Notas de Aula da Disciplina - Introdução ao R

Alexandre Nicolella
André Luiz Martins Pignata

2018

Introdução a Disciplina

A Ciência Econômica atual tem se preocupado em medir o comportamento dos agentes econômicos. Compreender as relações entre as variáveis e as causas e efeitos, conforme descrito pela teoria econômica. Dessa forma a Economia se tornou uma ciência baseada em fatos e que utiliza os princípios da estatística para testar as hipóteses levantadas pela teoria.

Nesse sentido surgem questões importantes e que afetam milhares de pessoas. Será que maior salário de professores melhora o nível educacional dos alunos? Será que maiores taxas de juros diminuem a expectativa de inflação? Será que a transformação de uma região administrativa em região metropolitana melhora a qualidade de vida dos cidadãos? Essas e muitas outras questões são de extrema relevância para melhoria de vida das pessoas.

Dessa forma, estamos interessados em dois aspectos. Para o caso dos professores, queremos saber se i) melhor salário melhora desempenho dos alunos, e; ii) se melhora qual o valor dessa melhora, ou seja, quanto aumenta o desempenho por cada real investido?

Para responder apropriadamente essas questões os alunos devem evoluir tanto nas disciplinas teóricas, microeconomia, macroeconomia e história e nas técnicas de estatística e econometria. Entretanto, para efetivamente colocar em prática todo esse conhecimento é necessário utilizar um pacote estatístico. Nossa opção aqui será o R.

Objetivos

O nosso objetivo nessa disciplina analisar e explorar a seguinte questão: Há diferencial de salário entre homens e mulheres no Brasil, ou seja, os homens ganham efetivamente mais que as mulheres para a mesma atividade desempenhada.

Para alcançar esse objetivo utilizaremos os dados da Pesquisa Nacional por Amostra de Domicílio (PNAD) 2014. A turma será dividida em grupos de dois alunos e cada grupo irá analisar um estado.

A avaliação é composta pela entrega sistemática dos exercícios semanais/quinzenais propostos em sala. O trabalho final será composto da junção desses diversos trabalhos e assim teremos uma análise detalhada do diferencial por Estado Brasileiro.

Definições

Nestas notas são utilizadas algumas formatações para identificar tipos de dados, comandos, variáveis e outras estruturas. São elas:

- <TEXTO>: valor que deve ser substituído conforme sua necessidade. Exemplo: dtNasc <- <ANO>, por ser substituído por dtNasc <- 2016

O R-project e as Boas Práticas

O software R

O R é uma linguagem e ambiente de desenvolvimento de Estatística e gráficos. É uma ferramenta poderosa, fornecendo ao seu usuário maior integração e qualidade gráfica e de análise. Alguns motivos para utilizar o R:

- **É Gratuito**: é um projeto open-source. Pode ser utilizado em qualquer sistema operacional e tem aberto seus códigos e pactos para poder ser inspecionado.
- **R é uma Linguagem**:Requer que seja escrito um script ao invés de clicar. A primeira vista uma característica negativa, entretanto, permite maior exploração, organização, memória da atividade, maior integração entre processos etc.
- **Gráficos e Visualizações**: É sem sombra de dúvida o pacote estatístico com melhor e mais poderosa ferramenta de elaboração de gráficos e visualização.
- **Pacotes Estatísticos**:Já possui muitas rotinas de análises já programadas nos diversos pacotes desenvolvidos, sendo muito bem documentados. Já possui muitas rotinas para regressão, regressão com séries temporais, regressão em painel, finanças, modelos de causalidade etc.
- **Fronteira do Conhecimento**: Os principais desenvolvimentos teóricos em Econometria tem sua aplicação demonstrada e desenvolvida utilizando o R. Isso é valido para todas as subáreas do conhecimento em econometria, séries temporais, painel, finanças, etc.
- **Recursos de Ajuda**: Há uma comunidade muito grande disponível para solucionar dúvidas e uma vasta documentação disponível para consulta na rede.
- **Conexão com Outros Pacotes**: O R integra com outros pacotes que automatizam o nosso trabalho cotidiano. Pode se conectar com o Python, Java, SQL, Latex etc.

Utilizando Interface Gráfica - O Rstudio

Pode-se realizar seu script diretamente no console do R. Ele irá realizar um comando por vez. O R é uma interface leve e com poucos recursos gráficos. Uma alternativa ao uso do R diretamente é o Rstudio, o qual é um editor de código ou um ambiente de desenvolvimento integrado. Ele possui quatro janela, sendo a primeira a janela de script (superior esquerda) onde escrevemos os comandos em R. A segunda janela é o console (inferior esquerda) similar ao que temos no R e onde os resultados são apresentados. Pode-se digitar comando diretamente no console do RStudio.
A terceira é a janela de ambiente e história (superior direita) ela armazena seus dados, valores e funções e a aba história possui a memoria dos comandos realizados. Por fim a quarta janela (inferior direita) apresenta os pacotes, os gráficos, os arquivos gerados e ajuda. Essa janela facilita a instalação de pacotes, carregamento de bibliotecas, visualização de gráficos e o caminho dos arquivos.

Ajuda

Para abrir a ajuda geral o seguinte abaixo pode ser utilizado e abrirá uma janela no seu navegador.

```
help.start()
```

Suponha que queiramos saber de uma função específica, assim pode-se utilizar o seguinte comando:

```
help(summary)
```

ou

```
?summary
```

Inclusive pode pedir um exemplo de como utilizar a função que está buscando

```
example(summary)
```

Uma alternativa é o Swirl. Ele é um pacote do R que ajuda seu aprendizado por meio do próprio R. É um tutoria desenhado em R para aprender no próprio programa os princípios básicos de programação, estatística e regressões. Veja como instalar:

```
install.packages("swirl")
library("swirl")

swirl()
```

Para sair do swirl digite o comando *bye()*.

Boas Práticas

É fundamental que o usuário seja organizado. Forma é muito importante! Assim o usuário deve adotar padrões que auxiliem na organização do seu script ou programa.

Case Sensitivy: O R diferencia letras minúsculas e maiúscula. Ou seja, m é diferente de M. Por exemplo, considere as três formas de escrever a palavra idade.

idade ou Idade ou IDADE

Cada uma delas representa variáveis diferentes.

> **DICA**: Sempre utilize as suas variáveis em minúsculo. Adote isso como regra geral.

Criando Bons Nomes: Vamos supor que queiramos criar uma variável que indique a idade que se formou na Universidade. Temos algumas opções:

- **id**: Ruim, pois não tem significado claro e pode confundir com a variável de identificação
- **idade_formou_na_universidade**: Ruim, pois o nome da variável é muito grande, difícil de escrever e de visualizar no banco de dados.
- **idade_form**: Bom nome, significativo, minúsculo e pequeno separa os dois nomes por underline
- **idadeForm**:Bom nome, significativo, minúsculo e pequeno separa os dois nomes por uma letra maiúscula.

  DICA: Adote uma regra de criação para você e evite mudar. Crie nomes pequenos e significativos. Nunca inicie uma variável com número.

Criando projeto no R

Para saber em qual diretório o R está utilizando para salvar seu espaço de trabalho utilize o seguinte comando:

```
getwd()
```

No RStudio, sempre prefira a criação de um projeto para a organização de seus dados, com isso, ao mudar de máquina (ou estrutura de diretórios) seu código continuará funcionando normalmente.

```
  File -> New Project
```

Identação é Importante

Identar é o recuo no texto em relação a margem. É importante que esse recuo exista para linhas do seu programa que são hierarquicamente conectadas. Vejamos dois exemplos com e sem identação:

Sem Identação

```
x=c()
x[1] = 3
for (i in 2:9) {
x[i]=2*x[i-1]
}
```

Note que a quarta linha desse programa está hierarquicamente conectada a linha 3 do "for", ou seja, é uma continuação do comando e portanto deve ser identado para demonstrar essa relação de dependência. Vejamos

Com Identação

```
x=c()
x[1] = 3
for (i in 2:9) {
  x[i]=2*x[i-1]
}
```

Inserindo Dados no R

Tipos de Variáveis

O R possui diversos tipos de variáveis. Alguns desses tipos são:

Vetores: Vamos inserir os dados do valor do dólar mensal para o ano de 2015 no Brasil.

```
dolar15 <- c(2.63, 2.81, 3.14, 3.04, 3.06, 3.11, 3.22, 3.51, 3.91, 3.87, 3.77, 3.87)
dolar15
```

```
##  [1] 2.63 2.81 3.14 3.04 3.06 3.11 3.22 3.51 3.91 3.87 3.77 3.87
```

Podemos inserir vetores de texto, por exemplo, valores acima de 3.5 serão classificados como alto e baixo se forem menores

```
dolar_alto <- c("Baixo", "Baixo", "Baixo",
                "Baixo", "Baixo", "Baixo",
                "Baixo", "Alto", "Alto",
                "Alto", "Alto", "Alto")
dolar_alto
```

```
##  [1] "Baixo" "Baixo" "Baixo" "Baixo" "Baixo" "Baixo" "Baixo" "Alto"
##  [9] "Alto"  "Alto"  "Alto"  "Alto"
```

Algumas manipulações importantes que podemos fazer com os vetores. Renomeando:

```
dolar15_2=dolar15
```

Trocando o primeiro elemento do vetor e dando o print do novo resultado:

```
dolar15_2[1]=2.64
dolar15_2
```

```
##  [1] 2.64 2.81 3.14 3.04 3.06 3.11 3.22 3.51 3.91 3.87 3.77 3.87
```

Algumas maneiras de pedir o print do vetor de preço do dólar. Somente o mês 7, todos menos o mês 7, meses de 1 até 7 etc:

```
dolar15_2[7]
```

```
## [1] 3.22
```

```
dolar15_2[-7]
```

```
##  [1] 2.64 2.81 3.14 3.04 3.06 3.11 3.51 3.91 3.87 3.77 3.87
```

```
dolar15_2[1:7]
```

```
## [1] 2.64 2.81 3.14 3.04 3.06 3.11 3.22
```

Podemos incorporar novos dados no nosso vetor de preço de dólar, vamos incorporar o preço do dólar de janeiro de 2016:

```
#colcar exemplo de inserir no inicio

#inserir no meio
dolar15_16 <- c(dolar15[1:6], 3.91,dolar15[7:12])
dolar15_16
```

```
##  [1] 2.63 2.81 3.14 3.04 3.06 3.11 3.91 3.22 3.51 3.91 3.87 3.77 3.87
```

```
#troca de posicoes
temp <- dolar15_16[3]
dolar15_16[3] <- dolar15_16[7]
dolar15_16[7] <- temp
```

String ou Texto:

String são as variáveis tipo texto. Esse tipo de variável já apareceu na seção anterior quando apresentamos um vetor com a classificação do dólar em Alto e Baixo. Vejamos mais uma vez. Podemos criar uma variável que contêm "valor dólar". Uma segunda maneira é criar um vetor com dois elementos "valor" e "dólar". O comando *paste* cola a variável texto "valor" e a variável texto "dólar", separado por um espaço.

```
a <- "valor dolar"
a
```

```
## [1] "valor dolar"
```

```
b <- c("valor","dolar")
b
```

```
## [1] "valor" "dolar"
```

```
b[1]
```

```
## [1] "valor"
```

```
paste(b[1],b[2],sep=' ')
```

```
## [1] "valor dolar"
```

Fator:

Fator são variáveis de classe. Fator armazenam os valores inteiros na forma um vetor com as quantidades das k classes e o vetor string dos valores originais. Vejamos o exemplo de um vetor. Uma pesquisa classifica seus entrevistados em quatro classificações de trabalho, Empregado com Carteira (EC), Empregado Sem carteira(ES), Empregado conta Própria(EP) e Desempregado (D):

```
statusEmprego <- c("EC","EC","EC","EC","EC","EC","EC","EC","ES","ES","ES","ES","ES","ES"
                   ,"ES","ES","ES","ES","ES","ES","ES","ES","ES","ES","ES","ES","ES","ES"
                   ,"ES","ES","ES","EP","EP","EP","EP","EP","EP","EP","EP","EP","EP","D"
                   ,"D","D","D","D","D","D","D","D","D","D","D","D")
summary(statusEmprego)
```

```
##    Length     Class      Mode
##        54 character character
```

Agora vamos transformar o vetor anterior em um fator

```
statusEmprego <- factor(statusEmprego)
summary(statusEmprego)
```

```
##  D EC EP ES
## 13  8 10 23
```

```
levels(statusEmprego)
```

```
## [1] "D"  "EC" "EP" "ES"
```

O comando *levels* fornece as classes existentes, no caso acima temos 4, sendo elas 1, 2, 3 e 4.

Fatores podem ser as características de raça, gênero, status familiar, status de saúde, qualidade do atendimento etc.

Matriz:

Uma maneira de inserir a matriz é manualmente. Abaixo temos dois exemplos da inserção de uma matriz. A primeira com a primeira coluna contendo a taxa de juros e a segunda linha a taxa de câmbio (mX). Essa matriz tem 25 linhas e duas colunas conforme indicado no final do comando e é feito o preenchimento por linha (*byrow=TRUE*). A segunda contem o índice de preço ao atacado (mY).

```
mX<-matrix(c(0.8398,  2.381604545,0.7827,  2.38309,0.7599,  2.325494737, 0.8155
             ,2.231725, 0.8583, 2.220280952,0.8174, 2.23486,0.9404, 2.224021739,0.8595,
             2.26742381,0.9006, 2.332254545,0.9448, 2.447643478,0.8379, 2.54823,0.9558,
             2.638736364,0.9293,    2.633619048,0.8185, 2.815838889,1.0361, 3.138863636,
             0.9483,    3.042595,0.9838,3.061075,1.0658,3.11112381,1.1773,3.222508696,
             1.1075,    3.513695238,1.1075, 3.905809524,1.1077, 3.879504762,1.0552,
             3.7758,1.1613, 3.870495455,1.0549, 4.051715
             ), nrow=25, ncol=2, byrow=TRUE)

mY<-matrix(c(612.228, 612.63,622.729,631.556,630.191,621.279,614.909,
             609.295,611.437,610.473,616.945,624.162,625.487,625.683,
             630.401,639.559,642.933,645.114,649.636,651.148,656.475,
             673.696,688.181,693.688,698.069), nrow=25, ncol=1, byrow=TRUE)
```

Podemos fazer operações com essas matrizes. Vejamos abaixo algumas possibilidades muito úteis, principalmente para compreensão da econometria. Primeiramente fizemos a média dos elementos das linhas 1 a 5 e colunas 1 a 2. Depois fazemos a média da coluna 1. O comando *diag* cria uma matriz diagonal, o comando *t* faz a transposta, o comando *%*%* multiplica matrizes, o comando *solve* faz a inversa.

```
mean(mX[1:5,1:2])
```

```
## [1] 1.55984
```

```
mean(mX[,1])
```

```
## [1] 0.954632
```

```
diag(4)
```

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

```
diag(5, 4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    5    0    0    0
## [2,]    0    5    0    0
## [3,]    0    0    5    0
## [4,]    0    0    0    5
```

```
tX<-t(mX)
mtX<-(tX%*%mX)
mtX
```

```
##          [,1]      [,2]
## [1,] 23.15878  70.57387
## [2,] 70.57387 218.35252
```

```
mtX1<-solve(mtX)
xy<-(tX%*%mY)
```

```
xy
```

```
##           [,1]
## [1,] 15274.05
## [2,] 46431.45
```

```
beta<-(mtX1%*%xy)
beta
```

```
##            [,1]
## [1,] 765.75792
## [2,] -34.85669
```

Vejamos na tabela abaixo alguns operadores matriciais mais utilizados.

Tabela 1: Operadores Matriciais no R

Operador	Significado
A %*% B	Multiplicaçõ de matriz
crossprod(A,B)	produto cruzado A'B
crossprod(A)	produto cruzado A'A
t(A)	Transposta
diag(x)	Cria uma matriz diagonal com os elementos de x
diag(A)	Retorna um vetor contendo a diagonal principal
diag(k)	sendo k um escalar, retorna a matriz identidade
solve(A)	Inversa de A se for quadrada
y<-eigen(A) y$val	autovalores de A
y$vec	autovetores de A
y$d =	vetor contendo o valor singular de A
y$rank	é o rank de A.
cbind(A,B,...)	Combina matrizes horizontalmente.
rbind(A,B,...)	Combina matrizes verticalmente.
rowMeans(A)	Retorna o vetor de média das linhas.
rowSums(A)	Retorna o vetor de soma das linhas.
colMeans(A)	Retorna o vetor de média das colunas.
colSums(A)	Retorna o vetor de soma das colunas.

Array:

São semelhantes as matrizes mas com a diferença que podem ter mais de duas dimensões.

Data Frame ou Banco de Dados:

Esse é um tipo mais geral de variável e consegue lidar na mesma estrutura com variaveis de tipos distintos como numérica, texto e fator. Um banco de dados similar aos outros programas estatísticos. Podemos criar essa variável de forma manual. Nosso banco de dados será composto por três variáveis, a primeira o mês (mes-variável tipo texto), a segunda o valor do dólar (dólar-numérica) e a terceira o Índice de Preços ao Atacado (ipa-numérica). Vejamos abaixo a criação desses três vetores. O comando *typeof* mostra qual o tipo de variável.

```
mes=c("1995.01", "1995.02","1995.03", "1995.04","1995.05", "1995.06",
        "1995.07", "1995.08", "1995.09", "1995.10", "1995.11", "1995.12")
typeof(mes)
```

```
## [1] "character"
```

```
dolar=c(0.8451, 0.8388, 0.8874,0.9055, 0.8954, 0.9120, 0.9268, 0.9400,
        0.9508, 0.9587, 0.9624, 0.9673)
```

```
typeof(dolar)
```

```
## [1] "double"
```

```
ipa=c(109.1570, 110.5190, 110.8850, 112.0860, 112.9190, 112.3730,
      113.6980,116.4910, 116.4270, 115.5800, 116.5920, 117.4700)
typeof(ipa)
```

```
## [1] "double"
```

Para criar o banco de dados utilizamos o seguinte comando:

```
dolar_ipa<-data.frame(mes, dolar, ipa)
```

Podemos modificar o nome das variáveis com o comando *names*. Entretanto, tem que renomear todas

```
names(dolar_ipa)<-c("month", "dolar", "inf_index")
dolar_ipa
```

```
##      month  dolar inf_index
## 1  1995.01 0.8451   109.157
## 2  1995.02 0.8388   110.519
## 3  1995.03 0.8874   110.885
## 4  1995.04 0.9055   112.086
## 5  1995.05 0.8954   112.919
## 6  1995.06 0.9120   112.373
## 7  1995.07 0.9268   113.698
## 8  1995.08 0.9400   116.491
## 9  1995.09 0.9508   116.427
## 10 1995.10 0.9587   115.580
## 11 1995.11 0.9624   116.592
## 12 1995.12 0.9673   117.470
```

Ou podemos renomear somente algumas com o comando *reshape*:

```
library(reshape)
dolar_ipa <- rename(dolar_ipa, c(month="mes", inf_index="ipa"))
dolar_ipa
```

```
##        mes  dolar     ipa
## 1  1995.01 0.8451 109.157
## 2  1995.02 0.8388 110.519
## 3  1995.03 0.8874 110.885
## 4  1995.04 0.9055 112.086
## 5  1995.05 0.8954 112.919
## 6  1995.06 0.9120 112.373
## 7  1995.07 0.9268 113.698
## 8  1995.08 0.9400 116.491
## 9  1995.09 0.9508 116.427
## 10 1995.10 0.9587 115.580
## 11 1995.11 0.9624 116.592
## 12 1995.12 0.9673 117.470
```

Podemos também listar variáveis do banco de dados, por exemplo, listar colunas de 1 a 2 ou listar por nome das variáveis, conforme apresentado abaixo:

```
dolar_ipa
dolar_ipa[2:3]
```

```
dolar_ipa[1:2,2:3]
dolar_ipa[c("mes","ipa")]
```

Entretanto, inserir dados na mão pode ser uma tarefa muito penosa e existem soluções bem mais simples e rápidas para inserção de dados. Nas seções seguintes veremos aprenderemos mais funções úteis para lidar com banco de dados.

Trabalhando com as variáveis:

Vamos retomar duas variáveis dolar15 e dólar_alto e vamos manipular essas duas variáveis. Primeiramente vejamos o número de elementos, estrutura, classe e nome:

```
length(dolar15)
```

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

```
str(dolar15)
```

```
##  num [1:12] 2.63 2.81 3.14 3.04 3.06 3.11 3.22 3.51 3.91 3.87 ...
```

```
class(dolar15)
```

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

```
names(dolar15)
```

```
## NULL
```

Observamos que a variável não possui labels. Vamos colocar os Labels nessa variável, ou seja, os rótulos.

```
names(dolar15) <-c("cambio15_jan","cambio15_fev","cambio15_mar","cambio15_abr",
                   "cambio15_mai","cambio15_jun","cambio15_jul","cambio15_ago",
                   "cambio15_set","cambio15_out","cambio15_nov","cambio15_dez")
```

Podemos combinar as duas variáveis de forma distintas, por exemplo combinar na forma de um vetor, combinar como coluna ou combinar como linha, vejamos a diferença:

```
#Precisa mudar essa parte de posição está confuso pois falamos de dataframe e aqui de vetor
comb1 <- c(dolar15,dolar_alto)
comb2<- cbind(dolar15,dolar_alto)
comb3 <-rbind(dolar15,dolar_alto)
comb4 <- data.frame(
              as.numeric(dolar15),
              dolar_alto
              ,stringsAsFactors = F)
comb1
```

```
##  [1] "2.63"  "2.81"  "3.14"  "3.04"  "3.06"  "3.11"  "3.22"  "3.51"
##  [9] "3.91"  "3.87"  "3.77"  "3.87"  "Baixo" "Baixo" "Baixo" "Baixo"
## [17] "Baixo" "Baixo" "Baixo" "Alto"  "Alto"  "Alto"  "Alto"  "Alto"
```

```
comb2
```

```
##       dolar15 dolar_alto
##  [1,] "2.63"  "Baixo"
##  [2,] "2.81"  "Baixo"
##  [3,] "3.14"  "Baixo"
##  [4,] "3.04"  "Baixo"
##  [5,] "3.06"  "Baixo"
##  [6,] "3.11"  "Baixo"
##  [7,] "3.22"  "Baixo"
```

```
##  [8,] "3.51"  "Alto"
##  [9,] "3.91"  "Alto"
## [10,] "3.87"  "Alto"
## [11,] "3.77"  "Alto"
## [12,] "3.87"  "Alto"
```

```
comb3
```

```
##            [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]
## dolar15    "2.63"  "2.81"  "3.14"  "3.04"  "3.06"  "3.11"  "3.22"  "3.51"
## dolar_alto "Baixo" "Baixo" "Baixo" "Baixo" "Baixo" "Baixo" "Baixo" "Alto"
##            [,9]   [,10]  [,11]  [,12]
## dolar15    "3.91" "3.87" "3.77" "3.87"
## dolar_alto "Alto" "Alto" "Alto" "Alto"
```

```
comb4
```

```
##    as.numeric.dolar15. dolar_alto
## 1                 2.63      Baixo
## 2                 2.81      Baixo
## 3                 3.14      Baixo
## 4                 3.04      Baixo
## 5                 3.06      Baixo
## 6                 3.11      Baixo
## 7                 3.22      Baixo
## 8                 3.51       Alto
## 9                 3.91       Alto
## 10                3.87       Alto
## 11                3.77       Alto
## 12                3.87       Alto
```

```
# Problema com transformações da variável tipo fator.
a <- '30'
as.numeric(a)
```

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

```
b <- c('10','20','30','40')
b[3]
```

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

```
as.numeric(b[3])
```

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

```
aDados <- as.factor(c('10','20','30','40','30'))
aDados[5]
```

```
## [1] 30
## Levels: 10 20 30 40
```

```
as.numeric(as.character(aDados[5]))
```

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

Vejamos quais objetos temos e vamos pedir para dar visualizar os objetos que acabamos de criar. Por fim removeremos o vetor comb1.

```
ls()
comb1
```

```
comb2
comb3
rm(comb1)
rm(list=ls())
```

Importando os Dados

Está disponibilizado no googleclassroom um banco de dados contendo os valores do dólar mensal de 1995 a 2015 e a evolução do Índice de Preço ao Atacado para o mesmo período. Esse arquivo está em formato csv (comma separated values). Diretório:

Para leitura desse arquivo em csv o seguinte comando é necessário *read.csv*, indicado que possui cabeçalho e que o separador é ";"

```
dolar_ipa_total<-read.csv("banco_de_dados/dolar_ipa.csv", head=TRUE,sep=";")
```

Para leitura de arquivos em Stata terá que utilizar o pacote *foreign*, conforme exemplo abaixo:

```
library(foreign)
stata_dolar <- read.dta("banco_de_dados/dolar_ipa.dta")
```

Além desses, o R é capaz de trabalhar com SQL, SAS, SPSS, Excel entre outros.

Exportanto os Dados

Podemos exportar os dados em diferentes formatos. Alguns exemplos são csv, texto delimitado, excel, stata. Vejamos em csv:

```
write.table(dolar_ipa_total, "banco_de_dados/dolarIpa.csv", sep=";")
```

Para exportar em Stata utilize os seguintes comandos:

```
library(foreign)
write.dta(dolar_ipa, paste(getwd(),"/../Banco de dados/dolarIpa.dta",sep=''))
```

Características dos Dados

Lidando com Dados Missing

Uma maneira de lidar com valores missing, por exemplo a partir de 2016 todos os valores são missing, seria fazer um subconjunto que veremos mais a frente. Agora seguiremos alguns passos para analisar os valores missing do nosso banco de dados. Primeiramente, analisamos se há valores missing no banco de dados:

```
is.na(dolar_ipa_total)
```

Podemos excluir os valores missing da análise de interesse, vamos fazer a média do dolar sem considerar os valores missing:

```
mean(dolar_ipa_total$tx_cambio)
```

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

```
mean(dolar_ipa_total$tx_cambio, na.rm=TRUE)
```

```
## [1] 2.040776
```

Podemos criar um novo banco de dados sem os valores missing.

```
dolar_ipa_total1 <- na.omit(dolar_ipa_total)
mean(dolar_ipa_total1$tx_cambio)
```

```
## [1] 2.040776
```

Outra maneira de excluir os valores missing seria a utilização do comando *subset* removendo as observações que contenham valor missing. Isso será explicado em seção a frente.

Pode-se também recodificar uma determinada variável para missing. Muito comum nas pesquisas do IBGE os valores missing serem identificados por um número, por exemplo 999999999999. Dessa forma podemos indicar que esse não é número e sim um valor missing da seguinte maneira:

```
dolar_ipa_total1$tx_cambio[dolar_ipa_total1$tx_cambio==99] <- NA
```

Operando o Banco de Dados

Criando uma Nova Variável

Vamos criar uma variável que seria o dobro da taxa de câmbio e chamaremos essa variável de cambio2x. Para criar a variável precisamos dizer primeiro qual o banco de dados em que queremos criar e qual o nome da variável separa por um . Depois temos que multiplicar por 2 a variável taxa de câmbio que está no banco de dados dolar_ipa_total1, conforme apresentado na expressão abaixo.

```
dolar_ipa_total1$cambio2x <- 2*dolar_ipa_total1$tx_cambio
```

Agora vamos criar uma variável binária que representa a mudança da âncora cambial no Brasil que ocorreu em janeiro de 1999. Novamente, precisamos indicar o banco de dados e o nome da variável no banco de dados.

```
#attach(dolar_ipa_total1)
dolar_ipa_total1$ancora[dolar_ipa_total1$Data > 1999.01] <- 0
dolar_ipa_total1$ancora[dolar_ipa_total1$Data <= 1999.01] <- 1
#head(dolar_ipa_total1)
#detach(dolar_ipa_total1)
dolar_ipa_total1$Data
```

```
##   [1] 1995.01 1995.02 1995.03 1995.04 1995.05 1995.06 1995.07 1995.08
##   [9] 1995.09 1995.10 1995.11 1995.12 1996.01 1996.02 1996.03 1996.04
##  [17] 1996.05 1996.06 1996.07 1996.08 1996.09 1996.10 1996.11 1996.12
##  [25] 1997.01 1997.02 1997.03 1997.04 1997.05 1997.06 1997.07 1997.08
##  [33] 1997.09 1997.10 1997.11 1997.12 1998.01 1998.02 1998.03 1998.04
##  [41] 1998.05 1998.06 1998.07 1998.08 1998.09 1998.10 1998.11 1998.12
##  [49] 1999.01 1999.02 1999.03 1999.04 1999.05 1999.06 1999.07 1999.08
##  [57] 1999.09 1999.10 1999.11 1999.12 2000.01 2000.02 2000.03 2000.04
##  [65] 2000.05 2000.06 2000.07 2000.08 2000.09 2000.10 2000.11 2000.12
##  [73] 2001.01 2001.02 2001.03 2001.04 2001.05 2001.06 2001.07 2001.08
##  [81] 2001.09 2001.10 2001.11 2001.12 2002.01 2002.02 2002.03 2002.04
##  [89] 2002.05 2002.06 2002.07 2002.08 2002.09 2002.10 2002.11 2002.12
##  [97] 2003.01 2003.02 2003.03 2003.04 2003.05 2003.06 2003.07 2003.08
## [105] 2003.09 2003.10 2003.11 2003.12 2004.01 2004.02 2004.03 2004.04
## [113] 2004.05 2004.06 2004.07 2004.08 2004.09 2004.10 2004.11 2004.12
## [121] 2005.01 2005.02 2005.03 2005.04 2005.05 2005.06 2005.07 2005.08
## [129] 2005.09 2005.10 2005.11 2005.12 2006.01 2006.02 2006.03 2006.04
## [137] 2006.05 2006.06 2006.07 2006.08 2006.09 2006.10 2006.11 2006.12
## [145] 2007.01 2007.02 2007.03 2007.04 2007.05 2007.06 2007.07 2007.08
## [153] 2007.09 2007.10 2007.11 2007.12 2008.01 2008.02 2008.03 2008.04
## [161] 2008.05 2008.06 2008.07 2008.08 2008.09 2008.10 2008.11 2008.12
## [169] 2009.01 2009.02 2009.03 2009.04 2009.05 2009.06 2009.07 2009.08
## [177] 2009.09 2009.10 2009.11 2009.12 2010.01 2010.02 2010.03 2010.04
## [185] 2010.05 2010.06 2010.07 2010.08 2010.09 2010.10 2010.11 2010.12
## [193] 2011.01 2011.02 2011.03 2011.04 2011.05 2011.06 2011.07 2011.08
## [201] 2011.09 2011.10 2011.11 2011.12 2012.01 2012.02 2012.03 2012.04
## [209] 2012.05 2012.06 2012.07 2012.08 2012.09 2012.10 2012.11 2012.12
## [217] 2013.01 2013.02 2013.03 2013.04 2013.05 2013.06 2013.07 2013.08
## [225] 2013.09 2013.10 2013.11 2013.12 2014.01 2014.02 2014.03 2014.04
## [233] 2014.05 2014.06 2014.07 2014.08 2014.09 2014.10 2014.11 2014.12
## [241] 2015.01 2015.02 2015.03 2015.04 2015.05 2015.06 2015.07 2015.08
## [249] 2015.09 2015.10 2015.11 2015.12 2016.01
```

Evite utilizar os comandon *attach* e *detach*, eles podem gerar problemas com variáveis do tipo fator e pode

gerar erros na compilação. Expresse completamente seu programa com a tabela e a variável a ser utilizada. Podemos também atualizar os nossos dados com os valores para o anos recentes. Para isso utilize o comando

```
Data=c("2016.02", "2016.03")
tx_cambio=c(3.8, 3.7)
IPA<-c(670.0000, 671.0000)
cambio2x<-c(7.6, 7.4)
ancora<-c(0,0)

dolar_ipa_novo<-data.frame(Data,tx_cambio,IPA,cambio2x,ancora)
dolar_atualizado <- rbind(dolar_ipa_total1, dolar_ipa_novo)
```

Operadores Aritméticos e Lógicos

```
dolar15>3
```

```
##  [1] FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [12]  TRUE
```

```
dolar15[dolar15>3]
```

```
##  [1] 3.14 3.04 3.06 3.11 3.22 3.51 3.91 3.87 3.77 3.87
```

```
dolar15[dolar15 < 3.0 | dolar15 > 3.5]
```

```
## [1] 2.63 2.81 3.51 3.91 3.87 3.77 3.87
```

```
dolar15[dolar15 > 3.0 & dolar15 < 3.5]
```

```
## [1] 3.14 3.04 3.06 3.11 3.22
```

```
mean(dolar15)
```

```
## [1] 3.328333
```

```
dolarM <- (dolar15[dolar15 < 3.5 & dolar15 > 3.0])
dolarM
```

```
## [1] 3.14 3.04 3.06 3.11 3.22
```

Tabela 2: Operadores Lógicos no R

Operador	Significado
<	Menor que
<=	Menor igual
>	Maior que
>=	Maior igual
==	Exatamente igual
!=	Diferente
!x	Não x
x | y	x OU y
x & y	x E y

Algumas Funções Importantes

Ordenando os Dados

```
dolar_order<-dolar_ipa_total1[order(dolar_ipa_total1$tx_cambio),]

dolar_order1<-dolar_ipa_total1[order(dolar_ipa_total1$tx_cambio, decreasing = TRUE),]
```

Fazendo Merge

Primeiramente vamos fazer a inserção de um novo banco de dados. Esse banco de dados contem as datas e as taxas de juros mensais.

```
tx_juros<-read.csv(file="banco_de_dados/tx_juros.csv", head=TRUE,sep=";")
names(tx_juros)<-c("Data", "Juros")
```

Agora precisamos juntar o arquivo da taxa de juros com o arquivo que contem os valores da taxa de câmbio.

```
dolar_juros <- merge(dolar_ipa_total1,tx_juros,by="Data",keep.all=TRUE)
```

Agora temos um banco único com taxa de câmbio, taxa de juros, ipa e âncora cambial

Agregando

Vamos criar um banco de dados que contenha os valores médios das variáveis. Para isso vamos agregar fazendo a média. Poderíamos utilizar outra função, como a soma, para fazer a agregação:

```
dados_media <-aggregate(dolar_juros, by=list(dolar_juros$ancora), FUN=mean, na.rm=TRUE)
dados_media
```

```
##   Group.1     Data tx_cambio      IPA cambio2x ancora    Juros
## 1       0 2007.148  2.279075 412.2728 4.558150      0 1.129818
## 2       1 1996.615  1.048676 126.7311 2.097351      1 2.395120
```

Vejamos a tabela de média por período com âncora cambial e sem âncora cambial. Primeiramente instale no R o pacote *xtable* e carregue a biblioteca:

```
library(xtable)
options(xtable.floating = FALSE)
options(xtable.timestamp = "")
```

Agora podemos criar uma tabela com os valores médios do dólar:

```
dolar_ancora<-xtable(dados_media, caption="Valor do dolar antes e após ancora cambial"
                     ,  label="Table:ancora")
print(dolar_ancora,include.rownames = FALSE)
```

Vejamos o resultado:

Tabela - Valores médios do dólar, inflação e juros antes e após ancora cambial.

```
dolar_ancora<-xtable(dados_media, caption="Valor do dólar antes e após ancora cambial"
                     ,  label="Table:ancora")
print(dolar_ancora,include.rownames = FALSE)
```

Group.1	Data	tx_cambio	IPA	cambio2x	ancora	Juros
0.00	2007.15	2.28	412.27	4.56	0.00	1.13
1.00	1996.61	1.05	126.73	2.10	1.00	2.40

Criando Subconjunto

Selecionando Variáveis:

Podemos estar interessado em manter somente algumas variáveis no nosso banco de dados, por exemplo, queremos manter Data, tx_cambio e IPA. Assim:

```
var_sel <- c("Data", "tx_cambio", "IPA")
dolar_sel1 <- dolar_juros[var_sel]

dolar_sel11 <- dolar_juros[c("Data", "tx_cambio", "IPA")]
```

Ou podemos fazer assim

```
dolar_sel2 <- dolar_juros[c(1:3,5:6)]
dolar_sel21 <- dolar_juros[c(1:3,5:ncol(dolar_juros))]
```

Excluindo Variáveis:

Agora vamos deletar a variável cambio2x e ancora que ficam na coluna 4 e 5, assim:

```
dolar_sel3 <- dolar_juros[c(-4,-5)]
```

Ou podemos fazer assim

```
dolar_juros1<-dolar_juros
dolar_juros1$cambio2x <- dolar_juros1$ancora <- NULL
```

Selecionando Variáveis:

Vamos selecionar as observações que pertencem ou estiveram no sistema de âncora cambial ou seja, onde ancora for 1:

```
  dolar_ancora <- dolar_juros[ which(dolar_juros$ancora==1),]
```

```
#atribuição no lugar de valor lógico - ERRO
dolar_ancora1 <- dolar_juros1[ which(dolar_juros1$ancora=1),]
```

Ou podemos selecionar as observações em que o dólar esteve entre R$2 e R$3:

```
dolar_int <- subset(dolar_juros, dolar_juros$tx_cambio >= 2 & dolar_juros$tx_cambio <= 3 )
```

Estatísticas Básicas

Uma boa análise descritiva é parte fundamental de qualquer trabalho realizado em economia aplicada. Vamos descrever algumas funções que podem nos auxiliar.

Estatísticas Descritivas

Para vetores e especificando a variável em um banco de dados podemos utilizar os seguintes comandos para calcular média, mediana, variância, máximo, encontrar elementos, saber qual o posicionamento de determinado elemento, e o tamanho do banco conforme apresentamos abaixo.

```
mean(dolar15)
```

```
## [1] 3.328333
```

```
mean(dolar_juros$tx_cambio)
```

```
## [1] 2.040776
```

```
median (dolar15)
```

```
## [1] 3.18
```

```
var(dolar15)
```

```
## [1] 0.1963242
```

```
max(dolar15)
```

```
## [1] 3.91
```

```
dolar15 == 3.91
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [12] FALSE
```

```
which(dolar15==3.91)
```

```
## [1] 9
```

```
length(dolar_ipa_total$tx_cambio) #tamanho
```

```
## [1] 757
```

```
length(na.omit(dolar_ipa_total$tx_cambio)) #tamanho da série
```

```
## [1] 253
```

Entretanto, há outras opções mais práticas para realizar a análise descritiva. Vamos retomar aqui o banco de dados dólar_juros e vamos fazer a estatística descritiva dos dados. Esse banco contêm os valores de câmbio, juros e índice de preço ao atacado desde 1995 até 2015, mensalmente. Totaliza 255 observações.

Alguns exemplos de funções úteis para realizar a análise descritiva são:

```
sapply(dolar_juros, mean, na.rm=TRUE)
sapply(dolar_juros, quantile, na.rm=TRUE)
```

Outras funções que pode ser utilizadas nesse comando são *sd, var, min, max, median, range, e quantile*

Podemos também utilizar

```
summary(dolar_juros)
```

```
##       Data        tx_cambio           IPA           cambio2x
##  Min.   :1995   Min.   :0.8388   Min.   :109.2   Min.   :1.678
##  1st Qu.:2000   1st Qu.:1.6881   1st Qu.:181.4   1st Qu.:3.376
##  Median :2005   Median :1.9823   Median :373.8   Median :3.965
##  Mean   :2005   Mean   :2.0408   Mean   :357.0   Mean   :4.082
##  3rd Qu.:2010   3rd Qu.:2.3771   3rd Qu.:497.9   3rd Qu.:4.754
##  Max.   :2016   Max.   :4.0517   Max.   :698.1   Max.   :8.103
##      ancora           Juros
##  Min.   :0.0000   Min.   :0.4816
##  1st Qu.:0.0000   1st Qu.:0.8807
##  Median :0.0000   Median :1.2236
##  Mean   :0.1937   Mean   :1.3749
##  3rd Qu.:0.0000   3rd Qu.:1.5969
##  Max.   :1.0000   Max.   :4.4082
```

```
summary(dolar_juros[2:6])
```

```
##    tx_cambio           IPA           cambio2x         ancora
##  Min.   :0.8388   Min.   :109.2   Min.   :1.678   Min.   :0.0000
##  1st Qu.:1.6881   1st Qu.:181.4   1st Qu.:3.376   1st Qu.:0.0000
##  Median :1.9823   Median :373.8   Median :3.965   Median :0.0000
##  Mean   :2.0408   Mean   :357.0   Mean   :4.082   Mean   :0.1937
##  3rd Qu.:2.3771   3rd Qu.:497.9   3rd Qu.:4.754   3rd Qu.:0.0000
##  Max.   :4.0517   Max.   :698.1   Max.   :8.103   Max.   :1.0000
##      Juros
##  Min.   :0.4816
##  1st Qu.:0.8807
##  Median :1.2236
##  Mean   :1.3749
##  3rd Qu.:1.5969
##  Max.   :4.4082
```

```
summary(dolar_juros[-c(1,4)])
```

```
##    tx_cambio           IPA             ancora           Juros
##  Min.   :0.8388   Min.   :109.2   Min.   :0.0000   Min.   :0.4816
##  1st Qu.:1.6881   1st Qu.:181.4   1st Qu.:0.0000   1st Qu.:0.8807
##  Median :1.9823   Median :373.8   Median :0.0000   Median :1.2236
##  Mean   :2.0408   Mean   :357.0   Mean   :0.1937   Mean   :1.3749
##  3rd Qu.:2.3771   3rd Qu.:497.9   3rd Qu.:0.0000   3rd Qu.:1.5969
##  Max.   :4.0517   Max.   :698.1   Max.   :1.0000   Max.   :4.4082
```

Ou ainda

```
install.packages("psych")
library(psych)
```

```
describe(dolar_juros, na.rm=TRUE)
describe(dolar_juros[2:7], na.rm=TRUE)
```

E podemos descrever os dado por grupo, por exemplo, vamos fazer a análise descritiva antes e depois da âncora cambial

```
describeBy(dolar_juros, dolar_juros$ancora)
describeBy(dolar_juros[2:8], dolar_juros$ancora)
describeBy(dolar_juros[2:8], dolar_juros$ano)
```

Frequência e Tabulações Cruzadas

Com base no banco de dados dólar_juros, vamos criar uma variável ano por meio do arredondamento. O comando *floor(x)* faz o arredondamento para baixo, *ceiling(x)* faz o arredondamento para cima e *round(x, digits = 2)* faz o arredondamento para duas casas decimais.

```
dolar_juros$ano <- dolar_juros$Data
dolar_juros$ano<-floor(dolar_juros$ano)

as.numeric(substr(dolar_juros$ano,1,4))
```

```
##   [1] 1995 1995 1995 1995 1995 1995 1995 1995 1995 1995 1995 1995 1996 1996
##  [15] 1996 1996 1996 1996 1996 1996 1996 1996 1996 1996 1997 1997 1997 1997
##  [29] 1997 1997 1997 1997 1997 1997 1997 1997 1998 1998 1998 1998 1998 1998
##  [43] 1998 1998 1998 1998 1998 1998 1999 1999 1999 1999 1999 1999 1999 1999
##  [57] 1999 1999 1999 1999 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000
##  [71] 2000 2000 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001
##  [85] 2002 2002 2002 2002 2002 2002 2002 2002 2002 2002 2002 2002 2003 2003
##  [99] 2003 2003 2003 2003 2003 2003 2003 2003 2003 2003 2004 2004 2004 2004
## [113] 2004 2004 2004 2004 2004 2004 2004 2004 2005 2005 2005 2005 2005 2005
## [127] 2005 2005 2005 2005 2005 2005 2006 2006 2006 2006 2006 2006 2006 2006
## [141] 2006 2006 2006 2006 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007
## [155] 2007 2007 2008 2008 2008 2008 2008 2008 2008 2008 2008 2008 2008 2008
## [169] 2009 2009 2009 2009 2009 2009 2009 2009 2009 2009 2009 2009 2010 2010
## [183] 2010 2010 2010 2010 2010 2010 2010 2010 2010 2010 2011 2011 2011 2011
## [197] 2011 2011 2011 2011 2011 2011 2011 2011 2012 2012 2012 2012 2012 2012
## [211] 2012 2012 2012 2012 2012 2012 2013 2013 2013 2013 2013 2013 2013 2013
## [225] 2013 2013 2013 2013 2014 2014 2014 2014 2014 2014 2014 2014 2014 2014
## [239] 2014 2014 2015 2015 2015 2015 2015 2015 2015 2015 2015 2015 2015 2015
## [253] 2016
```

Vamos agora fazer uma tabela de frequência, considerando a variável âncora cambial e ano.

```
library(xtable)
freqAnoAnc <- table(dolar_juros$ano,dolar_juros$ancora)
freqAnoAnc
```

```
##
##         0  1
##   1995  0 12
##   1996  0 12
##   1997  0 12
##   1998  0 12
##   1999 11  1
##   2000 12  0
##   2001 12  0
##   2002 12  0
##   2003 12  0
##   2004 12  0
##   2005 12  0
##   2006 12  0
##   2007 12  0
##   2008 12  0
##   2009 12  0
##   2010 12  0
##   2011 12  0
```

```
##   2012 12  0
##   2013 12  0
##   2014 12  0
##   2015 12  0
##   2016  1  0
```

```
print(xtable(freqAnoAnc))
```

```
## \begin{tabular}{rrr}
##   \hline
##  & 0 & 1 \\
##   \hline
## 1995 &   0 &  12 \\
##   1996 &   0 &  12 \\
##   1997 &   0 &  12 \\
##   1998 &   0 &  12 \\
##   1999 &  11 &   1 \\
##   2000 &  12 &   0 \\
##   2001 &  12 &   0 \\
##   2002 &  12 &   0 \\
##   2003 &  12 &   0 \\
##   2004 &  12 &   0 \\
##   2005 &  12 &   0 \\
##   2006 &  12 &   0 \\
##   2007 &  12 &   0 \\
##   2008 &  12 &   0 \\
##   2009 &  12 &   0 \\
##   2010 &  12 &   0 \\
##   2011 &  12 &   0 \\
##   2012 &  12 &   0 \\
##   2013 &  12 &   0 \\
##   2014 &  12 &   0 \\
##   2015 &  12 &   0 \\
##   2016 &   1 &   0 \\
##    \hline
## \end{tabular}
```

```
freqAnoAnc1<-as.data.frame.matrix(freqAnoAnc)
```

Podemos também fazer uma tabela de frequências marginais

```
margin.table(freqAnoAnc, 1)
```

```
##
## 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009
##   12   12   12   12   12   12   12   12   12   12   12   12   12   12   12
## 2010 2011 2012 2013 2014 2015 2016
##   12   12   12   12   12   12    1
```

```
margin.table(freqAnoAnc, 2)
```

```
##
##   0   1
## 204  49
```

Ou uma tabela de proporções

```
prop.table(freqAnoAnc)
prop.table(freqAnoAnc, 1)
prop.table(freqAnoAnc, 2)
```

Outra forma de tabela, é aquela que possui duas variáveis categóricas na linha e coluna e a média de uma outra variável como resultado. Muito parecido com a tabela dinâmica do Excel. Queremos o valor de dólar médio por ano e por âncora cambial.

```
install.packages("gdata")
library(gdata)
```

```
tabMedDolar<- tapply(dolar_juros$tx_cambio, list(dolar_juros$ano, dolar_juros$ancora), mean)
tabMedDolar
```

```
##             0        1
## 1995       NA 0.915850
## 1996       NA 1.004242
## 1997       NA 1.077192
## 1998       NA 1.159717
## 1999 1.842373 1.501100
## 2000 1.829408       NA
## 2001 2.349633       NA
## 2002 2.920350       NA
## 2003 3.077483       NA
## 2004 2.925117       NA
## 2005 2.434392       NA
## 2006 2.175325       NA
## 2007 1.947058       NA
## 2008 1.833767       NA
## 2009 1.996767       NA
## 2010 1.759408       NA
## 2011 1.674183       NA
## 2012 1.954000       NA
## 2013 2.157050       NA
## 2014 2.352942       NA
## 2015 3.330908       NA
## 2016 4.051700       NA
```

```
tabObsDolar<- tapply(dolar_juros$tx_cambio, list(dolar_juros$ano, dolar_juros$ancora), nobs)
tabObsDolar
```

```
##       0  1
## 1995 NA 12
## 1996 NA 12
## 1997 NA 12
## 1998 NA 12
## 1999 11  1
## 2000 12 NA
## 2001 12 NA
## 2002 12 NA
## 2003 12 NA
## 2004 12 NA
## 2005 12 NA
## 2006 12 NA
## 2007 12 NA
## 2008 12 NA
```

```
## 2009 12 NA
## 2010 12 NA
## 2011 12 NA
## 2012 12 NA
## 2013 12 NA
## 2014 12 NA
## 2015 12 NA
## 2016  1 NA
```

Correlação

A correlação é uma medida de associação muito utilizada em análises descritivas de dados. O comando apresentado faz três tipos de correlação a de *pearson*, *kendall* e *spearman*. Vejamos como realizar

```
cor(dolar_juros[2:6], use="complete.obs", method="pearson")
```

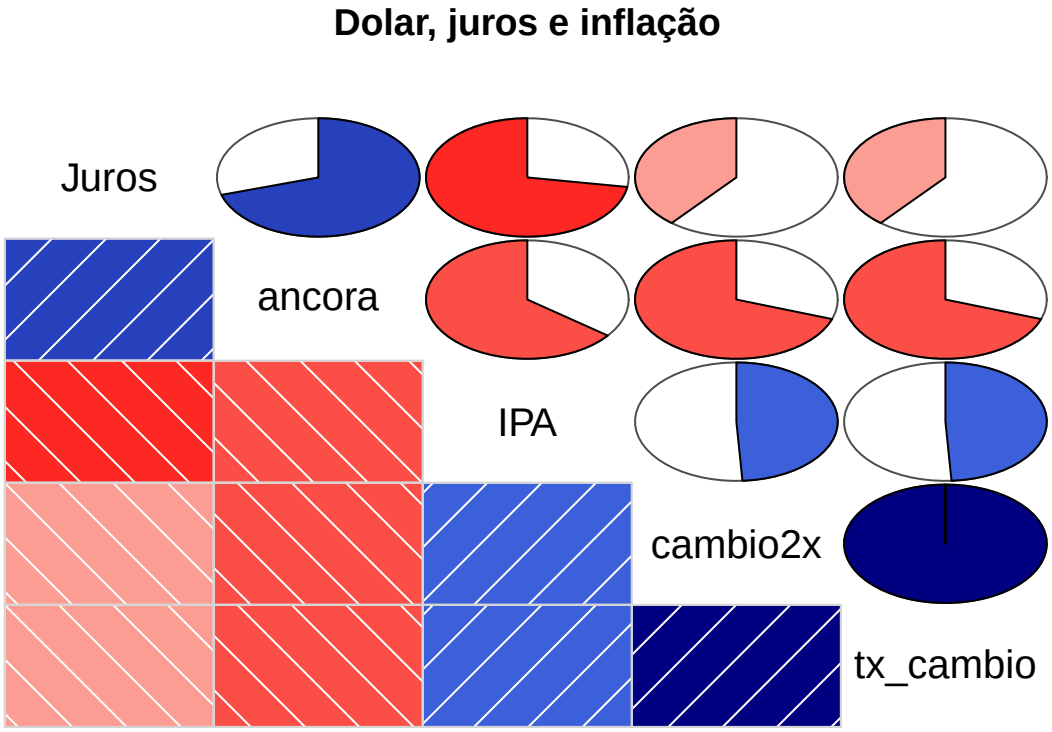
```
##            tx_cambio        IPA   cambio2x     ancora      Juros
## tx_cambio  1.0000000  0.4904771  1.0000000 -0.6965230 -0.3898958
## IPA        0.4904771  1.0000000  0.4904771 -0.6458906 -0.7242958
## cambio2x   1.0000000  0.4904771  1.0000000 -0.6965230 -0.3898958
## ancora    -0.6965230 -0.6458906 -0.6965230  1.0000000  0.7024247
## Juros     -0.3898958 -0.7242958 -0.3898958  0.7024247  1.0000000
```

```
cov(dolar_juros[2:6], use="complete.obs")
```

```
##            tx_cambio         IPA    cambio2x      ancora       Juros
## tx_cambio  0.4892451    60.05446   0.9784902  -0.1929085  -0.1945176
## IPA       60.0544605 30642.66993 120.1089210 -44.7687354 -90.4328937
## cambio2x   0.9784902   120.10892   1.9569804  -0.3858170  -0.3890353
## ancora    -0.1929085   -44.76874  -0.3858170   0.1567852   0.1983807
## Juros     -0.1945176   -90.43289  -0.3890353   0.1983807   0.5087381
```

O R oferece grande facilidade de visualização dos resultados. Em seção a frente veremos como elaborar gráficos, por enquanto ficamos com esse exemplo de correlograma

```
library(corrgram)
corrgram(dolar_juros[2:6], order=TRUE,
         lower.panel=panel.shade,
  upper.panel=panel.pie, text.panel=panel.txt,
  main="Dolar, juros e inflação")
```



Teste-t

Pode-se testar se o valor do dólar no período de âncora cambial foi estatisticamente diferente do período sem âncora, pode realizar o teste-t.

```
t.test(dolar_juros$tx_cambio~dolar_juros$ancora)
```

```
##
##  Welch Two Sample t-test
##
## data:  dolar_juros$tx_cambio by dolar_juros$ancora
## t = 29.094, df = 248.47, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.147106 1.313693
## sample estimates:
## mean in group 0 mean in group 1
##        2.279075        1.048676
```

Ou podemos testar a hipótese que o dólar foi estatisticamente igual a 3 de 1995 a 2015.

```
t.test(dolar_juros$tx_cambio,mu=3)
```

```
##
##  One Sample t-test
##
## data:  dolar_juros$tx_cambio
## t = -21.813, df = 252, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 3
## 95 percent confidence interval:
##  1.954172 2.127381
```

```
## sample estimates:
## mean of x
##  2.040776
```

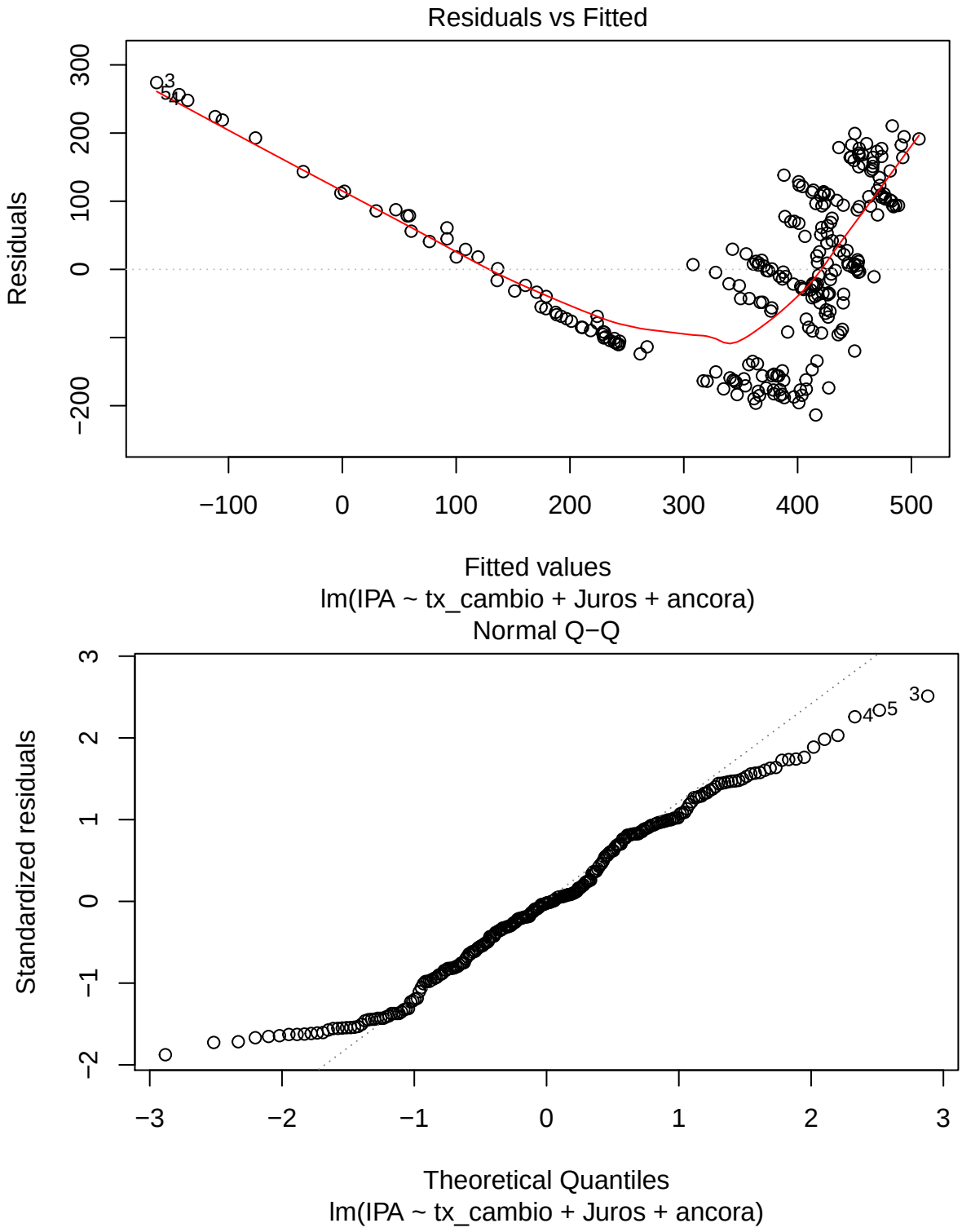
Regressão

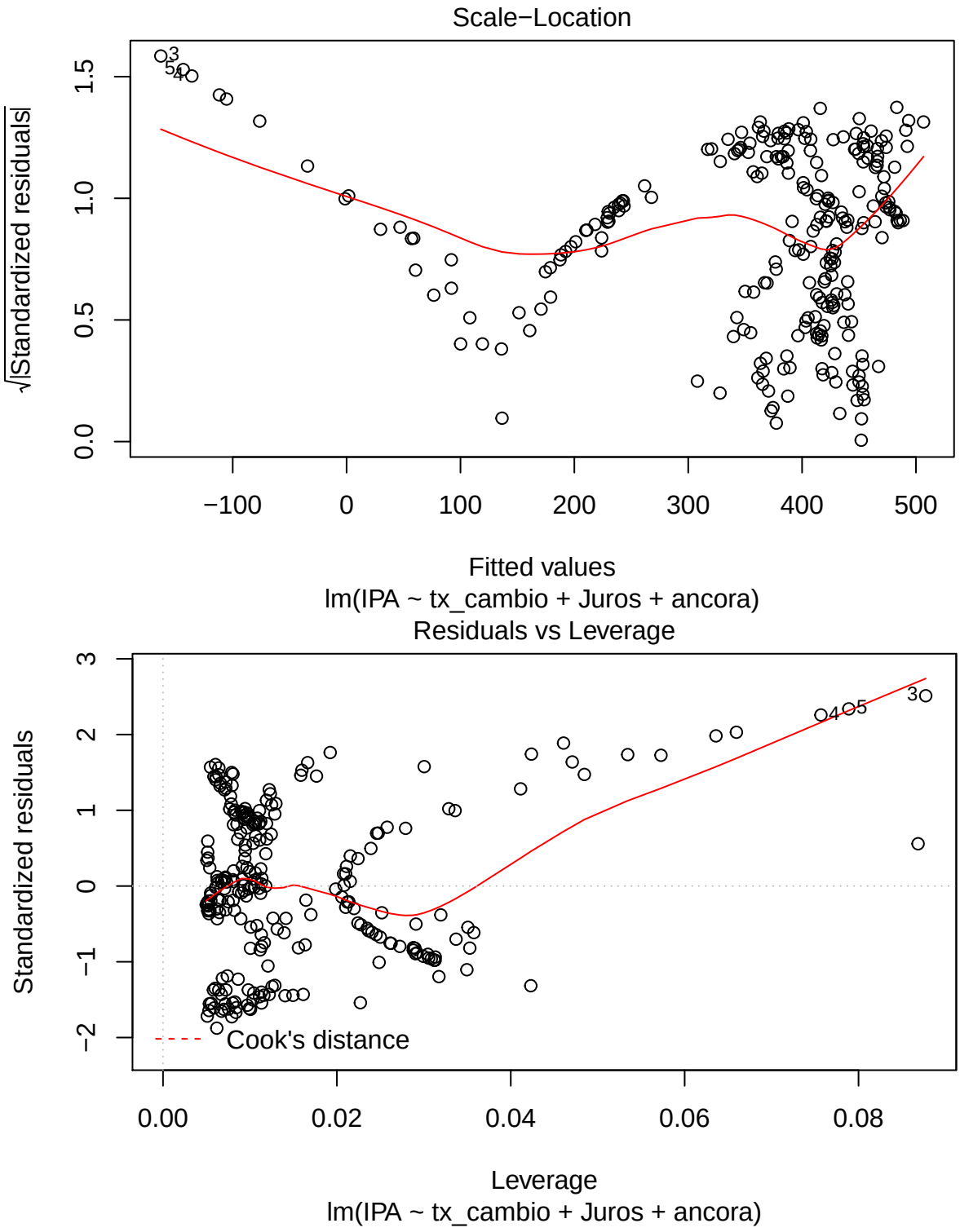
A análise de regressão se tornou ferramenta fundamental na análise econômica. Ela permite que haja entendimento das relações entre variáveis e da magnitude dessas relações. Vamos ver qual a influência do dólar e juros sobre o índice de preço ao atacado.

```
fit1 <- lm(IPA ~ tx_cambio + Juros+ancora, data=dolar_juros)
summary(fit1)
```

```
##
## Call:
## lm(formula = IPA ~ tx_cambio + Juros + ancora, data = dolar_juros)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -213.597  -89.754   -2.223   93.791  274.016
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   462.81      35.04  13.207  < 2e-16 ***
## tx_cambio      47.32      14.61   3.238  0.00137 **
## Juros        -140.20      14.45  -9.704  < 2e-16 ***
## ancora        -49.92      33.40  -1.495  0.13626
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 114.2 on 249 degrees of freedom
## Multiple R-squared:  0.5794, Adjusted R-squared:  0.5744
## F-statistic: 114.4 on 3 and 249 DF,  p-value: < 2.2e-16
```

```
plot(fit1)
```


Residuals vs Fitted
Residuals
300
200
100
0
−200
3
5
4
−100
0
100
200
300
400
500
Fitted values
lm(IPA ~ tx_cambio + Juros + ancora)
Normal Q−Q
Standardized residuals
3
2
1
0
−1
−2
3
4
5
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(IPA ~ tx_cambio + Juros + ancora)



Em capítulo a frente veremos detalhadamente a análise de regressão.

Utilizando by e with

O comando *with()* pode ser utilizado para realizar um teste utilizando uma variável de classe, como a âncora cambial. Veja abaixo a utilização do comando para fazer o teste t de diferença de médias da taxa de câmbio, na âncora cambial e fora desse regime.

```
  with(dolar_juros, t.test(tx_cambio ~ ancora))
```

O comando **with** pode facilitar a digitação de comandos mais complexos. O equivalente ao comando anterior (sem o **with**) seria:

```
  t.test(dolar_juros$tx_cambio ~ dolar_juros$ancora)
```

O comando *by()* pode ser utilizado para aplicar uma função em uma lista ou dataframe, agrupando por um parâmetro. Por exemplo, fazer a média da taxa de câmbio antes e depois da âncora cambial.

```
by(dolar_juros$tx_cambio, dolar_juros$ancora, function(x) mean(x))
 by(dolar_juros$tx_cambio, dolar_juros$ancora, function(x) c(sd(x),mean(x)))
```

Onde:

- 1 - dolar_juros$tx_cambio é a lista na qual será aplicada a função
- 2 - dolar_juros$ancora são os grupos que serão utilizados
- 3 - a função a ser aplicada, no caso **mean()**

Essa é outra maneira de utilizar o comando mas agora fazendo uma regressão ao invés da média.

```
by(dolar_juros, dolar_juros[,"ancora"], function(x) describe(x))

by(dolar_juros, dolar_juros[,"ancora"],
     function(x) lm(IPA ~ tx_cambio + Juros, data=x))
```

Ou podemos fazer assim

```
  tmp <- with(dolar_juros,
              by(dolar_juros, ancora,
           function(x) lm(IPA ~ tx_cambio + Juros, data=x)
                 )
             )

sapply(tmp, coef)
```

Gráficos

A interface gráfica do R é robusta e poderosa. Os desenvolvedores do R fizeram grande esforço de desenvolvimento de análise de dados via visualização gráfica. Pode-se dizer que essa é uma das vantagens do uso do R. Entretanto, como o controle é total sobre figuras e gráficos, isso torna a elaboração um pouco mais complicada.

> **DICA**: Desenvolva um bom modelo para cada tipo de gráfico e utilize esse para elaborar seus trabalhos. Um custo inicial mais alto que se amortiza na utilização.

Criando Gráficos

A criação de um gráfico pode ser bem simples utilizando o comando *plot(x,y)*. Podemos ainda facilmente adicionar uma linha de regressão utilizando o comando *abline()*. Entretanto o padrão do R é simples e sem cor. Veja abaixo:

```
plot(dolar_juros$IPA, dolar_juros$tx_cambio)
  abline(lm(dolar_juros$tx_cambio ~ dolar_juros$IPA))
  title("Gráfico de Dispersão IPA e taxa de câmbio: linha de regressão")
```

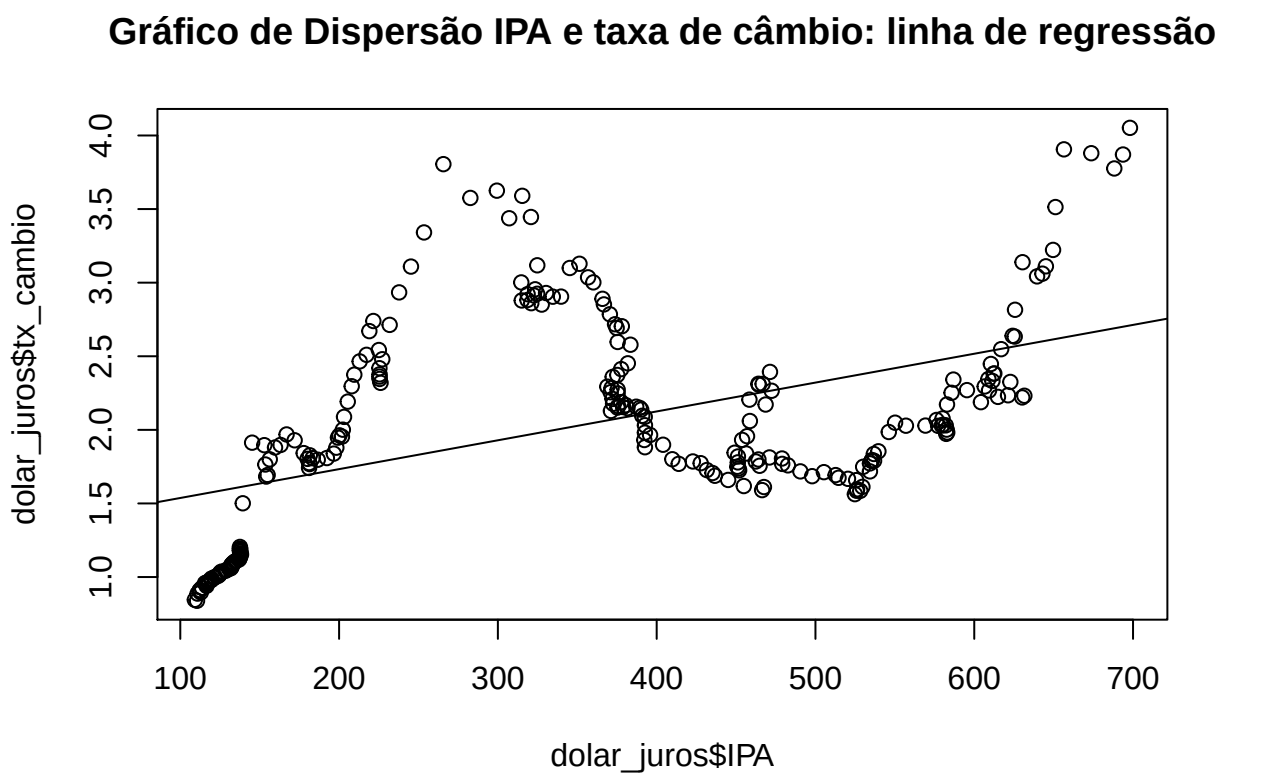
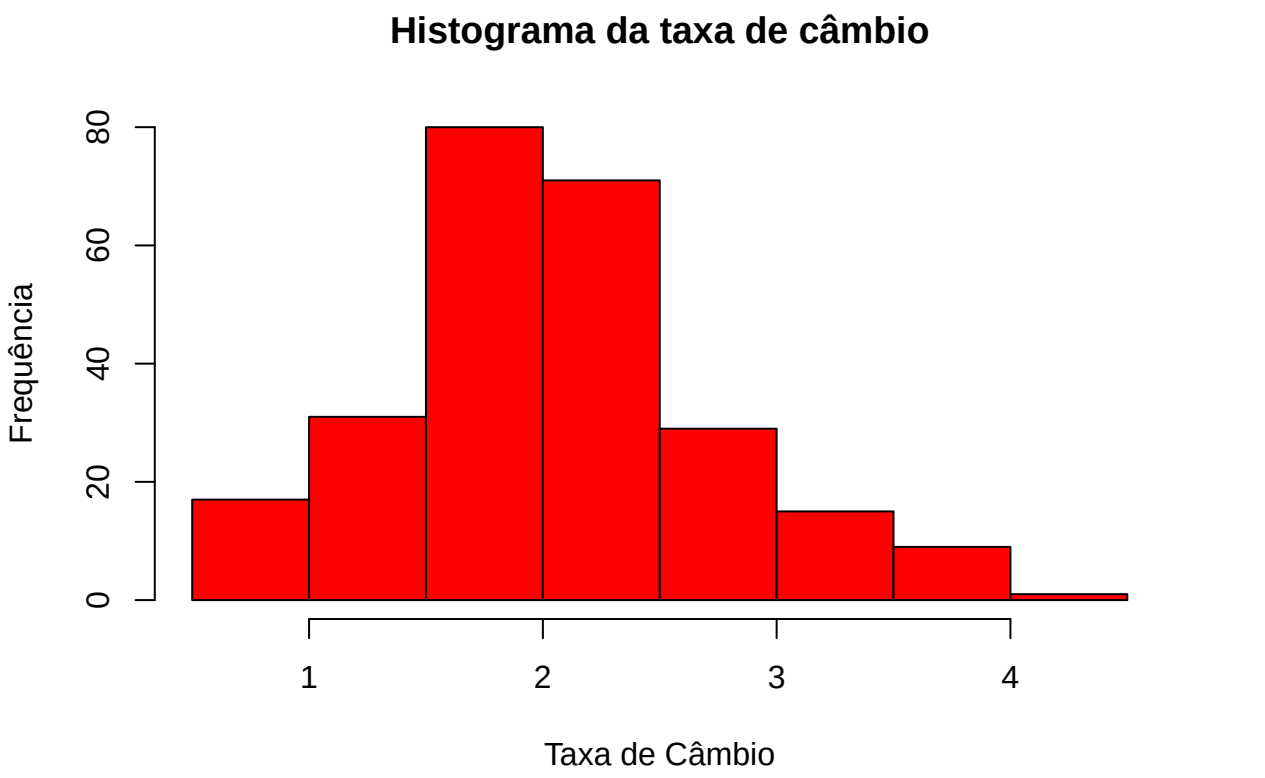


Gráfico de Densidade

Visualizar a distribuição empírica dos dados fornece uma grande quantidade de informação. Um gráfico básico em análise descritiva é o histograma, o qual fornece a distribuição de probabilidade empírica dos dados em um formato de barras. A área do histograma é igual a 1 e altura da sua barra da a densidade de observações em cada classe.

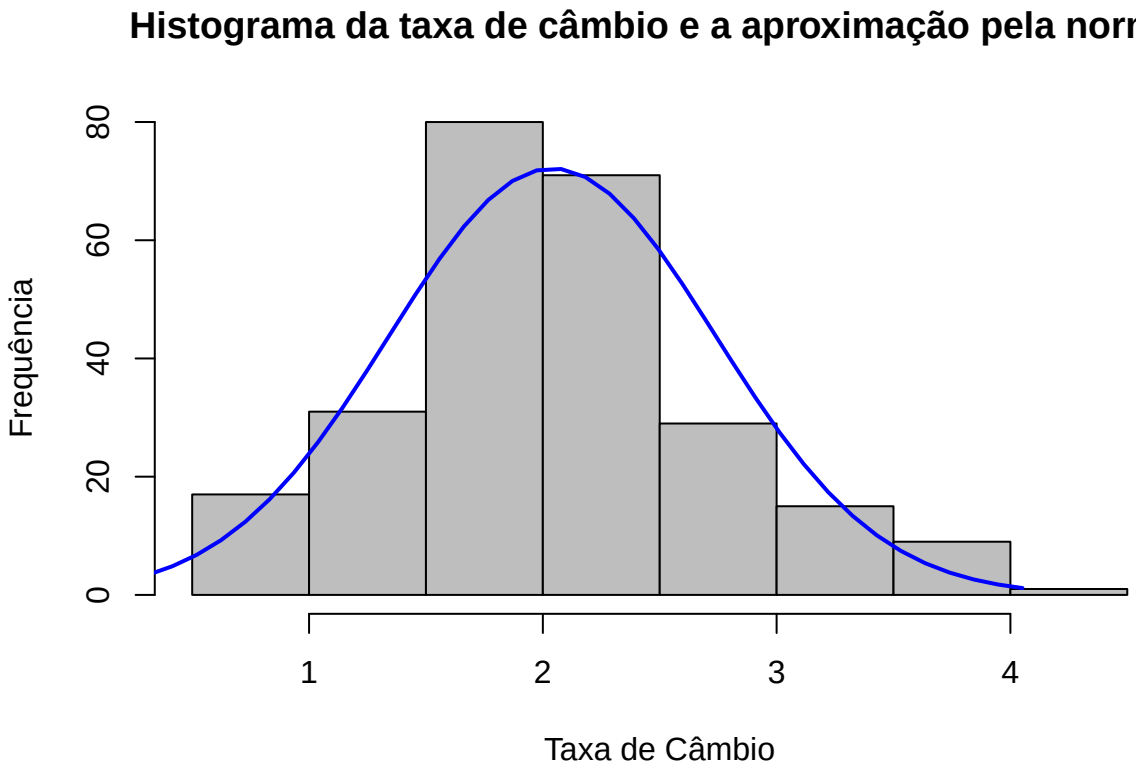
```
  hist(dolar_juros$tx_cambio, breaks=8, col="red",xlab="Taxa de Câmbio"
       , ylab="Frequência", main="Histograma da taxa de câmbio")
```



Podemos controlar o número de barras do histograma pelo comando *break=* e a cor do histograma pelo comando *col=*. Podemos plotar o histograma conjuntamente com um modelo de distribuição de probabilidade. Utilizaremos a distribuição normal com mesma esperança e desvio padrão.

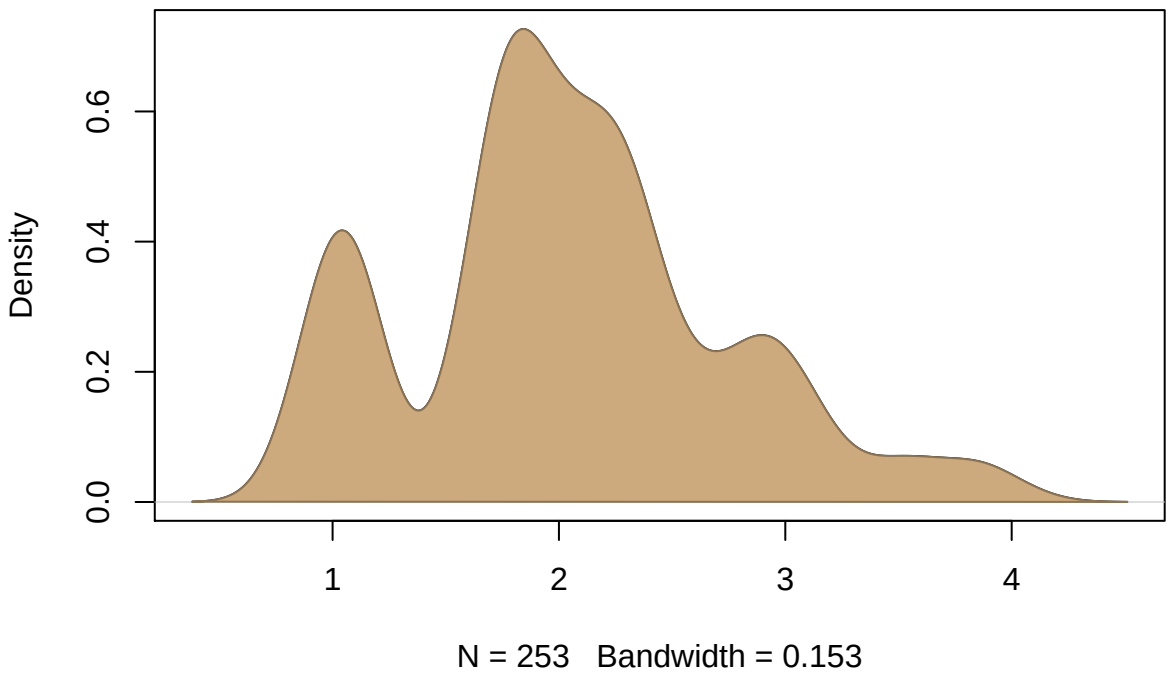
```
x <- dolar_juros$tx_cambio
h<-hist(x, breaks=10, col="gray",xlab="Taxa de Câmbio", ylab="Frequência",
        main="Histograma da taxa de câmbio e a aproximação pela normal")

xfit<-seq(0,max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```



Uma outra maneira de visualizar os dados é utilizando uma distribuição continua e não mais a discreta. Para isso, utiliza-se a densidade de Kernel para visualização da distribuição de probabilidade da taxa de câmbio. Vejamos

```
k <- density(dolar_juros$tx_cambio)
plot(k, main="Densidade de Kernel para a taxa de câmbio")
polygon(k, col="burlywood3", border="burlywood4")
```



Outra forma útil de visualizar os dados a é distribuição por classe, por exemplo distribuição de salários entre homens e mulheres, desempenho em testes de larga escala entre alunos de escola pública e privada, distribuição da taxa de câmbio no regime de âncora e fora do regime de âncora cambial. Vamos utilizar a densidade de Kernel para analisar a distribuição dos valores da taxa de câmbio na âncora cambial e após o regime. Para isso precisa instalar o pacote *sm*.

```
#install.packages("sm")
library(sm)
attach(dolar_juros)
ancora.f <- factor(dolar_juros$ancora, levels= c(0,1),
                labels = c("1 regime de âncora cambial",
                           "0 regime câmbio flutuante"))
sm.density.compare(dolar_juros$tx_cambio, dolar_juros$ancora, xlab="Taxa de Câmbio")
title(main="Comparação da taxa de câmbio entre os regimes cambias")
```

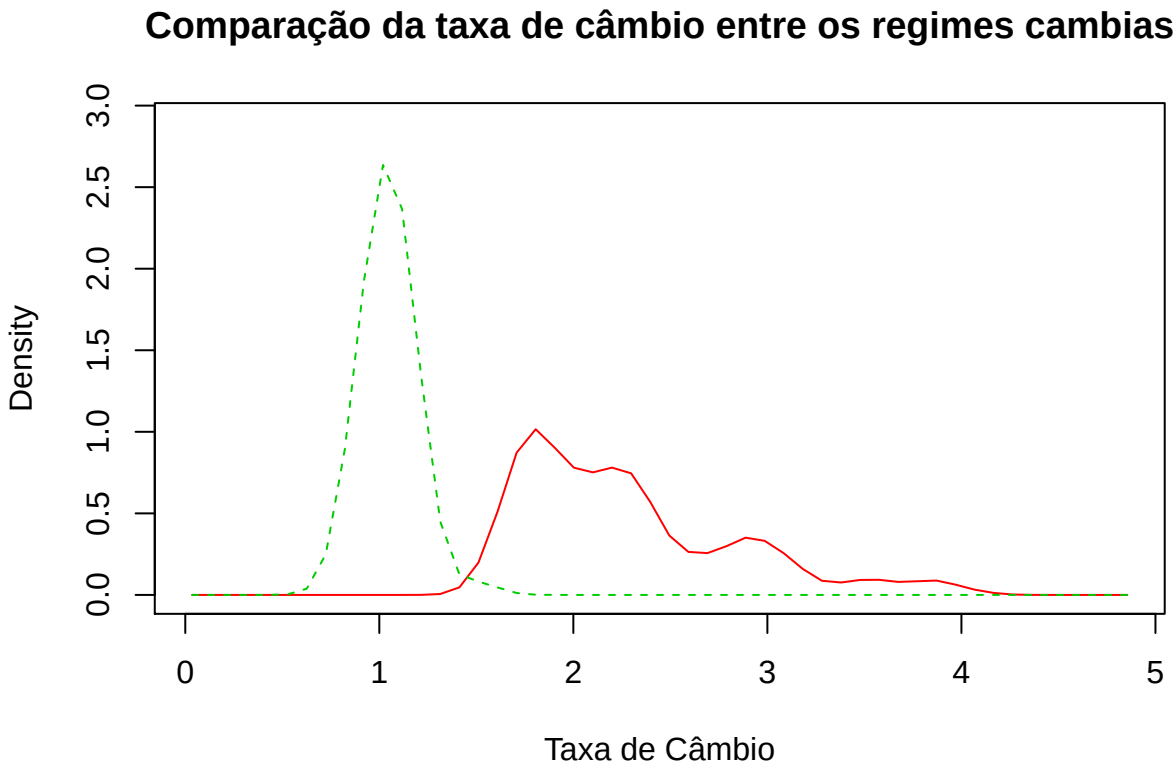
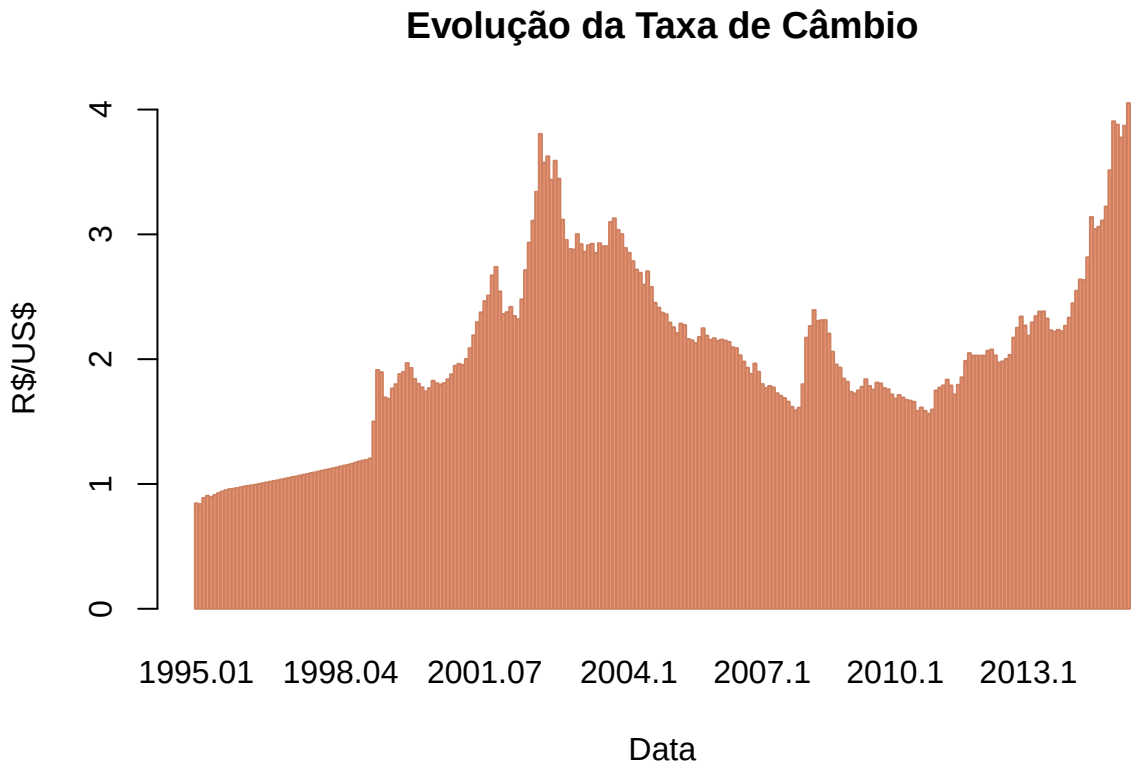


Gráfico de Barra

Em geral a utilização do gráfico de barras está relacionado ao entendimento da frequência de valores associados a uma determinada categoria. Por exemplo, imagine que temos um banco de dados com as pessoas classificadas como: i)Não trabalha; ii)Trabalha e iii)Desempregado. Teríamos três categorias e a frequência de pessoas em cada categoria. Poderíamos ainda dividir essa categoria entre homens e mulheres. Essa é a utilização mais padrão do gráfico de barras. Vamos fazer um uso menos padrão, mais muito utilizado pelas pessoas que a visualização da evolução de uma série, por exemplo a taxa de câmbio.

```
barplot(dolar_juros$tx_cambio, main="Evolução da Taxa de Câmbio",
        names.arg=dolar_juros$Data, col="lightsalmon2",border="lightsalmon3",
        xlab="Data", ylab="R$/US$")
```



Vamos supor que queiramos saber a evolução da taxa de câmbio por ano e não por mês como fizemos anteriormente. Como já havíamos incorporado a variável ano no nosso banco de dados, utilizaremos ela para montar uma tabela auxiliar para posterior elaboração do gráfico. Vejamos a tabela auxiliar:

```
bardolar <-aggregate(dolar_juros[,2:3], by=list(dolar_juros$ano),FUN=mean, na.rm=TRUE)
```

Agora fazemos o gráfico considerando as médias por ano.

```
barplot(bardolar$IPA, main="Evolução do IPA",names.arg=bardolar$Group.1, ylab="Indice"
        , xlab="Ano", col="darkblue")
```

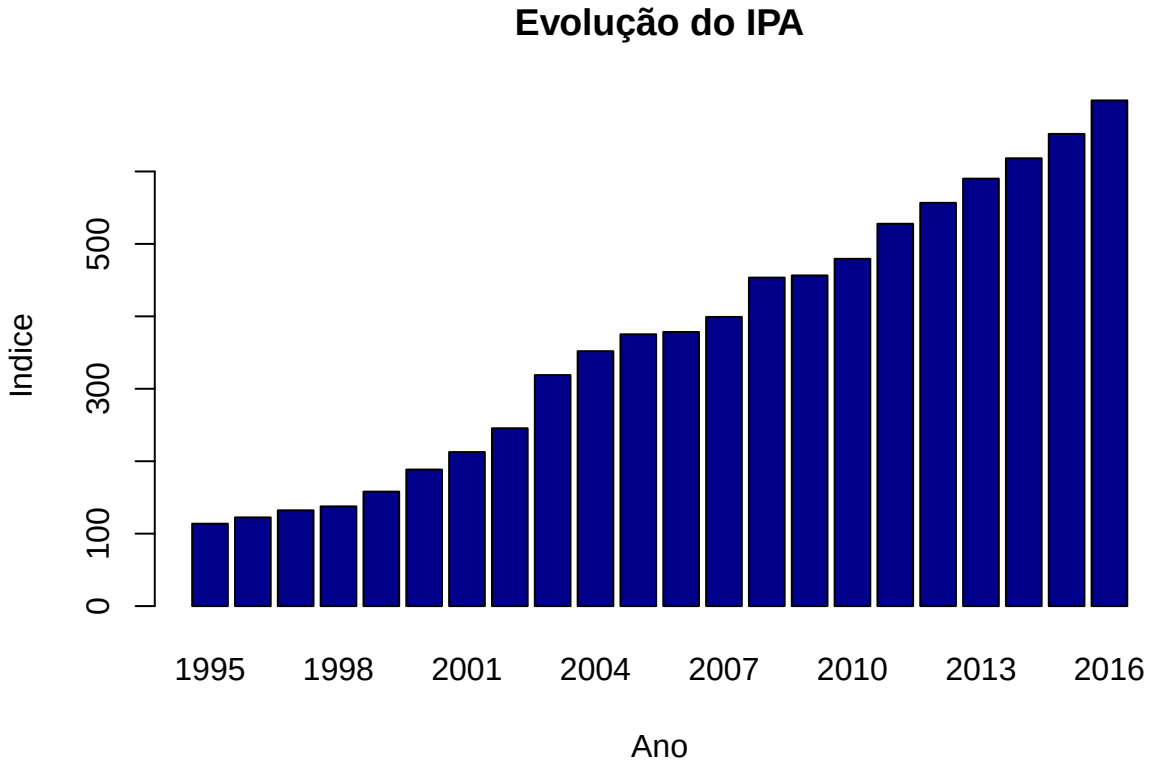


Gráfico de Linha

O gráfico de linha ou pontos é muito utilizado para visualização da evolução de séries. Vejamos alguns exemplos. Pelo fato da nossa variável que indica o mês gerar problemas de interpretação pelo R (1995.01 é janeiro de 1995, mas o R entende como um número) e isso ocasiona problema na plotagem dos gráficos, iremos criar uma variável de data utilizando essa variável presente no banco de dados. Veja abaixo. Utilizando essa variável podemos fazer a plotagem da taxa de câmbio. Alguns pontos importantes nesse gráfico são: *ylim* fornece os limites do eixo Y de 0 até 1.1x(o valor máximo), *type* é o tipo de gráfico, sendo *l* de linha, *p* de pontos, *o* de pontos e linhas.

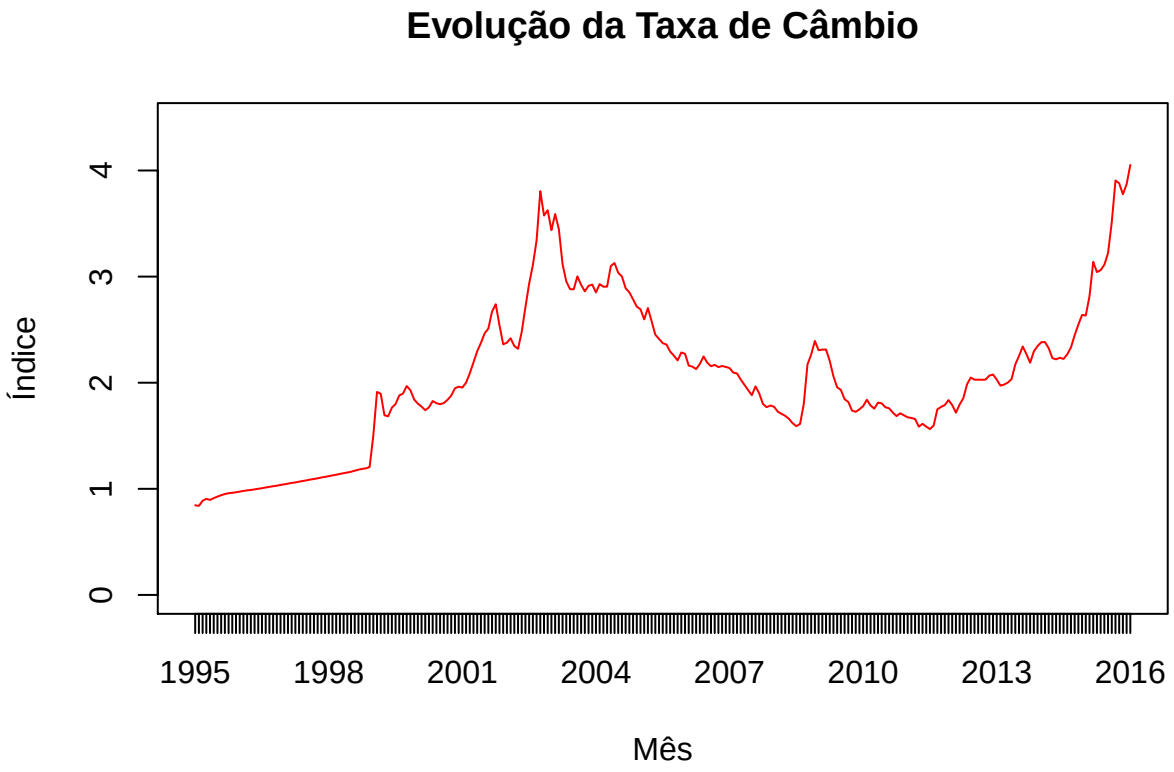
```
dolar_juros$mes=c(1:253)
library(stringr)
#CUIDADO, Data  tratada como numerico, portanto, ao converter para texto 1995.10, vira 1995.1
dolar_juros$meses<-str_replace(as.character(sprintf("%.2f",dolar_juros$Data)),pattern="[.]"
                               ,replacement="/")
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

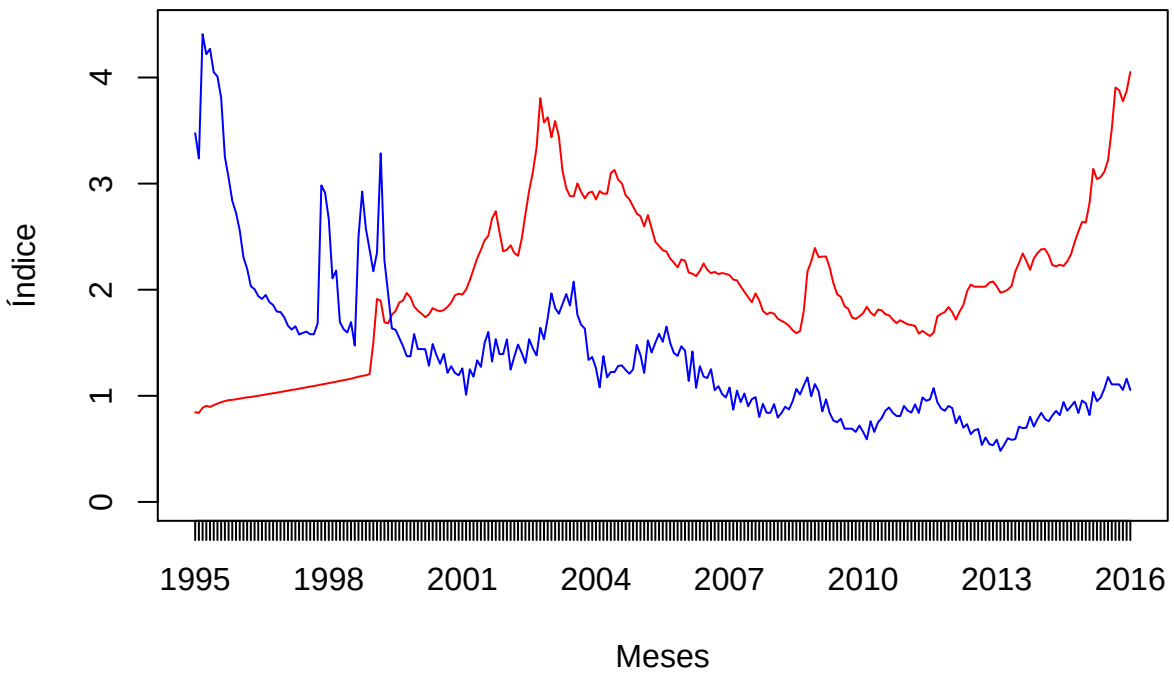
```
dolar_juros$meses <- as.yearmon(as.character(dolar_juros$meses), "%Y/%m")

plot(x=dolar_juros$meses, y=dolar_juros$tx_cambio, ylim=c(0,1.1*max(dolar_juros$tx_cambio))
     ,col="red", type="l", main="Evolução da Taxa de Câmbio", xlab="Mês", ylab="Índice")
```



Vamos agora montar um gráfico com a evolução da taxa de câmbio e juros juntos no mesmo eixo. Isso é possível pois ambas as séries possuem limites superiores e inferiores próximos. Conforme o gráfico abaixo. Note que na segunda plotagem, *points* não alteramos nenhum eixo.

```
plot(x=dolar_juros$meses, y=dolar_juros$tx_cambio, ylim=c(0,1.1*max(dolar_juros$tx_cambio))
     , col="red", type="l", main="Comparação da taxa de câmbio e juros", xlab = "Meses", ylab="Índice")
points(x=dolar_juros$meses, y=dolar_juros$Juros, col="blue", type="l")
```



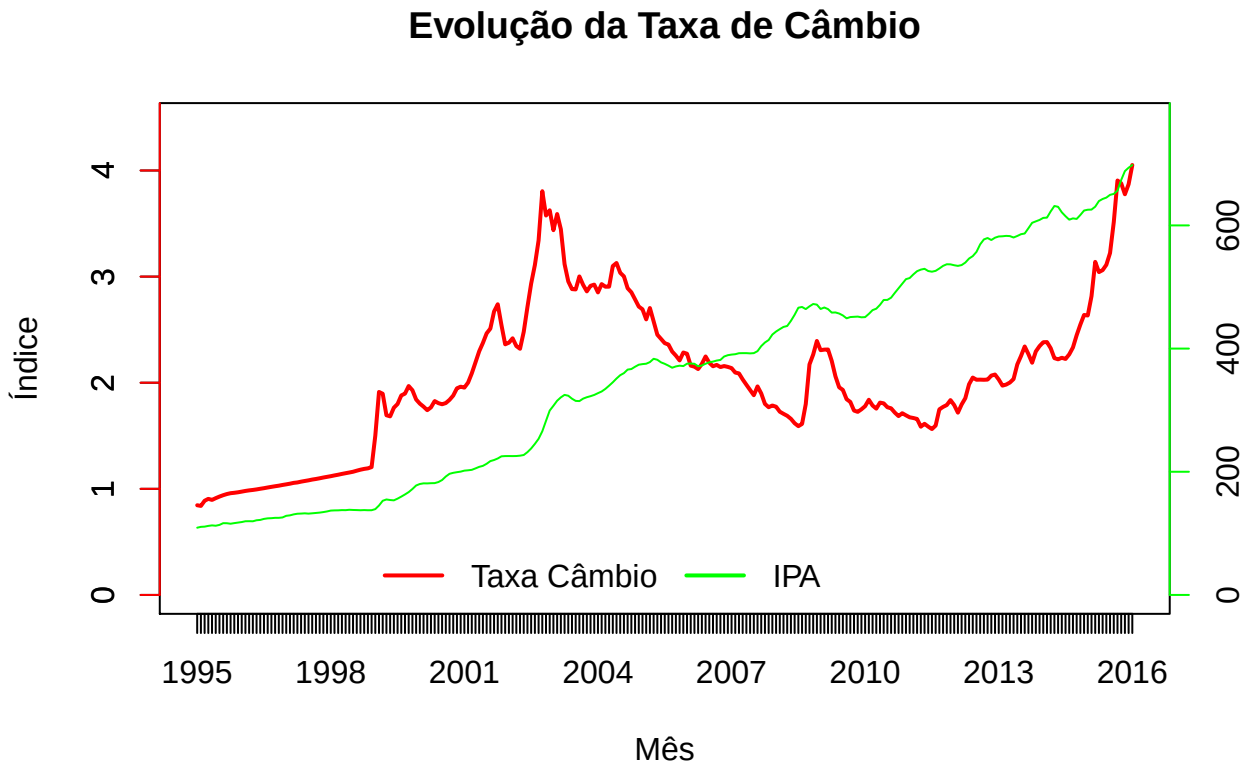
Se quiséssemos plotar a taxa de câmbio e o IPA, teríamos que adotar dois eixos, pois os limites superiores e inferiores de cada série são distintos. O gráfico abaixo mostra essa plotagem. Vejamos alguns detalhes. Primeiramente você deve plotar o primeiro gráfico com o comando *plot()*. O comando *axis (2,...)* pinta o eixo 2 que é o da esquerda, 3 é o superior, 4 o da direita e 1 o inferior. Depois utilize o comando *par(new=T)* que indica que deve sobrescrever os gráficos e por fim faça o plot do segundo sem mexer nos eixos, por isso utilize os comando *xaxt="n"* sem legenda de eixo e *axes=F* sem eixo. Algumas explicações: lwd= é o tamanho da linha, xpd = TRUE permite plotar fora da região do gráfico, horiz = TRUE diz que é para fazer uma legenda horizontal, inset = c(x,y) para mover a legenda em relação a localização inferior, bty = 'n' sem caixa, pch e col são o tipo e a cor dos pontos e cex= 2 faz a legenda duas vezes o tamanho da fonte. Para mudar as cores das linhas e gráficos veja *http://www.stat.columbia.edu/ tzheng/files/Rcolor.pdf*

```
plot(x=dolar_juros$meses, y=dolar_juros$tx_cambio, ylim=c(0,1.1*max(dolar_juros$tx_cambio))
     ,col="red", type="l", main="Evolução da Taxa de Câmbio", xlab="Mês", ylab="Índice"
     ,lwd=2.00)
axis(2, pretty(c(0, 1.1*max(dolar_juros$tx_cambio))), col='red')

par(new=T)

plot(x=dolar_juros$meses, y=dolar_juros$IPA, ylim=c(0,1.1*max(dolar_juros$IPA)), col="green", type="l",
axis(4, pretty(c(0, 1.1*max(dolar_juros$IPA))), col='green')

legend("bottom", c("Taxa Câmbio", "IPA"), xpd = TRUE, horiz = TRUE
       , inset = c(50,0), bty = "n",  col=c('red', 'green'), lwd=c(2, 2), cex = 1)
```



Podemos utilizar pontos ao invés de linhas e para isso troque o tipo de gráfico por *type="p"* e ai você terá que escolher o formato do ponto e seu tamanho. O comando *pch* escolhe o tipo de ponto e *cex* o seu tamanho. Veja os símbolos em *http://www.endmemo.com/program/R/pchsymbols.php*. Outros parâmetros gráficos como tipo de linha, tamanho de fonte, margem do gráfico, entre outros podem ser encontrados em *http://www.statmethods.net/advgraphs/parameters.html*

```
plot(x=dolar_juros$meses, y=dolar_juros$tx_cambio, ylim=c(0,1.1*max(dolar_juros$tx_cambio))
     ,col="red", type="p", main="Evolução da Taxa de Câmbio", xlab="Mês",
     ylab="Índice",pch=20 ,cex=0.30)
axis(2, pretty(c(0, 1.1*max(dolar_juros$tx_cambio))), col='red')

par(new=T)
plot(x=dolar_juros$meses, y=dolar_juros$IPA, ylim=c(0,1.1*max(dolar_juros$IPA)), col="green", type="p",
axis(4, pretty(c(0, 1.1*max(dolar_juros$IPA))), col='green')

legend("bottom", c("Taxa Câmbio", "IPA"), xpd = TRUE, horiz = TRUE, inset = c(50,
                                                                               0), bty = "n",  col=c('re
```

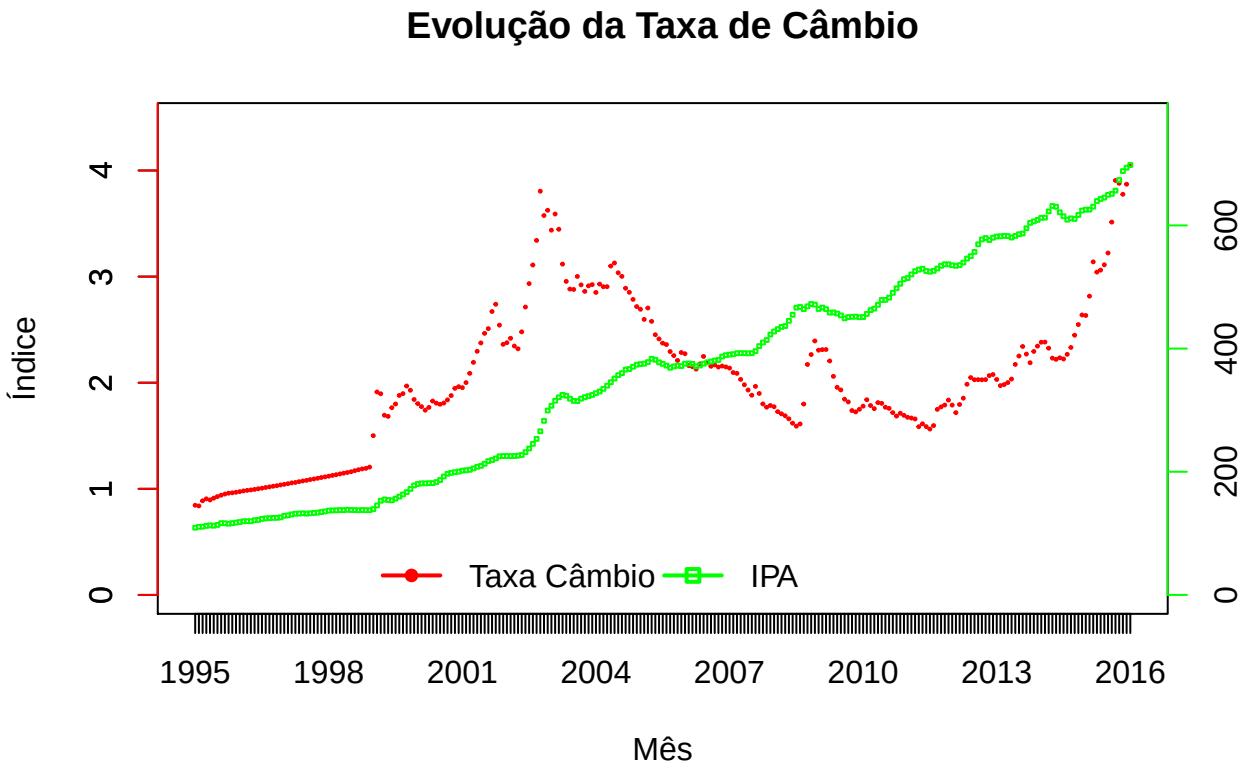
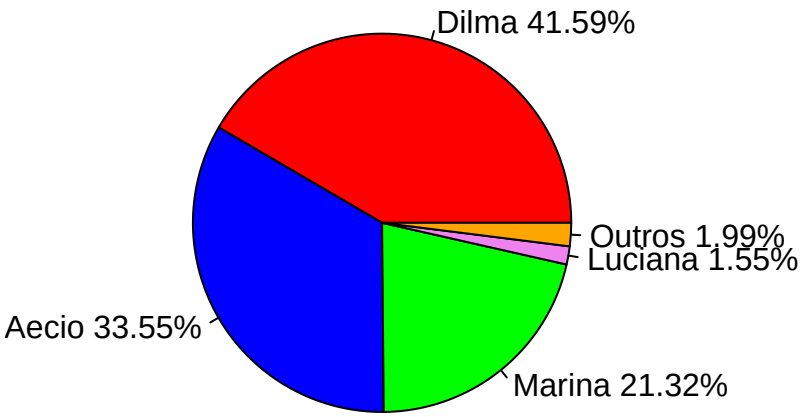


Gráfico de Pizza

O gráfico de pizza indica as proporções de uma determinada variável de classe. Infelizmente nosso banco de dados não possui essa variável, mas tomemos o exemplo das eleições de 2014. Temos uma variável de classe que são candidatos e os percentuais que cada um obteve no primeiro turno. Assim temos Dilma com 41.59%, Aécio com 33.55% e assim por diante. Veja o gráfico de pizza abaixo. Estamos interessados que o percentual de cada candidato apareça e portanto criamos um label (*lbls*).

```
colors = c("red", "blue", "green", "violet", "orange")
slices <- c(41.59, 33.55, 21.32, 1.55, 1.99)
lbls <- c("Dilma", "Aecio", "Marina", "Luciana", "Outros")
pct <- (slices/sum(slices)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie(slices,labels = lbls, col=colors, main="Resultado das Eleições 2014 - 1o Turno")
```



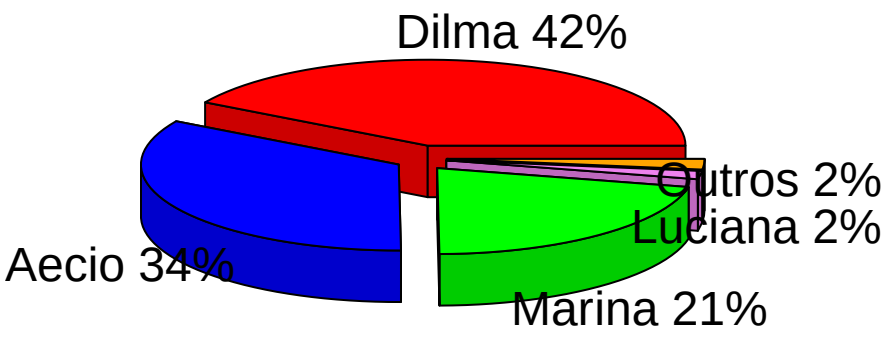
Uma alternativa que deve ser utilizada com muita cautela são os gráficos 3D. Vejamos em exemplo de gráfico de pizza 3D. É preciso instalar o pacote *plotrix*.

```
library(plotrix)
```

```
##
## Attaching package: 'plotrix'
```

```
## The following object is masked from 'package:psych':
##
##     rescale
```

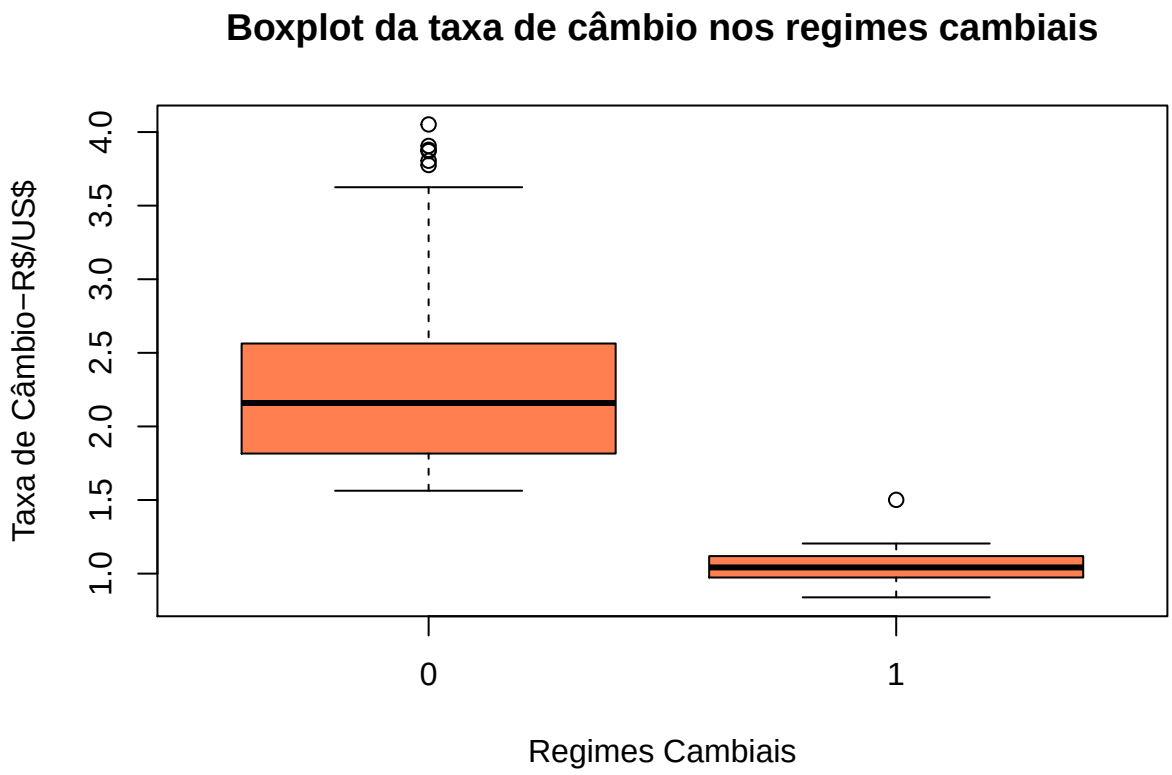
```
colors = c("red", "blue", "green", "violet", "orange")
slices <- c(41.59, 33.55, 21.32, 1.55, 1.99)
lbls <- c("Dilma", "Aecio", "Marina", "Luciana", "Outros")
pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie3D(slices,labels=lbls,explode=0.1,
      main="Resultado das Eleições 2014 - 1o Turno", col=colors)
```



Boxplot

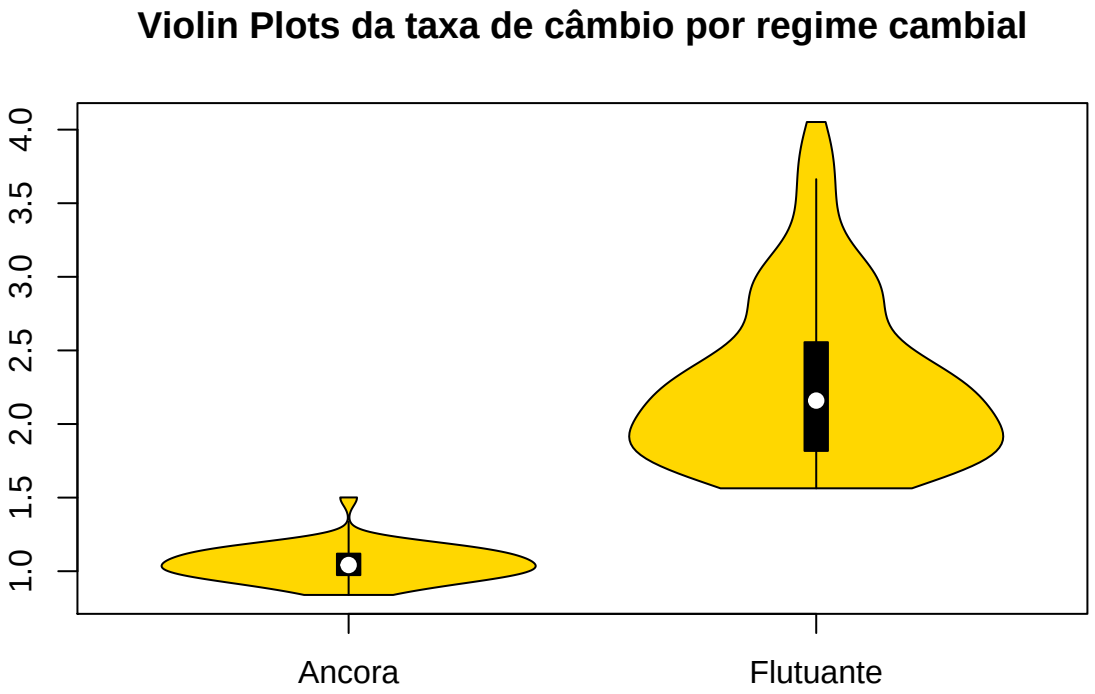
O boxplot é um gráfico que traz muitas informações e pode ser visto como a distribuição de probabilidade dos dados. O box contém 50% dos dados. O limite superior indica o percentil de 75% ($Q3$) e o limite inferior indica o percentil de 25% ($Q1$). A linha que corta o box indica a mediana, ou seja, $Q2$. Os bigodes são calculados com base na distância interquantílica, ou seja, $max\{min(dados); Limite\ inferior:\ Q1\text{-}1{,}5(Q3\text{-}Q1)\}$; e $min\{max(dados); Limite\ superior: Q3+1{,}5(Q3\text{-}Q1)\}$. Dados fora desses limites são classificados como suspeitos de serem outlier. Podemos observar a assimetria dos dados quando a mediana não está no meio da caixa, indicando maior densidade na menor distância entre os quartis Q1 ou Q3 e a mediana Q2. Vejamos agora o boxplot da taxa de câmbio na âncora e fora da âncora cambial.

```
boxplot(tx_cambio~ancora,data=dolar_juros, main="Boxplot da taxa de câmbio nos regimes cambiais",
          xlab="Regimes Cambiais", ylab="Taxa de Câmbio-R$/US$", col="coral")
```



O Violin Plot é muito parecido com o BoxPlot mas com a densidade de kernel rotacionada em cada um dos lados. Assim, indica a distribuição dos dados em cada ponto e vem anotado a mediana na forma de um ponto ou marca e um pequeno boxplot no centro do violin plot.

```
library(vioplot)
x1 <- dolar_juros$tx_cambio[dolar_juros$ancora==1]
x2 <- dolar_juros$tx_cambio[dolar_juros$ancora==0]
vioplot(x1, x2, names=c("Ancora", "Flutuante"), col="gold")
title("Violin Plots da taxa de câmbio por regime cambial")
```

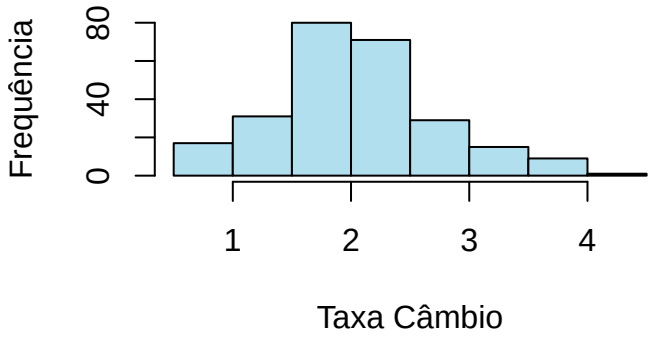


Aqui segue uma sugestão de plotar o histograma juntamente com o boxplot, possibilitando em um mesmo gráfico uma maior quantidade de observação. Veja abaixo:

```
par(fig=c(0,0.6,0,0.6), new=TRUE)
```

```
## Warning in par(fig = c(0, 0.6, 0, 0.6), new = TRUE): chamada de
## par(new=TRUE) sem plot
```

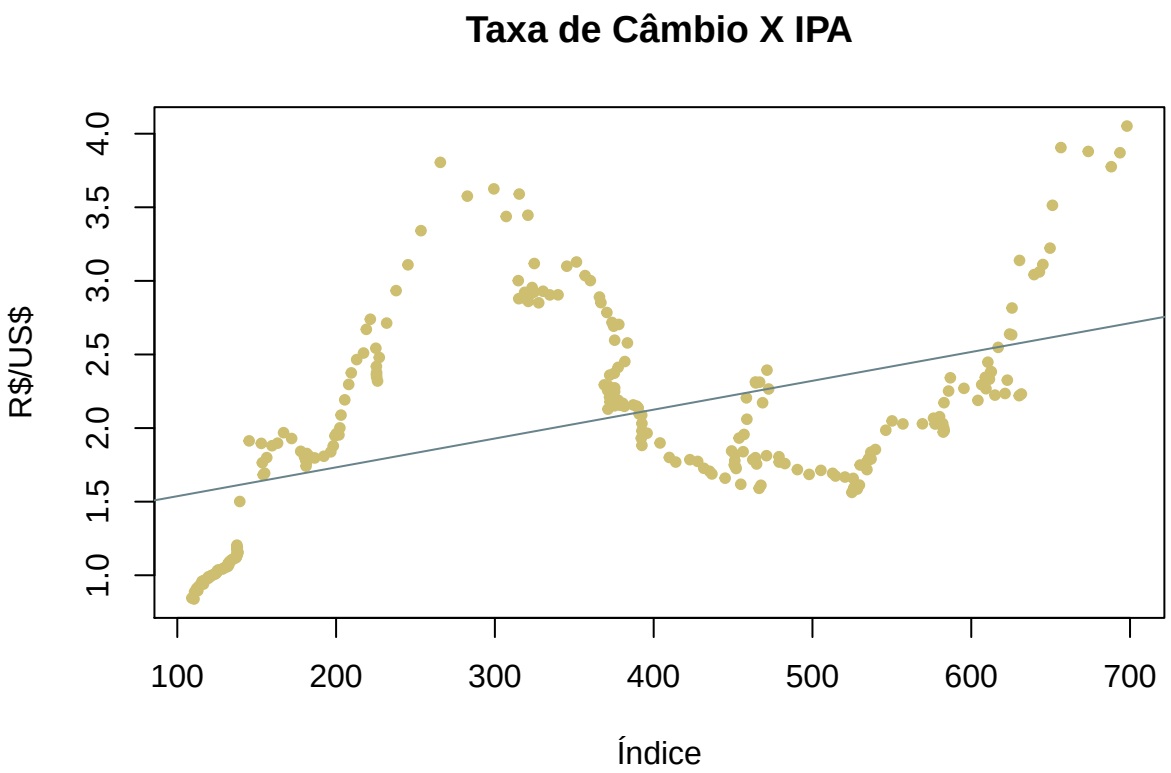
```
hist(dolar_juros$tx_cambio, main="", xlab="Taxa Câmbio", ylab="Frequência", col="lightblue2")
par(fig=c(0,0.6,0.3,0.8), new=TRUE)
boxplot(dolar_juros$tx_cambio, horizontal=TRUE, axes=FALSE, col="navajowhite2")
```



Scatter plot

O Scatter plot é conhecido como o gráfico de dispersão. Ele relaciona duas ou três variáveis, ou seja, plota X contra Y. Muito utilizado para ver o comportamento conjunto de duas séries. Vejamos a taxa de câmbio e o IPA.

```
plot(dolar_juros$IPA, dolar_juros$tx_cambio, main="Taxa de Câmbio X IPA",xlab="Índice", ylab="R$/US$", p
abline(lm(dolar_juros$tx_cambio~dolar_juros$IPA), col="lightblue4")
```



Podemos fazer o gráfico de dispersão considerando o momento de regime de âncora cambial e após o regime de âncora cambial. Para isso, temos que utilizar o pacote *car*. Foi desativado o comando que plota a regressão *reg=F* e a legenda aparecerá no canto superior esquerdo.

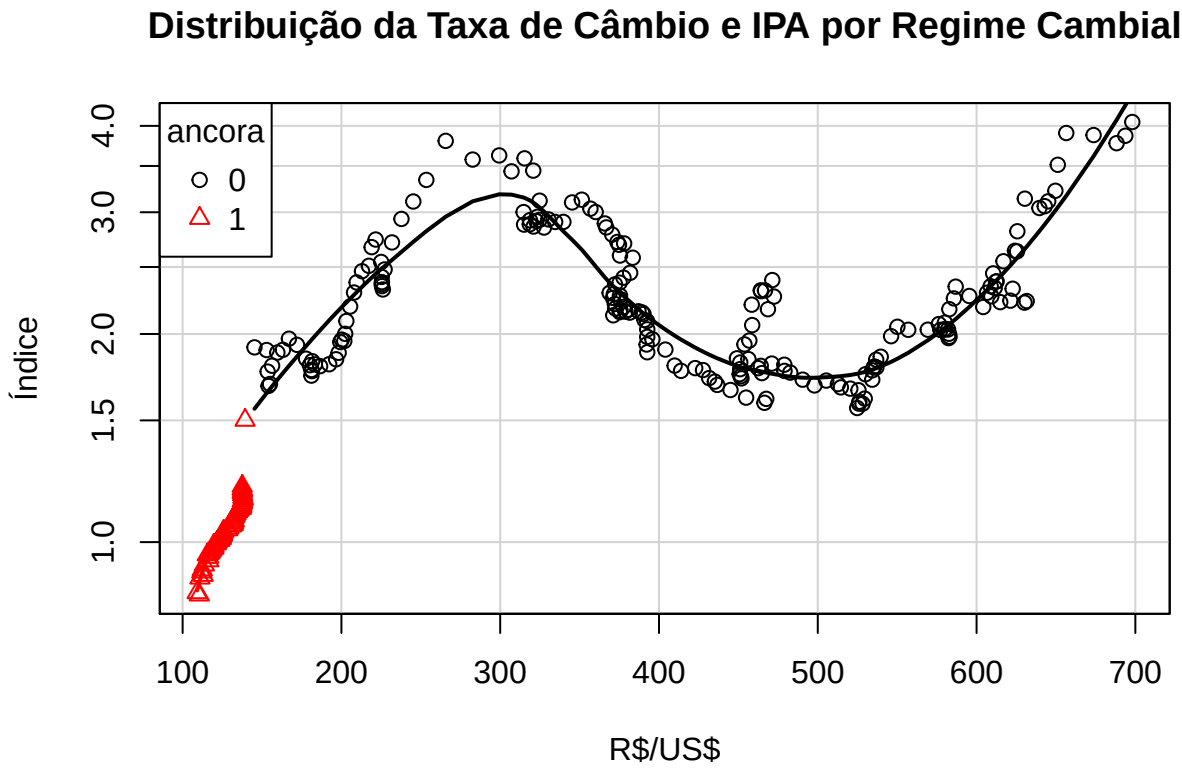
```
library(car)
```

```
## Warning: package 'car' was built under R version 3.2.4
```

```
##
## Attaching package: 'car'
```

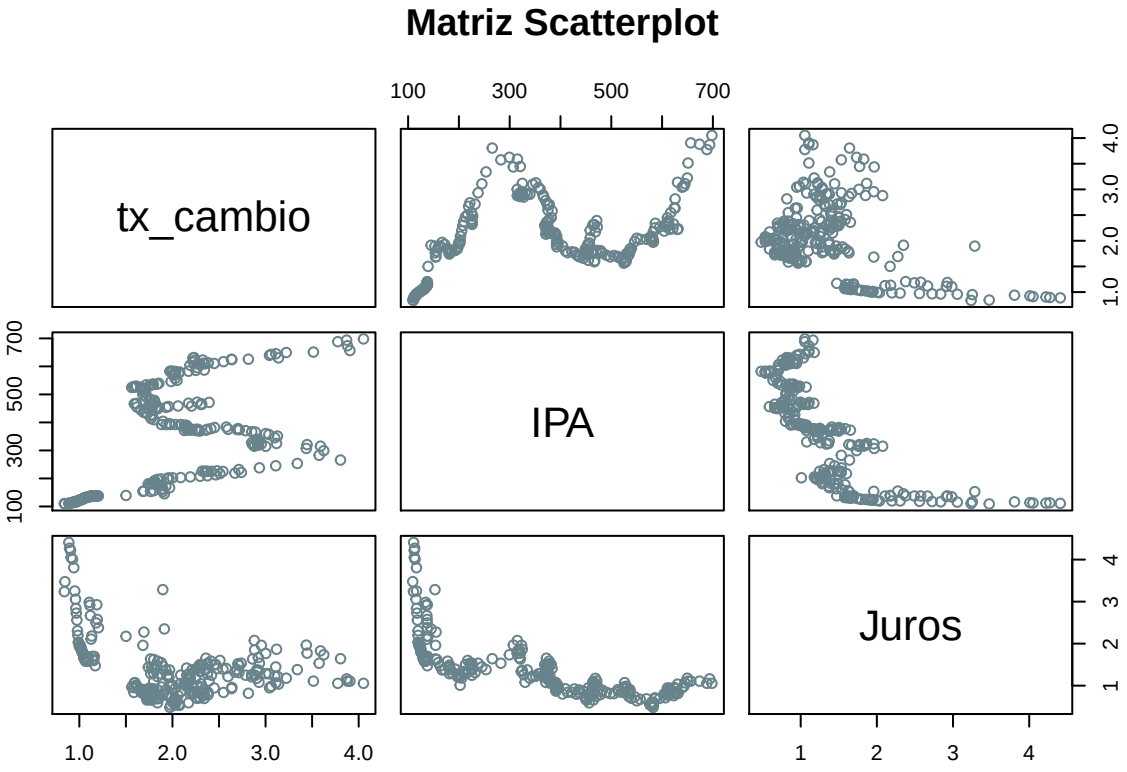
```
## The following object is masked from 'package:psych':
##
##     logit
```

```
scatterplot(tx_cambio ~ IPA | ancora, data=dolar_juros,
            xlab="R$/US$", ylab="Índice",
            main="Distribuição da Taxa de Câmbio e IPA por Regime Cambial",
            reg = F, log = "y",  legend.coords="topleft",
            labels=row.names(dolar_juros) )
```



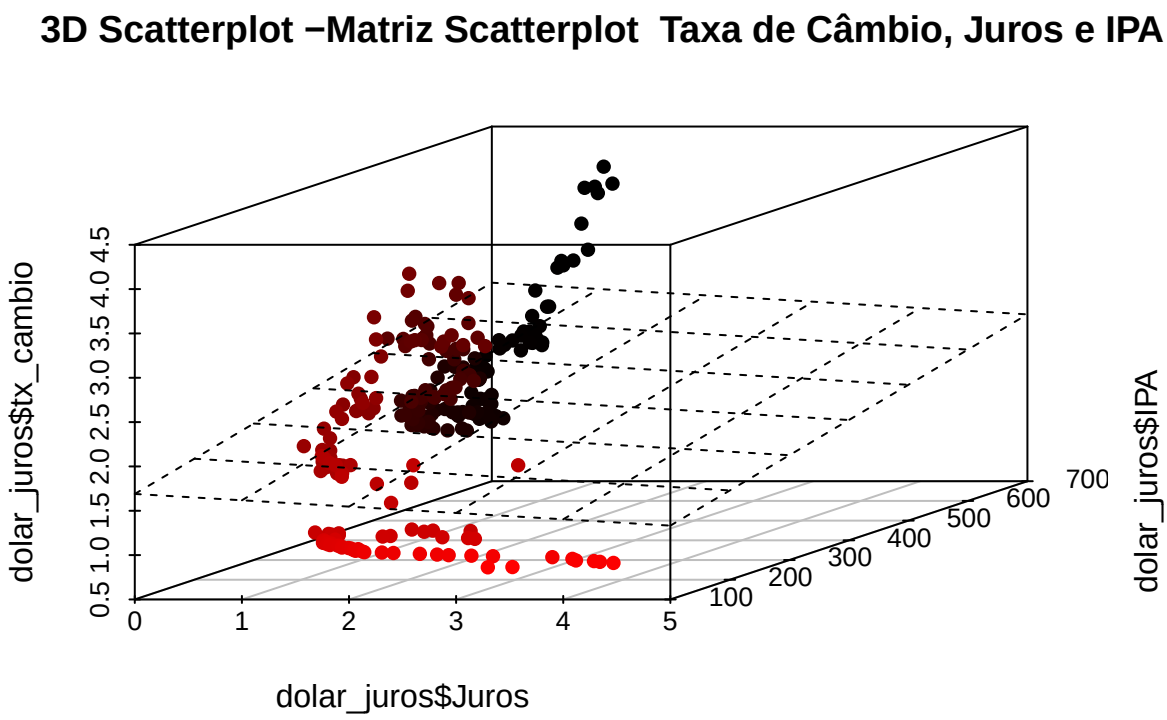
Outra maneira mais concisa de apresentar a dispersão conjunta dos dados é utilizar uma matriz de dispersão, ou seja, elabora vários gráficos de dispersão como apresentado abaixo.

```
pairs(~tx_cambio+IPA+Juros,data=dolar_juros,
        main="Matriz Scatterplot", col="lightblue4")
```



Uma maneira menos usual mas interessante, são os gráficos 3D. Novamente muito cuidado no seu uso e evite em textos acadêmicos a não ser que julgue muito necessário. Pode utilizar o pacote *scatterplot3d*. Vamos utilizar *type=p* para plotar pontos (alternativa é utilizar *type=h* que são splines) e vamos plotar o plano de regressão com o comando *fit* .

```
library(scatterplot3d)
s3d <-scatterplot3d(dolar_juros$Juros,dolar_juros$IPA,dolar_juros$tx_cambio, pch=16, highlight.3d=TRUE,
fit <- lm(dolar_juros$tx_cambio ~ dolar_juros$Juros+dolar_juros$IPA)
s3d$plane3d(fit)
```



Gráficos com GGPLOT

O pacote ggplot (e agora sua versão mais nova ggplot2) vem se consolidando como um dos pacotes mais usados para geração de gráficos. Ele tem algumas vantagens como:

- Bastante flexível
- Poderoso
- Muito personalizável
- Bonito!

Mas algumas coisas não deveriam ser feitas com ggplot, pois existem pacotes mais especializados:

- Gráficos 3D (pacote rgl)
- Grafos (pacote igraph)
- Gáficos interativos (pacote ggvis)

A gramática dos gráficos

Para criarmos gráficos com ggplot, utilizamos **blocos** de sintaxe que vão sendo empilhados para criar o gráfico pretendido

Os blocos com os quais podemos trabalhar são:

- **Dados**
- **Mapeamento estético**
- Objetos geométricos
- Transformações estatíticas
- Escalas
- Sistema de coordenadas
- Ajustes de posição

Histograma

```
library(ggplot2)
```

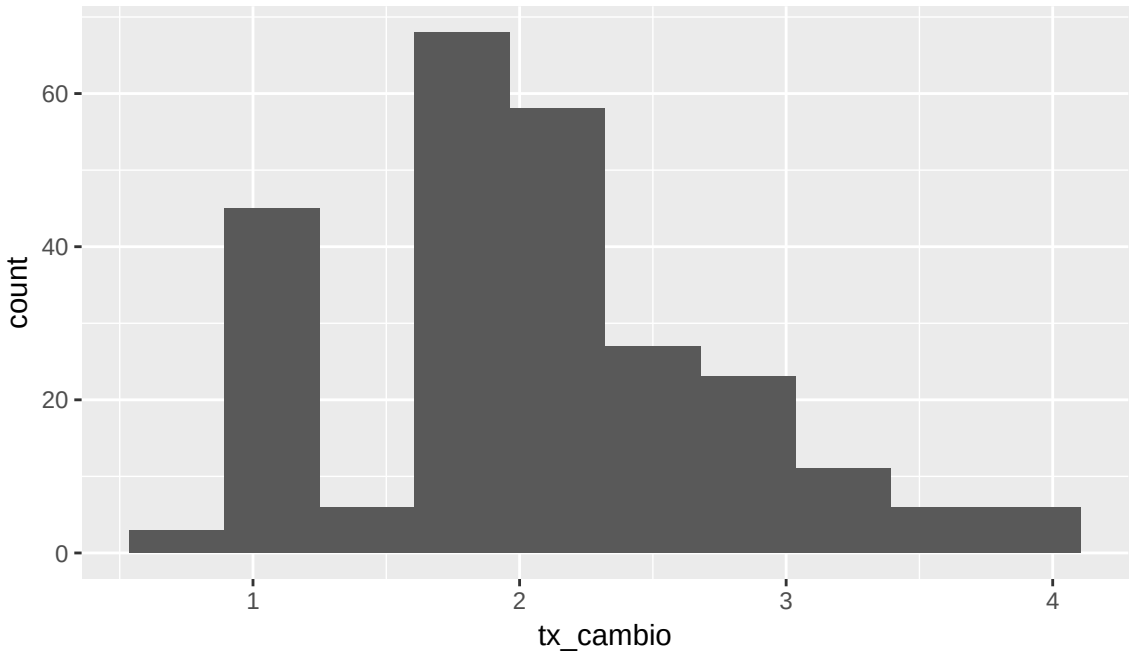
```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
## Warning in FUN(X[[i]], ...): failed to assign NativeSymbolInfo for env
## since env is already defined in the 'lazyeval' namespace
```

```
##
## Attaching package: 'ggplot2'
```

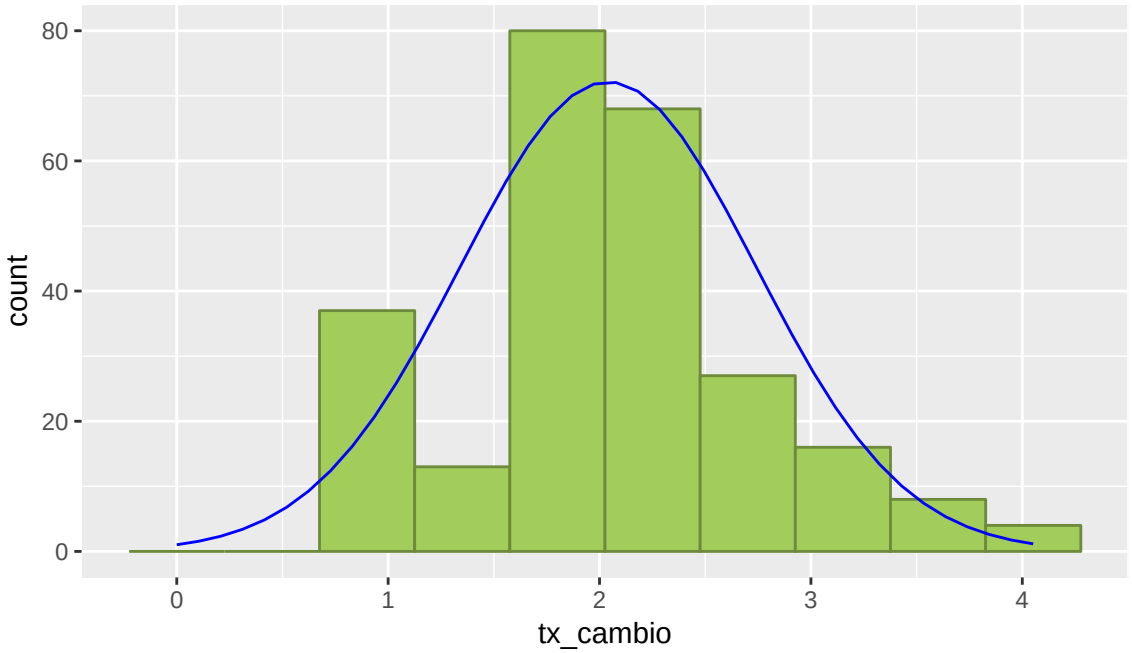
```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
ggplot(data=dolar_juros, aes(x = tx_cambio)) + geom_histogram(bins=10)
```




```
xfit<-seq(0,max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
dFit <- data.frame(x=xfit,y=yfit)

ggplot(data=dolar_juros, aes(x = tx_cambio)) + geom_histogram(bins=10, col="darkolivegreen4",fill=I("da
  geom_line(data=dFit,aes(x=x,y=y),col='blue')
```



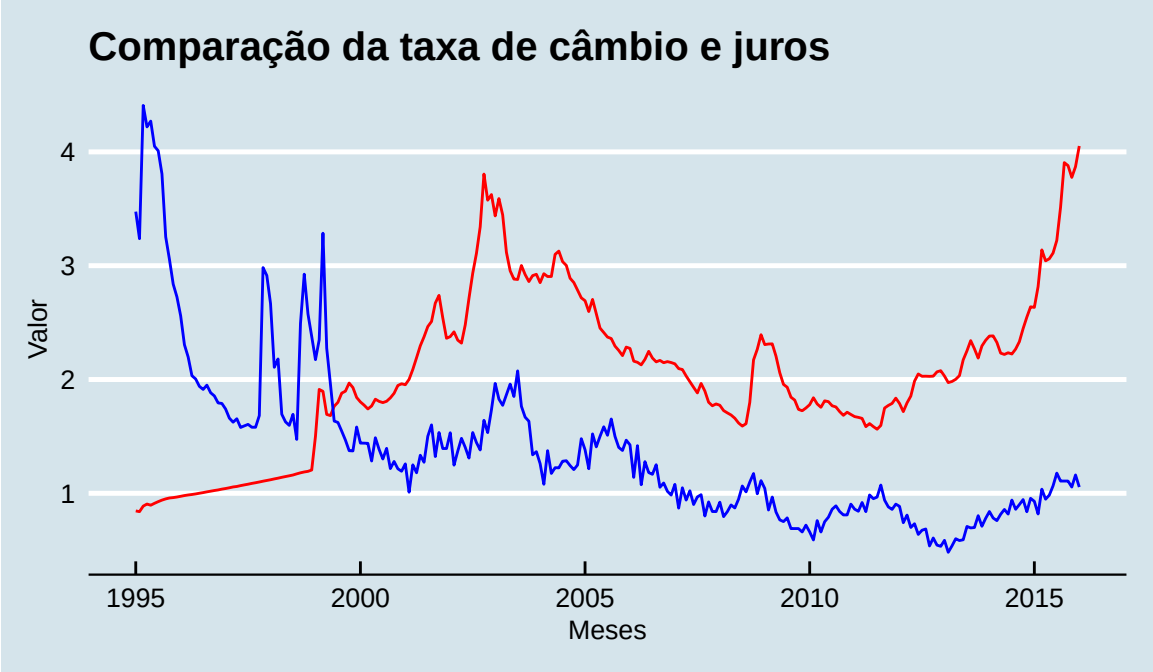
Gráficos de linha

```
library(ggthemes)
g <-  ggplot(data=dolar_juros)+
    geom_line(aes(x=meses,y=tx_cambio),col='red')+
    geom_line(aes(x=meses,y=Juros),col='blue')+
    labs(x='Meses',y='Valor',title='Comparação da taxa de câmbio e juros')
g+theme_economist()
```

```
## Warning: `panel.margin` is deprecated. Please use `panel.spacing` property
## instead
```

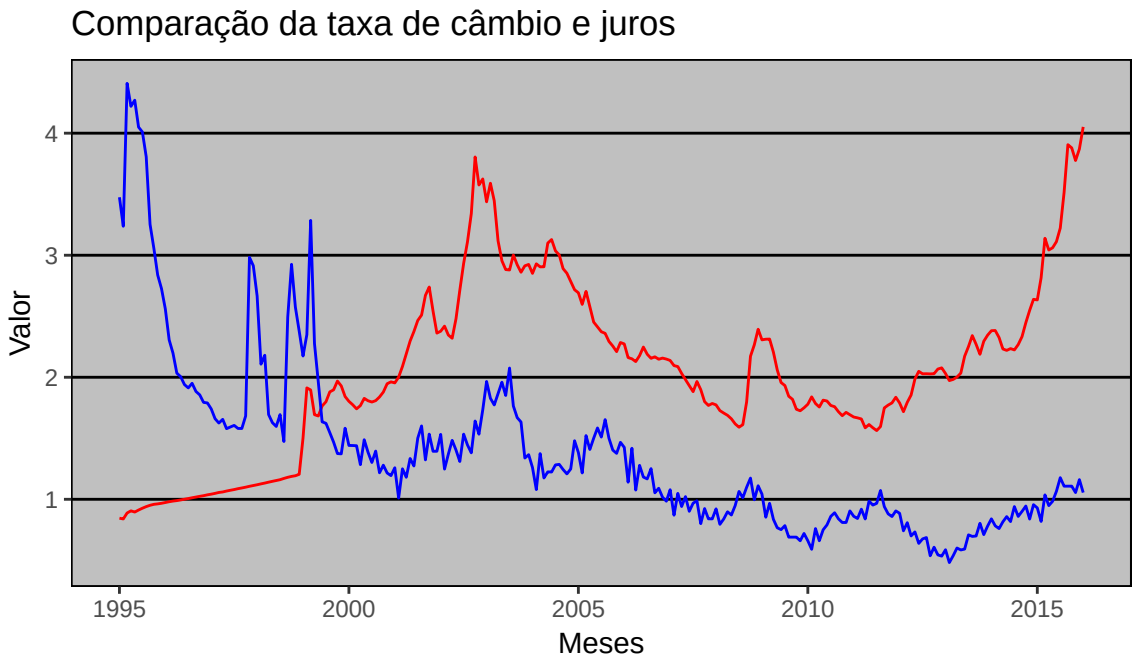
```
## Warning: `legend.margin` must be specified using `margin()`. For the old
## behavior use legend.spacing
```

```
## Don't know how to automatically pick scale for object of type yearmon. Defaulting to continuous.
```



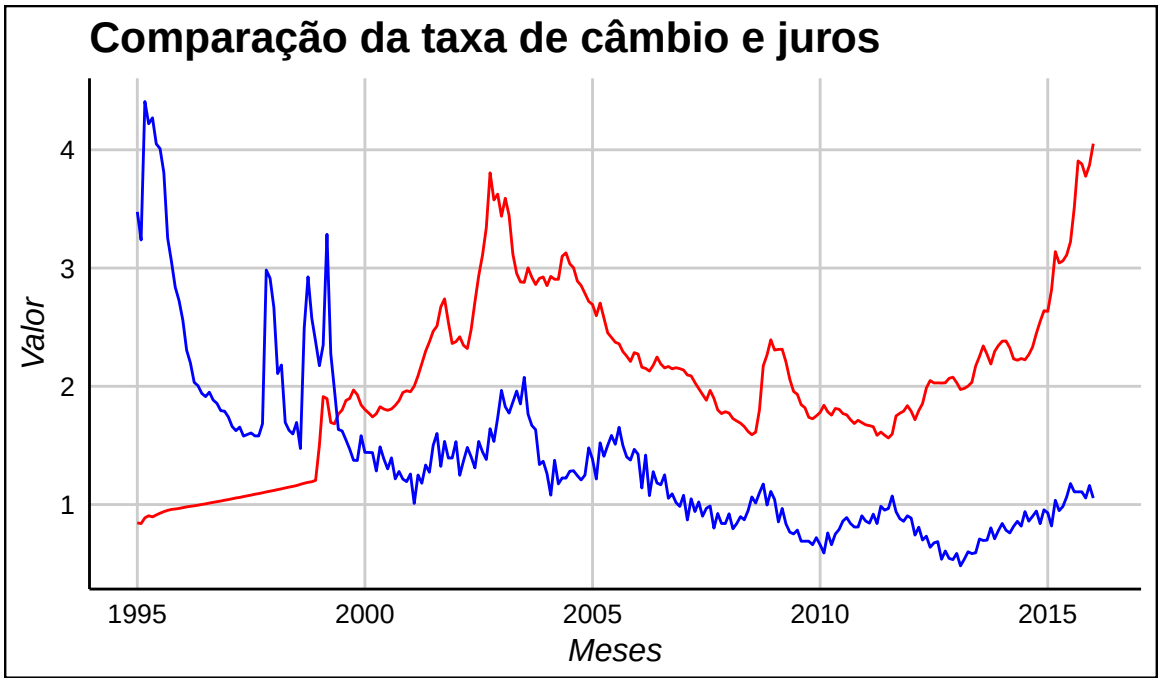

```
  g+theme_excel()
```

```
## Don't know how to automatically pick scale for object of type yearmon. Defaulting to continuous.
```




```
g+theme_gdocs()
```

```
## Don't know how to automatically pick scale for object of type yearmon. Defaulting to continuous.
```

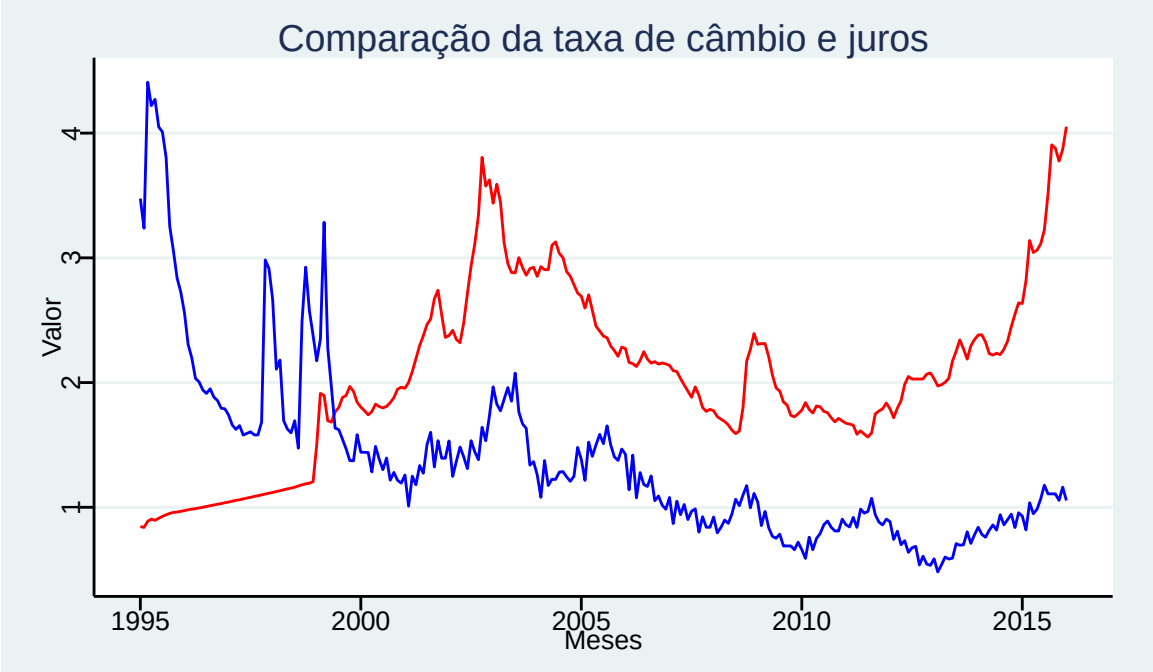



```
g+theme_stata()
```

```
## Warning: `panel.margin` is deprecated. Please use `panel.spacing` property
## instead
```

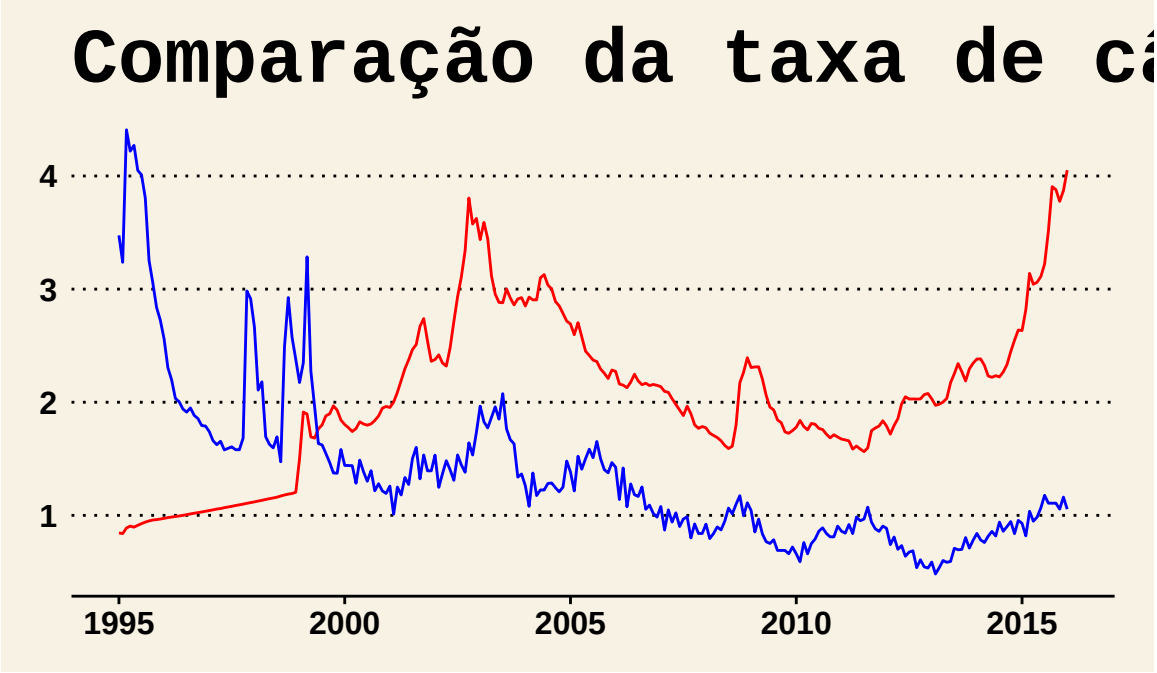
```
## Warning: `legend.margin` must be specified using `margin()`. For the old
## behavior use legend.spacing
```

```
## Don't know how to automatically pick scale for object of type yearmon. Defaulting to continuous.
```




```
g+theme_wsj()
```

```
## Don't know how to automatically pick scale for object of type yearmon. Defaulting to continuous.
```



Utilizando a Pesquisa por Amostra de Domicílios- PNAD

A PNAD

A Pesquisa Nacional por Amostra de Domicílio – PNAD - foi implantado no Brasil a partir de 1967, com a finalidade de produzir informações básicas para o estudo do desenvolvimento socioeconômico do País. Abrange a população residente nas unidades familiares, dividindo-se em duas partes: i) pesquisa de caráter permanente, que são as características gerais da população, tais como educação, trabalho, rendimento e habitação e, ii) pesquisa de caráter variável, tais como migração, fecundidade, nupcialidade, saúde, nutrição entre outros.

A pesquisa é feita anualmente, excetuando os anos de censo e é realizada no último trimestre do ano. Quanto à abrangência, a pesquisa é dividida em 5 grandes regiões sendo elas: Sul, Sudeste, Centro-oeste, Nordeste e Norte, onde esta última, dependendo do ano, refere-se somente a parcela urbana - até 2003 - (exceção para o Tocantins). A pesquisa de 2014 totalizou 362.627 pessoas pesquisadas nessas regiões e 151.291 domicílios. A PNAD é realizada através de amostras probabilísticas de domicílios obtidas em três estágios de seleção: unidades primárias (municípios); unidades secundárias (setores censitários); e unidades terciárias (unidades domiciliares). O processo de expansão da amostra utiliza estimadores de razão cuja variável independente é a projeção da população residente, segundo tipo de área (região metropolitana e não metropolitana).

Documentação Fornecida pelo IBGE

O IBGE fornece a documentação e os dados para abrir a PNAD. Os documentos que o IBGE fornecem estão listados abaixo e podem ser obtidos para o ano de 2014 no endereço: http://www.ibge.gov.br/home/estatistica/populacao/ trabalhoerendimento/ pnad2014/microdados.shtm

Dados

Na pasta Dados será disponibilizada os microdados da pesquisa separado por dados das pessoas e dados do domicílio. Eles são fornecidos em formato txt e separado por dimensão da variável.

Dicionários e inputs

Nesta pasta você encontra o dicionário de variáveis das pessoas e dos domicílios. Esse dicionário contêm a posição da variável, seu tamanho, seu código, sua descrição, tipo ou valores e a categoria desses valores. Esse dicionário é parte muito importante para compreensão do banco de dados.

Os inputs são os arquivos em formato sas (Programa Estatístico) para realizar a leitura do banco de dados. Um dicionário de leitura do banco. Não iremos utilizar.

Questionários

Aqui são apresentados os dois questionários efetivamente aplicados em campo, o de pessoas e o de domicílios. São muito importante pois mostram a sequência de aplicação das perguntas. Com o questionário somos capazes de analisar o encadeamento das perguntas ou a sequência de perguntas. Também é parte importante na elaboração do nossa análise.

Metodologia

Na metodologia é apresentada diversos arquivos que ajudam a elaboração de cruzamentos mais específicos ou definições mais precisas. Temos por exemplo os grupamentos de ocupação/atividade, relação de códigos de ocupação/atividade, unidades e notas metodológicas. Vale a pena analisar as notas metodológicas.

Leitura em R

O IBGE disponibiliza dois arquivos importantes para leitura da PNAD. O primeiro é o *dicPNAD2014.RData* que é o dicionário de leitura das variáveis de 2014. O segundo é o arquivo que, com base no dicionário, consegue ler os dados. Esse se chama *IBGEPesq.RData* que contem o pacote *le.pesquisa*. Infelizmente, o arquivo *IBGEPesq.RData* não funciona para versões mais novas do R. Entretanto, fizemos modificações no pacote *le.pesquisa*, que possibilitou o uso em versões mais atuais. Esse pode ser baixado no seguinte site

Lendo o Banco de Dados

Primeirmente é preciso carregar os dois arquivos importantes o *dicPNAD2014.RData*, nosso dicionário de 2014 e o arquivo corrigido *IBGEPesq.RData*. Para quem quiser alterar o pacote *le.pesquisa* utilize o comando *fix()*.

```
  load("~/Alexandre Nicolella/Aulas/FEA-RP/Computação III/Pnad 2014/Leitura_em_R/dicPNAD2014.RData") #
load("~/Alexandre Nicolella/Aulas/FEA-RP/Computação III/Pnad 2014/Leitura_em_R/IBGEPesq.RData") # Esse

fix(le.pesquisa) # abre o codigo fonte da função le.pesquisa
```

Vamos agora dar um nome mais curto ao caminho para o programa encontrar os nosso microdados. O primeiro chamado de *caminho_micro* fornece o caminho para os dados de domicílio e o segundo para os dados de pessoas.

```
  caminho_micro <-file.path("~/Alexandre Nicolella/Aulas/FEA-RP/Computação III/Pnad 2014/Dados/DOM2014.
caminho_micro_pes <-file.path("~/Alexandre Nicolella/Aulas/FEA-RP/Computação III/Pnad 2014/Dados/PES201
```

Com o comando abaixo podemos fazer a leitura das variáveis de domicílio e de pessoas. Note que temos que indicar o dicionário, os códigos e indicando o total de variáveis do banco e o nome das variáveis, tanto para domicílios como para pessoas. Quem faz a leitura é a função *le.pesquisa* devidamente arrumada para versões mais recentes do R.

```
  dados_dom <- le.pesquisa(dicionario=dicdom2014,
                           pathname.in=caminho_micro,
                           codigos=dicdom2014[c(1:65),2],
                           nomes=dicdom2014[c(1:65),2],
                           quant=99)



dados_pes <- le.pesquisa(dicionario=dicpes2014,
                         pathname.in=caminho_micro_pes,
                         codigos=dicpes2014[c(1:325),2],
                         nomes=dicpes2014[c(1:325),2],
                         quant=99)
```

```
  ind_maiores <- subset(dados_pes, v8005>=18, )
var_sel <- c("V4803")
esc_med <- ind_maiores[var_sel]
```

```
dados_pes$V4742[dados_pes$V4742==999999999999] <- NA
ddados_pes1 <- na.omit(dados_pes)

#V0207 -  Condição de ocupação do domicílio
#1  Próprio - já pago
#2  Próprio - ainda pagando
#3  Alugado
#4  Cedido por empregador
#5  Cedido de outra forma
#6  Outra condição
#7 Não aplicável

Def_hab<- table (dados_dom$V0207)
Def_hab
hist (dados_dom$V0207)
Def_hab/sum(Def_hab)
prop.table(Def_hab)
sum(Def_hab[3:6])/sum(Def_hab)
```

Criando uma variável chave

É necessário criar uma variável chave. Cada indivíduo e cada família terá sua identificação única e assim caso precisamos podemos voltar ao banco original, retirar novas variáveis e incluir no banco existente sem precisar refazer o trabalho todo. Para isso criaremos as seguintes variáveis

```
  dados_pes$idFam<-dados_pes$V0102*10000+10*dados_pes$V0103+dados_pes$V0403
dados_pes$idPessoa<-dados_pes$V0102*100000+dados_pes$V0103*100+dados_pes$V0301
```

Selecionando e limpando as variáveis

Agora vamos selecionar as variáveis para criarmos a renda total dos indivíduos.

```
  rendVar <- c("V1252","V1255", "V1258", "V1261", "V1264", "V1267", "V1270", "V1273","V9058", "V9101",
distRenda <- dados_pes[rendVar]
```

Para calcular a renda total dos indivíduos iremos utilizar diversas variáveis. Primeiramente calcularemos a renda não oriunda do trabalho, ou seja, advindas de benefícios ou aplicações financeiras. Calculo da renda não oriunda do trabalho será:

```
  distRenda$rendns<-rowSums(cbind(distRenda$V1252, distRenda$V1255, distRenda$V1258, distRenda$V1261, di
```

A renda do trabalho e a renda total podem ser assim calculadas:

```
  distRenda$rends<-rowSums(cbind(distRenda$V9532, distRenda$V9535, distRenda$V9982,  distRenda$V9985, di
distRenda$rendtot<-rowSums(cbind(distRenda$rendns+distRenda$rends), na.rm=TRUE)
```

Para calcular o salário por hora ou o salário mensal ajustado para 40 horas trabalho semanal temos que calcular a quantidade de horas trabalhadas:

```
  distRenda$hrtrab<-(rowSums(cbind(distRenda$V9058, distRenda$V9101, distRenda$V9105,distRenda$V0713), r
distRenda$salHora<-distRenda$rends/distRenda$hrtrab
```

Vejamos o que temos até agora com relação a distribuição de renda. Abaixo está o histograma que fornece a distribuição empírica dos dados. Ainda temos que melhorar o nosso banco

```r
hist(distRenda$rendtot, breaks=20, col="red",xlab="Taxa de Câmbio", ylab="Frequência", main="Histogra
```

Vamos incluir novas variáveis no nosso banco e começar a prática de operação com banco de dados.

Aula Prática

Feita a introdução ao uso do R agora estamos preparados para exercitar o que vimos. Nesse ponto o aluno será convidado e refazer praticamente todos os passos anteriores mas em uma situação real. Vamos utilizar o banco de dados da Pesquisa Nacional por Amostra de Domicílio(PNAD), pesquisa realizada pelo IBGE. Ela pode ser acessada em http://www.ibge.gov.br/home/estatistica/populacao/ trabalhoerendimento/ pnad2015/default.shtm.

Nosso objetivo é estudar o diferencial de salário entre homens e mulheres por estado do Brasil. O banco de dados será fornecido ao aluno. Para aqueles interessados em estudar como abrir a pesquisa e selecionar as variáveis indicamos fazer a disciplina Computação para economista II.

Essa prática será composta de três módulos. O primeiro será a parte de organização e limpeza do bando de dados. O segundo será a análise descritiva e a terceira a decomposição de Oaxaca. Esses módulos estão assim divididos durante as semanas:

Móduo I - Organização do banco

A primeira parte da elaboração de uma análise é organizar o banco de dados. O banco de dados contem as seguintes variáveis:

1. IdFam - Identificação Família
2. IdPessoa- Identificação da Pessoa
3. UF - Estado
4. V9532 - Rendimento em dinheiro do trabalho principal.
5. V9058 - Horas trabalhadas por semana no trabalho principal.
6. V0302 - Gênero
7. V0404 - Cor
8. V4803 - Anos de estudo
9. V9001 - Trabalhava na semana de referência
10. V4713 - Condição de atividade no trabalho principal no ano
11. V4814 - Condição de ocupação no ano
12. V8005 - Idade
13. V9892 - Idade que começou a trabalhar
14. V4817 - Grupamento ocupacionais do trabalho principal
15. V0711 - Posição na ocupação no trabalho da semana de referência
16. V4727 - Área censitária - Região Metropolitana
17. V4728 - Situação censitária - Urbano/rural

As atividades de organização e o tempo para realizar estão descrito nos ítens abaixo.

- Semana 1 - entrega 20 abril

  - Fazer o merge entre os dois banco de dados disponibilizados
  - Selecionar seu estado
  - Manter as pessoas entre 25 e 50 anos de idade.
- Semana 2 - entrega 27 abril
  - Analisar a taxa de ocupação e a taxa de atividade
  - Manter somente as pessoas que estão trabalhando.
- Semana 3 - entrega 4 maio
  - Realizar a limpeza do banco de dados
  - Criar a variável grupo de escolaridade.
  - Criar a variável experiência
  - Criar a variável salário por hora e ln do salário por hora

Móduo II - Análise Descritiva

Agora vamos realizar a análise descritiva dos dados. As atividades de análise e o tempo para realizar estão descrito nos itens abaixo.

Análise descritiva das variáveis

- Semana 4 - entrega 11 maio
  - Descrever a média, mediana, desvio padrão etc, das variáveis do banco total e dividido por homem e mulher. * Realizar o percentual de indivíduos do banco por cor, grupo de escolaridade, tipo de emprego, rural, urbano, região metropolitana.

Tabelas cruzadas

1. Semana 5 e 6 - entrega 25 maio
   (a) Tabela salário hora por idade e gênero
   (b) Tabela salário hora por escolaridade e gênero
   (c) Tabela salário hora por cor e gênero.
   (d) Tabela salário hora por idade, escolaridade e gênero
   (e) Tabela salário hora por cor, escolaridade e gênero
   (f) Tabela salário hora por região metropolitana, escolaridade e gênero
   (g) Tabela salário hora por urbano, escolaridade e gênero
   (h) Tabela ln do salário hora por percentil e por gênero

Análise gráfica

1. Semana 7 e 8 - entrega 8 junho
   (a) Gráfico de barras salário hora por grupo de escolaridade
   (b) Gráfico boxplot salário hora por gênero
   (c) Histograma do salário hora por gênero e raça
   (d) Gráfico Violinplot salário hora por gênero e por grupo de escolaridade
   (e) Gráfico Violinplot da experiência por gênero

(f) Matriz scatterplot entre salário hora, idade e experiência

(g) Densidade do ln salário hora por gênero

Móduo III - Análise de regressão e Oaxaca-blinder

Análise estatística

1. Semana 9 - entrega 15 de junho
   (a) Montar a matriz de correlação e o gráfico de correlação entre as variáveis contínuas do banco de dados, salário hora, experiência, anos de estudo, idade.

   (b) Fazer o teste de diferença de média do salário hora entre homens e mulheres total

   (c) Fazer o teste de diferença de média do salário hora entre homens e mulheres considerando grupo de idade

   (d) Fazer o teste de diferença de média do salário hora entre homens e mulheres considerando cor

   (e) Fazer o teste de diferença de média do salário hora entre homens e mulheres considerando grupo de escolaridade

   (f) Fazer o teste de diferença de média do salário hora entre homens e mulheres considerando grupo de escolaridade e cor

Análise Econométrica

1. Semana 10 - entrega 22 de junho
   (a) Montar um a regressão múltipla do salário contra as variáveis explicativas fornecidas

   (b) Fazer a decomposição de Oaxaca

Aula Prática II

O objetivo dessa aula prática é analisar a distribuição da renda per capita dos brasileiros por região do Brasil, Sul, Sudeste, Centro-oeste, Norte e Nordeste. Vamos utilizar a PNAD 2014 e calcular ao final os índices de desigualdade como Gini, Theil e índice de pobreza.

Módulo I - Montando o banco de dados

Agora que já temos alguma ideia de como trabalhar com a PNAD vamos explorar um pouco mais a fundo essa pesquisa. Primeiramente vamos montar um banco de dados contendo as seguintes variáveis

1. Identificação (chave)
2. Estado
3. Renda do trabalho, renda do não trabalho e renda total conforme apresentado no item anterior
4. Horas trabalhadas e salário por hora conforme apresentado no item anterior
5. Número de integrantes na família
6. Gênero
7. Cor
8. Anos de estudo
9. Trabalhava na semana de referência
10. Idade
11. Idade que começou a trabalhar
12. Tipo de ocupação
13. Região Metropolitana
14. Urbano/rural
15. Acesso a esgoto (BD domicílio)
16. Acesso a água (BD domicílio)
17. E demais variáveis que serão descritas nas atividades abaixo

As atividades práticas e o tempo para realizar estão descrito nos itens abaixo. As atividades devem ser entregues no dia, atividades fora do prazo não serão consideradas.

1. Semana 1 - entrega 20 abril
   (a) Criar os dois banco de dados com as variáveis acima. O Primeiro será criado com base nos dados das pessoas e outro criado com base nos dados do domicílio. Ver os dados apresentados acima
   (b) Fazer o merge desses três banco de dados, os dois acima e aquele criado em aula com base nos dados de renda da família.
   (c) Selecionar a região com a qual irá trabalhar (Sul, Sudeste, Centro-Oeste, Norte e Nordeste).
   (d) Realizar a limpeza do banco de dados.
2. Semana 2 e 3 - entrega 4 maio
   (a) Calcular a renda per capita familiar considerando a definição de Hoffmann, R. *"O rendimento domiciliar per capita (RDPC) é definido como a razão entre o rendimento domiciliar e o número de moradores, excluindo as pessoas cuja condição no domicílio é pensionista, empregado doméstico ou*

parente de empregado doméstico. São considerados apenas os domicílios particulares permanentes com declaração do rendimento domiciliar."

(b) Imputar esses dados de renda per capita para cada indivíduo, mantendo assim o banco por pessoa.

(c) Calcular a participação da renda do não trabalho e da renda do trabalho na renda per capita do familiar.

3. Semana 4 - entrega 11 maio

(a) Calcular a escolaridade média dos adultos da família e imputar no banco de dados por pessoa.

(b) Calcular o número de filhos e número de adultos na família e imputar no banco de dados por pessoa.

Móduo II - Análise Descritiva

Agora vamos realizar a análise descritiva dos dados. As atividades de análise e o tempo para realizar estão descrito nos itens abaixo.

1. Semana 5 - entrega 18 maio

(a) Descrever a renda per capita familiar com relação a média, mediana, desvio padrão etc.

(b) Descrever a renda per capita por decil de renda.

(c) Calcular a participação de pensões e aposentadoria na renda per capita familiar

(d) Realizar o percentual de indivíduos do banco por cor, grupo de escolaridade, tipo de emprego, rural urbano, região metropolitana.

2. Semana 6 e 7 - entrega 1 junho

(a) Montar o histograma da distribuição de renda

(b) Montar violinplot da distribuição de renda por nível médio de escolaridade dos pais.

(c) Montar a distribuição de renda e aproximar pela distribuição normal. Plotar no mesmo gráfico.

(d) Boxplot da distribuição de renda considerando região metropolitana e não metropolitana e rural urbano.

(e) Percentual da renda apropriada por decil da população

Móduo III - Análise dos resultados

Análise estatística

1. Semana 8 - entrega 8 junho

(a) Calcular a Curva de Lorenz e o índice de Gini (utilizar o pacote "ineq- https://cran.r-project.org/web/packages/ineq/ineq.pdf)

2. Semana 9 - entrega 15 de junho

(a) Montar o índice de pobreza e extrema pobreza para o ano de 2014. (utilizar o pacote "ineq- https://cran.r-project.org/web/packages/ineq/ineq.pdf)

Apresentação final

1. Semana 10 - entrega 22 de junho
   (a) Montar uma apresentação final consolidada.

Utilizando Controladores de Versão

Git

O Bitbucket

Controlando versões passo a passo

Primeiramente abra uma conta no Bitbucket. Crie um novo repositório entrando em:

Repository/Create new repository.

Entre no terminal do seu computador e forneça o endereço onde será criado o repositório local na sua máquina. Para fazer isso, copie o caminho e escreva no terminal:

cd C:/aulas.

Agora entre no Bitbucket e copie o endereço clone clicando em:

Clone dentro de *action.*

Cole esse endereço clone dentro do seu terminal. Agora está determinado onde será colocado localmente todo o trabalho do seu time. Para saber o que está versionado ou não, digite na linha de comando do seu terminal:

git status.

Mostra os status dos arquivos no seu repositório. Muitas vezes estamos interessados em versionar somente alguns arquivos. Para ignorar os arquivos pode-se criar um arquivo no editor de texto com o seguinte nome:

.gitignore

Esse possui dentro todos os arquivos e suas extensões que devem ser ignorados. Para versionar arquivos digite o comando:

git add

Na sequência os nomes das pastas e arquivos que serão versionados, por exemplo:

git add Apostila/

Para fazer o envio do seu trabalho para o repositório no Bitbucket utilize sempre dois passos. O primeiro é enviar o arquivo para o seu repositório local por meio do comando:

git commit -am "texto".

O texto deve ser algo que identifique o commit, por exemplo “1o commit”. Aqui ficam os arquivos que ainda estão inacabados e não prontos para a colaboração dos seus parceiros de trabalho.

Ao estar com os arquivos prontos para colaboração utiliza-se o comando:

git push

Esse envia do seu repositório local para o Bitbucket, assim seus parceiros podem trabalhar no arquivo que atualizou. Agora você e seus parceiros de trabalhos estão trabalhando ao mesmo tempo no mesmo arquivo. Ao começar a trabalhar novamente utilize o comando:

git pull

Esse traz os arquivos que estão no Bitbucket e atualiza o seu repositório local para que consiga trabalhar na versão mais atual. Trabalhe e ao finalizar proceda os passos para enviar seu trabalho para o Bitbucket.