Financial Analytics - modelagem ARIMA e Prophet

PADS

Paloma Vaissman Uribe

Insper

Jul 2023

Modelagem ARIMA

- Os modelos ARIMA são ferramentas poderosas para modelagem de dados que possuem autocorrelação, sendo uma das técnicas clássicas mais conhecidos além dos métodos de Suavização Exponencial.

AR: auto regressive

I: integrated

MA: moving average

Enquanto o componente AR diz respeito à **defasagens da própria série**, o componente MA se relaciona com as **defasagens dos erros (ruídos brancos)**. Por último, o componente de integração diz **quantas diferenciações** devem ser feitas para "estacionarizar" séries que possuem tendências estocásticas (raízes unitárias).

Modelos autoregressivos (AR)

São modelos denominados **autoregressivos**, em que o processo y_t depende linearmente dos valores observados no passado. Genericamente, temos um processo gerador AR(p) dado por:

$$y_t = \phi_0 + \phi_1 y_{t-1} + \ldots + \phi_p y_{t-p} + \varepsilon_t, \quad \varepsilon_t \sim RB(0, \sigma^2).$$

O AR(p) é estacionário se atender a algumas condições em seus coeficientes. Para o AR(1),

$$y_t = \phi_0 + \phi_1 y_{t-1} + \varepsilon_t, \quad |\phi_1| < 1.$$

Modelos autoregressivos (AR)

Como determinar a ordem p?

Para isso, usamos outra função denominada **autocorrelação parcial** (PACF), que tem como base os coeficientes de correlação parcial (p.ex. correlação entre y_t e y_{t-2}, mantido y_{t-1} constante). De maneira geral:

- Observar a ACF empírica (amostral) e verificar se esta tende rapidamente para zero.

- Observar a PACF empírica (amostral): espera-se que seja aproximadamente zero para todas as ordens superiores à ordem p do processo. Dizemos que a PACF "corta" no lag p.

Modelos de médias móveis (MA)

São modelos denominados **médias móveis**, em que a série y_t depende linearmente das defasagens dos erros (perturbações).

Genericamente, temos um modelo MA(q) dado por:

$$y_t = \theta_0 + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \ldots - \theta_p \varepsilon_{t-q}, \quad \varepsilon_t \sim RB(0, \sigma^2).$$

O processo $MA(q)$ é **sempre estacionário**.

Modelos de médias móveis (MA)

Como determinar a ordem q?

- Em geral, num processo estacionário MA(q), os q primeiros coeficientes da ACF são diferentes de zero e os restantes iguais a zero.

- Já os coeficientes da PACF, apresentam um decaimento amortecido para zero.

Modelo ARMA

Processo misto, autorregressivo e de médias móveis, um modelo ARMA(p,q) por ser escrito por:

$$y_t \Phi(L) = \alpha + \Theta(L)\varepsilon_t, \quad \varepsilon_t \sim RB(0, \sigma^2),$$

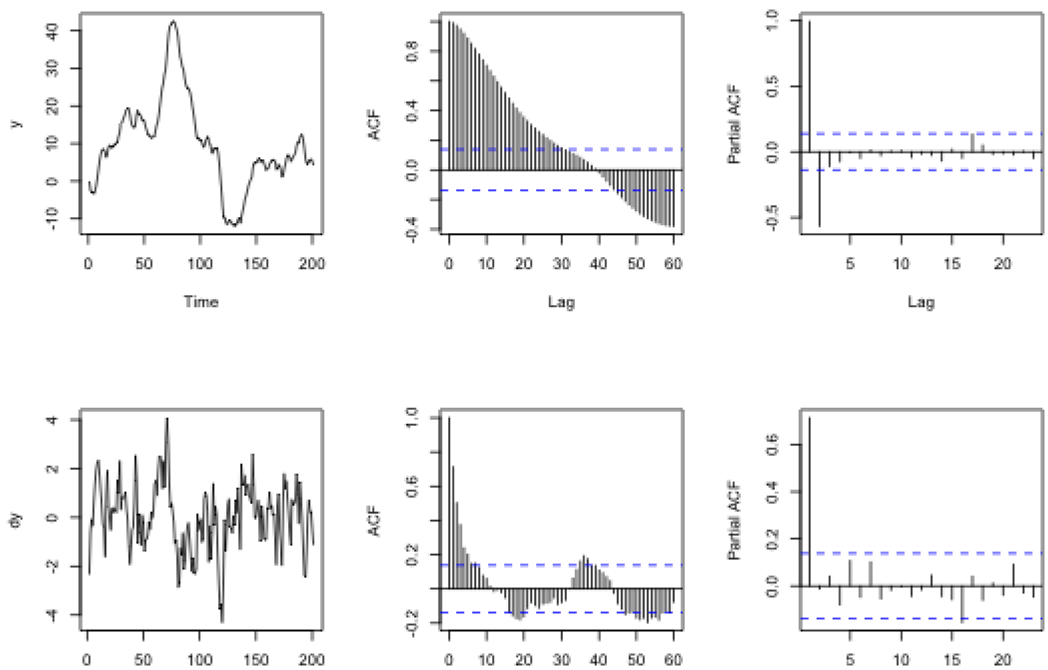
em que $\Phi(L)$ e $\Theta(L)$ são, respectivamente, os polinômios autogressivo de ordem p e de médias móveis de ordem q.

As condições de **estacionariedade** seguem as mesmas regras dos modelos AR(p) e MA(q).

Processo integrado ou tendência estocástica

Definição. Se $\Delta^d y_t = y_t - y_{t-d}$ é estacionário, para $d \geq 1$, então dizemos que y_t é integrado de ordem d e escrevemos $y_t \sim I(d)$.

Quando um processo é integrado de ordem 1, implica em trabalharmos com a variável diferençada uma vez. Assim, serão analisadas as variações dessa variável (taxas de crescimento).



Metodologia Box Jenkins

A metodologia de Box-Jenkins para a previsão se baseia no ajuste de **modelos ARIMA** para séries temporais de forma que a diferença entre os valores gerados pelos modelos e os valores observados resulte em séries de resíduos de comportamento aleatório em torno de zero (ruído branco).

No estabelecimento de um modelo ARIMA(p,d,q) para uma série temporal existem algumas etapas a considerar:

- Identificação das ordens dos componentes (p, d, q);
- Estimação do modelo proposto; e
- Diagnóstico dos resíduos.

Um modelo bem ajustado é aquele que os resíduos são i.i.d. (ruído branco).

Etapa 1: Identificação

Para identificação das ordens, temos que avaliar:

- O gráfico da série original;
- A ACF;
- A PACF.

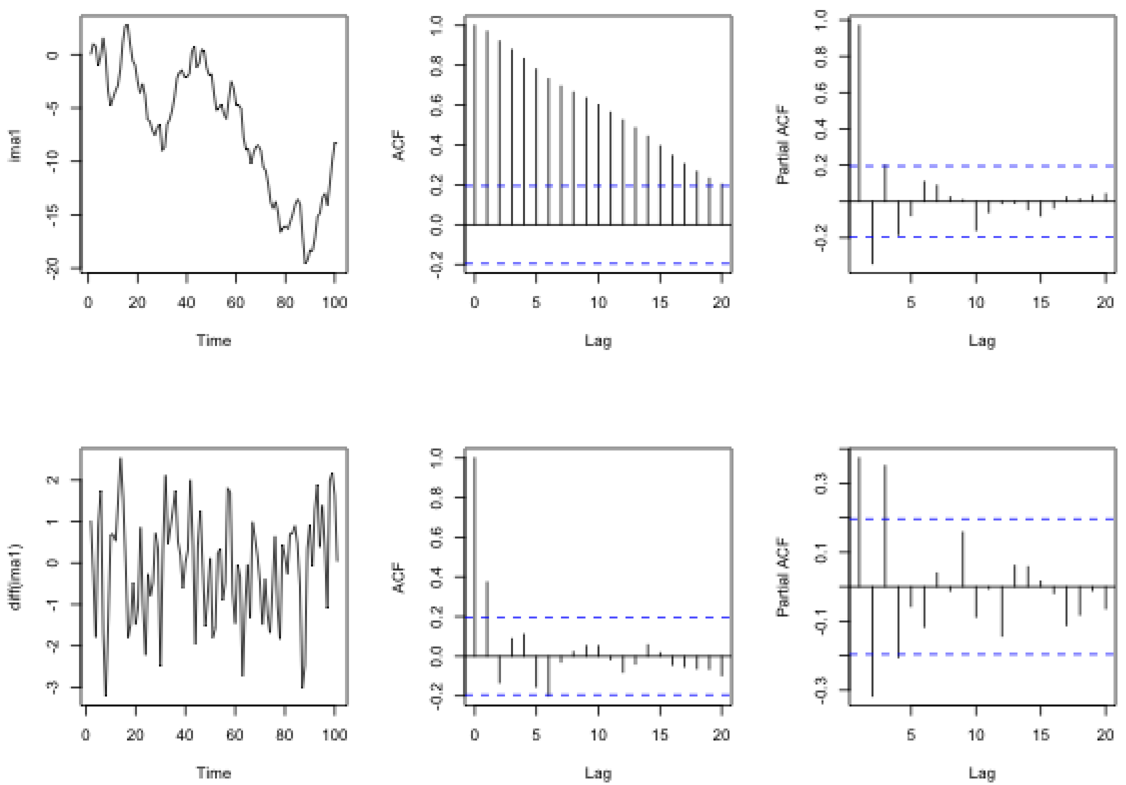
E em alguns casos realizar um teste de raíz unitária. Também podemos estimar vários modelos e avaliar o melhor através dos critérios de informação AIC e BIC.

Por exemplo, considere o processo IMA(1,1) ou ARIMA(0,1,1) simulado:

```
ima1 <- arima.sim(list(order = c(0,1,1), ma = 0.9), n = 100)
```

Etapa 1: Identificação

Verificamos pelos gráficos das ACFs de Δy_t e y_t que trata-se de um processo $I(1)$ tal que $\Delta y_t \sim MA(1)$ (veja que a ACF da primeira diferença "corta" no lag 1)



Resumo: identificação

Resumidamente:

Modelo	FAC	FACP
AR(p)	Infinita: decai para zero (exponencialmente ou segundo uma senóide amortecida).	Finita: decai bruscamente para zero a partir do lag p.
MA(q)	Finita: decai bruscamente para zero a partir do lag q.	Infinita: decai para zero (exponencialmente ou segundo uma senóide amortecida).
ARMA(p,q)	Infinita: decai para zero (exponencialmente ou segundo uma senóide amortecida).	Infinita: decai para zero (exponencialmente ou segundo uma senóide amortecida).

Em alguns casos, pode ser interessante utilizar os **critérios de informação AIC, cAIC e BIC**, escolhendo o modelo (ordens) que minimiza um deles. Outra opção muito utilizada é a função `auto.arima` do pacote `forecast`, que realiza essa minimização de forma automática.

Etapa 2: Estimação

Agora vamos estimar o modelo, utilizando o pacote `forecast`:

```
set.seed(1)
ima1 <- arima.sim(list(order = c(0,1,1), ma = 0.9), n = 100)
library(forecast)
Arima(ima1,order=c(0,1,1))
```

```
## Series: ima1
## ARIMA(0,1,1)
##
## Coefficients:
##          ma1
##       0.9087
## s.e.  0.0572
##
## sigma^2 = 0.8215:  log likelihood = -132.43
## AIC=268.87   AICc=268.99   BIC=274.08
```

Alternativa: identificação de modelos via critérios de informação (AIC)

Para o processo IMA(1,1):

```
p <- 3
q <- 3

resultsAIC <- matrix(0,p+1,q+1)
colnames(resultsAIC) <- c(0:p)
rownames(resultsAIC) <- c(0:q)

for (i in 0:p){
  for(j in 0:q){
  resultsAIC[i+1,j+1] <- Arima(ima1, order = c(i, 1, j))$aicc
  }
}
which(resultsAIC == min(resultsAIC), arr.ind = TRUE)
```

```
##   row col
## 0   1   2
```

Alternativa: identificação de modelos via critérios de informação (BIC)

Para o processo IMA(1,1):

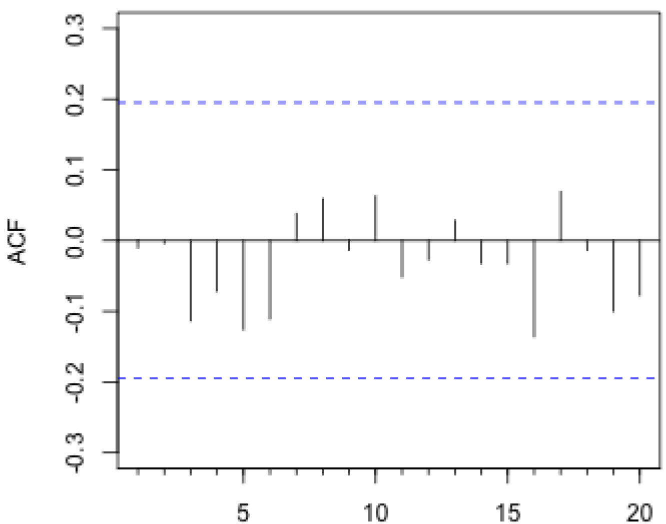
```
resultsBIC <- matrix(0,p+1,q+1)

colnames(resultsBIC) <- c(0:p)
rownames(resultsBIC) <- c(0:q)

for (i in 0:p){
  for(j in 0:q){
  resultsBIC[i+1,j+1] <- Arima(ima1, order = c(i, 1, j))$bic
  }
}
which(resultsBIC == min(resultsBIC), arr.ind = TRUE)
```

```
##   row col
## 0   1   2
```

Etapa 3: Diagnóstico

Estimado o modelo, precisamos analisar os resíduos. Podemos plotar os resíduos e fazer testes de autocorrelação dos mesmos. Utilizando o pacote `forecast`:

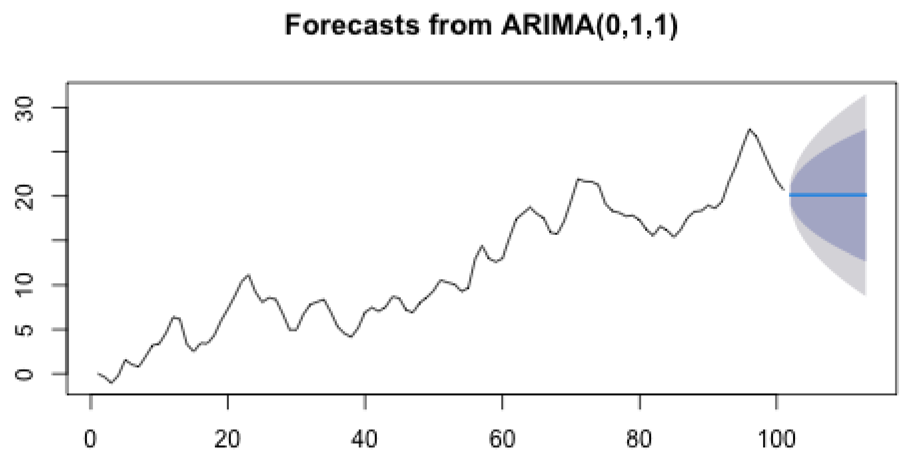
```
res <- Arima(ima1,order=c(0,1,1))$res
Acf(res,main="",xlab="")
```



Previsão de modelos ARIMA

Utilizando o exemplo anterior do IMA(1,1), podemos gerar as previsões utilizando o pacote `forecast`:

```
fit <- Arima(ima1,order=c(0,1,1))
plot(forecast(fit,h=12))
```



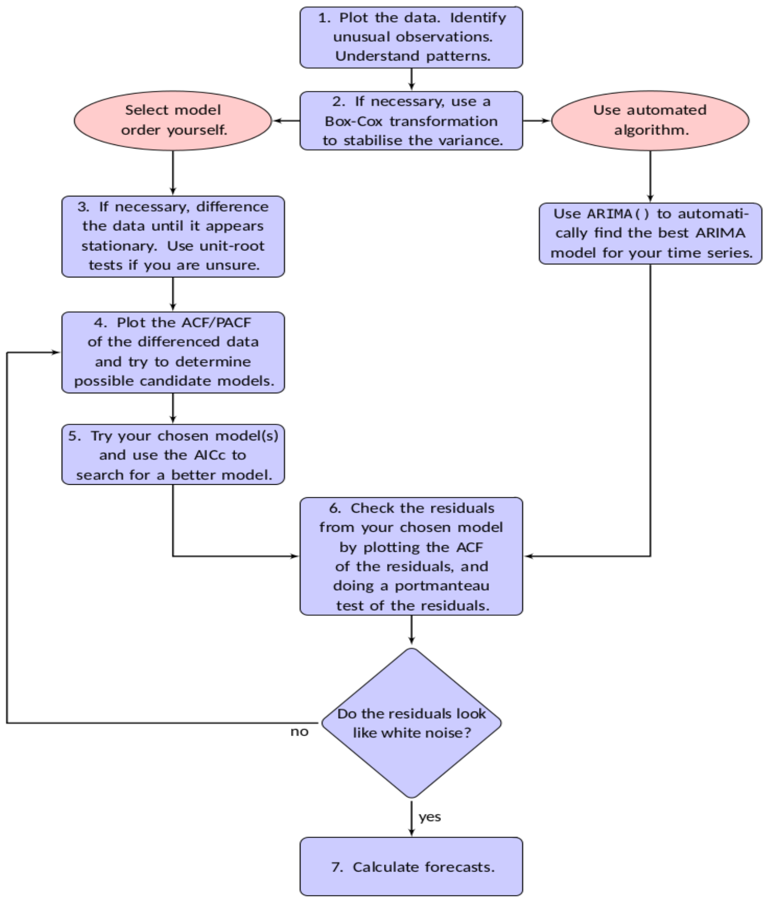
Pacote `fable`

- Se você gosta do tidyverse e afins, o professor Hyndman fez um pacote específico para trabalhar com dados nessa estrutura. É baseado no pacote anterior `forecast`.

- A função ARIMA() utiliza o mecanismo do `auto.arima` para achar a melhor combinação de orders de um ARIMA(p,d,q) utilizando um algoritmo stepwise.

- Ou seja, é posssível usar Box-Jenkins e/ou o algoritmo desenvolvido pelos autores, que baseia-se na minimização de critérios de informação, porém de um modo mais inteligente, sem ter que rodar todas as combinações possíveis.

Fluxo de identificação: Box-Jenkins vs Hyndman-Khandakar



Exercício usando `fable`

Vamos ler o livro na página https://otexts.com/fpp3/arima-r.html e escolher uma série presente no FRED (https://fred.stlouisfed.org/) para fazer a previsão. Para isso, analise a ACF e o gráfico da série, identifique, estime e analise os resíduos do modelo selecionado usando o pacote `fable`.

ARIMA com variáveis exógenas

O modelo ARIMAX simplesmente adiciona uma covariável ou regressor exógen x_t no lado direto da equação

$$y_t = \beta x_t + \phi_1 y_{t-1} + \ldots + \phi_p y_{t-p} + \epsilon_t - \theta_1 \epsilon_{t-1} - \ldots - \theta_q \epsilon_{t-q}.$$

Problemas: A presença de valores defasados da variável resposta y_t significa que o coeficiente β só pode ser interpretado condicionalmente aos valores anteriores de y_t, o que não é intuitivo. Por essa razão, muitas vezes prefere-se o modelo

$y_t = \beta x_t + \eta_t$

$\eta_t = \phi_1 \eta_{t-1} + \ldots + \phi_p \eta_{t-p} + z_t - \theta_1 z_{t-1} - \ldots - \theta_q z_{t-q},$

i.e., equivale a uma regressão tradicional, porém com os erros $\eta_t \sim ARMA(p, q)$.

Lembrete: se estamos lidando com variáveis integradas, devemos antes diferenciar y_t e x_t para evitar **regressão espúria**.

Tratamento da sazonalidade

- Vimos na parte de componentes de uma série que a sazonalidade é um padrão recorrente. Mas recorrente como? Se repete com qual frequência?
  - **periodicidade** da sazonalidade: indica de quanto em quanto tempo (dada a frequência dos dados) esse padrão se repete, por exemplo: as vendas mensais de uma loja parecem ter um pico recorrente no mês de dezembro e assim por diante.
- Veremos 3 modelos possíveis que tratam a sazonalidade:
  - Facebook Prophet
  - SARIMA
  - ARIMAX (com dummies sazonais)

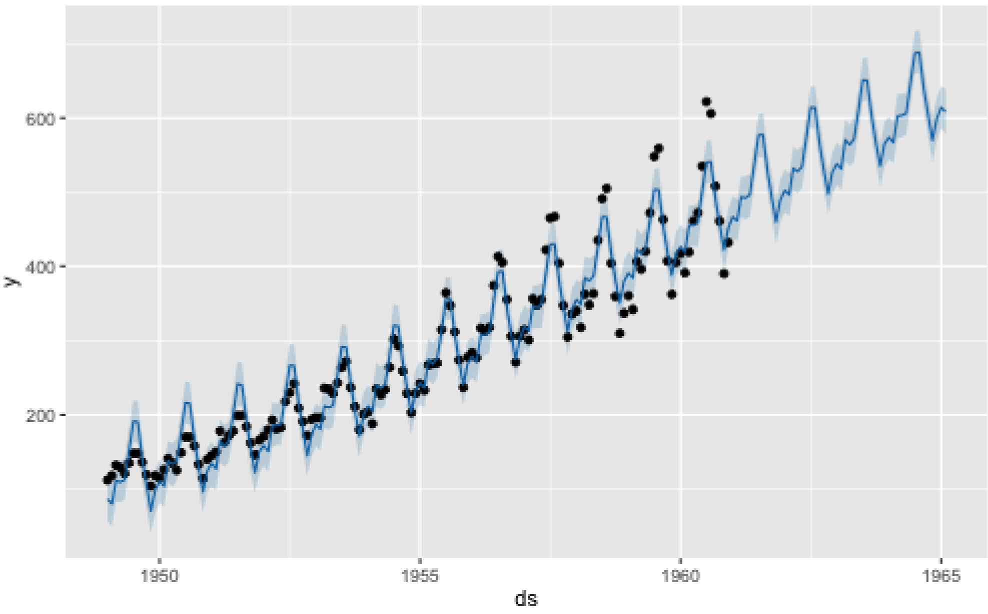
Modelagem da sazonalidade: prophet

Como já apresentado, vamos ilustrar o exemplo de sazonalidade com a clássica série Airline, que pode ser encontrada no R. Veja que o pacote Prophet tem como default uma **modelagem da sazonalidade aditiva**. Porém, parece não ser adequada, pois é muito alta no começo e muito baixa no fim.

```
library(prophet)
library(xts)
data(AirPassengers)
AP <- AirPassengers
df <- data.frame(ds = zoo::index(as.xts(AP)),y = coredata(as.xts(AP))
m <- prophet(df)
future <- make_future_dataframe(m, 50, freq = 'm')
forecast <- predict(m, future)
plot(m, forecast)
```

Modelagem da sazonalidade: prophet

Ajuste **sazonalidade aditiva**



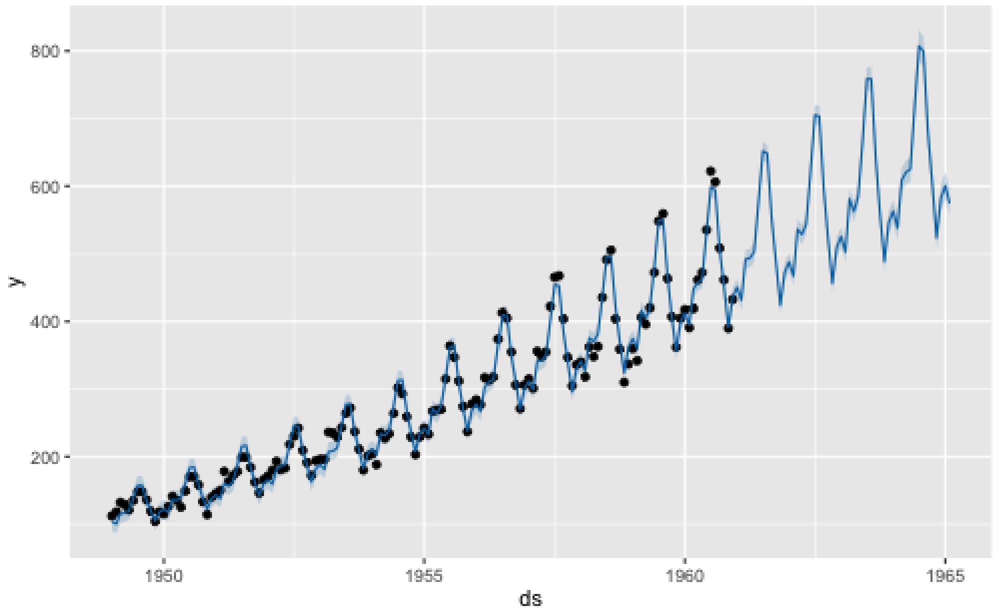
Modelagem da sazonalidade: prophet

Com a **sazonalidade multiplicativa** o modelo parece ter melhor ajuste. Isso acontece porque a sazonalidade é um múltiplo da tendência, aumentando com ela.

```r
library(prophet)
library(xts)
data(AirPassengers)
AP <- AirPassengers
df <- data.frame(ds = zoo::index(as.xts(AP)),y = coredata(as.xts(AP)))
m <- prophet(df, seasonality.mode = 'multiplicative')
future <- make_future_dataframe(m, 50, freq = 'm')
forecast <- predict(m, future)
plot(m, forecast)
```

Modelagem da sazonalidade: prophet

Ajuste **sazonalidade multiplicativa**



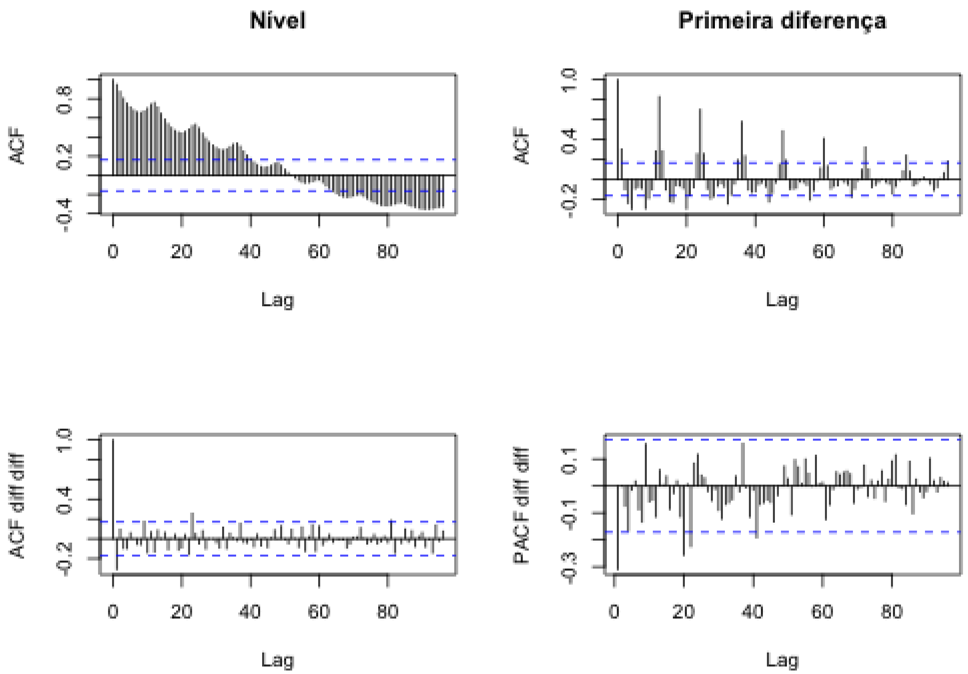
SARIMA

O modelo ARIMA sazonal incorpora fatores não sazonais e sazonais em um **modelo multiplicativo**. Uma notação abreviada para o modelo é SARIMA $(p, d, q) \times (P, D, Q)_S$, onde

- p, d e q são as ordens AR, I e MA,
- P, D e Q são as ordens AR, I e MA relacionadas à sazonalidade e
- S é a periodicidade.

Ajustando um SARIMA para a série AIRLINE

Primeiramente, vamos usar a amostra sem o ano de 1960. Em seguida, plotamos a ACF da primeira diferença e verificamos que a ACF nos lags sazonais multiplos de 12 é diferente de 0.



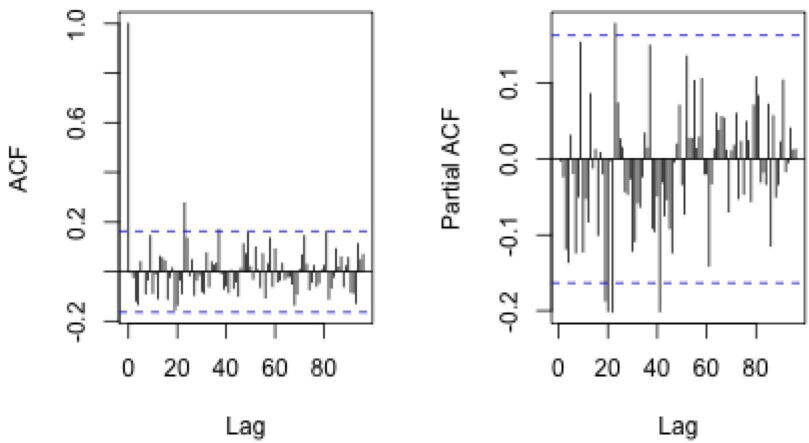
Ajustando um SARIMA para a série AIRLINE

Verificamos que tanto ACF quanto a PACF no lag 1 são diferentes de 0. Logo, podemos tentar combinações entre p=0:1 e q=0:1.

```
##           aic     aicc      bic
## [1,] 1020.393 1020.582 1029.019
## [2,] 1022.356 1022.674 1033.857
## [3,] 1031.508 1031.539 1034.383
## [4,] 1020.639 1020.733 1026.390
## [5,] 1020.493 1020.811 1031.994
## [6,] 1020.394 1020.488 1020.488
```

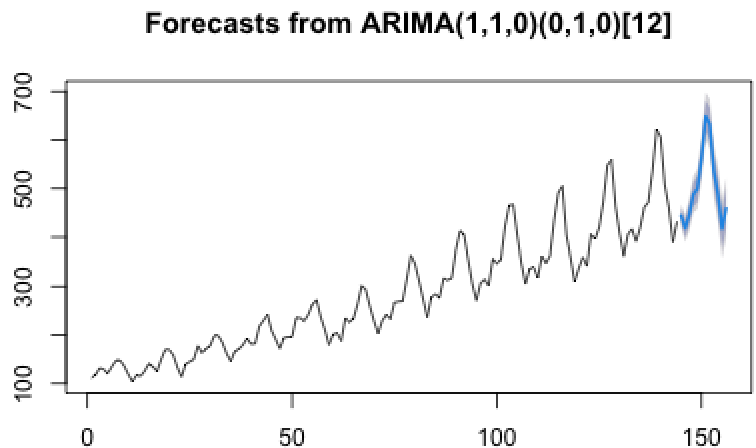
Melhor modelo SARIMA para a série AIRLINE

```
fit = Arima(y,order=c(1,1,0),seasonal=list(order=c(0,1,0),period=12)
par(mfrow=c(1,2))
acf(residuals(fit),96,main="")
pacf(residuals(fit),96,main="")
```



Melhor modelo SARIMA para a série AIRLINE

```
plot(forecast(fit,12))
```



Usando `auto.arima` para encontrar ordens do SARIMA

```r
library(forecast)
auto.arima(AP)
```

```
## Series: AP
## ARIMA(2,1,1)(0,1,0)[12]
##
## Coefficients:
##          ar1     ar2      ma1
##       0.5960  0.2143  -0.9819
## s.e.  0.0888  0.0880   0.0292
##
## sigma^2 = 132.3:  log likelihood = -504.92
## AIC=1017.85   AICc=1018.17   BIC=1029.35
```

Usando `fable` para encontrar ordens do SARIMA

```
library(fable)
```

```
## Carregando pacotes exigidos: fabletools

##
## Attaching package: 'fabletools'

## The following object is masked from 'package:forecast':
##
##     accuracy
```

```
airline_fable <- as_tsibble(AP) %>%
  model(stepwise = ARIMA(value))
glance(airline_fable)
```

```
## # A tibble: 1 × 8
##   .model   sigma2 log_lik   AIC  AICc   BIC ar_roots  ma_roots
##   <chr>     <dbl>   <dbl> <dbl> <dbl> <dbl> <list>    <list>
## 1 stepwise   132.   -505. 1018. 1018. 1029. <cpl [2]> <cpl [1]>
```

Ajustando um modelo aditivo para a série AIRLINE

```
library(lubridate)
months <- as.factor(month(df$ds)) #extracting month using lubridate
dummies <- as.data.frame(model.matrix(~months))
colnames(dummies) <- c("intercept","fev","mar","abr","mai","jun",
                       "jul","ago","set","out","nov","dez")
dummies$trend <- c(1:dim(dummies)[1]) # ading a deterministic trend
data.frame <- cbind(y,dummies)
attach(data.frame)
summary(lm(y~fev+mar+abr+mai+jun+jul+ago+set+out+nov+dez+trend))
```

Ajustando um modelo ARIMAX para a série AIRLINE

```
fit = Arima(y[1:132],order=c(1,1,0),method="ML",xreg=as.matrix(data.f
plot(forecast(fit,12,xreg=as.matrix(dummies[133:144,2:13])))
```