R Notebook

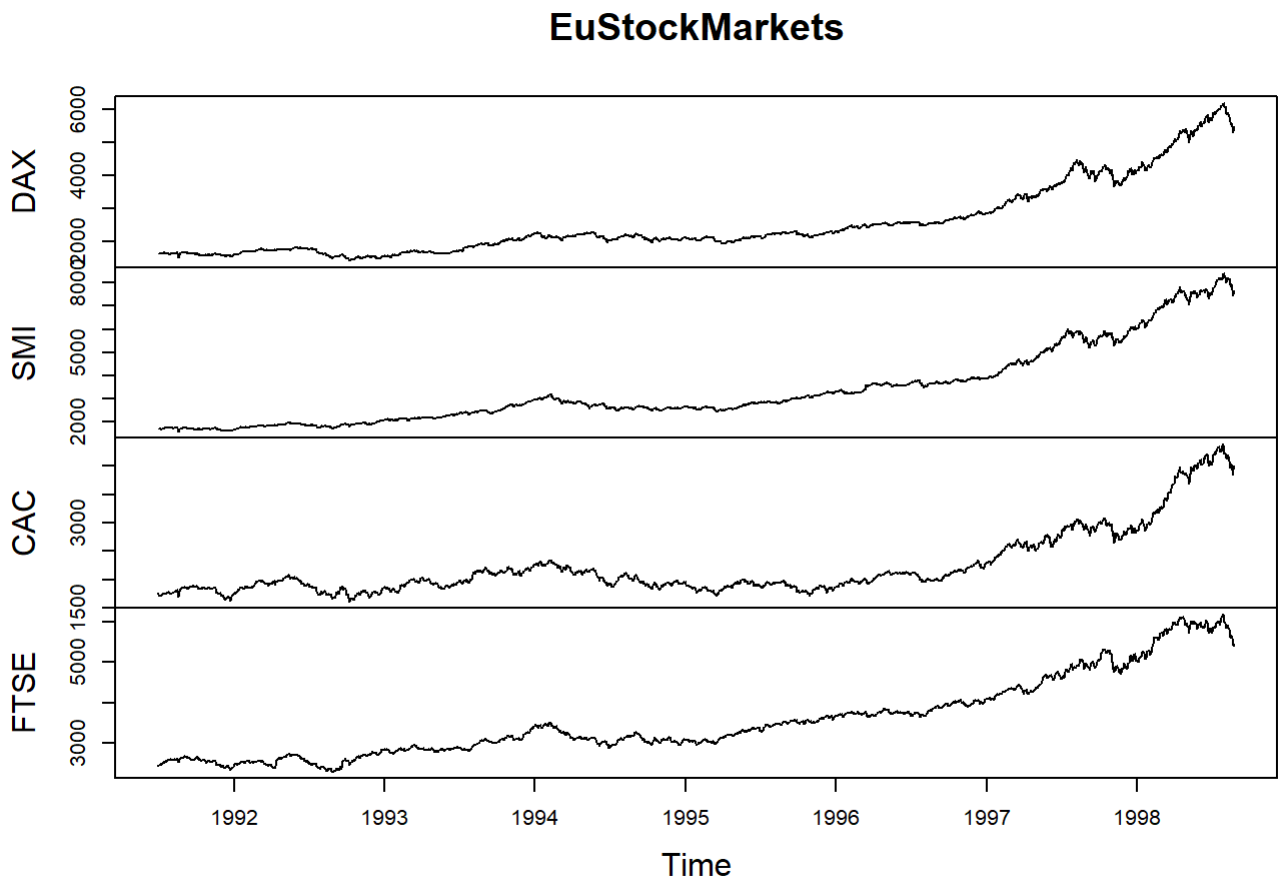
```
library(tseries)
```

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

```
head(EuStockMarkets, n = 5)
```

```
##          DAX    SMI    CAC   FTSE
## [1,] 1628.75 1678.1 1772.8 2443.6
## [2,] 1613.63 1688.5 1750.5 2460.2
## [3,] 1606.51 1678.6 1718.0 2448.2
## [4,] 1621.04 1684.1 1708.1 2470.4
## [5,] 1618.16 1686.6 1723.1 2484.7
```

```
plot(EuStockMarkets)
```




```
class(EuStockMarkets)
```

```
## [1] "mts"    "ts"     "matrix" "array"
```

> A classe mts (serie temporal multivariada) ts(serie temporal)

frequency : para descobrir a frequencia dos dados

```
frequency(EuStockMarkets)
```

```
## [1] 260
```

start e end : para mostrar o primeiro e o último tempo representado na série

```
start(EuStockMarkets)
```

```
## [1] 1991  130
```

```
end(EuStockMarkets)
```

```
## [1] 1998  169
```

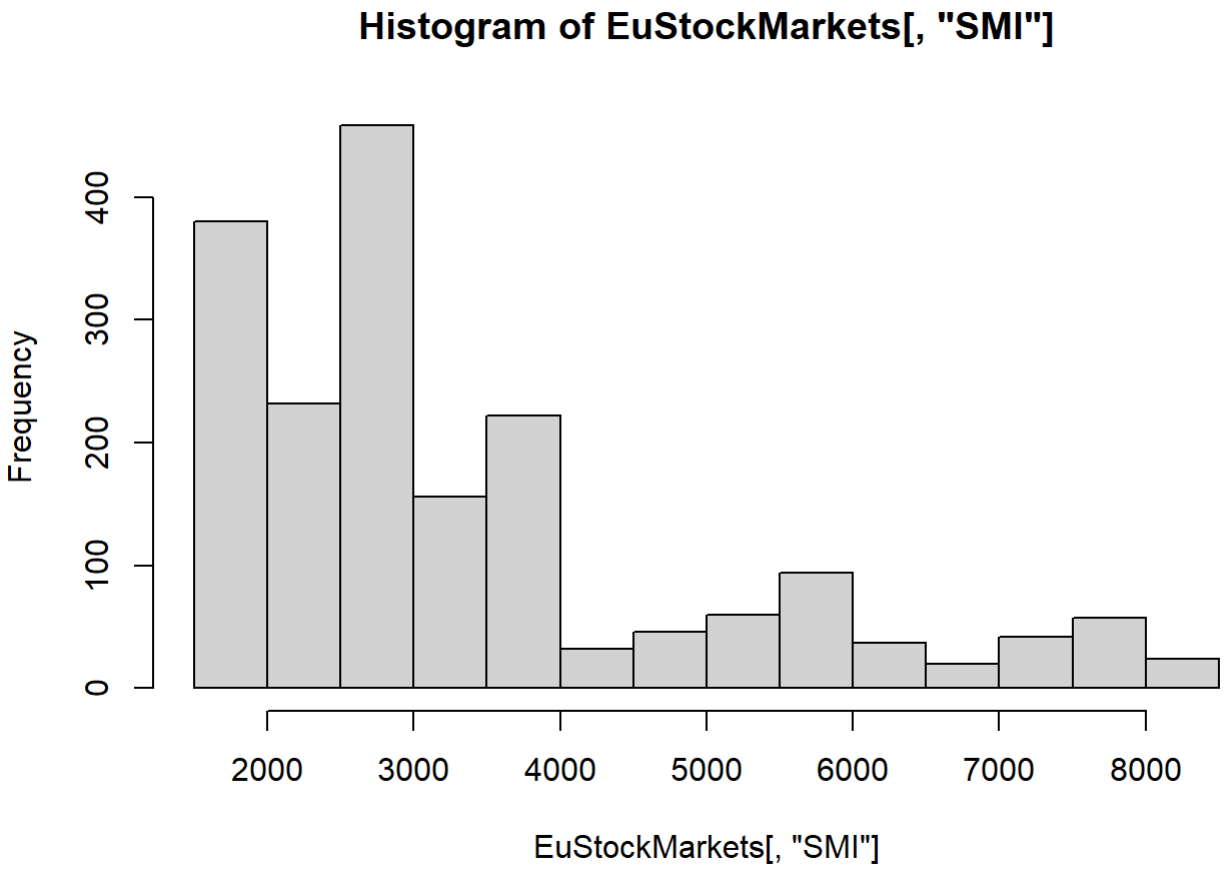
window : Obtem uma seção temporal

```
head( window(
  EuStockMarkets, start = 1996, end = 1998
), n = 4)
```

```
##          DAX    SMI    CAC   FTSE
## [1,] 2280.81 3277.9 1866.7 3658.3
## [2,] 2280.44 3317.1 1877.0 3676.4
## [3,] 2273.90 3297.7 1879.1 3676.7
## [4,] 2260.69 3297.7 1872.0 3689.3
```

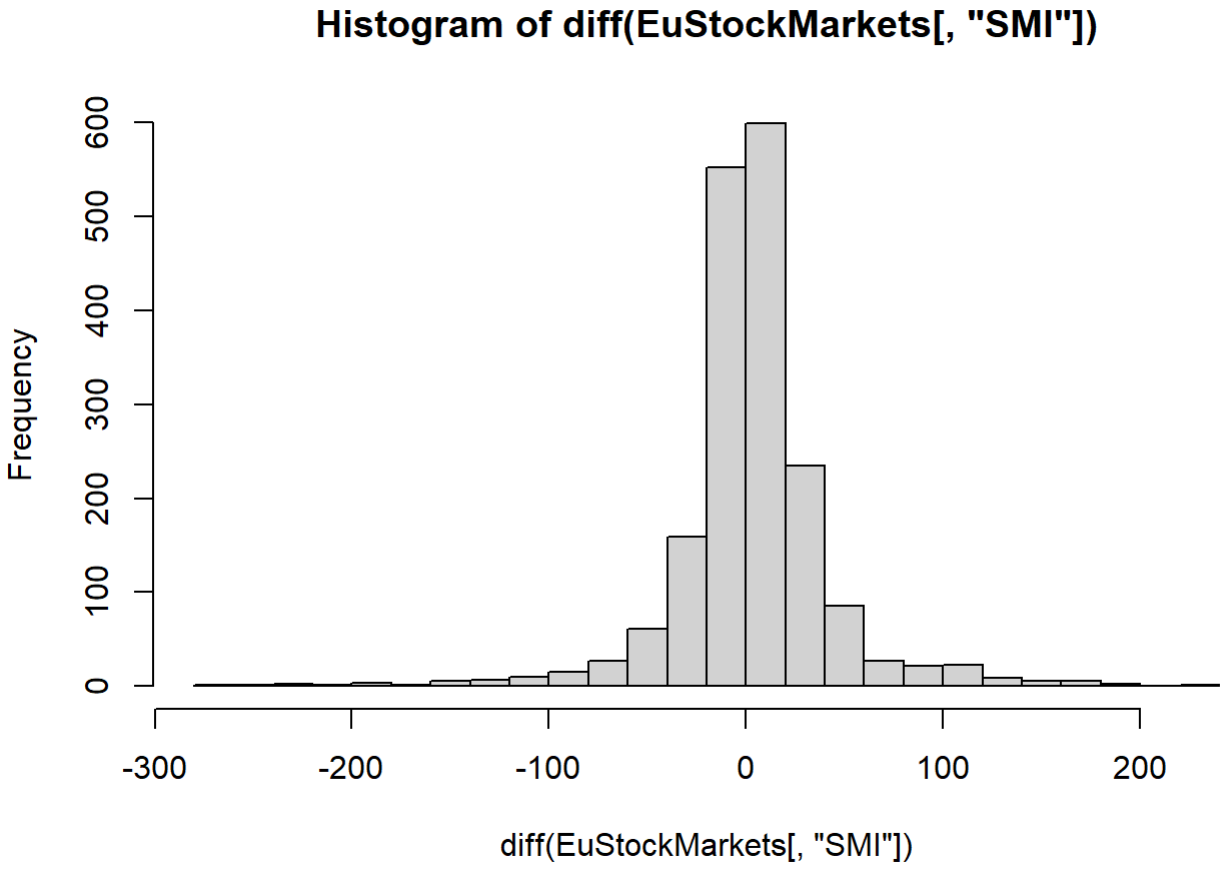
Histogramas

```
hist(EuStockMarkets[, "SMI"])
```



Histograma de dados não transformados é bastante amplo e não mostra uma distribuição normal

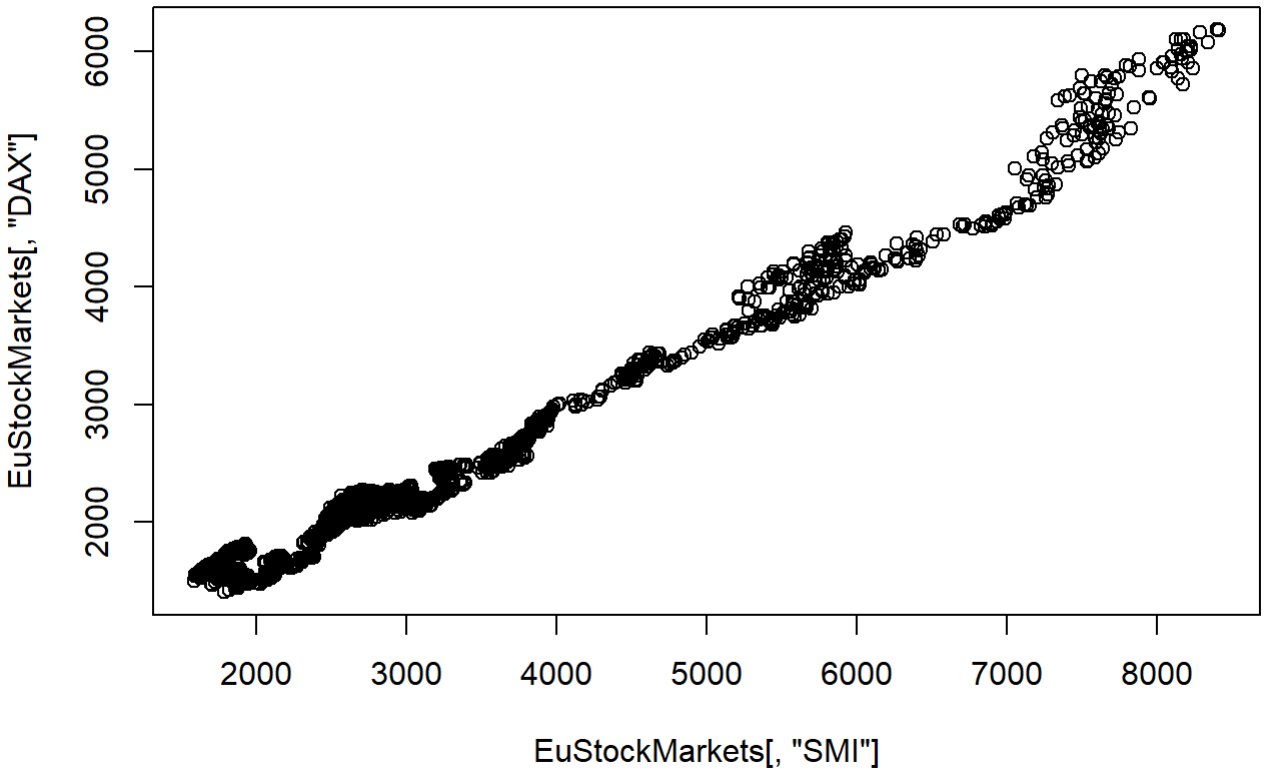
```
hist(diff(EuStockMarkets[, 'SMI']), 30)
```



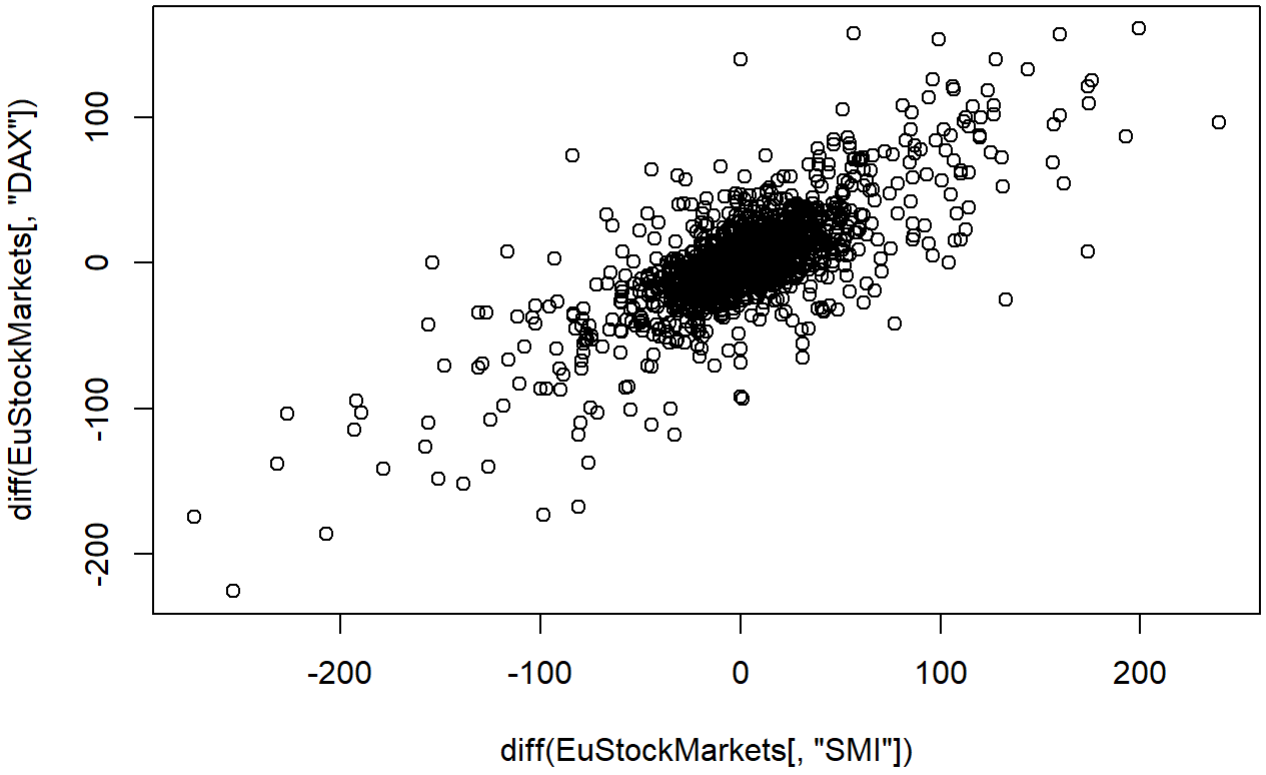
Dados transformados mostra um forma de dist. normal. O hist da diferença dos dados costuma ser mais interessante que um hist de dados não transformados

Dispersão

```
# Valores de duas ações ao longo do tempo
plot(EuStockMarkets[, "SMI"], EuStockMarkets[, "DAX"])
```




```
# Valores de mudanças diarias em relação a duas ações
#  ao longo do tempo
plot(diff(EuStockMarkets[, "SMI"]), diff(EuStockMarkets[, "DAX"]))
```



Apesar de mostrar uma forta correlação não podemos monetizar como trades atuantes

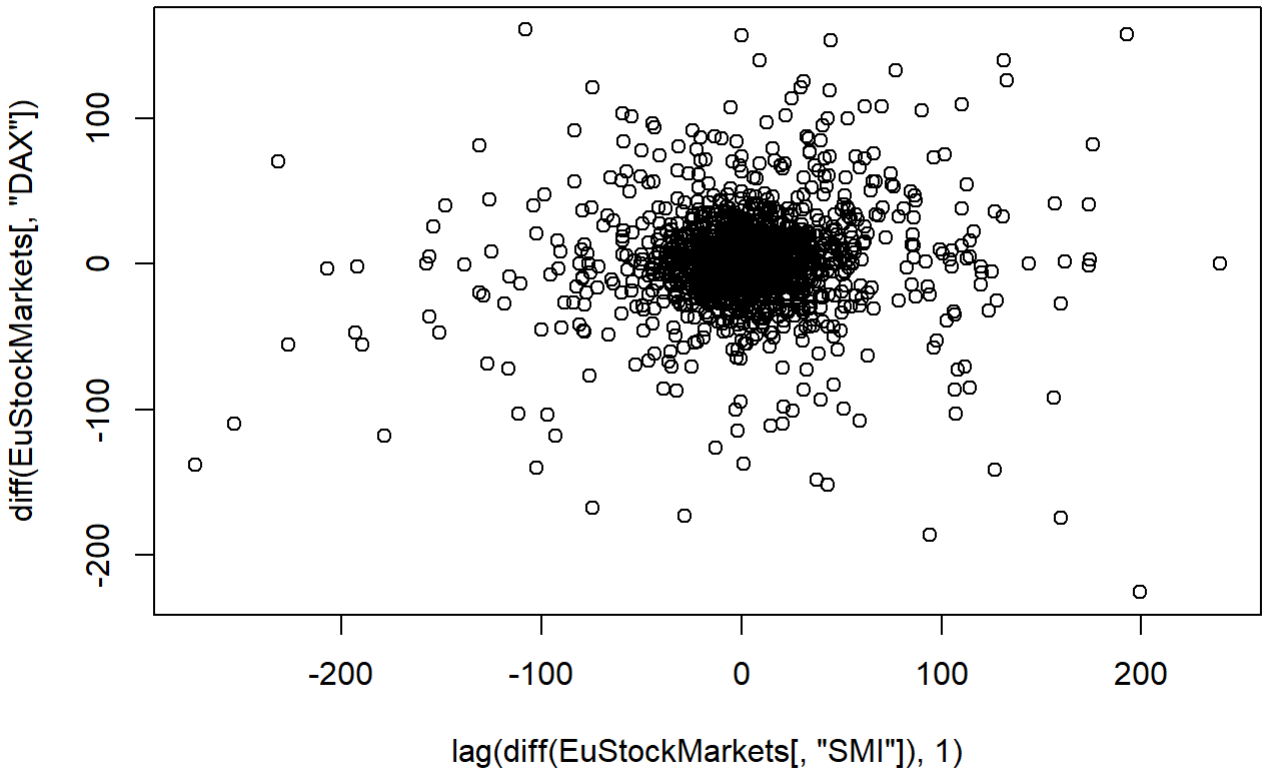
```
x <- ts(c(1, 2, 3, 4), start = 0, frequency = 1)
x
```

```
## Time Series:
## Start = 0
## End = 3
## Frequency = 1
## [1] 1 2 3 4
```

```
lag(x, 1)
```

```
## Time Series:
## Start = -1
## End = 2
## Frequency = 1
## [1] 1 2 3 4
```

```
plot(lag(diff(EuStockMarkets[, "SMI"]), 1), diff(EuStockMarkets[, "DAX"]))
```




```
import pandas as pd
```

```
a = pd.Series([10, 20, 30, 40], index=[1, 2, 3, 4])
a
```

```
## 1    10
## 2    20
## 3    30
## 4    40
## dtype: int64
```

```
a.shift(1)
```

```
## 1     NaN
## 2    10.0
## 3    20.0
## 4    30.0
## dtype: float64
```

Exatamente! Tanto no R quanto no Pandas:

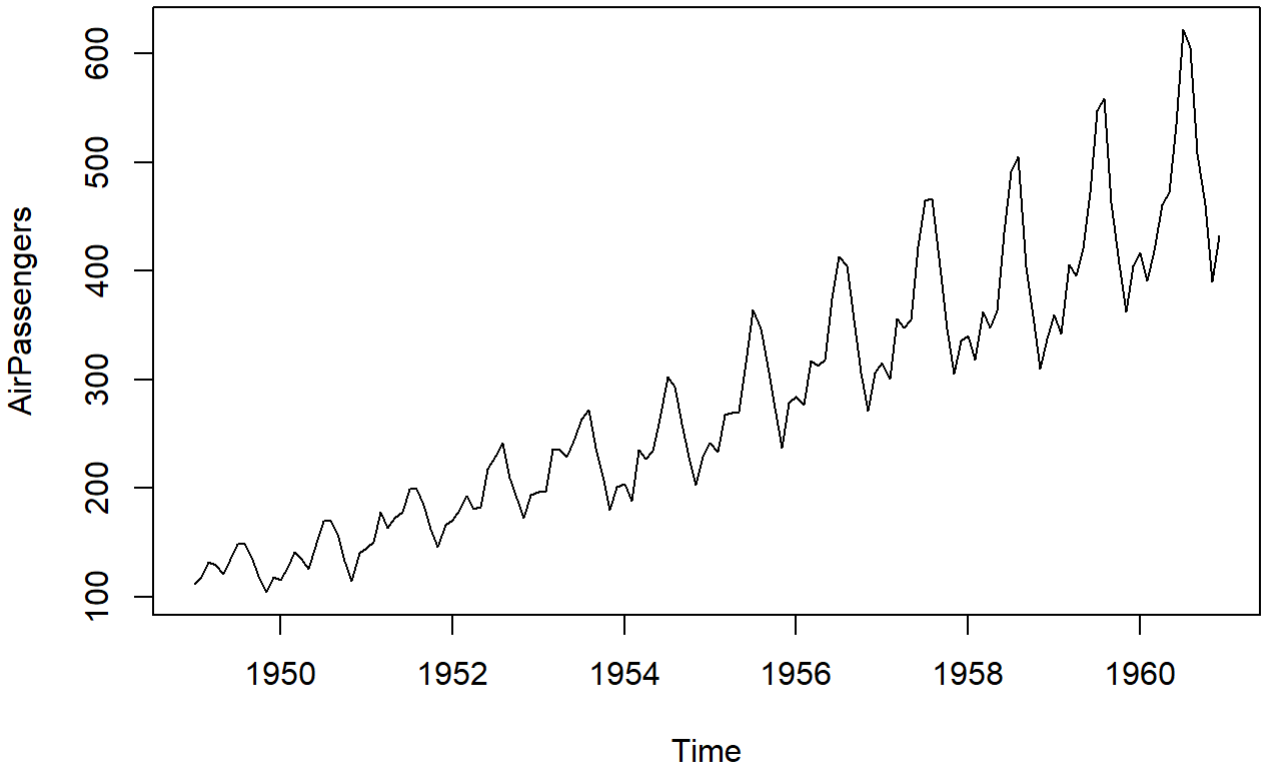
- **`lag` (R)** e **`shift` (Pandas)** deslocam os valores da série (ou coluna) no **eixo temporal**.
- Um deslocamento positivo, como `lag(x, 1)` ou `shift(1)`, **empurra os valores para frente no tempo**, o que na prática parece que estamos "olhando para trás" (ou seja, um **atraso**).
- Um deslocamento negativo, como `lag(x, -1)` ou `shift(-1)`, **puxa os valores para trás no tempo**, o que na prática parece que estamos "olhando para frente".

Métodos exploratórios específicos de séries temporais

Estacionariedade

A grosso modo uma série temporal estacionária é aquela que tem propriedades estatísticas razoavelmente estáveis ao longo do tempo. Sobre tudo no que diz respeito a **média e variancia.**

```
options(repr.plot.res = 300)
plot(AirPassengers)
```



- O grafico acima não é estacionaria, pois o valor médio está aumentando com tempo em vez de permanecer estável.
- A distancia entre as oscilações pico-vale em um base anual está crescendo, assim a variancia do processo está aumentando ao longo do tempo.
- Apresenta um comportamento sazonal

Teste de Dickey-fuller aumentado e KPSS

- **Hipótese nula (H0)**: A série **tem** uma raiz unitária, ou seja, **não é estacionária**.
- **Hipótese alternativa (H1)**: A série **não tem** uma raiz unitária, ou seja, **é estacionária**..

```
adf.test(AirPassengers)
```

```
## Warning in adf.test(AirPassengers): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  AirPassengers
## Dickey-Fuller = -7.3186, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

p_valor > -7.31 aceita H_0

Metodos KPSS

$Hipótese_{nula}(H0)$: A série é estacionária.

$Hipótese_{alternativa}(H1)$: A série não é estacionária

```
kpss.test(AirPassengers)
```

```
## Warning in kpss.test(AirPassengers): p-value smaller than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  AirPassengers
## KPSS Level = 2.7395, Truncation lag parameter = 4, p-value = 0.01
```

> p-valor < level rejeita H_0 então a serie não é estacionaria

> **Os teste não são remedios milagrosos para problema de estacionariedade**
>
> 1. Não conseguem distinguir muito as raizes unitarias
> 2. Falsos positivos para raizes unitarias

A estacionariedade é importante por uma série de motivos :

1. Grande número de modelos assume um processo estacionário
2. Um modelos de series temporal não estacionaria sofrerá variações em relação a sua acurácia ao mesmo tempo que as métricas da séria temporal variam.
3. A estacionariedade não é a unica suposição que os modelos de previsão fazem. Outra suposição é a normalização da distribuição das variaveis de entrada ou variavel preditiva.

Funçoes de janelas

```
# cacula a media movel usando a base R
set.seed(seed = 1)
x = rnorm(n = 100, mean = 0, sd = 10) + 1:100
head(x, 10)
```

```
##  [1] -5.264538  3.836433 -5.356286 19.952808  8.295078 -2.204684 11.874291
##  [8] 15.383247 14.757814  6.946116
```

```
# funcao filtro
mn =function(n) rep(1/n, n)
```

```
b =c(1, 2, 3, 4)

mean(b)
```

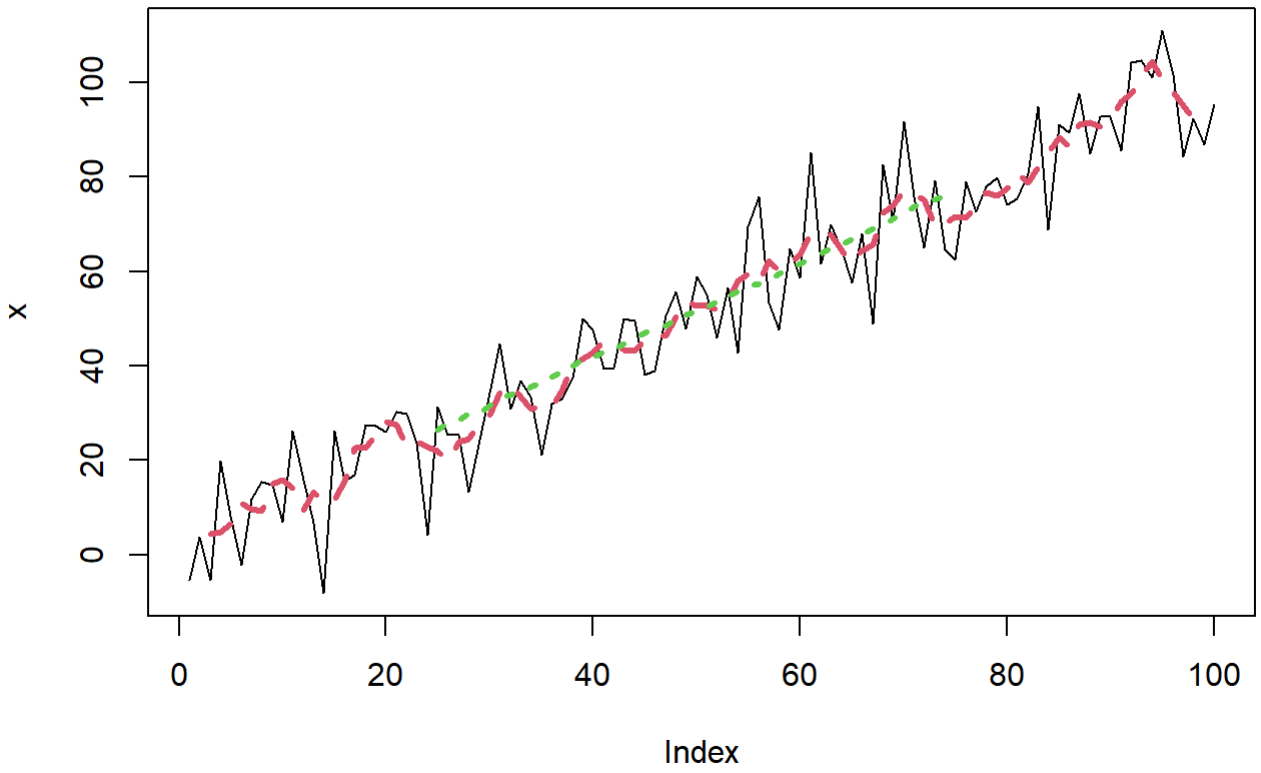
```
## [1] 2.5
```

```
filter(b, mn(2))
```

```
## Time Series:
## Start = 1
## End = 4
## Frequency = 1
## [1] 1.5 2.5 3.5  NA
```

```
plot(x, type='l', lwd = 1)

lines(filter(x, mn(5)), col = 2, lwd = 3, lty = 2)
lines(filter(x, mn(50)), col = 3, lwd = 3, lty = 3)
```




```
require(zoo)
```

```
## Carregando pacotes exigidos: zoo
```

```
##
## Anexando pacote: 'zoo'
```

```
## Os seguintes objetos são mascarados por 'package:base':
##
##     as.Date, as.Date.numeric
```

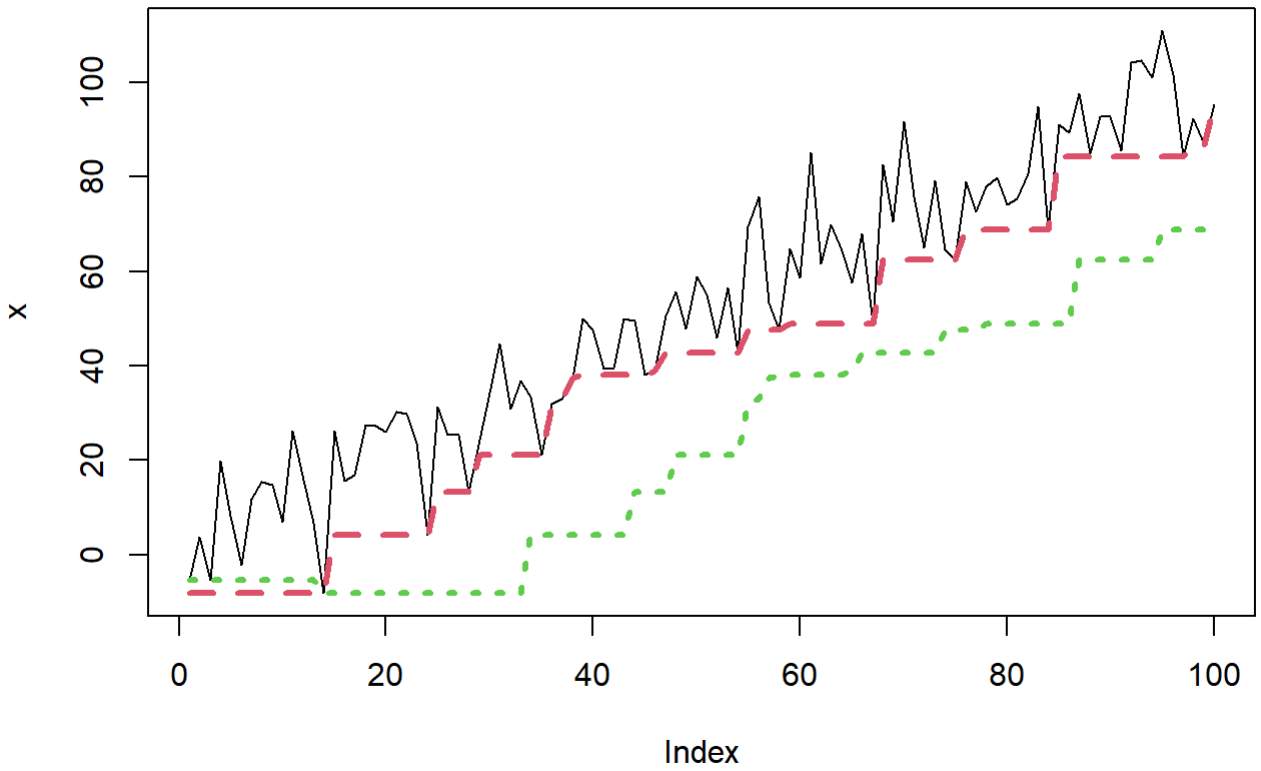
```
# Calculando a janela deslizante com alinhamento à esquerda
f1_ = rollapply(
  zoo(x), 20, function(w) min(w), align = "left", partial = TRUE
)

# Calculando a janela deslizante com alinhamento à direita
f2_ = rollapply(
  zoo(x), 20, function(w) min(w), align = "right", partial = TRUE
)
```

```
# Plotando a série original
plot(x, type="l", lwd = 1)

# Adicionando a linha de f1_ (mínimos com alinhamento à esquerda)
lines(f1_, col = 2, lwd = 3, lty = 2)

# Adicionando a linha de f2_ (mínimos com alinhamento à direita)
lines(f2_, col = 3, lwd = 3, lty = 3)
```



Janelas de espansão

para series temporais estácionarias

```
cummax(b)
```

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

```
cumsum(b)
```

```
## [1]  1  3  6 10
```

```
# Plotando a série original
plot(x, type="l", lwd = 1)


lines(cummax(x), col = 2, lwd = 3, lty = 2)


lines(cumsum(x)/1:length(x), col = 3, lwd = 3, lty = 3)
```