R Notebook

Pedro Henrique Moreira Pereira

Documento de análise exploratoria de dados em R

```
# Load de variáveis e de bibliotecas
library(readxl)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  3.0.0     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------ t
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
library(stringr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:dplyr':
##
##     intersect, setdiff, union
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(janitor)
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```r
library(zoo)
```

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

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

```r
library(tsbox)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
library(plotly)
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```r
library(xts) # lib para time series
```

```
##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```r
library(spotifyr)
library(rmarkdown)
library(purrr)
source("multiplot.R", local = TRUE)
```

(A) Criação do indice de variação

```
indice_var <- function(x) {
  indice <- x
  indice[1] <- 100
  for(i in 2:length(x)) {
    indice[i] <- (1 + x[i]/100) * indice[i-1]
  }
  return(indice)
}
```

Filtrando por ano

```
load(file = "us_change.rda")

data_nivel <- us_change %>%
  clean_names()


# Criando o DF com os indices e filtrando pelo ano
indice_df<- data_nivel %>%
  filter(quarter >= as.Date("2000-01-01")) %>%
  select(-quarter) %>%
  map_df(function(x) x %>% indice_var())

# Criando DF com a coluna de data da base original

df_date <- data_nivel %>%
   filter(quarter >= as.Date("2000-01-01")) %>%
   select(quarter)

# Criando a coluna quarter de volta na base de indices
indice_df <- indice_df %>%
  mutate(quarter = df_date$quarter)
```

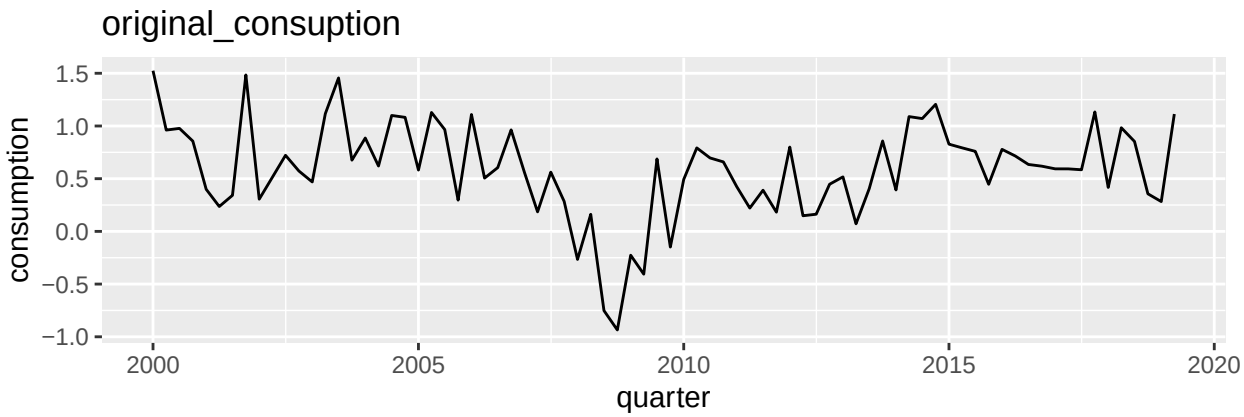
Multiplot data

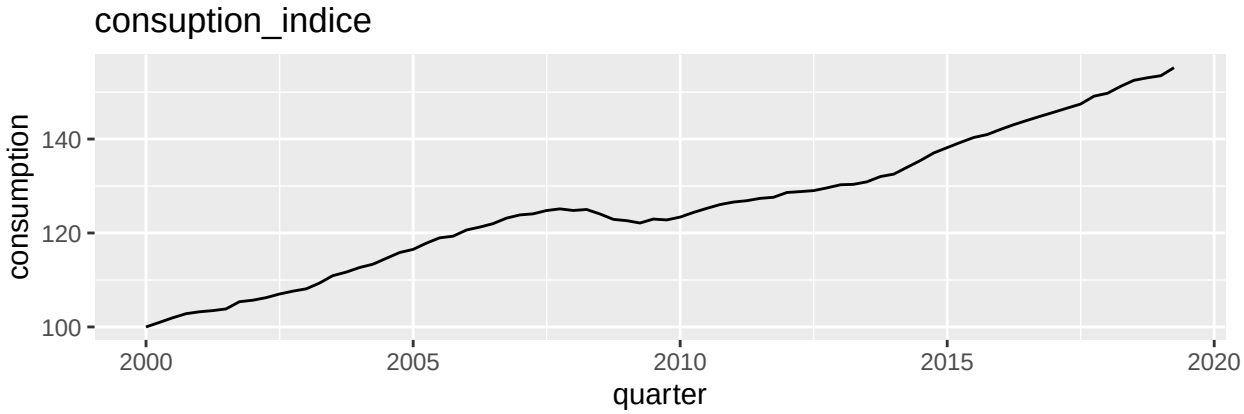
```
plot1 <- data_nivel %>%
  dplyr::filter((quarter >= as.Date("2000-01-01"))) %>%
  ggplot(aes(x = quarter, y = consumption)) +
    geom_line() +
    labs(title="original_consuption")
plot2 <- indice_df %>%
  ggplot(aes(x = quarter, y = consumption)) +
    geom_line() +
    labs(title="consuption_indice")
plot3 <- data_nivel %>%
  dplyr::filter((quarter >= as.Date("2000-01-01"))) %>%
  ggplot(aes(x = quarter, y = income)) +
    geom_line() +
    labs(title="original_income")
plot4 <- indice_df %>%
```

```
  ggplot(aes(x = quarter, y = income)) +
    geom_line() +
    labs(title="income_indice")
plot5 <- data_nivel %>%
  dplyr::filter((quarter >= as.Date("2000-01-01"))) %>%
  ggplot(aes(x = quarter, y = production)) +
    geom_line() +
    labs(title="original_production")
plot6 <- indice_df %>%
  ggplot(aes(x = quarter, y = production)) +
    geom_line() +
    labs(title="production_indice")
plot7 <- data_nivel %>%
  dplyr::filter((quarter >= as.Date("2000-01-01"))) %>%
  ggplot(aes(x = quarter, y = savings)) +
    geom_line() +
    labs(title="original_savings")
plot8 <- indice_df %>%
  ggplot(aes(x = quarter, y = savings)) +
    geom_line() +
    labs(title="savings_indice")
plot9 <- data_nivel %>%
  dplyr::filter((quarter >= as.Date("2000-01-01"))) %>%
  ggplot(aes(x = quarter, y = unemployment)) +
    geom_line() +
    labs(title="original_unemployment")
plot10 <- indice_df %>%
  ggplot(aes(x = quarter, y = unemployment)) +
    geom_line() +
    labs(title="unemployment_indice")
```

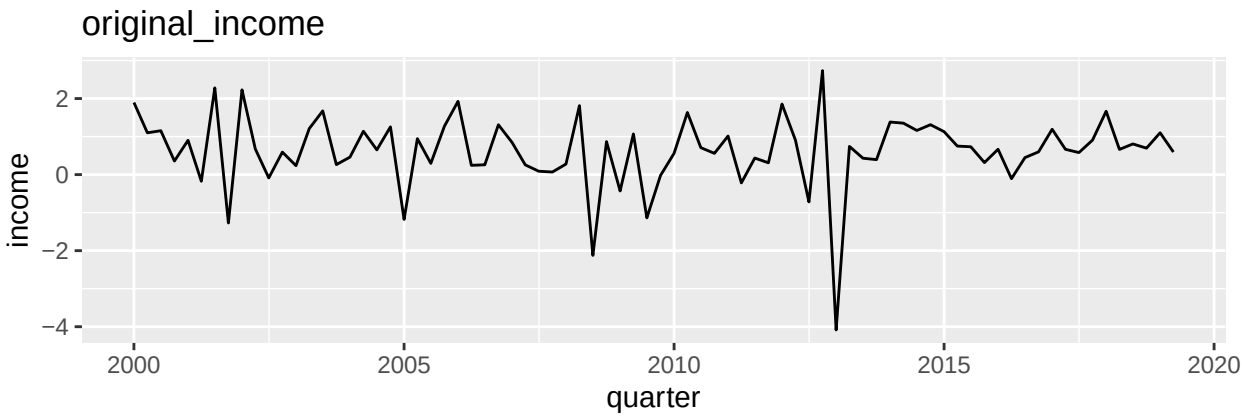
Plotando

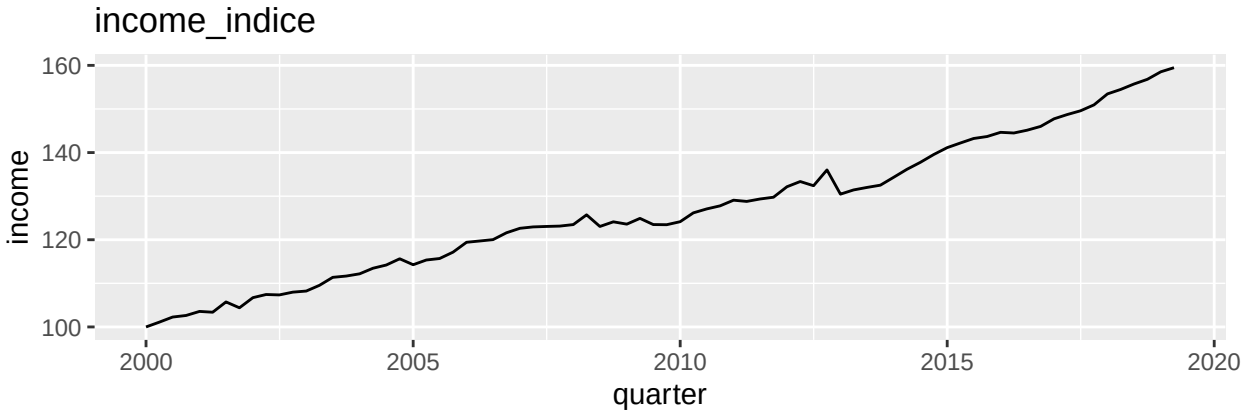
```
# Usando a função multiplot() apresentada na aula
multiplot(plot1,plot2)
```



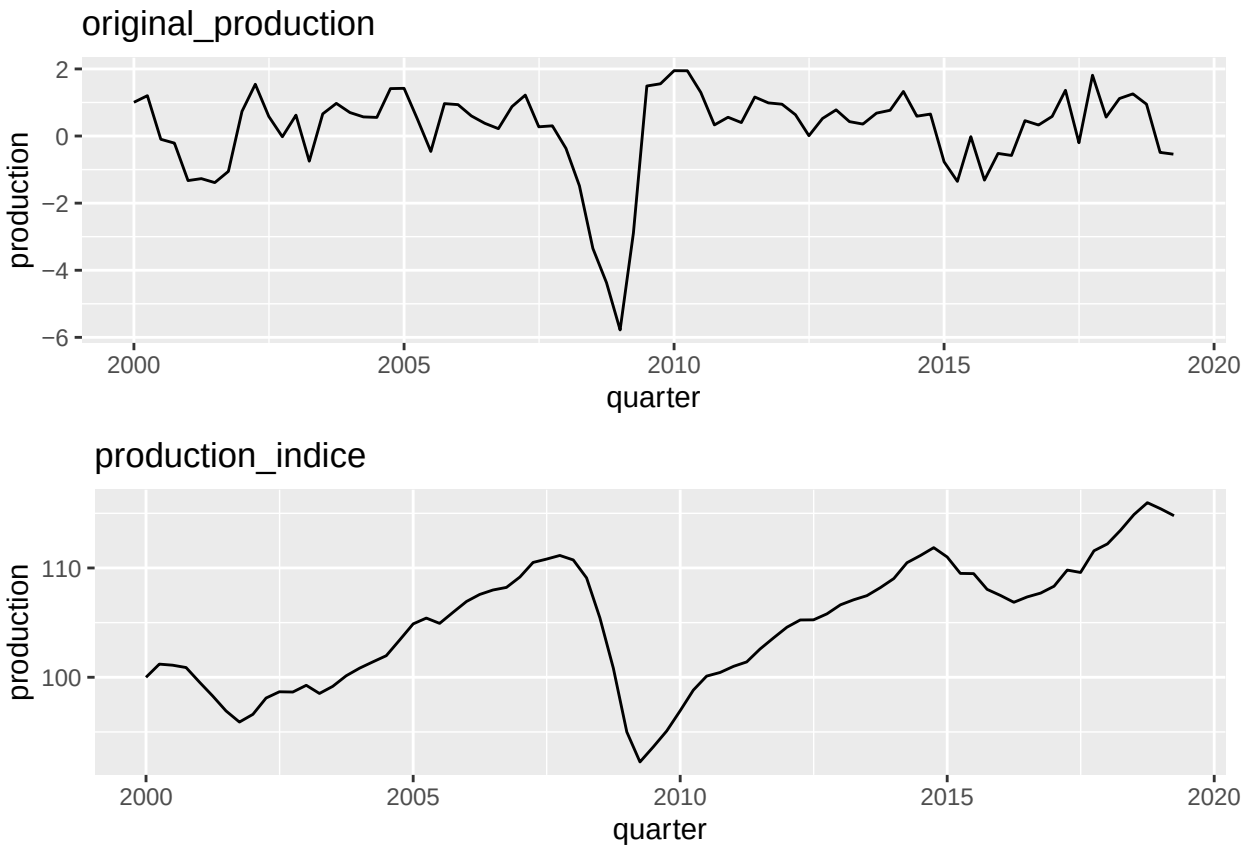



```
multiplot(plot3,plot4)
```

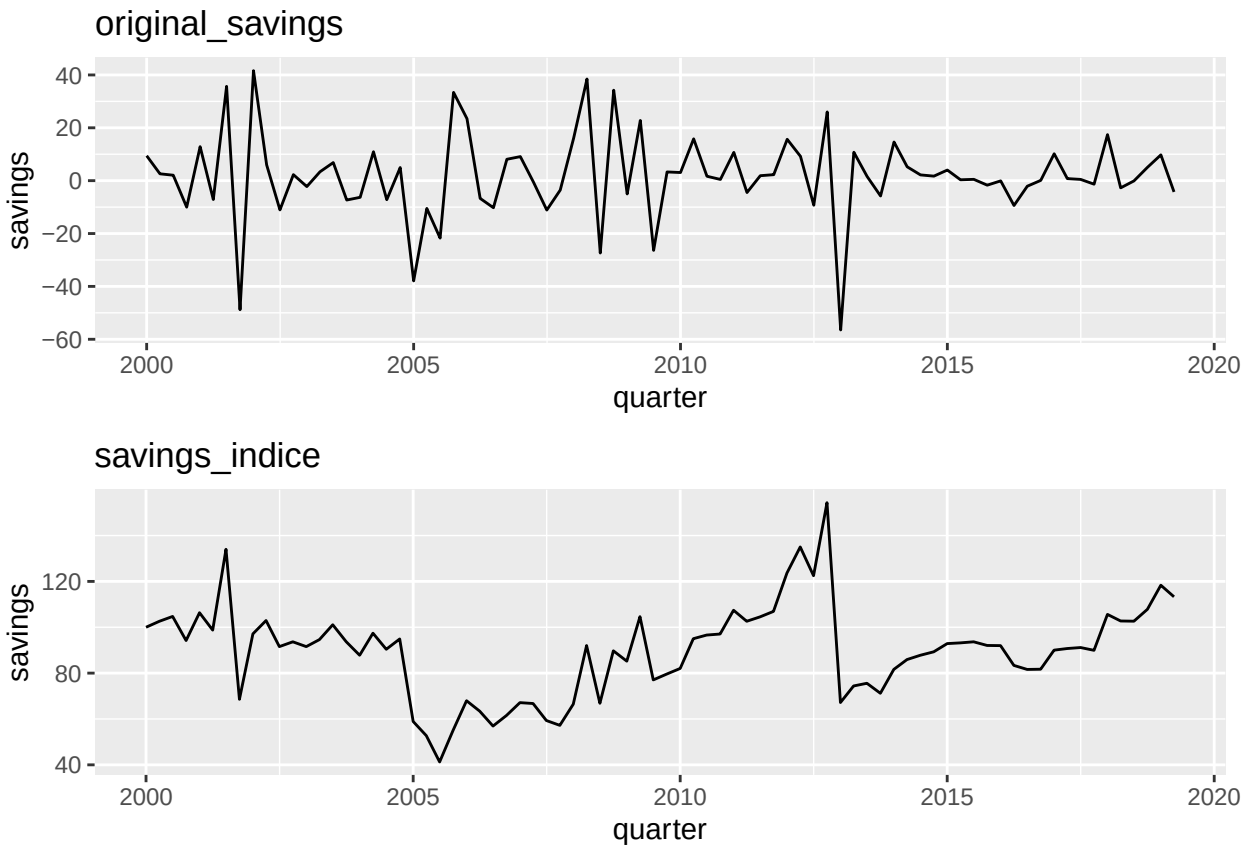




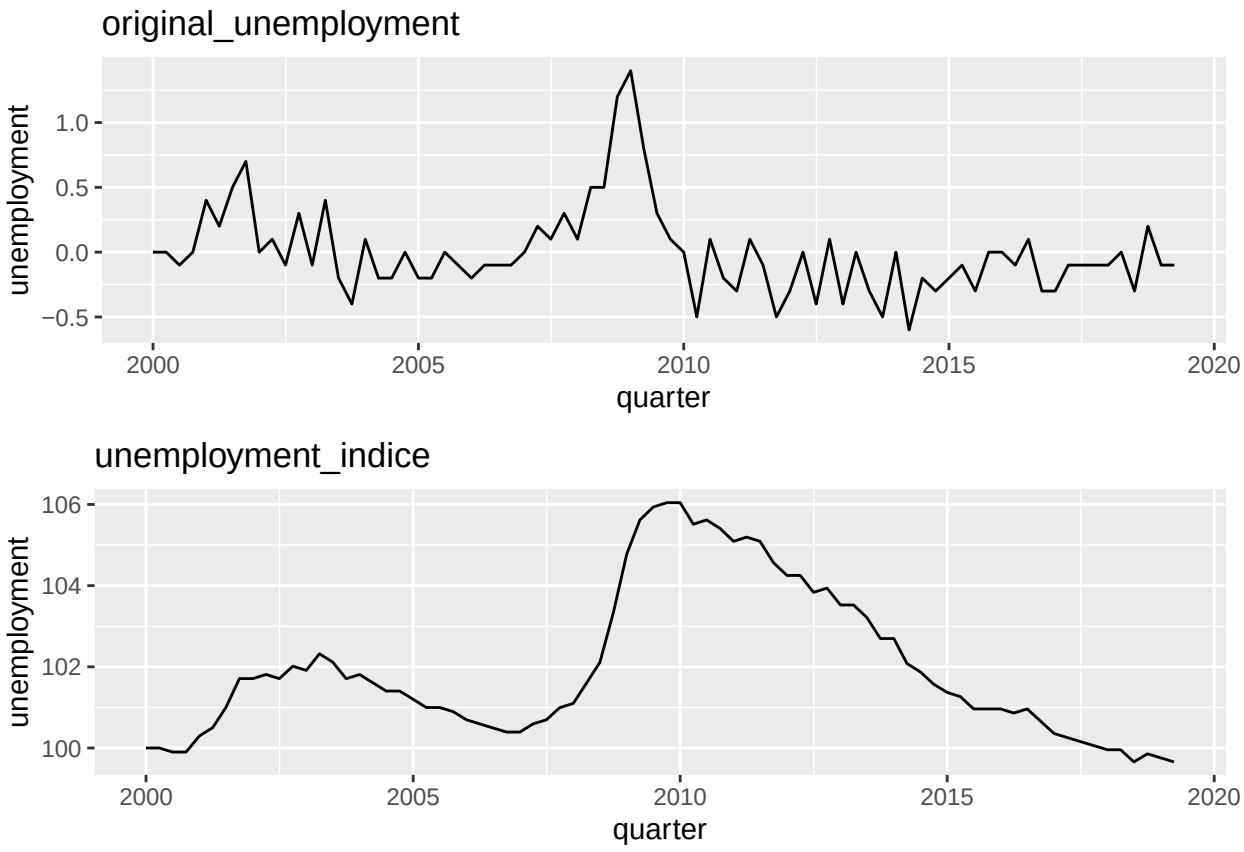

```
multiplot(plot5,plot6)
```




```
multiplot(plot7,plot8)
```

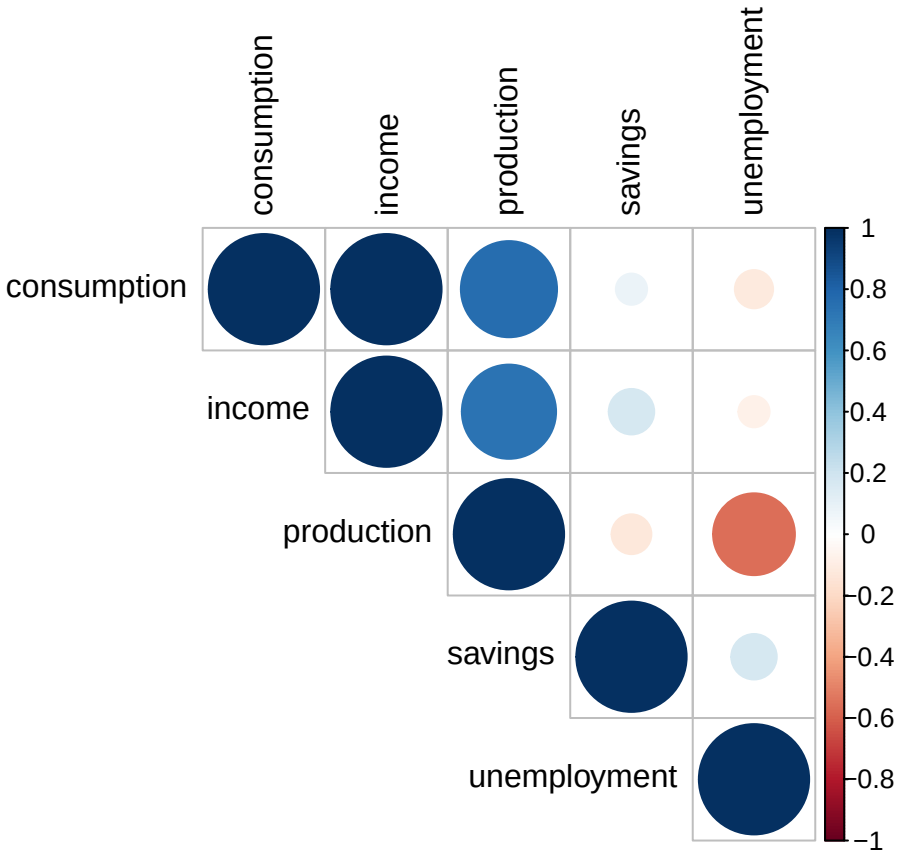



```
multiplot(plot9,plot10)
```



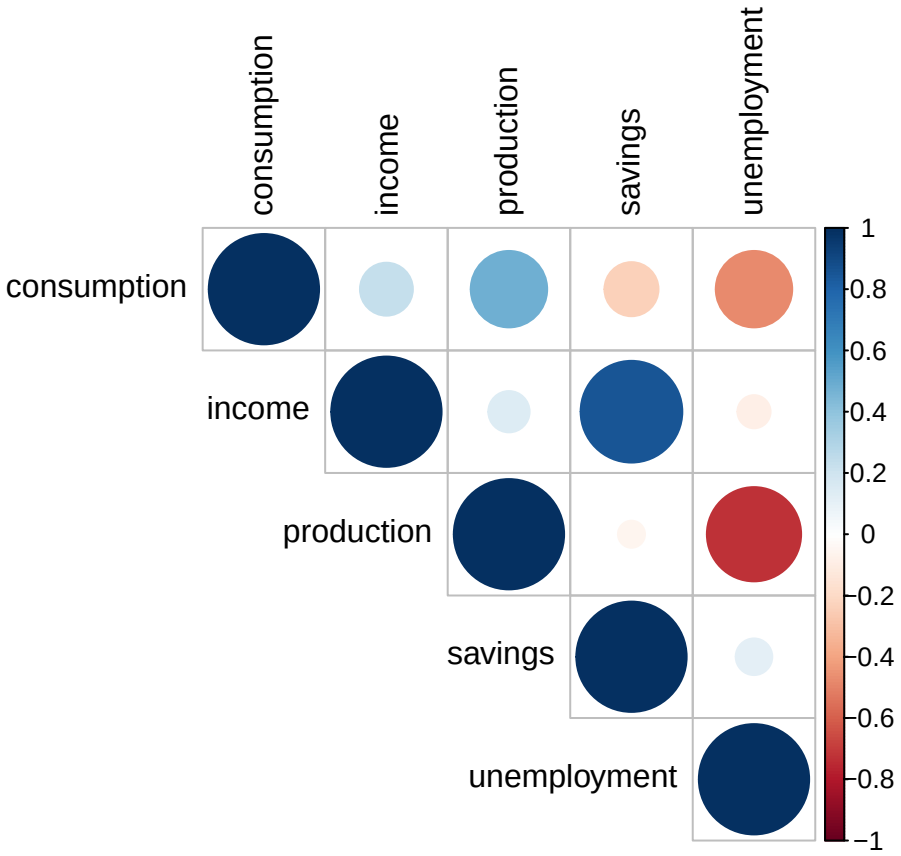
(B) Correlação com indice 100 para o primeiro trimestre do ano 2000

```
correl <-  cor(indice_df %>%
      select(-"quarter")) %>% round(2)

plot1 <- corrplot::corrplot(correl,
                    type = "upper",
                    tl.col = "black",
                    )
```



(B) Correlação da taxa de variação, sem índice mas filtrando o primeiro trimestre do ano 2000

```
correl2 <-  cor(data_nivel %>%
                  filter((quarter >= as.Date("2000-01-01"))) %>%
                  select(-"quarter")) %>% round(2)

plot2 <- corrplot::corrplot(correl2,
                   type = "upper",
                   tl.col = "black",
                   )
```



Conseguimos ver nesses gráficos de correlação a diferença entre usar índice ou apenas a taxa de variação para fazer a correlação. Usando um índice, você está normalizando os dados, fazendo com que a leitura da relação da informação fique padronizada. Sendo assim, os dois resultados são diferentes, pois estão em escalas difetentes.

(C) Plot de dispersão em linhas

```
new_data_nivel <- data_nivel %>%
  pivot_longer(-quarter) %>%
  select(date = quarter, name, value)

# Plotando gráfico da base de taxa de variação
p <- new_data_nivel %>%
  #filter( name %in% c("pmc", "pim_nivel")) %>%
  ggplot( aes(x = date, y = round(value,3), color = name)) +
  geom_line() +
  theme_bw()
ggplotly(p)
```

```
## TypeError: Attempting to change the setter of an unconfigurable property.
## TypeError: Attempting to change the setter of an unconfigurable property.
```

```r
# Plotando gráfico da base com o índice
new_indice_df <- indice_df %>%
  pivot_longer(-quarter) %>%
  select(date = quarter, name, value)

p <- new_indice_df %>%
  #filter( name %in% c("pmc", "pim_nivel")) %>%
  ggplot( aes(x = date, y = round(value,3), color = name)) +
  geom_line() +
  theme_bw()
ggplotly(p)
```

```
## TypeError: Attempting to change the setter of an unconfigurable property.
## TypeError: Attempting to change the setter of an unconfigurable property.
```

(D) É possível verificar que com os dados normalizados, o resultado final faz mais sentido. E é possível ver as relações entre as variáveis no gráfico de linhas (ao longo do tempo), pois é possível além de confirmar a correlação entre eles, mas saber o momento em que alguma mudança ocorreu. Ex: No gráfico com a base de indice, conforme a taxa de desemprego caia a taxa de crescimento da produção aumentava; O consumo e a renda se acompanham, conforme a renda aumenta o consumo também aumenta.

(E) É possível visualizar alguns movimentos atípicos em alguns momentos da base de dados. A linha de "poupança", não acompanha um padrão, sendo difícil fazer uma relação direta com outras variáveis.

Séries de tempo, ciclo, sazonalidade e tendência ("retail.xlsx")

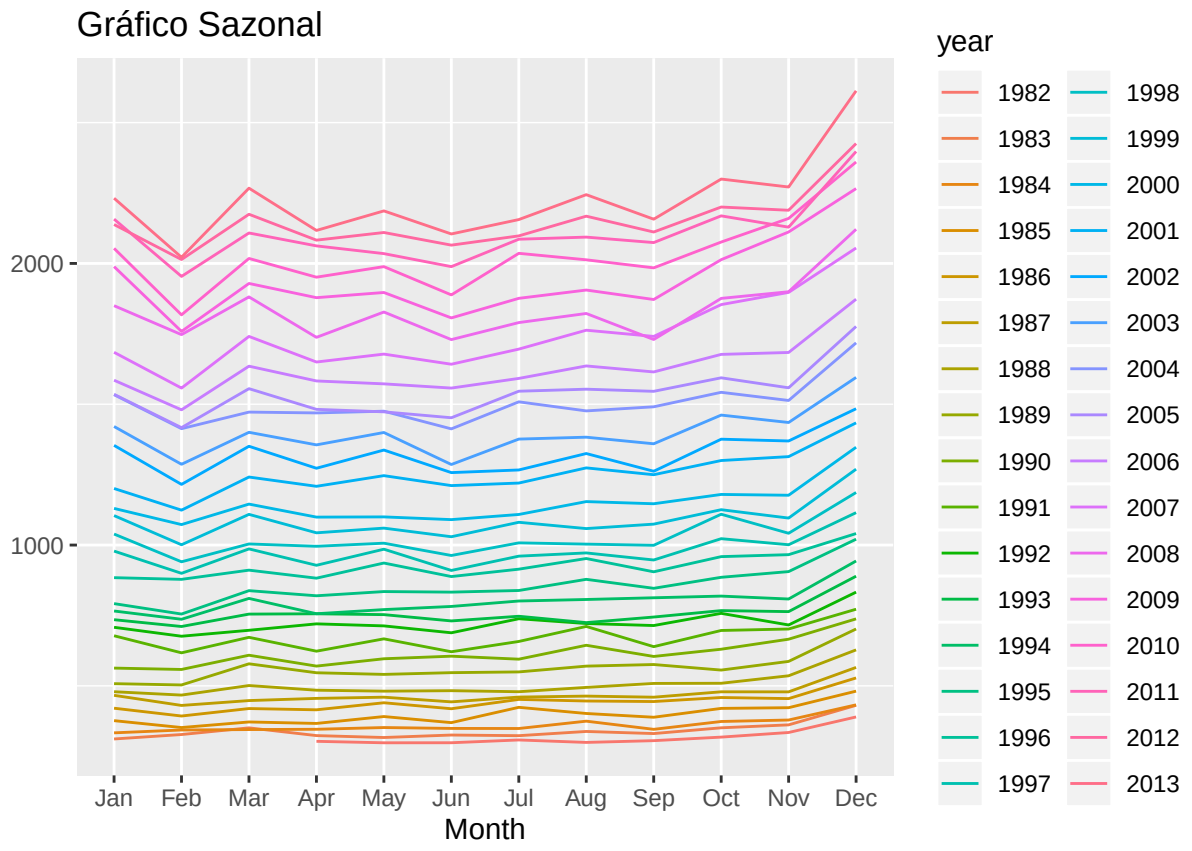
```
# lendo a base de dados, selecioando um range para a base, excluido o cabeçalho do arquivo, limpando os
data <- read_excel("retail.xlsx", range = "A2:GH383") %>%
  janitor::clean_names() %>%
  dplyr::rename(date = colnames(.)[1]) %>%
  mutate(date = as_date(date))

# É necessário não selecionar o cabeçalho do arquivo, pois quando se pega o cabeçalho os dados ficam bu
```

Criando um novo dataframe e o transformando em time series

```
df_ts <- data$a3349335t%>%
  stats::ts(
    start = c(
      lubridate::year(dplyr::first(data$date)),
      lubridate::month(dplyr::first(data$date))
    ),
    end = c(
      lubridate::year(dplyr::last(data$date)),
      lubridate::month(dplyr::last(data$date))
    ),
    frequency = 12
  )
```

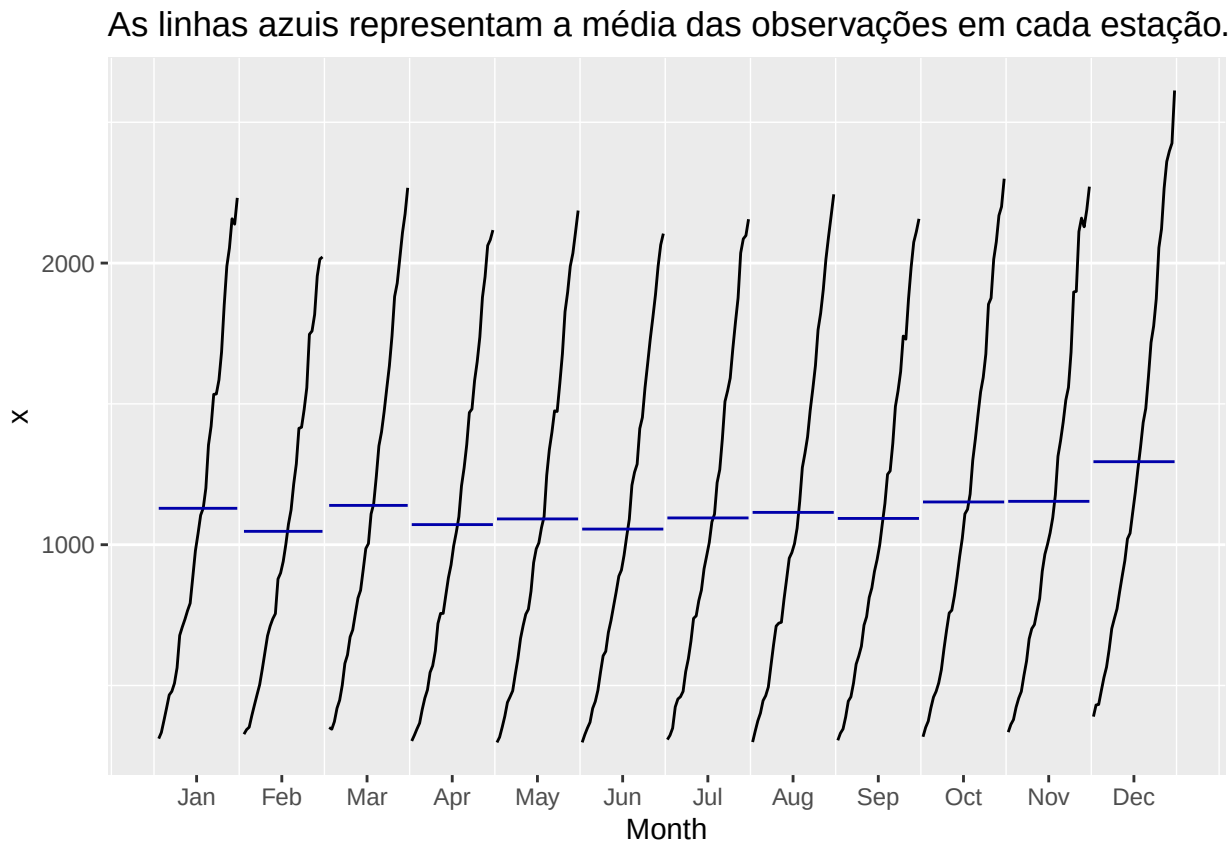
Plotando a série temporal em ggseasonplot()

```
plot_sazonal <- df_ts %>%
  forecast::ggseasonplot() +
  labs(title = "Gráfico Sazonal")
plot_sazonal
```



Plotando a série temporal em ggmonthplot()

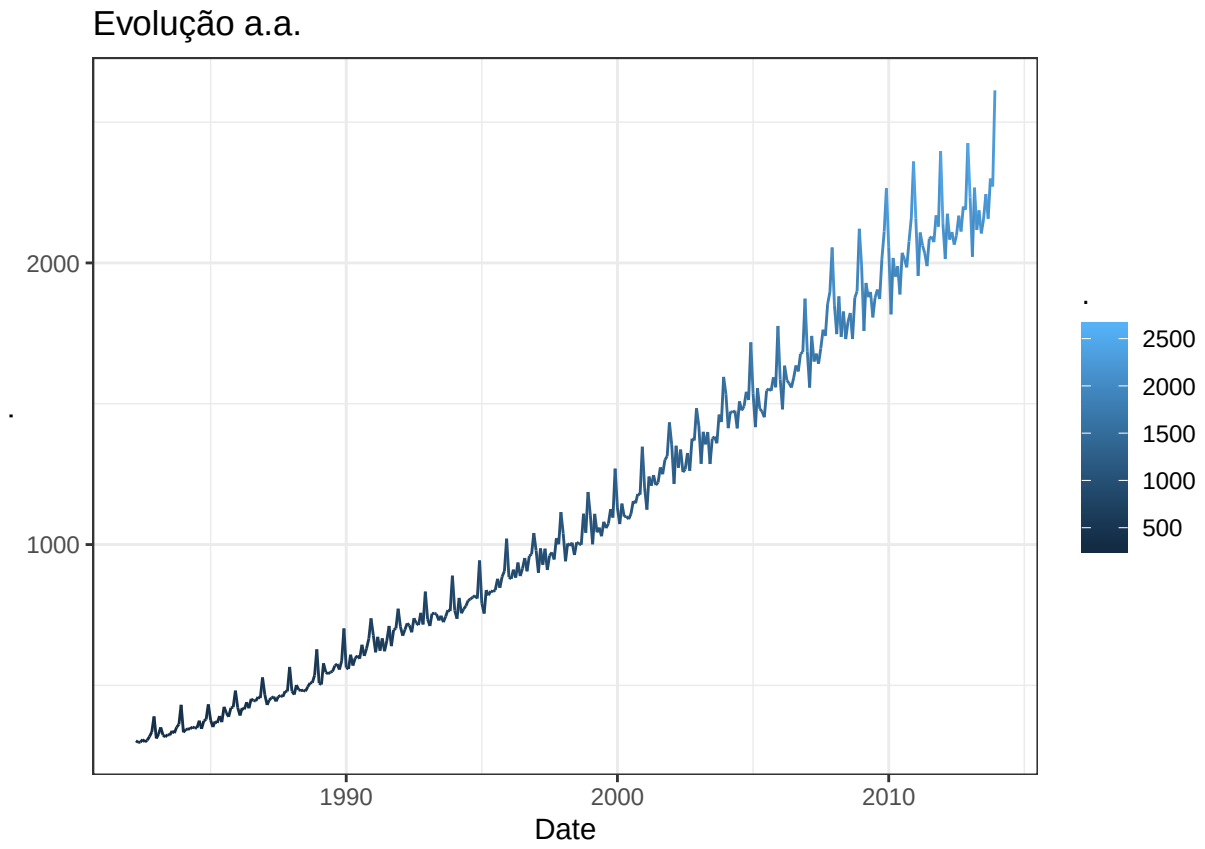
```
plot_month <- df_ts %>%
  forecast::ggmonthplot() +
  labs(title = "As linhas azuis representam a média das observações em cada estação.")
plot_month
```



Plotando a série temporal em geom_line()

```
plot_line <- df_ts %>%
  ggplot(aes(x = data$date, y = ., color = .)) +
  geom_line() +
  theme_bw()+
  labs(x = "Date",
       title = "Evolução a.a.")
plot_line
```

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



A série possui grande sazonalidade , poucos ciclos e tendencia de crescimento linear

Dataset Spotify

```
# Fazendo a leitura do dataframe e salvando em um .rds
# spotify_songs <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/maste
#saveRDS(spotify_songs, "df_spotify")


df_song <- readRDS("df_spotify")

# Filtrando base de dados por data, artista, nome da musica, popularidade da musica e genero musical
df_spotfy <- df_song %>%
  select(track_album_release_date,track_artist, track_name, track_popularity, playlist_genre) %>%
  as_tibble()

# Verificando quantos tipos de gêneros musicais existem na lista e seperando alguns dfs para plotagem
numero_generos <- df_spotfy %>% pull(playlist_genre) %>% unique()
numero_generos
```

```
## [1] "pop"   "rap"   "rock"  "latin" "r&b"   "edm"
```

```r
# Rap
df_rap <- df_spotfy %>%
  filter(playlist_genre == "rap") %>%
  mutate(ano = year(as_date(track_album_release_date))) %>%
  group_by(ano) %>%
  summarise(media = mean(track_popularity))

# Pop
df_pop <- df_spotfy %>%
  filter(playlist_genre == "pop") %>%
  mutate(ano = year(as_date(track_album_release_date))) %>%
  group_by(ano) %>%
  summarise(media = mean(track_popularity))

# Rock
df_rock <- df_spotfy %>%
  filter(playlist_genre == "rock") %>%
  mutate(ano = year(as_date(track_album_release_date))) %>%
  group_by(ano) %>%
  summarise(media = mean(track_popularity))

# Latin
df_latin <- df_spotfy %>%
  filter(playlist_genre == "latin") %>%
  mutate(ano = year(as_date(track_album_release_date))) %>%
  group_by(ano) %>%
  summarise(media = mean(track_popularity))

# R&B
df_r_b <- df_spotfy %>%
  filter(playlist_genre == "r&b") %>%
  mutate(ano = year(as_date(track_album_release_date))) %>%
  group_by(ano) %>%
  summarise(media = mean(track_popularity))

# EDM
df_edm <- df_spotfy %>%
  filter(playlist_genre == "edm") %>%
  mutate(ano = year(as_date(track_album_release_date))) %>%
  group_by(ano) %>%
  summarise(media = mean(track_popularity))

# Gráfico comparativo da evolução dos generos musicais ao longo dos anos
p <- plot_ly(df_rap, x = ~ano, y = ~media, type = 'scatter', mode = "lines+marker", name = "RAP") %>%
    add_trace(data = df_pop, x = ~ano, y = ~round(media,2), name = "POP") %>%
    add_trace(data = df_rock, x = ~ano, y = ~round(media,2), name = "ROCK") %>%
    add_trace(data = df_latin, x = ~ano, y = ~round(media,2), name = "LATIN") %>%
    add_trace(data = df_r_b, x = ~ano, y = ~round(media,2), name = "R&B") %>%
    add_trace(data = df_edm, x = ~ano, y = ~round(media,2), name = "EDM") %>%
      layout(title = "Evolução dos gostos musicais ao longo dos anos",
             xaxis = list(title ="Ano"),
             yaxis = list(title ="Média de popularidade a.a.")
             )
```

```
p
```

```
## TypeError: Attempting to change the setter of an unconfigurable property.
## TypeError: Attempting to change the setter of an unconfigurable property.
```

```
# df_spotfy %>%
#     ggplot( aes(x = track_album_release_date, y = track_popularity, color = track_artist)) +
#     geom_line()
```

É possível acompanhar o crescimento dos gêneros musicais através dos anos, e verificar qual se destacou mais em derterminada época.

Video Games Dataset

```
# Lendo e salvando base de dados de videogames
# video_games <- readr::read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/
#
# saveRDS(video_games, "videogames")]

df_video_game <- readRDS("videogames")

# Analisando a marca VALVE
```

```r
df_valve <- df_video_game %>%
  filter(developer == "Valve")

# Trocando NA para 0
df_valve$price[is.na(df_valve$price)] <- 0

df_game_plot <- df_valve %>% ggplot(aes(x = game, y = price, color = owners))+
  geom_point()+
  ggtitle("Relação do preço do jogo por quantidade de compras - Valve") +
  theme_bw()

ggplotly(df_game_plot)
```

```
## TypeError: Attempting to change the setter of an unconfigurable property.
## TypeError: Attempting to change the setter of an unconfigurable property.
```

```r
df_ubsoft <- df_video_game %>%
  filter(developer == "Ubisoft Montreal")

# Trocando NA para 0
df_ubsoft$price[is.na(df_ubsoft$price)] <- 0

df2_game_plot <- df_ubsoft %>% ggplot(aes(x = game, y = price, color = owners))+
  geom_point()+
```

```
  ggtitle("Relação do preço do jogo por quantidade de compras - Ubsoft") +
  theme_bw()

ggplotly(df2_game_plot)
```

```
## TypeError: Attempting to change the setter of an unconfigurable property.
## TypeError: Attempting to change the setter of an unconfigurable property.
```