

# Manual Econometria Aplicada

Lucas Augusto Silva Ribeiro  
Universidade Federal do Rio Grande do Sul(UFRGS)

2021-04-12



# Contents

<b>1</b>	<b>Comentários Iniciais</b>	<b>5</b>
<b>2</b>	<b>Introdução ao R</b>	<b>7</b>
2.1	Por que o R? . . . . .	7
2.2	Estrutura de dados no R . . . . .	8
2.3	Pacotes . . . . .	11
2.4	Lendo Arquivos . . . . .	13
2.5	Trabalhando com dados . . . . .	13
2.6	Gráficos . . . . .	14
2.7	Referências úteis para ajudar a mexer com o R . . . . .	16
2.8	Exercícios para treino . . . . .	17
<b>3</b>	<b>Estatísticas Padrão e Regressão linear</b>	<b>19</b>
<b>4</b>	<b>Séries Temporais no R</b>	<b>21</b>
4.1	Introdução a análise de séries temporais no R . . . . .	21
4.2	Introdução a séries temporais . . . . .	22
4.3	Modelo ARMA . . . . .	26
4.4	Análise de uma Série Temporal . . . . .	28
4.5	Metodologia de Box-Jenkins . . . . .	30



# Chapter 1

## Comentários Iniciais

Este é um pequeno manual de uso do R para estudos econométricos e análise de dados feito com o Rbookdown. É um trabalho em progresso que tem como objetivo auxiliar os alunos de econometria aplicada da turma 2020/2.

Para entrar em contato sobre erros encontrados, sugestões e dúvidas, estou disponível no e-mail: [lucas.augusto@ufrgs.br](mailto:lucas.augusto@ufrgs.br)



# Chapter 2

## Introdução ao R

### 2.1 Por que o R?

O uso do software R tem várias vantagens, principalmente no meio acadêmico mas também possui várias vantagens pessoais:

- É gratuito, isto é uma vantagem para que possa ser usado sem necessidade de pagar por uma licença tanto por você quanto pela empresa que te contrate.
- O R por ser uma linguagem de programação ajuda em entender conceitos e ideias de linguagens de programação, facilitando o aprendizado de outras línguas.
- A existência de pacotes facilita em muito o uso de análises, provavelmente o que você quer fazer já foi feito por alguém e não há a necessidade de reinventar a roda.
- Integrabilidade com  $\text{\LaTeX}$  permite uma ótima fonte para a criação de pesquisas reprodutivas.

- A comunidade do R é bem extensa então erros e problemas encontrados provavelmente terão alguma solução no StackOverflow. Além de ser muito fácil encontrar material prático e teórico.

## 2.2 Estrutura de dados no R

### 2.2.1 Vetores

O R possui algumas estruturas principais. Vetores, fatores, matrizes, data frames e listas. Vetores são a estrutura mais básica do R, podem armazenar tanto caracteres como números mas apenas um tipo no mesmo vetor. O tipo dos elementos do vetor permite as operações que se pode fazer com eles, por exemplo não é possível a soma de vetores com caracteres (diferente do Python). Para criar um vetor, é apenas escrever os elementos do vetor dentro da estrutura `c()` separados por vírgulas.

```
letras1 <- c("a", "b", "c")  
  
letras2 <- c("d", "e", "f")  
  
letras1+letras2
```

```
## Error in letras1 + letras2: non-numeric argument to binary operator
```

Este erro que ocorreu era esperado, justamente por tentarmos fazer uma operação designada apenas a objetos do tipo numérico com objetos do tipo char.

Ao passo que vetores feitos com valores numéricos e com valores lógicos podem ser somados sem problemas. Vetores lógicos podem ser somados pois TRUE tem valor 1 e FALSE tem valor 0. Isto é um recurso muito útil ao passo que podemos assim descobrir quantas variáveis possuímos faltando, ou do tipo que nos interesse.



Perceba que enquanto letras precisam de parênteses para serem alocados dentro de um vetor, números não precisam. Isto acontece pois ao usarmos letras sem parênteses o R procura um objeto. Ao criarmos objetos colocamos `ou =` ou `ou <-` para atribuir seu valor. Vetores possuem apenas uma dimensão então para selecionar um elemento de um vetor se faz `vetor[i]`, onde *i* é a posição do elemento.

```
numeros1 <- c(1,2,3)

numeros2 <- c(4,5,6)

a <- numeros1+numeros2

a
```

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

```
a[2]
```

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

### 2.2.2 Matrizes

Matrizes são uma concatenação de data que aceitam apenas valores numéricos. Percebam que criei um vetor com outros dois vetores para a construção dessa matriz. O formato que eles se encontram é devido ao chamado de uso de matriz, se não eles apenas ficariam concatenados um ao lado do outro.

```
matriz <- matrix(c(numeros1, numeros2), nrow=2, ncol = 3)

print(matriz)
```

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

```
print(c(numeros1, numeros2))
```

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

### 2.2.3 Data Frames

Data frames são uma concatenação de data que aceitam mais de um tipo de objeto. Cada coluna num data frame é constituído por um tipo de objeto diferente. Um jeito de se pensar em data frames no R é colocar vários vetores um do lado do outro ou um acima do outro.

Data Frames, assim como matrizes possuem duas dimensões então para selecionar um elemento se escreve `df[i,j]`. Outra maneira é indexar a coluna do `df` e depois selecionar o elemento da coluna. É o objeto mais utilizado na análise de dados.

É interessante notar que nesta apostila será usado, sempre que possível o uso do pacote `tidyverse` e suas funcionalidades. Neste sentido, é necessária uma pequena menção ao formato `tibble` que, de maneira prática, não é muito diferente de um data frame normal porém permite uma função `print` mais simples e detalhada, permitindo um uso do `print` no objeto sem muitos problemas.

```
df <- data.frame(numeros1, letras1, numeros2, letras2)

print(df)
```

```
##   numeros1 letras1 numeros2 letras2
## 1         1      a         4      d
## 2         2      b         5      e
## 3         3      c         6      f
```

```
df[1,3]
```

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

```
df$letras1[1]
```

```
## [1] "a"
```

### 2.2.4 Listas

Listas são elementos extremamente versáteis e podem ser pensadas como um vetor capaz de concatenar tudo dentro dele. São mais flexíveis e complexos que data frames.

```
lista <- list(numeros1, letras1, numeros2, letras2, df)
```

```
print(lista)
```

```
## [[1]]
## [1] 1 2 3
##
## [[2]]
## [1] "a" "b" "c"
##
## [[3]]
## [1] 4 5 6
##
## [[4]]
## [1] "d" "e" "f"
##
## [[5]]
##   numeros1 letras1 numeros2 letras2
## 1         1      a         4      d
## 2         2      b         5      e
## 3         3      c         6      f
```

## 2.3 Pacotes

O uso de pacotes simplifica em muito o uso do R. Simplifica código, aumenta as capacidades de importação de dados, permite o uso de novas funções e

criação de visualizações de dados mais bonitas. Para instalar um pacote, é só usar a função `install.packages()`. Caso haja alguma dúvida no que a função faz e quais argumentos são usados por ela, é só escrever `?função`, serve para base de dados dentro do R também.

Podemos instalar manualmente os pacotes ou instalar vários de uma vez. Interessante a instalação dos seguintes pacotes para a análise de dados e análise de séries temporais e leitura de dados:

- Tidyverse
- Openxlsx
- Foreign
- Tsibble
- Fable
- Feasts
- MTS
- urca
- vars

Uma maneira de instalar todos de uma vez é adicionar um objeto com o nome de todos os pacotes e então instalá-los:

```
pacotes <- c(tidyverse, openxlsx, foreign, tsibble, fable, feasts,
             mts, urca, vars)
install.packages(pacotes)
```

Claro que sempre há novos pacotes surgindo e substituindo antigos. Uma boa estratégia é ver na documentação do pacote (geralmente é só procurar o nome do pacote no google junto de *documentation*) quando foi sua última atualização e se ele tem sido atualizado periodicamente.

### 2.3.1 Chamando pacotes

Para chamar um pacote para utilizá-lo basta escrever `library(nome do pacote)`, note que ao contrário da instalação que requer aspas no nome do pacote, para chamá-lo isto não é necessário.

```
library(tidyverse)
library(openxlsx)
```

## 2.4 Lendo Arquivos

Para ler um arquivo no R, basta mandar ler o arquivo com a função apropriada para o tipo de arquivo. Por exemplo, para um arquivo csv usa-se ou `read.csv` ou `read_csv` (`readr`). Para ler `xlsx` usa-se `read.xlsx` (`openxlsx`). Faça isso enquanto atribui este valor a um objeto para que fique guardado nos objetos do R e possa ser manipulado, se não o R apenas irá ler o arquivo e colocar no seu terminal.

```
caminho <- "/home/lucas/Desktop/Mestrado/
estagio docencia/
Datasets/Econometrics with applied methods/
chapter7/xls/xm701inp.xlsx"

dados <- read.xlsx(caminho)

outrosdados <- read.csv("/home/lucas/Documents/bustabit.csv")
```

## 2.5 Trabalhando com dados

Primeiramente, é necessário explicar o operador pipe, `%>%`. Ele é o equivalente ao uso de uma função composta mas que deixa mais simples para um humano ler. Enquanto no R se queremos uma função composta faríamos assim como uma sintaxe matemática normal com uma  $h(g(f(x)))$ , o operador pipe nos permite que façamos  $f(x)\%>\%g(x)\%>\%h(x)$ . Pequenos exemplos serão dados com base de dados simples.

```
library(tidyverse)
carros <- mtcars

mtcars %>%
```

```
filter(cyl >= 6) %>%
  summarize(media_mpg = mean(mpg))
```

```
##      media_mpg
## 1      16.64762
```

```
mean(carros[carros$cyl >= 6,]$mpg)
```

```
## [1] 16.64762
```

Ambos geram o mesmo resultado porém o segundo é muito mais simples de ser utilizado. Uma última nota sobre uso de sintaxe para análise de dados, para bases muito grandes, existe o pacote `data.tables` que possui uma sintaxe levemente diferente do padrão do R, e é extremamente rápida. Ao se trabalhar com big data, é melhor usar aquele pacote e aprender a manuseá-lo.

## 2.6 Gráficos

Demonstrar graficamente o que está acontecendo é importante tanto para o leitor quanto para quem está escrevendo algo, logo, bons gráficos ajudam não só nossa análise como a interpretação dela por terceiros. E agora uma verdadeira última nota sobre análise de dados: podemos ter base de dados no formato wide ou no formato long.

O formato long nos permite colocar gráficos de maneira mais simples com o `ggplot` usando alguma característica como um separador. Em geral, os dados que trabalhamos em economia estão no formato wide.

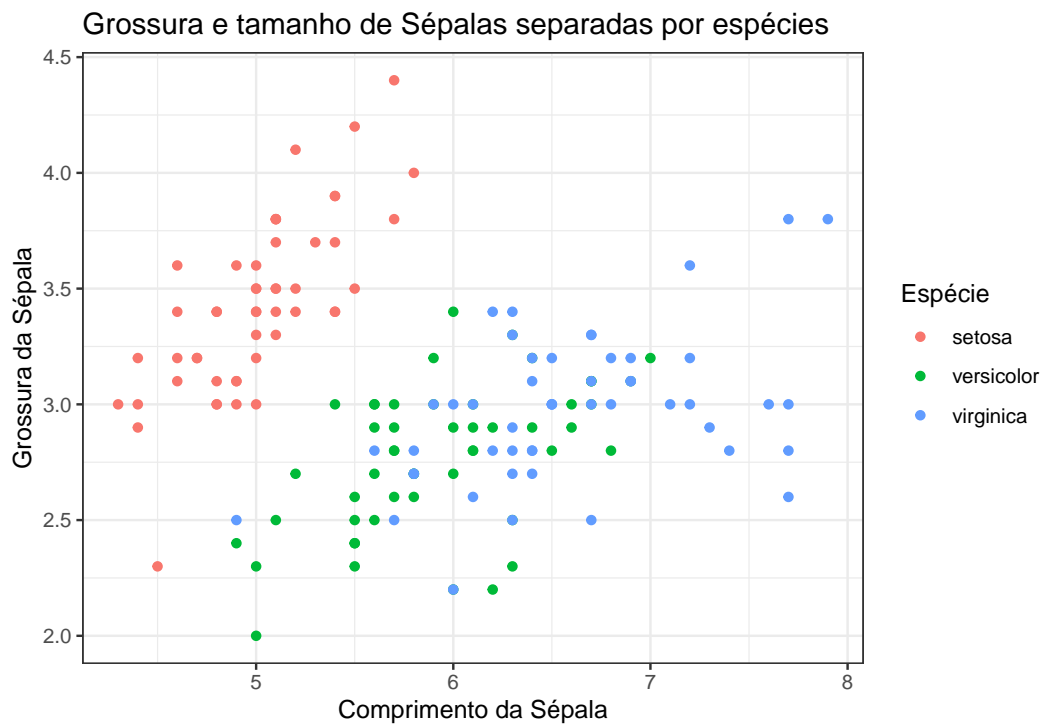
```
flores <- as_tibble(iris)

print(flores)
```

```
## # A tibble: 150 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>         <dbl>         <dbl>         <dbl> <fct>
```

```
## 1      5.1      3.5      1.4      0.2 setosa
## 2      4.9      3      1.4      0.2 setosa
## 3      4.7      3.2      1.3      0.2 setosa
## 4      4.6      3.1      1.5      0.2 setosa
## 5      5      3.6      1.4      0.2 setosa
## 6      5.4      3.9      1.7      0.4 setosa
## 7      4.6      3.4      1.4      0.3 setosa
## 8      5      3.4      1.5      0.2 setosa
## 9      4.4      2.9      1.4      0.2 setosa
## 10     4.9      3.1      1.5      0.1 setosa
## # ... with 140 more rows
```

```
ggplot(data = flores, aes(x=Sepal.Length, y = Sepal.Width, color = Species))+
  geom_point()+
  theme_bw()+
  labs(x= "Comprimento da Sépala", y= "Grossura da Sépala", color = "Espécie")+
  ggtitle("Grossura e tamanho de Sépala separadas por espécies")
```



A sintaxe do ggplot permite que sejam feitas várias modificações na maneira que se apresentam os dados, tentem mexer no argumento e layers que coloquei e terão diferentes gráficos. Cada layer é escrito na linha seguinte após o símbolo de `+`. A chamada padrao do ggplot é a que se encontra na primeira linha.

## 2.7 Referências úteis para ajudar a mexer com o R

- Data Types and Structures - Programming with R. <https://swcarpentry.github.io/r-novice-inflammation/13-supply-data-structures/>
- Data analytics for beginners. <https://data-flair.training/blogs/data-analytics-tutorial/>
- Livro: R for Data Science <https://r4ds.had.co.nz>
- Livro: R Graphics Cookbook, 2nd edition <https://r-graphics.org/>
- Livro: Efficient R Programming <https://csgillespie.github.io/efficientR/>
- Livro R Programming for Data Science <https://bookdown.org/rdpeng/rprogdatascience/>
- Hands on programming with R <https://rstudio-education.github.io/hopr/>
- Livro: Practical Data Science with R <https://www.amazon.com.br/Practical-Data-Science-Nina-Zumel/dp/1617291560>
- Site: Dicas e macetes no R [https://garhtarr.github.io/avfs/tips.html#:~:text=RStudio&text=In%20RStudio%20the%20keyboard%20shortcut,Option%20%20\(Mac\)](https://garhtarr.github.io/avfs/tips.html#:~:text=RStudio&text=In%20RStudio%20the%20keyboard%20shortcut,Option%20%20(Mac))

Obviamente, não é necessário ler todos os livros, alguns são mais completos que outros, escolha o que parecer mais interessante a você e siga com ele, alguns são ótimos manuais de bolso para consultar caso possua alguma dúvida. O *graphics cookbook*, por exemplo, é um bom manual quando houver a vontade de entender melhor o poder do pacote ggplot que não será abordado em aula.



## 2.8 Exercícios para treino

1. Procure uma base de dados qualquer presente no R, faça um filtro e crie um gráfico no R.
2. Usando o pacote `dbc` (leia sobre!), leia os dados de 2019 no RS que podem ser encontrados aqui:  
<http://www2.datasus.gov.br/DATASUS/index.php?area=0901&item=1&acao=26>

Leia a documentação e calcule a quantidade de mortes por ocorrência, segundo grupo do CID-10 no Rio Grande do Sul, separe entre homem e mulher. Compare com a tabela geral que é possível produzir no próprio site. Desafio: Baixe todos os dados, transforme tudo em um só data frame e faça uma comparação gráfica dos óbitos por estado.

3. Desafio: Usando o pacote `haven` para ler o arquivo do tipo `sav` da PED-DF (pesquisa de emprego e desemprego do DIEESE), tentar conseguir fazer com que valores totais da população ativa de 14 anos ou mais na tabela dos resultados mensais no mês de Agosto de 2019 seja o mesmo que o seu. (O valor é 2447)

As bases de dados e tabela para comparação podem ser encontrados em:

<https://www.dieese.org.br/analiseped/mensalBSB.html> (tabela)  
<https://www.dieese.org.br/analiseped/microdadosBSB.html> (microdados)

As bases de dados para o exercício 2 e 3 podem ser baixados também pelo meu github no endereço:

<https://github.com/Utrete/EconometriaAplicada>



## Chapter 3

# Estatísticas Padrão e Regressão linear

Esta sessão será feita após o término da parte de séries temporais e dados de painel. Devido a como o curso está progredindo no momento.

VEMAI



## Chapter 4

# Séries Temporais no R

### 4.1 Introdução a análise de séries temporais no R

O software R, sendo extremamente útil para análises estatísticas, possui uma ampla gama de opções, pacotes e métodos para se trabalhar com séries temporais. Trabalharemos principalmente com objetos do formato tipo `tsibble`, mas, quando necessário, outros tipos de objetos serão usados. Afinal, se o R é bem maleável, devemos explorar esta versatilidade dele. Os principais pacotes usados para o R na análise de séries temporais são, entre outros: `fable`, `feasts`, `tsibble`, `urca`, `MTS`, `vars`, `zoo`, etc.

#### 4.1.1 Objetos temporais e pacotes no R

Para se fazer análise temporal é necessário, obviamente, trabalhar com séries temporais. Objetos assim precisam possuir um indexador de tempo e valores observados. Pacotes como `lubridate` e `tsibble` permitem que mexamos com tipos de data `POSIXlt` para que sejam mais convenientes para trabalharmos. O pacote `tsibble` nos permite, também, converter um arquivo no R para um arquivo do tipo `ts` (time séries). Outras maneiras existem, como funções de coerção como do próprio R base como o `as.ts` e o do pacote `zoo`, o `as.zoo`. Em geral, toda função `as.”objeto”` é uma função que faz uma coerção de um tipo de objeto para outro tipo.

## 4.2 Introdução a séries temporais

### 4.2.1 Conceitos básicos de séries temporais

Ao aprendermos a análise de séries temporais aprendemos alguns conceitos fundamentais que são usados para a análise e criação de modelos. São com base neles que podemos analisar, predizer e testar nossos modelos. É necessário que esses conceitos estejam muito bem definidos e claros para o nosso uso deles. São eles:

- Estacionariedade
- Ruído Branco
- Sazonalidade
- Tendência
- Autocorrelação

### 4.2.2 Estacionariedade Fraca

O conceito de estacionariedade em séries temporais pode se referir tanto a estacionariedade estrita ou estacionariedade fraca. Ao trabalharmos com séries temporais, trabalhamos com o conceito de estacionariedade fraca.

Uma série, com variância finita, será fracamente estacionaria se:

- Sua média é constante ao longo do tempo
- $\gamma_{t,t-k} = \gamma_{0,k} \forall$  tempo  $t$  e lag  $k$

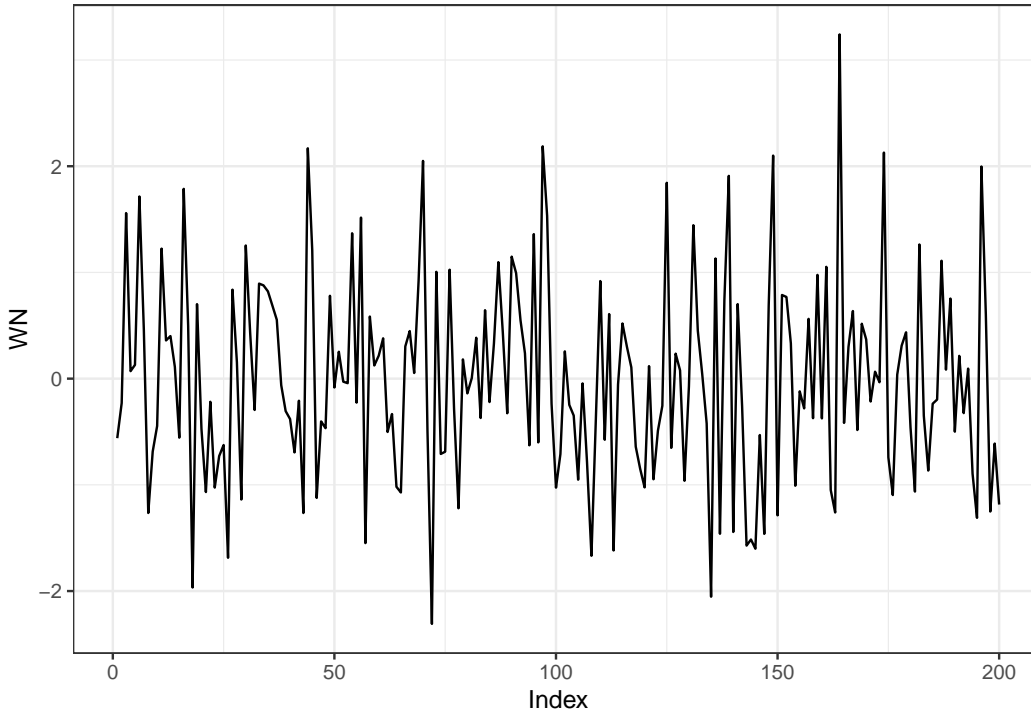
Onde  $\gamma_{t,t-k}$  é sua autocovariância entre o tempo  $t$  e o tempo  $t-k$ .

### 4.2.3 Ruído Branco(RB)

O ruído branco é uma das séries estacionárias mais importantes pois muitos processos podem ser construídos a partir de um ruído branco. Como o ruído branco é uma série fracamente estacionária, temos:

- $E(\varepsilon_t) = 0 \forall t$
- $E(\varepsilon_t^2) = \sigma^2 \forall t$
- $E(\varepsilon_t \varepsilon_{t-j}) = 0, \forall j \neq 0$

Em particular, se o ruído branco segue a distribuição  $\varepsilon_t \sim N(0, \sigma^2)$  este é chamado de ruído branco gaussiano.



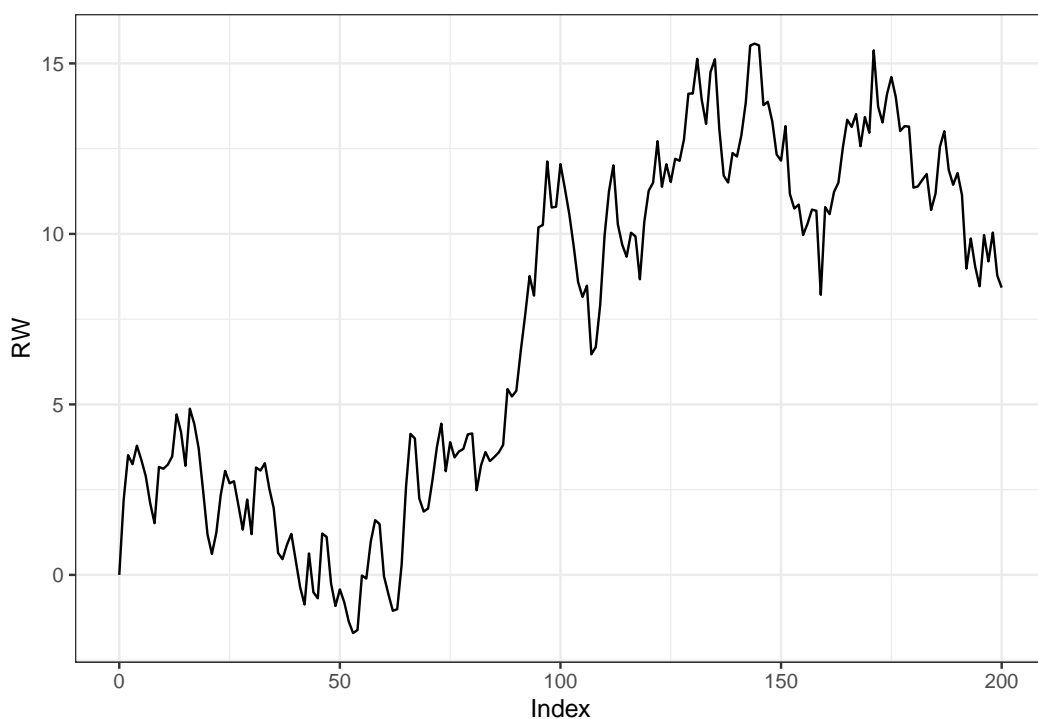
#### 4.2.4 Passeio Aleatório

Suponha agora que temos uma sequência de variáveis aleatórias identicamente e independentemente distribuídas de média 0 e variância  $\sigma_e^2$  construída como:

$$\left\{ \begin{array}{l} Y_1 = e_1 \\ Y_2 = e_1 + e_2 \\ \vdots \\ Y_t = e_1 + e_2 + \dots + e_t \end{array} \right. \quad \text{Alternativamente: } Y_t = Y_{t-1} + e_t \quad (4.1)$$

Este processo é denominado passeio aleatório e possui as seguintes propriedades:

- $E(y_t) = 0$
- $E(y_t^2) = t\sigma^2$
- $\gamma_{t,s} = cov(Y_t, Y_s) = t\sigma^2, \forall j \neq 0$



Compare este gráfico com o anterior, perceba que a média não é mais estável no tempo, esta série é não-estacionária.

#### 4.2.5 Médias Móveis(MA) e FAC

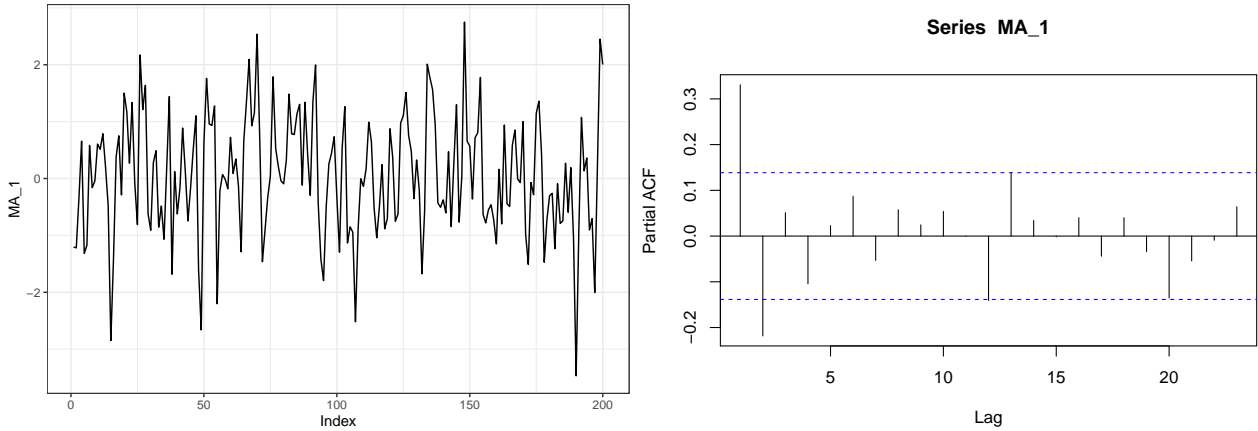
Considere o processo estocástico:

$$y_t = \mu + \varepsilon_t + \theta \varepsilon_t$$



Onde  $\varepsilon_t$  é um RB. Este processo é chamado de médias móveis. Como ele depende do erro contemporâneo e o erro imediatamente passado, é denotado como um MA(1).

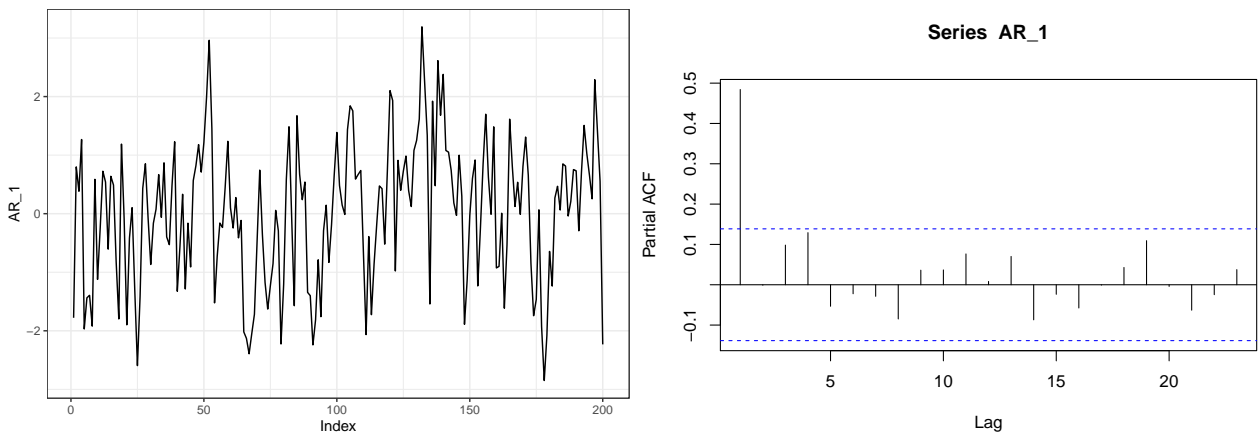
A Função de Auto-correlação (FAC) é o gráfico da autocorrelação contra a defasagem. A FAC permitirá identificar a ordem q de um processo MA(q).



#### 4.2.6 Processo AutoRegressivo(AR) e FACP

Considere, agora o seguinte processo estocástico:

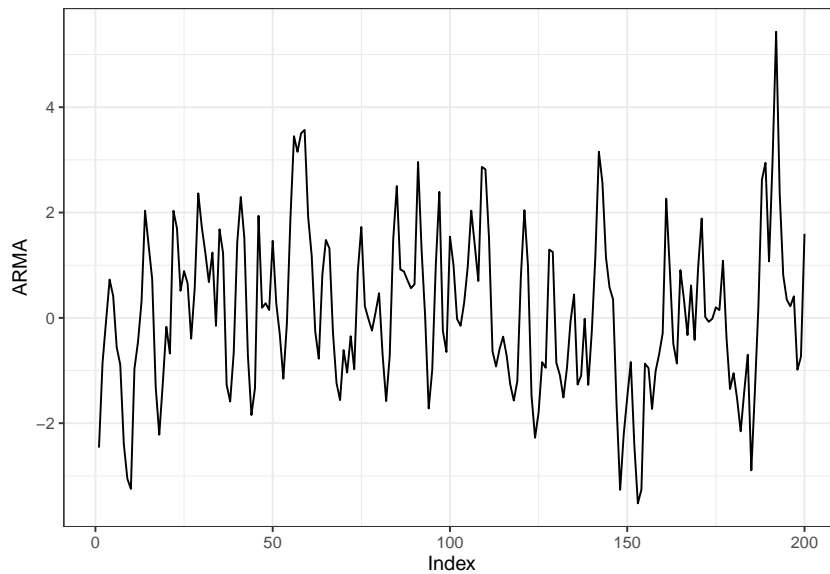
$$y_t = c + \phi y_{t-1} + \varepsilon_t$$



Onde  $\varepsilon_t$  é um RB. Este processo é denominado um processo AutoRegressivo de ordem 1, ou AR(1). A FACP consiste, *a grosso modo* de regredir  $y_t$  contra  $y_{t-1}$  e  $y_{t-2}$  e avaliar os coeficientes em um gráfico contra as ordens de defasagem. A FACP permitirá identificar a ordem  $p$  de um processo AR( $p$ ).

## 4.3 Modelo ARMA

Um processo ARMA, então será um processo que possui tanto um componente Autorregressivo quanto um de médias móveis. Caso ele seja estacionário, podemos fazer sua análise.

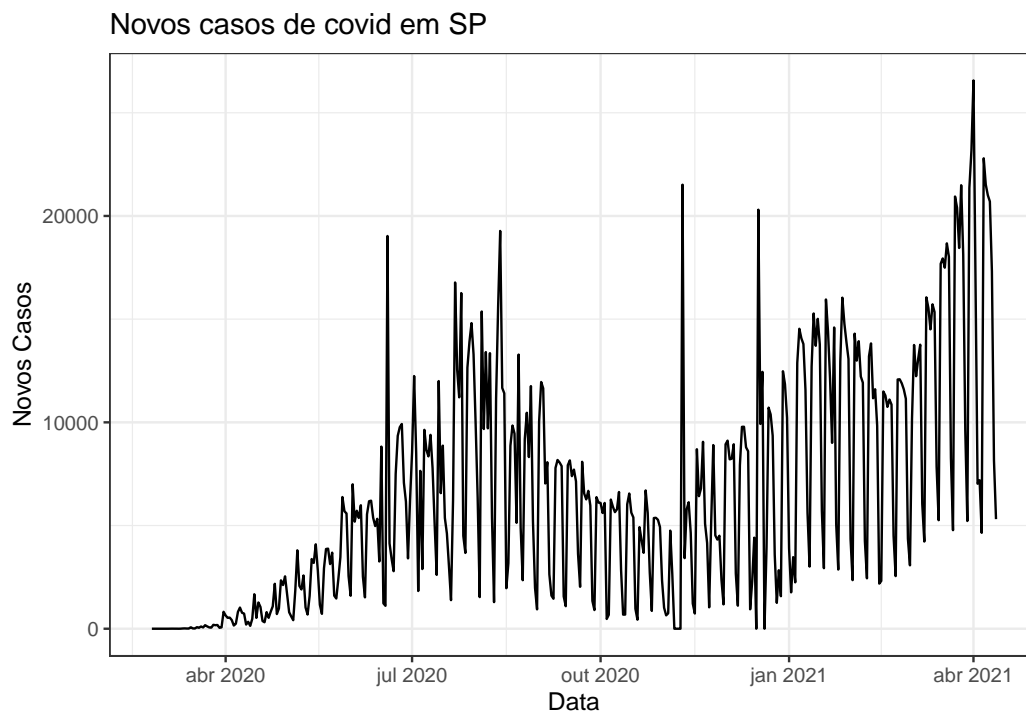


### 4.3.1 Componentes de uma Série Temporal

Toda série temporal mais complexa que trabalhamos possui pelo menos um dos seguintes fatores:

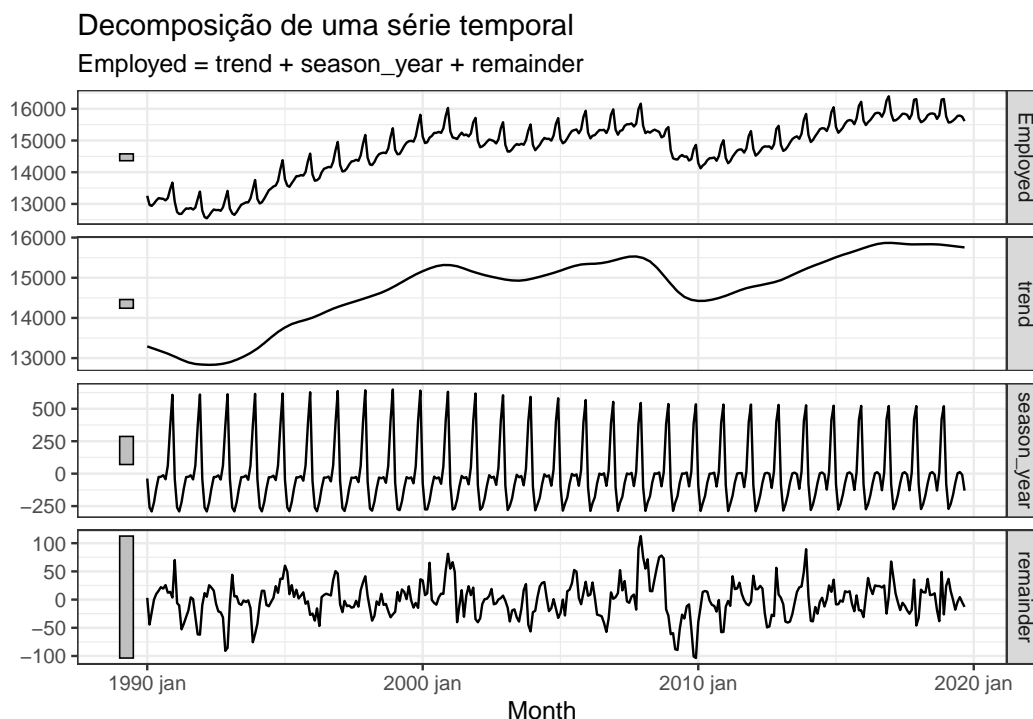
- Sazonalidade
- Ciclo
- Tendência

Estes componentes fazem com que nossa série não seja mais fracamente estacionária e precisamos, então, trabalhá-la para que possamos fazer alguma análise.



Esta série dos dados do covid possui tanto uma tendência positiva quanto uma sazonalidade a cada sete dias devido a forma que os dados são colocados no sistema. Modelos do tipo ARIMA não são bons para esse tipo de série mas caso fossem utilizados seria necessário corrigir a sazonalidade a cada 7 dias e a tendência antes de qualquer análise.

## 4.4 Análise de uma Série Temporal

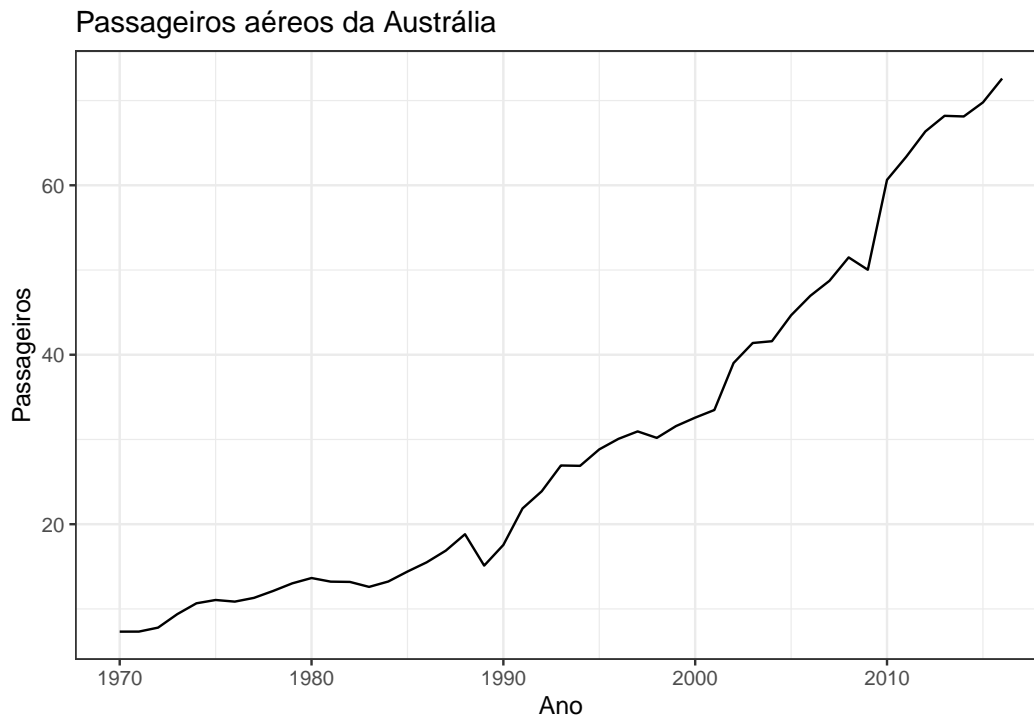


### 4.4.1 Metodologia Análise de uma Série Temporal por SARIMA

O primeiro passo ao analisar uma série temporal é fazer seu gráfico e analisá-lo, ver quais características ele possui e, a partir disso, ir trabalhando cada um de seus componentes para que o que sobra a nós seja algo estacionário que possamos analisar.

### 4.4.2 Tratando tendência

Uma série possui tendência quando há um crescimento ou decaimento de longo prazo no nível dos dados. Não precisa ser linear, pode ser visto como uma “mudança de direção.”



Quando uma série possui tendência ela não é estacionária e muito provavelmente possui uma raiz unitária. Precisamos, então, testar para a existência de uma raiz unitária e caso exista, lidar com ela.

O processo normalmente usado para lidar com a existência de uma raiz unitária é o uso de diferenças na série. Num  $ARIMA(p,d,q)$  enquanto  $p$  e  $q$  se referem a ordem das partes do AR e MA, respectivamente, o  $d$  é referente a integração. Ou seja, é o número de quantas vezes precisamos tirar a diferença da série para que ela se torne estacionária. Alguns dos testes utilizados é o ADF – Augmented Dickey Fueller e o Phillips-Person.

### 4.4.3 Tratando Volatilidade

Muitas vezes uma tendência nos traz um aumento de volatilidade, ou seja, nossa variância não é mais constante no tempo. Quando usamos modelos SARIMA, temos como hipótese uma variância constante pelo tempo pois são processos advindos de ruídos brancos. Se a variância não está constante em relação ao tempo, precisamos, então lidar com ele. Um método muito usado é trabalhar com o log dos valores a serem trabalhados. Uma generalização

para esse problema é o uso de uma transformação Box-Cox.

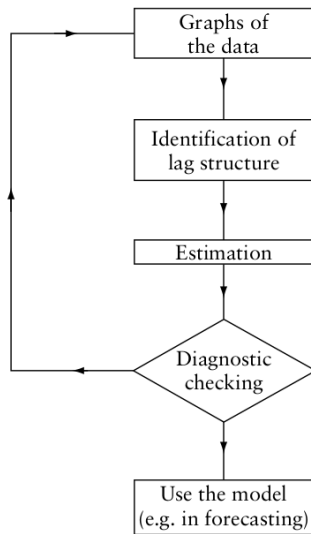
$$w_t = \begin{cases} \log(y_t) & \text{se } \lambda = 0 \\ \frac{y_t^\lambda - 1}{\lambda}, & \lambda \neq 0 \end{cases} \quad (4.2)$$

O pacote `feasts` possui uma função, `guerrero`, que procura o melhor  $\lambda$  que minimiza a volatilidade da série.

#### 4.4.4 Tratando Sazonalidade

Quando a série apresenta um componente sazonal precisamos mudar nosso  $\text{ARIMA}(p,d,q)$  para um  $\text{SARIMA}(p,d,q)\text{X}(P,D,Q)_m$ . Ficamos assim com uma parte não sazonal e uma parte sazonal, ambas com diferentes parâmetros autorregressivos, de médias móveis e integração.

### 4.5 Metodologia de Box-Jenkins



Esta imagem dá uma ideia bem geral de como se cria e analisa um modelo de acordo com a metodologia de Box-Jenkins. Nesta metodologia podemos chegar a parâmetros diferentes dos modelos ou então ao mesmo modelo por caminhos diferentes. Em diversos livros é encontrado imagens parecidas com a mesma idéia de como fazer o passo a passo ao lidar com o modelo

### 4.5.1 Estimando o modelo

Como dito na imagem, é necessário estimar o modelo mas como se faz isso? Existem diversos passos que podemos tomar neste momento. Uma rotina padrão é analisar o critério de informação, seguido de uma análise dos erros e uma predição retirando uma quantidade de tamanho  $p$  das observações e usando  $t - p$  para se fazer uma predição e comparar o predito e o realizado.

### 4.5.2 Critério de Informação

Critério de Informação	Fórmula	Detalhes
BIC(p,q)	$\ln \hat{\sigma}^2 + n \frac{\ln T}{T}$	Parcimonioso e assintoticamente consistente
AIC(p,q)	$\ln \hat{\sigma}^2 + n \frac{2}{T}$	Melhor para pequenas amostras, tende a escolher modelos sobreparametrizados
HQ(p,q)	$\ln \hat{\sigma}^2 + n \frac{2}{T} \ln \ln T$	Assintoticamente consistente, menos forte que o BIC

### 4.5.3 Análise dos Resíduos

Após usar os gráficos de ACF e PACFR para chegar aos parametros do modelo, queremos que os resíduos possuam apenas um ruído branco, ou seja, extraímos todas as informações relevantes para o modelo e temos apenas agora

um erro i.i.d de média 0 e variância  $\sigma^2$ . Alguns testes que podemos realizar são:

- FAC e FACP nos resíduos
- Teste de Ljung-Box – ausência de autocorrelação entre os resíduos
- Teste para normalidade (Shapiro-Wilk, Jarque-Bera)

#### 4.5.4 Predição

Um último passo é a parte de predição onde pegamos a série até um momento  $h$  no tempo,  $h < t$ , fazemos uma predição com o modelo que obtemos e analisamos como ele se comporta quando comparado a série original. Tanto em questão de valor quanto em questão de tendências, direção, etc.

#### 4.5.5 Material útil

- Livro: Forecasting, Principles and Practice 3ª edição (fpp3). <https://otexts.com/fpp3/non-seasonal-arma.html>
- Livro: Econometria de séries temporais, 2ª edição COLOCAR SITE INTEIRO AMAZON
- Livro: Econometric methods with applications in business and economics COLOCAR SITE INTEIRO AMAZON
- Livro: Introduction to Econometrics with R <https://www.econometrics-with-r.org/index.html>
- Livro Principles of Econometrics with R <https://bookdown.org/ccolonescu/RPoE4/>