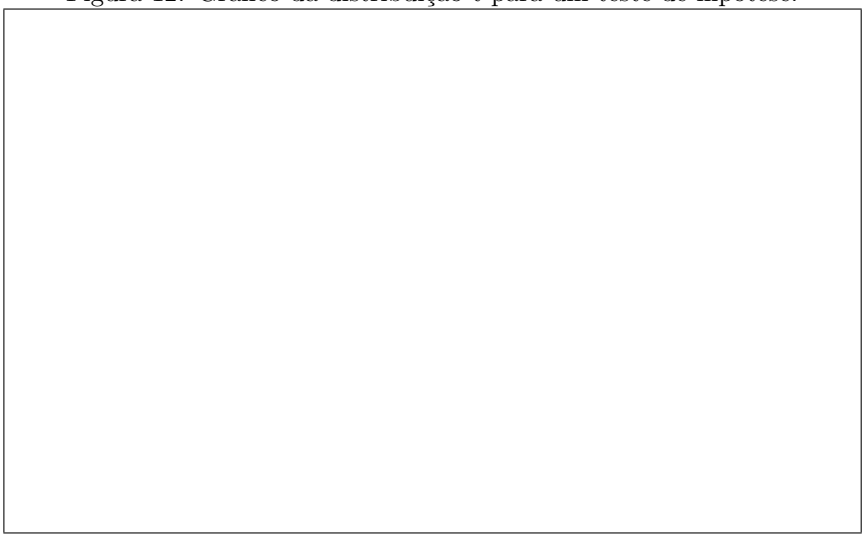
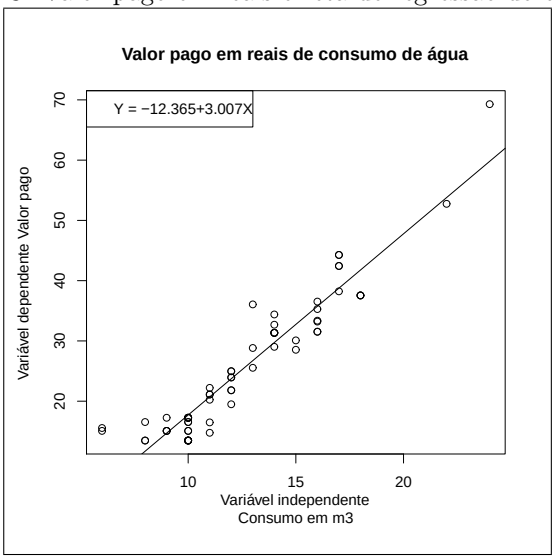


Figura 13. Valor pago em reais e reta de regressão do consumo




Acima observamos o gráfico de dispersão entre consumo e valor de consumo de água.

```
> plot(consumo,valor, main="Valor pago em reais de consumo de água",
ylab="Varíavel dependente Valor pago", xlab="Variável independente
Consumo em m3")
> legend("topleft", "Y = -12.365+3.007X")
> lines(consumo,fitted(md),lty=3)
```

### 3.13.4 ANOVA

Na parte da análise de variância é apresentado a resposta do consumo e dos resíduos, os graus de liberdade correspondentes são 1 e 63, a soma de quadrados do consumo é 7408.8, a soma de quadrados dos resíduos é 872.3, os quadrados médios correspondentes são calculados dividindo-os pelos graus de liberdade individuais, sendo o quadrado médio do consumo igual a 7408.8, é o quadrado médio dos resíduos, conhecido comumente com $\hat{\sigma}^2 = 13.8$. O valor da distribuição F, corresponde ao quociente dos quadrados médios do consumo sobre os quadrados médios dos resíduos, sendo o seu valor 537.07.

```
> anova(md)
Analysis of Variance Table

Response: valor
          Df Sum Sq Mean Sq F value    Pr(>F)
consumo    1 7408.8  7408.8  535.07 < 2.2e-16 ***
Residuals 63  872.3    13.8
```

### 3.13.5 Hipóteses do modelo

A tabela do ANOVA, é construída formulando os passos de um teste de hipótese:

1. H0: O modelo não é adequado (Conjuntamente os $\beta_i = 0$)
   Ha: Caso contrario (Conjuntamente os $\beta_i \neq 0$)
   Pela definição da nossa hipótese alternativa(Ha), nosso teste é bicaudal.

2. O nível de significância $\alpha = 0.05$

3. A estatística de prova F, é o quociente de duas qui-quadrado, no numerador o quadrado médio do consumo com 1 gl, e o denominador o quadrado médio dos resíduos ($\hat{\sigma}^2$) com 63 gl. Denominamos de valor calculado $F_0 = 535.07$.

4. O valor tabular corresponde a uma $F_{(1,63)}(0.975)$, no R é calculada por:

   ```
   > qf(0.975,1,63)
   [1] 5.272703
   ```

5. A regra de decisão é se $F_0 > F_{(1,63)}(0.975)$, não há evidencias para aceitar H0. Nosso exemplo $535.07 > 5.2727$, por tanto, concluímos em favor da hipótese alternativa, que conjuntamente os $\beta_i \neq 0$, ou o modelo é adequado.

### 3.13.6 As previsões

Para realizar ajustes de previsão, imaginemos que estamos interessados em saber quanto será o valor pago para um consumo de 20 $m^3$ e 25 $m^3$ podemos, realizar o seguinte comando:

```
> predict(md,newdata=data.frame(consumo=c(20,25)),
interval="confidence")
       fit      lwr      upr
1 47.77441 45.64589 49.90294
2 62.80927 59.46250 66.15605
```

Podemos interpretar que para um consumo de $20m^3$ de água, teremos em média um valor pago de 47.77 reais, com um intervalo de confiança que pode variar entre 45.65 e 49.90 reais no valor pago.
Para um consumo de $25m^3$ de água, teremos em média um valor pago de 62.81 reais, com um intervalo de confiança que pode variar entre 59.46 e 66.16 reais no valor pago.

### 3.13.7 Os resíduos

Podemos identificar que os resíduos do modelo proposto seguem uma distribuição normal ao longo do tempo, usamos o teste de normalidade de Jarque Bera. Atenção o teste com ks.test do pacote *stats*, é usado para normalidade de dados de coorte transversal.

```
>library(tseries)
> resíduos=residuals(md)
> jarque.bera.test(resíduos)

        Jarque Bera Test

data:  resíduos
X-squared = 9.5102, df = 2, p-value = 0.008608
```

Gráficamente a normalidade dos resíduos é obtido no R por:

```
> z=seq(-4,4,length=1000)
> respmodd=md$res/sd(md$res)
> plot(z, dnorm(z), main="resíduos padronizados e distribuição
teórica ", xlab="z", ylab="densidade", ylim=c(-0.1,0.5))
> lines(density(respmodd), col="red")
> legend(-3.8,0.38,legend=c("N(0,1)", "resíduos"),
col=c("black","red"),lty=1:1)
```

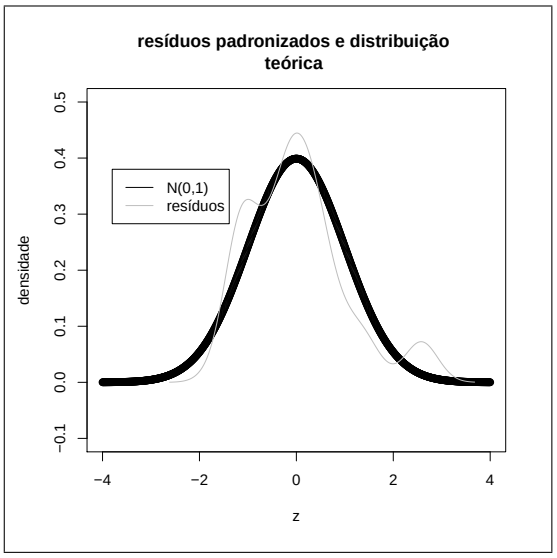



Figura 14. Distribuição empírica dos resíduos do modelo.

Pelo teste de normalidade de Jarque Bera, podemos afirmar que não há evidencias para aceitar a normalidade dos dados. Comandos gráficos adicionais podem ser obtidos com as seguintes funções:

```
qqnorm(residuals(md),plot.it=T)
qqline(residuals(md))
```

### 3.13.8 Componentes adicionais da função lm

Nosso modelo(md) tem componentes adicionais:

```
> names(md)
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "xlevels"       "call"          "terms"         "model"
```

A forma de ser chamados é :

```
> md$call
lm(formula = valor ~ consumo)
> md$rank
 [1] 2
```

## 3.14 O modelo quadrático

Em situações onde os dados não seguem uma tendência linear, uma proposta em favor do modelo quadrático, ou de potência, é adequada, no exemplo 2 do consumo de água, uma inspeção visual (ver figura 13) permite considerar um modelo quadrático, a fim de melhorar o ajuste da série de consumo de água.

### 3.14.1 Modelo matemático

$$y = \hat{\beta}_1 + \hat{\beta}_2 x + \hat{\beta}_3 x^2 + \epsilon \tag{3.39}$$

Para os dados do consumo de água:

$$valorpago = \hat{\beta}_1 + \hat{\beta}_2 consumo + \hat{\beta}_3 consumo^2 + \epsilon \tag{3.40}$$

### 3.14.2 Implementação no R

```
> m2=lm(valor~consumo+I(consumo^2))
> summary(m2)

Call:
lm(formula = valor ~ consumo + I(consumo^2))

Residuals:
    Min      1Q  Median      3Q     Max
-5.4546 -2.7179 -0.4616  2.1583 10.4806

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.89565    4.53139   0.639 0.525164
consumo       0.63696    0.67124   0.949 0.346337
I(consumo^2)  0.08523    0.02375   3.588 0.000658 ***

Residual standard error: 3.413 on 62 degrees of freedom
Multiple R-squared: 0.9128,     Adjusted R-squared:  0.91
F-statistic: 324.4 on 2 and 62 DF,  p-value: < 2.2e-16
```

Um modelo sem intercepto é proposto a seguir:

```
> m3=lm(valor~consumo+I(consumo^2)-1)
> summary(m3)

Call:
lm(formula = valor ~ consumo + I(consumo^2) - 1)

Residuals:
    Min      1Q  Median      3Q     Max
-5.4715 -2.6125 -0.2055  2.0475 10.3046

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
consumo      1.058471   0.123749   8.553 3.86e-12 ***
I(consumo^2) 0.070978   0.008153   8.705 2.10e-12 ***

Residual standard error: 3.397 on 63 degrees of freedom
```

```
Multiple R-squared: 0.9857,     Adjusted R-squared: 0.9852
F-statistic:  2168 on 2 and 63 DF,  p-value: < 2.2e-16
```

## 3.15 Modelos de potência

Os modelos de regressão potencial tem uma extensão natural a partir do modelo quadrático, asi podemos construir o modelo cúbico com a seguinte sentencia:

```
> m4=lm(valor~consumo+I(consumo^2)+I(consumo^3))
```

Outros modelos que o R permite construir é usando a função poly().

```
> m5=lm(valor~poly(consumo,3))
```

Para um modelo de potência $k$, basta fazer:

```
> mk=lm(valor~poly(consumo,k))
```

Nota: m4 e m5 não apresentam as mesmas estimativas.

## 3.16 Comparando modelos

Quando realizamos uma análise de regressão, estamos interessados se alguma variável independente não considerada no modelo é relevante para um modelo ser adequado, para comparar dois modelos já analisados usamos a função **anova()**, da seguinte forma

```
> anova(md,m2)
Analysis of Variance Table

Model 1: valor ~ consumo
Model 2: valor ~ consumo + I(consumo^2)
  Res.Df    RSS Df Sum of Sq      F   Pr(>F)
1     63 872.32
2     62 722.34  1    149.98 12.873 0.000658 ***
```

Podemos concluir que a inclusão do termo quadrático como regressor é significativo ($p - valor < 0,05$), e por tanto o modelo de regressão quadrático tem melhor performance de ajuste se comparado como o modelo de regressão linear simples. Para visualizar esta comparação podemos usar os seguintes comandos:

```
> plot(consumo,valor,axes=F)
> lines(consumo,fitted(md))
> lines(consumo,fitted(m2),col=2)
```

## 3.17 Derivação dos EMQO

A ideia intuitiva que temos para determinar os estimadores dos parâmetros do modelo de regressão linear simples é encontrar uma função que minimize os erros, ou que o ajuste do modelo seja o mais próximo possível da variável dependente Y, isto pode ser proposto da seguinte maneira:

$$\epsilon = Y - \hat{\beta}_1 - \hat{\beta}_2 X = Y - \hat{Y} \tag{3.41}$$

Como a esperança dos resíduos é zero: $E(\epsilon) = 0$, por hipóteses, não teria sentido minimizar uma função linear dos resíduos, consideraremos por tanto minimizar a soma dos quadrados dos resíduos, da seguinte forma:

$$Z = \sum_{i=1}^{n} \hat{\epsilon}^2 = \sum_{i=1}^{n} (Y - \hat{\beta}_1 - \hat{\beta}_2 X)^2 \tag{3.42}$$

Para determinar as estimativas dos parâmetros derivamos parcialmente a equação acima em termos de $\hat{\beta}_1$ e $\hat{\beta}_2$, da seguinte maneira:

$$\frac{\partial Z}{\partial \hat{\beta}_1} = 2 \sum (Y - \hat{\beta}_1 - \hat{\beta}_2 X)(-1) = 0 \tag{3.43}$$

$$\sum Y = \sum \hat{\beta}_1 + \sum \hat{\beta}_2 X \tag{3.44}$$

$$\sum Y = n\hat{\beta}_1 + \hat{\beta}_2 \sum X \tag{3.45}$$

$$\frac{\partial Z}{\partial \hat{\beta}_2} = 2 \sum (Y - \hat{\beta}_1 + \hat{\beta}_2 X)(-X) = 0 \tag{3.46}$$

$$\sum XY = \sum \hat{\beta}_1 X + \sum \hat{\beta}_2 X^2 \tag{3.47}$$

$$\sum XY = \hat{\beta}_1 \sum X + \hat{\beta}_2 \sum X^2 \tag{3.48}$$

Para um sistema de duas equações e duas incógnitas, podemos determinar as estimativas dos parâmetros da seguinte forma:

$$\hat{\beta}_2 = \frac{\sum (XY) - (\sum Y * \sum X)/n}{\sum X^2 - (\sum X^2)/n} = \frac{\sum (Y - \bar{Y})(X - \bar{X})}{\sum (X - \bar{X})^2} \tag{3.49}$$

$$\hat{\beta}_1 = \bar{Y} - \hat{\beta}_2 \bar{X} = \frac{\sum Y - \hat{\beta}_2 \sum X}{n} \tag{3.50}$$

## 3.18 Extensão da Regressão Linear Simples

Uma extensão natural do modelo de regressão linear simples é o modelo de regressão múltipla:

$$y = \beta_1 + \beta_2 x_2 + \beta_3 x_3 + ... + \beta_p x_p + \epsilon \tag{3.1}$$

e na forma matricial escrita como:

$$y = X\beta + \epsilon, \tag{3.51}$$

onde:

- $y$ é um vetor $n \times 1$ de observações da variável dependente;
- $X$ é uma matriz fixa contendo observações sobre as variáveis explicativas, de dimensão $n \times p$ (sendo $p < n$);
- $\beta$ é um vetor com $p \times 1$ parâmetros desconhecidos;
- $\epsilon$ é um vetor $n \times 1$ de resíduos aleatórios com média 0.

Utilizando o método de mínimos quadrados, obtemos o sistema de equações normais, dado na forma matricial por

$$X'X\widehat{\beta} = X'y. \tag{3.52}$$

onde a matrix de dados:

$$X' = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ x_{21} & x_{22} & x_{23} & \dots & x_{2n} \\ x_{31} & x_{32} & x_{33} & \dots & x_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ x_{p1} & x_{p2} & x_{p3} & \dots & x_{pn} \end{bmatrix}$$

a variável dependente é:

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}$$

O vetor de parâmetros é $\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \dots \\ \beta_p \end{bmatrix}$

$$X'X = \begin{bmatrix} n & \sum x_2 & \sum x_3 & \dots & \sum x_p \\ \sum x_2 & \sum x_2^2 & \sum x_2x_3 & \dots & \sum x_2x_p \\ \sum x_3 & \sum x_2x_3 & \sum x_3^2 & \dots & \sum x_3x_p \\ \dots & \dots & \dots & \dots & \dots \\ \sum x_p & \sum x_px_2 & \sum x_px_3 & \dots & \sum x_p^2 \end{bmatrix}$$

Se $X'X$ for uma matriz não singular, a solução do sistema será

$$\widehat{\beta} = (X'X)^{-1}X'y. \tag{3.53}$$

onde:

$$X'y = \begin{bmatrix} \sum y \\ \sum x_2y \\ \dots \\ \sum x_py \end{bmatrix}$$

Os elementos do vetor $\widehat{\beta}$ são as estimativas de mínimos quadrados dos parâmetros associados às variáveis explicativas. Tais estimativas medem o efeito linear de cada variável de $X$ sobre $y$, após terem sido descontadas de ambas as influências lineares de todas as outras variáveis explicativas incluídas no modelo.

### 3.18.1 Matriz de dispersão

A matrix de dispersão $D$, é dada pela seguinte equação:

$$D = [\hat{\beta} - E(\hat{\beta})][\hat{\beta} - E(\hat{\beta})]' = (X'X)^{-1}\sigma^2 \tag{3.54}$$

### 3.18.2 ANOVA

| Fontes de variação | Gl | SQ | QM | F | p-valor |
|---|---|---|---|---|---|
| Da Regressão | $p-1$ | $\hat{\beta}'X'y - C$ | $\frac{\hat{\beta}'X'y - C}{p-1}$ | $\frac{QMReg}{QMRes}$ | |
| Do Resíduo | $n-p$ | $y'y - \hat{\beta}'X'y$ | $\frac{y'y - \hat{\beta}'X'y}{n-p}$ | | |
| Total | $n-1$ | $y'y - C$ | | | |

$QMReg$: significa quadrado médio da regressão.
$QMRes$: significa quadrado médio dos resíduos.
$C = n\bar{y}^2$.

## 3.19 Critério de Seleção de modelos

Estes critérios surgem para penalizar a inclusão de novas variáveis. Assim cada variável adicionada ao modelo, apresenta 1 grau de liberdade a menos. Dentre estes critérios temos:

1. Akaike, o qual usa a seguinte função de penalização

$$AIC = \log(\frac{SQRes}{n}) + \frac{2p}{n} \tag{3.55}$$

onde p, é o número de regressores do modelo, n, é o total de dados, e SQR, é a soma de quadrados do resíduo.

2. Scharz, propos o critério de informação Bayesiana, definido como

$$BIC = \log(\frac{SQRes}{n}) + \frac{p}{n}\log(n) \tag{3.56}$$

O BIC penaliza mais fortemente que o AIC, quando $n > 8$. Uma alteração do BIC, é o BICc, que é fornecida em alguns funções do projeto R.

## 3.20 Exemplo 3

Considere consumo de água de uma residência (extensão do exemplo 2), com as seguintes variáveis: valor pago em reais (valor), consumo de água em $m^3$ (consumo), dias de consumo (diasconsumo), e construção da residência (Construindo), onde o valor 1 é construindo e o valor 0, não construindo. Um modelo completo é descrito a seguir:

### 3.20.1 O modelo

Considere o modelo de regressão na forma matricial:

$$y = X\beta + \epsilon, \tag{3.57}$$

O qual pode ser escrito na forma de modelo de regressão múltipla da seguinte forma:

$$valorpago = \beta_1 + \beta_2 consumo + \beta_3 diasconsumo + \beta_4 Construindo \tag{3.58}$$

### 3.20.2 As hipóteses

Consideremos a hipótese nula $H0$, que o modelo não é adequado ($X\beta = 0$), contra a hipótese alternativa $Ha$, que o modelo é adequado:

1. $H0 : X\beta = 0$
   $Ha : X\beta \neq 0$

2. O nível de significância considerado $\alpha = 0.05$

3. A estatística de prova:

$$F = \frac{QMReg}{QMres} = \frac{(\hat{\beta}' X' y - C)/p - 1}{(y'y - \hat{\beta}' X' y)/(n - p)} \tag{3.59}$$

4. Regra de decisão: Se o $p - valor > 0.05$, há evidencias em favor de $H0$, caso contrario em favor de $Ha$, (ver ANOVA)

### 3.20.3 Estimação dos parâmetros

para estimar os 4 parâmetros do modelo proposto ($\beta_1, \beta_2, \beta_3, \beta_4$), usamos os seguintes comandos no R:

```
> mc=lm(valor~consumo+diasconsumo+Construindo)
> summary(mc)
Call:
lm(formula = valor ~ consumo + diasconsumo + Construindo)
Residuals:
    Min      1Q  Median      3Q     Max
-7.2955 -1.9715 -0.3418  1.8726  9.7028

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.7294    12.2383  -0.141  0.88809
consumo       2.7564     0.1571  17.547  < 2e-16 ***
diasconsumo  -0.2747     0.4080  -0.673  0.50322
Construindo   3.6809     1.2387   2.972  0.00424 **

Residual standard error: 3.52 on 61 degrees of freedom
Multiple R-squared: 0.9087,     Adjusted R-squared: 0.9043
F-statistic: 202.5 on 3 and 61 DF,  p-value: < 2.2e-16
```

Interpretação: a partir das estimativas dos parâmetros do modelo, podemos afirmar, que um proposta em favor de um modelo sem intercepto (valor do $p - valor = 0.888$) é adequado (pois $p - valor > 0.05$), o qual é dado a seguir:

```
> m1=lm(valor~consumo+diasconsumo+Construindo-1)
> summary(m1)

Call:
lm(formula=valor ~ consumo + diasconsumo + Construindo - 1)

Residuals:
    Min      1Q  Median      3Q     Max
-7.3419 -1.9108 -0.3446  1.8175  9.6925

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
consumo      2.75811    0.15536  17.753  < 2e-16 ***
diasconsumo -0.33177    0.05995  -5.534 6.72e-07 ***
Construindo  3.68759    1.22800   3.003  0.00385 **

Residual standard error: 3.492 on 62 degrees of freedom
Multiple R-squared: 0.9851,     Adjusted R-squared: 0.9844
F-statistic:  1367 on 3 and 62 DF,  p-value: < 2.2e-16
```

Interpretação: podemos observar que todas as variáveis independentes que compõem o modelo são significativas individualmente($p < 0.05$), por tanto estatisticamente estas variáveis fazem parte do modelo. O coeficiente de determinação é $R^2_{mult} = 0.985$. Para esclarecer se o modelo é adequado ($X\beta = 0$), realizamos o análise de variância ANOVA.

**O ANOVA**

```
> anova(m1)
Analysis of Variance Table

Response: valor
            Df Sum Sq Mean Sq   F value    Pr(>F)
consumo      1  49175   49175 4033.0722 < 2.2e-16 ***
diasconsumo  1    736     736   60.3480 9.986e-11 ***
Construindo  1    110     110    9.0175  0.003852 **
Residuals   62    756      12
```

## 3.21 Modelo linear parcial

Termos de potência nas variáveis regressoras podem ser adicionados construindo modelos de regressão múltipla, porem a escolha pode tornar-se lenta, por tanto uma extensão semi-paramétrica, considerando o modelo parcial, temos:

$$y = \beta_1 + g(X_1) + \beta_2 X_2 + \cdots + \epsilon \tag{3.60}$$

Onde $g$ é uma função desconhecida que será estimada a partir dos dados, usamos a função **bs()** do pacote **splines**, da seguinte forma

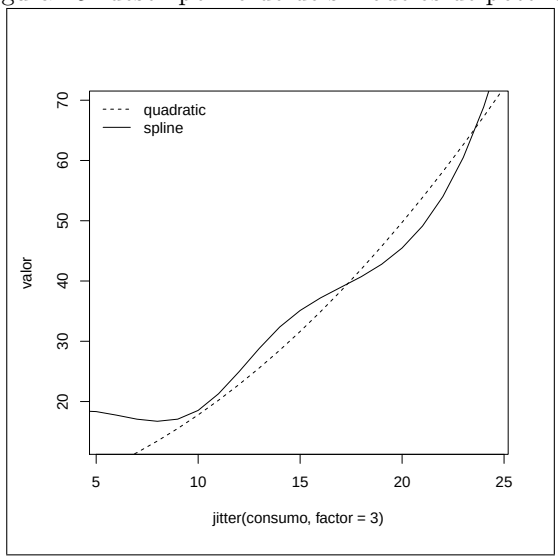
```
> library(splines)
> pm1=lm(valor~bs(consumo, df=5)+diasconsumo+Construindo)
> summary(pm1)

Call:
lm(formula=valor ~ bs(consumo, df=5) + diasconsumo + Construindo)

Residuals:
   Min     1Q Median     3Q    Max
-5.663 -1.436 -0.369  1.668  7.242

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)           18.4351     9.6864   1.903   0.0621 .
bs(consumo, df = 5)1  -0.9266     3.4455  -0.269   0.7890
bs(consumo, df = 5)2  -3.1656     2.5040  -1.264   0.2113
bs(consumo, df = 5)3  27.3786     3.5786   7.651 2.62e-10 ***
bs(consumo, df = 5)4  19.6916     4.5197   4.357 5.57e-05 ***
bs(consumo, df = 5)5  51.1325     3.4760  14.710  < 2e-16 ***
diasconsumo           -0.1033     0.3217  -0.321   0.7493
Construindo            2.5003     0.9987   2.503   0.0152 *

Residual standard error: 2.738 on 57 degrees of freedom
Multiple R-squared: 0.9484,     Adjusted R-squared: 0.942
```

```
F-statistic: 149.6 on 7 and 57 DF,  p-value: < 2.2e-16
```

A sugestão de usar graus de liberdade ($df = 5$), e dada no critério de seleção de modelo (AIC), o padrão é $df = 3$, para confirmar usamos a seguinte função:

```
ps_m1 <- lapply(3:9, function(i)
lm(valor ~ bs(consumo, df = i) + disconsumo + Construindo))
structure(sapply(ps_m1, AIC, k = log(nrow(ca))),.Names = 3:9)
      3        4        5        6        7        8        9
356.0836 344.9000 344.4504 346.0979 350.3472 352.5446 353.3633
```

Figura 15. desempenho de dois modelos de potência




Houve uma mudança nas estimativas nas variáveis dias de consumo e construção. Para realizar o gráfico acima, usamos os seguintes comandos:

```
m2=lm(formula = valor ~ consumo + I(consumo^2))
pm <- data.frame(consumo= 0:40, diasconsumo=with(ca,
mean(diasconsumo[Construindo == 1])),Construindo = 1)
pm$yhat1 <- predict(m2, newdata = pm)
pm$yhat2 <- predict(pm1, newdata = pm)
plot(valor ~ jitter(consumo, factor = 3), pch = 30,
col = rgb(0.5, 0.5, 0.5, alpha = 0.02), data = ca)
lines(yhat1 ~ consumo, data = pm, lty = 2)
lines(yhat2 ~ consumo, data = pm)
legend("topleft", c("quadratic", "spline"), lty =
c(2,1),bty = "n")
```

## 3.22 Seleção de regressores

Em alguns casos é necessário se decidir quais variáveis incluir no modelo de regressão. Há varias formas de contornar esta situação:

1. Use o teste F, começando com modelos com vários regressores e teste exclusões de variáveis ou grupo de variáveis, pode usar a regressão **stepwise**, da seguinte forma

```
> step(m1)
Start:  AIC=165.48
valor ~ consumo + diasconsumo + Construindo - 1

              Df Sum of Sq    RSS    AIC
<none>                     756.0 165.48
- Construindo  1     110.0  865.9 172.31
- diasconsumo  1     373.4 1129.4 189.58
- consumo      1    3842.8 4598.8 280.85

Call:
lm(formula = valor ~ consumo + diasconsumo + Construindo - 1)

Coefficients:
    consumo  diasconsumo  Construindo
     2.7581      -0.3318       3.6876
```

   Podemos observar que no modelo completo o AIC é 165.48 e a medida que excluímos uma variável regressora temos um acréscimo no valor do critério de informação Akaike.

2. Uso do coeficiente de determinação, $R^2$, este critério tem suas limitações, este coeficiente é não decrescente, isto significa que a inclusão de uma variável não importante, aumenta o valor do $R^2$, da mesma forma que se a inclusão da variável regressora seja importante.

3. Uso do R quadrado ajustado, pois pode aumentar ou diminuir quando incluímos novas variáveis, este valor é calculado da seguinte forma:

$$\bar{R}^2 = 1 - \frac{SQRes/n-p}{SQT/n-1}, \tag{3.61}$$

   Tanto o $R^2$ como $\bar{R}^2$, são disponibilizados pela função **summary()**. Podemos afirmar que os modelos $m1$ e $m3$, apresentam melhor performance por este critério. Podemos usar o critério de seleção de modelos descrito nas seções anteriores, da seguinte forma

```
> AIC(md,m1,m2,m3,m4)
   df      AIC
md  3 359.2519
m1  4 351.9462
m2  4 348.9891
m3  3 347.4158
m4  6 329.2190
> BIC(md,m1,m2,m3,m4)
   df      BIC
md  3 365.7751
m1  4 360.6437
m2  4 357.6867
m3  3 353.9390
m4  6 342.2653
```

Por este critério podemos observar que o modelo polinomial de ordem 4, apresenta melhor performance para o melhor ajuste do adequado.

4. Usar o teste que compara modelos não encaixados, por exemplo o modelo $md$ esta encaixado em $m1, m2, m3$, e $m4$, porem $m1$ não é encaixado com $m3$ e $m4$, no exemplo temos:

```
> encomptest(m1,m3)
Encompassing test

Model 1: valor ~ consumo + diasconsumo + Construindo - 1
Model 2: valor ~ consumo + I(consumo^2) - 1
Model E: valor ~ consumo + diasconsumo + Construindo +
I(consumo^2) - 1
          Res.Df Df       F    Pr(>F)
M1 vs. ME     61 -1 13.6058 0.0004822 ***
M2 vs. ME     61 -2  5.3787 0.0070574 **
```

Outro teste, para comparar modelos não encaixados pode ser usando a seguinte função

```
coxtest(formula1, formula2, data = list())
```

## 3.23 Especificação do modelo

Para superar que o modelo esta bem especificado, como linearidade do modelo, inclusão de regressores importantes, e outras situações, usamos testes com o de Ramsey.

### 3.23.1 Teste de especificação Reset de Ramsey

As hipóteses do modelo são: $H0$: O modelo esta corretamente especificado, e $Ha$: O modelo não esta corretamente especificado.

```
> resettest(m1)

        RESET test
data:  m1
RESET = 7.923, df1 = 2, df2 = 60, p-value = 0.0008842
> resettest(m3)
        RESET test
data:  m3
RESET = 0.0436, df1 = 2, df2 = 61, p-value = 0.9574
```

Não há evidencias para aceitar o modelo completo (m1), já o modelo quadrático (m3), esta bem especificado. Testes como normalidade dos resíduos podem ser calculados usando a função:

```
> jarque.bera.test(residuals(m1))
```

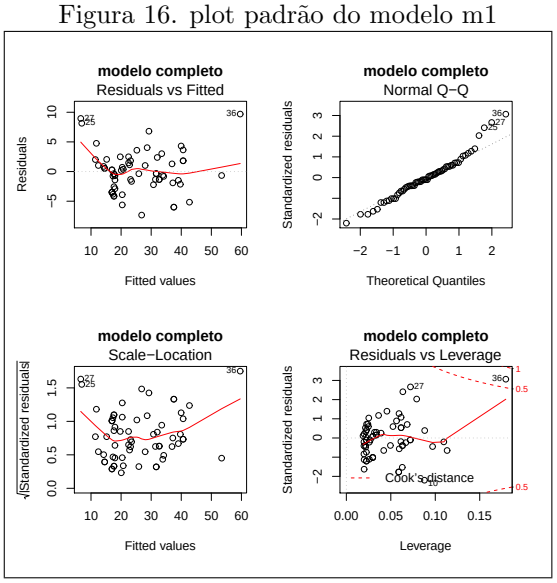
Não há evidencias da normalidade dos erros.

## 3.24 Gráficos

Um gráfico do modelo m1 correspondente é:

```
> par(mfrow=c(2,2))
> plot(m1, main="modelo completo")
```

Figura 16. plot padrão do modelo m1



Da figura acima podemos observar o gráfico de dispersão entre os resíduos e os valores estimados, percebendo certa homogeneidade dos resíduos, no gráfico seguinte observamos a normalidade dos resíduos, com leve afastamento das observações 25,27,36, continuando com o seguinte gráfico observamos que a diferença com o primeiro gráfico é que no lugar dos resíduos, é visualizado a raiz dos resíduos com os valores estimados, finalmente no último gráfico apresenta os pontos de alavanca dos resíduos utilizando a distancia de Cook (isto sera visto nas seguintes seções).

## 3.25 Teste de mudança estrutural

É possível que alguns parâmetros não sejam constantes ao longo de todas as observações, em dados de séries temporais, basicamente é usado o teste do arco iris em que separamos as primeiras $n_1$ observações, este é o ponto de separação onde houve mudança estrutural, podemos usar a função **sctest()** do pacote **strucchange**. O pacote **gap** disponibiliza a função **chow.test()**, como método alternativo de cálculo. Considere os dados de consumo de água, onde $n_1 = 30$. A hipóteses nula (H0): Acima dos primeiros 30 meses de consumo não há mudança estrutural

```
> sctest(valor~consumo, type="Chow", point=30)

        Chow test

data:  valor ~ consumo
F = 16.995, p-value = 1.36e-06


> sctest(valor~consumo+diasconsumo+Construindo,
type="Chow", point=30)

        Chow test

data:  valor ~ consumo + diasconsumo + Construindo
F = 6.459, p-value = 0.0002335
```

Observamos que não há evidencias para aceitar $H0$, em favor da mudança estrutural. No caso que não estamos interessados de usar o ponto de mudança use a função **cumsum()**, do pacote **strucchange**

## 3.26 Teste de outliers

O objetivo do teste é determinar se uma dada observação é um outlier (dado que não corresponde a média das observações). As hipóteses são

$H0$: A n-ésima observação não corresponde a média das observações
$Ha$: caso contrario.
Use a função **outlierTest()** do pacote **car**, No exemplo precedente temos:

```
> outlierTest(m1)

No Studentized residuals with Bonferonni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferonni p
36 3.299283          0.0016213      0.10538
```

Podemos afirmar que há evidencias para aceitar que o mês 36 não corresponde a média das observações, por tanto aquele mês é um outlier.

## 3.27 Detecção de observações atípicas

O objetivo é detetar observações que apresentem padrão atípico. Observações influentes são aquelas que apresentam uma contribuição relativamente grande para o ajuste, esta influencia pode ser por dois fatores, primeiro por padrão atípico de regressores ou segundo por erros demasiado grandes.

1. Alavancagem, observações cujos regressores apresentam padrão atípico são chamados de pontos de alavanca ou observações de alta alavancagem. Para medir os graus de alavancagem de diferentes observações a medida padrão é
$$h_t = X_t'(X'X)^{-1}X_t' \tag{3.62}$$
definida como medida de alavancagem da t-ésima observação, e $X_t$ é um vetor de $p \times 1$ seja
$$H = X(X'X)^{-1}X' \tag{3.63}$$
Multiplicando $H$ por $y$, temos:
$$Hy = X(X'X)^{-1}X'y = X\hat{\beta} = \hat{y} \tag{3.64}$$
onde $h_1, h_2, \cdots, h_t$ são elementos diagonais de $H$. No caso de uma regressão linear simples, temos que:
$$h_t = \frac{1}{n} + \frac{(x_t - \bar{x})^2}{\sum(x_t - \bar{x})^2} \tag{3.65}$$
Onde $\bar{x} = \sum x_t/n$, $h_t$ aumenta a medida que $x_t$ se afasta de $\bar{x}$. Inicialmente observações com alavancagem 2 ou 3 vezes maior que a média são pontos de alavanca ou se $h_t > 2p/n$. No R use a função **hatvalues()**, onde $\mathrm{V}(\hat{\epsilon}_i|X) = \sigma^2(1 - h_{ii})$.

2. Resíduos padronizados, observações cujos valores da variável dependente são atípicos são chamados de outliers, o problema desta medida é que no

caso de erros grandes, os resíduos correspondentes nem sempre são uma boa indicação dos erros verdadeiros. O valor é dado por

$$r_i = \frac{\hat{\epsilon}}{\hat{\sigma}\sqrt{(1-h_{ii})}} \tag{3.66}$$

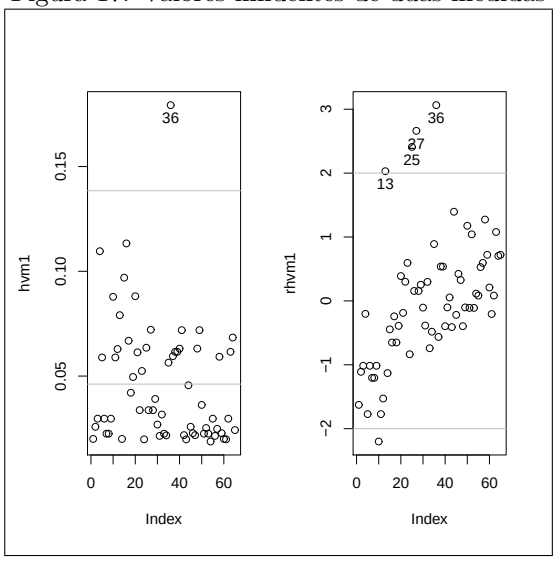
no R, usamos a função **rstandard()**

3. Distancia de Cook, pode ser calculado a partir dos resíduos padronizados e os valores influentes, da seguinte forma

$$C_t = \frac{r_t^2 h_t}{p(1-h_t)} \tag{3.67}$$

regra de bolso, $C_t > \frac{4}{n-p}$

Figura 17. Valores influentes de duas medidas




Considere o modelo m1, a identificação dos valores influentes usando o R:

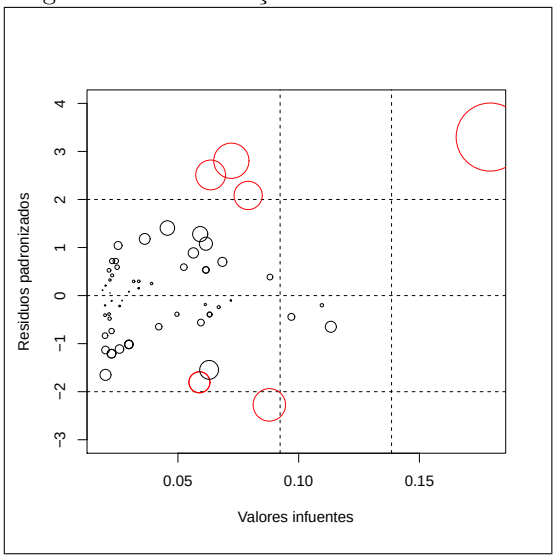
```
hvm1=hatvalues(m1)
hvm1
        1             2            65
0.01999192    0.02578687   -0.02422222
plot(hvm1); abline(h=4/62, col=2); identify(hvm1)
rhvm1=rstandard(m1)
par(mfrow=c(1,2))
plot(hvm1)
abline(h=c(1,3)*mean(hvm1),col=8)
iden=which(hvm1>3*mean(hvm1))
```

```
 text(iden,hvm1[iden],rownames(ca)[iden],pos=1,xpd=TRUE)
 plot(rhvm1)
 abline(h=c(-2,2),col=8)
 iden=which(rhvm1>2)
 text(iden,rhvm1[iden],rownames(ca)[iden],pos=1,xpd=TRUE)
```

Um agoritmo para identificar gráficamente estes valores influentes é dado por:

```
 vinfluentes=function(modelo, scale=10, col=c(1,2),
 labels=names(rst),...){
 vi=hatvalues(modelo) rst=rstudent(modelo)
 dc=sqrt(as.vector(cookd(modelo)))
 escala=scale/max(dc)
 p=length(coef(modelo)) n=length(rst)
 co=sqrt(4/(n-p))
 plot(vi,rst,xlab="Valores infuentes",
 ylab="Residuos padronizados", type="n",
 ylim=c(-3,4), ...)
 abline(v=c(2,3)*p/n, lty=2)
 abline(h=c(-2,0,2), lty=2)
 for(i in 1:n)
 points(vi[i],rst[i], cex=escala*dc[i],
 col=if(dc[i]>co) col[2] else col[1])
 if(labels[1]!=FALSE) identify(vi,rst,labels)}
> library(car)
> vinfluentes(m1) # identifique com o mause
[1] 10 13 25 27 36
> dc=cooks.distance(m1)  #distancia de cook
```

Figura 18. Identificação de valores influentes



## 3.28 Consistência

**Definição 2** *Convergência em probabilidade. Seja $Z_n, n \in N$ uma sequencia de variáveis aleatórias e seja $c \in \Re$, dizemos que $Z_n$ converge em probabilidade para c, denotado por $Z_n \xrightarrow{p} c$ ou $plim(Z_n) = c, \qquad \forall \epsilon > 0.$*

pode ser escrito como $\lim_{n\to\infty} P(|Z_n - c|] < \epsilon) = 1$

**Definição 3** *Consistência. Um estimador $\hat{\theta}$ de um parâmetro $\theta \in \Theta$, sendo $\Theta$ um espaço paramétrico, é dito ser consistente se $\hat{\theta} \xrightarrow{p} \theta, \ \forall \theta \in \Theta$*

Para identificar a relação entre consistência e não viés, consideremos alguns estimadores.

1. $\hat{\theta}_1 = \frac{1}{n+1} \sum Z_n$
2. $\hat{\theta}_2 = \frac{0.5}{n} \sum Z_n$
3. $\hat{\theta}_3 = 0.01 Z_1 + \frac{0.99}{n-1} \sum Z_n$

Para o primeiro estimador temos que é viesado:

$$E(\hat{\theta}_1) = E(\frac{1}{n+1} \sum Z_n) = \frac{1}{n+1} \sum E(Z_n) = \frac{1}{n+1} n\theta \neq \theta$$

Para verificar a consistência fazemos:

1. $\lim_{n\to\infty} E(\hat{\theta}_1) = \lim_{n\to\infty} \frac{n}{n+1}\theta = \theta$
2. $\lim_{n\to\infty} V(\hat{\theta}_1) = \lim_{n\to\infty} \frac{1}{(n+1)^2} V(\sum Z_n) = \lim_{n\to\infty} \frac{n}{(n+1)^2}\sigma^2 = 0$. Por tanto é consistente.

Para o segundo estimador temos que é viesado e inconsistente (prove).
Para o terceiro estimador temos que é não viesado e inconsistente (prove).

## 3.29 Iterações

Quando os regressores apresentam dependência existe interação, podemos identificar o grau de interação das variáveis, caso não há interação dizemos que as variáveis são independentes, em nosso exemplo de consumo de água, é natural que quando estamos construindo há mais consumo de água, ou a mais dias de consumo, mais consumo. Para exemplificar:

```
> m_int=lm(valor~consumo+Construindo*consumo+diasconsumo*consumo)
> summary(m_int)

Call:
lm(formula = valor ~ consumo + Construindo * consumo + diasconsumo
*consumo)

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)        -8.84548   43.60920  -0.203   0.8400
consumo             3.51575    3.43121   1.025   0.3097
Construindo        -8.55260    5.40296  -1.583   0.1188
diasconsumo         0.04108    1.42980   0.029   0.9772
consumo:Construindo 0.82826    0.35774   2.315   0.0241 *
consumo:diasconsumo -0.03210    0.11216  -0.286   0.7757

Residual standard error: 3.415 on 59 degrees of freedom
Multiple R-squared: 0.9169,     Adjusted R-squared: 0.9099
F-statistic: 130.3 on 5 and 59 DF,  p-value: < 2.2e-16
```

Podemos interpretar que há interação significativa ($p < 0.05$) entre consumo e construção, já no caso de consumo e dias de consumo não há interação significativa ($p > 0.05$).

## 3.30 Regressão separada

Em alguns casos estamos interessados em analisar regressões separadas por vários níveis, no exemplo do consumo de água estamos interessados em determinar qual o desempenho das estimativas quando estamos construindo e quando não estamos construindo, exemplificando

```
> m_cons=lm(valor~Construindo/(consumo+diasconsumo))
> msc=matrix(coef(m_cons), nrow=2)
> m1=lm(valor~consumo+diasconsumo+Construindo-1)
> rownames(msc)=levels(Construindo)
> colnames(msc)=names(coef(m1))[1:2]
> msc
       consumo diasconsumo
[1,] 21.152708   3.3981281
[2,] -9.665703  -0.9184493
```

Podemos confirmar que o valor de consumo e dias de consumo é diferente para os dois níveis de construção. Para visualizar um gráfico de regressão por separado podemos usar a função

```
> coplot(valor~Construindo|consumo+diasconsumo,panel = panel.smooth)
```

Uma outra alternativa, é separar o conjunto de dados em duas matrizes de dados, a ideia é comparar duas ou mais regressão num mesmo gráfico alem de verificar a variação da tangente de inclinaão em cada estrato. Exemplo verificar as regressões de valor em função do consumo, quando estamos ou não construindo.

```
plot(consumo,valor,pch=as.numeric(Construindo))
legend("topleft",legend=c("não construindo","construindo"),pch=0:1)
noconstruindo=ca[Construindo==0,]
construindo=ca[Construindo==1,]
mnc=lm(valor~consumo,data=noconstruindo)
mc=lm(valor~consumo,data=construindo)
summary(mnc)
summary(mnc)
abline(mnc,lty=2)
abline(mc,lty=3)
```

## 3.31 Regressão quantilica

Em algumas situações, as estimativas dos parâmetros apresentam mudanças bruscas ou repentinas, a cada quantidade das observações processada. A função **rq()** do pacote **quantreg**, proporciona resultados, para monitorar as mudanças das estimativas a cada percentual de observações de interesse, um exemplo é dado a seguir:

```
> library(quantreg)
> mrq1=rq(m1, tau=seq(0.2,0.8,by=0.15),data=ca)
> summary(mrq1)

Call: rq(formula = m1, tau = seq(0.2, 0.8, by = 0.15), data = ca)

tau: [1] 0.2
Coefficients:
            coefficients lower bd upper bd
consumo      3.01000      2.45247  3.13831
diasconsumo -0.51969     -0.59977 -0.35503
Construindo  0.00000     -1.13537  4.59762

Call: rq(formula = m1, tau = seq(0.2, 0.8, by = 0.15), data = ca)

tau: [1] 0.35
Coefficients:
            coefficients lower bd upper bd
consumo      2.66710      2.40001  2.92237
diasconsumo -0.32681     -0.52897 -0.22469
Construindo  3.47798     -0.67739  4.78711
```

```
Call: rq(formula = m1, tau = seq(0.2, 0.8, by = 0.15), data = ca)

tau: [1] 0.5
Coefficients:
            coefficients lower bd upper bd
consumo      2.61371      2.52489  2.97180
diasconsumo -0.27772     -0.40779 -0.23535
Construindo  3.63515      1.71949  6.55572

Call: rq(formula = m1, tau = seq(0.2, 0.8, by = 0.15), data = ca)

tau: [1] 0.65
Coefficients:
            coefficients lower bd upper bd
consumo      2.72463      2.43338  3.13545
diasconsumo -0.29537     -0.40229 -0.14411
Construindo  4.98220      1.08493  7.49433

Call: rq(formula = m1, tau = seq(0.2, 0.8, by = 0.15), data = ca)

tau: [1] 0.8
Coefficients:
            coefficients lower bd upper bd
consumo      2.65200      2.10188  3.22867
diasconsumo -0.22847     -0.38592 -0.00817
Construindo  6.04000      2.83924  9.45696
```

Podemos observar que as estimativas dos parâmetros, teve algumas mudanças como é o caso da Construção, a medida que atingimos o total das observações temos um crescimento maior no valor do consumo de água, as outras duas variáveis não teve mudanças repentinas expressivas.

## 3.32 Derivação dos EMQO na regressão Múltipla

A ideia é estender as estimativa para $p+1$ parâmetros, isto pode ser proposto da seguinte maneira:

$$\epsilon = Y - \hat{\beta}_1 - \hat{\beta}_2 X_1 - \hat{\beta}_3 X_2 - \cdots - \hat{\beta}_{p+1} X_p = Y - \hat{Y} \tag{3.68}$$

consideraremos por tanto minimizar a soma dos quadrados dos resíduos, da seguinte forma:

$$Z = \sum_{i=1}^{n} \hat{\epsilon}^2 = \sum_{i=1}^{n} (Y - \hat{\beta}_1 - \hat{\beta}_2 X_1 - \hat{\beta}_3 X_2 - \cdots - \hat{\beta}_{p+1} X_p)^2 \tag{3.69}$$

Para determinar as estimativas dos parâmetros da regressão múltipla, temos:

$$\frac{\partial Z}{\partial \hat{\beta}_1} = 2\sum(Y - \hat{\beta}_1 - \hat{\beta}_2 X_1 - \hat{\beta}_3 X_2 - \cdots - \hat{\beta}_{p+1} X_p)(-1) = 0 \tag{3.70}$$

$$\sum Y = \sum \hat{\beta}_1 + \sum \hat{\beta}_2 X_1 + \sum \hat{\beta}_3 X_2 + \cdots + \sum \hat{\beta}_{p+1} X_p \tag{3.71}$$

$$\sum Y = n\hat{\beta}_1 + \hat{\beta}_2 \sum X_1 + \hat{\beta}_3 \sum X_2 + \cdots + \hat{\beta}_{p+1} \sum X_p \tag{3.72}$$

$$\frac{\partial Z}{\partial \hat{\beta}_2} = 2\sum(Y - \hat{\beta}_1 - \hat{\beta}_2 X_1 - \hat{\beta}_3 X_2 - \cdots - \hat{\beta}_{p+1} X_p)(-X_1) = 0 \tag{3.73}$$

$$\sum X_1 Y = \sum \hat{\beta}_1 X_1 + \sum \hat{\beta}_2 X_1^2 + \sum \hat{\beta}_3 X_1 X_2 + \cdots + \sum \hat{\beta}_{p+1} X_1 X_p \tag{3.74}$$

$$\sum X_1 Y = \hat{\beta}_1 \sum X_1 + \hat{\beta}_2 \sum X_1^2 + \hat{\beta}_2 \sum X_1 X_2 + \cdots + \sum \hat{\beta}_{p+1} X_p \tag{3.75}$$

$$\frac{\partial Z}{\partial \hat{\beta}_{j+1}} = 2\sum(Y - \hat{\beta}_1 - \hat{\beta}_2 X_1 - \cdots - \hat{\beta}_{j+1} X_j - \cdots)(-X_j) = 0 \tag{3.76}$$

$$\sum X_j Y = \sum \hat{\beta}_1 X_j + \sum \hat{\beta}_2 X_1 X_j + \cdots + \sum \hat{\beta}_{j+1} X_j^2 + \cdots \tag{3.77}$$

$$\sum X_j Y = \hat{\beta}_1 \sum X_j + \hat{\beta}_2 \sum X_1 X_j + \hat{\beta}_2 \sum X_2 X_j + \cdots + \hat{\beta}_{p+1} \sum X_j Xp \tag{3.78}$$

## 3.33 Violação de um pressuposto básico

Nos capítulos prévios temos tratado os modelos de regressão sob a perspectiva de não violar algum dos pressupostos básicos descritos anteriormente. No entanto, em muitas aplicações práticas que envolvem modelagem de regressão, onde o comportamento de uma variável de interesse é explicado a partir de sua relação com variáveis auxiliares assumindo-se, em geral, que esta relação seja linear. No caso de violar a suposição do modelo ser verdadeiro (adequado) isto pode ser explicado por não incluir em nosso modelo variáveis que são importantes ou por que incluímos sem poder explicativo, sendo assim a omissão de regressores relevantes causa viés, e a inclusão de regressores irrelevantes causa ineficiência.

Uma suposição constantemente feita é a de homoscedasticidade, ou seja, assume-se que todos os erros do modelo possuem variâncias idênticas. Esta suposição, contudo, é violada em muitas situações, em especial, quando o interesse reside na modelagem de dados de corte transversal. Neste caso, é muito comum que os dados apresentem heterocedasticidade, ou seja, variâncias condicionais não-constantes, ver Cribari-Neto [7], onde

$$y_i = \beta_1 + \beta_2 x_i + \sigma_i \epsilon_i \qquad i = 1, 2, \ldots n \tag{3.79}$$

a variância $V(\epsilon) = \sigma_i^2$ não é constante. Dessa forma, a variância da perturbação depende, por exemplo, da variável independente $X$.

Vale destacar que apenas sobre normalidade da estrutura de erros, o estimador de mínimos quadrados ordinários dos parâmetros coincide com o estimador de máxima verossimilhança. Além disso, sob normalidade, o estimador de MQO é o melhor estimador na classe dos estimadores não viesados, ou seja, o estimador de MQO é eficiente, Ferreira [10]. Com estas hipóteses do modelo clássico de regressão linear, os estimadores MQO possuem certas características estatísticas desejáveis, sintetizadas nas propriedades de MELNT (Melhor Estimativa Linear não-Tendenciosa). Contudo, na prática, como sabemos se a propriedade MELNT é valida? Por exemplo, como podemos verificar se os EMQO são não viesados? Isto é feito através dos experimentos MONTE CARLO, que são basicamente experimentos de simulação realizados com auxilio de um computador. O modelo de regressão linear geral, descrito é da forma $y = X\beta + \epsilon$; onde $y$ é um vetor $(n \times 1)$ de observações da variável dependente, $X$ é uma matriz fixa de posto completo de dimensão $(n \times p)$, onde $p < n$, contendo observações sobre as variáveis explicativas. $\beta = (\beta_1, \beta_2, \cdots, \beta_p)$ é um vetor $(p \times 1)$ de parâmetros desconhecidos e $\epsilon$ é um vetor $(n \times 1)$ de distúrbios aleatórios (erros) com média 0 e matriz de covariância $\Omega = \text{diag}(\sigma_1^2, \sigma_2^2, \cdots, \sigma_n^2)$. Quando os erros são homoscedásticos, então $\sigma_i^2 = \sigma^2 > 0$, ou seja, $\Omega = \sigma^2 I_n$, onde $I_n$ é a matriz identidade de ordem $n$. O estimador de mínimos quadrados ordinários de $\beta$ é dado por $\widehat{\beta} = (X'X)^{-1}X'y$, cuja média é $\beta$ (isto é, ele é não-viesado) e cuja variância é dada por

$$\Psi = (X'X)^{-1}(X'\Omega X)(X'X)^{-1}. \tag{3.80}$$

Quando todos os erros possuem a mesma variância, ou seja, $\Omega = \sigma^2 I_n$, esta expressão é simplificada para $\sigma^2(X'X)^{-1}$, podendo ser facilmente estimada como $\widehat{\sigma}^2(X'X)^{-1}$, onde $\widehat{\sigma}^{\,2} = \widehat{\epsilon}'\widehat{\epsilon}/(n-p)$. Aqui,

$$\widehat{\epsilon} = (I_n - X(X'X)^{-1}X')y = My$$

onde $\widehat{\epsilon}$ representa o vetor $(n \times 1)$ de resíduos de mínimos quadrados.

## 3.34 Aleatoriedade

Segundo Zafon [36], o termo aleatoriedade, baseado em conceitos estatísticos, significa algum evento que acontece com qualquer distribuição de probabilidade, onde normalmente se associa uma falta de influência ou correlação, a menos que especificado de outra forma. Em termos de computação, aleatoriedade refere-se à geração ou uso de um conjunto de sequências de números aleatórios dentro de algum conjunto finito.

Mas, o que é um número aleatório? Na verdade não existe um número aleatório, mas sim um conjunto sequencial de números independentes que possuem uma distribuição de probabilidade específica de ocorrer. Isso significa que cada número na sequência possui uma probabilidade de ocorrer independente do anterior e este número é obtido meramente ao acaso. Computacionalmente falando,

existem técnicas para gerar uma sequência de números pseudo-aleatórios, pois é impossível gerar, apenas através de algoritmos computacionais, uma sequência de números realmente aleatória.

## 3.35 Gerador de Números Aleatórios

A disponibilidade de números aleatórios é extremamente útil em várias situações como, por exemplo, em amostragem de uma população ou em tomada de decisões. Algumas das propriedades que um bom gerador de números pseudo-aleatórios deve possuir:

a) Os números gerados devem seguir uma distribuição uniforme;

b) Os números devem ser estatisticamente independentes entre si;

c) A sequência não deve se repetir nunca (teoricamente isso é impossível mas, na prática, um período suficientemente grande é o bastante);

d) Os algoritmos de geração desses números devem ser rápidos de modo que os recursos computacionais possam ser concentrados nas simulações.

O processo de geração de números aleatórios, para ser eficiente, deve atender a alguns requisitos, como velocidade e conformidade. Infelizmente, esses requisitos, são contraditórios, pois geradores muito rápidos em geral não fazem operações mais complexas, o que permitiria melhor conformidade segundo Goldreich [13]. Um outro trabalho de construção de geradores independentes de números aleatórios é dado por Zafon e Manacero [36]. Nesse trabalho foi adotado um gerador congruencial linear como base do gerador uniforme. Uma definição formal de um Gerador de Números Aleatórios (GNA) pode ser encontrada em L'Ecuyer [21]

**Definição.** Um GNA é uma estrutura $G = (S, s_0, f, U, g)$, onde
a) $S$ é um conjunto finito de estados.
b) $s_0 \in S$ é o estado inicial do gerador, também chamado de semente.
c) $f$ é a função de transição entre os estados.
d) $U$ é um conjunto finito de estados de saída do gerador.
e) $g$ é a função de saída do gerador. O GNA inicia na semente $s_0$ e $u_0 = g(s_0)$. A partir deste ponto, para todo $i = 1, 2, \ldots n$, tem-se $s_i = f(s_{i-1})$ e $u_i = g(s_i)$.
f) Dada a mesma semente, será produzida sempre a mesma sequência de números pseudo-aleatórios.
h) Valores sucessivos da seqüência devem ser independentes e uniformemente distribuídos.

Uma outra característica interessante de um GNA é seu período. Dado que $S$ é finito, a sequência de estados é necessariamente periódica. O período é definido como o menor inteiro positivo $p$ tal que $S_{p+n} = S_n$ para todo $n$.

## 3.36 Testes de Normalidade

Dentro da literatura existem diversos testes propostos para avaliar se uma determinada amostra provém de uma população com distribuição normal ou não normal. Geralmente, isto é medido com o chamado poder do teste (conhecido como $1-\beta$, onde $\beta$ é a probabilidade de aceitar a hipóteses nula $H_0$, dado que esta é falsa).

Ferreira [10] apresenta um estudo de análise de sensibilidade dos testes de normalidade de Jarque-Bera e Lilliefors em modelos de regressão linear, violando alguns dos pressupostos básicos como o de não autocorrelação das perturbações, e homoscedasticidade. Para o estudo utiliza 10000 réplicas de Monte Carlo, com tamanhos de amostras de $10, 50, 100, 500$ e 1000 e níveis de significância de $10\%, 5\%$, e $1\%$. O autor concluiu que em modelos de regressão com estrutura de erros não correlacionados e homoscedásticos, os testes de Jarque-Bera e Lilliefors apresentaram comportamento bastante satisfatórios. Além disso, quando há autocorrelação fraca dos erros os desempenhos dos testes não sofrem alterações significativas; já na autocorrelação forte os desempenhos dos testes ficam comprometidos, pois as taxas de rejeição superam os valores nominais dos níveis de significância sugeridos acima. Para o caso da violação do pressuposto de homoscedasticidade (heteroscedasticidade), as taxas de rejeição atingem 100% em muitos casos. Ainda apresenta o poder do teste, concluindo que os testes apresentaram bom poder, destacando-se o teste de Jarque-Bera na maioria dos casos.

### 3.36.1 Teste Jarque-Bera (JB)

Trata-se de um teste assintótico. As hipóteses a serem testadas são:

$H_0$ : o erro $\epsilon_i$ do modelo de regressão linear possui distribuição normal

contra

$H_1$ : o erro $\epsilon_i$ do modelo de regressão linear possui distribuição não normal.

O erro do tipo I dado por $\alpha = \mathrm{P}(\text{Rejeitar } H_0 | H_0 \text{ é verdadeira})$, escolhe-se o nível de significância, que tipicamente é 1%, 5% e 10%. Para a construção do teste de Jarque-Bera utilizaremos as medidas de tendência central:

$$\bar{\epsilon} = \frac{1}{n}\sum_{i=1}^{n}\epsilon_i \tag{3.81}$$

onde $n$ é o número de observações da amostra, os $\epsilon_i$ são os erros não observáveis da amostra, e $\bar{\epsilon}$ é a média amostral, conhecido também como momento centrado em torno da média de ordem 1. Temos,

$$\widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(\epsilon_i - \bar{\epsilon})^2, \tag{3.82}$$

$$DP = \sqrt{\widehat{\sigma}^2}, \tag{3.83}$$

onde $\widehat{\sigma}^2$ é definido como estimação da variância dos erros, determinando a variabilidade dos dados, conhecido também como momento centrado em torno da média de ordem 2. $DP$ é comumente conhecido como desvio padrão. Extensões destes momentos são

$$\widehat{\sigma}^3 = \frac{1}{n}\sum_{i=1}^{n}(\hat{\epsilon}_i - \overline{\epsilon})^3, \tag{3.84}$$

$$A = \widehat{\sigma}^3/(DP)^3, \tag{3.85}$$

onde $\widehat{\sigma}^3$ é conhecido como momento centrado em torno da média de ordem 3. O valor de $A$ mede o grau de assimetria da curva, que pode ser assimétrica negativa ($A > 0$), assimétrica positiva ($A < 0$), ou simétrica ($A = 0$).

Outro momento que é necessário na construção do teste Jarque-Bera é

$$\widehat{\sigma}^4 = \frac{1}{n}\sum_{i=1}^{n}(\hat{\epsilon}_i - \overline{\epsilon})^4 \tag{3.86}$$

$$k = \widehat{\sigma}^4/(DP)^4, \tag{3.87}$$

onde $k$ é denominado de Curtose, podendo ser platicurtose ou achatada se $k < 3$, mesocurtose ou normal se $k = 3$ e lepto curtose se $k > 3$.
A estatística de Jarque-Bera é dada por

$$JB = n\left[\frac{A^2}{6} + \frac{(k-3)^2}{24}\right]. \tag{3.88}$$

### 3.36.2 Teste Lilliefors

O teste de Lilliefors é uma adaptação do Teste de Kolmogorov-Smirnov para testar normalidade. Como antes, queremos estudar erros do tipo I. Dada uma amostra $\hat{\epsilon}_i$, para $i = 1, 2, \ldots, n$, procede-se da seguinte forma:

1. Calcula-se a média amostral

$$\overline{\epsilon} = \frac{1}{n}\sum \hat{\epsilon}_i \tag{3.89}$$

2. Calcula-se a variância amostral

$$s^2 = \frac{1}{n-1}\sum(\hat{\epsilon}_i - \overline{\epsilon})^2 \tag{3.90}$$

3. Transformamos a amostra, calculando

$$Z_i = \frac{\hat{\epsilon}_i - \overline{\epsilon}}{s} \tag{3.91}$$

4. Para cada valor $Z_i$, calcula-se a proporção $\mathcal{L}(Z_i)$ dos valores da amostra que não excedem $Z_i$.
5. Finalmente, determina-se a probabilidade $\mathcal{N}(Z_i)$ de que $Z_i$ tenha sido obtido de uma normal com média 0 e desvio-padrão 1.

O critério de Lilliefors é baseado no número

$$L = \max_i \{|\mathcal{L}(Z_i) - \mathcal{N}(Z_i)|, |\mathcal{L}(Z_i) - \mathcal{N}(Z_{i-1})|\}. \tag{3.92}$$

Os quantis da distribuição dessa estatística podem ser encontrados, por exemplo, em Connover (1999).

### 3.36.3 Simulação de Monte Carlo

Um comentário sobre os experimentos Monte Carlo, utilizando a função de regressão populacional de duas variáveis, é apresentado por [14]:

Os experimentos de Monte Carlo é composto dos seguintes passos:

1. Suponha que no modelo (3.1) os valores reais dos parâmetros sejam $\beta_1 = 20$ e $\beta_2 = 0,6$.
2. Escolha o tamanho da amostra, digamos $n = 25$.
3. Fixe os valores do regressor $X$ para cada observação. Ao todo você terá 25 valores de $X$.
4. Suponha que, de uma tabela de números aleatórios, você escolha 25 valores e os chame de $\epsilon_i$.
5. Como você conhece $\beta_1$, $\beta_2$, $x_i$ e $\epsilon_i$, utilizando (3.2) obterá 25 valores de $y_i$.
6. Utilizando agora os 25 valores de $y_i$ assim gerados, você os regressa nos 25 valores de $X$ escolhidos no passo 3, obtendo os estimadores de mínimos quadrados $\widehat{\beta}_1$ e $\widehat{\beta}_2$.
7. Suponha que você repita este experimento mais 99 vezes, utilizando a cada vez os mesmos valores de $\beta_1$, $\beta_2$ e $X$. Naturalmente, os valores $\epsilon_i$ irão variar de um experimento para outro. Por tanto você tem ao todo 100 experimentos, gerando assim 100 valores de $\widehat{\beta}_1, \widehat{\beta}_2$. (Na prática se realizam bem mais do que 100 desses experimentos.)
8. Tome as médias dessas 100 estimativas e as chame de $\bar{\widehat{\beta}}_1$ e $\bar{\widehat{\beta}}_2$.
9. Se esses valores médios forem quase iguais aos verdadeiros valores $\beta_1$ e $\beta_2$ admitidos por hipóteses no passo 1, o experimento Monte Carlo "demonstra" que os estimadores por mínimos quadrados são realmente não-viesados ou não-tendenciosos.

Estes passos caracterizam a natureza geral dos experimentos de Monte Carlo.

### 3.36.4 Implementação no R

Para exemplificar um número de 1000 simulações (repetições) de experimentos de Monte Carlo de um modelo de regressão linear simples sem violação de um pressuposto básico e proposto a seguir, usamos uma semente inicial. A variável independente x, é gerada a partir de uma amostra aleatória de 1000 elementos com distribuição normal, com média 20 e variância 5. Na saída são apresentados os histogramas e a função acumulada correspondente a cada estimativa dos parâmetros $\beta_1$ e $\beta_2$.

```
mc.sim=function(r=1000){
set.seed(100)
beta1=20
beta2=0.6
x=rnorm(1000,20,5)
sigma2=20
n=length(x)
x=as.matrix(x)
X=cbind(1,x)
linear.predictor=beta1+beta2*x
y.simulated=linear.predictor[,1]+matrix(rnorm(n*r, mean=0,
sd=sqrt(sigma2)),n,r)
estimate=solve(t(X)%*%X)%*%t(X)%*%y.simulated
par(mfrow=c(2,2), pty="s")
hist(estimate[1,], col=8, breaks=12, xlab="estimativa de b1",
main="histograma de b1", ylab="frequencia", xlim=c(15,25))
hist(estimate[2,], col=8, breaks=12, xlab="estimativa de b2",
main="histograma de b2", ylab="frequencia", xlim=c(0.4,0.8))
plot(ecdf(estimate[1,]),main="distribuição de b1")
plot(ecdf(estimate[2,]),main="distribuição de b2")

return(c(mean(estimate[1,]),mean(estimate[2,])))
 }
> mc.sim(1000)

[1] 19.992603  0.600685
```

Observação: podemos verificar o teorema de Gauss-Markov, que a medida que aumentamos o tamanho da amostra as estimativas dos parâmetros não apresentam viés e a sua variância são eficientes. Para amostras pequenas, a variabilidade de $\beta_1$ e $\beta_2$ é grande.

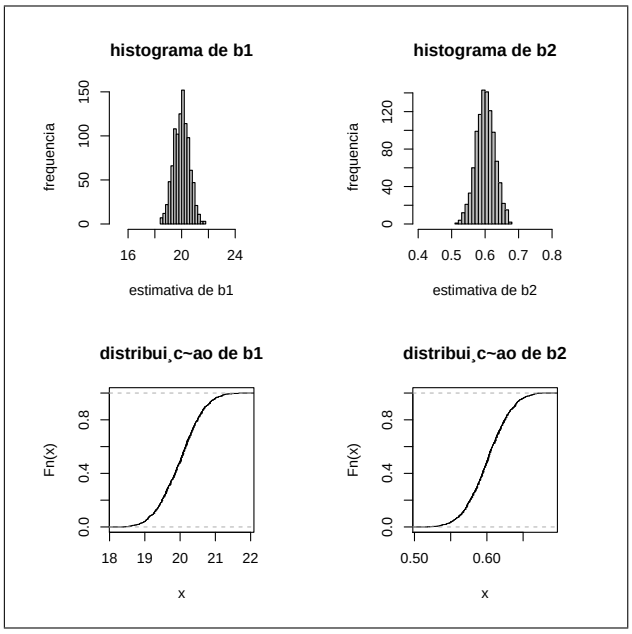



Figura 19. Simulação MC de um modelo de regressão linear simples.

Se assumirmos o fato do modelo ser não homocedástico, podemos modificar a linha 6 da seguinte forma:

```
sigma2=rchisq(length(x1),2*x^2) # com a saída seguinte:
> mc.sim(1000)
[1]  0.6008048 19.9863130
```

Se alem do fato o modelo ser não homocedástico acrescentarmos outra violação do pressuposto básico de não normalidade dos erro (erros uniformes), podemos modificar a linha 11 da seguinte forma:

```
y.simulated=linear.predictor[,1]+matrix(runif(n*r,-10, 10),n,r),
```

A saída é:

```
> mc.sim(1000)
[1] 19.9746838  0.6016005
```

Uma extensão para modelos de regressão múltipla com presença de não normalidade dos erros, assumindo que os erros são $U(-10, 10)$, com presença de heterocedasticidade, assumindo que a variância dos resíduos dependem em termos quadráticos da variável $x_1$. Presença de multicolinearidade é proposto considerando $x2$ ua variável aleatória dependente de $x1$. Para disponibilizar a força da colinearidade entre estas variáveis, podemos adicionar no algoritmo abaixo na linha da função **return()** o comando cor(x1,x2).

```
# Regressão multipla não normal,
heterocedástico e com multicolinearidade

mc.sim=function(r=1000)
{
set.seed(100)
beta1=20
beta2=0.6
beta3=0.8
x1=rnorm(1000,20,5)
x2=x1+rnorm(1000,2,2)
sigma2=rchisq(length(x1),2*x1^2)
n=length(x1)
x1=as.matrix(x1)
x2=as.matrix(x2)
X=cbind(1,x1,x2)
linear.predictor=beta1+beta2*x1+beta3*x2
y.simulated=linear.predictor[,1]+matrix(runif(n*r,-10,10),n,r)
estimate=solve(t(X)%*%X)%*%t(X)%*%y.simulated
par(mfrow=c(3,3), pty="s")
hist(estimate[1,], col="red", breaks=12, xlab="estimativa de b1",
main="histograma de b1", ylab="frequencia", xlim=c(15,25))
hist(estimate[2,], col="blue", breaks=12, xlab="estimativa de b2",
main="histograma de b2", ylab="frequencia", xlim=c(0.4,0.8))
hist(estimate[3,], col="blue", breaks=12, xlab="estimativa de b3",
main="histograma de b3", ylab="frequencia", xlim=c(0.5,1.3))
plot(ecdf(estimate[1,]),main="distribuição de b1")
plot(ecdf(estimate[2,]),main="distribuição de b2")
plot(ecdf(estimate[3,]),main="distribuição de b3")
boxplot(estimate[1,], main="estimativa de b1")
boxplot(estimate[2,], main="estimativa de b2")
boxplot(estimate[3,], main="estimativa de b3")
return(c(mean(estimate[1,]),mean(estimate[2,]),mean(estimate[3,])))
}
mc.sim(1000)
```

Com a seguinte saída,

```
[1] 20.0239072  0.5971375  0.8018691
```

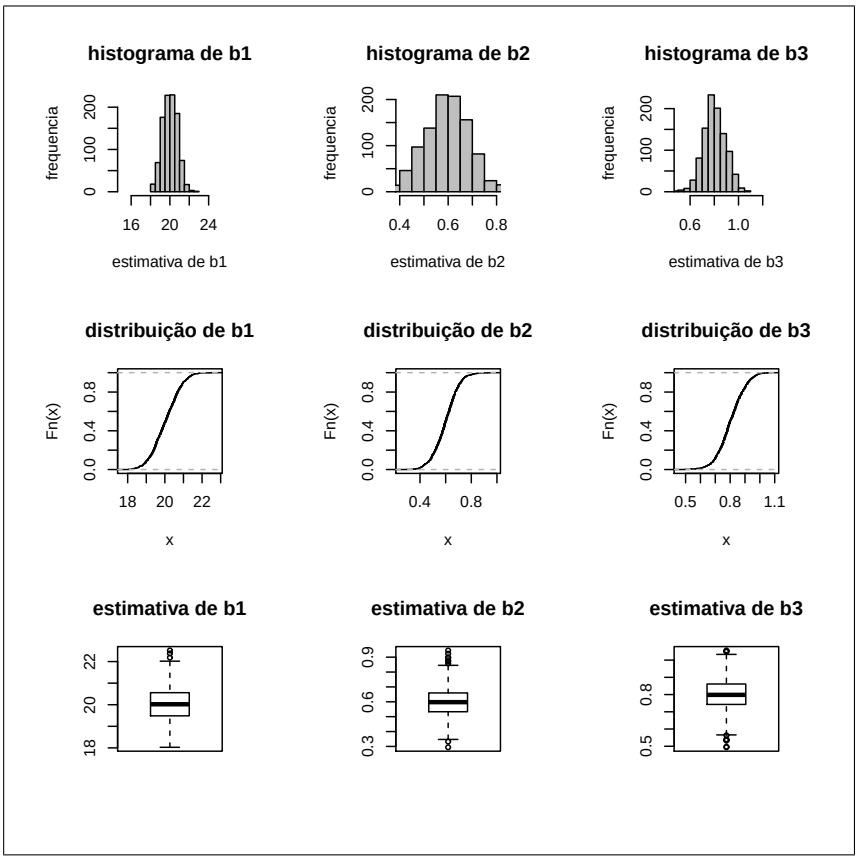



Figura 20. Estimativa dos parâmetros na violação de três pressupostos

## 3.37 Exercícios

1. Qual a diferença de um modelo de regressão populacional e de um modelo de regressão amostral?
2. Que entende por estimativa, parâmetro, resíduo, termo aleatório, variável independente e dependente?
3. Quais são os passos de um teste de hipóteses?
4. Com a definição 3.1 determine $\sigma^2$, pelo método de mínimos quadrados, e pelo método de máxima verosimilhança e calcule o valor esperado de $\sigma^2$, para os dois métodos.
5. Por que é importante especificar um modelo, na teoria econométrica?
6. Exemplifique 2 modelos lineares nos parâmetros, e dois modelos lineares nas variáveis.

7. Considere um conjunto hipotético referente a quantidade de chuva (ch) em $mm^3$, que cai por cada hectare de produção de batatas (pr), onde são colocados dois quantidades de adubo (ad) representados a seguir. Especifique um modelo adequado.

```
ch=c(2,2,1,2,2,3,0.5,6,4,5,6,7,8,8,8,3,3,2,1,1)
ad=c(0.4,0.4,0.4,0.4,0.4,0.8,0.4,0.8,0.4,0.8,0.8,0.8,0.8,
0.8,0.8,0.8,0.8,0.4,0.4,0.4)
pr=c(30,31,22,33,34,41,16,52,37,44,42,40,38,39,38,41,40,37,
23,20)
```

8. No exemplo anterior proponha um modelo lin-log dado por pr=f(log(ch)), quais são suas conclusões?

9. No mesmo exemplo proponha um modelo log-lin dado por log(pr)=f(ch), quais são suas conclusões?

10. Aplique o teste de Jarque Bera para os resíduos e construa o gráfico.

11. Ainda para a produção de batatas, determine o ANOVA, faça os gráficos correspondentes e interprete.

12. Proponha um modelo de regressão múltipla, onde pr=f(ad,ch). Determine e interprete as estimativas dos parâmetros.

13. No modelo de regressão múltipla, determine $X'X$ , $(X'X)^{-1}$, $cov(\hat{\beta})$, os intervalos de confiança para os parâmetros.

14. Determine as equações normais do modelo de regressão de potência k.

15. No tratamento matricial do modelo de regressão linear, determine $E(\hat{\beta})$, e $D = E[\hat{\beta} - \beta][\hat{\beta} - \beta]'$.

16. Construa na linguagem C, programas para gerar números aleatórios normais, e números aleatórios t.

17. Construa na linguagem C, um programa para gerar estatísticas como, mínimo, máximo, soma e média.

18. No R determine quantos geradores de números aleatórios estão disponíveis. A algum pacote que acrescente a quantidade de geradores padrão, pesquise?

19. Pesquise o pacote **quantreg** com a função **qr()**, e implemente no exemplo da produção de batatas.

20. Implemente simulações de experimentos de Monte Carlo de um modelo de Regressão quadrática sem violação de um pressuposto básico.

21. Implemente simulações de experimentos de Monte Carlo de um modelo de Regressão quadrática com violação de não normalidade dos erros.

22. Escreva os comandos no R, de um objeto do tipo m=lm(), para determinar: O resumo dos resultados, os coeficientes, os resíduos, ajuste, análise de variância, gráfico.

23. Com a seguinte saída no consumo de água, determine os Intervalos de confiança bicaudal com 5 por cento de significância do valor pago.

```
predict(lm(valor~consumo), newdata=data.frame(consumo=c(20,25)), se.fit=T)
```

24. Ainda com as dados de consumo de água em uma residência use oas seguintes comandos para identificar no gráfico os tamanhos dos resíduos como segmentos no modelo de regressão linear simples

```
m2=lm(valor~consumo)
plot(valor~consumo)
abline(m2,col=2)
segments(consumo,fitted(m2),consumo,valor)
```

25. Determine a significância estatística da correlação entre consumo e valor de consumo de água, utilizando os coeficientes de correlaçao de Pearson, Spearman, e Kendall.

26. Identifique e responda em detalhe, que esta sendo calculado no R. par(mfrow=c(2,3)), confint(), vcov(), mk=lm(y poli(k)).

27. Identifique e explique em detalhe os comandos no R, para seleção de regressores.

28. Derive os EMQO para o modelo de regressão de potencia de ordem k.

29. Considere os anos e a Taxa média de crescimento geométrico da cidade de Aracaju, a partir de um modelo quadrático, construa um gráfico de previsão destas taxas para $2011 - 2030$ com os comandos a seguir:

```
#Censos e Taxas em  Aracaju - Brasil
anosaju=c(1970,1980,1991,2000,2010)
taxaaju=c(0.04870,0.04785,0.02921,0.01537,0.02154)

mod3=lm(taxaaju~anosaju+I(anosaju^2))
pmod3=predict(mod3,data.frame(anosaju=2011:2030),se.fit=TRUE)
plot(anosaju,taxaaju, ylab="Taxa Geométrica",xlim=c(1960,2040),ylim=c(0,0.05),
lines(anosaju,predict(mod3))
abline(v=2010,lty=2, col="gray")
legend("topright", legend="previsões",lty=4)
lines(2011:2030,pmod3$fit,lty=4)
pmod3$fit
```

# Capítulo 4

# Programação em C

Atualmente, pesquisas e simulações em algumas ciências aplicadas, como em Econometria por exemplo, requerem alguma programação. Um exemplo é o bootstrap que usualmente requer investigações onde se faça uma re-amostragem dos dados originais com reposição. Outros métodos como amostrador de Gibbs e Integração de Monte Carlo requerem igualmente uma intensiva programação.

A linguagem de programação Fortran foi, por algum tempo, a linguagem padrão para a comunidade científica. Outras linguagens, tais como C, C++ e Pascal, têm mostrado boa performance com algoritmos de computação numérica. Para lidar com matrizes muito grandes as linguagens mais apropriadas são GAUSS, MATLAB, Ox, e S-PLUS.

Muitos sistemas operacionais, como Windows e UNIX, são escritos em C que, além disso, têm inúmeras de vantagens sobre outras linguagens. A linguagem de programação C é membro da Família ALGOL, e é muito similar a Ada e Pascal, mas se distancia do BASIC, FORTRAN e LISP (Harbinson and Steele (1995)). Na linguagem C, o usuário pode facilmente implementar novas funções, escrever um programa de simulação Monte Carlo e rapidamente modificar a codificação quando necessário. Outro detalhe importante é que C tem acesso direto à memória do computador através do uso de ponteiros, os quais são extremamente úteis em diversas aplicações. Os compiladores de C têm considerável liberdade na seleção e ordem de avaliação de expressões.

## 4.1 Simulação em C

Nosso estudo baseia-se em 1000 e 10000 réplicas de simulação de Monte Carlo. O objetivo é avaliar o desempenho em amostras de tamanho finito do Teste Jarque-Bera de normalidade dos erros no modelo linear de regressão. Inicialmente consideramos o modelo de regressão linear simples dado por

$$y_i = \beta_1 + \beta_2 x_i + \epsilon_i \qquad i = 1, 2, \ldots n \tag{4.1}$$

Aleatoriedade refere-se à geração ou uso de um conjunto verdadeiramente aleatório, de sequências de números aleatórios. Como sabemos, na prática utiliza-se uma sequência pseudo-aleatória de números.

## 4.2 Simulação de Números Aleatórios

Na construção de geradores aleatórios, o processo deve obedecer algumas restrições de velocidade e cobertura do intervalo de geração. Porém, no uso em simulações é necessário que o processo seja capaz de gerar números que obedeçam diferentes funções de distribuição de probabilidade (fdp), tais como uniforme e normal, Papoulis e Pillai (2002). Nesses casos, esses geradores devem ser independentes, para que não ocorram interferências entre os números gerados em cada distribuição. A partir do gerador para a distribuição uniforme é possível definir geradores para outras funções de distribuição, como a normal, exponencial, gama, entre outras (Knuth [17]).
Dentre as varias funções de distribuição existentes, utilizamos as funções uniforme e a normal por uma média e um desvio padrão. Essas funções foram escolhidas devido a sua usabilidade. Para determinar se os erros amostrais no modelo de regressão linear simples provem de uma distribuição populacional normal, abordaremos o erro do tipo I, e o poder do teste $(1 - \beta)$ descrito no capítulo 2. O algoritmo de geração de números aleatórios uniformes utilizado será o desenvolvido por George Marsaglia.

## 4.3 A simulação e o teste de Jarque-Bera

Na simulação consideramos os tamanhos amostrais 20, 40, 60, 80, 100, 200 e 500. Além disso, o teste de Jarque-Bera de normalidade é baseado em uma aproximação assintótica, pois utilizamos valores críticos da distribuição Qui-quadrado que somente são válidos quando há infinitas observações. Assim, num modelo de regressão que utiliza um número finito de observações, é importante verificar como o teste se comporta em amostras de tamanho típico.

## 4.4 Métodos

Para estimar os parâmetros do modelo descrito em (3.1), utilizaremos o método dos mínimos quadrados. Para gerar sequencias números aleatórios para a co-variável $x_i$ utiliza-se a distribuição uniforme, cujo método utilizado é o gerador congruencial multiplicativo com carregamento descrito por Marsaglia. No caso dos erros $\epsilon_i$ utilizamos o gerador dado pelo método polar a fim de obter sequências pseudo-aleatórias gaussianas.

Por compilação entende-se a conversão de um programa escrito num dado nível de abstração em outro de nível inferior. Historicamente, o termo surgiu da conversão de um programa para o nível do assembly. Contudo, a maior parte dos utilitários conhecidos como compiladores permitem, com uma única linha de comando, passar diretamente o nível da linguagem máquina, executando na realidade 4 programas distintos, correspondentes a 4 fases diferentes: pré-processamento, compilação, montagem (com assembler) e união (com o linker). A versão mais atualizada do manual do compilador de C distribuído pelo projeto GNU, o gcc, está disponível online em:
`http://www.gnu.org/software/gcc/onlinedocs/`.
utilizamos o compilador de linha de comando gcc. O gcc da GNU é um dos compiladores de C mais versáteis que existem no mercado e, além disso, é software de domínio livre, o que significa que temos liberdade para:

1. Executar o software, qualquer que seja o nosso propósito.
2. Estudar o modo como o software funciona e adaptá-lo às nossas necessidades
3. Melhorar o software e distribuir esses melhoramentos para benefício da comunidade em geral.

O gcc suporta os padrões modernos da linguagem C (como o ANSI C), ao mesmo tempo que mantém a compatibilidade com os compiladores e estilos mais antigos.

## 4.5 Resultados

Para as simulações foi adotado o modelo de regressão linear simples da forma $y_i = \beta_1 + \beta_2 x_i$, $i = 1, 2, \ldots, n$. Os tamanhos amostrais considerados foram 20, 40, 60, 80, 100, 200 e 500. Para esses tamanhos amostrais, os valores da covariável foram replicados considerando um gerador pseudo-aleatório uniforme, ou seja, cada valor de $x_i$, foi replicado em 1000 e 10000 simulações Monte Carlo. Utilizou-se o teste de Jarque-Bera para confirmar se os erros do modelo de regressão linear simples de uma amostra, provém de uma população normal. Nas amostras geradas selecionamos 3 diferentes erros do tipo I ($\alpha = \text{p(Aceitar } H_0 | H_0$ é falsa)): $\alpha = 10\%$, $\alpha = 5\%$ e $\alpha = 1\%$. Para o poder do teste mantivemos esses valores de $\alpha$.

A validação dos testes de números aleatórios ocorreu através dos testes Quiquadrado, no caso de Jarque-Bera. Para o teste de Lilliefors utilizamos o procedimento descrito em 3.36.2.

| Taxas de Rejeição Jarque-Bera | | | | | | |
|---|---|---|---|---|---|---|
| | 1000 réplicas | | | 10000 réplicas | | |
| n | $\alpha = 10\%$ | $\alpha = 5\%$ | $\alpha = 1\%$ | $\alpha = 10\%$ | $\alpha = 5\%$ | $\alpha = 1\%$ |
| 20 | 2,10 | 1,40 | 0,80 | 2,66 | 1,67 | 0,79 |
| 40 | 4,90 | 3,10 | 1,30 | 4,40 | 2,96 | 1,50 |
| 60 | 5,00 | 3,04 | 1,50 | 5,60 | 3,84 | 1,69 |
| 80 | 6,40 | 3,80 | 1,90 | 5,72 | 3,59 | 1,64 |
| 100 | 6,60 | 3,30 | 1,70 | 6,28 | 4,25 | 1,96 |
| 200 | 7,30 | 5,00 | 1,80 | 7,35 | 4,40 | 1,66 |
| 500 | 6,90 | 3,30 | 0,80 | 9,03 | 4,83 | 1,63 |

Tabela 4.1: Taxas de rejeição Jarque-Bera para erros com distribuição normal

Na Tabela 4.1 apresentamos os resultados do teste de Jarque-Bera para os níveis nominais de 10%, 5% e 1%, com erros normais. Uma inspeção nessa tabela nos permite verificar que o desempenho desse teste melhora consideravelmente quando o tamanho da amostra cresce.

| Taxas de rejeição Lilliefors | | | | | | |
|---|---|---|---|---|---|---|
| | 1000 réplicas | | | 10000 réplicas | | |
| n | $\alpha = 10\%$ | $\alpha = 5\%$ | $\alpha = 1\%$ | $\alpha = 10\%$ | $\alpha = 5\%$ | $\alpha = 1\%$ |
| 20 | 10,80 | 6,20 | 1,40 | 9,73 | 4,95 | 1,04 |
| 40 | 11,60 | 6,50 | 1,10 | 9,73 | 4,78 | 0,81 |
| 60 | 11,40 | 5,90 | 1,00 | 10,31 | 4,93 | 1,13 |
| 80 | 10,70 | 5,10 | 1,10 | 10,18 | 4,84 | 0,99 |
| 100 | 11,00 | 5,00 | 1,00 | 10,77 | 5,56 | 1,21 |
| 200 | 12,20 | 5,90 | 1,80 | 10,60 | 5,14 | 0,97 |
| 500 | 9,40 | 4,40 | 1,10 | 10,84 | 5,28 | 1,18 |

Tabela 4.2: Taxas de rejeição Lilliefors para erros com distribuição normal

Na Tabela 4.2 apresentamos os resultados do teste de Lilliefors para os níveis nominais de 10%, 5% e 1%, com erros normais. Observando esta tabela podemos concluir que este teste apresenta resultados satisfatórios mesmo com tamanhos de amostra pequenos, ou seja, os índices de rejeição estão sempre próximos do valor real. Comparando as tabelas 4.1 e 4.2, concluímos que ambos os testes têm desempenho satisfatório, porém o teste de Lilliefors teve destaque na convergência dos índices de rejeição.

| Poder do teste Jarque-Bera: $(1-\beta)\%$ | | | | | | |
|---|---|---|---|---|---|---|
| | 1000 réplicas | | | 10000 réplicas | | |
| n | $\beta = 10\%$ | $\beta = 5\%$ | $\beta = 1\%$ | $\beta = 10\%$ | $\beta = 5\%$ | $\beta = 1\%$ |
| 20 | 31,50 | 26,70 | 21,10 | 30,25 | 26,16 | 20,19 |
| 40 | 58,50 | 55,00 | 47,10 | 59,44 | 54,73 | 47,31 |
| 60 | 71,30 | 68,10 | 62,20 | 75,57 | 71,60 | 64,76 |
| 80 | 82,70 | 80,10 | 74,40 | 84,05 | 81,18 | 75,32 |
| 100 | 90,30 | 87,30 | 83,50 | 91,45 | 89,54 | 85,03 |
| 200 | 99,00 | 98,60 | 98,10 | 99,30 | 99,04 | 98,05 |
| 500 | 100,00 | 100,00 | 100,00 | 100,00 | 100,00 | 100,00 |

Tabela 4.3: Poder do teste Jarque-Bera para erros com distribuição $t_{(3)}$

| Poder do teste Lilliefors: $(1-\beta)\%$ | | | | | | |
|---|---|---|---|---|---|---|
| | 1000 réplicas | | | 10000 réplicas | | |
| n | $\beta = 10\%$ | $\beta = 5\%$ | $\beta = 1\%$ | $\beta = 10\%$ | $\beta = 5\%$ | $\beta = 1\%$ |
| 20 | 32,50 | 24,00 | 11,80 | 31,58 | 22,92 | 11,61 |
| 40 | 47,80 | 39,30 | 20,40 | 48,57 | 39,04 | 23,24 |
| 60 | 60,40 | 48,70 | 31,50 | 62,92 | 51,83 | 34,65 |
| 80 | 69,70 | 61,40 | 44,30 | 72,71 | 62,79 | 45,13 |
| 100 | 81,40 | 71,70 | 56,40 | 81,04 | 72,70 | 56,20 |
| 200 | 97,10 | 93,80 | 84,40 | 96,94 | 94,11 | 85,68 |
| 500 | 100,00 | 100,00 | 99,80 | 99,99 | 99,97 | 99,87 |

Tabela 4.4: Poder do teste Lilliefors para erros com distribuição $t_{(3)}$

Os erros do modelo de regressão $\epsilon_i$ são independentemente e identicamente distribuídos seguindo uma distribuição $N(0;1)$. Para a a geração de dados heterocedásticos utilizou-se o seguinte processo:

1. Gerar um valor $x_i$ da variável $X$ com distribuição uniforme.
2. Gerar o erro $\epsilon_i$ a partir de uma normal com média 0 e variância $\sigma_i = x_i^2$.
3. Calcular $y_i = \beta_1 + \beta_2 x_i + \epsilon_i$.

Os resultados estão dispostos na tabela da próxima página.

| Taxas de Rejeição Jarque-Bera: Modelo heterocedástico | | | | | | |
|---|---|---|---|---|---|---|
| | 1000 réplicas | | | 10000 réplicas | | |
| n | $\beta = 10\%$ | $\beta = 5\%$ | $\beta = 1\%$ | $\beta = 10\%$ | $\beta = 5\%$ | $\beta = 1\%$ |
| 20 | 32,50 | 24,00 | 11,80 | 31,58 | 22,92 | 11,61 |
| 40 | 47,80 | 39,30 | 20,40 | 48,57 | 39,04 | 23,24 |
| 60 | 60,40 | 48,70 | 31,50 | 62,92 | 51,83 | 34,65 |
| 80 | 69,70 | 61,40 | 44,30 | 72,71 | 62,79 | 45,13 |
| 100 | 81,40 | 71,70 | 56,40 | 81,04 | 72,70 | 56,20 |
| 200 | 97,10 | 93,80 | 84,40 | 96,94 | 94,11 | 85,68 |
| 500 | 100,00 | 100,00 | 99,80 | 99,99 | 99,97 | 99,87 |

Tabela 4.5: Taxas de rejeição Jarque-Bera: Modelo heterocedástico

Para executar as simulações utilizamos um computador com processador Intel® Pentium 4 de 3.2 HGz, com 1 GB de memória RAM. O sistema operacional dessa máquina é o Microsoft Windows XP Profissional, Service Pack 2.

Para os tempos de execução, obtivemos os seguintes resultados:

- Jarque-Bera com 1000 réplicas e erros normais: 0,39 seg.
- Jarque-Bera com 10000 réplicas e erros normais: 3,81 seg.
- Jarque-Bera com 1000 réplicas e erros $t_{(3)}$: 0,32 seg.
- Jarque-Bera com 10000 réplicas e erros $t_{(3)}$: 3,13 seg.
- Lilliefors com 1000 réplicas e erros normais: 163,95 seg.
- Lilliefors com 10000 réplicas e erros normais: 1622,38 seg.
- Lilliefors com 1000 réplicas e erros $t_{(3)}$: 164,46 seg.
- Lilliefors com 10000 réplicas e erros $t_{(3)}$: 1628,30 seg.

## 4.6 Código C: Teste de normalidade de Jarque-Bera

```
/* Simulacao Monte Carlo: Teste Jarque-Bera com erros normais*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

/* Gerador de números aleatorios uniformes (George Marsaglia) */
#define s1new (s1=(18000*(s1&0xFFFF)+(s1>>16)))
#define s2new (s2=(30903*(s2&0xFFFF)+(s2>>16)))
#define UNI (((s1new<<16)+(s2new&0xFFFF))*2.32830643708e-10)
static unsigned long s1=362436069, s2=521288629;
#define setseed(seed1,seed2) {s1=seed1;s2=seed2;}

/* Estruturas */
struct ponto{
       double beta1;
       double beta2;
};

struct rej{
       int rej01;
       int rej05;
       int rej10;
};

/* Gerador de números aleatorios normais (Metodo polar) */
double rann(void);

/* Geracao das amostras */
void GeraAmostra(int n, double e[],double x[], double y[],
                 double b1,double b2);

/* Estimacao dos parametros beta1(coeficiente constante) e beta2 */
struct ponto estimar(int n,double x[],double ycalculado[]);

/* Teste de Bera-Jarque */
struct rej BJ(int n, double erroest[]);


/* Tabela de resultados - Parte 1 */
void ImprimeTabela1(int n);
```

```
/* Tabela de resultados - Parte 2 */
void ImprimeTabela2(int n);


int main(void)
{
    int n[8]={20,40,60,80,100,200,500};  // Tamanhos amostrais
    int Rep1=1000,Rep2=10000; // número de replicas
    int i,j,k;     //Contadores
    double b1=2.0,b2=1.0;  // Valores reais dos parametros
    struct ponto estimados; // Armazena parametros estimados
    double *x,*ycalculado,*e;   // Amostra
    double *yest,*erroest; // Valores estimados para y
                           //e para o erro
    struct rej RetornoBJ;
    int NumRej01, NumRej05, NumRej10;
    double start_time, end_time;


    /* Bera-Jarque com erros normais*/
    /* Tabela de resultados - Parte 1 */
    ImprimeTabela1(Rep1);


    start_time=clock(); // Iniciando cronometro
    /* 1000 Replicas */
    for(k=0;k<7;k++)
    {
        x=calloc(n[k],sizeof(double));
        ycalculado=calloc(n[k],sizeof(double));
        e=calloc(n[k],sizeof(double));
        yest=calloc(n[k],sizeof(double));
        erroest=calloc(n[k],sizeof(double));


        NumRej01=0, NumRej05=0, NumRej10=0;
        for(j=0;j<Rep1;j++)
        {
            /* Geracao da amostra */
            GeraAmostra(n[k],e,x,ycalculado,b1,b2);

            /* Estimacao de parametros */
            estimados=estimar(n[k],x,ycalculado);

            /* Valores estimados de y e dos erros*/
```

```
            for(i=0;i<n[k];i++)
            {
                yest[i]=estimados.beta1+estimados.beta2*x[i];
                erroest[i]=yest[i]-ycalculado[i];
            }

            RetornoBJ=BJ(n[k],erroest);
            NumRej01+=RetornoBJ.rej01;
            NumRej05+=RetornoBJ.rej05;
            NumRej10+=RetornoBJ.rej10;
        }

        printf("\n");
        for(i=0;i<10;i++) printf(" ");
        printf(" %4d   %5.2lf    %5.2lf    %5.2lf",
               n[k],100.0*NumRej10/Rep1,100.0*NumRej05/Rep1,
               100.0*NumRej01/Rep1);

        /* Liberar memoria */
        free(x);free(ycalculado);free(e);free(yest);free(erroest);

        printf("\n");
    }

    // Finalizando cronometro
    end_time=clock();
   // impressao do tempo de execucao do programa
   printf("\n\nTEMPO DE EXECUCAO:    %10.4f segundos\n",
                  ((double)(end_time- start_time))/CLOCKS_PER_SEC);


    /* 10000 Replicas */
    /* Tabela de resultados - Parte 2 */
    ImprimeTabela2(Rep2);

    start_time=clock(); // Iniciando cronometro

    for(k=0;k<7;k++)
    {
        x=calloc(n[k],sizeof(double));
        ycalculado=calloc(n[k],sizeof(double));
        e=calloc(n[k],sizeof(double));
        yest=calloc(n[k],sizeof(double));
        erroest=calloc(n[k],sizeof(double));

        NumRej01=0, NumRej05=0, NumRej10=0;
```

```
        for(j=0;j<Rep2;j++)
        {
            /* Geracao da amostra */
            GeraAmostra(n[k],e,x,ycalculado,b1,b2);

            /* Estimacao de parametros */
            estimados=estimar(n[k],x,ycalculado);

            /* Valores estimados de y e dos erros*/
            for(i=0;i<n[k];i++)
            {
                yest[i]=estimados.beta1+estimados.beta2*x[i];
                erroest[i]=yest[i]-ycalculado[i];
            }

            RetornoBJ=BJ(n[k],erroest);
            NumRej01+=RetornoBJ.rej01;
            NumRej05+=RetornoBJ.rej05;
            NumRej10+=RetornoBJ.rej10;
         }

         printf("\n");
         for(i=0;i<10;i++) printf(" ");
         printf(" %4d   %5.2lf    %5.2lf    %5.2lf",
                n[k],100.0*NumRej10/Rep2,100.0*NumRej05/Rep2,
                100.0*NumRej01/Rep2);

         /* Liberar memoria */
         free(x);free(ycalculado);free(e);free(yest);free(erroest);

         printf("\n");
    }
   // Finalizando cronometro
    end_time=clock();

   // impressao do tempo de execucao do programa
   printf("\n\nTEMPO DE EXECUCAO:    %10.4f segundos\n",
                  ((double)(end_time- start_time))/CLOCKS_PER_SEC);


    printf("\n\n");
    system("pause");
}

/* Gerador de números aleatorios normais (Metodo polar) */
double rann(void)
```

```
{
    static int iset=0;
    static double gset;
    double fac, rsq, v1, v2;

    if(!iset)
    {
        do
        {
             v1 = 2.0*UNI -1.0;
             v2 = 2.0*UNI -1.0;
             rsq = v1*v1 + v2*v2;
        }
        while (rsq>=1||!rsq);

        fac = sqrt(-2.0*log(rsq)/rsq);
        gset = v1*fac;
        iset = 1;

        return (v2*fac);
    }
    else
    {
        iset = 0;
        return gset;
    }
}

/* Geracao das amostras */
void GeraAmostra(int n, double e[],double x[], double y[],
                 double a,double b)
{
    int i;

    /* Geracao dos erros e[i] */
    for(i=0;i<n;i++)
        e[i]=rann();

    /* Gerando x uniformente e calculando y=A+Bx*/
    for(i=0;i<n;i++)
    {
        x[i]=UNI;

        /* Valor real de y */
        y[i]=a+b*x[i]+e[i];
    }
```

```
}


/* Estimacao dos parametros beta1  e beta2 */
struct ponto estimar(int n,double x[],double ycalculado[])
{
     int i;
     double num=0,den=0, mediax=0, mediay=0;
     struct ponto est;

     // Calculo das medias de x e de y
     for(i=0;i<n;i++)
     {
         mediax+=x[i];
         mediay+=ycalculado[i];
     }
     mediax/=n;
     mediay/=n;


     /* Estimacao do parametro beta2 (coeficiente angular)*/
     for(i=0;i<n;i++)
     {
         num+=((ycalculado[i]-mediay)*(x[i]-mediax));
         den+=((x[i]-mediax)*(x[i]-mediax));
     }

     est.beta2=num/den;

     /* Estimacao do parametro beta1 (coeficiente linear)*/
     est.beta1=mediay-est.beta2*mediax;

     return est;
}


/* Teste de Bera-Jarque */
struct rej BJ(int n, double erroest[])
{
     int i;
     double SomaErros=0,soma=0, soma2=0, soma3=0, soma4=0;
     double m, variancia,dp,assimetria,curtose,jb;
     struct rej Rejeicao={0,0,0};

     double static vc01=9.210340372, vc05=5.991464547;
     double vc10=4.605170186;
```

```
     for(i=0;i<n;i++)
         SomaErros+=erroest[i];

     m = SomaErros/n; // media dos erros

     for(i=0;i<n;i++)
     {
         soma+=(erroest[i]-m);
         soma2+=((erroest[i]-m)*(erroest[i]-m));
         soma3+=((erroest[i]-m)*(erroest[i]-m)*(erroest[i]-m));
         soma4+=((erroest[i]-m)*(erroest[i]-m)*
                (erroest[i]-m)*(erroest[i]-m));
     }

     variancia=soma2/(n-1);
     dp=sqrt(variancia);

     assimetria=(soma3/(n-1))/(dp*dp*dp);
     curtose=(soma4/(n-1))/(dp*dp*dp*dp);
     jb=n*(assimetria*assimetria/6.0+
        (curtose-3.0)*(curtose-3.0)/24.0);

     if(jb>=vc01)
        Rejeicao.rej01=1;

     if(jb>=vc05)
        Rejeicao.rej05=1;

     if(jb>=vc10)
        Rejeicao.rej10=1;

     return Rejeicao;
}

/* Tabela de resultados - Parte 1 */
void ImprimeTabela1(int n)
{
     int i;

     for(i=0;i<10;i++) printf(" ");
     for(i=0;i<34;i++) printf("-");
     printf("\n");
     for(i=0;i<20;i++) printf(" ");
     printf("Bera-Jarque\n");
```

```
     for(i=0;i<10;i++) printf(" ");
     for(i=0;i<34;i++) printf("-");
     printf("\n");
     for(i=0;i<20;i++) printf(" ");
     printf("Replicas: %d",n);
     printf("\n");
     for(i=0;i<10;i++) printf(" ");
     printf("\n");
     for(i=0;i<12;i++) printf(" ");
     printf("  n    0.10     0.05     0.01\n");
     for(i=0;i<10;i++) printf(" ");
     for(i=0;i<34;i++) printf("-");
}

/* Tabela de resultados - Parte 2 */
void ImprimeTabela2(int n)
{
     int i;

     for(i=0;i<10;i++) printf(" ");
     for(i=0;i<34;i++) printf("-");
     printf("\n");
     for(i=0;i<20;i++) printf(" ");
     printf("Replicas: %d",n);
     printf("\n");
     for(i=0;i<10;i++) printf(" ");
     printf("\n");
     for(i=0;i<12;i++) printf(" ");
     printf("  n    0.10     0.05     0.01\n");
     for(i=0;i<10;i++) printf(" ");
     for(i=0;i<34;i++) printf("-");
}
```

## 4.7 Código C: Poder do teste de normalidade de JB

Com relação ao programa anterior, foi feita apenas uma mudança. Os erros foram gerados a partir de uma distribuição t com três graus de liberdade. Seu código C é dado a seguir:

```
/* Gerador de números aleatorios: distribuicao t */
double rant(double a)
{
   static double c,e,p
 \begin{flushleft}
```

```
 \end{flushleft}
 ,q,r,vm,old_a;
   double tru,u,v;

   /* Passo 0: Inicializacao */
   if (a != old_a)
   {
       old_a = a;
       r = 1.0 / a;
       p = 1.0 / (1.0 + r);
       q = -0.25 * (a + 1);
       c = 4.0 * pow(p, q);
       e = 16.0 / c;
       vm = (a>1.0)? sqrt(p+p) * pow( (1.0-r) * p, 0.25*(a-1.0) ):1.0;
   }

   for (;;)
   {
       /* Passo 1 */
       u = UNI;

       /* Passo 2 */
       v = UNI;
       v = vm * (v + v - 1.0);
       tru = v / u;

       /* Passo 3 */
       if ( c * u <= 5.0 - tru * tru)
           return(tru);
       if (a >= 3.0)
           if (u*(tru*tru+3.0) >= e)
               continue; /* goto 1 */

       /* Passo 4 */
       if (u <= pow(1.0 + tru * tru * r , q))
           return(tru);
   }
}
```

## 4.8 Código C: Teste de normalidade de Lilliefors

Mudança do teste de normalidade. Lembramos que no teste de Lilliefors a amostra precisa ser ordenada, para isso usamos o algoritmo quicksort. Para

testar o poder do teste, utilizamos novamente uma amostra t com três graus de liberdade. O seguinte código foi adicionado:

```
/* Distribuicao empirica de amostras */
double *DistEmp(int n, double x[])
{
    int i,j;
    int *freq;
    double *dist;

    freq = calloc(n,sizeof(int));
    dist = calloc(n,sizeof(double));

    for(i=0;i<n;i++)
        freq[i]=0;

    for(j=0;j<n;j++)
    {
        for(i=0;i<n;i++)
            if(x[i]<=x[j])
               freq[j]+=1;
    }


    for(i=0;i<n;i++)
    {
        dist[i]=(double)freq[i]/n;
    }

    free(freq);
    return dist;
}

/* Funcao de distribuicao normal */
double phi(double x)
{
   return exp(-x*x/2.0)/sqrt(2*3.14159265);
}


/* Integral numerica no intervalo (a,b), dividido em n partes */
double PNormal(double a, double b, int n)
{
    int i;
    double h,area=0.0;
```

```
    h=(b-a)/n;
    for(i=0;i<n;i++)
        area+=(phi(a+i*h)+phi(a+(i+1)*h))*h/2.0;

    return area;
}

/* Subalgoritmo utilizado pelo quicksort */
int particiona(int left, int right, double x[])
{
   int i = left+1, j = right;
   double temp, c = x[left];
   while (1) {
      while (i <= right && x[i] <= c) i++;
      while (c < x[j]) j--;
      if (i >= j) break;
      temp = x[i]; x[i] = x[j]; x[j] = temp;
      i++; j--;
   }
   temp = x[left]; x[left] = x[j]; x[j] = temp;
   return j;
}


/* Algoritmo de ordenacao */
void quicksort (int left, int right, double x[])
{
   int j;
   if (left < right) {
      j = particiona(left, right, x);
      quicksort(left, j-1, x);
      quicksort(j+1, right, x);
   }
}


/* Teste de Lilliefors */
struct rej LF(int n, double erroest[])
{
     int i;
     double SomaErros=0,soma2=0,T,T1;
     double media, variancia,dp;
     double *z,*DistZ;
     struct rej Rejeicao={0,0,0};
     double fN;
```

```
/* Valores criticos */
double static vcN20[3]={0.2226,0.1920,0.1764};
double static vcN40[3]={0.1616,0.1386,0.1275};
double static vcN60[3]={0.157,0.175,0.210};
double static vcN80[3]={0.136,0.152,0.182};
double static vcN100[3]={0.122,0.136,0.163};
double static vcN200[3]={0.086,0.096,0.115};
double static vcN500[3]={0.054,0.060,0.072};
double static vcN1000[3]={0.003,0.004,0.005};


/* Ordenando os valores dos erros */
quicksort(0,n-1,erroest);

for(i=0;i<n;i++)
    SomaErros+=erroest[i];

media = SomaErros/n; // media dos erros

for(i=0;i<n;i++)
{
    soma2+=((erroest[i]-media)*(erroest[i]-media));
}

variancia=soma2/(n-1);
dp=sqrt(variancia);

z=calloc(n,sizeof(double));
DistZ=calloc(n,sizeof(double));

for(i=0;i<n;i++)
{
    z[i]=(erroest[i]-media)/dp;
}

/* Distribuicao acumulada dos z[i] (empirica)
   Mede o valor de P(z<=z[i]) empiricamente */
DistZ = DistEmp(n,z);


/* Estatistica de Teste */
T1=DistZ[0]-PNormal(-6,z[0],100);
T=PNormal(-6,z[0],100)-DistZ[0];
if(T>T1) T1=T;
T=DistZ[0]-PNormal(-6,z[1],100);
if(T>T1) T1=T;
```

```
T=PNormal(-6,z[1],100)-DistZ[0];
if(T>T1) T1=T;


for(i=1;i<n;i++)
{
    T=DistZ[i]-PNormal(-6,z[i],100);
    if(T>T1) T1=T;

    T=PNormal(-6,z[i],100)-DistZ[i];
    if(T>T1) T1=T;

    T=PNormal(-6,z[i],100)-DistZ[i-1];
    if(T>T1) T1=T;

    T=DistZ[i-1]-PNormal(-6,z[i],100);
    if(T>T1) T1=T;
}


/* ********** Verificando rejeicao ********** */
/* n = 20 */
if(n==20)
{
   if(T1>vcN20[0])
      Rejeicao.rej01=1;
   if(T1>vcN20[1])
      Rejeicao.rej05=1;
   if(T1>vcN20[2])
      Rejeicao.rej10=1;
}

/* n = 40 */
if(n==40)
{
   if(T1>vcN40[0])
      Rejeicao.rej01=1;
   if(T1>vcN40[1])
      Rejeicao.rej05=1;
   if(T1>vcN40[2])
      Rejeicao.rej10=1;
}


/*  n > 50*/
if(n>50)
```

```
     {
        fN=(0.83+n)/sqrt(n)-0.01;
        if(T1>1.035/fN)
           Rejeicao.rej01=1;
        if(T1>0.895/fN)
           Rejeicao.rej05=1;
        if(T1>0.819/fN)
           Rejeicao.rej10=1;
     }


     free(z);
     free(DistZ);

     return Rejeicao;
}
```

## 4.9 Código C: Jarque-Bera: Modelo heterocedástico

No modelo heterocedástico, para cada valor $x_i$ geramos um erro $\epsilon_i$ com distribuição uniforme de média 0 e desvio padrão $\sigma = x_i^2$. Foi feita apenas uma alteração no código que gera a amostra:

```
/* Geracao das amostras */
void GeraAmostra(int n, double e[],double x[], double y[],
                 double a,double b)
{
    int i;
    double dp;

    /* Gerando x uniformente e calculando y=A+Bx*/
    for(i=0;i<n;i++)
    {
        x[i]=UNI;
        dp=x[i]*x[i];
        e[i]=Normal(0,dp);

        /* Valor real de y */
        y[i]=a+b*x[i]+e[i];
    }
}
```

# Capítulo 5

# Casos especiais de Regressão

## 5.1 ANOVA em delineamentos

Em situações que o interesse é identificar se determinada variável ou conjunto de variáveis sofrem heteogeneidade e causam efeito significativo, usamos o análise de variância. Um outro teste como o de tukey é usado para comparações múltiplas de médias.

### 5.1.1 Inteiramente causalizado

Considere 4 típos de juros anuais cobrados a 20 clientes.

```
> dados=c(25,31,22,33,26,25,26,29,20,28,28,31,23,27,25,34,21,24,29,28)
> juros=factor(rep(paste("juro",1:4, sep=""),5))
> resposta=aov(dados~juros)
> anova(resposta) #pode usar summary alternativamente
Analysis of Variance Table

Response: dados
          Df Sum Sq Mean Sq F value   Pr(>F)
juros      3 163.75  54.583  7.7976 0.001976 **
Residuals 16 112.00   7.000

> par(mfrow=c(2,2))
> plot(resposta)
> resultado1=TukeyHSD(resposta,"juros")
> resultado1
```

### 5.1.2 Bloco causalizado

Em este experimento consideramos que os blocos são homogêneos. Exemplo. Considere a diminuição de 4 tipos de juros mensais distribuídos em 5 regiões.

```
dad=c(2,5,2,5,3,7,4,3,2,6,5,4,4,5,1,3,2,5,4,4)
bloco=gl(5,4,label=c(paste("região",1:5)))
juros=rep(paste("juros",1:4),5)
tabela=data.frame(bloco,juros=factor(juros),dad)
tabela
saida=aov(dad~juros+bloco,tabela)
anova(saida)
Analysis of Variance Table

Response: dad
          Df Sum Sq Mean Sq F value   Pr(>F)
juros      3   25.2     8.4  6.0000 0.009731 **
bloco      4    3.2     0.8  0.5714 0.688544
Residuals 12   16.8     1.4
```

Interpretação Observe que os juros teve significância estatística em consequência, a diminuição dos juros nos mercados apresentou diferença significativa. Observe que o bloco (regiões) não teve significância estatística em consequência as regiões apresentam diminuição de juros semelhantes (em média iguais)

Análise dos Resíduos

```
residuos=resid(saida) #
sum(residuos)  #soma é preaticamente zero
tapply(dad,juros,sum) # soma dos juros
tapply(dad,juros,mean) # média dos juros
tapply(dad,bloco,sum) # soma das regiões ou blocos
tapply(dad,bloco,mean) # média das regiões
```

Gráficos para identificar heterogeneidade entre os blocos.

```
library(lattice)
xyplot(dad~juros,data=tabela,groups=bloco,type=c("p","r"),
col=c(1,2,3,4,5),pch=c(1,2,3,4,5),
key=list(x=0.7,y=0.85,points=list(col=c(1,2,3,4,5),pch=c(1,2,3,4,5)),
text=list(levels(tabela$bloco))))

xyplot(dad~bloco,data=tabela,groups=juros,type=c("p","r"),
col=c(1,2,3,4),pch=c(1,2,3,4),
key=list(x=0.7,y=0.85,points=list(col=c(1,2,3,4),pch=c(1,2,3,4)),
text=list(levels(tabela$juros))))
```

Teste Tukey para comparação de Médias entre os grupos

```
resultado3=TukeyHSD(saida,"bloco") Tarefa
resultado3
plot(resultado3)

resultado4=TukeyHSD(saida,"juros",conf.level=0.99)
resultado4
plot(resultado4)
```

## 5.2 Modelos de efeitos Fixos e aleatórios

Estimação de modelos de efeitos fixos e aleatórios é utilizado em dados de painel, onde os valores em cada corte transversal dos dados varia no intercepto. Um modelo padrão de efeitos fixos é dado por:

$$y_{it} = \alpha + \mu_i + \beta x_{it} + \epsilon_{it} \tag{5.1}$$

Considere um exemplo hipotético com os juros cobrados por empréstimos em 10 bancos colhidos em 12 meses (x), e como varável resposta a moratória em meses dos clientes contemplados.

```
set.seed(10)
ef=rep(1:12,10) # 12 Meses
x=c(3,5,7,4,8,7,5,7,8,5,6,9,9,10,12,4,5,8,4,8,9,5,7,7,5,8,9,8,10,13,
9,10,12,4,5,8,4,8,9,5,7,7,5,8,5,6,7,8,6,8,4,8,7,5,7,8,5,6,9,9,9,10,
12,4,5,8,4,8,9,5,7,7,5,8,9,8,10,13,9,10,12,4,5,8,4,8,9,5,7,6,10,13,9,
10,12,4,5,8,4,8,7,7,8,3,4,5,8,4,8,9,5,7,7,5,8,9,8,10,13,9)
# juros de 10 bancos colhidos ao acaso em 12 meses
y=x+3*runif(120) # tempo em meses de moratória
plot(y~factor(ef)+x) # dois gráficos para visualizar os agrupamentos
mf=lm(y~factor(ef)+x) # modelo linear com efeito fixo
summary(mf) #ressumo do modelo
lines(x,fitted(mf),col=2) #ajuste do modelo estimado
tapply(x,ef,mean) # média dos juro cobrados em cada mês
plot(tapply(x,ef,mean),type="l") #gráfico de linhas para as médias dos juros
abline(h=mean(x),col=2) # linha horizontal tracejada (média de x)
plot(mf) #Recursos gráficos do modelo mf
table(x) #tabela de frequencias de x
table(ef,x) #Tabela de frequencias cruzadas entre x e ef
hist(x)   # Histograma
boxplot(x~ef) # Boxplot dos juros
resposta=aov(x~factor(ef))# análise de variância
TukeyHSD(resposta,"factor(ef)")#Teste de comparações múltiplas
```

Um modelo padrão de efeitos aleatórios é dado por:

$$y_{it} = (\alpha + \alpha_i) + \mu_i + \beta x_{it} + \epsilon_{it} \tag{5.2}$$

Exemplo: Considere os gasto em alimentação fora de casa de 120 trabalhadores. Onde consideramos 12 meses e cada mês tem como efeito aleatório os dias úteis (variando de 17 a 23 dias). A variável aleatória x é o salário dos 120 trabalhadores.

```
set.seed(10)
ale=round(runif(120:17,23) # Dias úteis
x=c(1695,2825,3955,2260,4520,3955,2825,3955,4520,2825,3390,5085,
5085,5650,6780,2260,2825,4520,2260,4520,5085,2825,3955,3955,2825,
4520,5085,4520,5650,7345,5085,5650,6780,2260,2825,4520,2260,4520,
5085,2825,3955,3955,2825,4520,2825,3390,3955,4520,3390,4520,2260,
4520,3955,2825,3955,4520,2825,3390,5085,5085,5085,5650,6780,2260,
2825,4520,2260,4520,5085,2825,3955,3955,2825,4520,5085,4520,5650,
7345,5085,5650,6780,2260,2825,4520,2260,4520,5085,2825,3955,3390,
5650,7345,5085,5650,6780,2260,2825,4520,2260,4520,5650,6780,2260,
2825,4520,2260,4520,4520,2825,3390,3955,4520,3390,4520,2260,4520,
3955,2825,3955,4520) # salário de trabalhadores
y=round(0.1273*x+rnorm(120,0,50)) # gasto em alimentação fora de casa
plot(y~factor(ale)+x)
library(nlme)
male=lme(y~x,random=~1+x|ale)
summary(male)
lines(x,fitted(male),col=2)
tapply(x,ale,mean)
plot(tapply(x,ale,mean),type="l")
abline(h=mean(x),col=2)
plot(male)
table(x)
table(ale,x)
hist(x)
boxplot(x~ale)
boxplot(y~ale)
abline(h=mean(y),col=2)
mean(y)
resposta=aov(x~factor(ale))
TukeyHSD(resposta,"factor(ale)")
```

# Capítulo 6

# Superando os pressupostos básicos

## 6.1 Multicolinearidade

Duas ou mais variáveis são colineares se possuem relação exata, ou seja, se um dos vetores é uma combinação linear dos outros (como se fossem retas paralelas). A correlação exata raramente ocorre e maiores detalhes deste fato pode ser encontrado em Gujarati (2000), porem correlações fortes (correlação $r$ acima de $|0.8|$) já são perigosas, assim como regressores auxiliares de cada regressor sobre os demais for alto.
Apenas a correlação entre variáveis independentes é problemática. A relação forte de cada uma das variáveis independentes $x_i$ com a variável dependente $y_i$ é desejável. Quando existem mais de duas variáveis independentes relacionadas fortemente fala-se em *multicolinearidade*.

A multicolinearidade afeta os coeficientes da equação de regressão de forma significativa, os denominados $\hat{\beta}_i$, alterando o valor e até o sinal em relação ao que ocorreria se não houvesse este problema. Na presença de correlação alta, os coeficientes de regressão estimados tendem a ser imprecisos e as estimativas dos coeficientes variam bastante de uma amostra para outra. Quando há colinearidade, as estimativas dos mínimos quadrados ainda são não-tendenciosas e eficientes, porém o erro padrão dos coeficientes tende a ser grande, e o teste baseado na estatística t de Student calculará significância menor que a real. Os coeficientes não são confiáveis, impossibilitando o uso dos modelos para análise do mercado ou previsão de valores. Outro efeito da colinearidade é que torna-se difícil obter interpretações sobre o efeito isolado de cada uma das variáveis. Nos casos de correlação alta, uma das alternativas é a remoção da variável mais afetada. Isso pode introduzir tendências, sendo mais adequado substituir esta variável por outra menos colinear mas que tenha aproximadamente a mesma construção teórica, ou por uma variável que seja a combinação das colineares. Nem sempre a remoção ou substituição da variável afetada é uma boa solução.

Quando se trabalha com predição de valores e existem indicações de que a colinearidade encontrada continuará no futuro e o modelo poderá apresentar bons resultados.

Podemos observar a existência de colinearidade através da matriz de correlação das variáveis independentes.
A sequencia a seguir é identificar a natureza da multicolinearidade, se de fato é um problema e quando, as suas consequências práticas.

### 6.1.1 Natureza

A natureza da multicolinearidade ou existência de uma perfeita ou exata relação linear entre algumas ou todas as variáveis independentes de um modelo de regressão, em geral esta suposição não é realista. É comum a relação quase perfeita entre as variáveis independentes (as variáveis são inter correlacionadas). A multicolinearidade é uma questão de grau e não de natureza. A distinção significativa não esta entre a presença ou ausência de multicolinearidade mas entre seus vários graus. Como a multicolinearidade se refere à condição de variáveis independentes que se presume não estocásticas, é uma característica da amostra, e não da população.
Uma das consequências da multicolinearidade perfeita é que não pode ser estimado os parâmetro, invalidando o MELNT.
Se é quase-perfeita os estimadores de M.Q.O são não tendenciosos, eficientes e consistentes MELNT. Alem disso os teste de hipóteses não são afetados.
Se o interesse é a previsão a multicolinearidade não representa um problema, os contrario diminuem o erro dos resíduos, e as previsões serão não tendenciosas, também os intervalos de confiança para previsão permanecem válidos.

### 6.1.2 Identificação

1. O $R^2$ alto, porem $t-ratio$ baixos, implica teste $F$ altamente significativo.

2. A correlação (a pares) das variáveis independentes muito alto.

3. Estimativa dos parâmetros sensível a especificação ou correlação a pares não é muito alta mas existe uma relação quase perfeita entre mais do que duas variáveis.

### 6.1.3 Corrigindo

1. Se objetivo é previsão ignorar multicolinearidade.

2. Eliminação de variáveis, geral para o particular.

3. Reformulando o modelo, usando razão entre variáveis ou transformando as variáveis para taxa.

4. Usando informação externa, assumir que um dos parâmetros foi estimado fora do modelo e impor como verdadeiro.

5. Usar estimativa de cross-section da elasticidade renda para obter a elasticidade preço numa especificação de séries temporais.

6. Aumentar o tamanho da amostra.

### 6.1.4 Testes de Multicolinearidade

1. Padronização dos regressores: este teste para detectar multicolinearidade é usando o determinante de $X'_m X_m$, onde $X_m = [x_2^m, x_3^m, \cdots, x_p^m]$, padronize os regressores da seguinte forma:

$$X_{nj}^m = \frac{x_{nj} - \bar{x}_j}{\sqrt{\sum (x_{nj} - \bar{x}_j)^2}}, \qquad \bar{x}_j = \frac{1}{n} \sum x_{nj} \tag{6.1}$$

   onde, $j = 2, 3, \cdots, p$. A dimensão de $X_m = n \times (p-1)$. Calcule o determinante de $X'_m X_m$, onde se $|X'_m X_m| = 0$, há multicolinearidade exata, e se $|X'_m X_m| = 1$, as colunas de X são ortogonais. Quanto mais próximo de 0 mais severo o problema.

2. Fatores de inflação de variância (vif): observando os elementos diagonais de $(X'_m X_m)^{-1}$, estes elementos são chamados de fatores de inflação de variância. Se um **vif** for maoir que 5, isto é indicativo de multicolinearidade, no R usamos a função **vif()** do pacote **car**

3. teste de Farrar e Glauber:
   Considere as seguintes hipóteses, $H0$: Há ausência de multicolinearidade, versus a hipótese $Ha$: caso contrário, com a estatística de teste quiquadrado definido pela seguinte fórmula:

$$X^2 = -[n - 1 - 1/6 * (2p + 5)] \ln det(mcor(X)) \tag{6.2}$$

   onde $mcor(X)$, e a matriz de correlação das variáveis independentes.

### 6.1.5 Exemplo

Considere os dados de consumo de água de uma residência durante o período dos meses de 12/2005 a 04/2011, fornecidos em formato txt (consumoagua2.txt) salvo na pasta padrão do R. Usando a função **vif()**

```
> ca=read.table("consumoagua2.txt", header=TRUE)
> attach(ca)
> m1=lm(valor~consumo+diasconsumo+Construindo-1)
> vif(m1)
    consumo diasconsumo Construindo
  22.100045   18.091433    2.102514
```

Podemos observar que há presença de multicolinearidade ($vif > 5$), para contornar eliminaremos a variável consumo

```
>m6=lm(valor~diasconsumo+Construindo)
> vif(m6)
diasconsumo Construindo
    1.01605     1.01605
```

Podemos observar que com o modelo m6 não há presença de multicolinearidade ($vif < 5$)
Teste da presença de alta multicolinearidade de Farrar e Glauber. Inicialmente determinamos a matriz de correlações dos regressores.

```
# mm matriz de variáveis independentes
> mm=cbind(consumo,diasconsumo,Construindo)
> cor(mm)

              consumo diasconsumo Construindo
consumo     1.0000000   0.2496357   0.5968966
diasconsumo 0.2496357   1.0000000   0.1256844
Construindo 0.5968966   0.1256844   1.0000000
```

Posteriormente aplicamos o teste:

```
> x2=-(65-1-(1/(6*11)))*log(det(cor(mm)))
> x2
[1] 32.36007
> xt=qchisq(0.95,3)
> xt
[1] 7.814728
```

O valor observado x2= 32.36, é maior que o valor tabelado xt= 7.815, concluem-se que há presença forte de multicolinearidade. Em consequência a variável consumo sai da analise, o modelo que elimina a multicolinearidade é:

```
> m5=lm(valor~diasconsumo+Construindo-1)
> mm2=cbind(diasconsumo,Construindo)
> cor(mm2)
            diasconsumo Construindo
diasconsumo   1.0000000   0.1256844
Construindo   0.1256844   1.0000000
> x2=-(65-1-(1/(6*9)))*log(det(cor(mm2)))
> x2
[1] 1.018756
> xt=qchisq(0.95,1)
> xt
[1] 3.841459
```

Como o valor observado x2= 1.019, é menor que o valor tabelado xt= $3,84$, concluem-se que a presença de multicolinearidade não é significativa. Em consequência as duas variáveis dias consumo e Construindo, compõem as variáveis independentes. Cabe lembrar que nosso interesse é melhorar a significância das estimativas do parâmetros. Quando a presença de multicolinearidade é quase exata, podemos contornar-la com a obtenção de mas dados, uma outra alternativa é usando a regressão ridge.

### 6.1.6 Regressão ridge

Os estimadores de mínimos quadrados são não viciados na presença de multicolinearidade e as previsões podem ser melhoradas usando um método chamado de regressão ridge. A ideia é que introduzindo um viés nas estimativas dos parâmetros do modelo podemos encontrar uma matriz de variâncias e covariâncias menor que os estimados por MQO. A regressão ridge é uma família de estimadores dada por:

$$\hat{\beta}_\lambda = (X'X + \lambda I)^{-1}X'y, \qquad \lambda > 0 \tag{6.3}$$

Note que se $\lambda = 0$, obtemos o EMQO, se $\lambda > 0$, e não estocástico, então temos:

$$E(\hat{\beta}_\lambda) = E((X'X + \lambda I)^{-1}X'y) = (X'X)^{-1}X'X\beta \neq \beta \tag{6.4}$$

Por tanto o estimaor ridge é viesado. Para determinar o tamanho do vies calculamos:

$$E(\hat{\beta}_\lambda) = \beta - [\lambda^{-1}(X'X') + I]^{-1}\beta \tag{6.5}$$

Por tanto, $VIES(\hat{\beta}_\lambda = -\lambda[\lambda I + (X'X)]^{-1}\beta$. Para o cálculo de este estimador usamos as propriedade de descomposição de matrizes. Como estamos introduzindo um viés nas estimativas dos parâmetros, podemos Encontrar, que $V(\hat{\beta}_j) > V(\hat{\beta}_j(\lambda))$, (uma medida para confirmar o ganho com este estimador é usar o mape, ver dethalhes no capítulo de séries temporais). O estimador ridge é uma transformação linear do EMQO, para corrigir a multicolinearidade, onde $\lambda$ é um parâmetro de encolhimento. Para exemplificar

```
>library(MASS)
> rr=lm.ridge(m1,lambda = seq(0,0.1,0.01))
> rr
> m1
> plot(lm.ridge(m1,lambda = seq(0,0.1,0.01)))
> rr1=lm.ridge(m1,lambda =0.1)
>rr1
    consumo diasconsumo Construindo
  2.6489068  -0.2905588   4.1726722
> rrc=2.6489068*consumo-0.2905588*diasconsumo+4.1726722*Construindo
> rr2=lm.ridge(m1,lambda =0.2)
    consumo diasconsumo Construindo
  2.5516146  -0.2539091   4.6047813
```

```
>rr2
> rr2c=2.5516146*consumo-0.2539091*diasconsumo+4.6047813*Construindo
> f1=fitted(m1)
> f=cbind(f1,rrc,rr2c)
> mape=abs((valor-f)/f)*100
> apply(mape,2,mean)
      f1      rrc     rr2c
13.47194 12.71836 12.24492
```

Para a escolha do valor ideal de $\lambda$, podemos usar o estimador proposto por Hoel, Kennard e Baldwin:

$$\hat{\lambda} = \frac{(\lambda - 1)\hat{\sigma}_m^2}{\hat{\beta}'\hat{\beta}} \tag{6.6}$$

onde:

$$\hat{\sigma}_m^2 = \frac{(y_m - X_m\hat{\beta}_m)'(y_m - X_m\hat{\beta}_m)}{n - p} \tag{6.7}$$

$$\hat{\beta}_m = (X'_m X_m)^{-1} X'_m y_m \tag{6.8}$$

## 6.2 Heterocedasticidade

Homoscedasticidade é a variância constante dos resíduos. Esta é uma propriedade fundamental, que deve ser garantida, sob pena de invalidar toda a análise estatística nos modelos de regressão. Deseja-se que os erros sejam aleatórios, se isto não ocorre, há heterocedasticidade. Significa dizer que ha chances de ocorrerem erros grandes (ou pequenos). Há tendências nos erros. Por exemplo, se na avaliação de terrenos a equação obtida indica erros maiores para os imóveis mais caros, progressivamente (quanto maior o imóvel, maior o erro), não há variância constante.
As consequências da heterocedasticidade são que as estimativas dos parâmetros da regressão os $\hat{\beta}_i$ não são tendenciosas (não viesados), continua sendo consistente, mas o teorema de Gaus Markov deixa de valer ou seja as estimativas dos parâmetros são ineficientes e suas respectivas variâncias são tendenciosas. Os testes $t$ e $F$ tendem a dar resultados incorretos. Neste caso, os resultados não são confiáveis, ou seja, o modelo pode parecer bom, mas ele não é adequado aos dados, na verdade. A heterocedasticidade inicialmente pode ser verificada através de gráficos de resíduos. Os gráficos dos resíduos contra os valores reais e contra os valores calculados pela equação são importantes. Se os pontos estão distribuídos aleatoriamente, sem mostrar um comportamento definido, há homoscedasticidade. Mas se existe alguma tendência (crescimento/decrescimento/oscilação), então há heterocedasticidade. Havendo heterocedasticidade, podem ser tentadas transformações nas variáveis (geralmente logarítmicas) ou outras soluções mais complexas. O modelo deve ser

modificado. No modelo heterocedástico assumimos que a matriz de variâncias e covariâncias dos erros é dada por:

$$\Phi = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_2^2 & \cdots & 0 & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \cdots & 0 & \sigma_n^2 \end{bmatrix}$$

onde $\Phi = dig(\sigma_1^2, \sigma_2^2, \cdots, \sigma_n^2)$ A covariancia dos parâmetros estimados é dado por:

$$Cov(\hat{\beta}) = (X'X)^{-1}X'\Phi X(X'X)^{-1} \tag{6.9}$$

Para exemplificar situações onde a presença de heterocedasticidade é comum, considere os índices de cotação da Bovespa ($\times 1000$) em formato "ibovm.txt", durante os períodos de $2001-2010$, disponibilizados pela média mensal a seguir:

```
 2001  2002  2003  2004  2005  2006  2007  2008  2009  2010
17.12 13.40 11.49 23.41 24.43 36.19 43.44 59.65 39.57 68.58
16.60 13.23 10.35 21.97 26.55 37.55 45.16 62.54 40.17 65.94
15.33 13.90 10.92 21.92 27.67 37.77 44.00 61.54 39.48 69.07
14.43 13.34 12.09 21.81 25.51 39.19 48.05 64.24 45.22 69.74
14.69 12.52 13.09 18.88 24.81 39.04 51.23 71.21 50.89 62.58
14.87 11.69 13.49 20.22 25.43 35.07 53.65 67.23 52.06 63.33
13.95 10.31 13.55 21.74 25.25 36.30 56.20 59.77 52.07 64.14
13.41  9.79 14.01 22.27 27.01 36.92 52.16 55.46 56.66 66.58
11.01  9.64 16.09 22.70 29.86 36.17 56.36 50.59 59.20 67.79
10.96  9.18 17.78 23.37 29.84 38.63 62.68 38.14 63.99 70.62
12.80 10.03 19.01 24.05 31.15 41.20 62.45 35.91 66.00 70.38
13.32 10.84 21.16 25.54 33.13 43.32 63.47 37.56 68.10 68.55
```

No R os dados são disponibilizados seguindo os seguintes comandos:

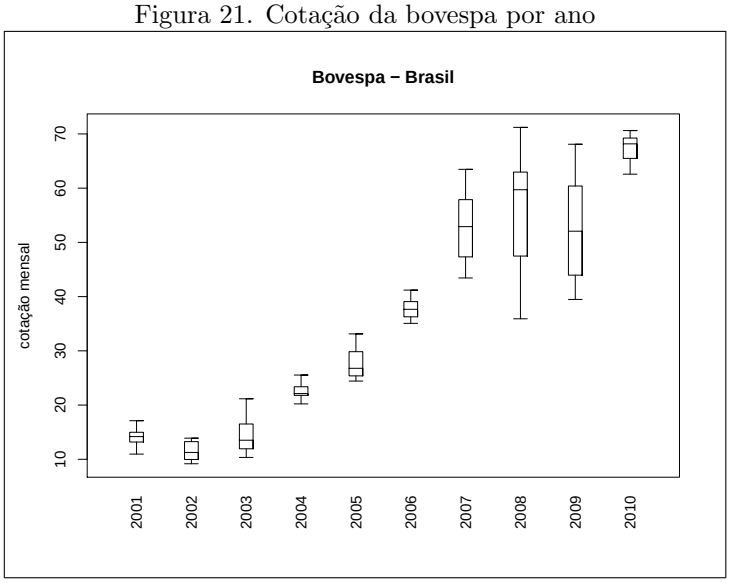
```
> ibov=read.table("ibovm.txt",header=T)
> colnames(ibov)=c("2001","2002","2003","2004","2005","2006","2007",
  "2008","2009","2010")
> ibovm=apply(ibov,2,mean)
> xi=c(1:10)
> mi=lm(ibovm~xi-1)
> summary(mi)

Call:
lm(formula = ibovm ~ xi - 1)

Residuals:
   Min     1Q Median     3Q    Max
-5.635 -4.587 -1.163  3.135  7.800
```

```
Coefficients:
   Estimate Std. Error t value Pr(>|t|)
xi    6.491      0.259   25.06 1.23e-09 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 5.082 on 9 degrees of freedom
Multiple R-squared: 0.9859,     Adjusted R-squared: 0.9843
F-statistic: 628.1 on 1 and 9 DF,  p-value: 1.232e-09

> anova(mi)
Analysis of Variance Table

Response: ibovm
          Df  Sum Sq Mean Sq F value    Pr(>F)
xi         1 16221.3 16221.3  628.08 1.232e-09 ***
Residuals  9   232.4    25.8
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

Na saída acima, considerando as médias anuais determinamos um modelo de regressão adequado, porem considerando os meses da cotação, pode negar o pressuposto da variância constante, um boxplot, monstra visualmente a presença de heterocedasticidade.

```
> bplot(ibov2,ylab="cotação mensal", main="Bovespa - Brasil")
```

Figura 21. Cotação da bovespa por ano



### 6.2.1 Testes de Heterocedasticidade

Na literaturá á vários testes que identificam a presença de heterocedasticidade, nesta unidade, mostraremos apenas aqueles que são disponibilizados pelo pacote **lmtest** do R.

#### Teste de Quandt-Goldfeld

Os passos deste teste são os seguintes:

1. $H0 : \sigma_1^2 = \sigma_2^2 = \cdots = \sigma_n^2$
   $Ha : \sigma_1^2 \neq \sigma_2^2, ou, \sigma_1^2 \neq \sigma_3^2 \cdots$, pelo menos uma desigualdade estrita.
2. Ordenar os dados segundo a magnitude de X.
3. Eliminar $c$ observações amostrais, em geral $c = 1/4$ dos dados. O teste fica mais poderoso
4. Estimam-se as duas regressões separadas pelas $c$ observações.
5. A estatística de prova é a $F_0$, com a $SQRes_1/SQRes_2$.
6. Os graus de liberdade é $[(n-c)/2-p-1]/[(n-c)/2-p-1]$, p é o número de parâmetros
7. Se o $F_0$ é próximo de 1, não há presença de Heterocedasticidade.
8. Se o $F_0$ se afastar de 1, a presença de heterocedasticidade é seria.

Onde $SQRes$, representa a soma de quadrados do resíduo.
Para implementar o teste, considere a função **gqtest()** do pacote **lmtest**, aplicando no modelo **mi**, onde:

```
> library(lmtest)
> gqtest(mi)

        Goldfeld-Quandt test

data:  mi
GQ = 1.3331, df1 = 4, df2 = 4, p-value = 0.3936
```

Pelo teste de Quandt-Goldfeld, pode-se afirmar que para dados anuais do Bovespa, não há evidencias para a presença de heterocedasticidade.
Este mesmo teste é aplicado para a cotação diária da Bovespa, (a cotação foi extraída do IPEA-DATA), os resultados são dados a seguir:

```
> bov=read.table("bovespa.txt",header=T)
> dim(bov)
[1] 2703    1
```

```
> dias=c(1:2703)
> bov=cbind(dias,bov)
> attach(bov)
The following object(s) are masked _by_ '.GlobalEnv':

    dias
> md=lm(cotacao~dias)
> summary(md)

Call:
lm(formula = cotacao ~ dias)

Residuals:
   Min     1Q Median     3Q    Max
-23607  -6419  -1038   6218  23101

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 2236.0514   313.5662   7.131 1.27e-12 ***
dias          23.4455     0.2009 116.717  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 8149 on 2701 degrees of freedom
Multiple R-squared: 0.8345,     Adjusted R-squared: 0.8345
F-statistic: 1.362e+04 on 1 and 2701 DF,  p-value: < 2.2e-16

> gqtest(md)

        Goldfeld-Quandt test

data:  md
GQ = 4.5783, df1 = 1350, df2 = 1349, p-value < 2.2e-16
```

Por tanto há evidencias da presença de heterocedasticidade num nível mais desagregado (dias úteis).

### Teste de Breusch-Pagan

O teste anterior é exato, não envolvendo aproximação, no teste a seguir assumimos que:

$$\sigma_i^2 = h(\alpha_1 + \alpha_2 Z_{i2} + \cdots, \alpha_s Z_{is}) \tag{6.10}$$

onde $Z's$ são as variáveis que afetam as variâncias, $h$ é uma função duas vezes diferenciável. As hipóteses são:
$H0 : \alpha_1 = \alpha_2 = \cdots = \alpha_s$ (Homocedástico)

$Ha : min|\alpha_1|, |\alpha_2|, \cdots, |\alpha_s| > 0$ (Heterocedástico)
Estatística do teste
Seja $\hat{\epsilon}_i$, i-ésimo resíduo de MQO, obtenha $\tilde{\sigma}^2 = \frac{\sum \hat{\epsilon}_i^2}{n}$, considere

$$m_i = \hat{\epsilon}_i^2 - \tilde{\sigma}^2 \tag{6.11}$$

e regredindo $m$ em $Z$, obtenha a soma de quadrados do resíduo da regressão auxiliar ($SQR_A$), e faça $BP = (SQR_A)/2$, forma alternativa

$$BP = \frac{m'Z(Z'Z)^{-1}Z'm}{2\tilde{\sigma}^4} \tag{6.12}$$

Um problema é que a distribuição de $BP$ é desconhecida, uma aproximação sobre $H0$, é dada pela distribuição qui-quadrado ($\chi^2_{s-1}gl$)
Regra de decisão: Rejeita-se $H0$ se $BP > \chi^2_{\alpha,s-1}$. Este teste é aproximado e geral, onde não precisamos especificar $h$.

**Teste de Koenker's**

As hipóteses do teste anterior permanecem válidas, porem no teste anterior não funciona bem não violação de normalidade dos erros, por tanto uma modificação de padronização é feita no teste $BP$, da seguinte forma:

$$BP_m = n * R_A^2 \tag{6.13}$$

ou alternativamente

$$n\frac{m'Z(Z'Z)^{-1}Z'm}{m'm} \tag{6.14}$$

que tem aproximação da distribuição qui-quadrado($\chi^2_{s-1}gl$)

Implementando os dois testes nos dados diários da Bovespa, temos

```
> bptest(md,studentize=FALSE) # Teste BP

        Breusch-Pagan test

data:  md
BP = 14.9066, df = 1, p-value = 0.000113

> bptest(md,studentize=TRUE)  # Teste Koenker

        studentized Breusch-Pagan test

data:  md
BP = 19.8011, df = 1, p-value = 8.593e-06
```

Nos dois testes concluímos que não há evidencias para aceitar $H0$, em favor de aceitar que pelo menos há duas variâncias dos erros distintas. Alternativa: este teste pode ser usado o **ncv.test()** do pacote **car**.

**Harrison McCabe**

O teste Harrison-McCabe fracciona a soma de quadrados do resíduos a partir de um certo ponto, A hipótese nula é determinada pelo tamanho da fração que como padrão assume o valor 0.5, define-se, assim $H0 =$ a fração é 0.5, a hipótese nula é rejeitada se o valor for menor que 0.5. Implementando o teste no R com o pacote **lmtest** temos:

```
> hmctest(md)

        Harrison-McCabe test

data:  md
HMC = 0.42, p-value < 2.2e-16
```

### 6.2.2 Corrigindo a Heterocedasticidade

Intuitivamente quando X é pequeno, a variabilidade é pequena, indicando que as observações são muito informativas sobre a reta, e recebem peso alto na estimação de mínimos quadrados ponderados, quando X é grande a variabilidade é grande, então as observações são pouco informativas e elas recebem peso baixo no processo de estimação. Para corrigir a Heterocedasticidade conhecendo $\sigma^2$, podemos usar o método de mínimos quadrados ponderados, que no modelo de regressão simples é dado por:

$$\frac{y_i}{\sigma} = \frac{\beta_0}{\sigma} + \frac{\beta_1 X_1}{\sigma} + v_i \tag{6.15}$$

Nosso exemplo de regressão com os dados agregados anuais da Bovespa, podemos eliminar a heterocedasticidade, da seguinte forma

$$\frac{cotacao_i}{\sigma} = \frac{\beta_1 ano_i}{\sigma} + v_i \tag{6.16}$$

onde: $v_i = \epsilon_i/\sigma$

Implementado no R temos:

```
> lmh=lm(ibovm/ibovsd ~ xi/ibovsd -1)
> summary(lmh)

Call:
lm(formula = ibovm/ibovsd ~ xi/ibovsd - 1)

Residuals:
    Min      1Q  Median      3Q     Max
-2.5566 -1.9388  0.7331  2.0788  4.7476

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
```

```
xi          3.04248    0.25257  12.046 2.08e-06 ***
xi:ibovsd -0.22827    0.03353  -6.807 0.000137 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 2.624 on 8 degrees of freedom
Multiple R-squared: 0.9589,     Adjusted R-squared: 0.9486
F-statistic: 93.32 on 2 and 8 DF,  p-value: 2.854e-06
```

Para corrigir a presença de heterocedasticidade quando $\sigma^2$ é desconhecido, podemos admitir que $\sigma^2$ é diretamente proporcional com a variável regressora que ordena os dados (no caso os dias de cotação), assim podemos dividir na equação do modelo (md) os dias, para tal efeito propomos dois modelos, o primeiro modelo dividido pela raiz dos dias:

```
> cr=cotacao/sqrt(dias)
> dr=sqrt(dias)
> invb=1/sqrt(dias)
> lm1=lm(cr~dr+invb)
> summary(lm1)

Call:
lm(formula = cr ~ dr + invb)

Residuals:
     Min       1Q   Median       3Q      Max
-2024.03  -142.58   -27.03   130.38   646.89

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -1070.5233    18.9513  -56.49   <2e-16 ***
dr             37.7680     0.4338   87.07   <2e-16 ***
invb        19986.7852   128.9229  155.03   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 215.8 on 2700 degrees of freedom
Multiple R-squared: 0.8996,     Adjusted R-squared: 0.8995
F-statistic: 1.209e+04 on 2 and 2700 DF,  p-value: < 2.2e-16
```

O segundo modelo dividido pelos dias.

```
> cd=cotacao/dias
> invd=1/dias
> lm2=lm(cd~invd)
> summary(lm2)
```

```
Call:
lm(formula = cd ~ invd)

Residuals:
    Min      1Q  Median      3Q     Max
-411.99  -11.43    1.20   10.24  262.61

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.598e+00  3.154e-01   24.09   <2e-16 ***
invd        1.666e+04  1.279e+01 1302.94   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 16.26 on 2701 degrees of freedom
Multiple R-squared: 0.9984,     Adjusted R-squared: 0.9984
F-statistic: 1.698e+06 on 1 and 2701 DF,  p-value: < 2.2e-16
```

### 6.2.3 Estimador HC0 de White

Uma questão que precisamos resolver é encontrar um bom estimador para $Cov(\hat{\beta})$ na presença de heterocedasticidade. Sobre homoscedasticidade temos

$$Cov(\hat{\beta}) = \sigma(X'X)^{-1}$$

e por tanto

$$\widehat{Cov}(\hat{\beta}) = \hat{\sigma}(X'X)^{-1} \tag{6.17}$$

no caso de heterocedasticidade, é estimada por

$$\widehat{Cov}(\hat{\beta}) = \hat{\Psi}_0 = (X'X)^{-1}X'\hat{\Phi}_0X(X'X)^{-1} \tag{6.18}$$

Para estimar a consistência de $\Phi$, que tem como estimador $\hat{\Phi}_0$ dado por:

$$\hat{\Phi}_0 = \begin{bmatrix} \hat{\epsilon}_1^2 & 0 & \cdots & 0 & 0 \\ 0 & \hat{\epsilon}_2^2 & \cdots & 0 & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \cdots & 0 & \hat{\epsilon}_n^2 \end{bmatrix}$$

Este estimador é consistente sobre homoscedasticidade e heterocedasticidade de forma desconhecida, pois:

$$plim(\widehat{cov}(\hat{\beta})(\widehat{cov}(\hat{\beta})^{-1} = I_p, \qquad n \to \infty \tag{6.19}$$

Mesmo sem normalidade dos erros e sem homoscedasticidade, $\hat{\beta}_j \xrightarrow{d} N(\beta_j, V(\hat{\beta}_j))$, para $j = 1, 2, \cdots, p$, assim:

$$\frac{\hat{\beta}_j - \beta_j}{\sqrt{V(\hat{\beta}_j)}} \xrightarrow{d} N(0, 1) \tag{6.20}$$

O estimador $\hat{V}(\hat{\beta}_j)$ é consistente para $V(\hat{\beta}_j)$. Por tanto para contornar este situação usamos a estatística quasi-t, com a seguinte hipótese:
$H0 : \beta_j = \beta_j^0$, contra a hipótese alterativa de $Ha : \beta_j \neq \beta_j^0$. A estatística de prova é dada por:

$$\frac{\hat{\beta}_j - \beta_j^0}{\sqrt{\hat{V}(\hat{\beta}_j)}} \xrightarrow{d} N(0,1) \tag{6.21}$$

Onde: $\hat{V}(\hat{\beta}_j) = (X'X)^{-1}X'\hat{\Phi}_0X(X'X)^{-1}$

Um problema com este estimador é que tende a ser bastante viesado quando n não é grande, especialmente quando há presença de pontos de alavanca. Tende a subestimar as variâncias verdadeiras, e por tanto o teste torna-se liberal (não conservador), pois o tamanho real do teste tende a ser maior do que a probabilidade do erro tipo I, para corrigir esta deficiência surge o estimador HC1.

### 6.2.4 Estimador HC1 de Hinkler

Podemos construir consistentemente a matriz de covariâncias dos parâmetros estimados da seguinte forma:

$$\widehat{Cov}(\hat{\beta}) = \hat{\Psi}_1 = (X'X)^{-1}X'\hat{\Phi}_1X(X'X)^{-1} \tag{6.22}$$

Onde

$$\hat{\Phi}_1 = \frac{n}{n-p}diag(\hat{\epsilon}_1^2, \hat{\epsilon}_1^2, \cdots, \hat{\epsilon}_n^2) \tag{6.23}$$

este estimador no limite ($n \to \infty$) é igual ao HCO.

### 6.2.5 Estimador HC2 de Horn-Duncan

Este estimador é dado por:

$$\widehat{Cov}(\hat{\beta}) = \hat{\Psi}_2 = (X'X)^{-1}X'\hat{\Phi}_2X(X'X)^{-1} \tag{6.24}$$

Onde

$$\hat{\Phi}_2 = diag(\frac{\hat{\epsilon}_1^2}{1-h_1}, \frac{\hat{\epsilon}_1^2}{1-h_2}, \cdots, \frac{\hat{\epsilon}_n^2}{1-h_n}) \tag{6.25}$$

O estimador HC2 é não viesado sobre homoscedasticidade.

### 6.2.6 Estimador HC3 de Davidson-MacKinnon

Este estimador é uma aproximação de "Jackknife", dado por:

$$\widehat{Cov}(\hat{\beta}) = \hat{\Psi}_3 = (X'X)^{-1}X'\hat{\Phi}_3X(X'X)^{-1} \tag{6.26}$$

Onde

$$\hat{\Phi}_3 = diag(\frac{\hat{\epsilon}_1^2}{(1-h_1)^2}, \frac{\hat{\epsilon}_1^2}{(1-h_2)^2}, \cdots, \frac{\hat{\epsilon}_n^2}{(1-h_n)^2}) \tag{6.27}$$

Os testes que usam HC3, tendem a funcionar melhor que os que usam HC0.

### 6.2.7 Estimador HC4 de Cribari Neto

Este estimador não considera uma constante quadrática para a potencia dos valores influentes e sim um potencia nos valores influentes, as quais variam de observação para observação, dado por:

$$\widehat{Cov}(\hat{\beta}) = \hat{\Psi}_4 = (X'X)^{-1}X'\hat{\Phi}_4 X(X'X)^{-1} \tag{6.28}$$

Onde

$$\hat{\Phi}_4 = diag(\frac{\hat{\epsilon}_1^2}{(1-h_1)^{\delta_1}}, \frac{\hat{\epsilon}_1^2}{(1-h_2)^{\delta_2}}, \cdots, \frac{\hat{\epsilon}_n^2}{(1-h_n)^{\delta_n}}) \tag{6.29}$$

$$\delta_n = min(4, \frac{h_n}{\bar{h}}) = min(4, \frac{nh_n}{p}) \tag{6.30}$$

e

$$\bar{h} = \frac{\sum h_n}{n} = p/n \tag{6.31}$$

### 6.2.8 Estimador HC5 de Cribari-Souza

Uma alteração do HC4 é dado por:

$$\widehat{Cov}(\hat{\beta}) = \hat{\Psi}_5 = (X'X)^{-1}X'\hat{\Phi}_5 X(X'X)^{-1} \tag{6.32}$$

Onde

$$\hat{\Phi}_5 = diag(\frac{\hat{\epsilon}_1^2}{(1-h_1)^{\delta_1}}, \frac{\hat{\epsilon}_1^2}{(1-h_2)^{\delta_2}}, \cdots, \frac{\hat{\epsilon}_n^2}{(1-h_n)^{\delta_n}}) \tag{6.33}$$

$$\delta_n = \min(\frac{nh_n}{p}, \max(4, \frac{n\kappa h_{max}}{p})) \tag{6.34}$$

e $h_{max}$ : alavancagem máxima, $\kappa$ : constante entre 0 e 1 (Usa-se $\kappa = 0.7$ nas simulações). Para exemplificar estes estimadores consideremos o consumo de água em uma residência.

```
> library(sandwich)
> coeftest(m1,vcov=vcovHC(m1))
> vcov=vcovHC(m1)
> vcov
               consumo    diasconsumo Construindo
consumo      0.06220938  -0.024252971 -0.17343741
diasconsumo -0.02425297   0.009637151  0.06351919
Construindo -0.17343741   0.063519195  1.94060840
# Para incorporar a matriz de covariância usamos
> t(sapply(c("const","HC0","HC1","HC2","HC3","HC4"),
function(x) sqrt(diag(vcovHC(m1,type=x)))))
        consumo diasconsumo Construindo
const 0.1553605  0.05995221    1.228001
HC0   0.2178130  0.08653240    1.289346
HC1   0.2230204  0.08860119    1.320171
```

```
HC2   0.2328432  0.09206926    1.340030
HC3   0.2494181  0.09816899    1.393057
HC4   0.2764422  0.10747053    1.380511
```

## 6.3 Autocorrelação

O nosso objetivo é tenta dar respostas a natureza da autocorrelação, as consequências teóricas e praticas, se a autocorrelação esta relacionada com as perturbações não observáveis, como saber se há autocorrelação numa dada situação, e como corrigir o problema da autocorrelação. Tanto na presença de heterocedasticidade como de autocorrelação, os estimadores usuais de MQO, embora não viesados, consistentes e assintoticamente normal, já não possuem variância mínima (não é eficiente) entre todos os estimadores lineares não tendenciosos. O seja não são MELNT.

### 6.3.1 Natureza da autocorrelação

Um dos pressupostos do modelo de regressão linear clássico é:

$$E(\epsilon_i, \epsilon_j) = 0 \qquad \forall i \neq j \tag{6.35}$$

Como se pressupõe que a média de $E(\epsilon_i) = 0 = E(\epsilon_j)$, podemos escrever que a covariância, $cov(\epsilon_i, \epsilon_j) = 0$. Esta característica das perturbações de regressão é conhecida como não auto-regressão. Porem caso haja dependência nas perturbações, temos autocorrelação. Simbolicamente:

$$E(\epsilon_i, \epsilon_j) \neq 0 \tag{6.36}$$

Dizemos que os erros são esféricos quando $cov(\epsilon) = E(\epsilon\epsilon') = \sigma^2 I_n$, esta esfericidade pode ser violada por ter elementos na diagonal não constantes (heterocedasticidade) ou por elementos fora da diagonal diferente de zero, neste último caso dizemos que á presença de autocorrelação. Um exemplo dado em Gujarati (2000)[14], considera que si estivermos lidando com dados de corte transversal com regressão de consumo familiar sobre a renda familiar, não vamos esperar que o efeito no consumo de uma família, decorrente de um aumento em sua renda, vai afetar o consumo de uma outra família. No entanto se o acompanhamento de consumo familiar sobre a renda familiar de uma mesma família ao longo de uma série temporal (corte longitudinal, espera-se que no aumento em sua renda aumente também seu consumo).

#### Terminologia

Autocorrelação: Correlação defasada de uma dada série consigo mesma, defasada em um número de unidades de tempo.

Correlação serial: Correlação defasada entre duas séries diferentes. Assim uma correlação entre duas séries defasada em um período de tempo do tipo: $u_1, u_2, ..., u_{10}$, e $u_2, u_3, \ldots, u_{11}$ , é um exemplo de autocorrelação, enquanto uma correlação entre séries temporais do tipo: $u_1, u_2, ..., u_{10}$, e $v_2, v_3, ..., v_{11}$, em que $u$ e $v$ são duas séries temporais diferentes.
O presuposto de que no tempo as relações estimadas a partir de observações, envolvem perturbações auto-regressivas é tão comum e por tanto:

$$E(\epsilon_t, \epsilon_{t-s}) \neq 0 \qquad \forall t > s \tag{6.37}$$

Esta expressão implica que a perturbação que ocorre no tempo $t$ relaciona-se à perturbação que ocorre no tempo $t - s$.
As consequências da auto-regressão na estimação pressupõe que:

$$E(\epsilon_t, \epsilon_{t-s}) = cov(\epsilon_t, \epsilon_{t-s}) = \rho^s \sigma_\epsilon^2 \tag{6.38}$$

**Geração das Perturbações**

A questão que tem que ser respondida agora refere-se à maneira pela qual as perturbações são geradas de modo que elas se relacionam uma à outra. O modelo mais usado é geradas no seguinte esquema:

$$\epsilon_t = \rho\epsilon_{t-1} + \nu_t \qquad t = 1, 2, \ldots \tag{6.39}$$

Onde $\nu_t \approx N(0, \sigma_\nu^2)$ e pressupõe-se independente de $\epsilon_t$. Uma relação como na equação acima é conhecida como esquema auto-regressivo de primeira ordem. Implica que cada perturbação corrente seja igual a uma *porção* da perturbação precedente mais um efeito aleatório. Através de uma substituição, temos:
$\epsilon_t = \rho\epsilon_{t-1} + \nu_t$
$\epsilon_t = \rho(\rho\epsilon_{t-2} + \nu_{t-1}) + \nu_t$
...
$\epsilon_t = \rho^t\epsilon_0 + \rho^{t-1}\nu_1 + \ldots + \rho\nu_{t-1} + \nu_t$
Assumindo que

$$\epsilon_0 \approx N(0, \frac{\sigma_\nu^2}{1-\rho^2}) \tag{6.40}$$

temos

$$var(\epsilon_t) = \frac{\sigma_\nu^2}{1-\rho^2} \tag{6.41}$$

A correlação é por tanto:

$$corr(\epsilon_t, \epsilon_{t-1}) = \frac{cov(\epsilon_t, \epsilon_{t-1})}{\sqrt{var(\epsilon_t)var(\epsilon_{t-1})}} = \rho \tag{6.42}$$

$\rho$ é a correlação entre erros sucessivos, por tanto

$$cov(\epsilon_t, \epsilon_{t-k}) = \rho^k \sigma_\nu^2 \tag{6.43}$$

A matriz de covariâncias dos resíduos é

$$\Phi = E(\epsilon\epsilon') = \frac{\sigma_\nu^2}{1-\rho^2}\begin{bmatrix} 1 & \rho & \rho^2 & \cdots & \rho^{n-1} \\ \rho & 1 & \rho & \cdots & \rho^{n-2} \\ \vdots & & & & \vdots \\ \rho^{n-1} & \rho^{n-2} & \rho^{n-3} & \cdots & 1 \end{bmatrix}$$

e $\Phi = \sigma_\epsilon^2\Omega$ as correlações naturalmente decaem geometricamente a medida que os erros se afastam.

### 6.3.2 Diagnostico da Autocorrelação

**Teste de Durbin Watson**

Este teste é o mais comum na literatura, e serve para diagnosticar em que grau a autocorrelação de primeira ordem esta presente. As hipóteses são as seguintes:

1. H0: Não há presença de autocorrelação de primeira ordem ($d = 2$).
   Ha: Há presença de autocorrelação.

2. nível de significância: $\alpha = 0,05$

3. Estatística de Prova:
$$d = \frac{\sum_{t=2}^{n}(\hat{\epsilon}_t - \hat{\epsilon}_{t-1})^2}{\sum_{t=1}^{n}\hat{\epsilon}_t^2} \tag{6.44}$$

4. Regra de Decisão:
   Se $d \approx 2$, há ausência de autocorrelação;
   Se $d \approx 0$, há autocorrelação positiva;
   Se $d \approx 4$, há autocorrelação negativa;

Onde $\approx$ significa próximo do valor. Ver maiores detalhes em Gujarati(2000).

**Exemplo 1**

Considere a cotação diária da Bovespa, para determinar a presença de autocorrelação, temos:

```
> md=lm(cotacao~dias)
> dwtest(md)

        Durbin-Watson test
```

```
data:  md
DW = 0.008, p-value < 2.2e-16
alternative hypothesis: true autocorrelation is greater than 0
```

Conclusão, como o valor Durbin-Watson é próximo de zero e o p-valor menor que 0.05, concluímos que não há evidencias de aceitar H0, em favor de presença de autocorrelação positiva de primeira ordem.

**Exemplo 2**

Considere o consumo de água em uma residência, aplicando o teste de Durbin, temos:

```
> m1 = lm(valor ~ consumo + diasconsumo + Construindo - 1)
> dwtest(m1)
        Durbin-Watson test

data:  m1
DW = 1.4882, p-value = 0.01273
alternative hypothesis: true autocorrelation is greater than 0
```

**Teste assintótico**

Quando n é grande, podemos construir as hipóteses
$H0 : \rho = 0$, versus $Ha : \rho \neq 0$, onde:

$$\hat{\rho} \approx N(\rho, \frac{1-\rho^2}{n}) \tag{6.45}$$

Sobre $H0, \hat{\rho} \approx N(0, 1/n)$, com estatística de teste $\sqrt{n}\hat{\rho} \approx N(0, 1)$.
A regra de decisão: Rejeita-se $H0$ se $\sqrt{n}|\hat{\rho}| > z_{\alpha/2}$, onde, $z_{\alpha/2}$, é o quantil $1 - \alpha/2$ da distribuição normal padrão.
Para facilitar o entendimento de uma série temporal consideraremos daqui em diante $n = t$

**Teste de Durbin**

Quando $y_{t-1}$ é regressor (é estocástico), viola a pressuposição básica do modelo. Para regressar $y_{t-1}$ realizamos:

$$y_t = \gamma y_{t-1} + X_t'\beta + \epsilon_t, \qquad t = 1, 2, \cdots, T \tag{6.46}$$

Aqui $\epsilon_t = \rho\epsilon_{t-1} + \nu_t$, cujas hipóteses são:
$H0 : \rho = 0$, versus $Ha : \rho \neq 0$.
A estatística de teste é:

$$h = \hat{\rho}\sqrt{\frac{T}{1 - T(\hat{V}(\hat{\gamma}))}} \tag{6.47}$$

Assumindo que $T\hat{V}(\hat{\gamma}) < 1$, sobre $H0$, $h \xrightarrow{d} N(0,1)$. Quando T não é muito grande o teste não funciona muito bem.

### 6.3.3 Eliminação da Autocorrelação

Para eliminar a autocorrelação, onde $\epsilon_t$ é gerado por um processo autorregressivo de primeira ordem, podemos construir o seguinte modelo:

$$y_t - \rho y_{t-1} = \beta_0 * (1-\rho) + \beta_1(X_t - \rho X_{t-1}) + \nu_t \tag{6.48}$$

No exemplo da cotação da Bovespa, para a eliminação de autocorrelação:

```
> diast=dias[-1]
> diast1=dias[2703]
> cotacaot=cotacao[-1]
> cotacaot1=cotacao[-2703]
> diast1=dias[-2703]
> e=residuals(md)
> et=e[-1]
> et1=e[-2703]
> core=cor(et,et1)
> yt=cotacaot-core*cotacaot1
> xt=diast-core*diast1
> mea=lm(yt~xt-1)
> mea
> dwtest(mea)

        Durbin-Watson test

data:  mea
DW = 2.0505, p-value = 0.9054
alternative hypothesis: true autocorrelation is greater than 0
```

No exemplo acima com a transformação do modelo podemos concluir que não há evidencias de presença de autocorrelação.

Para o exemplo de consumo de água em uma residência, eliminamos:

```
> e=residuals(m1)
> e1=cor(e[-1],e[-65])
> wt=valor[-1]-e1*valor[-65]
> z1=consumo[-1]-e1*consumo[-65]
> z2=diasconsumo[-1]-e1*diasconsumo[-65]
> z3=Construindo[-1]-e1*Construindo[-65]
> msa=lm(wt~z1+z2+z3-1)
> dwtest(msa)
```

```
        Durbin-Watson test
data:  msa
DW = 2.1924, p-value = 0.7585
alternative hypothesis: true autocorrelation is greater than 0
```

O resultado do teste de Durbin-Watson acusa a ausência de autocorrelação, ou em outras palavras não há evidencias para rejeitar $H0$.

Uma alternativa simples, que em geral elimina a autocorrelação, é usar as primeiras diferenças, tanto nas variáveis endógenas quanto as exógenas.

## 6.4 Exercícios

1. Na sua opinião você prefere modelos com muitas variáveis regressoras (independentes), ou não, justifique sua resposta.

2. O que entende por parsimonia na construção de um modelo de regressão?

3. Pesquise a visão de Ballentine sobre a multicolinearidade.

4. Ao final, a multicolinearidade é um problema ou uma solução? justifique suas afirmações.

5. Determine as estimativas dos parâmetros na presença da multicolinearidade Perfeita

6. Determine as estimativas dos parâmetros na presença de multicolinearidade quasi-perfeita.

7. Construa um modelo polinomial (ou de potência) para os dados de consumo de água de uma residência, onde o valor pago esta em função do consumo num modelo cúbico. Diagnostique a presença ou não de multicolinearidade.

8. No exemplo anterior faça uma regressão passo passo para incluir qual o modelo de potência com o coeficiente de correlação quadrático alto sem presença de multicolinearidade.

9. Que acontece com a matriz de variâncias e covariâncias $cov(\hat{\beta})$ na presença de multicolinearidade perfeita.

10. Com os dados de consumo de água, identifique um modelo que supere todos os pressupostos básicos e comente.

11. Use os experimentos de Monte Carlo e construa modelos de regressão com 5 variáveis independentes com presença de multicolinearidade. O que acontece com a distribuição das estimativas dos parâmetros quando n é grande (maior que 1000000)

# Capítulo 7

# Estimadores

Em estatística, um estimador é uma função das observações usada para estimar um parâmetro ($\theta$) da população. Ao valor do estimador ($\hat{\theta}$) chama-se estimativa. O parâmetro é o verdadeiro valor de uma característica de interesse, que raramente é conhecido. Em modelos de regressão por exemplo estes estimadores podem ser os $\hat{\beta}_i, \hat{\epsilon}_i$, etc.

## 7.1 Propriedades de um Estimador

1. Estimativa é o valor numérico obtido pelo estimador $\hat{\theta}$ em uma amostra.
2. Definição de não viés. Um estimador é não viesado se: $E(\hat{\theta}) = \theta$, onde o viés é dado por: viés$(\hat{\theta}) = E(\hat{\theta} - \theta) = E(\hat{\theta}) - \theta$
3. Erro quadrático médio ($EQM$). É dada por: $EQM(\theta) = E(\hat{\theta} - \theta)^2 = V(\theta)+$viés$(\hat{\theta})^2$
4. Eficiência de um Estimador: $\hat{theta}$ é um estimador eficiente de $\theta$ se forem satisfeitas: $E(\hat{\theta}) = \theta$ e $V(\hat{\theta})?V(\tilde{\theta})$ , onde $\tilde{\theta}$ é qualquer outro estimador de $\theta$.
5. Consistência. Um estimado é consistente se: $plim(\hat{\theta}) = \theta$ e $\underbrace{lim}_{n\to\infty} EQM(\hat{\theta}) = 0$
   . Ou é consistente quando seu valor se aproxima do verdadeiro valor do parâmetro à medida que aumenta-se o tamanho da amostra. No R podemos construir algoritmos simples para identificar o viés de algum estimador dado a seguir:

```
#Exemplo considere 10 amostras de uma N(10,4), determine o viés pontual dessa amostra.
set.seed(20)
vies=mean(rnorm(10,10,4))-10
```

```
vies
[1] -1.449323
```

Para estender este conceito consideremos um algoritmo com simulação de Montecarlo para o comportamento do viés:

```
ns= seq(2, 1000,by=2)#inicia em 2 termina  em 1000 a cada 2
vies= numeric(length(ns))  #prepara 500 elementos (vetor de zeros)
for (i in 1:length(ns)){      #laço
amostra= rnorm(ns[i],10,4) #números aleatórios N(10,4)
vies[i]= mean(amostra)-10  #diferença: média da amostra com parâmetro
}
plot(ns, vies)        #gráfico do tamanho e a média da amostra
abline(h=0,col=2)
{
```

Deixamos como exercício aumentar o tamanho da amostra e verificar o desempenho do viés, comente em detalhe.
Vejamos como podemos ilustrar a Consistência usando simulação. A idéia básica é a seguinte:

```
#1. escolher uma distribuição e seus parâmetros,
#2. definir o estimador,
#3. definir uma sequência crescente de valores de tamanho de amostras,
#4. obter uma amostra de cada tamanho,
#5. calcular a estatística para cada amostra,
#6. fazer um gráfico dos valores das estimativas contra o tamanho de amostra, r

#Média da distribuição Normal com média 10 e variância 4, o estimador é a média

ns= c(2, seq(5, 1000, by=5), seq(1010, 5000, by=10))#2,201(5)400(10)
estim= numeric(length(ns))  #prepara 601 elementos (vetor de zeros)
for (i in 1:length(ns)){      #laço (de 1 a 601 fazer)
amostra= rnorm(ns[i], 10, 4) #números aleatórios N(10,4)
estim[i]= mean(amostra)      #aloca a média da amostra em estim
}
plot(ns, estim)        #gráfico do tamanho e a média da amostra
abline(h=10) # linha da média =10

#Tarefa refaça o algoritmo para amostras maiores e com N(20,5)
#Tarefa refaça o algoritmo para amostras maiores e com exp(.2)
```

Algoritmo com simulação de montecarlo do EQM para vários tamanhos de amostra e em populações normais:

```
ns= seq(6, 500,by=2)  #inicia em 2 termina  em 1000 a cada 2
```

```
eqm= numeric(length(ns))
var= numeric(length(ns))
vies= numeric(length(ns))  #prepara 498 elementos (vetor de zeros)
for (i in 1:length(ns)){      #laço
amostra= rnorm(ns[i],10,4) #números aleatórios N(10,4)
vies[i]= mean(amostra)-10  #diferença entre a média da amostra com o parâmetro
var[i]= var(amostra)/ns[i] #Variancia da média da amostra
eqm[i]=var[i]+(vies[i])^2  #EQM (erro quadrático médio)
 }
#visualização gráfica
par(mfrow=c(2,2))
plot(ns, eqm, type="l",ylim=c(-0.1,1),main="Erro Quadrático Médio")
#gráfico do tamanho e EQM
abline(h=0,col="gray")
plot(ns,vies,type="l", main="Vies")
abline(h=0,col="gray")
plot(ns,var,type="l", main="Variancia do estimador")
abline(h=0,col="gray")
plot(density(vies), main="Densidade do Vies")
```

# Capítulo 8

# Regressão não linear

Nos modelos lineares, a estimação dos parâmetros, cai no problema de resolver um sistema de equações lineares com relação aos coeficientes de regressão desconhecidos. Existe uma solução única e, portanto, obtemos uma forma analítica de estimação dos parâmetros. Esta forma é a mesma para qualquer modelo e qualquer conjunto de dados. Além disso, como os coeficientes são combinações lineares das observações, pela teoria estatística, demonstra-se que a distribuição amostral dos coeficientes estimados de regressão segue uma distribuição $t$ ou normal, assim, podemos realizar testes de hipóteses, calcular intervalos de confiança para esses coeficientes populacionais de regressão.

Quando falamos de modelos de regressão não linear estamos tratando de modelos que são não lineares nos parâmetros. Existem muitas situações nas quais não é desejável, ou mesmo possível, descrever um fenômeno através de um modelo de regressão linear. Ao invés de se fazer uma descrição puramente empírica do fenômeno em estudo, pode-se, a partir de suposições importantes sobre o problema (frequentemente dadas através de uma ou mais equações diferenciais), trabalhar no sentido de obter uma relação teórica entre as variáveis observáveis de interesse. O problema, diferentemente do caso linear, é que os parâmetros entram na equação de forma não linear, assim, nós não podemos simplesmente aplicar fórmulas para estimar os parâmetros do modelo. Em muitas situações, necessitam-se menos parâmetros nos modelos não lineares do que nos lineares, isto simplifica e facilita a interpretação.

Os modelos não lineares podem ser escritos como:

$$Y_i = f(X_i, \beta) + \epsilon_i \tag{8.1}$$

onde $f(X_i, \beta)$ é uma função não linear; os erros, $\epsilon_i$, tem média zero, variância constante, e não são correlacionados. Assume-se que os erros seguem distribuição normal, são independentes e têm variância constante. $\beta$ é o vetor de parâmetros do modelo. Para iniciar apresentamos três modelos não lineares:

## 8.1 Modelo exponencial

$$Y_i = \beta_0 \exp(\beta_1 X_i) + \epsilon_i \tag{8.2}$$

onde $\beta_0$ e $\beta_1$ são os parâmetros do modelo; $X_i$ são constantes conhecidas como variável preditora e $\epsilon_i$ são os termos do erro, independentes, com distribuição normal de média 0 (zero) e variância $\sigma^2$. Diferenciando $f$ com respeito aos parâmetros $\beta_0$ e $\beta_1$ obtemos:

$$\frac{\partial f}{\partial \beta_0} = \exp(\beta_1 X_i) \tag{8.3}$$

$$\frac{\partial f}{\partial \beta_1} = \beta_0 X_i \exp(\beta_1 X_i) \tag{8.4}$$

### 8.1.1 Medida de ajuste

Em modelos de regressão há varias medidas de qualidade de ajuste dos dados, tais como o coeficiente de correlação, coeficiente de determinação, critérios de informação (AIC, BIC), porem neste livro usaremos uma medida universal para ajustar os dados próximos dos seus valores reais, esta medida chama-se de erro absoluto médio percentual, e na literatura é definido por **mape**, o calculo é dada por:

$$mape = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i / y_i| * 100 \tag{8.5}$$

### 8.1.2 Exemplo

Considere as projeções a cada década do tamanho da população Brasileira (ver no apêndice a chamada dos dados), desde 1870 até 2010. Um modelo de regressão não linear da família exponencial é

```
> mpop.ex= glm(popbrasil ~ano, family=Gamma(link=log))
>summary(mpop.ex)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.952e+01  6.904e-01  -57.24   <2e-16 ***
ano          2.233e-02  3.558e-04   62.76   <2e-16 ***
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1  1

(Dispersion parameter for Gamma family taken to be 0.003544175)
    Null deviance: 12.735063  on 14  degrees of freedom
Residual deviance:  0.046475  on 13  degrees of freedom
AIC: 75.832
Number of Fisher Scoring iterations: 3
> mape1=(sum(abs((popbrasil-fitted(mpop.ex))/popbrasil))*100)/15
> mape1
[1] 4.366599
```

```
> plot(ano,popbrasil)
> lines(ano,fitted(mpop.ex),col=8)
```

Podemos afirmar que o modelo é adequado pelo teste F, os parâmetros são significativos pelo teste t para cada parâmetro, e as previsoes do modelo tem um bom ajuste, pois o erro de previsão em percentual (mape)é de 4.33

## 8.2 Modelo logístico

$$Y_i = \frac{\beta_0}{1 + \beta_1 \exp(\beta_2 X_i)} + \epsilon_i \tag{8.6}$$

da mesma forma que no modelo anterior, diferenciando obtemos:

$$\frac{\partial f}{\partial \beta_0} = \frac{1}{1 + \beta_1 \exp(\beta_2 X_i)} \tag{8.7}$$

$$\frac{\partial f}{\partial \beta_1} = \frac{\exp(\beta_2 X_i)}{(1 + \beta_1 \exp(\beta_2 X_i))^2} \tag{8.8}$$

$$\frac{\partial f}{\partial \beta_2} = \frac{\beta_1 \exp(\beta_2 X_i X_i)}{(1 + \beta_1 \exp(\beta_2 X_i))^2} \tag{8.9}$$

Para ajustar um modelo logístico para as projeções do tamanho da população do Brasil (décadas), precisamos de valores iniciais para os parâmetros, isto é rotineiro quando usamos a estimação de modelos não lineares, porem a eleição destes valores iniciais devem seguir alguns critérios e ser razoavelmente próximos dos seus verdadeiros valores, assim calcularemos os valores iniciais para a estimativa dos parâmetros (Os $\beta_i's$). Para modelo crescentes, utilizamos de modo geral que $\beta_1 ini$ é maior que a última projeção da população Brasileira, assim sera maior que 190, por tanto um valor inicial será $\beta_1 = 220$. Para estimar o valor inicial de $\beta_2 ini$, usamos a primeira projeção estimada em milhões de habitantes do Brasil (9.533659) então substituímos em nosso modelo inicial da seguinte forma:

$$9.533659 = \frac{220}{1 + \exp^{\beta_2 + \beta_3 0}} \tag{8.10}$$

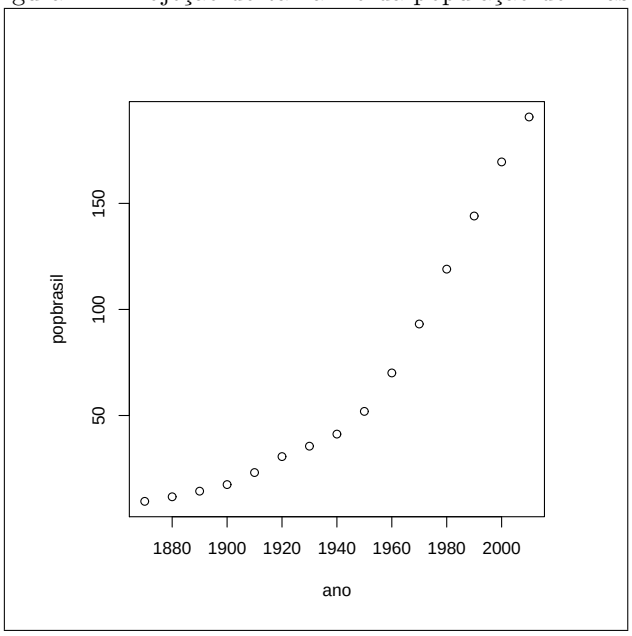
Resolvendo esta equação temos que $\beta_2 ini \cong 3.1$. Finalmente o valor inicial de $\beta_3 ini$, pode ser calculado usando a segunda projeção da população, dada a seguir

$$11.689938 = \frac{220}{1 + \exp^{3.1 + \beta_3}} \tag{8.11}$$

A solução desta equação estima o valor inicial de $\beta_3 ini \cong -0.22$. Com estes valores iniciais, usamos a função **nls()** do pacote **nls2**, da seguinte forma:

```
library(nls2)
tempo= 0:14
pop.b <- nls(popbrasil ~ beta1/(1 + exp(beta2 + beta3*tempo)),
start=list(beta1 = 220, beta2 = 3.1, beta3 = -0.22), trace=T)
```

Figura 22. Projeção do tamanho da população do Brasil




A saída dos resultados é dado na plataforma R.

```
pop.b <- nls(popbrasil ~ beta1/(1 + exp(beta2 + beta3*tempo)),
start=list(beta1 = 220, beta2 = 3.1, beta3 = -0.22), trace=T)
 18689.5 :  220.0000000   3.1000000  -0.2200000
12062.99 :  372.1359626   3.9467894  -0.2394392
229.5459 :  333.6289610   3.9016451  -0.3001283
217.1894 :  380.9831173   4.0275197  -0.2880894
179.4752 :  384.8339780   4.0395886  -0.2901861
179.4711 :  384.0791519   4.0396609  -0.2904526
179.4711 :  384.0546846   4.0397399  -0.2904686
179.4711 :  384.0494772   4.0397417  -0.2904707
```

As estimativas dos parâmetros são dados na saída do comando **summary()**.

```
> summary(pop.b)

Formula: popbrasil ~ beta1/(1 + exp(beta2 + beta3 * tempo))

Parameters:
       Estimate Std. Error t value Pr(>|t|)
beta1 384.04948   60.14841   6.385 3.48e-05 ***
beta2   4.03974    0.09384  43.049 1.60e-14 ***
beta3  -0.29047    0.01913 -15.180 3.39e-09 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

```
Residual standard error: 3.867 on 12 degrees of freedom

Number of iterations to convergence: 7
Achieved convergence tolerance: 2.861e-06
```

Podemos concluir que os três parâmetros são estatisticamente significativos.

## 8.3 Modelo alternativo

Um modelo linear generalizado usando **glm()** é proposto para estimar o tamanho da população e comparar-lo com o modelo *logístico* da seção anterior.

```
mpop.b <- glm(popbrasil ~ano, family=gaussian(link=log))
mpop.b
mape1=(sum(abs((popbrasil-fitted(mpop.ex))/popbrasil))*100)/15
mape1
[1] 4.366599
mape2=(sum(abs((popbrasil-fitted(pop.b))/popbrasil))*100)/15
mape2
[1] 9.872469
mape3=(sum(abs((popbrasil-fitted(mpop.b))/popbrasil))*100)/15
mape3
9.89976
```

O modelo alternativo, tem um erro médio absoluto percentual maior que o modelo logístico e exponencial. Para comparar o tamanho da população projetada e as estimativas dos dois modelos ( exponencial e alternativo), podemos fazer

```
plot(ano, popbrasil)
lines(ano,fitted(pop.b))
lines(ano,fitted(mpop.ex), lty=4)
legend("topleft", legend=c("Logistico","Exponencial"),lty=1:4)
```

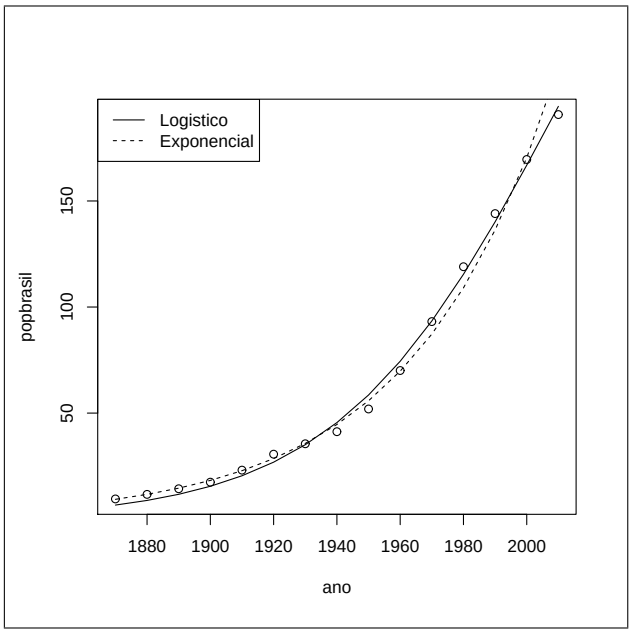



Figura 23. Dois modelos para ajustar o tamanho da população

Embora o valor do *mape* seja menor do modelo exponencial, isto não significa que seja melhor que o modelo proposto, para explicar este fato podemos observar que este modelo tem uma aceleração maior que a taxa de crescimento populacional real e que a taxa de crescimento do modelo proposto. Veja a comparação do ajuste dos modelos no gráfico acima. Observamos que o modelo exponencial ajusta-se bem ao tamanho de população, porem no final da série (última década) acelera o crescimento populacional, o que implica que suas previsões futuras não refletem o verdadeiro crescimento populacional, como podemos observar na previsão com seus respectivos erros de previssãopara 2015 e 2020 para os dois modelos.

```
novo=data.frame(tempo=c(14.5,15))
predict(pop.b, novo, se.fit = TRUE)
[1] 208.5062 222.2381
novo2=data.frame(ano=c(2015,2020))
p = exp(predict(mpop.ex, novo, interval="'confidence"'))
p
       1        2
238.1832 266.3136
```

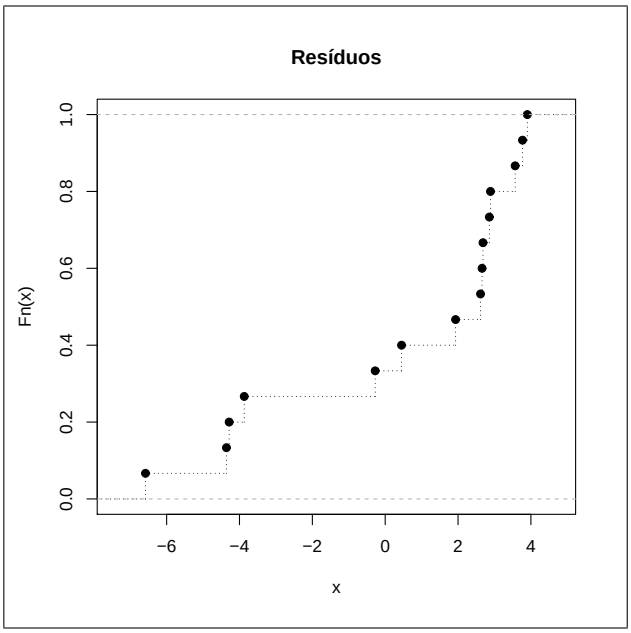



Figura 24. Função acumulada dos resíduos do modelo logistico

Podemos observar que as previsões do tamanho da população para os anos de 2015 e 2020, são conservadoras para o modelo proposto, e são não conservadores para o modelo exponencial. A distribuição acumulada pode ser observada no gráfico acima e pode ser escrito como:

```
plot(ecdf(residuals(pop.b)),verticals=T,lty=3, main="Resíduos")
```

## 8.4 Regressão sobre variáveis dummies

A introdução de variáveis qualitativas, frequentemente de variáveis dummies, torna o modelo de regressão linear uma ferramenta extremamente flexível, capaz de lidar com muitos problemas encontrados em estudos empíricos.

### 8.4.1 Natureza das variáveis dummies.

Na analise de regressão, a variável dependente é muitas vezes influenciada, não somente pelas variáveis que podem ser quantificadas em alguma escala bem definida (como: renda, produtos, preços, custos, altura e temperatura), mas também por variáveis de natureza essencialmente qualitativas ( como: raça, sexo, cor, religião, nacionalidade, guerras terremotos, greves, mudanças na política de governo). Por exemplo, mantendo constante todos os demais fatores verifica-se que professoras universitárias ganham menos que seus colegas homens, e que os não brancos ganham menos que os brancos. Este padrão pode resultar da discriminação sexual ou racial, mas que qualquer que seja a

razão, variáveis qualitativas como sexo ou raça de fato influencia a variável dependente, e claramente devem ser incluídas nas variáveis explicativas. Como tais variáveis qualitativas geralmente indicam a presença ou ausência de uma qualidade ou atributo, tais como homem e mulher, negro ou branco, católico e não católico, um método para quantificar tais atributos e construir variáveis artificiais que assumam valores com valores: 1 ou 0, onde 0 indica ausência de um atributo e 1 indica a presença de um atributo, por exemplo, 1 - homem, 0 - mulher; ou 1 - formação superior e 0 - indica o contrario, estas variáveis são chamadas de variáveis dummies. São nomes alternativos: variáveis indicadoras, variáveis binárias, variáveis categóricas, variáveis dicotômicas. As variáveis dummies são usadas nos modelos de regressão tão facilmente quanto ás variáveis quantitativas, alias um modelo de regressão pode conter variáveis explicativas que são exclusivamente dummies, por natureza tais modelos são chamados de modelos de analise de variância (ANOVA). Como exemplo considere os dados em Gujarati (2000), com o seguinte modelo:

$$y_i = \alpha + \beta D_i + \epsilon_i \tag{8.12}$$

Onde $Yi =$ Salário anual de um professor universitário.
$Di = 0$, sexo feminino; 1: sexo masculino.

O modelo acima permite verificar se o sexo provoca alguma diferença no salário de um professor universitário, supondo naturalmente que idade, título acadêmico, e anos de experiência são mantidos constantes assim,
Salário médio da professoras universitárias: $E(Y|Di = 0) = \alpha$
Salário médio dos professores universitários: $E(Y|Di = 0) = \alpha + \beta$
Onde $\alpha$ indica o termo intercepto como salário médio de professoras, e $\beta$ informa enquanto o salário de um professor difere da sua colega.

**Os dados no R**

```
> D=c(0,0,0,0,0,1,1,1,1,1)
> y = c( 19, 18, 18.5, 17, 17.5, 22, 21.7, 21, 20.5, 21.2)
> regdummi=lm(y~D)
> summary(regdummi)
> plot(y, main="salário do professor por sexo")
> abline(a=21,28, b=0, col=8)
> abline(a=18, b=0 )
> legend(2,22,legend="homem")
> legend(8,17.5,legend="mulher")
```

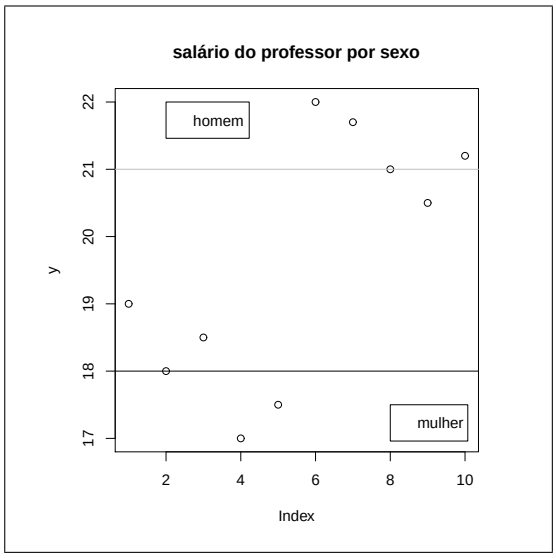



Figura 25. Comparação de salários para os professores

Interpretação: A salário médio de uma professora é de 18 unidades monetárias ($\alpha$) e o salário de um professor é de 21 unidades monetárias ($\alpha + \beta$)

### 8.4.2 Regressão sobre variáveis qualitativas e quantitativas

Considere o seguinte modelo:

$$y_i = \alpha_1 + \alpha_2 D_i + \beta x_i + \epsilon_i \tag{8.13}$$

Onde $Yi =$ Salário anual de um professor universitário.
$Di = 0$, sexo feminino; 1: sexo masculino.
$x_i =$ anos de experiência de ensino.
Admitindo $E(\epsilon_i) = 0$, podemos identificar os salários da seguinte forma.
Salário médio de uma professora universitária

$$E(y_i|x_i, D_i = 0) = \alpha_1 + \beta x_i \tag{8.14}$$

E o salário médio de um professor universitário

$$E(y_i|x_i, D_i = 1) = \alpha_1 + \alpha_2 + \beta x_i \tag{8.15}$$

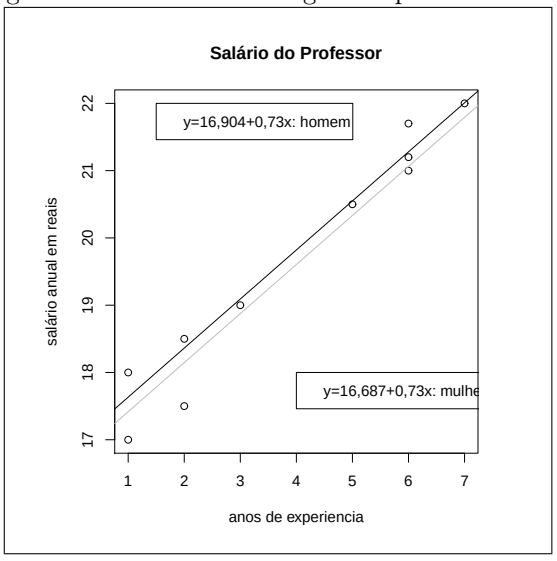
No R, acrescentando a os dados dos salários dos professores os anos de experiencia, temos:

```
> x=c(3,1,2,1,2,7,6,6,5,6)
> mrd=lm(y~D+x)
> summary(mrd)
```

```
Call:

lm(formula = y ~ D + x)

Residuals:

    Min      1Q  Median      3Q     Max
-0.6458 -0.2300 -0.0300  0.2969  0.5833

Coefficients:

            Estimate Std. Error t value Pr(>|t|)
(Intercept)  16.6875     0.4086  40.836 1.38e-09 ***
D             0.2175     0.8819   0.247   0.8123
x             0.7292     0.1994   3.657   0.0081 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.4369 on 7 degrees of freedom
Multiple R-squared: 0.9566,     Adjusted R-squared: 0.9442
F-statistic: 77.15 on 2 and 7 DF,  p-value: 1.703e-05
```

A interpretação geométrica deste modelo, é que as funções salário de professores universitários homens e mulheres, em relação aos anos de experiência de ensino, têm a mesma inclinação ($\beta$), porem diferentes interceptos. No R podemos construir o gráfico a seguir:

Figura 26. Duas retas de regressão para os salários



```
> plot(x,y, main="Salário do Professor", xlab="anos de experiencia",
> ylab="salário anual em reais")
> abline(a=16.6875,b=0.7292,col=8)
> abline(a=16.904,b=0.7292)
> legend(4,18,legend="y=16,687+0,73x: mulher")
> legend(1.5,22,legend="y=16,904+0,73x: homem")
```

**Teste de Chow**

Para testar a estabilidade estrutural dos modelos de regressão usamos a função **sctest()**, da biblioteca **strucchange**. Na suposição das variáveis não apenas afetam o intercepto, mais sim o coeficiente de inclinação, deve se testar se este coeficiente de inclinação apresenta diferença significativa nos subgrupos. No exemplo do salário dos professores, determinamos:

```
> library(strucchange)
Carregando pacotes exigidos: sandwich
> sctest(y~D+x,type="Chow", point=5)

        Chow test

data:  y ~ D + x
F = 0.0015, p-value = 0.9999
```

Pelo valor do p-valor, podemos afirmar que não há evidencias para rejeitar a hipótese de igualdade na inclinação dos salários (Não há mudanças no coeficiente de inclinação dos salários entre professoras e professores).

## 8.5 Modelos de regressão para respostas binarias

Modelar o comportamento de variáveis dependentes que só assumem valores 0 ou 1, através de uma análise de regressão, onde a $Pr(y = 0)$ ou $Pr(y = 1)$ é afetada pelos regressores. Consideremos a função de ligação denotada por $G(x, \beta)$, onde:
$Pr(y = 1|x) = G(x, \beta)$,
$Pr(y = 0|x) = 1 - G(x, \beta)$,
e $x$: vetor $p \times n$,
$\beta$:vetor $p \times 1$ de parâmetros.
Precisamos encontrar uma estrutura para regredir.

### 8.5.1 Modelos de probabilidade linear (MPL)

Nossa função de ligação para este modelo é $G(x, \beta) = x'\beta$, para este modelos temos que $E(y|x) = 1*G(x, \beta)+0*(1-G(x, \beta)) = G(x, \beta)$, cai em nosso modelo

linear de regressão, o qual apresenta os seguintes problemas:
$E(erro) = 0, V(erro) = x'\beta(1-x'\beta)$, por tanto o erro é heterocedástico, Não há restrições para $\epsilon = Pr(y = 1|x)$, podendo obter estimativas de $\epsilon$ fora de $[0, 1]$.

Uma outra estrutura para regredir é usar modelos para respostas binarias, onde $G(x, \beta)$ esta restrito ao intervalo inteiro padrão($[0, 1]$), onde

$$\lim x'\beta \to -\infty, \qquad Pr(y = 1|x) = 0 \tag{8.16}$$

$$\lim x'\beta \to \infty, \qquad Pr(y = 1|x) = 1 \tag{8.17}$$

Para a escolha de G, usamos os seguintes modelos:

### 8.5.2 Modelo Probit

A função de ligação G tem distribuição normal padrão,

$$Pr(y = 1|x) = \int_{-\infty}^{x'\beta} \phi(t)dt = \Phi(x'\beta) \tag{8.18}$$

Onde $\phi$ é a função de densidade da normal padrão, e $\Phi$ é a função de distribuição acumulada da normal padrão.

### 8.5.3 Modelo Logit

A ligação de G é a função de distribuição logística, dada por:

$$Pr(y = 1|x) = \frac{\exp x'\beta}{1 + \exp x'\beta} = \Lambda(x'\beta) \tag{8.19}$$

Modelos alternativos sem simetria são apresentados a seguir

## 8.6 Modelo Weibull

Usamos a função da distribuição:

$$Pr(y = 1|x) = \exp - \exp x'\beta \tag{8.20}$$

## 8.7 Modelo Log-log complementar

Usamos a função da distribuição:

$$Pr(y = 1|x) = 1 - \exp - \exp -x'\beta \tag{8.21}$$

Dentre os modelos apresentados acima o mais usado é o logit, a distribuição logit e probit são semelhante na parte central, já nas caudas o modelo logit fornece maiores probabilidades de $y = 0$, quando $x'\beta$ é muito pequeno, e $y = 1$, quando $x'\beta$ é muito grande.

**Estimação**

A estimação dos parâmetros do modelo logit é feita por máxima verosimilhança, consideramos cada observação como um ensaio de Bernoulli, onde:

$$Pr(Y_1 = y_1, Y_2 = y_2, \cdots, Y_t = y_t) = \prod_{y_t=0} [1 - G(X'\beta)] \prod_{y_t=1} [G(x'\beta)] \quad (8.22)$$

A função de verosimilhança é

$$L(\beta|y, x) = \prod_{t=1}^{T} [1 - G(X'\beta)]^{1-y_t} \prod [G(x'\beta)]^{y_t} \quad (8.23)$$

Onde a função de log-verosimilhança é dada por:

$$l(\beta|x, y) = \log L = \sum^{T} (1 - y_t) * \log[1 - G(X'\beta)] + y_t * [G(x'\beta)] \quad (8.24)$$

Apresentamos um tabela para algumas distribuição de probabilidade com sus respectivas funções de ligação .

| Distribuição | Lig. Canônica | Função de variância |
|---|---|---|
| Normal | $\mu$ | 1 |
| Poisson | $\log(\mu)$ | $\mu$ |
| Binomial | $\log \frac{\mu}{1-\mu}$ | $\mu(1-\mu)$ |
| Gamma | $\mu^{-1}$ | $\mu^2$ |
| Gausiana inversa | $\mu^{-2}$ | $\mu^3$ |

Para implementar modelos de regressão para respostas binárias usamos a função **glm()**

### 8.7.1 Exemplo 1

Considere o conjunto de dados (dados1), referente $Xi$ = renda em milhares de dólares, Ni = número de famílias com renda $Xi$, e ni1 = número de famílias que possuem uma casa. (Pág 562. Gujarati)

```
> attach(dados1)
> dados1
    x  Ni   ni1
1   6   40    8
2   8   50   12
3  10   60   18
4  13   80   28
5  15  100   45
6  20   70   36
7  25   65   39
8  30   50   33
9  35   40   30
10 40   25   20
```

```
######################
Logit

> lgm=glm(formula=cbind(ni1, Ni - ni1)~x, family=binomial(logit))
> summary(lgm)

Call:   glm(formula = cbind(ni, Ni - ni1) ~ x, family = binomial(logit))

Deviance Residuals:
     Min        1Q     Median        3Q       Max
-0.67026  -0.35105  -0.18558   0.06073   1.06817

Coefficients:
             estimate  Std. Error  z value  Pr(>|z|)
(Intercept) -1.60234    0.20403     -7.853  4.05e-15 ***
x            0.07907    0.01011      7.819  5.34e-15 ***

Null deviance: 72.7581  on 9  degrees of freedom
Residual deviance:  2.3542  on 8  degrees of freedom
AIC: 49.01
> fitlgm=fitted(lgm)
######################
Modelo Probit

> Pgm=glm(formula=cbind(ni1, Ni - ni1)~x, family=binomial(probit))
> summary(Pgm)
fitpgm=fitted(Pgm)
######################
Modelo cloglog

> cloglogm=glm(formula=cbind(ni1, Ni - ni1)~x,
family=binomial(cloglog))
> summary(cloglogm)
> fitcloglogm=fitted(cloglogm)

> fitclog=fitcloglogm*Ni
> fitlglm=fitlgm*Ni
> fitpglm=fitpgm*Ni
> comparacao=data.frame(ni1,fitlglm, fitpglm, fitclog)
> comparacao
   ni1   fitlglm     fitpglm    fitclog
1    8   9.781599    9.720856  10.65496
2   12  13.745853  13.721940  14.57061
3   18  18.451892  18.464347  19.09529
4   28  28.816154  28.858391  28.95371
5   45  39.738898  39.768847  39.32643
```

```
6   36  34.632031  34.542022  33.51850
7   39  38.512363  38.324448  37.21732
8   33  34.172042  34.031625  33.49879
9   30  30.489303  30.475689  30.57855
10  20  20.659865  20.757408  21.20739
```

### 8.7.2 Exemplo 2

Uma empresa esta buscando uma regra para discriminar a qualidade dos funcionários. Em seus arquivo de dados dispor de 109 registros com as seguintes informações dos funcionários $x1$: idade dos funcionários, $x2$: sexo, $x3$: anos de experiência na função, $x4$: estado civil, $x5$: número de filhos, $x6$: nota de 0 - 10 de um teste de avaliação de conhecimento na sua área, $x7$: teste psicotécnico de 0 a 50, $x8$: satisfação com a vida pessoal. $Y$, definido a priori como grupo1 (desempenho médio) e grupo 2 (melhor desempenho), 0 se faz parte do grupo 1 e 1 se faz parte do grupo 2. ver dados no apêndice.

```
> func=read.table("func.txt",header=T)
> attach(func)
> glm1=glm(grupo ~ tconhec +tpsico, family=binomial(logit))

> summary(glm1)

Call:

glm(formula = grupo ~ tconhec + tpsico, family = binomial(logit))

Deviance Residuals:

    Min       1Q   Median       3Q      Max
-3.1987  -0.0997   0.0130   0.1078   1.3017

Coefficients:

            Estimate Std. Error z value Pr(>|z|)
(Intercept) -44.2310    11.2683  -3.925 8.66e-05 ***
tconhec      -0.4924     0.2662  -1.850 0.064336 .
tpsico        7.7609     2.1575   3.597 0.000322 ***

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 149.552  on 108  degrees of freedom
Residual deviance:  38.339  on 106  degrees of freedom
AIC: 44.339

Number of Fisher Scoring iterations: 8
```

```
> fit=fitted(glm1)
> com=data.frame(grupo,fit) #compara os ajustes do modelo
> anova(glm1)
> glm2=glm(grupo ~ tconhec + tpsico +aexper +ecivil +idade +
nfilhos +satisf +sexo, family=binomial(logit), data=func)
> fit=fitted(glm1) #ajuste do segundo modelo
```

Para discriminar a sensibilidade do modelo usamos o teste a seguir

### 8.7.3 Teste de Hosmer Lemeshow

Este teste foi implementado para avaliar o ajuste de um modelo logistico, a função implementada no R é dada por:

```
> hosmer=function (y, yhat, g = 10)
+ {
+   cutyhat <- cut(yhat, breaks = quantile(yhat, probs = seq(0,
+         1, 1/g)), include.lowest = T)
+     obs <- xtabs(cbind(1 - y, y) ~ cutyhat)
+    expect <- xtabs(cbind(1 - yhat, yhat) ~ cutyhat)
+     chisq <- sum((obs - expect)^2/expect)
+      P <- 1 - pchisq(chisq, g - 2)
+     c("X^2" = chisq, Df = g - 2, "P(>Chi)" = P)
+ }
> hosmer(grupo, fit2)
      X^2          Df    P(>Chi)
5.1208521   8.0000000   0.7445847
> hosmer(grupo, fit)
      X^2          Df    P(>Chi)
7.1358400   8.0000000   0.5220501
```

## 8.8 Exercícios

1. Considere os seguintes dados, referentes ao percentual de carboidratos, idade, peso e proteína de 20 diabéticos

| carbo | idade | peso | proteina |
|---|---|---|---|
| 33 | 33 | 100 | 14 |
| 40 | 47 | 92 | 15 |
| 37 | 49 | 135 | 18 |
| 27 | 35 | 144 | 12 |
| 30 | 46 | 140 | 15 |
| 43 | 52 | 101 | 15 |
| 34 | 62 | 95 | 14 |
| 48 | 23 | 101 | 17 |
| 30 | 32 | 98 | 15 |
| 38 | 42 | 105 | 14 |
| 50 | 31 | 108 | 17 |
| 51 | 61 | 85 | 19 |
| 30 | 63 | 130 | 19 |
| 36 | 40 | 127 | 20 |
| 41 | 50 | 109 | 15 |
| 42 | 64 | 107 | 16 |
| 46 | 56 | 117 | 18 |
| 24 | 61 | 100 | 13 |
| 35 | 48 | 118 | 18 |
| 37 | 28 | 102 | 14 |

2. Use a função **lm()**, para construir um modelo de regressão multipla.

3. Use a função anova(), para o modelo sugerido.

4. Use a função **glm()**, para construir um modelo linear generalizado.

5. Use a função anova(), para o modelo do item anterior.

6. Compare os modelos.

7. Considere os dados do peso do recém nascido **birthwt** do pacote **MASS** e construa um modelo binomial, onde o baixo peso do recém nascido esta em função da raça, fumo, hipertensão e peso da mãe.

8. Considere a variável resposta concentração de inseticida (insect), o número de moscas (moscas), o número de moscas mortas (mortas) e a probabilidade de moscas (prob), na tabela a seguir

| moscas | mortas | prob | insect |
|---|---|---|---|
| 59 | 6 | 0.10169 | 1.69070 |
| 60 | 13 | 0.21667 | 1.72420 |
| 62 | 18 | 0.29032 | 1.75520 |
| 56 | 28 | 0.50000 | 1.78420 |
| 63 | 52 | 0.82540 | 1.81130 |
| 59 | 53 | 0.89831 | 1.83690 |
| 62 | 61 | 0.98387 | 1.86100 |
| 60 | 60 | 1.00000 | 1.88390 |

9. Com os dados da tabela construa um modelo logit.
10. Com os dados da tabela construa um modelo probit.
11. Com os dados da tabela construa um modelo cloglog.
12. Compare os três modelos pelo criterio AIC.
13. Compare os três modelos pelo criterio do desvio residual.
14. Construa um data frame, para comparar e visualizar os ajustes do modelo.
15. Construa o mape dos modelos ajustados.
16. Com os dados do R, identifique o que esta sendo executado

```
library(MASS)
attach(birthwt)
race <- factor(race, labels = c("white", "black", "other"))
ptd <- factor(ptl > 0)
ftv <- factor(ftv)
levels(ftv)[-(1:2)] <- "2+"
bwt <- data.frame(low = factor(low), age, lwt, race,
smoke = (smoke > 0), ptd, ht = (ht > 0), ui = (ui > 0), ftv)
detach("birthwt")
options(contrasts = c("contr.treatment", "contr.poly"))
glm(low ~ ., binomial, bwt)
```

# Capítulo 9

# Séries temporais

Uma série temporal é uma coleção de observações feitas sequencialmente ao longo do tempo. A característica mais importante deste tipo de dados é que as observações vizinhas têm dependência e estamos interessados em analisar e modelar esta dependência. Enquanto em modelos de regressão por exemplo a ordem das observações é irrelevante para a análise, em séries temporais a ordem dos dados é crucial. Vale notar também que o tempo pode ser substituído por outra variável como espaço, profundidade, etc.[9]

Como a maior parte dos procedimentos estatísticos foi desenvolvida para analisar observações independentes o estudo de séries temporais requer o uso de técnicas específicas. Dados de séries temporais surgem em vários campos do conhecimento como Economia (preços diários de ações, taxa mensal de desemprego, produção industrial), Medicina (eletrocardiograma), Epidemiologia (número mensal de novos casos de meningite), Meteorologia (precipitação pluviométrica, temperatura diária, velocidade do vento), etc.

Os objetivos principais da análise de séries temporais é investigar o mecanismo gerador da série, descrever o comportamento, e identificar a presença de periodicidade ou sazonalidade e finalmente fazer previsões de valores futuros da série.

## 9.1 Pacotes e funções para série temporais no R

R contem várias pacotes e funções que são usadas para análise de séries temporais.

1. forecast: pacote para modelos de séries temporais

2. dse pacote para modelos de séries temporais multivariado (dse1, para estimação de sistemas dinâmicos; dse2, para extensão de sistemas dinâmicos como o métodos de raiz especifica).

3. acf: função para as funções de autocorrelação e função de autocorrelação parcial.

4. arima (auto.arima(), arima.sim(), $\cdots$): funções para estimar modelos ARMA, ARMAX, ARIMA, SARIMA.

5. fracdiff: pacote para modelos de memória longa ou Arima fracionados (ARFIMA).

6. HoltWinters: função para predição e ajustamento de algoritmos de HoltWinters com e sem sazonalidade (Aditivo e multiplicativo), métodos de suavização.

7. tsdiag: função de diagnostico para modelos de ajuste ARIMA.

8. uroot: pacote para testes de raiz untaria como Dickey-Fuller, Phillips-Perron, Hylleberg-Engle-Granger-Yoo, etc.

9. tseries: pacote para séries temporais e analise financeira, com modelos garch, e modelos arima, testes de Jarque -Bera, e testes de estacionaridade, etc.

10. urca: pacote para testar modelos de raiz unitária (Dicker e Fuller, Schmidt e Phillips, Phillips e Perron, e otros.

11. seriesChaos: função para modelos de séries temporais não linear, com sistema lorenz, ajuste recursivo, e espaço temporal.

12. tsfa: pacote de analise fatorial de séries temporais, simulação e predição.

## 9.2 Fontes

Séries temporais podem ser encontradas em páginas confiáveis na internet. Séries economicas do Brasil são encontradas em http://www.ipeadata.gov.br, outras séries temporais podem ser encontrados em fontes como o IBGE, no sitio: http://www.ibge.gov.br. Como exemplo, considere os dados baixados do ipeadata, correspondente a cotação dos dias úteis dos índices da BOVESPA, durante os anos de $2000 - 2010$. Assumindo que há em média 245 dias úteis no Brasil, os dados podem se chamados no R da seguinte maneira.

```
> bov=read.table("bovespa.txt",header=T) # dados em formato txt
> tbovespa=ts(bov,start=c(2000,1),frequency=245)
> plot.ts(tbovespa, main="cotação diaria da Bovespa", xlab="anos")
```

Figura 27. Série da Bolsa de valores de São Paulo.


Um segundo exemplo pode ser os índices econômicos mensais do Brasil no período de 1980 até abril de 2011, denominado de **decon.txt** e disponíveis em http://www.ipeadata.gov.br. As variáveis em estudo são:

1. anomes; corresponde ao ano e o mês da informação coletada,

2. tcc: taxa média de cambio do dólar compra, em reais,

3. tcv: taxa média de cambio do dólar venda, em reais,

4. bc: balança comercial em US (Dólares),

5. exp: exportações FOB em US (Dólares),

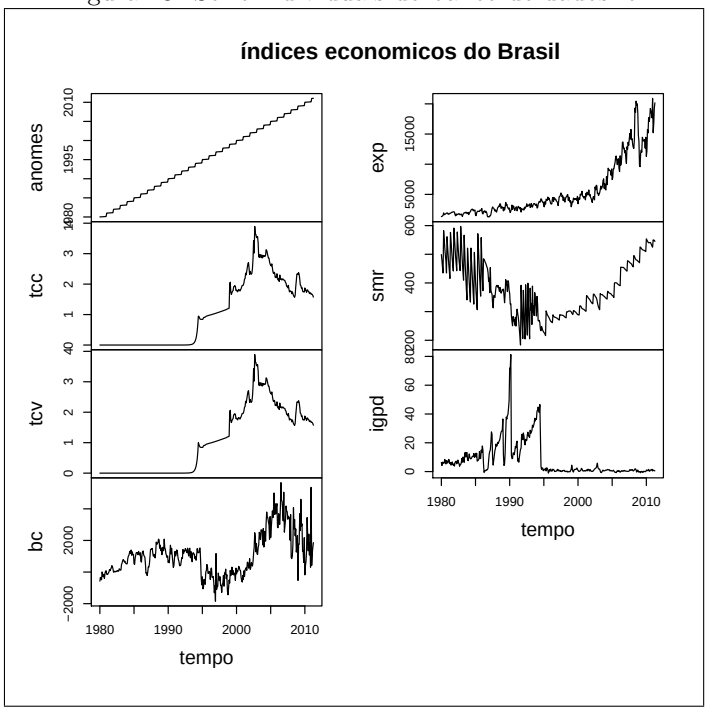
6. smr: salario mínimo real (IPEA) em reais,

7. igpd: índice geral de preços distribuiçao interna (igpdi).

Estas séries podem ser baixados um formato xls, simultaneamente do site do IPEA. A chamada dos dados e os gráficos no R e dada a seguir,

```
> ie=read.table("decon.txt",header=T)
> dim(ie)
[1] 376   7
> attach(ie)
> names(ie)
[1] "anomes" "tcc"    "tcv"    "bc"     "exp"    "smr"    "igpd"
> ie.ts=ts(ie,start=c(1980,1),frequency=12)
> plot.ts(ie.ts, main="Indicadores Econômicos do Brasil",xlab="tempo")
```

Figura 28. Série individuais do banco de dados ie.




Para obter uma serie de ie, podemos fazer:

```
> plot.ts(bc, main="Balança Comercial do Brasil em US")
```

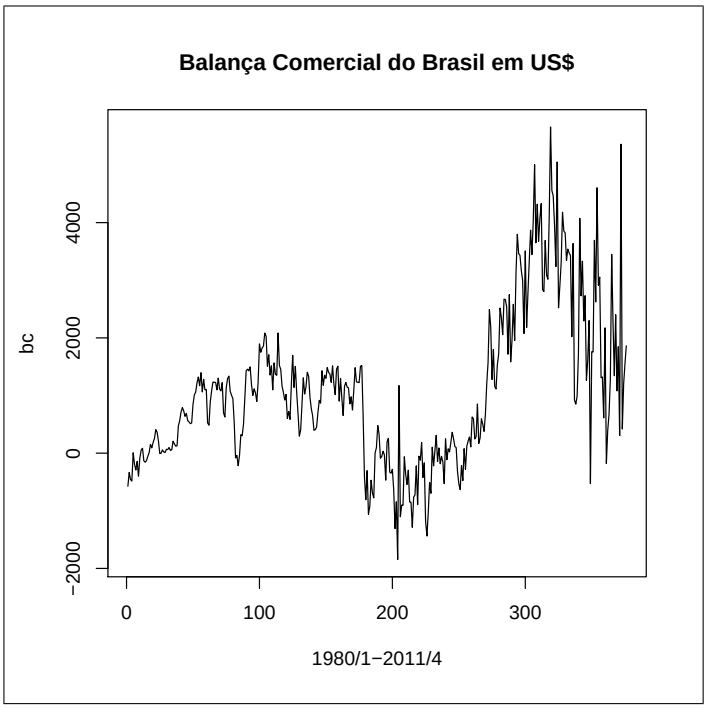



Figura 29. Uma série extraída do banco de dados ie.

## 9.3 Regressão linear com séries temporais

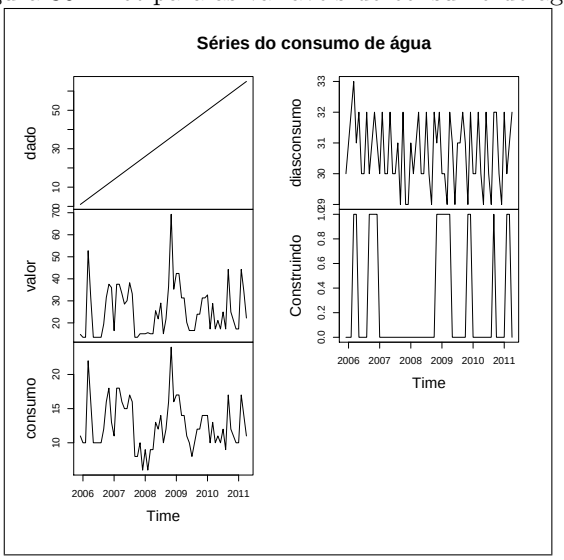
Em econometria frequentemente usamos o método de mínimos quadrados para ajustar um conjunto de observações, usando a função **lm()**, porem podemos construir convenientemente os dados a uma classe de séries temporais usando a função **ts()**. O objetivo de melhorar o ajuste das observações, estimando por métodos iterativos. Algumas propriedades são necessárias para implementar no formato de série temporal, como a data de inicio da série (start), e que típo o período que a série apresenta, (observações diárias, mensais, anuais).

Para usar modelos dinâmicos e relacionar nossa série temporal com os ***lags*** (retardos da série ou observações anteriores), podemos usar uma função **dynlm()**, do pacote **dynlm**. Para exemplificar consideremos o valor de consumo no instante $t$ pode estar relacionada com o valor de consumo no instante imediatamente anterior $t_1$ (mês anterior), da seguinte maneira: $valor_t = \beta_1 + \beta_2 diasconsumo_t + \beta_3 valor_{t-1}$, ou um modelo com as primeiras diferenças da seguinte forma: $valor_t - valor_{t-1} = \beta_1 + \beta_2(consumo_t - consumo_{t-1})$. Para exemplificar, primeiro usamos a função **ts()**:

```
> ca.ts=ts(ca,start=c(2005,12),frequency=12)
```

```
> plot(ca.ts,main="Séries do consumo de água")
```

Figura 30. Plot para as variáveis do consumo de água




Para construir os modelos descritos, temos

```
> mc1=dynlm(formula = valor ~ diasconsumo + L(valor))
> mc2=dynlm(d(valor)~d(consumo))
> summary(mc1)

Time series regression with "numeric" data:
Start = 1, End = 65

Call:
dynlm(formula = valor ~ diasconsumo + L(valor))

Residuals:
       Min         1Q     Median         3Q        Max
-1.022e-14 -2.383e-16  1.106e-16  5.037e-16  3.328e-15

Coefficients:
             Estimate Std. Error    t value Pr(>|t|)
(Intercept)  8.774e-15  5.205e-15  1.686e+00  0.09690 .
diasconsumo -4.706e-16  1.718e-16 -2.740e+00  0.00801 **
L(valor)     1.000e+00  1.682e-17  5.944e+16  < 2e-16 ***

Residual standard error: 1.498e-15 on 62 degrees of freedom
Multiple R-squared:     1,      Adjusted R-squared:     1
F-statistic: 1.846e+33 on 2 and 62 DF,  p-value: < 2.2e-16
```

Para visualizar o ganho de ajuste do modelo mc1 se comparado com o modelo m1, temos:

```
> plot.ts(valor)
> lines(fitted(m1),col=4)
> lines(fitted(mc1),col=8)
```

Um segundo exemplo para séries da classe ts(), com as características definidas nos indicadores econômicos (ie.ts), e usar regressões com as primeiras diferenças de uma variável, por exemplo considere a relação da balança comercial com as exportações, pode ser construído a seguinte regressão temporal:
Modelo 1 (me1).

$$bc_t = \beta_0 + \beta_1 exp_t + \beta_2 bc_{t-1} + \epsilon_t \tag{9.1}$$

Modelo 2 (me2).

$$bc_t = \beta_1 exp_t + \beta_2 bc_{t-1} + \beta_2 bc_{t-2} + \epsilon_t \tag{9.2}$$

Os dois modelos gerados no R é:

```
> library(dynlm)
> me1=dynlm(bc~exp+L(exp),data=ie.ts)
Time series regression with "ts" data:
Start = 1980(2), End = 2011(4)

Call:
dynlm(formula = bc ~ exp + L(exp), data = ie.ts)

Residuals:
    Min      1Q  Median      3Q     Max
-2963.0  -751.9   115.2   688.9  2805.8

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 151.18904   84.18325   1.796   0.0733 .
exp           0.41509    0.06194   6.701 7.66e-11 ***
L(exp)       -0.24954    0.06272  -3.979 8.34e-05 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 1044 on 372 degrees of freedom
Multiple R-squared: 0.3881,     Adjusted R-squared: 0.3848
F-statistic:   118 on 2 and 372 DF,  p-value: < 2.2e-16

##########################################################

> me2=dynlm(bc~exp+L(bc,1)+L(bc,2)-1,data=ie.ts)
> summary(me2)
```

```
Time series regression with "ts" data:
Start = 1980(3), End = 2011(4)

Call:
dynlm(formula = bc ~ exp +L(bc, 1) + L(bc, 2) - 1, data = ie.ts)

Residuals:
    Min      1Q  Median      3Q     Max
-2488.2  -244.6     2.9   262.6  3774.3

Coefficients:
         Estimate Std. Error t value Pr(>|t|)
exp      0.031757   0.007318   4.339 1.84e-05 ***
L(bc, 1) 0.414529   0.046141   8.984  < 2e-16 ***
L(bc, 2) 0.429927   0.046094   9.327  < 2e-16 ***
Residual standard error: 625.3 on 371 degrees of freedom
Multiple R-squared: 0.8676,     Adjusted R-squared: 0.8665
F-statistic: 810.4 on 3 and 371 DF,  p-value: < 2.2e-16
> deviance(me1)
[1] 405120372
> deviance(me2)
[1] 145081812
```

Os gráficos por separado das séries balança comercial e exportações, assim como o ajuste dois modelos me1 e me2 com a balança comercial é dado a seguir:

```
> plot(ie.ts[,c("bc","exp")],lty=c(3,1),plot.type="single",ylab="")
> legend("toplef",legend=c("balança","exportações"), lty=c(3,1),
bty="n")
> plot.ts(ie.ts[,"bc"],ylab="balança",xlab="tempo")
> f1=fitted(me1)
> f2=fitted(me2
> lines(f1,lty=2)
> lines(f2,lty=3)
> legend(1980,5000, legend=c("me1","me2"), lty=2:3)
```

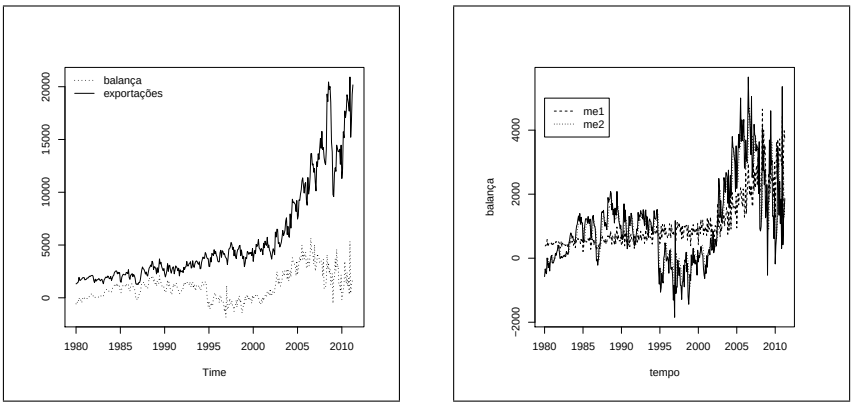



Figura 31. plot de duas séries e ajustes de dos modelos me1 e me2.

## 9.4 Deflacionando uma série

Em algumas séries é pertinente o uso da série real, desprovida de qualquer efeito inflacionário. Considere o IGPDI de gosto de 1994 até abril 2011 (igp) coletada do IPEADATA (ver dados ie), para realizar este deflacionamento usamos:

```
> attach(ie)
> igp=igpd[-c(1:175)]
> igp.ts=ts(igp, start=c(1994,8),frequency=12)
> igpdi=igp.ts/100
> igpdi=igpdi/igpdi[length(igpdi)]
> plot.ts(igpdi, main="IGPDI do Brasil")
> lines(igp.ts,lty=3)
> legend("topleft", c("IGPDI real", "IGPDI deflacionado"),
lty=1:3, bty="n")
```

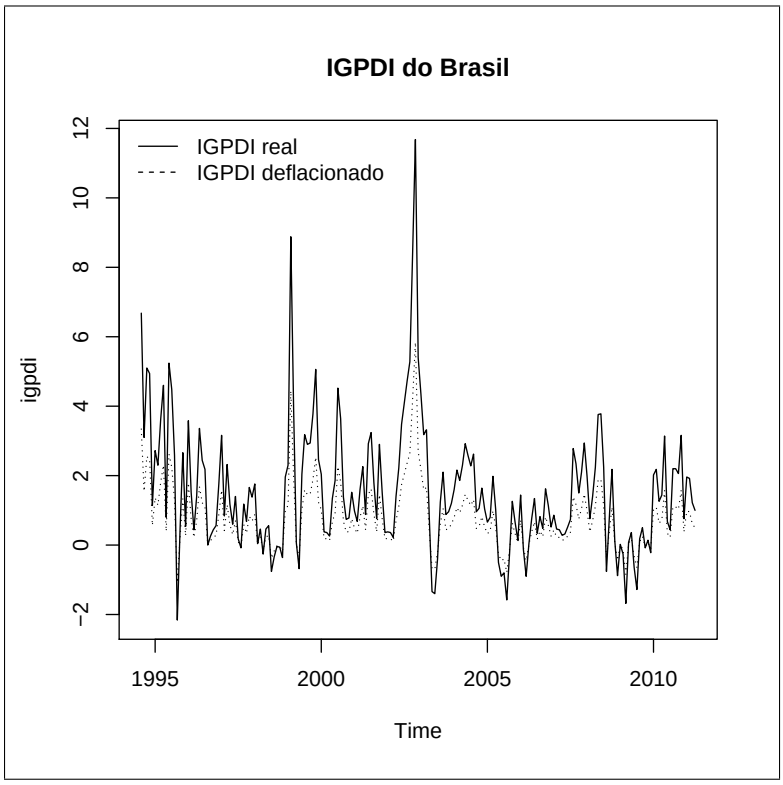



Figura 32. IGPDI - real e deflacionado.

## 9.5 Técnicas descritivas de uma série

Ao se analisar uma ou mais séries temporais a representação gráfica dos dados sequencialmente ao longo do tempo é fundamental, pode revelar padrões de comportamento importantes, como: tendências de crescimento (ou decrescimento), quando séries apresentam apenas este comportamento, modelos de regressão tem bom desempenho, porem quando a série apresentam padrões cíclicas, outras alternativas de ajuste são pertinentes, os quais serão abordados neste capítulo. A definição da tendencia e sazonalidade é: Seja $X_t, t = 1, 2, \ldots, T$, uma série temporal $X_t = T_t + S_t + \epsilon_t$, onde $T_t$, representa a tendência, $S_t$, a sazonalidade e $\epsilon_t$, um ruído branco (erro aleatório), com variância $\sigma_\epsilon^2$. Inicialmente o gráfico temporal deve ser sempre o primeiro passo e antecede qualquer análise.

### 9.5.1 Retornos

Para avaliar os riscos de uma carteira de ativos financeiros, o risco é medido em termos de variações de preços. Considere $P_t$ o preço de um ativo no instante

$t$. A variação de preços entre os instantes $t-1$ e $t$ é dada por: $\Delta P_t = P_t - P_{t-1}$, e o *retorno líquido simples* deste activo, no mesmo instante é definido por:

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}} = \frac{\Delta P_t}{P_{t-1}} \tag{9.3}$$

$R_t$, também chamado de *taxa de retorno*.
O *retorno composto* ou simplesmente *retorno* é dado por: $log(1 + R_t)$. na pratica usa-se os retornos que tem propriedades desejáveis como estacionaridade e ergodicidade próprios da modelagem ARMA, ARIMA ou GARCH. exemplo considere a cotação da Bovespa, os retornos e algumas características gráficas destas duas séries, são apresentadas a seguir:

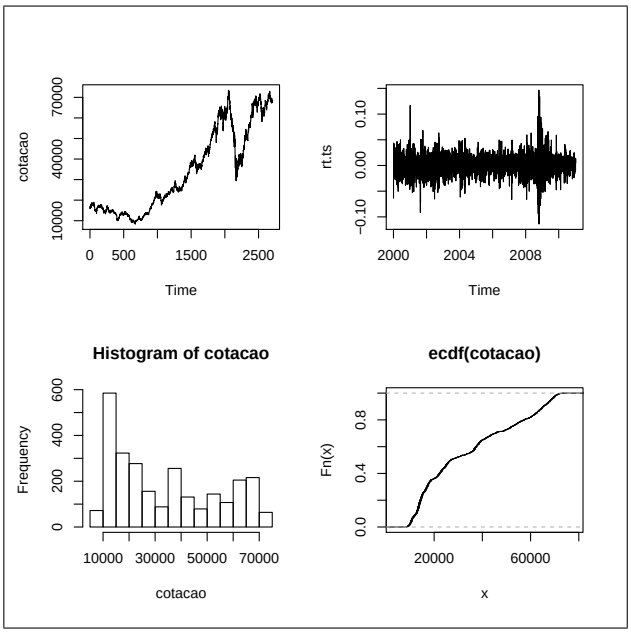



Figura 33. Características da cotação da Bovespa.

```
>attach(bov)

> rt=diff(cotacao)/cotacao[-length(cotacao)]

> rt.ts=ts(rt,start=c(2000,2),frequency=245)

> par(mfrow=c(2,2))

> plot.ts(cotacao)

> plot.ts(rt.ts)
```

```
> hist(cotacao)

> plot(ecdf(cotacao))
```

### 9.5.2 Decomposição clássica de una série

Muitas das propriedades observadas em uma série temporal $X_t$ podem ser captadas assumindo-se a seguinte forma de decomposição:

$$X_t = T_t + C_t + \epsilon_t \tag{9.4}$$

onde $T_t$ é uma componente de tendência, $C_t$ é uma componente cíclica ou sazonal e $\epsilon_t$ é uma componente aleatória ou ruído (a parte não explicada, que espera-se ser puramente aleatória). A componente cíclica se repete a cada intervalo fixo s, i.e

$$\ldots = C_{t-2s} = C_{t-s} = C_t = C_{t+s} = C_{t+2s} = \ldots \tag{9.5}$$

Assim, variações periódicas podem ser captadas por esta componente. Para visualizar a descomposição da cotação e os retornos da bovespa, usamos a função **decompose()**, ea função **stl()**, da seguinte maneira:

```
> dcotacao=decompose(cotacao.ts)

> stlcotacao=stl(cotacao.ts,s.window=13)

> drt.ts=decompose(rt.ts)

> stlrt.ts=stl(rt.ts, s.window=13)

> plot(stlcotacao, main="Decomposição da cotação da bovespa")

> plot(stlrt.ts, main="Decomposição dos retornos da bovespa")
```

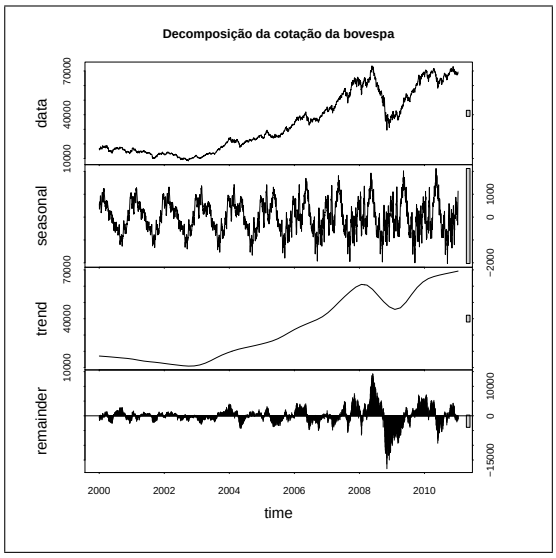



Figura 34. Cotação dos índices da Bovespa.

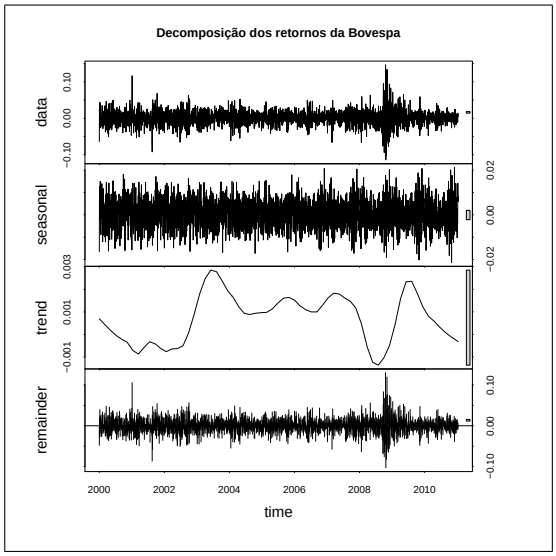



Figura 35. Retorno dos índices da Bovespa.

### 9.5.3 Autocorrelação

Uma importante ferramenta para se identificar as propriedades de uma série temporal é o coeficiente de autocorrelação.

**Definição 4** *Seja* $X_t$, $t \in T$ / $Var(x_t) < \infty \quad \forall t$. *A função de Autocovariância* $\gamma_X(.,.)$ *de* $X_t$ *é definida como:*

$$\gamma_X(r,s) = E[(X_r - E(X_r))(X_s - E(X_s))], \qquad r,s \in T. \tag{9.6}$$

no exemplo dos retornos da Bovespa, seja $s = r - 1$, então

$$\gamma_X(r, r-1) = E[(X_r - E(X_r))(X_{r-1} - E(X_{r-1}))]. \tag{9.7}$$

```
> cov(rt[-length(rt)],rt[-1])
[1] 2.210996e-06
```

se $r = s$, então $\gamma_X(r, s) = Var(X_r)$.

```
> var(rt)
[1] 0.0003957635
```

Notação alternativa:

$$\gamma(h) = Cov(X_t, X_{t+h}) = Cov(X_t, X_{t-h}) \tag{9.8}$$

**Teorema 5** *Seja $\gamma(h)$, $h = 0, 1, 2, \ldots$, a função de autocovariancia de um processo estacionario, então:*

1. $\gamma(0) \geq 0$
2. $|\gamma(h)| \leq \gamma(0)$
3. $\gamma(.)$ *é par.*

**Definição 6** *Seja $X_t, t \in T$ um processo estacionário. A função de Autocorrelação do processo é $\rho(h) = corr(X_t, X_{t+h}) = \gamma(h)/\gamma(0)$, e onde $-1 \leq \rho(h) \leq 1$*

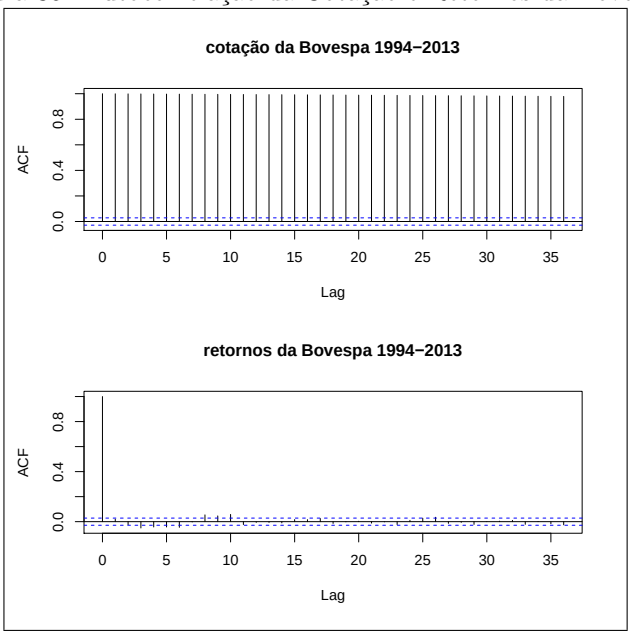
### 9.5.4 O Correlograma

Um gráfico com os $k$ primeiros coeficientes de autocorrelação como função de $k$ é chamado de correlograma e pode ser uma ferramenta poderosa para identificar características da série temporal. Porém isto requer uma interpretação adequada do correlograma, i.e. devemos associar certos padrões do correlograma como determinadas características de uma série temporal. Esta nem sempre é uma tarefa simples e a seguir são dadas algumas indicações. A primeira questão que podemos tentar responder através do correlograma é se uma série temporal é aleatória ou não. Para uma série completamente aleatória os valores defasados são não correlacionados e espera-se que $\rho(h) = 0$.

Considere as autocorrelações da cotação da Bovespa e os Retornos da Bovespa

```
> par(mfrow=c(2,1))
> acf(bov, main="cotação da Bovespa 2000-2010")
> acf(rt, main="retornos da Bovespa 2000-2010")
```

Figura 36. Autocorrelação da Cotação e Retornos da Bovespa.




### 9.5.5 Processos Estacionários

Suposições de estacionaridade (significa flutuação em torno de uma reta) ou transformação de uma série em estacionaria é comum em séries temporais.

**Definição 7** *Um Processo estocástico é uma família de variáveis aleatórias* $\{X_t(\omega)\}_{t\in T}$ *, definidas no mesmo espaço de probabilidade* $(\Omega, \Im, P)$*, onde* $\omega \in \Omega$ *e* $T$ *é um conjunto arbitrário. Aqui,* $\Omega$ *é o espaço amostral,*$\Im$ *é uma* $\sigma$*-algebra de* $\Omega$ *e* $P$ *é uma medida de probabilidade em* $\Im$*.*

**Definição 8** *Uma série temporal* $X_t, t \in \mathbb{N}$ *é dita ser fortemente estacionaria se as funções de distribuição* $\forall\ n \in \mathbb{N}$ *e* $\forall t_1, \ldots, t_n, t_{1+n}, \ldots, t_{n+h} \in T$ *são idênticas, onde* $T$ *é o conjunto de índices em geral. O conjunto* $T$ *é comumente tomado como o conjunto dos números inteiros* $\mathbb{Z} = \{0, \pm 1, \pm 2, \cdots\}$*. Uma série temporal é uma realização de um certo processo estocástico. Os dois primeiros momentos de* $\{X_t(\omega)\}_{t\in\mathbb{Z}}$ *são definidos como*

$$E[X_t] = \mu_t \ e\ E(X_t - \mu_t)^2 = \sigma_t^2,$$

e a correlação é dada por

$$\frac{Cov(X_t, X_{t+h})}{\sqrt{\sigma_t^2 \sigma_{t+h}^2}} \qquad para\ h \in \mathbb{Z}, \tag{9.9}$$

**Definição 9** *Estacionaridade. Um processo estocástico é dito ser fracamente estacionário se e somente se:*

*1.* $E[X_t] = \mu$*, para todo* $t \in \mathbb{Z}$*,*
*2.* $E(X_t - \mu)^2 = \sigma^2, 0 < \sigma^2 < \infty$*, para todo* $t \in \mathbb{Z}$*,*
*3.* $R(h) = Cov(X_t, X_{t+h})$ *depende apenas de* $h$*, para todo* $t \in \mathbb{Z}$*.*

As autocorrelações $\rho(h)$ são obtidas normalizando as autocovariâncias através da sua divisão pelo produto dos respectivos desvios padrão, i.e., $\rho(h) = \frac{R(h)}{R(0)}$. O exemplo mais simples de um processo estacionário é o processo de ruído branco *(RB)*, definido como uma sequência de variáveis aleatórias não-correlacionadas com média constante e variância constante (estritamente positiva e finita) ao longo do tempo.

**Definição 10** *Processo linear geral pode ser representado por :* $\{X_t\}$

$$X_t = \sum_{-\infty}^{\infty} \psi_j \epsilon_{t-j}, \qquad t \in \mathbb{Z},$$

*onde* $\{\epsilon_t\} \sim RB(0, \sigma^2)$ *e* $\{\psi_j\}$ *é uma sequencia de constantes com* $\sum_{-\infty}^{\infty} |\psi_j| < \infty$*.*

**Definição 11** *Função geratriz de autocovariâncias. Seja:* $\{X_t\}$*, um processo estacionário com função de autocovariâncias* $R(h)$ *que satisfaz* $\sum_{h=-\infty}^{\infty} |R(h)| < \infty$*. A função geratriz de autocovariâncias de* $\{X_t\}$ *é definida como*

$$g(z) = \sum_{h=-\infty}^{\infty} R(h) z^h$$

*onde* $z$ *é um escalar complexo.*

### 9.5.6 Passeio Aleatório

Seja: $\epsilon_t$ um processo discreto puramente aleatório com média $\mu$ e variância $\sigma^2$. Um processo $X_t$ é chamado de passeio aleatório se $X_t = X_{t-1} + \epsilon_t$.

## 9.6 Modelos de séries temporais

O estudo de séries temporais pode ser motivado pelo interesse em investigar o mecanismo gerador de um conjunto de dados observados ao longo do tempo para descrever sua dinâmica com o objetivo de gerar previsões acerca de seu comportamento futuro. Para tanto são construídos modelos probabilísticos que pertencem a um domínio temporal previamente estabelecido [27].

### 9.6.1 Alisamento exponencial

É uma classe de algoritmos de previsão, os mais conhecidos são:

**Algoritmo simples**

A componente não observável deste algoritmo é o nível, o interesse é estimar este nível e projetá-lo para gerar previsões futuras. As estimativas do nível é:

$$N_t = \alpha y_t + \alpha(1-\alpha)y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + \cdots, \qquad 0 < \alpha < 1 \tag{9.10}$$

$\alpha$ é constante de alisamento. A forma de recorrência é:

$$N_t = \alpha y_t + (1-\alpha)N_{t-1} \tag{9.11}$$

A forma de correção de erros é:

$$N_t = N_{t-1} + \alpha\epsilon_t \tag{9.12}$$

Onde $\epsilon_t$: erro de previsão um passo à frente

**Algoritmo de Holt**

Este algoritmo apresenta duas componentes não observáveis: o nível e a tendência, sendo que esta última não precisa ser globalmente fixa, podendo evoluir ao longo do tempo. A forma de recorrência é:

$$N_t = \alpha y_t + (1-\alpha)[N_{t-1} + T_{t-1}] \tag{9.13}$$
$$T_t = \beta(N_t - N_{t-1}) + (1-\beta)T_{t-1} \tag{9.14}$$

Escolhe-se $0 < \alpha, \beta < 1$, que minimizem $\sum \epsilon_t^2$. A forma de correção de erros é:

$$N_t = N_{t-1} + T_{t-1} + \alpha\epsilon_t \tag{9.15}$$
$$T_t = T_{t-1} + \alpha\beta\epsilon_t \tag{9.16}$$

**Algorimo de Holt-Winters(HW)**

A suposição é que a série possuí três componentes não observáveis: nível, tendência e sazonalidade. Há dois enfoques para este algoritmo, o aditivo e o multiplicativo.

**HW aditivo**

$$N_t = \alpha(y_t - F_{t-s}) + (1-\alpha)[N_{t-1} + T_{t-1}] \tag{9.17}$$
$$T_t = \beta(N_t - N_{t-1}) + (1-\beta)T_{t-1} \tag{9.18}$$
$$F_t = \gamma(y_t - N_t) + (1-\gamma)F_{t-s} \tag{9.19}$$

Escolhe-se $0 < \alpha, \beta, \gamma < 1$. A forma de correção de erros é dado por:

$$N_t = N_{t-1} + T_{t-1} + \alpha\epsilon_t \tag{9.20}$$
$$T_t = T_{t-1} + \alpha\beta\epsilon_t \tag{9.21}$$
$$F_t = F_{t-s} + \gamma(1-\gamma)\epsilon_t \tag{9.22}$$

**HW multiplicativo**

$$N_t = \alpha \frac{y_t}{F_{t-s}} + (1-\alpha)[N_{t-1} + T_{t-1}] \tag{9.23}$$

$$T_t = \beta(N_t - N_{t-1}) + (1-\beta)T_{t-1} \tag{9.24}$$

$$F_t = \gamma \frac{y_t}{N_t} + (1-\gamma)F_{t-s} \tag{9.25}$$

Escolhe-se $0 < \alpha, \beta, \gamma < 1$. A forma de correção de erros é dado por:

$$N_t = N_{t-1} + T_{t-1} + \frac{\alpha}{F_{t-s}}\epsilon_t \tag{9.26}$$

$$T_t = T_{t-1} + \frac{\alpha\beta}{F_{t-s}}\epsilon_t \tag{9.27}$$

$$F_t = F_{t-s} + \frac{\gamma(1-\gamma)}{N_t}\epsilon_t \tag{9.28}$$

Uma aplicação de estes algoritmos é dado a seguir:

```
> hwc=HoltWinters(cotacao.ts)
> hwr=HoltWinters(rt.ts)
> hwcm=HoltWinters(cotacao.ts, seasonal="multiplicative")
```

Para os retornos dos índices da bovespa não é pertinente o uso de algoritmos de Holt-Winters multiplicativo, por apresentar valores zero. Para visualizar gráficos e o resumo dos parâmetros dos modelos acima propostos, usamos:

```
> plot.ts(cotacao.ts)
> lines(fitted(hwc)[,1],col=2)
> f=cbind(hwc,hwr,hwcm)
> f
             hwc           hwr           hwcm
fitted       Numeric,9832 Numeric,9828 Numeric,9832
x            Integer,2703 Numeric,2702 Integer,2703
alpha        0.9307564    0.001454172  0.8457155
beta         0.001083805  0.001273331  0
gamma        1            0.2754765    1
coefficients Numeric,247  Numeric,247  Numeric,247
seasonal     "additive"   "additive"   "multiplicative"
SSE          1572141540   1.249631     2218260761
call         Expression   Expression   Expression
```

Para realizar previsões futuras da cotação 6 dias à frente, usamos a função: predict(hwc,6).

## 9.7 Processos auto-regressivos e de médias móveis

Para abordar os processo ARMA ou auto-regressivos e de médias móveis, precisamos identificar algumas propriedades que são fundamentais neste tipo de processos.

$$\Delta x_t = x_t - x_{t-1} \tag{9.29}$$

Para uma n-ésima diferença temos

$$\Delta^n x_t = \Delta^{n-1} x_t - \Delta^{n-1} x_{t-1} = \sum_{r=1}^{n} (-1)^r \binom{n}{r} x_{t-r} \tag{9.30}$$

Como exemplo considere:

$$\Delta^2 x_t = \Delta x_t - \Delta x_{t-1} = x_t - x_{t-1} - (x_{t-1} - x_{t-2}) = x_t - 2x_{t-1} + x_{t-2}$$

Operador de defassagem (B)

$$Bx_t = x_{t-1}, \qquad B^2 x_t = x_{t-2}, \qquad B^n x_t = x_{t-n} \tag{9.31}$$

Na metodologia de Box e Jenkin, é preciso identificar os estágios do ciclo iterativo, os quais são:

1. Uma classe de modelo de séries temporais é especificada (proposta)
2. Uma etapa de identificação a partir das funções de autocorrelação, autocorrelação parcial, função de autocovariância.
3. A fase de estimar os parâmetros do modelo proposto
4. O diagnostico do modelo através de critérios de seleção de modelos, com por exemplo AIC, BIC, estabilidade estrutural do modelo são requeridas

### 9.7.1 Processos autorregressivos

Um processo autorregressivo de primeira ordem $AR(1)$ é dado por:

$$x_t = c + \phi x_{t-1} + \nu_t, \qquad \nu \approx RB(0, \sigma^2),\ c, \phi \in \Re \tag{9.32}$$

Este processo é estacionário se $|\phi| < 1$
O processo autorregressivo de ordem $p$, $AR(p)$ é dado por:

$$x_t = c + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \nu_t \tag{9.33}$$

Onde $\nu \approx RB(0, \sigma^2),\ c, \phi \in \Re$. Uma forma alternativa de escrever estes processos é dado por:

$$(1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p) x_t = c + \nu_t \tag{9.34}$$

Centralizando a variável em torno de $c$, podemos escrever como $\phi(B)x_t = \nu_t$. No caso de um $AR(p)$, é estacionário se todas as raízes de $\phi(z) = 0$ estiver

fora do circulo unitário.

Todo processo autorregressivo é invertível. Exemplo, para identificar um modelo como sendo autorregressivo, Box-Jenkins consideraram que se a função de autocorrelação têm um decaimento exponencial (observar a função de autocorrelação, FAC), somente $\phi$ é significativo (observar a função de autocorrelação parcial, FACP) á indícios da presença de um processo $AR(1)$. Ver detalhes para identificação de modelos AR(p), em Moretin e Toloi [27].

### 9.7.2 Processo de médias móveis

O processo de médias moveis de primeira ordem $MA(1)$ é dado por:

$$x_t = \mu + \theta_1 \epsilon_{t-1} + \epsilon_t, \qquad \epsilon \approx RB(0, \sigma^2),\ \mu, \theta \in \Re \tag{9.35}$$

Este processo é invertível se $|\theta| < 1$.
A autocovariância dada por:

$$\rho_1 = \frac{\theta}{1+\theta^2} \tag{9.36}$$

O processo de médias moveis de ordem $q$, $MA(q)$ é dado por:

$$x_t = \mu + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} \tag{9.37}$$

Onde $\epsilon \approx RB(0, \sigma^2)$. Uma forma alternativa de escrever estes processos é dado por:

$$(1 - \theta_1 B - \theta_2 B^2 - \cdots - \theta_q B^q)\epsilon_t = x_t + \mu \tag{9.38}$$

Centralizando a variável em torno de $\mu$, podemos escrever como $\theta(B)\epsilon_t = x_t$. No caso de um $MA(q)$, é invertível se todas as raízes de $\theta(z) = 0$ estiver fora do circulo unitário. Todo processo de médias móveis é estacionário. Exemplo, para identificar se um modelo como sendo de médias moveis $MA(1)$, somente $\rho_1$ é significativo ou diferente de zero (ver função de autocorrelação) e a função de autocorrelação parcial deve ter um decaimento exponencial dominante. Ver detalhes para identificação de modelos $MA(q)$, em Moretin e Toloi [27] .

### 9.7.3 Processo ARMA

Seja $\{X_t\}$ um processo que satisfaz a equação em diferenças dada por

$$\Phi(B)X_t = \Theta(B)\epsilon_t, \tag{9.39}$$

onde $\{\epsilon_t\}$ é ruído branco, ie., $\{\epsilon_t\} \sim RB(0, \sigma_\epsilon^2)$, $B$ é operador de defasagem definido como $B^m X_t = X_{t-m}$, $m = 1, \ldots, p$, $\Phi(z) = 1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$ e $\Theta(z) = 1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$.

O processo $\{X_t\}$ definido em (9.39) é chamado processo auto-regressivo e de médias móveis, ARMA$(p, q)$.

**Definição 12** *Invertibilidade. Um processo* $\{X_t\}$ *com representação* $ARMA(p,q)$ *é invertível se existem constantes* $\{\pi_j\}$ *tais que* $\sum_0^\infty \mid \pi_j \mid < \infty$ *e* $\epsilon_t = \sum_0^\infty \pi_j X_{t-j}$, *para todo* $t \in \mathbb{Z}$.

Seguindo as definições 6 e 8 o processo (9.39) é estacionário e invertível se as raízes de $\Phi(z) = 0$ e $\Theta(z) = 0$ são não comuns e encontram-se fora do círculo unitário.

**Definição 13** *Causalidade. Um processo* $\{X_t\}$ *com representação* $ARMA(p,q)$ *é causal, ou função causal de* $\{\epsilon_t\}$, *se existem constantes*$\{\psi_t\}$ *tais que* $\sum_{j=0}^\infty |\psi_j| < \infty$, *e* $X_t = \sum_{j=0}^\infty \psi_j \epsilon_{t-j}$ *para todo* $t \in \mathbb{Z}$.

Note que as propriedades de invertibilidade e causalidade não são apenas do processo $\{X_t\}$, mas também da relação entre os processos $\{X_t\}$ e $\{\epsilon_t\}$ da definição da equação ARMA apresentada em (9.39). Invertibilidade e causalidade garantem que há uma solução única estacionária, com probabilidade um, para a equação ARMA.

## Função de Autocovariâncias e densidade espectral de um processo ARMA$(p,q)$.

O cálculo da função de autocovariância as para um processo $\{X_t\}$ com representação ARMA$(p,q)$ causal é realizado através das equações

$$R(k) - \phi R(k-1) - \cdots - \phi R(k-p) = \sigma_\epsilon^2 \sum_{j=0}^{\infty} \theta_{k+j}\psi_j, \qquad 0 \neq k < m,$$
$$R(k) - \phi R(k-1) - \cdots - \phi R(k-p) = 0, \qquad k \geq m,$$

onde $m = \max(p, q+1), \psi_j - \sum_{k=1}^p \phi_k \psi_{j-k} = \theta_j, j = 0, 1, \ldots, \psi_j = 0$ para $j < 0, \theta_0 = 1$ *e* $\theta_j = 0$ para $j \in \{0, 1, \ldots, q\}$.

O espectro de $\{X_t\}$ é dado por

$$f_{ARMA}(\lambda) = \frac{\sigma_\epsilon^2 |\Theta(\exp^{-i\lambda})|^2}{2\phi |\Phi(\exp^{-i\lambda})|^2}, \qquad \lambda \in [-\pi, \pi] \tag{9.40}$$

### 9.7.4 Processo ARIMA$(p,d,q)$

Seja $d$ um inteiro não negativo. $\{x_t\}$ é um processo auto-regressivo integrado e de médias móveis ARIMA$(p,d,q)$ se $Y_t = (1-B)^d X_t$ é um processo ARMA(p,q) causal. Esta definição sugere que $\{X_t\}$ satisfaz a equação em diferenças da forma

$$\Phi(B)(1-B)^d X_t = \Theta(B)\epsilon_t, \qquad \{\epsilon_t\} \sim RB(0, \sigma_\epsilon^d).$$

**Exemplo 1**

Para exemplificar a modelagem arima usaremos a função **auto.arima()** do pacote **forecast**, para a cotação e os retornos dos índices da bovespa, disponíveis em **bov**.

```
>
> library(forecast)
>bov=read.table(file.choose(),header=T) #bovespa15072013.txt
> attach(bov)
> #Arima para a cotação do Bovespa
> cotarima=auto.arima(cotacao)
> summary(cotarima)
Series: cotacao
ARIMA(2,1,2)

Coefficients:
          ar1     ar2     ma1      ma2
      -0.2896  0.7015  0.2468  -0.7336
s.e.   0.0805  0.0800  0.0758   0.0747

sigma^2 estimated as 0.4172:  log likelihood=-4624.42
AIC=9258.83   AICc=9258.85   BIC=9291.12

Training set error measures:
                     ME      RMSE       MAE        MPE     MAPE      MASE
Training set 0.01047288 0.6458221 0.4019317 0.03303618 1.582666 0.9999299
                    ACF1
Training set 0.009322901
> par(mfrow=c(2,1))
> plot.ts(cotacao)
> lines(fitted(cotarima),col=4)
> plot.ts(cotacao,xlim=c(4700,4730), ylim=c(45,48))
> abline(v=4711,lty=2,col=8)
> previsto=predict(arima(cotacao,order=c(2,1,2)),4)
> previsto
$pred
Time Series:
Start = 4712
End = 4715
Frequency = 1
[1] 46.70404 46.68037 46.66341 46.65171

$se
Time Series:
Start = 4712
End = 4715
```

```
Frequency = 1
[1] 0.6458907 0.8940655 1.0797783 1.2303234

> lines(previsto$pre,col=4)
> bovespa161907=c(46.869,47.407,47.656,47.400)
> lines(4712:4715,bovespa161907,col=2)
> mapexterno=mean(abs(bovespa161907-previsto$pred)/bovespa161907)*100
> mapexterno
[1] 1.386548
```

A saída gráfica é dada por:

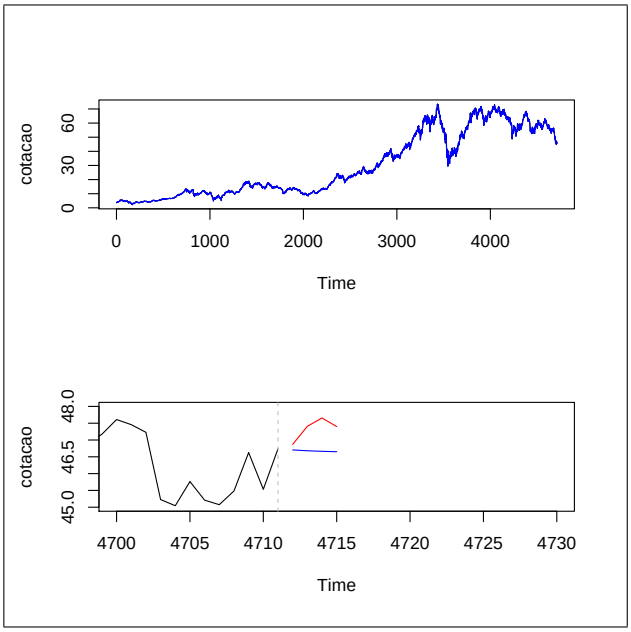



Figura a. Cotação da Bovespa e previsões

Para os retornos usamos:

```
> #Arima para os retornos da Bovespa

> rt=diff(cotacao)/cotacao[-length(cotacao)]
> rtarima=auto.arima(rt)
> summary(rtarima)
Series: rt
ARIMA(3,0,3) with non-zero mean

Coefficients:
         ar1      ar2      ar3      ma1      ma2     ma3  intercept
```

```
      0.4257  0.6206  -0.7684  -0.4073  -0.6461  0.7077      8e-04
s.e.  0.0501  0.0642   0.0682   0.0556   0.0703  0.0711      3e-04

sigma^2 estimated as 0.0005216:  log likelihood=11117.27
AIC=-22218.53   AICc=-22218.5   BIC=-22166.87

Training set error measures:
                      ME       RMSE       MAE MPE MAPE      MASE         ACF1
Training set 2.430886e-05 0.02283862 0.0157432 NaN  Inf 0.7074504 -0.005039146
> par(mfrow=c(2,1))
> plot.ts(rt)
> lines(fitted(rtarima),col=4)
> plot.ts(rt,xlim=c(4700,4730), ylim=c(-0.1,0.1))
> abline(v=4710,lty=2,col=8)
> previstor=predict(arima(rt,order=c(3,0,3)),4)
> previstor
$pred
Time Series:
Start = 4711
End = 4714
Frequency = 1
[1] -6.696700e-04 -9.413669e-04 -1.080333e-03  3.628575e-05

$se
Time Series:
Start = 4711
End = 4714
Frequency = 1
[1] 0.02283862 0.02284251 0.02284604 0.02288278

> lines(previstor$pre,col=4)
> retorno161907=c(0.026464323,0.002802858,0.011478803,0.005252389)
> lines(4711:4714,retorno161907,col=2)
> mapexterno=mean(abs(retorno161907-previstor$pred)/retorno161907)*100
> mapexterno
[1] 111.2093
> error=mean(abs(retorno161907-previstor$pred))*100
> error
[1] 1.216336
```

A saída gráfica é dada por:

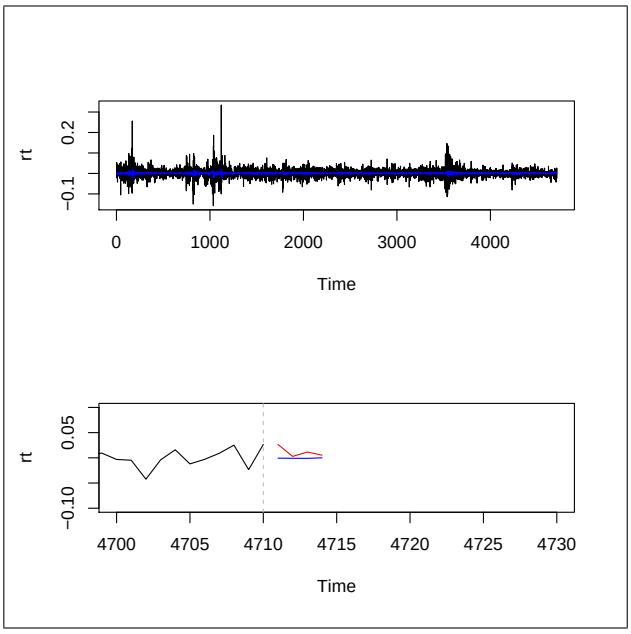



Figura b. Retornos da Bovespa e previsões.

### 9.7.5 Simulações de modelos Arima

O R contem algumas funções para simulação de modelos autorregressivos e de médias móveis. Exemplo: simulação de séries temporais (T = 200) de modelo estacionário com processo autorregressivo gaussiano AR(1), com parâmetros 0.5 e 0.95.

```
> y = arima.sim(n=200, list(ar=0.5), innov=rnorm(200))
> y = arima.sim(n=200, list(ar=0.95), innov=rnorm(200))
> plot.ts(y) #gráfico correspondente
> acf(y)
```

Identifique matematicamente as simulações de modelos AR, MA e ARIMA abaixo descritos.

```
> AR2=arima.sim(n = 100, list( ar = c(0.8897, -0.4858)))
> AR1=arima.sim(n = 100, list( ar = 0.8897))
> MA1=arima.sim(n = 100, list( ma = 1))
> MA2=arima.sim(n = 100, list( ma = c(-2, 1.5)))
> ARMA11=arima.sim(n = 100, list(ar=0.8897, ma=-0.2279))
> ARMA22=arima.sim(n = 100, list(ar=c(0.8897, -0.4858),
ma=c(-0.2279, 0.2488)))
> ARIMA110=arima.sim(list(order=c(1,1,0),ar = 0.7),n = 100)
> ARIMA111=arima.sim(list(order = c(1,1,1), ar = 0.7,
ma = 2), n = 100)
```

```
> par(mfrow=c(2,2), pty="s")
> plot(AR1, main="AR(0.8897)")
> plot(AR2, main="AR(0.8897, -0.4858)")
> plot(MA1,  main="MA(1.000)")
> plot(MA2,  main="MA(-2, 1.5)")
```

### 9.7.6 Modelos Sarima

Os modelos SARIMA são a extensão dos modelos ARIMA (*sazonal multiplicativo*) e contêm uma parte não sazonal da série, com parâmetros $(p, d, q)$, e uma sazonal, com parâmetros $(P, D, Q)$. O modelo mais geral é dado pela equação:

$$\Phi(B)\Phi(B)(1-B)^D(1-B)^d X_t = \theta(B)\Theta(B)\epsilon_t, \qquad \{\epsilon_t\} \sim RB(0, \sigma_\epsilon^d).$$

Para exemplificar considere uma série $x_t$, observada mês a mês por um modelo $ARIMA(1,0,0)$:

$x_t = \phi x_{t-1} + \epsilon_t$,
para modelar por estações do ano, através de um processo $AR(1) = ARIMA(1,0,0)$ sazonal:

$x_t = \Phi_4 x_{t-4} + \epsilon_t$,
para modelar ano a ano através do processo $MA(1) = ARIMA(0,0,1)$ sazonal:

$x_t = \epsilon_t - \Theta\epsilon_{t-12}$.

**Exemplo 2**

Considere a cotação da bovespa do exemplo 1, para identificar o modelo sarima, usamos

```
> library(forecast)
> acotacao=auto.arima(cotacao.ts)
> summary(acotacao)
> aretorno=auto.arima(rt.ts)
> summary(aretorno)
```

**Exemplo 3**

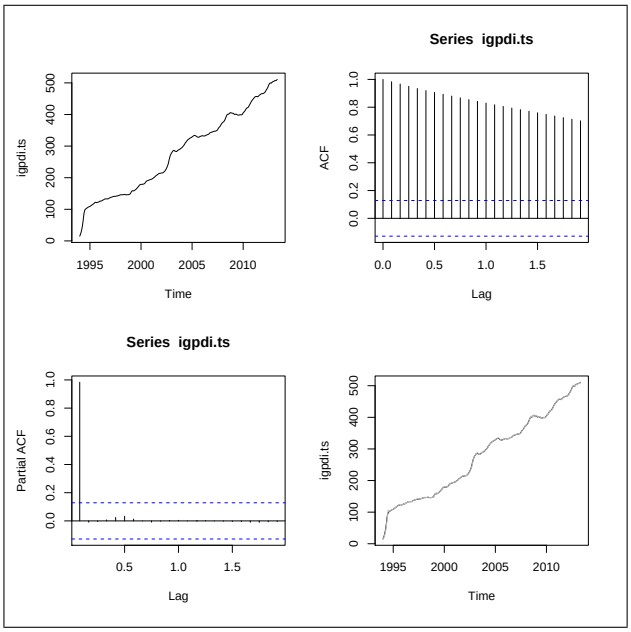



Figura 37. Características da série do IGPDI

Considere a série aicima do índice geral de preços distribuição interna do Brasil por mês desde janeiro de 1944 até abril de 2009, disponíveis no site do IPEA.

```
> library(forecast)
> igpdi=read.table("igpdi.R")
> attach(igpdi)
> igpdi.ts=ts(igpdi,start=c(1944,2), frequency=12)
> par(mfrow=c(2,2))
> plot.ts(igpdi.ts)
> acf(igpdi.ts)
> pacf(igpdi.ts)
> plot.ts(igpdi.ts)
> lines(fitted(auto.arima(igpdi.ts)),col=8)
```

Para realizar previsões com o respectivo gráfico para os meses de maio a setembro de 2009, fazemos

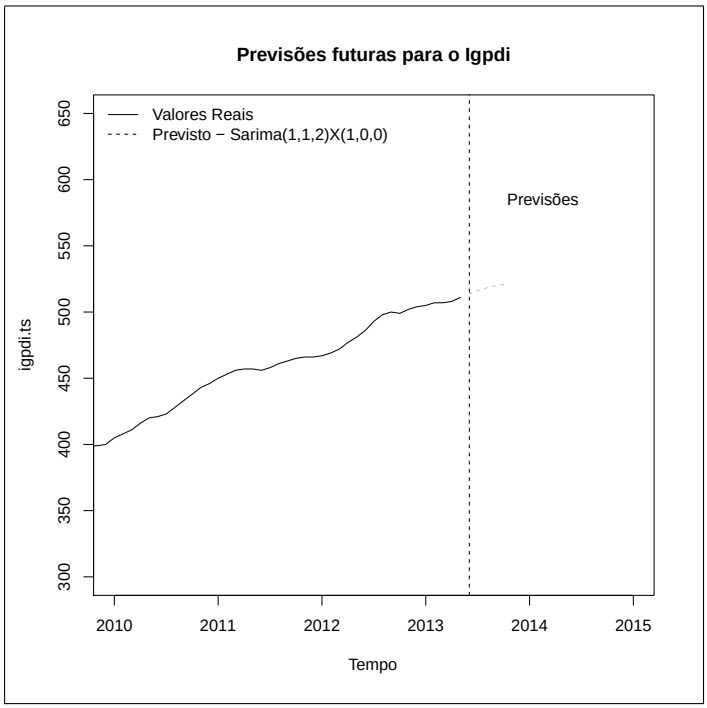



figura 38. Série do IGPDI e previsões.

```
> ma=auto.arima(igpdi.ts)
> pa=predict(ma,5)
$pred
           May       Jun       Jul       Aug       Sep
2009 0.1770981 0.3448479 0.3154377 0.3794410 0.3502937

se
          May      Jun      Jul      Aug      Sep
2009 3.494638 4.815471 5.716056 6.182726 6.527294
plot.ts(igpdi.ts,xlab="Tempo", main=
"Previsões futuras para o Igpdi",xlim=c(2000.1,2015))
abline(v=2009.3, lty=2)
legend("topleft", c("Valores Reais",
"Previsto - Arima(3,1,2)"),lty = 1:2, bty="n")
legend(2009.4,60, legend="Previsões", bty="n")
lines(pa$pred, type="l", col=8, lty=2)
```

### Arima vs Sarima

Consideremos o salário mínimo real mensal do Brasil desde janeiro de 1980, até abril de 2011, do banco de dados **ie**

```
> attach(ie)
> names(ie)
[1] "anomes" "tcc" "tcv" "bc" "exp" "smr" "igpd"
> smr1=auto.arima(smr)
> smr.ts=ts(smr,start=c(1980,1),frequency=12)
> smr2=auto.arima(smr.ts)

> summary(smr1) # Modelo ARIMA
Series: smr
ARIMA(5,1,5)

Coefficients:
      ar1    ar2    ar3    ar4    ar5   ma1    ma2  ma3   ma4    ma5
    -0.475 -0.551 -0.942 -0.487 -0.509 0.044 -0.09 0.603 0.228 -0.165
s.e. 0.086  0.081  0.029  0.083  0.072 0.096  0.08 0.070 0.098  0.074

sigma^2 estimated as 950.2:  log likelihood = -1819.82
AIC = 3661.64   AICc = 3662.36   BIC = 3704.83

In-sample error measures:
        ME       RMSE        MAE        MPE       MAPE
 0.3716551 30.7837859 20.7415191 -0.5311600  5.7073384

> summary(smr2) # Modelo SARIMA
Series: smr.ts
ARIMA(4,1,2)(2,0,1)[12]

Coefficients:
        ar1    ar2     ar3    ar4     ma1     ma2   sar1    sar2    sma1
    -0.0434 0.2968 -0.0186 0.3641 -0.3192 -0.6237 1.1152 -0.2260 -0.5691
s.e. 0.0896 0.0641  0.0516 0.0518  0.0881  0.0805 0.1775  0.1273  0.1638

sigma^2 estimated as 843:  log likelihood = -1799.3
AIC = 3618.6   AICc = 3619.2   BIC = 3657.87
```

As previsões correspondentes para os meses de maio a outubro de 2011 são:

```
> predict(smr1,6)
Time Series:
Start = 377
End = 382
Frequency = 1
pred: 541.4022 536.4573 550.9956 550.1748 550.2760 541.2245
se  : 30.82479 35.46528 35.69889 36.44613 40.53671 41.60850

> predict(smr2,6)
2011
```

```
          May      Jun      Jul      Aug      Sep      Oct
pred: 545.9977 545.2510 546.5741 544.9905 544.5428 540.7435
se  :  29.03376 34.43006 35.70931 36.24331 39.00929 40.13379
```

### 9.7.7 Processos ARIMA fracionários

O modelo ARIMA$(p, d, q)$ é algumas vezes denominado como processo geral com diferenciação fracionária (General Fractional Differenced process) (ARFIMA) quando o parâmetro $d$ (grau de diferenciação) assume valores não inteiros. Este modelo é utilizado para dados de séries hidrológicas. A mais importante característica do modelo ARFIMA$(p, d, q)$ e a propriedade de longa dependência "long-memory"quando $d \in (0.0, 0.5)$, e pequena dependência "short-memory"
quando $d \in (-0.5, 0.0)$.

Longa dependência (ou persistência) é caracterizada pela presença, na série, de uma significante dependência entre as observações mesmo para distantes "lags". Esta característica tem sido observada em séries de diferentes áreas de estudos tais como, meteorologia, astronomia, hidrologia e economia[7]. No modelo ARIMA$(p, d, q)$, $d \in (-0.5, 0.5)$, as características de longa e curta dependência podem ser notadas pelo comportamento da função espectral e da função de autocorrelação.

No domínio do tempo, as autocorrelações decaem lentamente de uma forma hiperbólica, isto e, $\rho_k \sim k^{-d}$, o oposto das autocorrelações produzidas pelo modelo ARMA$(p, q)$ (Box and Jenkins (1976)), as quais têm decaimento exponencial $\rho_k \sim a^{-d}$, $0 < a < 1$. No domínio da frequência a função espectral tende a infinito quando a frequência se aproxima de zero, i.e., $f(\lambda) \to \infty$ quando $\lambda \to 0$.

Se $d \in (-0.5, 0.0)$, o processo tem a propriedade de curta dependência. No domínio da frequência isto é indicado pelo comportamento da função espectral que se aproxima de zero quando a frequência também se aproxima de zero. No domínio do tempo a função de autocorrelação poderá exibir dependências negativas entre observações distantes, portanto, o tipo de dependência e essencialmente determinado pelo valor fracionário de $d$. Se $d = 0$, o processo é chamado de memoria curta[12]. Formalmente o processo ARFIMA$(p, d, q)$ é definido como a seguir:

Seja $d \in \mathbb{R}$. $\{X_t\}$ segue um processo ARFIMA$(p, d, q)$ se satisfaz a equação em diferenças da forma

$$\Phi(B)(1 - B)^d X_t = \Theta(B)\epsilon_t, \tag{9.41}$$

com $\Phi(z) = 1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$ e $\Theta(z) = 1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$, $\{\epsilon_t\}$ sendo um processo ruído branco com média 0 e variância $\sigma_\epsilon^2$. O filtro de diferenciação fraccionaria $(1 - B)^d$ é definido pela expansão binomial

$$(1-B)^d = \sum_{j=0}^{\infty} \pi_j B^j,$$

onde $\pi_j = \frac{\Gamma(j-d)}{\Gamma(j+1)\Gamma(-d)}, j = 0, 1, \ldots,$ e $\Gamma(.)$ é a função gamma definida em $\mathbb{R} - \mathbb{Z}^-$ :

$$\Gamma(x) = \begin{cases} \int_0^\infty t^{x-1} \exp^{-t} dt, & x > 0; \\ \infty, & x = 0; \\ x^{-1}\Gamma(1+x), & x < 0. \end{cases}$$

Quando $d \in (-0.5, 0.5)$ e as raízes dos polinômios $\Phi(z) = 0$ e $\Theta(z) = 0$ são não-comuns e estão fora do círculo unitário, o processo definido em (9.41) é estacionário e invertível e com função de densidade espectral dada por

$$f_{ARFIMA}(\lambda) = f_{ARMA}(\lambda) \left\{ 2sin\left(\frac{\lambda}{2}\right) \right\}^{-2d}, \qquad \lambda \in [-\pi, \pi], \tag{9.42}$$

onde $f_{ARMA}(\lambda)$ está definida em (9.40).

Hosking[16], mostrou que valores $d \geq 0.5$ $\{X_t\}$ é não estacionário e invertível, e ainda que séries com representação $ARFIMA(p, d, q)$ com $d \in (0, 0.5)$ apresentam estacionariedade e memória longa. Assim, no que se segue nós consideraremos o processo ARFIMA(p, d, q) $d \in (0, 0.5)$.

### 9.7.8 Métodos de estimação do parâmetro $d$ em modelos $ARFIMA(p, d, q)$

Existem vários estimadores do parâmetro de diferenciação fracionária $d$, que podem ser classificados em semi-paramétricos e paramétricos. Os primeiros envolvem a estimação simultânea dos parâmetros do modelo, em geral utilizando o método de máxima verossimilhança. Nos procedimentos semi-paramétricos, a estimação dos parâmetros do modelo é realizada em dois passos: primeiro estima-se o parâmetro de memória longa $d$, por exemplo, através de um modelo de regressão do logaritmo da função periodograma e, posteriormente, estimam-se os parâmetros auto-regressivos e de médias móveis. O estimador mais popular dentro dessa classe é o estimador proposto por Geweke e Porter-Hudak (GPH); variantes foram desenvolvidas por Reisen [31], Robinson, entre outros [9].

#### Estimação de $d$ usando a função periodograma (GPH)

Seja $f(\lambda_j)$ a função definida em (9.42), para $\lambda_j = \frac{2/pij}{n}, j = 0, 1, \ldots, \lfloor \frac{n}{2} \rfloor$, onde $n$ é o tamanho amostral. O logaritmo de $f(\lambda_j)$ pode ser escrito como:

$$\ln f(\lambda_j) = \ln f_u(0) - d \ln \left\{ 2 \sin \left( \frac{\lambda_j}{2} \right) \right\}^2 + \ln \frac{f_u(\lambda_j)}{f_u(0)}, \tag{9.43}$$

onde $f_u(\lambda)$ é a densidade espectral de $U_t = (1-B)^d\{X_t\}$. Aqui, $\lfloor . \rfloor$ denota a parte enteira.

Geweke e Porter-Hudak sugerem um estimador semi-paramétrico de $d$, adicionando $\ln I(\lambda)$ em ambos os lados da equação (9.43) e considerando as frequências próximas de zero, obtendo a aproximação:

$$\ln I(\lambda_j) \approx \ln f_u(0) - d \ln \left\{ 2 \sin \left( \frac{\lambda_j}{2} \right) \right\}^2 + \ln \frac{I(\lambda_j)}{f(\lambda_j)}, \tag{9.44}$$

que sugere a equação de regressão dada por

$$\ln I(\lambda_j) \approx \beta_0 + \beta_1 \ln \left\{ 2 \sin \left( \frac{\lambda_j}{2} \right) \right\}^2 + \epsilon_j, \qquad j = 1, 2, \ldots, g(n),$$

onde $\beta_0 = ln f_u(0), \beta_1 = -d$ e $g(n)$ é a amplitude de banda (bandwidth), que corresponde ao número de frequências utilizadas na regressão. Os erros $\{\epsilon_j\}$ são assintoticamente independentes com distribuição Gumbel de média 0 e variância $\frac{\pi}{6}$ (ver [31]). O estimador GPH é dado por

$$d_{GPH} = -\frac{\sum_{i=1}^{g(n)} (x_i - \bar{x}) \ln I(\lambda_j)}{\sum_{i=1}^{g(n)} (x_i - \bar{x})^2} \tag{9.45}$$

onde $x_i = ln\{2sin(\frac{\lambda_j}{2})\}^2$. Geweke e Porter-Hudak sugerem considerar $g(n) = n^\alpha, 0 < \alpha < 1$. Algumas propriedades assintóticas do estimador dado em (9.45) foram derivadas por Hurvich, Deo e Brodsky e Velasco.

### Estimação de $d$ usando a função periodograma suavizado ($\mathbf{GPH}_S$)

Tomando à expressão (9.42) e usando resultados assintóticos [31].

Para $\ln\{I_s(\lambda)/f(\lambda)\}$ pode-se escrever a equação de regressão da forma

$$\ln f_s(\lambda_j) = \ln f_u(0) - d \ln \left\{ 2 \sin \left( \frac{\lambda_j}{2} \right) \right\}^2 + \ln \frac{f_s(\lambda_j)}{f(\lambda_j)} + \ln \frac{f_u(\lambda_j)}{f_u(0)}, \tag{9.46}$$

Restringindo o domínio de $j$, $1 < j < g(n)$, e escolhendo a função $g(n)$ como anteriormente, pode-se reescrever (9.46):

$$\ln I_s(\lambda_j) \approx \ln f_u(0) - d \ln \left\{ 2 \sin \left( \frac{\lambda_j}{2} \right) \right\}^2 + \ln \frac{f_s(\lambda_j)}{f(\lambda_j)}, \tag{9.47}$$

A equação (9.47) é uma forma aproximada da equação de regressão linear simples, isto é,

$$y_j = a + bx_j + \epsilon_j \qquad j = 1, 2, \ldots, g(n), \tag{9.48}$$

onde $y_j = \ln f_s(\lambda_j)$, $b = d$, $x_j = \ln(2\sin(\lambda_j/2))^2$, $\epsilon_j = \ln(f_s(\lambda_j)/f(\lambda_j))$ e $a = \ln f_u(0)$.

Como notado anteriormente, quando $-0.5 < d < 0.0$ no modelo ARIMA$(p, d, q)$, Os erros $\{\epsilon_j\}$ são assintoticamente independentes com distribuição Gumbel de média 0 e variância $\frac{\pi}{6}$ [31].

O estimador de $d$ obtido pelo método de regressão utilizando a função periodograma suavizado com a janela Parzen é dado por:

$$d_{SP} = -\frac{\sum_{i=1}^{g(n)}(x_i - \bar{x}) \ln I_s(\lambda_j)}{\sum_{i=1}^{g(n)}(x_i - \bar{x})^2} \tag{9.49}$$

onde $x_i = ln\{2sin(\frac{\lambda_j}{2})\}^2$.

**Estimação de máxima verossimilhança.**

A função de verossimilhança de $\mathbf{X} = (X_1, ..., X_n)$ proveniente de um processo ARFIMA$(p, d, q)$ pode ser expressa na forma

$$L(\eta, \sigma_a^2) = (2\pi\sigma_a^2)^{-1/2} \exp\left[-\frac{1}{\sigma_a^2}\sum_{j=1}^{n}(X_j - \hat{X}_j)^2/r_{j-1}\right] \tag{9.50}$$

em que $\eta = (d, \phi_1, ..., \phi_p, \theta_1, ..., \theta_q)$, $\hat{X}_j$, $j = 1, ..., n$, são as previsões um passo à frente e $r_{j-1} = (\sigma_a^2)^{-1}E(X_j - \hat{X}_j)^2)$. Os estimadores de máxima verossimilhança dos parâmetros são dados por

$$\hat{\sigma}_{MV}^2 = n^{-1}S(\hat{\eta}_{MV}), \tag{9.51}$$

onde

$$S(\hat{\eta}_{MV}) = \sum_{j=1}^{n}(X_j - \hat{X}_j)^2/r_{j-1}$$

e $\hat{\eta}_{MV}$ é o valor de $\eta$ que minimiza

$$\ell(\eta) = \ln(S(\eta|n)) + n^{-1}\sum_{j=1}^{n}\ln r_{j-1}.$$

entretanto, o cálculo de $\ell(\eta)$ é bastante lento. Um procedimento alternativo é considerar uma aproximação para $\ell(\eta)$ dada por

$$\ell(\eta) \simeq \ell_*(\eta) = \ln \frac{1}{n} \sum_j \frac{I_n(w_j)}{2\pi f(w_j;\eta)}, \tag{9.52}$$

em que

$$I_n(w_j) = \frac{1}{n} \left| \sum_{t=1}^{n} X_t e^{-itw_j} \right|$$

é o periodograma dos dados,

$$f(w_j;\eta) = \frac{\sigma_a^2 |1 - \theta_1 e^{-iw_j} - ... - \theta_q e^{qiw_j}|^2}{2\pi |1 - \phi_1 e^{iw_j} - ... - \phi_p e^{piw_j}|^2} \cdot |1 - e^{iw_j}|^{-2}$$

é a função densidade espectral do processo $X_t$ e $\sum_j$ é a soma sobre todas as frequências de Fourier, $w_j = 2\pi j/n \in (-\pi, \pi]$.

Hannan e Fox e Taqqu mostram que:

1. O estimador $\hat{\eta}_{MV}$ que minimiza (9.52) é consistente;
2. se $d > 0$,
$$\hat{\eta} \mathcal{N}(\eta, n^{-1} A^{-1}(\eta)),$$
em que $A(\eta)$ é uma matriz de ordem $(p+q+1)\times(p+q+1)$ com $(j,k)$-ésimo elemento dado por
$$A_{jk}(\eta) = \frac{1}{4\pi} \int_{-\pi}^{\pi} \frac{\partial \ln f(\lambda;\eta)}{\partial \eta_j} \frac{\partial \ln f(\lambda;\eta)}{\partial \eta_k} d\lambda;$$
3. a variância é estimada por
$$\hat{\sigma}^2_{MV} = \frac{1}{n} \sum_j \frac{I_n(w_j)}{2\pi f(w_j^{\cdot}, \hat{\eta}_{MV})}.$$

Para o exemplo aumentamos o tamanho da série da cotação da bovespa desde 4 de julho de 1994 até 21 de outubro de 2011, totalizando 4286 observações de dias úteis. Para a estimação dos modelos ARFIMA usando o método de Geweke e Porter-Hudak (GPH), usamos os dados da cotação em primeira diferença. Uma segunda série com os retornos dos índices da bovespa também é analisada.

```
> fdGPH(diff(cotacao), bandw.exp = 0.5)
d
[1] 0.1009003

sd.as
[1] 0.08852625
```

```
sd.reg
[1] 0.08677864

> fdGPH(retorno, bandw.exp = 0.5)
d
[1] -0.2195567

sd.as
[1] 0.08852625

sd.reg
[1] 0.0973079
```

Concluímos que usando o método de Geweke e Porter-Hudak (GPH), identificamos que as primeiras diferenças da cotação dos indices da Bovespa segue um processo de memoria longa. Para os retornos da bovespa identificamos um processo de memoria curta. Usando o método de Valderio e Reisen:

```
> fdSperio(diff(cotacao), bandw.exp = 0.5, beta = 0.9)
d
[1] 0.04624686

sd.as
[1] 0.03335958

sd.reg
[1] 0.04057482

> fdSperio(retorno, bandw.exp = 0.5, beta = 0.9)
d
[1] -0.1009049

sd.as
[1] 0.03335958

sd.reg
[1] 0.03422587
```

Com este método de estimação confirmamos que as diferenças da cotação dos índices da Bovespa apresentam memoria longa, e os retronos memoria curta. Para estimar os paramêtros do ARFIMA(p,d,q) automaticamente, podemos usar o algoritmo de Hyndman-Khandakar(2008) para selecionar $p$, e $q$, alem de uma estimação de máxima verosimilhança baseado em Haslett e Raftery(1989) para a inclusão do parâmetro $d$. A função usada é arfima() do pacote forecast.

```
> arfima2=arfima(cotacao, drange = c(0, 0.5))
```

```
> summary(arfima2)

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
d       0.49991    0.00011 4544.77   <2e-16 ***
ma.ma1 -0.66686    0.01469  -45.40   <2e-16 ***
ma.ma2 -0.59540    0.01713  -34.76   <2e-16 ***
ma.ma3 -0.46457    0.01470  -31.61   <2e-16 ***
ma.ma4 -0.37472    0.01707  -21.95   <2e-16 ***
ma.ma5 -0.24192    0.01814  -13.34   <2e-16 ***

[d.tol = 0.0001221, M = 100, h = 5.183e-05]
Log likelihood: -4918 ==> AIC = 9838.226 [1 deg.freedom]

> predict(arfima2,5)
     Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
4287       55.30969 54.33977 56.27961 53.82633 56.79306
4288       55.61433 54.12389 57.10478 53.33489 57.89378
4289       55.44609 53.49153 57.40066 52.45685 58.43534
4290       55.39996 53.06094 57.73899 51.82274 58.97719
4291       55.44727 52.78455 58.10999 51.37499 59.51955
> plot.ts(cotacao)
> lines(fitted(arfima2),col=2)
> mape2=mean(abs((cotacao-fitted(arfima2))/cotacao))*100
[1]2.53814

> arfima3=arfima(retorno, drange = c(-0.5, 0))
> summary(arfima3)

Coefficients:
       Estimate
d         0.037
ar.ar1    0.674
ar.ar2    0.409
ar.ar3   -0.548
ma.ma1    0.696
ma.ma2    0.440
ma.ma3   -0.515
[d.tol = 0.0001221, M = 100, h = 0.0001052]
Log likelihood:  9983 ==> AIC = -19964.86 [1 deg.freedom]
> predict(arfima3,5)
     Point Forecast       Lo 80      Hi 80       Lo 95      Hi 95
4286   0.0024360235 -0.02775225 0.03262430 -0.04373295 0.04860500
4287   0.0022052977 -0.02798648 0.03239708 -0.04396905 0.04837964
4288   0.0013011508 -0.02890215 0.03150445 -0.04489081 0.04749311
4289   0.0010178410 -0.02924350 0.03127918 -0.04526288 0.04729857
```

```
4290   0.0004637973 -0.02983511 0.03076270 -0.04587438 0.04680197
```

Para construir modelos alternativos ARFIMA(p,d,q), podemos usar a função fracdiff(), do pacote fracdiff.

```
> fcot3=fracdiff(cotacao,nar=0,nma=9)
> summary(fcot3)

Call:
  fracdiff(x = cotacao, nar = 0, nma = 9)

Coefficients:
     Estimate Std. Error  z value Pr(>|z|)
d    0.499814   0.000258 1937.634  < 2e-16 ***
ma1 -0.672774   0.015182  -44.313  < 2e-16 ***
ma2 -0.623987   0.020828  -29.959  < 2e-16 ***
ma3 -0.554392   0.020144  -27.522  < 2e-16 ***
ma4 -0.526709   0.015312  -34.398  < 2e-16 ***
ma5 -0.476342   0.017948  -26.541  < 2e-16 ***
ma6 -0.401625   0.020994  -19.130  < 2e-16 ***
ma7 -0.288952   0.018109  -15.956  < 2e-16 ***
ma8 -0.232043   0.019893  -11.665  < 2e-16 ***
ma9 -0.103939   0.020978   -4.955 7.25e-07 ***

[d.tol = 0.0001221, M = 100, h = 4.888e-05]
Log likelihood: -4638 ==> AIC = 9278.332 [1 deg.freedom]
Log likelihood: -4638 ==> AIC = 9278.332 [1 deg.freedom]
> predict(fcot3,5)
     Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
4287       55.24700 54.31520 56.17881 53.82193 56.67208
4288       55.15348 53.71748 56.58948 52.95731 57.34965
4289       55.14661 53.24669 57.04652 52.24094 58.05228
4290       55.13461 52.81360 57.45561 51.58494 58.68428
4291       55.29935 52.57981 58.01890 51.14017 59.45854

> mape3=mean(abs((cotacao-fitted(fcot3))/cotacao))*100
> mape3
[1] 2.218838
```

Para a cotação dos indices da bovespa o modelo alternativo (fcot3), apresentou melhor MAPE que o modelo arfima2. Uma sugestão para comparar modelos dos retornos da bovespa é usar o erro relativo percentual. Para simular um modelo arfima, podemos usar a função arfima.sim(), do pacote forecast, exemplo:

```
x <- fracdiff.sim( 1000, ma = -.98, d = .48)$series
```

## 9.8 Modelos não lineares estruturais

Estes modelos são apropriadas para séries financeiras onde a variância condicional evolui no tempo, e modelos lineares do tipo ARIMA, não são adequados para descrever o comportamento de estas séries. Modelos com alta volatilidade mais conhecidos são o ARCH (Heterocedasticidade condicional autorregressiva) e o GARCH ( generalização dos ARCH). iniciaremos apresentando um teste autocorrelação residual.

### 9.8.1 Teste de Ljung e Box

Após estimar o modelo, precisamos identificar o comportamento dos resíduos estimados do modelo. Um teste para as autocorrelações dos resíduos estimados, que a pesar de não detectar quebras especificas no comportamento do ruído branco, pode indicar se esses valores são muito altos é dado pela estatística:

$$Q(k) = n(n-2)\sum_{j=1}^{k}\frac{\hat{r}_j^2}{(n-j)} \tag{9.53}$$

A qual terá uma distribuição $\chi^2$ com $k-p-q$ graus de liberdade. A distribuição assintótica é obtida sobre a hipóteses de que $k = k(n) \longrightarrow \infty$ quando $n \longrightarrow \infty$. Uma variação é o teste de Box-Pierce [27].

### 9.8.2 ARCH

Utilizado para modelar séries financeiras que apresentam a variância condicional evoluindo no tempo, e em consequência os modelos lineares tipo ARIMA não são adequados para descrever esse tipo de comportamento. Um destes modelos não lineares é o ARCH (autorregresive conditional heterocedasticity), estes modelos são não lineares no que se refere a variância. O objetivo é modelar a variância condicional conhecida como volatilidade, comumente usa-se os retornos. Manifestações da volatilidade numa série financeira.

1. A volatilidade aparece em grupos, de maior ou menor variabilidade.
2. A volatilidade evolui continuamente no tempo podendo ser considerada estacionária.
3. Ela reage positivamente ou negativamente, podendo ser considerada estacionaria.

Considere umaa série de retornos $X_t$ é não correlacionada serialmente, não entanto sua volatilidade depende dos retornos passados considerando uma função quadrática. A definição é dado por:

$$X_t = \sqrt{h_t}\epsilon_t, h_t = \alpha_0 + \sum_{i=1}^{r}\alpha_i X_{t-i}^2 \tag{9.54}$$

Onde $\epsilon_t$ é uma sequência de variáveis aleatórias independentes e identicamente distribuídas (iid) com média zero e variância um. $\alpha_0 > 0, \alpha_i \geq 0, i > 0$. Na pratica $\epsilon_t$, tem distribuição normal padrão ou distribuição t. Para implementar no R, temos que alertar que estes modelos pertencem a uma classe denominada de 4, que fornece as saídas com @ ao invés de $ próprios da classe 3 desenvolvidas até aqui.

Um primeiro passo para a construção destes modelos é ajustar modelos ARMA, para remover a correlação serial na série original, e posteriormente assumir que nosso resíduo é um ARCH(r), ou $\epsilon \approx ARCH(r)$.
Para implementar um exemplo com os retornos da bovespa, consideraremos a função garchFit(), do pacote fGarch. Consideremos o preço de um ativo financeiro, como o índice de cotação da Bovespa. A Idea é modelar a série dos retornos por um ARMA(1,0), e os resíduos estimados são gerados por um ARMA(2,0). O retorno desta série financeira usando o R com a função garchFit() do pacote fGarch é dada por: A idéia é sugerir que a série dos retornos segue um ARMA(0,1), e os resíduos um ARCH(2)

```
library(fGarch)
arch=garchFit(~arma(0,1)+garch(2,0),rt)
summary(arch)
resumo=arch@fit
resumo
plot(arch)
```

nesta saída resumo, estão incluídas todas as estatísticas correspondentes ao modelo, incluindo convergência, iterações, a log-verosimilhança os coeficientes do modelo, a matriz hessiana das estimativas dos parâmetros, critérios de informação, a série estimada dos retornos, alem da série estimada dos resíduos. Para o plot, basta selecionar a série estimada de interesse.

### 9.8.3 GARCH

Este modelo é mais parcimonioso que os modelos ARCH, e é definido por

$$X_t = \sqrt{h_t \epsilon_t}, h_t = \alpha_0 + \sum_{i=1}^{r} \alpha_i X_{t-i}^2 + \sum_{j=1}^{s} h_{t-j} \tag{9.55}$$

Onde $\epsilon_t$ é normal ou t-student, alem disso uma sequencia de variáveis aleatórias independentes e identicamente distribuídas (iid) com média zero e variância um. $\alpha_0 > 0, \alpha_i \geq 0, \beta_j \geq 0, \sum_i = 1^q(\alpha_i + \beta_j) < 1, q = \max(r, s)$. A função usada é **garch()**, do pacote **tseries** e o modelo padrão do R é um GARCH(1,1), com a seguinte estrutura:

$$y_t = \sigma_t \nu_t, \qquad \nu \approx N(0,1) i.i.d., \sigma_t = \omega + \alpha y_{t-1}^2 + \beta \sigma_{t-1}^2, \qquad \omega, \alpha > 0, \beta \geq 0 \tag{9.56}$$

Para aplicar o modelo mais simples, para a série de retornos da bovespa usamos os seguintes comandos:

```
> garchr=garchFit(~arma(1,0)+garch(2,3),rt)
> plot(residuals(rar))
> plot(residuals(rarima))
> garchr=garchFit(~arma(1,0)+garch(1,1),rt)

> ajustegarch=garchr@fit$series$h
> predict(ajustegarch,5)
     Point Forecast        Lo 80        Hi 80         Lo 95        Hi 95
4286   0.0003738443 0.0002657738 0.0004819149  2.085646e-04 0.0005391241
4287   0.0003771122 0.0002227740 0.0005314504  1.410723e-04 0.0006131521
4288   0.0003803800 0.0001886256 0.0005721345  8.711687e-05 0.0006736432
4289   0.0003836479 0.0001587702 0.0006085255  3.972721e-05 0.0007275685
4290   0.0003869157 0.0001314454 0.0006423860 -3.792403e-06 0.0007776238
```

Um modelo mais elaborado, é considerar que a série dos retornos segue um ARMA(10,0), e os resíduos um GARCH(1,1), sugerido pot Moretin e Toloi [27] é:

```
garchmoretin=garchFit(formula = ~arma(10, 0) + garch(1, 1), data = rt)
> summary(garchmoretin)

Title:
 GARCH Modelling

Call:
 garchFit(formula = ~arma(10, 0) + garch(1, 1), data = rt)

Mean and Variance Equation:
 data ~ arma(10, 0) + garch(1, 1)
<environment: 0x03740cc0>
 [data = rt]

Conditional Distribution:
 norm

Coefficient(s):
   mu       ar1        ar2       ar3       ar4       ar5        ar6
 1.55e-03  8.77e-03 -2.43e-02 -3.16e-02 -3.27e-02 -2.98e-02  -4.76e-03
   ar7       ar8        ar9       ar10     omega      alpha1     beta1
-1.36e-02  1.54e-02  1.16e-02  4.16e-02  1.28e-05  1.22e-01   8.54e-01

Std. Errors:
 based on Hessian

Error Analysis:
         Estimate  Std. Error  t value Pr(>|t|)
```

```
mu      1.545e-03   2.791e-04    5.537 3.08e-08 ***
ar1     8.767e-03   1.629e-02    0.538  0.59042
ar2    -2.432e-02   1.603e-02   -1.517  0.12926
ar3    -3.155e-02   1.585e-02   -1.991  0.04650 *
ar4    -3.272e-02   1.599e-02   -2.046  0.04075 *
ar5    -2.985e-02   1.587e-02   -1.881  0.06000 .
ar6    -4.755e-03   1.583e-02   -0.300  0.76388
ar7    -1.357e-02   1.557e-02   -0.872  0.38335
ar8     1.542e-02   1.555e-02    0.991  0.32147
ar9     1.162e-02   1.546e-02    0.751  0.45252
ar10    4.164e-02   1.526e-02    2.729  0.00636 **
omega   1.269e-05   2.455e-06    5.168 2.36e-07 ***
alpha1  1.224e-01   1.160e-02   10.559  < 2e-16 ***
beta1   8.544e-01   1.404e-02   60.833  < 2e-16 ***

Log Likelihood:
 10752.44    normalized:  2.509322

Description:
 Fri Nov 04 16:34:36 2011 by user: user

Standardised Residuals Tests:
                                Statistic  p-Value
 Jarque-Bera Test   R    Chi^2  426.7988   0
 Shapiro-Wilk Test  R    W      0.9884059  0
 Ljung-Box Test     R    Q(10)  6.86544    0.7380843
 Ljung-Box Test     R    Q(15)  10.90783   0.7591006
 Ljung-Box Test     R    Q(20)  14.24748   0.8177295
 Ljung-Box Test     R^2  Q(10)  15.45189   0.1164347
 Ljung-Box Test     R^2  Q(15)  16.84624   0.3281391
 Ljung-Box Test     R^2  Q(20)  23.55899   0.262187
 LM Arch Test       R    TR^2   15.2171    0.2297781
Information Criterion Statistics:
      AIC       BIC       SIC      HQIC
-5.012109 -4.991320 -5.012130 -5.004766
> gm=garchmoretin@fit$series$h
> predict(gm,5)
     Point Forecast        Lo 80        Hi 80         Lo 95        Hi 95
4286   0.0003759518 0.0002674082 0.0004844955  2.099486e-04 0.0005419550
4287   0.0003789948 0.0002228952 0.0005350944  1.402611e-04 0.0006177286
4288   0.0003820378 0.0001876322 0.0005764435  8.472005e-05 0.0006793556
4289   0.0003850809 0.0001568135 0.0006133482  3.597610e-05 0.0007341856
4290   0.0003881239 0.0001286022 0.0006476455 -8.780257e-06 0.0007850280

plot(ecdf(residuals(garchmoretin)))
x=seq(-0.3,0.3,0.01)
```

```
lines(x,pnorm(x,mean=mean(residuals(garchmoretin)), sd=sd(residuals(garchmoretin)))
```

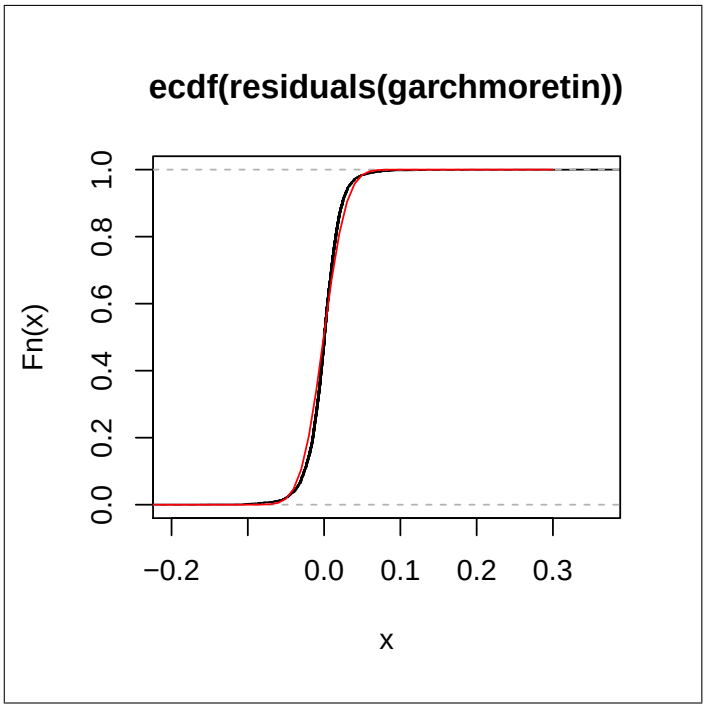



Figura 39. Distribuição dos resíduos padronizados dos retornos.

Interpretação: o ajustamento do modelo acima verificamos que os parâmetros do modelo são significantes. Os valores da estatística de Ljung-Box para os resíduos padronizados são dados por Q(10)=6,86 (0,74) e Q(20)=14,25 (0,82) enquanto que para os quadrados dos resíduos Q(10)=15,45 (0,12) e Q(20)=23,55 (0,23). Analisando os níveis descritos podemos concluir que o modelo é adequado para modelar os retornos diários da IBovespa.

## 9.9 Raízes unitárias

Para modelos com tendência determinística é possível encontrar uma raiz unitária, esta raiz unitária esta relacionada com o ruído branco do modelo. Dizemos que um ruído branco é I(0), se segue um processo autorregressivo de primeira ordem AR(1). Para gerar os dados considere o ruido branco $\epsilon_t$ dado por.

$$\epsilon_t = \alpha_0 \epsilon_{t-1} + \nu_t, \qquad \nu \approx N(0, \sigma^2) \tag{9.57}$$

O objetivo é testar as seguintes hipóteses:
$H0 : \alpha_0 = 1 \rightarrow \epsilon_t \approx I(1)$
$Ha : |\alpha| < 1 \rightarrow \epsilon_t \approx I(0)$

Há varias estatísticas para testar estas hipóteses, em particular as mais comuns são Phillips-Perron, Schmidt-Phillips e Dickey e Fuller, para implementar este teste considere o consumo de energia elétrica hidráulica na industria no Brasil qde. - GWh X 100 - Eletrobrás, no período de fevereiro de 1976 a dezembro de 2008, extraídas do IPEADATA:

```
> cee=read.table(file.choose(),header=T)
> colnames(cee)="consumo"
> attach(cee)
> plot.ts(consumo)
> cee.ts=ts(consumo,start=c(1976,2),frequency=12)
> plot.ts(cee.ts)
```

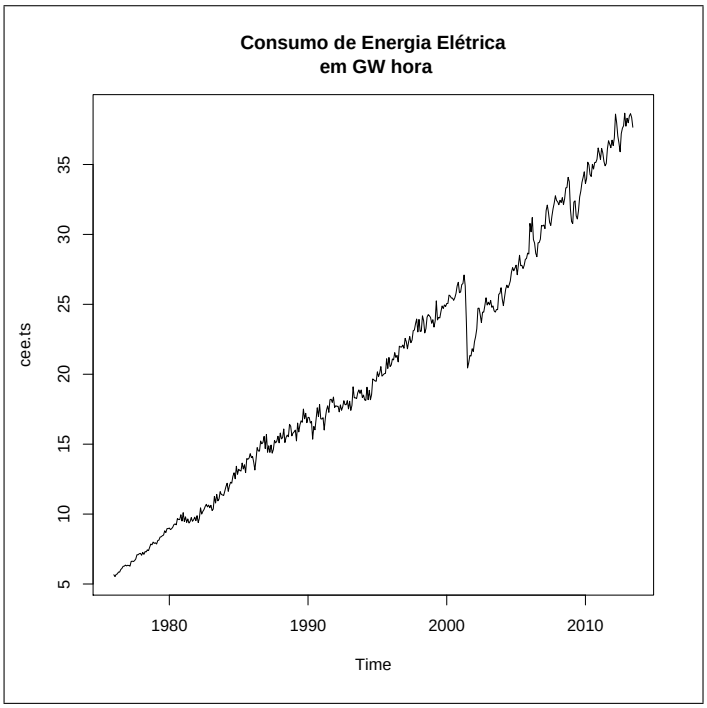



Figura 40. Consumo de Energia em GWh X 100 no Brasil.

```
> library(urca)
> t= ur.pp(cee.ts, signif=0.05) # teste de Phillips-Perron
> t= ur.sp(cee.ts, signif=0.05) # teste de Schmidt-Phillips
> t= ur.df(cee.ts, signif=0.05) # teste de Dickey e Fuller
```

A partir das saídas dos testes, não há evidencias para rejeitar H0. Por tanto há evidencias da presença de um AR(1).

## 9.10 Exercícios

1. Com os dados de consumo de água construa uma regressão com séries temporais da forma $valor_t = \beta_1 + beta_2 valor_{t-2}$. Sugestão use L(valor,2)

2. Construa a seguinte regressão com séries temporais: $\texttt{bc} = \beta_0 + \beta_1 \texttt{tcv} + \beta_2 \texttt{bc}_{i-3} + \epsilon_i$

3. Realize previsões futuras dos retornos dos índices da Bovespa, usando algoritmos de Holt-Winters.

4. Realize previsões futuras da cotação dos índices da Bovespa usando Holt-Winter multiplicativo, e comparar com o modelo aditivo.

5. Analise o igpdi do Brasil a partir de agosto de 1994 (inicio do plano Real), usando a modelagem Arima e compare-o com o modelo feito na seção correspondente.

6. Use a função **cumsum()** para acumular a série de IGPDI, extraído do IPEADATA, e use a função **auto.arima()**, para identificar o melhor ARIMA, em sequência converta os dados com a função **ts()** e identifique o modelo SARIMA. De suas conclusões.

7. Colete os dados referentes a cotação do dolar diário (deixe a última semana como controle de previsão), construa três modelos ARIMA(0,1,0), ARIMA(0,1,1), e ARIMA(1,1,0). Para estes modelos realize as previsões futuras (última semana), calcule o mape do ajuste da série (intra série) e o mape das previsões futuras (extra série). Quais são suas conclusões.

8. Construa um plot, comparando as três séries estimadas no item anterior.

9. Colete dados referente ao ICMS de Sergipe, deflacione a série com o IGPDI, e identifique qual o melhor modelo ARIMA.

10. Do item anterior construa um modelo alternativo a sugestão sua e compare as previsões internas (mape interno)

11. Construa um banco de dados atualizados de indicadores econômicos, como balança comercial, exportações, taxa de cambio do dólar (compra e venda), indice geral de preços, IGPDI, diretamente do IPEADATA.

12. Com a série atualizada dos indicadores econômicos construa modelos Arima, Sarima, e de Holt Winters.

13. Calcule o mape dos modelos construídos no item anterior.

14. Uma série estritamente crescente (série acumulada), pode seguir um processo de memória longa, justifique sua resposta.

15. Com os dados do IPEADATA, identifique se o processo inflacionário brasileiro é de memória longa.

16. Com os dados da cotação diária dos indices da Bovespa, use a função fracdiff(), para construir um arfima com parâmetros ARMA(9,11) e calcule o MAPE.

17. Teste a raíz unitária do exemplo anterior.

18. Usa a função **po.test()**, do pacote **tseries**, para identificar se há cointegração no processo inflacionario

19. Considere os dados em formato txt, referente a geração mensal de energia elétrica hidráulica no Brasil (GWh x 1.000), no período de 1996.1-2010.12 baixados do IPEADATA. Implemente no R os comandos para: Chamar os dados, disponibilizar em formato visível no R, converter para uma série temporal, plotar a decomposição da série, visualizar o correlograma e correlograma parcial, plotar a série temporal, identificar a integração da série.

20. Ainda com a geração de energia construa dois modelos de séries temporais sendo o melhor ARIMA, com seu respectivo mape (implemente) e 6 previsões na frente, o melhor SARIMA, com seu respectivo mape e 6 previsões na frente, Construa um gráfico apropriado para visualizar a série real e a série do melhor modelo alem das previsões do melhor modelo.

21. Construa os retornos atualizados da cotação da bovespa, e gere um modelo $GARCH(1,2)$.

# Capítulo 10

# Simulações no ambiente R

## 10.1 Desempenho de modelos AR e MA no R

Um dos objetivos das séries ao longo do tempo é identificar o processo gerador desta série temporal. Em situações práticas o processo gerador de uma série pode ser identificado por um caso particular de modelos, como a modelagem ARIMA (auto regressivos e médias móveis integradas), seguindo a metodologia Box e Jenkins, entretanto desconhecemos o desempenho destes modelos usando funções alocadas na plataforma R. Para identificar este desempenho (o qual foi classificado) foram geradas simulações para vários parâmetros de modelos AR e MA, e comparadas com suas respectivas séries teóricas. Este trabalho tenta esclarecer algumas lacunas enquanto ao desempenho das funções: arima.sim() e auto.arima(), no R.
No R, podemos fazer uso destas funções na biblioteca forecast. O interesse é determinar o percentual de acertos das series simuladas com seus correspondentes serie teóricas representam a

### 10.1.1 Classificação

A classificação do desempenho das funções em porcentagem, é dada da seguinte maneira:

1. De 0 a 50 do percentual de acertos, desempenho fraco (F)
2. Maior de 50 a 70 do percentual de acertos, desempenho médio (M),
3. Acima de 70 do percentual o desempenho é ideal ou bom (B).

### 10.1.2 Os cenários de simulação

Os cenários de comparação são vários tamanhos de simulação, vários tamanhos da amostra da série gerada, e vários tamanhos de parâmetros estimados.

Para determinar o desempenho das séries geradas por simulações e comparadas com as séries teóricas, foi implementado algoritmos que geram séries por simulação e retorna a ordem de uma série teórica. A primeira série simulada é um $ARIMA(0,0,3)$, ou em termos simples um MA(3), com parâmetros: $\theta_1 = 0, \theta_2 = 0, \theta_3 = 0.12, 0.24, 0.36, 0.48, 0.6$, o que representa uma série de médias móveis de ordem 3, com 5 parâmetros alternativos de simulação para esta ordem. A série teórica é construída a partir da série simulada, em termos simples usamos a função do melhor ARIMA disponibilizada pelo R.

Para 10 e 100 simulações e com tamanhos da série de 50 e 100 observações geradas, as frequências não revelam a verdadeira ordem teórica de um MA(3), e sim um MA(0), para todos os parâmetros da série simulada exceto quando há 100 simulações e tamanho dá série de 100, considerando os parâmetros, $\theta_3 = 0.36, 0.48, 0.6$. Para 1000 simulações e com tamanhos da série de 1000, as observações geradas por simulação de um MA(3), tem com retorno teórico de um MA(3) para todos os parâmetros simulados. A tabela a seguir identifica o desempenho da função auto.arima().

### 10.1.3 O algoritmo

O algoritmo para um tamanho 1000 de simulação de um MA(3), com tamanhos da série (amostra) de 1000 é apresentado a seguir.

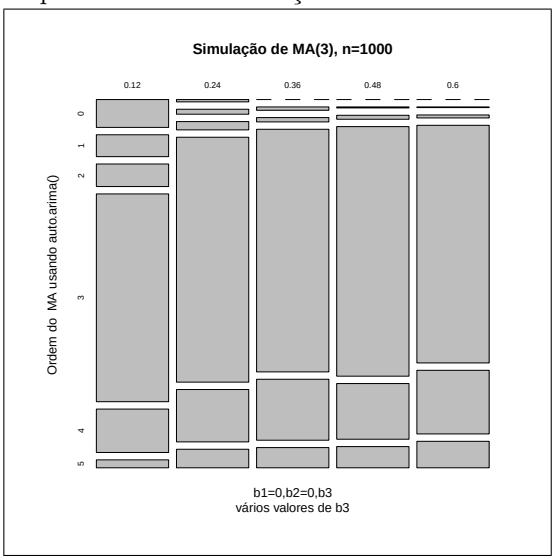
```
set.seed(20)
estMA=matrix(0,nrow=5, ncol=1000)
for (i in 1:5){ ma3=0.12*i
for (j in 1:1000) {
simts=arima.sim(n=100, model=list(ma=c(0,0,ma3)))
estMA[i,j]=auto.arima(simts,start.q=3)$arma[2]
colnames(estMA)=c(1:1000)
}
}
detectar=table(row(estMA), estMA)
dimnames(detectar)=list(ma3=paste(0.12*(1:5)),
Order=paste(0:(dim(detectar)[2]-1)))
detectar
plot(detectar,main="Simulação de MA(3), n=100",sub="vários valores de b3",
ylab="Ordem do  MA usando auto.arima()", xlab="b1=0,b2=0,b3")

Order
ma3  0  1  2  3   4    5

0.12 84  66 68 627  131 24
0.24 7  15 25 739  158 56
0.36 0  10 13 732  184 61
0.48 0  3 12 753  168 64
```

```
0.60 0  2 9   717  192 80
```

### 10.1.4 Os resultados

| MA(3) com parâmetros 0.12,0.24,0.36,0.48,0.6 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 10 simulações | | | 100 simulações | | | 1000 simulações | | |
| $\theta_3$ | 50 | 100 | 1000 | 50 | 100 | 1000 | 50 | 100 | 1000 |
| 0.12 | F | F | F | F | F | M | F | F | M |
| 0.24 | F | F | F | F | F | B | F | F | B |
| 0.36 | F | M | F | F | M | B | F | M | B |
| 0.48 | F | M | F | F | M | B | F | M | B |
| 0.60 | F | M | F | F | M | B | F | M | B |

Figura 41. Desempenho de 1000 simulações de tamanho 1000 de um MA(3).




A segunda série simulada é um ARIMA(1,0,0), ou em termos simples um AR(1), com parâmetros: $\phi_1 = 0.12, 0.24, 0.36, 0.48, 0.6$, o que representa uma série autor regressiva de ordem 1, com 5 parâmetros alternativos de simulação para esta ordem.
Os resultados para 10 e 100 simulações e com tamanhos da série de 50 e 100 observações geradas, as frequências não revelam a verdadeira ordem teórica de um AR(1), e sim um AR(0), para os parâmetros $\phi_1 = 0.12, 0.24, 0.36$ para os outros parâmetros de $\phi_1 = 0.48, 0.6$ as simulações revelaram as maiores frequências como a verdadeira ordem teórica. Para o tamanho da série de 1000 observações geradas por simulação de um AR(1) apenas, $\phi_1 = 0.12$ não tem as maiores frequências como seu retorno teórico de um AR(1). Ver tabela a seguir:

| AR(1) com parâmetros 0.12,0.24,0.36,0.48,0.6 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 10 simulações | | | 100 simulações | | | 1000 simulações | | |
| $\phi_1$ | 50 | 100 | 1000 | 50 | 100 | 1000 | 50 | 100 | 1000 |
| 0.12 | F | F | F | F | F | F | F | F | F |
| 0.24 | F | F | F | F | F | M | F | F | M |
| 0.36 | F | M | M | M | M | B | F | M | M |
| 0.48 | M | F | B | M | B | B | M | B | B |
| 0.60 | M | B | B | M | B | M | M | B | B |

O algoritmo para 1000 simulação com tamanhos da série de 1000 é apresentado a seguir.

```
set.seed(20)
estAR=matrix(0,nrow=5, ncol=1000)
for (i in 1:5){ ar1=0.12*i
for (j in 1:1000) {
simts=arima.sim(n=1000, model=list(ar=c(ar1,0,0)))
estAR[i,j]=auto.arima(simts,start.p=1)$arma[1]
colnames(estAR)=c(1:1000)
}
}

detectar=table(row(estAR), estAR)
dimnames(detectar)=list(ar1=paste(0.12*(1:5)),
Order=paste(0:(dim(detectar)[2]-1)))
detectar
plot(detectar,main="Simulação de AR(1), n=1000",xlab=expression(phi[1]),
ylab="Ordem do  AR usando auto.arima()", sub="vários valores de um AR(1)")
 detectar
    Order
ar1      0   1   2   3   4   5
  0.12 489 379 109  17   6   0
  0.24 233 612 116  28   8   3
  0.36  69 696 180  47   7   1
  0.48  39 714 198  38   9   2
  0.6    9 742 201  36   8   4
```

O gráfico correspondente é apresentado a seguir:

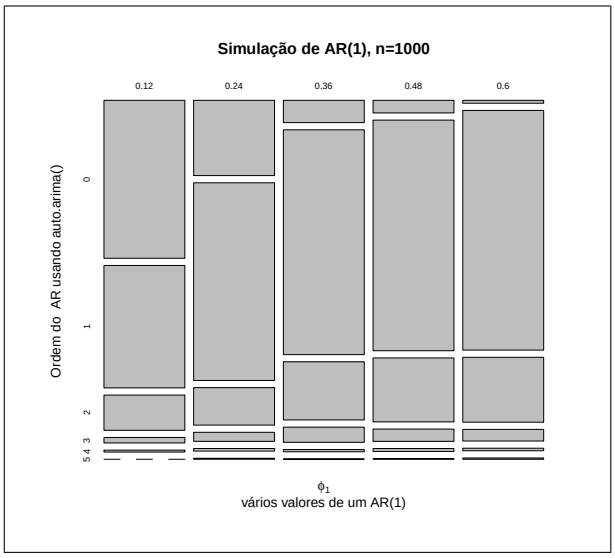



Figura 42. Desempenho de 1000 simulações de tamanho 1000 de um AR(1).

## 10.2 Simulações para determinar o verdadeiro nível de significancia

### 10.2.1 O caso da distribuição da média amostral

Embora simulações não é pertinente para resolver problemas, pois Montecarlo é uma técnica poderosa porem por construção imprecisa [11], há lacunas enquanto ao desempenho de uma determinada plataforma para diferentes tamanhos de amostra. Costuma-se utilizar as simulações de Monte Carlo para avaliar o desempenho de estatísticas ou para procedimentos de estimação em tamanhos de amostras prefixados. Inicialmente precisamos conhecer este desempenho em simulações para pequenas amostras e grandes amostras(assintoticamente). A função do R que esta sendo avaliada é a rnorm(), que inicialmete gera amostras aleatórias normais. A ideia é verificar qual o verdadeiro nível de significância usando como algoritmo a distribuição da média da amostra para vários tamanhos de números aleatórios gerados e para vários tamanhos de simulação. Por exemplo considere o seguinte algorimo de simulação para um nível de significância de 0.05 e para tamanhos de amostra $n = 10$ e para tamanhos de simulação $s = 10$

```
#nível de 0.05 e n=10 usando normal
#s=10
set.seed(10)
alpha=0.05;n=10      # nível de significancia e tamanhos de amostra
s=10               #número de simulações
n.rejeicoes=0        #número de rejeições
```

```
for(i in 1:s)
{
x=rnorm(n)           #simulação da população
est=mean(x)*sqrt(n)/sd(x)    #Estatística amostral
if(abs(est)>qnorm(1-alpha/2))
n.rejeicoes=n.rejeicoes+1  #rejeita acrescenta em 1, se o modulo exede o valor crít
}
verdadeironivel=n.rejeicoes/s #Proporção de rejeicões
n.rejeicoes
2
verdadeironivel
0.2
```

Para verificar o desempenho do verdadeiro nível de significância apresentamos a seguinte tabela:

| Verdadeiros Níveis de signifiância usando rnorm() | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | 0.05 | | | 0.10 | |
| s | n | 10 | 100 | 1000 | 10 | 100 | 1000 |
| 10 | | 0.20 | 0.10 | 0.00 | 0.40 | 0.20 | 0.00 |
| 100 | | 0.17 | 0.03 | 0.06 | 0.22 | 0.08 | 0.13 |
| 1000 | | 0.10 | 0.05 | 0.06 | 0.16 | 0.10 | 0.11 |
| 100000 | | 0.08 | 0.05 | 0.05 | 0.14 | 0.10 | 0.10 |
| 1000000 | | 0.08 | 0.05 | 0.05 | 0.13 | 0.10 | 0.10 |

Visulizando a tabela acima podemos identificar que a a distribuição da média da amostra segue uma distribuição normal para tamanhos de amostra ideais a partir de 100 e o número de simulações acima de 1000, para os níveis de significancia de 0.05 e 0.10.

### 10.2.2 O caso da diferência de médias

Inicialmente construimos o teste t para duas amostras independentes com variâncias iguais (homocedasticas) mas desconhecidas.

```
testet=function(x,y)
{
options(digits=3)
n1=length(x)
n2=length(y)
dm=sqrt(((n1-1)*sd(x)^2+(n2-1)*sd(y)^2)/(n1+n2-2))
est2=(mean(x)-mean(y))/(dm*sqrt(1/n1+1/n2))
return(est2)
}
```

Para determinar o verdadeiro nível de significancia ($\alpha = 0.05$)usamos o seguinte algoritmo:

```
#Programando uma simulação de montecarlo para estimar o
```

```
#Verdadeiro nível de significancia com duas populações normais
#com variâncias iguais mais desconhecidas
set.seed(20)
alfa=.05; n1=10; n2=10  #Declarando alfa, n1 e n2
N=1000           #Número de simulações
n.rejeicao=0       #contador do número de rejeições
for(i in 1:N)
{
x=rnorm(n1) #simulação de uma populaçao normal padrao x
y=rnorm(n2) #simulação de uma populaçao normal padrao y
estatistica.t=testet(x,y) #declara a estatistica t construída.

#estatistica.t=t.test(x,y) #função já definida no R, computa a estatistica t.
#if(abs(estatistica.t$statistic)>qt(1-alfa/2,n1+n2-2))
#if(abs(estatistica.t$statistic)>qnorm(1-alfa/2) # quando n é grande

if(abs(estatistica.t)>qt(1-alfa/2,n1+n2-2))

n.rejeicao=n.rejeicao+1 #rejeita se |T| exede o valor crítico
}
verdadeira.significancia=n.rejeicao/N # estima a proporção de rejeições
verdadeira.significancia
0.055
```

## 10.3 Comparando duas distribuições

Para comparar duas distribuições por exemplo uma normal e uma exponencial, podemos fazer usa da estatística testet já construída paraum número grande de simulações. Ainda podemos gerar o seu correspondente gráfico, com o seguinte algoritmo:

```
#Simulacao para comparar uma normal e uma exponencial com seu respectivo gráfico
#usando a função testet construída
misimulacao=function() testet(rnorm(n1,10,2),rexp(n2,0.1))
vetort=replicate(10000,misimulacao())
plot(density(vetort),main="Densidade da comparação e t(18)",xlim=c(-5,8),ylab="Densidade",yli
curve(dt(x,df=18), add=T)
legend(4,.3,c("simulação","t(18)"),lwd=c(3,1))
```

## 10.4 Exercicios

1. Qual o desempenho das funções auto.arima() e arima.sim() em modelos AR(p) nas 2 situações: p é pequeno (próximo de 0), ou p é grande (acima de 4) em: Simulações pequenas e série grande, simulações grandes e série pequena, simulações grandes e série grande.

2. Qual o desempenho da função rnorm() para avaliar o verdadeiro nível de significância de 0.01. Sugestão use os tamanhos e o número de simulações adotado no algoritmo do verdadeiro nível de significância.

3. Extender o algoritmo da diferência de médias para simular outros tamanhos de amostra e para variancias iguais e conhecidas.

4. extenda o algoritmo anterior para determinar o verdadeiro nível de significancia com duas populações t com gl iguais.

5. extenda o algoritmo anterior para determinar o verdadeiro nível de significancia com duas populacoes t com gl iguais

6. implemente um algoritmo para comparar uma normal e uma gamma com seu respectivo gráfico.

# Capítulo 11

# Geradores de números aleatórios: Testes DIEHARD

## Introdução

A geração de números aleatórios é de vital relevância para o processo de obtenção de dados em simulações. Entretanto, muitos geradores de números pseudo-aleatórios não satisfazem a padrões rigorosos para tal fim. A escolha de tais geradores não deve ser feita ao acaso, sendo necessário um conjunto de testes para auxiliar nessa escolha. Este trabalho descreve o Sistema DIEHARD, que é um rigoroso conjunto de testes para geradores de números pseudo-aleatórios. Os geradores mais utilizados e outros tradicionalmente conhecidos foram testados por esse sistema.

A necessidade de geração de números aleatórios aparece em diversas aplicações práticas. É muito comum, por exemplo, precisarmos inserir dados imprevisíveis e não-determinísticos em uma determinada amostragem ou gerar uma função de distribuição estatística através de sequências de números.

Os geradores de números pseudo-aleatórios produzem tais sequências não sendo, porém, capazes de gerar sequências (algorítmicas) verdadeiramente aleatórias. Fragilidades desse tipo são abordados por Knuth, (1998). Sendo assim, é necessário identificar suas características, avaliando seu desempenho, aleatoriedade e o grau de independência dos valores gerados. Nesta linha de argumentação tem-se a introdução de testes estatísticos projetados para esse fim. A teoria dos testes estatísticos fornece algumas medidas quantitativas para aleatoriedade, não havendo, como esperado, uma quantidade exata desses a serem realizados para se ter a garantia de que uma dada sequência é verdadeiramente aleatória ou não. Quanto mais testes forem executados, maiores serão as chances de ser detectada a aleatoriedade ou não-aleatoriedade da sequência examinada.

A literatura proporciona testes estatísticos tais como: DIEHARD, Crypt-XS, NIST Statistical Test Suite. Neste trabalho foram simulados três testes estatísticos do Sistema DIEHARD, por serem considerados mais rigorosos, Knuth (1998).

## 11.1 Geração de números aleatórios

Um gerador de números aleatórios ideal produz uma sequência, provavelmente, infinita de números não correlacionados e uniformemente distribuídos em quaisquer intervalos numéricos escolhidos. Estes geradores têm um número como "semente", e são definidos de maneira iterativa, devendo, todavia, idealmente, produzir sequências de números com as propriedades encontradas naquelas sequências de números verdadeiramente aleatórias (Krawczyk, 1992). Na geração de números pseudo-aleatórios, muitos geradores podem apresentar diversos problemas, e.g. formação de ciclo, os quais dependem da função iterativa e da semente inicialmente atribuída (Knuth,1998).

## 11.2 Testes estatísticos

Devido à importância dos geradores de números pseudo-aleatórios nas mais diversas áreas de aplicação, pesquisadores desenvolveram testes estatísticos para avaliar a aleatoriedade dos números gerados. Em Soto (1999) são apresentados os testes conforme transcritos na Tabela 1. Este trabalho abordará com maiores detalhes o conjunto de testes DIEHARD.

Tabela 11.1: Conjunto de Testes Estatísticos.

| Origem/Afiliação | Testes Estatísticos |
|---|---|
| D. Knuth/*Stanford University* | *The Art of Computer Programming Semi-numerical Algoritmos* |
| G. Marsaglia/*Florida State University* | DIEHARD |
| H. Gustafson/*Queensland University of Tecnology* | Crypt-XS |
| A. Menezes/CRC Pres, Inc | *Handbook of applied Cryptograpy* |
| A. Rukhin/ NIST ITL | NIST Statistical Test Suide |

## 11.3 O Sistema DIEHARD

Os testes DIEHARD foram utilizados para avaliar alguns dos geradores de números pseudo-aleatórios mais utilizados e conhecidos, na tentativa de comprovar a adequação da utilização desses em aplicações seguras. Este sistema foi

desenvolvido por George Marsaglia, professor e pesquisador do Departamento de Estatísticas e Pesquisas em Computação da Universidade da Flórida. Ele consiste num conjunto de testes estatísticos que procuram identificar padrões e distribuições não uniformes em sequências aleatórias, e.g. Marsaglia, (1984 e 2004).

A seguir é dado um resumo dos testes estatísticos do sistema DIEHARD:

(1) **Espaçamento entre aniversários**
O problema é de se determinar quantas pessoas são necessárias para que pelo menos duas delas tenham a mesma data de aniversário, com probabilidade não inferior a 50%, que em um ano de 365 dias é de vinte e três pessoas. A ideia do teste é considerar como datas de nascimentos números de 8 *bits* e testar o espaçamento entre duas ocorrências.

(2) **Permutação com 5 elementos e sobreposição**
Utiliza-se uma sequência de $10^6$ inteiros de 32 bits. Cada conjunto de 5 inteiros consecutivos desta sequência é classificado de acordo com as 5! inversões de paridade entre as permutações. Procede-se uma inserção de um dígito mais à direita e eliminação do mais à esquerda (sobreposição). Devido a essa sobreposição, mostra-se que em termos estatísticos as classes de inversões passam de 5! para $5! - 21$. O teste é executado duas vezes, tem-se então a geração de duas sequências de $10^6$ inteiros.

(3) **Posto de matrizes 31×31 e 32×32**
Consideram-se 40.000 matrizes de $31 \times 31$ e $32 \times 32$ que são divididas de acordo com os postos das mesmas. Cada um dos elementos das matrizes é formado a partir de números de 31 (32) *bits*, sendo portanto matrizes com entradas 0 ou 1. Pode-se mostrar que para tais matrizes e com entradas aleatórias, os postos das mesmas são, via de regra, maiores ou iguais que 29. O teste considera a frequência de ocorrência dos postos de 29 a 31 (ou 32).

(4) **Posto de matrizes 6×8**
A exemplo do caso anterior, são considerados os postos de matrizes (100.000 delas). De cada seis números inteiros aleatórios de 32 *bits* um dado *byte* (conjunto de 8 *bits*) específico é escolhido e esses *bytes* são utilizados para formar uma matriz 6x8. Procede-se então o cálculo do posto da matriz resultante.

(5) **Fluxo de bits**
O arquivo de entrada é visto como um fluxo de *bits*, da forma $b_1$, $b_2$, etc. Considera-se um alfabeto binário onde cada palavra é composta por 20 letras. A primeira palavra é $b_1 b_2 \cdots b_{20}$, a segunda é $b_2 b_3 \cdots b_{21}$ e assim por diante. O teste conta o número de palavras de 20 letras ausentes numa sequência de $2^{21}$ palavras sobrepostas de 20 letras.

(6) **Sobreposição de pares, quádruplas e DNA**
• No teste de sobreposição de pares são contadas as palavras com 2 letras, de um alfabeto de $2^{10}$ letras, que não aparecem numa sequência de $2^{21}$ palavras sobrepostas.
• O teste de sobreposição de quádruplas é similar ao anterior; entretanto, nele são consideradas palavras de 4 letras de um alfabeto com $2^5$ letras.
• No teste do DNA o alfabeto tem 4 letras e as palavras possuem 10 letras.
• Pode-se mostrar que a quantidade média de palavras "perdidas" tem comportamento igual para os três casos, isto é, distribuição normal com igual média e igual desvio padrão.

(7) **Quantidade de 1's em fluxo de bytes**
Dado um arquivo de inteiros de 32 *bits*, este é examinado como uma sequência de *bytes*. Cada *byte* deve ocorrer com probabilidades iguais respectivamente a 1, 8, 28, 56, 70, 56, 28, 8, 1 de um total de 256 possibilidades. São contadas as ocorrências de cada palavra numa sequência de 256.000 palavras contendo 5 letras sobrepostas.

(8) **Quantidade de 1's para um byte específico**
Como no teste anterior, considera-se um arquivo de entrada como uma sequência de números inteiros de 32 *bits*. Para cada inteiro, um *byte* específico é escolhido, e.g., o mais à esquerda. Cada *byte* pode conter de 0 a 8 algarismos 1, com probabilidades iguais respectivamente a 1, 8, 28, 56, 70, 56, 28, 8, 1 sobre um total de 256 possibilidades. Esses *bytes* específicos produzem palavras de 5 letras, sendo que as palavras, a exemplo do teste anterior, também se sobrepõem, de modo que cada nova palavra é formada pelas últimas 4 letras da palavra anterior mais a próxima letra lida. As ocorrências são testadas numa sequência de 256.000 palavras.

(9) **Teste do estacionamento**
Simula-se um estacionamento onde seja possível estacionar aleatoriamente um carro, representado por um círculo de raio 1. O objetivo do teste é estacionar sucessivamente os carros no estacionamento um a um. Cada tentativa de se estacionar um carro conduz a uma colisão (já existe um carro na posição escolhida) ou a um sucesso. Cada sucesso acarreta um incremento à lista dos carros já estacionados. Pode-se mostrar que a distribuição da quantidade de sucessos é uma dada distribuição normal.

(10) **Distância mínima em quadrado**
Nesse teste escolhem-se $n = 8000$ pontos aleatórios em um quadrado de lado 10.000. É determinada a distância mínima entre os $(n^2 - n)/2$ pares de pontos. Pode-se mostrar que se a distribuição dos pontos for realmente aleatória, então $d^2$, o quadrado da distância mínima, obedece a uma certa distribuição exponencial.

(11) **Esferas mínimas**
São escolhidos 4000 pontos aleatórios num cubo de aresta 1000. Cada ponto é considerado como sendo o centro de uma esfera cujo raio é o

segmento de reta desse centro até o ponto vizinho mais próximo. Pode-se mostrar que o volume da menor destas esferas obedece a uma dada distribuição exponencial.

(12) **Compressão**
São utilizados valores de ponto flutuante no intervalo [0, 1), onde cada um desses é obtido a partir de um número inteiro do arquivo de testes. O método começa com o inteiro $k = 2^{31} - 1$ e é determinado o número $J$ de iterações que são necessárias para reduzir $k$ a 1 usando a "compressão" $k = [kU]$, onde $U$ é um número do arquivo testado. Os valores $J$'s são agrupados em 43 classes e as frequências de ocorrência calculadas.

(13) **Somas sobrepostas**
A exemplo do teste anterior, são escolhidos inteiros com representação em ponto flutuante para que se obtenha uma sequência $U_1, U_2, ...$ de variáveis uniformes no intervalo $[0, 1]$. Calcula-se então as somas sobrepostas, $S_1 = U_1 + \cdots + U_{100}$, $S_2 = U_2 + \cdots + U_{101}$ e assim por diante. É possível mostrar que os valores $S_i$'s podem ser transformados em valores de uma distribuição uniforme.

(14) **Corridas**
Determina-se quantas sequências ( denomina-se corridas) de uma dada sucessão ininterrupta de dígitos está em ordem crescente (ou decrescente). Uma corrida de comprimento $k$ consiste em exatamente $k$ dígitos em ordem crescente (ou decrescente). Por exemplo, a sequência |129|8530|789| têm duas corridas crescentes 129 e 789, e uma corrida decrescente 8530.

(15) **Craps**
Vem de um jogo em que cada lance corresponde ao arremesso de 2 dados. São jogadas 200.000 partidas e contabiliza-se tanto a quantidade de vitórias quanto o número de lançamentos de dados necessários para terminar cada jogo. Pode-se mostrar que a quantidade de vitórias tem distribuição normal caso os dados não sejam viciados.

(16) **Máximo divisor comum**
Considere dois inteiros sucessivos $u$ e $v$ produzidos pelo gerador de números aleatórios. Seja $k$ o número de passos necessários para encontrar o máximo divisor comum $x$ de $u$ e $v$ através do algoritmo de *Euclides*. Então $k$ pode ser aproximado por uma distribuição binomial com $p = 0.376$ e $n = 50$, enquanto $x$ possui uma distribuição muito próxima a $Pr[X = i] = c/i^2$ com $c = 6/\pi^2$. Nesse teste são gerados 10 milhões de pares $(u, v)$ e as frequências resultantes são comparadas com as distribuições de $k$ e de $x$ mencionadas acima.

(17) **Teste do gorila**
O Teste do gorila procura testar bits específicos dos inteiros de 32 bits gerados. Especifica-se um determinado bit a ser estudado entre 0 e 31. Esse teste necessita de 67.108.889 $(2^{26} + 25)$ números aleatórios e, a partir

desses números, gera-se uma string de $2^{26} + 25$ bits utilizando-se o bit escolhido. Contam-se, então, o número de segmentos de $2^6$ bits que não aparecem nessa string. Esse valor aproxima-se de uma distribuição normal com média 24.687.971 e desvio padrão 4.170. O teste é aplicado em cada um dos bits (0 até 31).

Tabela 11.2: DIEHARD com seus respectivos testes estatísticos

| Teste DIEHARD | Teste estatístico |
|---|---|
| 1. Espaçamento entre aniversários | $\chi^2$ com 1000 gl |
| 2. Permutação: 5 elem. e sobreposição | $\chi^2$ com 99 gl |
| 3. Posto de matrizes $31 \times 31$ e $32 \times 32$ | $\chi^2$ com 3 gl |
| 4. Posto de matrizes $6 \times 8$ | $\chi^2$ com 2 gl |
| 5. Fluxo de bits | Normal ($\mu = 141909, \sigma = 428$) |
| 6. Sobreposição de pares e DNA | Normal ($\mu = 141909, \sigma = 290$) |
| 7. Quant. de 1's em fluxo de bytes | $\chi^2$ com 2500 gl |
| 8. Quant. de 1's para um byte espec. | $\chi^2$ com 2500 gl |
| 9. Estacionamento | Normal($\mu = 3523, \sigma = 21,9$) |
| 10. Dist. mínima em quadrado | Kolmogorov-Smirnov |
| 11. Esferas mínimas | Kolmogorov-Smirnov |
| 12. Compressão | $\chi^2$ com 42 gl |
| 13. Somas sobrepostas | Kolmogorov-Smirnov |
| 14. Rodadas | Kolmogorov-Smirnov |
| 15. Craps | $\chi^2$ com 21 gl |
| 16. Máximo divisor comun | Binomial(n=50,p=0,376) |
| 17. Teste do gorila | Normal($\mu = 24687971, \sigma = 4170$) |

## 11.4 Material e métodos

O teste estatístico conhecido utilizado nesse trabalho foi o teste de Kolmogorov-Smirnov (KS[1]: $D_{máx} = máx|\widehat{F}(z_i) - F_0(z_i)|$), onde $\widehat{F}(z_i)$ é a função de distribuição normal acumulada na $i$-ésima classe, e $F_0$ a frequência relativa observada acumulada. O teste KS foi implementado em C para os testes da distância mínima em quadrados e das esferas mínimas para avaliar a aleatoriedade dos números gerados pelo gerador de G. Marsaglia (ver código fonte em C em anexo).

## 11.5 Resultados

Os resultados para os testes abordados são dados a seguir:

[1]KS: Kolmogorov- Smirnov

### 11.5.1 Distância mínima em quadrados

Nesse teste gerou-se 640.000 números pseudo aleatórios entre 0 e 9 com o gerador congruencial GM, que compõem os $n = 8000$ pontos para 10 réplicas (amostras). Cada ponto possui duas coordenadas, cada uma com 4 algarismo, totalizando $2\times4\times8000\times10 = 640.000$ pontos. Foram determinadas as distâncias mínimas $d_1, d_2, \cdots d_{10}$ (uma para cada réplica) entre todos os $(n^2-n)/2$ pares de pontos. O quadrado das distâncias mínimas tem distribuição exponencial com média 0.995 e os p-valores seguiram uma distribuição uniforme $[0, 1)$ para níveis de significâncias $\alpha = 0.1$, 0.05, 0.01 utilizando o teste KS como sugere o sistema DIEHARD. Ver tabela DM no anexo 2. Neste trabalho um outro conjunto de 640.000 números pseudo-aleatórios constituídos pelos dígitos do número $\pi$ foi avaliado nas mesmas condições acima, superando exatamente todas as fases do GM. O destaque do gerador GM em relação aos dígitos gerados pelo número $\pi$, é, na comparação amostra a amostra onde as distâncias mínimas são maiores para os geradores dos dígitos do número $\pi$. Ver tabela DM-Pi no anexo 4.

### 11.5.2 Esferas mínimas

Um outro teste abordado foi o das esferas mínimas. Gerou-se 640000 números pseudo aleatórios (de 0 a 9) com o gerador congruencial GM, que compõem os 4000 pontos para cada uma das 20 réplicas (amostras). Foram determinados os pontos como sendo o centro de uma esfera cujo raio é o segmento de reta desse centro ao ponto vizinho mais próximo. O volume da menor destas esferas tem distribuição exponencial com média 30 e os p-valores seguiram uma distribuição uniforme $[0, 1)$ para níveis de significâncias $\alpha = 0.1$, 0.05, 0.01 utilizando o teste KS como sugere o sistema DIEHARD. Ver tabela DME no anexo 3. Da mesma forma que no teste anterior os dígitos gerados pelo número $\pi$, superam todas as fases que os geradores de GM, no entanto, observando a tabela DME-Pi no anexo 5, as amostras apresentam maiores distâncias confirmando o resultado do teste da distância mínima.

## 11.6 Conclusões

A partir das simulações feitas concluímos que:

1) Para o testes da distância mínima, os geradores de números pseudo aleatórios G. Marsaglia, e os dígitos gerados pelo valor $\pi$, mostraram aleatoriedade, passando pelo teste KS, para níveis de significância de $\alpha = 0.1$, 0.05 e 0.01. O destaque dos geradores de números pseudo aleatórios GM, mostraram menores distâncias nas amostras.

2) Similarmente, para o teste das esferas mínimas, os geradores de números pseudo aleatórios G. Marsaglia, e os dígitos gerados pelo valor $\pi$, mostraram aleatoriedade, passando pelo teste KS, para níveis de significância de $\alpha =$

0.1, 0.05, 0.01. O destaque novamente é para os geradores de números pseudo-aleatórios GM, que apresentaram menores distâncias nas amostras.

3) Ambos geradores mostraram boa performance na avaliação dos testes de Distância mínima e Mínima esfera, e portanto, passaram nos dois testes DIEHARD, considerados rigorosos.

4) Recomenda-se a partir deste trabalho que o uso dos geradores de números pseudo-aleatórios de George Marsaglia (GM) sejam utilizados, porem o uso dos dígitos do número $\pi$ como números pseudo-aleatórios não é descartada.

5) Finalmente surge uma motivação maior para avaliar o desempenho dos dígitos do número $\pi$, nos restantes dos testes DIEHARD.

## 11.7 Algoritmo em C

### 11.7.1 Anexo 1. Código C que implementa o teste da distância mínima.

```
/* Testes DieHard - Distancia Minima */ #include <stdio.h>
#include <stdlib.h> #include <math.h> #define REPLICAS 10 #define
N 8000 struct ponto {   int x;     int y; }; struct rej{
       int rej01;
       int rej05;
       int rej10;
};
// Geração de pontos a partir do arquivo de dados
struct ponto GeraPonto(FILE *arq);
// Calculo da distancia entre dois pontos
double distancia(struct ponto, struct ponto);
// Calculo da distancia minima de um conjunto de pontos
double distanciamin(int, struct ponto []);
// Distribuicao empirica
double *DistEmp(int, double []);
// Subalgoritmo utilizado pelo QuickSort
int particiona(int left, int right, double x[]);
// Algoritmo de ordenacao QuickSort
void quicksort (int left, int right, double x[]);
// Funcao de densidade da distribuicao normal
double phi(double x);
// Integral numerica no intervalo (a,b), dividido em n partes
double PNormal(double a, double b, int n);
// Teste KS (Kolmogorov - Smirnov)
struct rej ks(int n, double []);
// Visualizacao dos resultados
void resultados(int n, double [], double [],
struct rej rejeicoes);

int main(void)
{
    FILE *arq;
    int i,k;
    struct ponto pontos[N];
    struct rej rejeita;
    double d, soma, dmin=0;
    double dminimas[REPLICAS];  // Valores das distancias minimas
    double p[REPLICAS];  // p-valores para as distancias minimas

    arq = fopen("Amostra.num","r");
```

```
    // Replicando ...
    for(k=0;k<REPLICAS;k++)
    {
       // Determinacao dos pontos
       for(i=0;i<N;i++)
           pontos[i]=GeraPonto(arq);
       // Calculo da distancia minima e do p-valor associado
       dmin = distanciamin(N,pontos);
       dminimas[k]= dmin;
       p[k]=1-exp(-dmin*dmin/0.995);
    }
    rejeita = ks(REPLICAS,p);
    resultados(REPLICAS,dminimas,p,rejeita);
    fclose(arq);
    return 0;
}
```

```
/* Obtencao das coordenadas dos pontos a partir de um arquivo
de numeros */
struct ponto GeraPonto(FILE *arq)
{   struct ponto p;
    int x=0, y=0, digito, i;

    // Coordenada x
    for(i=0;i<4;i++)
    {   if(fscanf(arq,"%d",&digito)!=EOF)
            x=10*x+digito;
    }
    // Coordenada y
    for(i=0;i<4;i++)
    {   if(fscanf(arq,"%d",&digito)!=EOF)
         y=10*y+digito;
    }
    p.x=x;
    p.y=y;
    return p;
}
/* Distancia entre dois pontos */ double distancia(struct ponto
p, struct ponto q) {
   return sqrt((p.x-q.x)*(p.x-q.x)+(p.y-q.y)*(p.y-q.y));
}
/* Distancia minima entre os pontos de um conjunto (armazenados
em um vetor) */ double distanciamin(int n, struct ponto pontos[])
{   double d,dmin;
    int i,j;
    dmin = distancia(pontos[0],pontos[1]);
    // Calculo da distancia minima
    for(i=0;i<n;i++)
        for(j=i+1;j<n;j++)
        {
            d = distancia(pontos[i],pontos[j]);
            if(d<dmin)
               dmin = d;
        }
    return dmin;
}
```

```
/* Subalgoritmo utilizado pelo quicksort */
int particiona(int left, int right, double x[])
{  int i = left+1, j = right;
   double temp, c = x[left];
   while (1) {
      while (i <= right && x[i] <= c) i++;
      while (c < x[j]) j--;
      if (i >= j) break;
      temp = x[i]; x[i] = x[j]; x[j] = temp;
      i++; j--;
   }
   temp = x[left]; x[left] = x[j]; x[j] = temp;
   return j;
}

/* Algoritmo de ordenacao QuickSort */
void quicksort (int left, int right, double x[])
{  int j;
   if (left < right) {
      j = particiona(left, right, x);
      quicksort(left, j-1, x);
      quicksort(j+1, right, x);
   }
}

/* Funcao de distribuicao normal */
double phi(double x)
{
   return exp(-x*x/2.0)/sqrt(2*3.14159265);
}

/* Integral numerica no intervalo (a,b), dividido em n partes */
double PNormal(double a, double b, int n)
{   int i;
    double h,area=0.0;

    h=(b-a)/n;
    for(i=0;i<n;i++)
        area+=(phi(a+i*h)+phi(a+(i+1)*h))*h/2.0;
    return area;
}
```

```
/* Distribuicao empirica de amostras */
double *DistEmp(int n, double x[])
{
    int i,j;
    int *freq;
    double *distr;

    freq = calloc(n,sizeof(int));
    distr = calloc(n,sizeof(double));

    for(i=0;i<n;i++)
        freq[i]=0;

    for(j=0;j<n;j++)
    {
        for(i=0;i<n;i++)
            if(x[i]<=x[j])
               freq[j]+=1;
    }

    for(i=0;i<n;i++)
    {
        distr[i]=(double)freq[i]/n;
    }

    free(freq);
    return distr;
}

/* Teste KS (Kolmogorov - Smirnov) */
struct rej ks(int n, double x[])
{
    int k, T, T1;
    double Z[REPLICAS];  // Normalizacao dos p-valores
    double *DistZ;//Distribuicao empirica das distancias minimas
    struct rej Rejeicao={0,0,0};
    double fN;
    double static vcN10[3]={0.4566,0.3687,0.3226};
    // Valores criticos
```

```
    // Normalizacao dos p-valores (Dados: media = 0.5 ,  var = 1/12)
     for(k=0;k<n;k++)
         Z[k]=(x[k]-0.5)/sqrt(1.0/12.0);

    // Ordenando Z
    quicksort(0,n-1,Z);
    // Distirbuicao empirica de Z
    DistZ = DistEmp(n,Z);

    //* Estatistica de Teste
    T1=DistZ[0]-PNormal(-6,Z[0],100);
    T=PNormal(-6,Z[0],100)-DistZ[0];
    if(T>T1) T1=T;
    T=DistZ[0]-PNormal(-6,Z[1],100);
    if(T>T1) T1=T;
    T=PNormal(-6,Z[1],100)-DistZ[0];
    if(T>T1) T1=T;

    for(k=1;k<n;k++)
    {   T=DistZ[k]-PNormal(-6,Z[k],100);
        if(T>T1) T1=T;

        T=PNormal(-6,Z[k],100)-DistZ[k];
        if(T>T1) T1=T;

        T=PNormal(-6,Z[k],100)-DistZ[k-1];
        if(T>T1) T1=T;

        T=DistZ[k-1]-PNormal(-6,Z[k],100);
        if(T>T1) T1=T;
    }
    /* ********** Verificando rejeicao ********** */
     if(T1>vcN10[0])
         Rejeicao.rej01=1;
     if(T1>vcN10[1])
        Rejeicao.rej05=1;
     if(T1>vcN10[2])
        Rejeicao.rej10=1;
     free(DistZ);
     return Rejeicao;
}
```

```
// Visualizacao dos resultados
void resultados(int n, double min[], double p[],
struct rej rejeicoes)
{
     FILE *arq2;
     int i;
     arq2 = fopen("DistMinima.txt","w");

     fprintf(arq2,"  ");
     for(i=0;i<70;i++) fprintf(arq2,"-");
     fprintf(arq2,"\n");
     for(i=0;i<15;i++) fprintf(arq2," ");
     fprintf(arq2,"Testes DieHard:  Distancia Minima\n" );
     fprintf(arq2,"  ");
     for(i=0;i<70;i++) fprintf(arq2,"-");
     fprintf(arq2,"\n");
     fprintf(arq2,"%8sDados: Sequencia de 640000 algarismos\n","");
     fprintf(arq2,"%8sGerador: Congruencial de G. Marsaglia\n" );
     fprintf(arq2,"  ");
     for(i=0;i<70;i++) fprintf(arq2,"-");
     fprintf(arq2,"\n");

 fprintf(arq2,"    Amostra    Dist.Min.    Dist.Min^2     p-valor\n\n");
 for(i=0;i<10;i++)
 fprintf(arq2,"%7s%02d%4s%10.5f%4s%10.5f%3s%10.5f\n"
         ,"",i+1,"",min[i],"",min[i]*min[i],"",p[i]);

     fprintf(arq2,"  ");
     for(i=0;i<70;i++) fprintf(arq2,"-");
     fprintf(arq2,"\n      Resultados do teste de Kolmogorov-Smirnov\n\n");
     fprintf(arq2,"%6s%4s%11s%4s%7s%4s%7s%4s\n"
           ,"","Alfa","","0.01","","0.05","","0.10");
     fprintf(arq2,"%6s%6s%3s%5.2f%6s%5.2f%6s%5.2f\n","","Rejeicao(%)","",
        (double)rejeicoes.rej01,"",(double)rejeicoes.rej05/n,"",
        (double)rejeicoes.rej10/n);

     fprintf(arq2,"  ");
     for(i=0;i<70;i++) fprintf(arq2,"-");
     fclose(arq2);
}
```

### 11.7.2 Anexo 2. Saída do teste da Distância mínima no quadrado usando o gerador de Marsaglia

**Tabela DM:** Saída do teste da Distância mínima no quadrado, usando o teste KS para os p-valores.

```
-------------------------------------------------------
          Testes DieHard:  Distância Mínima

-------------------------------------------------------
        Dados: Seqüência de 640000 algarismos
        Gerador: Congruencial de G. Marsaglia

-------------------------------------------------------
    Amostra    Dist.Min.    Dist.Min^2      p-valor

       01       1.00000       1.00000       0.63396
       02       0.00000       0.00000       0.00000
       03       0.00000       0.00000       0.00000
       04       1.00000       1.00000       0.63396
       05       1.00000       1.00000       0.63396
       06       1.00000       1.00000       0.63396
       07       1.00000       1.00000       0.63396
       08       1.00000       1.00000       0.63396
       09       1.00000       1.00000       0.63396
       10       1.00000       1.00000       0.63396

-------------------------------------------------------
      Resultados do teste de Kolmogorov-Smirnov

      Alfa            0.01       0.05       0.10
      Rejeicao(\%)    0.00       0.00       0.00

-------------------------------------------------------
```

### 11.7.3 Anexo 3. Saída do teste da Esfera mínima usando o gerador de Marsaglia

**Tabela DME:** Saída do teste da Esfera mínima, usando o teste KS para os p-valores.

```
-----------------------------------------------------
          Testes DieHard:  Esferas Mínimas

-----------------------------------------------------
        Dados: Sequencia de 640000 algarismos
        Gerador: Congruencial de G. Marsaglia

-----------------------------------------------------
    Amostra    Raio.Min.    Raio.Min^3     p-valor

       01       1.00000       1.00000      0.03278
       02       1.00000       1.00000      0.03278
       03       0.00000       0.00000      0.00000
       04       0.00000       0.00000      0.00000
       05       1.00000       1.00000      0.03278
       06       1.00000       1.00000      0.03278
       07       0.00000       0.00000      0.00000
       08       0.00000       0.00000      0.00000
       09       0.00000       0.00000      0.00000
       10       1.00000       1.00000      0.03278
       11       1.00000       1.00000      0.03278
       12       0.00000       0.00000      0.00000
       13       1.00000       1.00000      0.03278
       14       1.00000       1.00000      0.03278
       15       1.00000       1.00000      0.03278
       16       0.00000       0.00000      0.00000
       17       1.00000       1.00000      0.03278
-----------------------------------------------------
       Resultados do teste de Kolmogorov-Smirnov
       Alfa           0.01       0.05       0.10
       Rejeicao(\%)   0.00       0.00       0.00
-----------------------------------------------------
```

### 11.7.4 Anexo 4. Saída do teste da distância mínima usando os 640.000 primeiros dígitos do número $\pi$

**Tabela DM-Pi: Saída do teste da distância mínima, usando o teste KS para os p-valores.**

```
------------------------------------------------------
          Testes DieHard:  Distancia Mínima

------------------------------------------------------
      Dados: Seqüência de 640.000 algarismos de Pi

------------------------------------------------------
    Amostra    Dist.Min.    Dist.Min^2      p-valor

       01       1.00000       1.00000      0.63396
       02       1.41421       2.00000      0.86602
       03       0.00000       0.00000      0.00000
       04       1.00000       1.00000      0.63396
       05       1.00000       1.00000      0.63396
       06       1.41421       2.00000      0.86602
       07       0.00000       0.00000      0.00000
       08       0.00000       0.00000      0.00000
       09       1.00000       1.00000      0.63396
       10       1.41421       2.00000      0.86602

------------------------------------------------------
      Resultados do teste de Kolmogorov-Smirnov

      Alfa            0.01       0.05       0.10
      Rejeicao(\%)    0.00       0.00       0.00

------------------------------------------------------
```

### 11.7.5 Anexo 5. Saída do teste da Esfera mínima usando os 640.000 primeiros dígitos do número $\pi$

**Tabela DME-Pi: Saída do teste da Esfera mínima, usando o teste KS para os p-valores obtidos.**

```
------------------------------------------------------
          Testes DieHard:  Esferas Mínimas

------------------------------------------------------
```

Dados: Seqüência de 640.000 algarismos de Pi

| Amostra | Raio.Min. | Raio.Min^3 | p-valor |
|---|---|---|---|
| 01 | 0.00000 | 0.00000 | 0.00000 |
| 02 | 0.00000 | 0.00000 | 0.00000 |
| 03 | 1.00000 | 1.00000 | 0.03278 |
| 04 | 1.00000 | 1.00000 | 0.03278 |
| 05 | 0.00000 | 0.00000 | 0.00000 |
| 06 | 1.00000 | 1.00000 | 0.03278 |
| 07 | 1.00000 | 1.00000 | 0.03278 |
| 08 | 1.00000 | 1.00000 | 0.03278 |
| 09 | 0.00000 | 0.00000 | 0.00000 |
| 10 | 1.73205 | 5.19615 | 0.15903 |
| 11 | 0.00000 | 0.00000 | 0.00000 |
| 12 | 1.41421 | 2.82843 | 0.08997 |
| 13 | 1.00000 | 1.00000 | 0.03278 |
| 14 | 0.00000 | 0.00000 | 0.00000 |
| 15 | 0.00000 | 0.00000 | 0.00000 |
| 16 | 0.00000 | 0.00000 | 0.00000 |
| 17 | 0.00000 | 0.00000 | 0.00000 |

| Resultados do teste de Kolmogorov-Smirnov | | | |
|---|---|---|---|
| Alfa | 0.01 | 0.05 | 0.10 |
| Rejeicao(\%) | 0.00 | 0.00 | 0.00 |

# Referências Bibliográficas


[1] Ballie, RT., Chung, CF. e Tieslau, MA.(1996):*Analising inflation by the fractionally integrated ARFIMA-GARCH model*. Journal of applied econometrics, 11, 23 − 40.

[2] Beran, J. (1994): *Statistics for long memory process*. New York: Chapman and Hall.

[3] Borde, Arvind (1992). TEX *by example*. A beginner´s guide. Ed. Academic Press Professional. United States of America.

[4] Carneiro, Orlando (1997). *Ecomometria Básica. Teoria e Aplicações*. Editora Atlas. Segunda Edição. São Paulo.

[5] Conover, W. J. (1999) *Practical nonparametric statistics*. 3.ed. New York.

[6] Cribari Neto, F.; Cabral de Araújo Gois Matheus (2002). *Uma Análise de Monte Carlo do Desempenho de Estimadores de Matrizes de Covarância sob Heterocedasticidade de Forma Desconhecida*. RBE, Rio de Janeiro 56(2), p.309-334.

[7] Cribari Neto, F (2002). *C for Econometricians*. Computational Economics 14. p.135-149.

[8] Reisen, V., Cribari-Neto, F., Jensen, Mark. *Long Memory Inflationary Dynamics: The Case of Brazil*. Studies in Nonlinear Dynamics & Econometrics. *Vol. 7, Issue 3.*(2003)

[9] Ehlers,Ricardo.(2007): Análise de séries Temporais. Universidade Federal do Paraná.

[10] Ferreira , M (2006). *Análise da sensibilidade dos testes de normalidade de Jarque-Bera e Lilliefors em modelos de regressão Linear*. Rev. Mat. Estat, v.24, n.4, p.89-98. São Paulo.

[11] Frery, Alejandro; Cribari-Neto, Francisco.(2011).*Elementos de Estatística Computacional Usando Plataformas de Software Livre/Gratuito*. Publicações Matemáticas. Editora IMPA.

[12] Geweke, J. and Porter-Hudak, S. (1983). The estimation and aplication of long memory time series models. *Journal of Time Series Analysis*, **4**, 221 – 238.

[13] Goldreich, O. and Micali, S. (1986). *How to construct random functions.* Journal of the Association for Computing Machinery, 33(4), p.792 – 807.

[14] Gujarati, Damodar N (2000). *Ecomometria Básica.* Makron Books. Terceira Edição. São Paulo.

[15] Hill, Carter R.; Griffiths, William E.; Judge, George G (2003). *Ecomometria.* Editora Saraiva. Segunda Edição. São Paulo.

[16] Hosking, J. R. M.(1981). Fractional differencing.Biometrica,**68**, 165 – 176.

[17] Knuth, D.E. (1981). *The Art of Computer Programming, Third Edition. Volume 2: Seminumerical. Algorithms*, Addison-Wesley Publishing Company, Massachusetts.

[18] Korgi, Rodrigo de Castro (2003). *El universo LaTeX.* Editora Facultad de Ciencias. Segunda Edição. Bogota.

[19] Krawczyk, H. *How to Predict Congruential Generators.* In:Journal of Algorithms, V. 13, N. 4, 1992.

[20] L'Ecuyer, P (1994). *Uniform random number generation.* Annals of Operations Research, pp. 77 – 120.

[21] L'Ecuyer, P (2001). *Software for uniform random number generation: distinguishing the good and the bad.* In Proceedings of the 2001 Winter Simulation Conference.

[22] Marsaglia, G. *The Marsaglia Random Number including the DIEHARD Battery of Tests of Randomness*
(Disponível em: http://stat.fsu.edu/pub/diehard/).

[23] Marsaglia, G. *A Current View of Random Number Generators, Keynote Address, Computer Science and Statistics.* In: 16th Symposium on the Interface, Atlanta. Published by Elsevier Press, 1984.

[24] Marsaglia, G.(1993) *Monkey Tests for Random Number Generators.* In: Computers and Mathematics with Applications, 9, p.1-10, 1993.

[25] McCullough (1998). *Benchmarking Statistical software. American Statistician*, Fortcoming.

[26] Mingoti(2005).*Analise De Dados Atraves De Metodos De Estatistica Multivariada - Uma Abordagem Aplicada.* Editora UFMG. Belo Horizonte.

[27] Morettin, A. P., Clélia, M. C.(2006) *Análise de séries temporais*. Editora Egard Blucher, $2^{da}$ edição.

[28] Papoulis, A. e Pillai, S. (2002). *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 4th edition.

[29] R Development Core Team (2010). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

[30] Reisen, V., Lemonte, A.(2006). The fracdiff package Version $1.3 - 1$.

[31] Reisen, V. (2007): Minicurso: MODELO ARFIMA.

[32] Soto, J. *Statistical Testing of Random Number Generators*. In: Proceedings of the 22nd National Information Systems Security Conference, Crystal City, Virginia, 1999.
(Disponível em: http://csrc.nist.gov/rng/nissc-paper.pdf).

[33] VenablesWN, Ripley BD (2000). *S Programming*. Springer-Verlag, New York.

[34] Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. 4th edition. Springer-Verlag, New York.

[35] Wong, P. C (1990). *Random number generation without multiplication*. In 9th Annual Intl Phoenix Conf on Computers and Communications, p.217-221.

[36] Zafon, G; Manacero Jr, Aleardo (2006). *Construção de Geradores independentes de números aleatórios para diferentes distribuições probabilísticas*. Anais do XXVI Congresso de SBC, p.16-21. Campo Grande. MS

# Apêndice A

# Dados utilizados

## A.1 Dados para discriminar os funcionários

Uma empresa esta buscando uma regra para discriminar a qualidade dos funcionários (definido como variável grupo). Em seus arquivo de dados dispor de 109 registros com as seguintes informações dos funcionários x1: idade dos funcionários, x2: sexo (1 homem; 0 mulher), x3: anos de experiência na função, X4: estado civil, x5: número de filhos, x6: nota de 0 a 10 de um teste de avaliação de conhecimento na sua área, x7: teste psicotécnico de 0 a 50, x8: satisfação com a vida pessoal. Y, definido a priori como grupo1 (desempenho médio) e grupo 2 (melhor desempenho), 0 se faz parte do grupo 1 e 1 se faz parte do grupo 2. ver (Mingoti, 2005)

```
> funcao
    grupo idade sexo ecivil nfilhos aexper tpsico tconhec satisf
1       0    24    1      1       0      2    7.7    36.0      1
2       0    29    1      1       0      4    7.9    36.0      1
3       0    38    1      0       1      6    8.6    40.8      0
4       1    27    1      1       0      3    8.5    44.0      1
5       1    33    1      1       0      4    8.0    36.0      1
6       1    29    1      1       1      4    8.5    42.0      1
7       0    38    1      1       0      8    9.2    44.8      1
8       0    35    0      0       1      5    7.9    46.0      1
9       1    32    1      0       0      4    9.9    44.0      1
10      0    36    0      1       0      6    8.3    40.0      1
11      0    33    0      1       0      5    6.4    32.0      1
12      0    36    1      0       3      4    7.2    44.0      1
13      1    30    0      0       2      3    9.5    44.0      1
14      1    26    1      1       0      2    8.8    40.0      1
15      0    37    0      1       0      5    8.2    50.0      1
16      1    25    0      1       2      2    9.4    44.0      1
17      1    38    1      1       0      6    9.0    40.0      0
```

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 18 | 1 | 31 | 1 | 0 | 4 | 5 | 9.2 | 40.0 | 1 |
| 19 | 1 | 37 | 1 | 0 | 5 | 7 | 9.4 | 44.0 | 1 |
| 20 | 0 | 38 | 1 | 1 | 0 | 5 | 6.9 | 36.0 | 1 |
| 21 | 1 | 30 | 1 | 0 | 2 | 4 | 8.9 | 40.0 | 1 |
| 22 | 0 | 29 | 1 | 0 | 1 | 3 | 7.4 | 36.0 | 1 |
| 23 | 1 | 34 | 0 | 1 | 1 | 4 | 9.1 | 40.0 | 1 |
| 24 | 1 | 25 | 0 | 1 | 0 | 2 | 8.9 | 40.0 | 0 |
| 25 | 1 | 34 | 1 | 0 | 0 | 4 | 8.3 | 40.0 | 1 |
| 26 | 0 | 30 | 1 | 1 | 0 | 3 | 7.8 | 36.0 | 0 |
| 27 | 0 | 37 | 1 | 1 | 0 | 5 | 8.2 | 43.2 | 1 |
| 28 | 0 | 26 | 1 | 1 | 0 | 3 | 7.2 | 36.0 | 1 |
| 29 | 1 | 33 | 1 | 0 | 1 | 6 | 9.8 | 44.0 | 1 |
| 30 | 0 | 25 | 1 | 1 | 0 | 2 | 7.9 | 36.0 | 1 |
| 31 | 0 | 26 | 1 | 1 | 0 | 2 | 6.5 | 32.0 | 0 |
| 32 | 0 | 32 | 1 | 1 | 0 | 4 | 6.8 | 32.0 | 1 |
| 33 | 0 | 28 | 1 | 0 | 1 | 4 | 7.3 | 36.0 | 1 |
| 34 | 1 | 30 | 1 | 1 | 0 | 5 | 8.9 | 40.0 | 1 |
| 35 | 1 | 27 | 1 | 1 | 0 | 2 | 8.6 | 40.0 | 1 |
| 36 | 1 | 34 | 0 | 1 | 0 | 5 | 8.9 | 40.0 | 0 |
| 37 | 1 | 35 | 1 | 1 | 0 | 5 | 9.4 | 44.0 | 1 |
| 38 | 0 | 24 | 0 | 1 | 0 | 2 | 6.6 | 32.0 | 1 |
| 39 | 0 | 30 | 1 | 1 | 0 | 3 | 6.4 | 32.0 | 1 |
| 40 | 1 | 29 | 1 | 1 | 0 | 4 | 8.8 | 40.0 | 0 |
| 41 | 1 | 31 | 0 | 1 | 0 | 4 | 8.6 | 40.0 | 1 |
| 42 | 1 | 28 | 1 | 1 | 0 | 4 | 9.4 | 44.0 | 1 |
| 43 | 1 | 37 | 1 | 1 | 0 | 8 | 8.2 | 40.0 | 1 |
| 44 | 0 | 36 | 1 | 1 | 0 | 6 | 6.8 | 32.0 | 1 |
| 45 | 0 | 33 | 1 | 1 | 0 | 6 | 7.4 | 36.0 | 1 |
| 46 | 1 | 38 | 1 | 0 | 3 | 10 | 9.0 | 40.0 | 1 |
| 47 | 0 | 25 | 1 | 1 | 0 | 2 | 6.3 | 32.0 | 1 |
| 48 | 0 | 27 | 1 | 1 | 0 | 2 | 6.9 | 36.0 | 0 |
| 49 | 1 | 37 | 1 | 1 | 0 | 6 | 8.5 | 40.0 | 1 |
| 50 | 0 | 24 | 1 | 1 | 0 | 2 | 7.2 | 36.0 | 1 |
| 51 | 1 | 35 | 0 | 1 | 0 | 5 | 9.9 | 44.0 | 1 |
| 52 | 0 | 33 | 1 | 0 | 1 | 4 | 7.6 | 36.0 | 1 |
| 53 | 1 | 29 | 1 | 1 | 0 | 3 | 8.5 | 40.0 | 1 |
| 54 | 1 | 27 | 1 | 1 | 0 | 2 | 8.7 | 40.0 | 1 |
| 55 | 1 | 27 | 0 | 1 | 0 | 4 | 8.6 | 40.0 | 1 |
| 56 | 1 | 27 | 0 | 0 | 0 | 4 | 9.5 | 44.0 | 1 |
| 57 | 1 | 32 | 1 | 1 | 0 | 4 | 8.8 | 40.0 | 1 |
| 58 | 1 | 34 | 1 | 0 | 0 | 5 | 9.2 | 40.0 | 1 |
| 59 | 1 | 32 | 0 | 1 | 0 | 5 | 8.1 | 36.0 | 0 |
| 60 | 0 | 31 | 1 | 0 | 2 | 5 | 7.7 | 36.0 | 1 |
| 61 | 0 | 25 | 1 | 1 | 0 | 2 | 6.7 | 32.0 | 1 |
| 62 | 1 | 38 | 1 | 0 | 2 | 7 | 8.5 | 40.0 | 1 |
| 63 | 0 | 29 | 1 | 1 | 0 | 5 | 7.9 | 36.0 | 1 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 64 | 0 | 25 | 1 | 0 | 1 | 2 | 7.6 | 36.0 | 1 |
| 65 | 1 | 28 | 1 | 1 | 0 | 3 | 9.3 | 40.0 | 1 |
| 66 | 0 | 37 | 1 | 1 | 0 | 6 | 6.5 | 32.0 | 1 |
| 67 | 1 | 30 | 1 | 1 | 0 | 4 | 9.6 | 44.0 | 1 |
| 68 | 0 | 32 | 0 | 1 | 0 | 4 | 6.9 | 36.0 | 0 |
| 69 | 1 | 28 | 1 | 1 | 0 | 3 | 9.0 | 40.0 | 0 |
| 70 | 1 | 25 | 0 | 1 | 0 | 2 | 9.0 | 40.0 | 0 |
| 71 | 1 | 26 | 0 | 1 | 0 | 2 | 8.0 | 36.0 | 0 |
| 72 | 0 | 33 | 0 | 1 | 0 | 5 | 7.9 | 36.0 | 1 |
| 73 | 1 | 35 | 1 | 0 | 2 | 5 | 8.7 | 40.0 | 1 |
| 74 | 1 | 33 | 1 | 1 | 0 | 4 | 9.8 | 44.0 | 1 |
| 75 | 1 | 33 | 1 | 0 | 0 | 5 | 9.7 | 44.0 | 1 |
| 76 | 1 | 38 | 0 | 0 | 4 | 9 | 9.3 | 40.0 | 0 |
| 77 | 1 | 31 | 0 | 1 | 0 | 4 | 8.6 | 40.0 | 1 |
| 78 | 1 | 28 | 1 | 1 | 0 | 3 | 8.4 | 40.0 | 1 |
| 79 | 0 | 30 | 0 | 0 | 1 | 3 | 6.5 | 32.0 | 1 |
| 80 | 0 | 35 | 1 | 1 | 0 | 4 | 7.0 | 36.0 | 0 |
| 81 | 0 | 29 | 0 | 1 | 0 | 3 | 7.2 | 36.0 | 1 |
| 82 | 1 | 33 | 1 | 1 | 0 | 5 | 9.1 | 40.0 | 1 |
| 83 | 0 | 35 | 1 | 1 | 0 | 4 | 7.5 | 36.0 | 1 |
| 84 | 1 | 32 | 0 | 1 | 0 | 5 | 9.3 | 40.0 | 0 |
| 85 | 1 | 37 | 1 | 1 | 0 | 4 | 8.4 | 40.0 | 1 |
| 86 | 0 | 31 | 1 | 0 | 1 | 4 | 7.4 | 36.0 | 1 |
| 87 | 1 | 34 | 1 | 0 | 0 | 6 | 9.7 | 44.0 | 0 |
| 88 | 0 | 32 | 1 | 0 | 0 | 5 | 7.8 | 36.0 | 0 |
| 89 | 0 | 30 | 1 | 1 | 0 | 3 | 7.9 | 36.0 | 0 |
| 90 | 1 | 28 | 1 | 1 | 0 | 3 | 8.2 | 40.0 | 1 |
| 91 | 0 | 34 | 0 | 0 | 2 | 6 | 6.8 | 32.0 | 1 |
| 92 | 1 | 35 | 1 | 0 | 2 | 6 | 8.1 | 36.0 | 1 |
| 93 | 1 | 31 | 0 | 0 | 1 | 4 | 8.7 | 40.0 | 1 |
| 94 | 1 | 28 | 0 | 1 | 0 | 4 | 9.5 | 44.0 | 1 |
| 95 | 1 | 45 | 0 | 0 | 4 | 16 | 9.8 | 44.0 | 0 |
| 96 | 1 | 40 | 0 | 0 | 2 | 12 | 9.2 | 40.0 | 0 |
| 97 | 0 | 30 | 1 | 1 | 0 | 4 | 7.7 | 36.0 | 1 |
| 98 | 0 | 31 | 1 | 1 | 0 | 3 | 7.5 | 36.0 | 1 |
| 99 | 1 | 32 | 1 | 0 | 1 | 4 | 8.0 | 36.0 | 1 |
| 100 | 1 | 24 | 0 | 1 | 0 | 2 | 8.5 | 40.0 | 0 |
| 101 | 1 | 34 | 1 | 1 | 0 | 6 | 8.9 | 40.0 | 1 |
| 102 | 0 | 26 | 0 | 1 | 0 | 3 | 7.8 | 36.0 | 0 |
| 103 | 1 | 35 | 1 | 0 | 1 | 8 | 9.3 | 40.0 | 1 |
| 104 | 1 | 32 | 0 | 1 | 0 | 5 | 9.0 | 40.0 | 1 |
| 105 | 0 | 24 | 1 | 1 | 0 | 2 | 7.6 | 36.0 | 1 |
| 106 | 0 | 27 | 1 | 1 | 0 | 2 | 7.3 | 36.0 | 1 |
| 107 | 0 | 24 | 1 | 1 | 0 | 2 | 7.3 | 36.0 | 0 |
| 108 | 0 | 27 | 1 | 0 | 0 | 3 | 7.4 | 36.0 | 1 |
| 109 | 1 | 42 | 1 | 0 | 2 | 14 | 9.9 | 44.0 | 1 |

## A.2 Consumo de água em uma residencia

Considere os dados de consumo de água de uma residência durante o período de 12/05 a 04/11. Com as seguintes variáveis: valor em reais, consumo em $m^3$, dias de consumo, e Construindo o imóvel (1:sim, 2:não). Dados próprios do autor.

```
> ca
   dado valor consumo diasconsumo Construindo
1     1 14.76      11          30           0
2     2 13.47      10          31           0
3     3 13.47      10          32           0
4     4 52.75      22          33           1
5     5 31.53      16          31           1
6     6 13.47      10          32           0
7     7 13.47      10          30           0
8     8 13.47      10          30           0
9     9 13.47      10          32           0
10   10 19.49      12          30           1
11   11 31.53      16          31           1
12   12 37.55      18          32           1
13   13 36.06      13          31           1
14   14 16.48      11          30           0
15   15 37.55      18          32           0
16   16 37.55      18          30           0
17   17 33.36      16          30           0
18   18 28.53      15          32           0
19   19 30.09      15          30           0
20   20 38.23      17          30           0
21   21 33.21      16          31           0
22   22 13.47       8          29           0
23   23 13.47       8          32           0
24   24 15.07      10          29           0
25   25 15.07       6          29           0
26   26 15.07       9          31           0
27   27 15.55       6          30           0
28   28 15.07       9          31           0
29   29 15.07       9          32           0
30   30 25.54      13          30           0
31   31 21.81      12          30           0
32   32 29.02      14          32           0
33   33 15.07      10          30           0
34   34 21.81      12          29           0
35   35 36.53      16          32           0
36   36 69.29      24          31           1
37   37 35.29      16          32           1
```

```
38   38 42.44      17          30          1
39   39 42.44      17          30          1
40   40 31.34      14          29          1
41   41 31.34      14          32          1
42   42 20.24      11          31          0
43   43 16.54      10          29          0
44   44 16.54       8          31          0
45   45 16.54      10          31          0
46   46 23.94      12          32          0
47   47 23.94      12          31          0
48   48 31.34      14          29          1
49   49 31.34      14          32          1
50   50 32.69      14          30          0
51   51 17.25      10          30          0
52   52 28.83      13          32          0
53   53 17.25      10          30          0
54   54 21.11      11          29          0
55   55 17.25      10          32          0
56   56 24.97      12          30          0
57   57 17.25       9          29          0
58   58 44.27      17          32          1
59   59 24.97      12          32          0
60   60 21.11      11          30          0
61   61 17.25      10          29          0
62   62 17.25      10          32          0
63   63 44.27      17          30          1
64   64 34.39      14          31          1
65   65 22.21      11          32          0
```

## A.3 Projeção dos tamanhos da população do Brasil

As projeções do tamanho da população Brasileira foi calculada usando a taxa de crescimento populacional entre dois censos consecutivos e foram fornecida em formato txt, como o nome popbrasil.txt, no R usamos os comandos:

```
pb=read.table("popbrasil.txt",header=T)
attach(pb)
pb
 ano   popbrasil
1870   9.533.659
1880  11.689.938
1890  14.333.915
1900  17.438.434
```

```
1910  23.113.566
1920  30.635.605
1930  35.542.924
1940  41.236.315
1950  51.944.397
1960  70.070.457
1970  93.139.037
1980 119.002.706
1990 144.047.741
2000 169.544.443
2010 190.732.694
plot(ano,popbrasil)
```