Limpeza de Dados para Séries Temporaias

```
# Importando libs
require(zoo, quietly = TRUE)
require(data.table, quietly = TRUE)
require(forecast, quietly = TRUE)
```

Limpe seu dados

Lidando com dados ausentes

Os dados ausentes são ainda mais comuns na análise de séries temporais do que na análise de dados transversais, por que a carga da amostragem longitudinaç é bastante pesada:

- Séries temporais incompletas são batantes comuns

Métodos para lidar com dados ausentes mais comuns são:

1. imputação
2. interpolação
3. Exclusão dos periodos de tempos afetados

Preparando um conj. dados para testar metodologias de imputação

```
path = 'C:\\Users\\mateu\\Documents\\MEGA\\Projetos-git\\analise-pratica-series-temporais_predicao-stats

unemp = fread(input = path, sep = ',')

head(x = unemp, n = 5)
```

```
##          DATE UNRATE
##        <IDat>  <num>
## 1: 1948-01-01    3.4
## 2: 1948-02-01    3.8
## 3: 1948-03-01    4.0
## 4: 1948-04-01    3.9
## 5: 1948-05-01    3.5
```

```
# Convertendo para date
unemp[, DATE := as.Date(DATE)]
```

```r
# definindo o index
setkey(unemp, DATE)
```

```r
head(x = unemp, n = 4)
```

```
## Key: <DATE>
##          DATE UNRATE
##        <Date>  <num>
## 1: 1948-01-01    3.4
## 2: 1948-02-01    3.8
## 3: 1948-03-01    4.0
## 4: 1948-04-01    3.9
```

Gerando um conj. de dados onde os dados estão aleatoriamente ausentes.

```r
set.seed(10)
rand.unemp.idx = sample(1:nrow(unemp), .1*nrow(unemp))

rand.unemp = unemp[-rand.unemp.idx]
```

Gerando um conj. de dados onde os dados possuem maior prob. de ausencia quando o desemprego é alto

```r
high.unemp.idx = which(unemp$DATE > 8)
num.to.select  = .2 * length(high.unemp.idx)

high.unemp.idx = sample(high.unemp.idx, )
bias.unemp = unemp[-high.unemp.idx]
```

Como excluimos as linhas da nossa tabela de dados para criar um conj. de dados com dados ausentes, precisamos ler as datas ausentes e o valores NA.

```r
todas.datas = seq(
  from = unemp$DATE[1], to = tail(unemp$DATE, 1),
  by = "months"
)
todas.datas[1:4]
```

```
## [1] "1948-01-01" "1948-02-01" "1948-03-01" "1948-04-01"
```

```r
rand.unemp = rand.unemp[J(todas.datas), roll = 0]
bias.unemp = bias.unemp[J(todas.datas), roll = 0]
```

```r
rand.unemp[, rtp := is.na(UNRATE)]
```

```r
head(
  x = rand.unemp[rtp == TRUE], n = 3
)
```

```
##          DATE UNRATE    rtp
##        <Date>  <num> <lgcl>
## 1: 1949-01-01     NA   TRUE
## 2: 1950-02-01     NA   TRUE
## 3: 1950-09-01     NA   TRUE
```

Forward Fill É transferir o último valor conhecido para o valor ausente anterior.

```
na.locf(coluna, na.rm = FALSE)
```

Imputando no R :

```
# Para aleatorio
rand.unemp[, impute.ff := na.locf(UNRATE, na.rm = FALSE)]

# Para o com maior prob. de desemprego
bias.unemp[, impute.ff := na.locf(UNRATE, na.rm = FALSE)]
```

Plotando

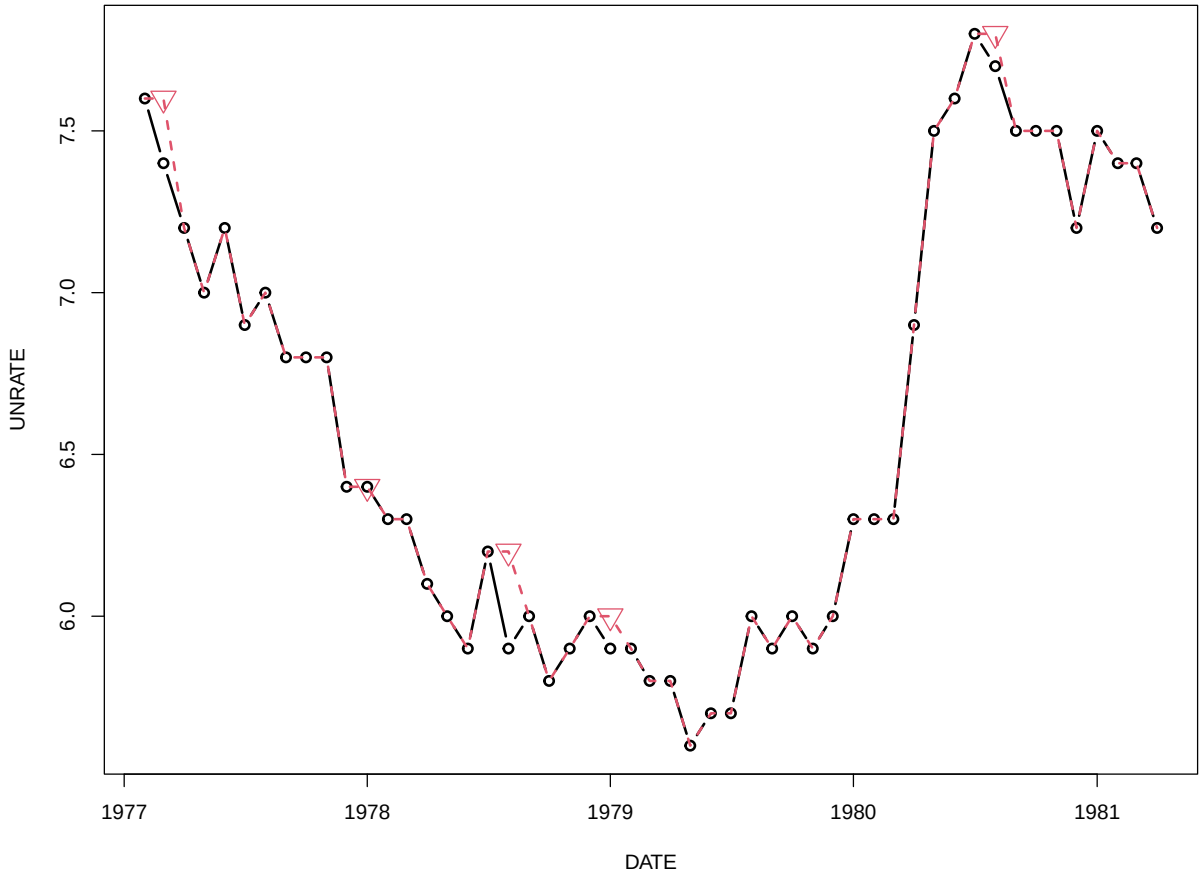
```
# plot de um gráfico de amostra que mostra as partes achatadas

unemp[
  350:400,
  plot(DATE, UNRATE, col = 1, lwd = 2, type = 'b')
]
```

```
## NULL
```

```
rand.unemp[
  350:400,
  lines(
    DATE, impute.ff, col = 2, lwd = 2, lty = 2
  )
]
```

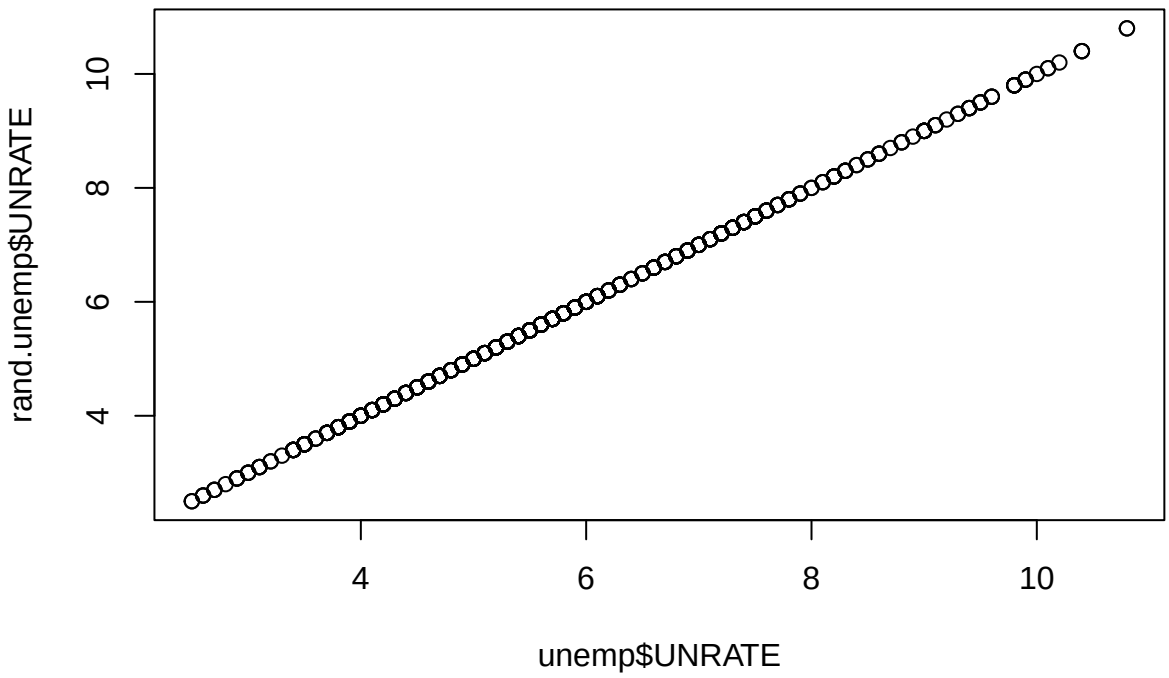
```
## NULL
```

```
rand.unemp[350:400][rtp == TRUE,
                    points(
                      DATE, impute.ff,
                      col = 2, pch = 6, cex = 2
                    )]
```




```
## NULL
```

```
plot(
  unemp$UNRATE, rand.unemp$UNRATE
)
```



Pode imputar os valor de trás para frente , contudo, se quiser treinar um modelo isso trata de um lookhead.

Em alguns forward fill é melhor maneira de preencher valores ausentes, mesmo que métodos "mais sofisticados" sejam possíveis.

- Não é exigente em termos computacionais, pode ser facilmente aplicados a dados em tempo real.

Média Móvel Semelhante ao forward fill mas imputa dados com uma média model ou mediana. Voce usa entradas provenientes de multiplos tempos recentes no passado.

Situaçõess em que a média móvel melhor se adéqua á tarefa em questão do que um forward fill :

1. Se os dados forem ruidosos e voce tem razões para duvidar do valor de qualquer ponto de dados individual em relação a um média geral, recomenda-se usar uma média móvel.
   - Forward Fill pode incluir mais ruídos aleatório do que a métrica "verdadeira" que lhe interessa. Média movel pode remover parte desse ruído.

```
rand.unemp[,
           impute.rm.nolookahead := rollapply(
             c(NA, NA, UNRATE), 3,
                function(x){
                  if (!is.na(x[3])) x[3] else mean(x, na.rm = TRUE)
                }
           )]
```

```
bias.unemp[,
           impute.rm.nolookahead := rollapply(
             c(NA, NA, UNRATE), 3,
                function(x){
                  if (!is.na(x[3])) x[3] else mean(x, na.rm = TRUE)
                }
           )]
```

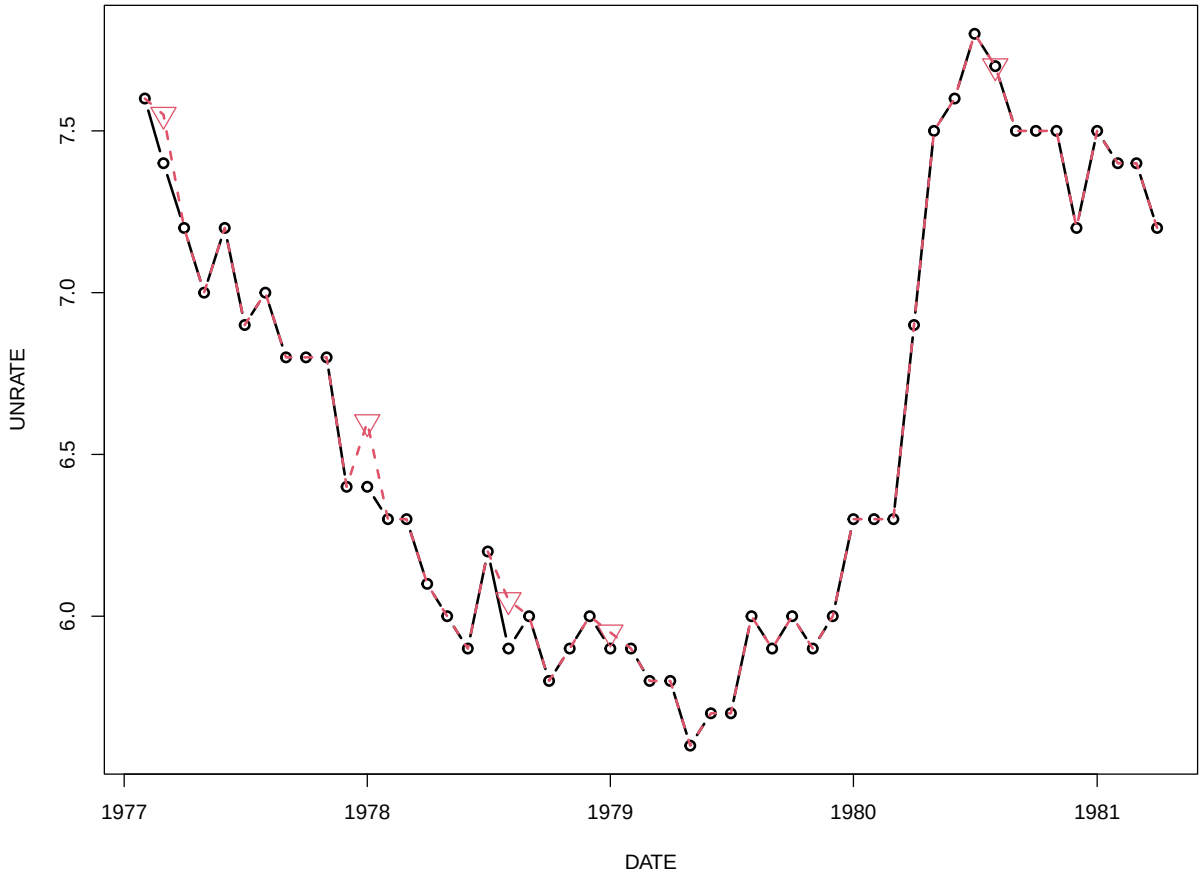
```
# plot de um gráfico de amostra que mostra as partes achatadas

unemp[
  350:400,
  plot(DATE, UNRATE, col = 1, lwd = 2, type = 'b')
]
```

```
## NULL
```

```
rand.unemp[
  350:400,
  lines(
    DATE, impute.rm.nolookahead, col = 2, lwd = 2, lty = 2
  )
]
```

```
## NULL
```

```
rand.unemp[350:400][rtp == TRUE,
                    points(
                      DATE, impute.rm.nolookahead,
                      col = 2, pch = 6, cex = 2
                    )]
```




```
## NULL
```

Exemplo real do rollapply:

```
a = rollapply( c(NA, NA, c(1 , 2, NA, NA, 1, 2, 2, NA, 10) ), 3,
                function(x){
                  if (!is.na(x[3])) x[3] else mean(x, na.rm = TRUE) }
           )
a
```

```
## [1]  1.0  2.0  1.5  2.0  1.0  2.0  2.0  2.0 10.0
```

Retornando....

Caso não esteja preocupado em fornecer seus dados a um modelo e se sente a vontade de construir um lookahead. Pode usar a media movel incluindo dados do passado e futuro.

```
rand.unemp[,
           complete.rm := rollapply(
             c(NA, UNRATE, NA), 3,
             function(x){
```

```
              if (!is.na(x[2])) x[2]
              else mean(x, na.rm = TRUE)
            }
          )
          ]
```

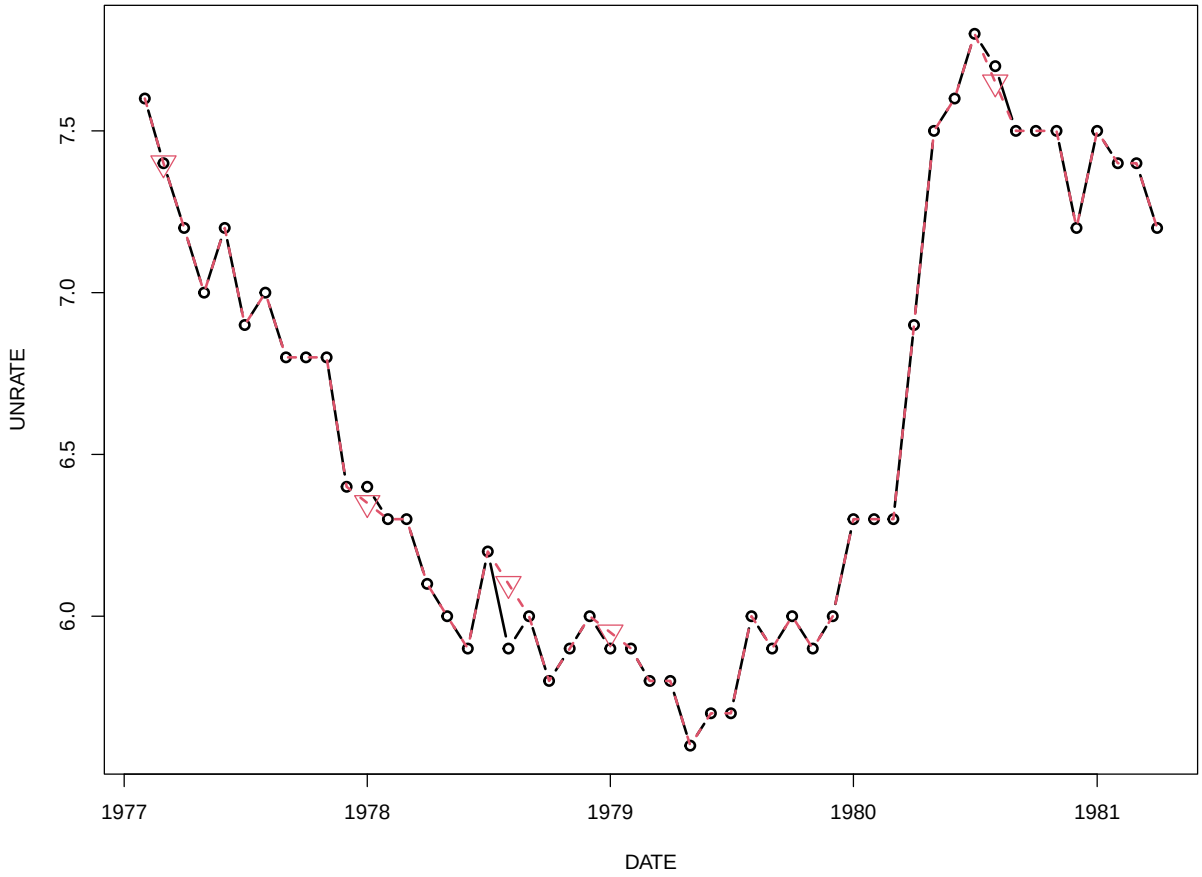
```
# plot de um gráfico de amostra que mostra as partes achatadas

unemp[
  350:400,
  plot(DATE, UNRATE, col = 1, lwd = 2, type = 'b')
]
```

```
## NULL
```

```
rand.unemp[
  350:400,
  lines(
    DATE, complete.rm, col = 2, lwd = 2, lty = 2
  )
]
```

```
## NULL
```

```
rand.unemp[350:400][rtp == TRUE,
                    points(
                      DATE, complete.rm,
                      col = 2, pch = 6, cex = 2
                    )]
```




```
## NULL
```

Usar a média movel, **reduz a variancia** no conj. de dados.

Interpolação Usa para determinar dados de valores ausentes com base em restrições geometricas sobre como queremos que os dados gerais se comportem.

Pode ser feito semelhante a média movel se quiser imputar valores com base em valores passados e futuros(Isso cria um lookahead)

```
rand.unemp[, impute.li.lookahead := NULL]
```

```
## Warning in `[.data.table`(rand.unemp, , `:=`(impute.li.lookahead, NULL)):
## Tentado atribuir NULL a coluna 'impute.li.lookahead', mas essa coluna não
## existe para remover
```

```
rand.unemp[, impute.li := NULL]
```

```
## Warning in `[.data.table`(rand.unemp, , `:=`(impute.li, NULL)): Tentado
## atribuir NULL a coluna 'impute.li', mas essa coluna não existe para remover
```

```
#interpolaçao linear com lookhead
rand.unemp[,impute.li.lookahead := na.approx(UNRATE, rule=2)]
bias.unemp[,impute.li.lookahead := na.approx(UNRATE, rule=2)]
```

```
#interpolaçao polinomial com lookhead
rand.unemp[,impute.sp.lookahead := na.spline(UNRATE)]
bias.unemp[,impute.sp.lookahead := na.spline(UNRATE)]
```

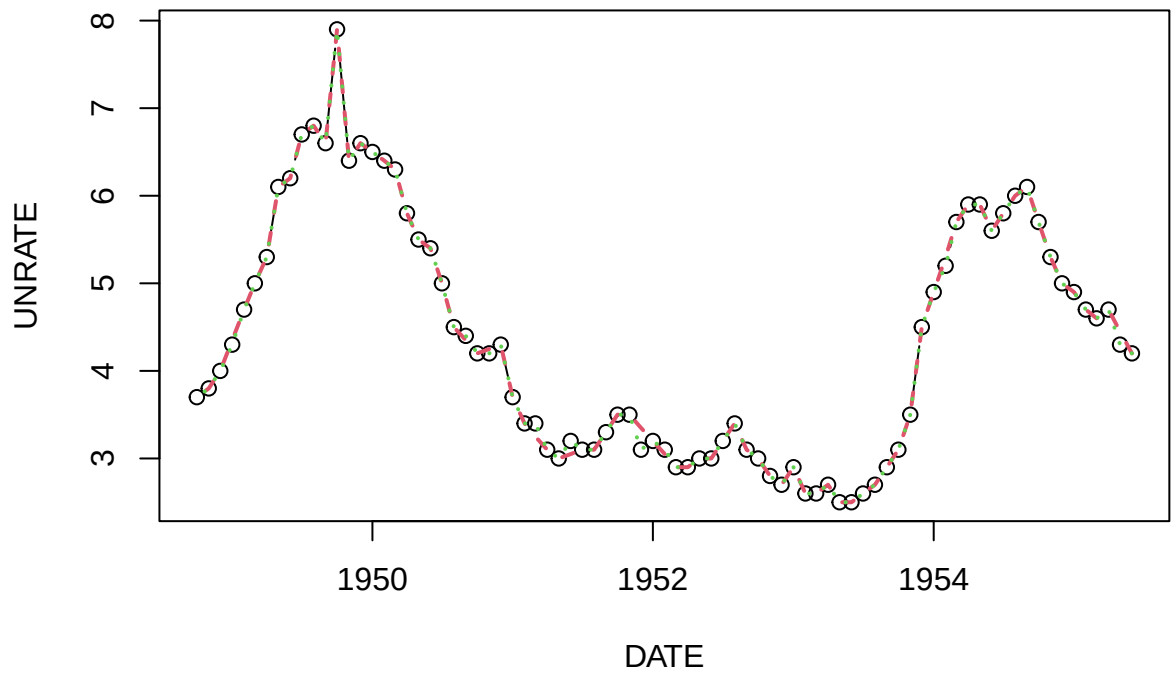
```
use.idx = 90:10
unemp[
  use.idx, plot(DATE, UNRATE, col = 1, type = 'b')
]
```

```
## NULL
```

```
rand.unemp[
  use.idx, lines(DATE, impute.li.lookahead, col = 2, lwd = 2, lty = 2)
]
```

```
## NULL
```

```
bias.unemp[
  use.idx, lines(DATE, impute.sp.lookahead, col = 3, lwd = 2, lty = 3)
]
```



```
## NULL
```

```
# sem lookhead utilizer o rollapply
na_approx_no_lookahead <- function(x) {
  for (i in seq_along(x)) {
    if (is.na(x[i])) {
      # Interpolação apenas até o índice atual
      x[1:i] <- na.approx(x[1:i], na.rm = FALSE)
    }
  }
  return(x)
}
```

Utilizar quando:

1. Há tendencias

Quando não utilizar

1. Quando não tem tendencias

```
rand.unemp[,
  lapply(.SD,
         function(x){
           return(mean((x - unemp$UNRATE)**2, na.rm = TRUE))
         }),
  .SDcols = c("impute.ff","impute.rm.nolookahead","complete.rm",   "impute.li.lookahead","impute.sp.look
]
```

Comparação

```
##      impute.ff impute.rm.nolookahead complete.rm impute.li.lookahead
##          <num>                 <num>       <num>               <num>
## 1: 0.003613744           0.004869514  0.00133452         0.001324381
##    impute.sp.lookahead
##                  <num>
## 1:          0.00239141
```

Suavização de dados

Não é raro antes de analisar, suavizar os dados. Sobre tudo quando se trata de visualizações que tem como objetivo contar um historia compreensivel sobre os dados.

Por que está suavizandos? a suavização pode ter muitas finalidades

1. Preparação dos dados
2. Geração de caracteristicas
3. Predição : a forma mais simples de predição para alguns tipos de processos é reversão á média, que voce obtem ao fazer a predição a partir de uma caracteristicas suavizadas
4. Visualização

Suavização exponencial Se quizer tratar os dados mais recentes como os dados mais informativos, nesse caso, a suavização exponencial é uma boa opção.

```
import pandas as pd
path = "C:\\Users\\mateu\\Documents\\MEGA\\Projetos-git\\analise-pratica-series-temporais_predicao-stats
data = pd.read_csv(path)
```

```
data.columns
```

```
## Index(['date', 'passengers'], dtype='object')
```

```
data["air.5"] = data["passengers"].ewm(alpha=0.5).mean()
data["air.9"] = data["passengers"].ewm(alpha=0.9).mean()
```

O parametro alpha é o fator de suavização, quanto mais alto o valor de alpha mais rapido é atualizado proximo ao seu preco atual.

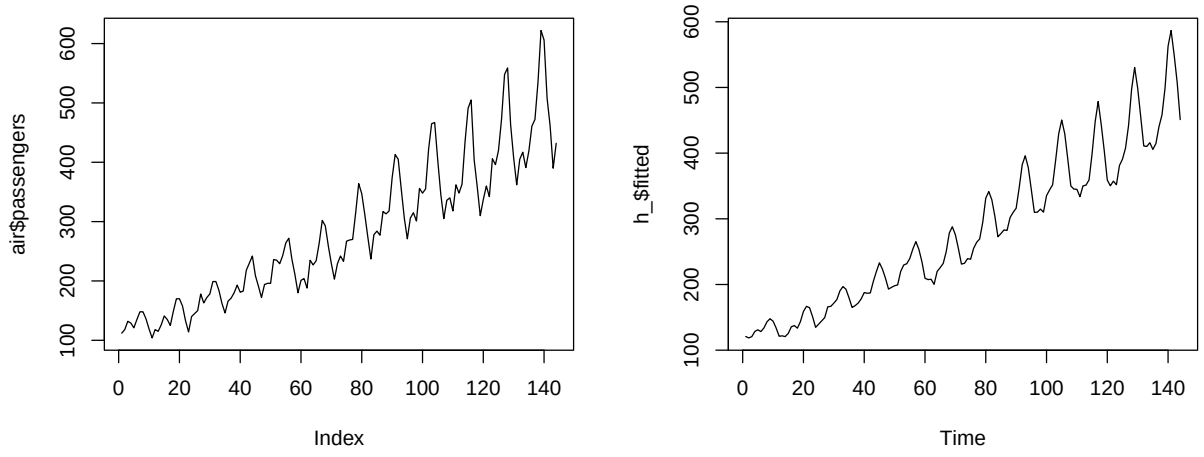
> **Suavização exponencial simples :** não apresenta um bom desempenho no caso de dados com uma tendencia a longo prazo

Métodos de Holt e HW São dois metodos de suavização exponencial aplicados a dados com uma tendencias ou com tendencias e sazonalidades

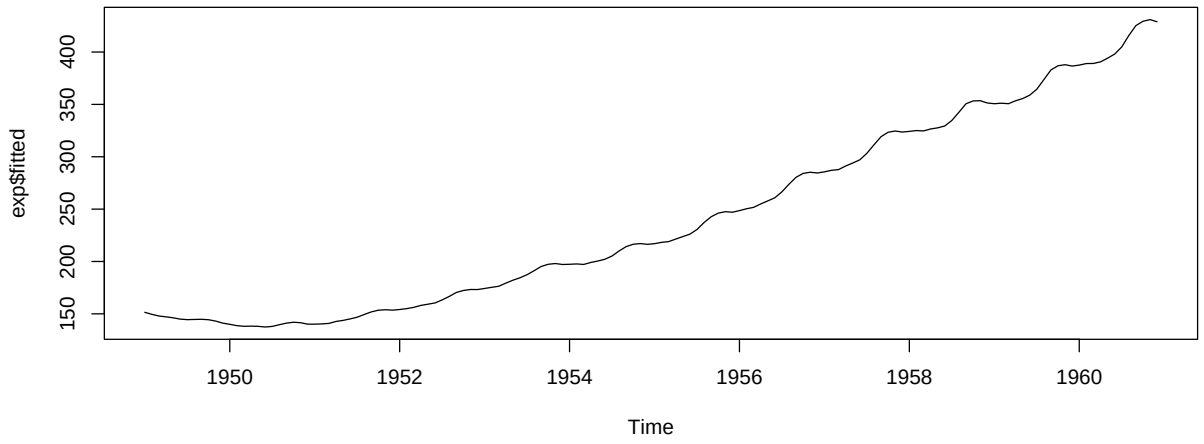
```
path = "C:\\Users\\mateu\\Documents\\MEGA\\Projetos-git\\analise-pratica-series-temporais_predicao-stats
air = fread(path)
```

```
h_ = holt(air$passengers, alpha = 0.5, h = length(air$passengers))
```

```
par(mfrow=c(1,2))
plot(air$passengers, type = 'l')
plot(h_$fitted, type='l')
```



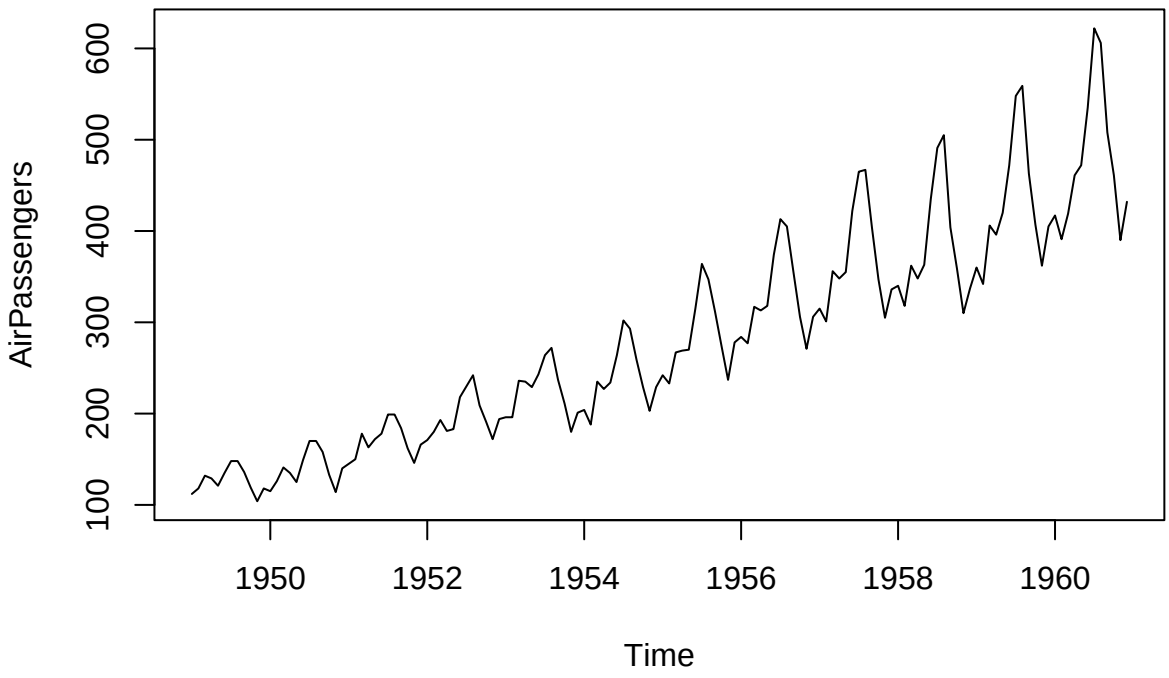
```r
# Suavização exponencial simples
exp = ses(AirPassengers, alpha = 0.05)
plot(exp$fitted)
```



Dados Sazonais

Sazonalidade é qualquer tipo de comportamento recorrente no qual a frequencia é estável.

```r
plot(AirPassengers)
```




```
plot(AirPassengers, type='p')
```

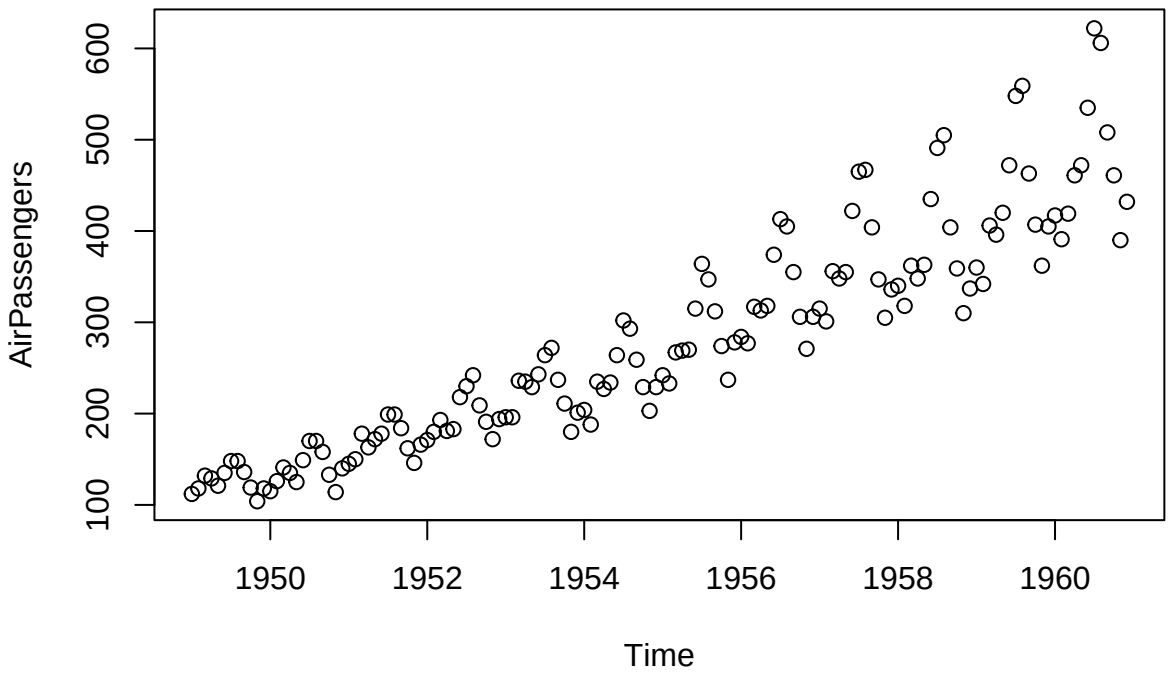


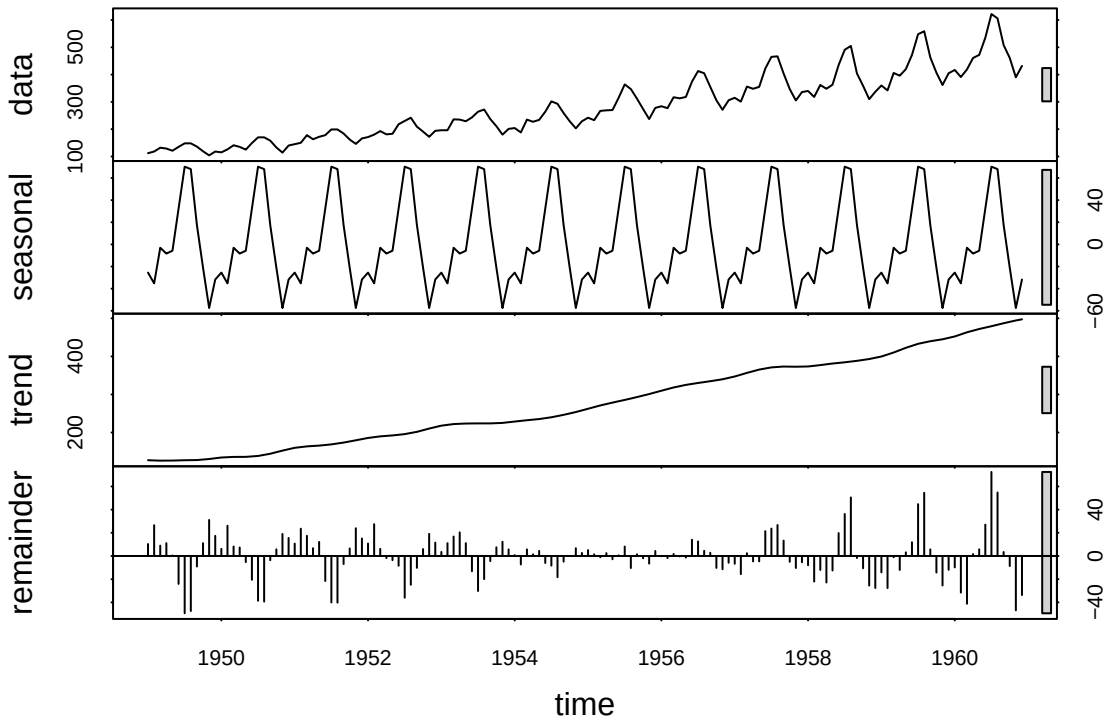
Gráfico de dispersão mostra algumas informações mais claramente do que o gráfico de linha. Variancia de nossos dados está aumentando, assim como a média.

Esses dados apresentam uma tendencia, assim provavelmente recorreremos a data transformation:

- log transform
- data differencing

Decompondo dos dados no R

```
plot(stl(AirPassengers, "periodic"))
```



TREND : tendencia

SEASONAL : sazonal