Séries Temporais

Fatec 2025

Definição

- dados coletados em intervalos regulares de tempo
  - Década, ano, mês, dia, hora, minuto, segundo
  - Dependência de ordem

Uma série temporal é um conjunto sequencial de pontos de dados, medido tipicamente em tempos sucessivos

- Usado para:
  - Compreender um fenômeno
  - Prever eventos

Diferença entre Análise e Previsão

Análise de Séries Temporais (Descritiva):

- • Busca entender o comportamento dos dados ao longo do tempo
- • Identifica **tendências** (crescimento ou queda ao longo do tempo), **sazonalidades** (padrões que se repetem, como estações do ano) e **ruído**
  - • **Ruído**: variações aleatórias ou imprevisíveis nos dados que não seguem um padrão claro. Exemplo: flutuações inesperadas nas vendas por causa de um evento não planejado

• **Previsão (Forecasting)**:

- • Objetivo é estimar **valores futuros** com base nos dados passados
- • O futuro não pode ser observado diretamente, então usamos os dados históricos para **extrapolar**
  - • **Extrapolação**: processo de usar tendências observadas nos dados atuais para prever valores fora do intervalo conhecido (ou seja, no futuro)

Componentes

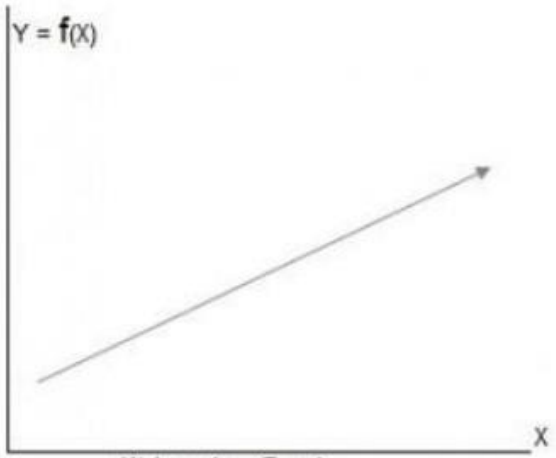
Dependência da ordem

- Total de pessoas atendidas em um Hospital por dia

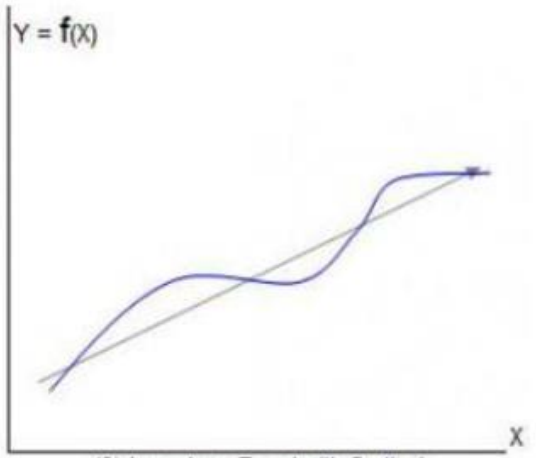
01/03	02/03	03/03
323	298	404

Componentes de séries temporais

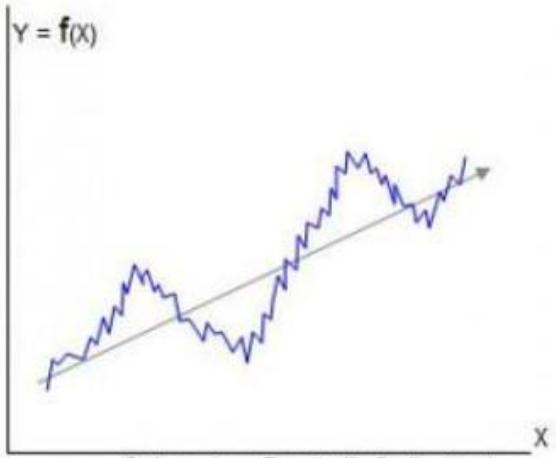
- Tendência
  - Movimento de longo prazo: crescimento, queda ou estabilidade
  - Ex.: crescimento populacional
- Ciclo
  - Flutuações irregulares e de longo
  - Causa: fatores macroeconômicos ou sistêmicos
  - Frequência: irregular e imprevisível
  - Previsibilidade: baixa
  - Ex.: aumento de vendas em Dezembro
- Sazonalidade
  - Padrão regular e previsível que se repede em intervalos
  - Causas: calendário ou clima
  - Frequência: constante e conhecida
  - Previsibilidade: alta
  - Ex.: recessões e expansões econômicas
- Irregularidade ou ruído
  - Variações aleatórias sem padrão



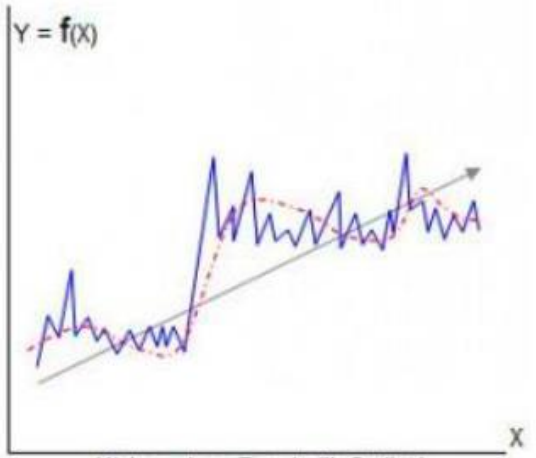
(1) Long-term Trend



(2) Long-term Trend with Cyclical Variations/Movements



(3) Long-term Trend with Cyclical and Seasonal Variations/Movements



(4) Long-term Trend with Cyclical, Seasonal, and Random Variations/Movements

Componentes

- Em **séries curtas**, muitas vezes **não se distingue ciclo de tendência** e o ciclo pode ser ignorado
- O **ruído** é o que sobra após remover os demais componentes
- Nem toda série tem todos os componentes
  - por exemplo, séries de dados climáticos podem ter apenas **tendência** e **sazonalidade**

Forma de representação

- **Modelo aditivo:**

$$Y(t) = T(t) + S(t) + C(t) + R(t)$$

- **Modelo multiplicativo:**

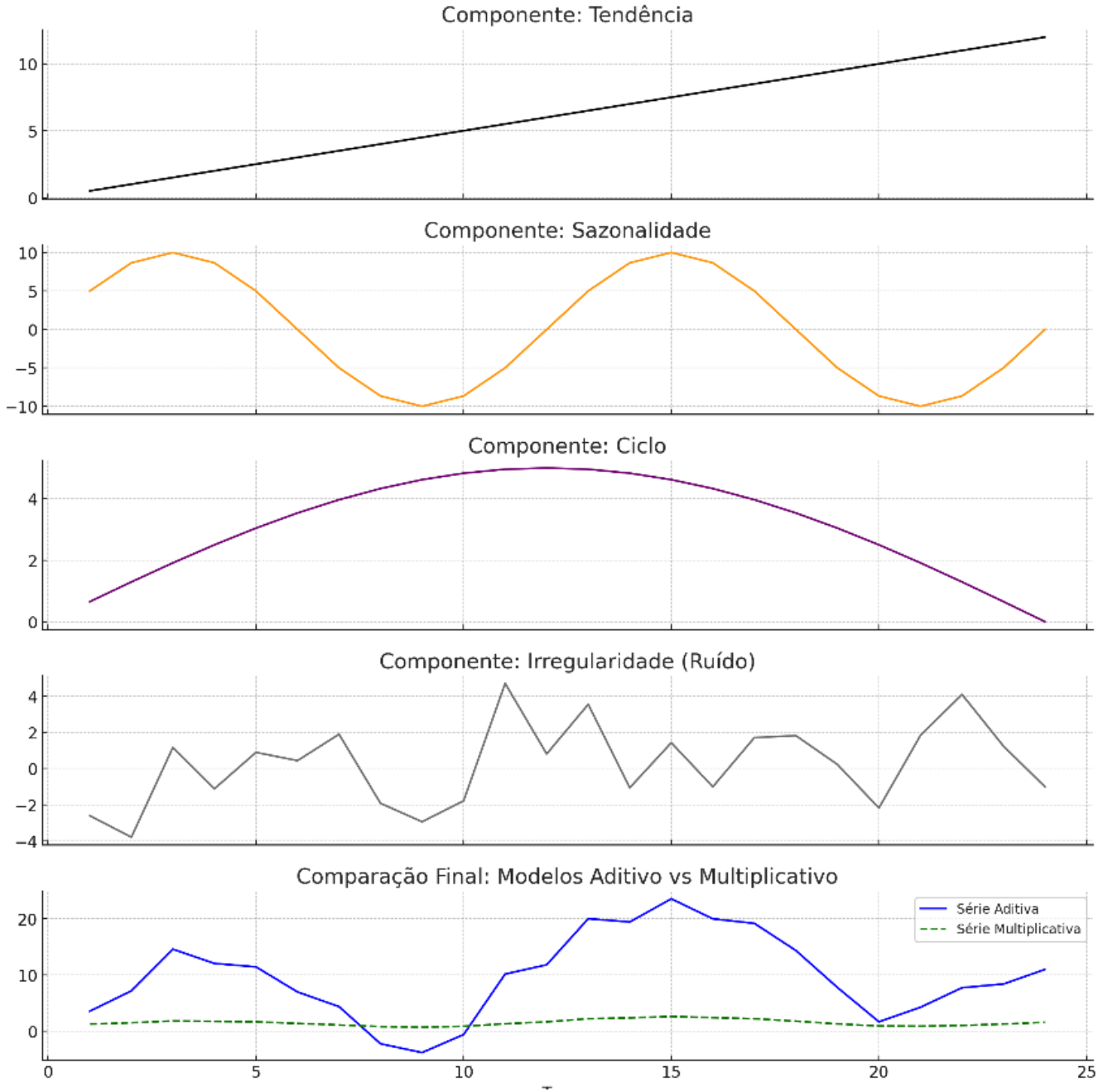
$$Y(t) = T(t) \times S(t) \times C(t) \times R(t)$$

Use o modelo aditivo quando as flutuações forem constantes em magnitude
Use o multiplicativo quando variarem proporcionalmente ao nível da série

agenciadenoticias.ibge.gov.br

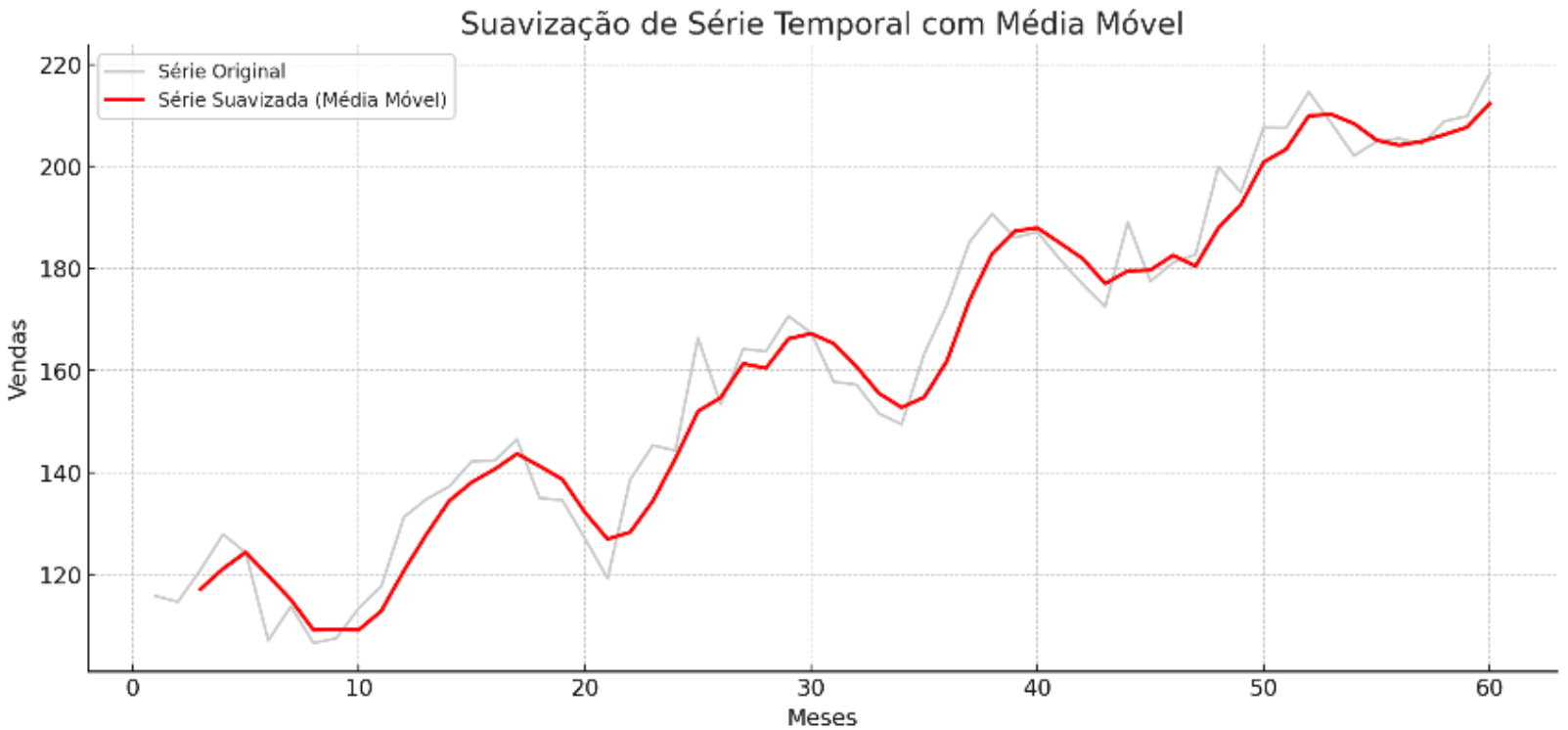
Comparando os modelos

- **Modelo Aditivo**
  - Assume que os componentes da série temporal se somam
  - É apropriado quando a amplitude das variações sazonais é constante ao longo do tempo
- **Modelo Multiplicativo**
  - Assume que os componentes se multiplicam
  - É adequado quando a amplitude das variações sazonais aumenta ou diminui proporcionalmente ao nível da série
- Se as flutuações sazonais mantêm uma amplitude constante, o modelo aditivo é mais apropriado
- Se as flutuações sazonais aumentam em amplitude conforme o nível da série aumenta, o modelo multiplicativo pode ser mais adequado



Média Móvel

- Técnica de suavização a fim de reduzir ruídos permitindo ver melhor as tendências e sazonalidades de uma série temporal



Média Móvel

Uma loja vende picolés e registra as vendas dos últimos 7 dias.

Dia	Vendas (unidades)
Segunda	30
Terça	28
Quarta	35
Quinta	33
Sexta	50
Sábado	60
Domingo	55

A cada dia, tiramos a **média dos 3 últimos dias** (incluindo o dia atual):

Dia	Vendas	Média Móvel (3 dias)
Segunda	30	—
Terça	28	—
Quarta	35	(30+28+35)/3 = **31,0**
Quinta	33	(28+35+33)/3 = **32,0**
Sexta	50	(35+33+50)/3 = **39,3**
Sábado	60	(33+50+60)/3 = **47,7**
Domingo	55	(50+60+55)/3 = **55,0**

- Suaviza picos e vales (sábado e terça)
- Identifica tendências (aumento de vendas durante a semana)
- Pode ser usada para previsões curtas: aumentar o estoque aos finais de semana)

Média Móvel em R

```r
# Instalar pacote, se necessário
install.packages("zoo")
```

```r
library(zoo)
```

```r
# Dados de vendas por dia
dias <- c("Seg", "Ter", "Qua", "Qui", "Sex", "Sáb", "Dom")
vendas <- c(30, 28, 35, 33, 50, 60, 55)
```

```r
# Calcular média móvel de 3 dias
media_movel_3 <- rollmean(vendas, k = 3, align = "right", fill = NA)
```

```r
# Gerar gráfico
plot(vendas, type = "o", col = "blue", xaxt = "n",
     xlab = "Dia da Semana", ylab = "Vendas (unidades)",
     main = "Vendas Diárias vs Média Móvel (3 dias)",
     ylim = c(min(vendas), max(vendas)))
axis(1, at = 1:7, labels = dias)
lines(media_movel_3, type = "o", col = "orange")
# Adicionar legenda
legend("topleft", legend = c("Vendas Diárias", "Média Móvel 3 dias"),
       col = c("blue", "orange"), lty = 1, pch = 1)
```

Tabela Calendário no Power BI

- Funções para criar uma tabela calendário em Dax:
- Calendarauto()
- Calendar(date(aa,mm,dd), date(aa,mm,dd)

Prática power Bi

Dataset.csv

- Carregar os dados
  - Criar uma coluna para o dia, mês e ano
  - Em DAX: day, month e year
- Pré-processamento:
  - Qual a menor data?
  - Qual a maior data?
  - Qual o menor valor de vendas? E o maior? E a média?

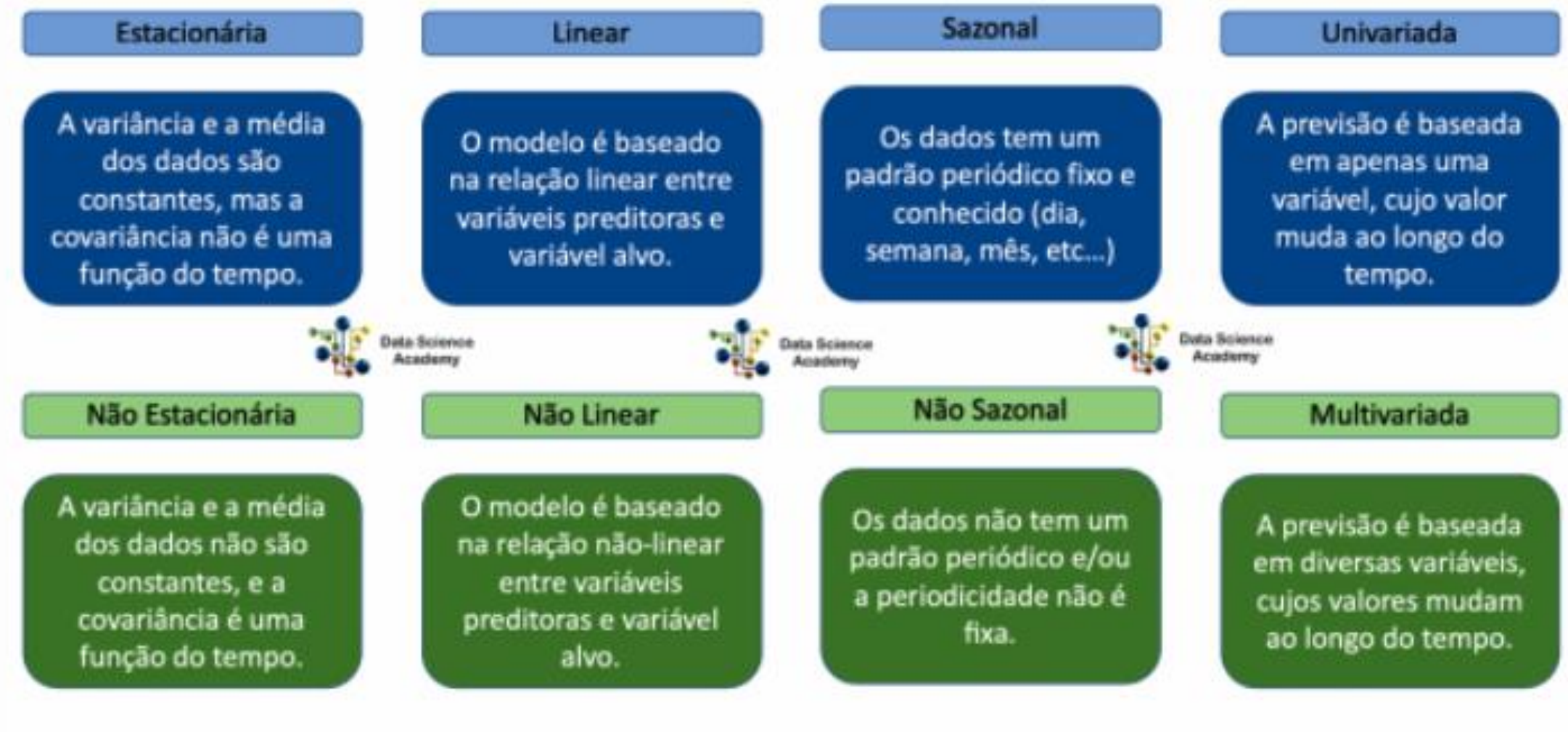
Prática Power BI

- Crie um gráfico de linhas por dia, mês e ano
- Crie a média móvel:

```
MediaMovel_3Dias :=
AVERAGEX(
    -- Cria uma tabela de datas de 3 dias, terminando na data atual do contexto
    DATESINPERIOD(
        dataset[Data],          -- Coluna de datas
        MAX(dataset[Data]),     -- Data atual do contexto
        -2,                     -- Pega os 2 dias anteriores + o dia atual = 3 dias
        DAY
    ),
    CALCULATE(SUM(dataset[Total_vendas]))  -- Soma das vendas para cada data da janela
)
```

- Crie uma linha de tendência
- Faça a previsão para 2024

Tipos de séries temporais



Estacionárias x Não estacionárias

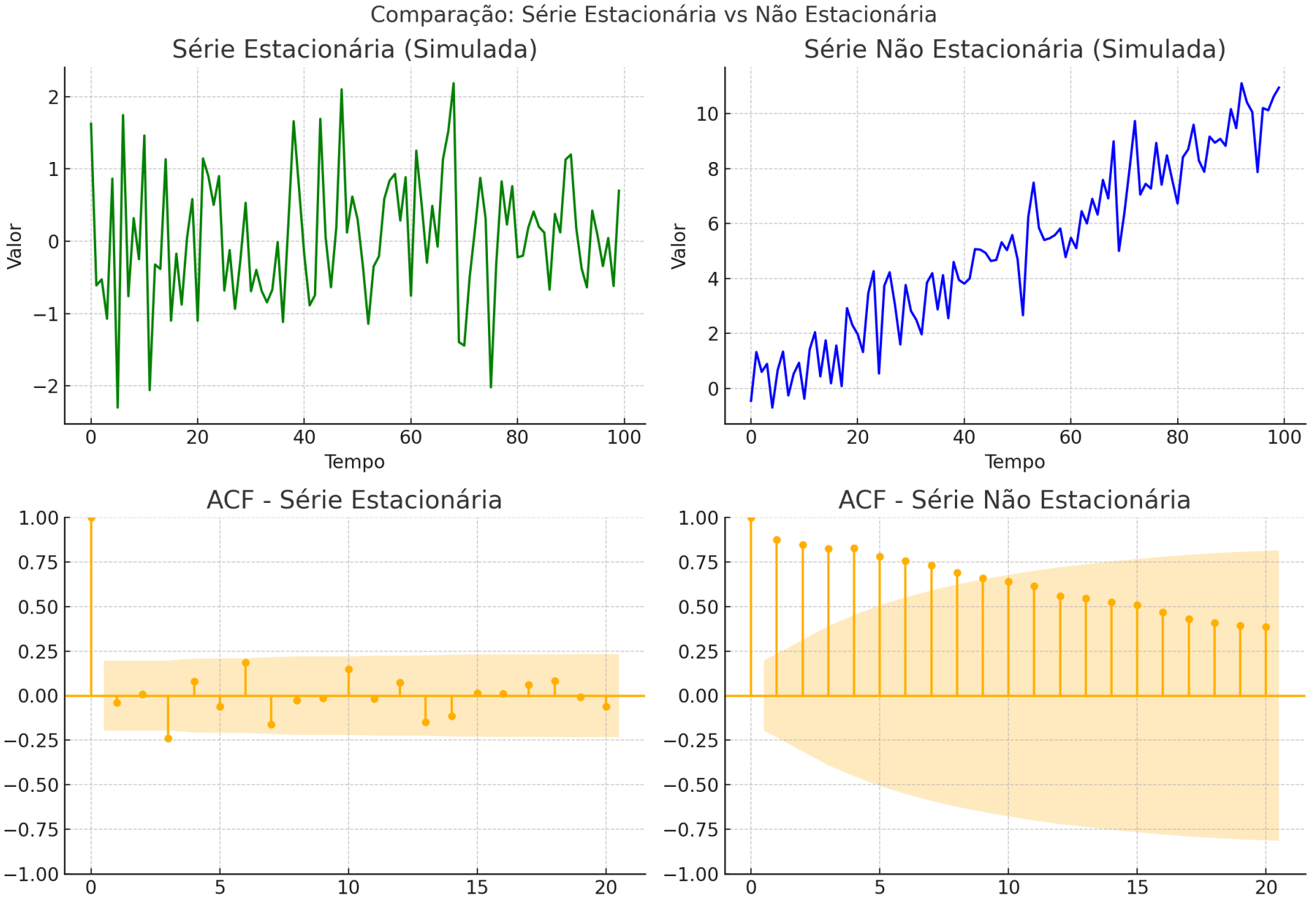
```r
# Instalar, se necessário: install.packages("ggplot2")
library(ggplot2)
# fixa a semente de números aleatórios para os resultados sejam sempre os mesmos
set.seed(123)

# Série estacionária
tempo <- 1:100
estacionaria <- rnorm(100, mean = 10, sd = 2) # gera aleatórios com distrib normal
media_est <- mean(estacionaria)
dp_est <- sd(estacionaria) #desvio padrão, medida de dispersão

# Série não estacionária (com tendência)
nao_estacionaria <- cumsum(rnorm(100, mean = 0.5, sd = 2)) #soma acumulada
media_nao_est <- mean(nao_estacionaria)
dp_nao_est <- sd(nao_estacionaria)

# Data frame combinado
df <- data.frame(
  Tempo = rep(tempo, 2),
  Valor = c(estacionaria, nao_estacionaria),
  Tipo = rep(c("Estacionária", "Não Estacionária"), each = 100)
)

# Médias e desvios para adicionar faixas no gráfico
faixas <- data.frame(
  Tipo = c("Estacionária", "Não Estacionária"),
  media = c(media_est, media_nao_est),
  upper = c(media_est + dp_est, media_nao_est + dp_nao_est),
  lower = c(media_est - dp_est, media_nao_est - dp_nao_est)
)
```

Estacionárias x Não estacionárias

```
# Plot com ggplot2 e faixas de variância
ggplot(df, aes(x = Tempo, y = Valor)) +
  geom_line(color = "steelblue") +
  facet_wrap(~Tipo, scales = "free_y") +
  geom_hline(data = faixas, aes(yintercept = media), linetype = "dashed", color = "darkred") +
  geom_hline(data = faixas, aes(yintercept = upper), linetype = "dotted", color = "orange") +
  geom_hline(data = faixas, aes(yintercept = lower), linetype = "dotted", color = "orange") +
  labs(title = "Séries Temporais: Estacionária vs Não Estacionária",
       subtitle = "Linhas: média (vermelha tracejada) e ±1 desvio padrão (laranja pontilhada)",
       y = "Valor", x = "Tempo") +
  theme_minimal()
```

Estacionárias x Não estacionárias

- ACF significa Autocorrelation Function (Função de Autocorrelação)
- A autocorrelação mostra se os valores passados ajudam a prever os valores futuros da própria série
- A ACF da série estacionária cai rapidamente, típico de comportamento estável
- A ACF da série não estacionária decai lentamente, indicando dependência persistente no tempo



Função de Autocorrelação

```r
# Garantir dados sem NA
dados <- na.omit(airquality)
acf(dados$Ozone, main = "ACF de Ozone")
```

Em séries com autocorrelação, os valores do passado influenciam os do presente
A ACF mede essa influência para diferentes defasagens

Padrão na ACF	Significado prático
Queda rápida e oscilação em torno de 0	Série **estacionária** (sem dependência de longo prazo)
Decaimento lento ou sustentado	Série **não estacionária** (possui tendência ou ciclo persistente)
Picos regulares em lags fixos	Indica **sazonalidade**

Linear x Não Linear

```r
# Definir uma semente para reprodutibilidade
set.seed(123)

# Período de tempo
tempo <- 1:100

# Série linear: y = a*t + erro
serie_linear <- 0.5 * tempo + rnorm(100, mean = 0, sd = 2)

# Série não linear: y = a*t^2 + erro
serie_nao_linear <- 0.05 * tempo^2 + rnorm(100, mean = 0, sd = 20)

# Criar os gráficos lado a lado
par(mfrow = c(1, 2))  # Dois gráficos na mesma janela

# Gráfico da série linear
plot(tempo, serie_linear, type = "l", col = "blue", lwd = 2,
     main = "Série Temporal Linear", xlab = "Tempo", ylab = "Valor")
abline(lm(serie_linear ~ tempo), col = "red", lty = 2)  # Regressão linear

# Gráfico da série não linear
plot(tempo, serie_nao_linear, type = "l", col = "darkgreen", lwd = 2,
     main = "Série Temporal Não Linear", xlab = "Tempo", ylab = "Valor")
lines(lowess(tempo, serie_nao_linear), col = "orange", lty = 2)  # Suavização local

par(mfrow = c(1,1))  # Reset layout
```

- **Série Linear:** A tendência segue uma linha reta, mesmo com ruído
- **Série Não Linear:** A tendência é curva, não pode ser ajustada adequadamente por uma reta
- **lm() e lowess()**: usados para mostrar a tendência linear e suavizada, respectivamente

Linear x Não Linear

```r
# Carregar séries
data("AirPassengers")
data("co2")

# Transformar AirPassengers com log (lineariza a tendência)
log_air <- log(AirPassengers)

# Gráficos lado a lado
par(mfrow = c(1, 2))

# Série aproximadamente linear (após log)
plot(log_air, main = "Série Temporal Linear (log AirPassengers)",
     col = "blue", lwd = 2, ylab = "Log(Nº passageiros)", xlab = "Ano")
abline(lm(log_air ~ time(log_air)), col = "red", lty = 2)
legend("topleft", legend = "Tendência Linear", col = "red", lty = 2, bty = "n")

# Série não linear: CO2 em Mauna Loa
plot(co2, main = "Série Temporal Não Linear (CO2)",
     col = "darkgreen", lwd = 2, ylab = "Concentração (ppm)", xlab = "Ano")
lines(lowess(time(co2), co2), col = "orange", lty = 2)
legend("topleft", legend = "Tendência Não Linear", col = "orange", lty = 2, bty = "n")

par(mfrow = c(1, 1))  # Reset layout
```

Linear x Não Linear

```r
# Carregar bases
data("AirPassengers")
data("nottem")

# 1. PLOTAR AS DUAS SÉRIES LADO A LADO
par(mfrow = c(1, 2))  # Layout horizontal

# Série aproximadamente linear (com log)
log_air <- log(AirPassengers)
plot(log_air, col = "blue", lwd = 2, xlab = "Ano", ylab = "Log Passageiros",
     main = "Série Linear: log(AirPassengers)")
abline(lm(log_air ~ time(log_air)), col = "red", lty = 2)

# Série não linear
plot(nottem, col = "darkgreen", lwd = 2, xlab = "Ano", ylab = "Temperatura (F)",
     main = "Série Não Linear: nottem")
lines(lowess(time(nottem), nottem), col = "orange", lty = 2)

par(mfrow = c(1, 1))  # Reset layout
```

Linear x Não Linear

```r
# 2. DECOMPOSIÇÕES

# Decomposição da série linear
decomp_linear <- decompose(log_air)
# Decomposição da série não linear
decomp_nao_linear <- decompose(nottem)

# Mostrar decomposições uma abaixo da outra
par(mfrow = c(2, 1))
plot(decomp_linear, col = "blue", xlab = "Ano")
mtext("Decomposição: Série Linear (log AirPassengers)", side = 3, line = 0.5)

plot(decomp_nao_linear, col = "darkgreen", xlab = "Ano")
mtext("Decomposição: Série Não Linear (nottem)", side = 3, line = 0.5)

par(mfrow = c(1, 1))  # Reset layout
```

Linear x Não Linear

- No primeiro par de gráficos, compare:
  - A **reta vermelha** em log(AirPassengers) mostra que **uma tendência linear é um bom ajuste**
  - A curva em nottem (linha laranja) mostra que **não há tendência clara**, e o comportamento é **mais oscilante**
- Na decomposição:
  - AirPassengers apresenta **tendência e sazonalidade bem definidas**.
  - nottem tem **sazonalidade suave**, mas **tendência quase ausente**, com variação menos previsível

Série Sazonal e não sazonal

```r
# Definir semente para reprodutibilidade
set.seed(42)

# Criar sequência de tempo (ex: 5 anos mensais)
tempo <- 1:60  # 60 meses = 5 anos

# Série não sazonal: tendência + ruído
serie_nao_sazonal <- 0.5 * tempo + rnorm(60, mean = 0, sd = 3)

# Série sazonal: tendência + padrão cíclico + ruído
sazonalidade <- 10 * sin(2 * pi * tempo / 12)  # ciclo anual (12 meses)
serie_sazonal <- 0.5 * tempo + sazonalidade + rnorm(60, mean = 0, sd = 3)

# Gráficos lado a lado
par(mfrow = c(1, 2))  # Dois gráficos por linha

# Gráfico da série não sazonal
plot(tempo, serie_nao_sazonal, type = "l", col = "blue", lwd = 2,
     main = "Série Não Sazonal", xlab = "Tempo (meses)", ylab = "Valor")
abline(lm(serie_nao_sazonal ~ tempo), col = "red", lty = 2)

# Gráfico da série sazonal
plot(tempo, serie_sazonal, type = "l", col = "darkgreen", lwd = 2,
     main = "Série Sazonal", xlab = "Tempo (meses)", ylab = "Valor")
lines(lowess(tempo, serie_sazonal), col = "orange", lty = 2)

par(mfrow = c(1,1))  # Reset layout
```

Série Sazonal e não sazonal

```r
# Carregar a base AirPassengers (já vem com o R)
data("AirPassengers")

# Criar série não sazonal simulada com tendência e ruído
set.seed(123)
tempo <- 1:length(AirPassengers)
serie_nao_sazonal <- 2 * tempo + rnorm(length(tempo), mean = 0, sd = 20)
ts_nao_sazonal <- ts(serie_nao_sazonal, start = c(1949, 1), frequency = 12)

# Gráficos lado a lado: Sazonal x Não Sazonal
par(mfrow = c(1, 2))

# Série Sazonal: AirPassengers
plot(AirPassengers, main = "Série Sazonal: AirPassengers",
     col = "darkgreen", lwd = 2, ylab = "Nº de Passageiros", xlab = "Ano")

# Série Não Sazonal
plot(ts_nao_sazonal, main = "Série Não Sazonal (Simulada)",
     col = "blue", lwd = 2, ylab = "Valor Simulado", xlab = "Ano")

# Reset layout
par(mfrow = c(1, 1))
```

1.Primeiro gráfico: comparação entre série sazonal (AirPassengers) e não sazonal (simulada com ruído e tendência linear)

2.Segundo gráfico: decomposição da série AirPassengers, usando o log para que a decomposição aditiva funcione melhor (a série é originalmente **multiplicativa**)

Série Sazonal e não sazonal

```r
# Decomposição da série AirPassengers (aditiva)
# Como AirPassengers tem crescimento exponencial, aplicamos log para aproximar da aditividade
log_air <- log(AirPassengers)
decomp <- decompose(log_air, type = "additive")
```

```r
# Gráfico da decomposição
plot(decomp, col = "darkred", xlab = "Ano")
```

- **Painel 1 (Observado)**: Série original, já com crescimento e variação sazonal.
- **Painel 2 (Tendência)**: Mostra o crescimento do número de passageiros ao longo do tempo.
- **Painel 3 (Sazonalidade)**: Repetição **cíclica mensal**, destacando a **alta temporada**.
- **Painel 4 (Resíduo)**: Variações que **não são explicadas** pela tendência nem pela sazonalidade.

Univariada x Multivariada

- **Série temporal univariada**: acompanha apenas uma variável ao longo do tempo
- **Série temporal multivariada**: acompanha **várias variáveis relacionadas** no tempo
- Em **multivariada**, você pode investigar **correlações temporais** entre variáveis (ex: temperatura alta = umidade baixa?)

Os **componentes** de uma série temporal — tendência, sazonalidade e ruído — são **extraídos** de **uma série univariada**

```r
# Preparação
set.seed(123)
tempo <- 1:100

# Série univariada: exemplo de temperatura
temperatura <- 20 + 0.1 * tempo + rnorm(100, 0, 1)
ts_temperatura <- ts(temperatura, start = c(2020, 1), frequency = 12)

# Série multivariada: temperatura e umidade
umidade <- 70 - 0.05 * tempo + rnorm(100, 0, 2)
ts_multi <- ts(cbind(Temperatura = temperatura, Umidade = umidade),
               start = c(2020, 1), frequency = 12)

# Gráficos lado a lado
par(mfrow = c(1, 2))

# Série univariada
plot(ts_temperatura, col = "blue", lwd = 2,
     main = "Série Temporal Univariada",
     ylab = "Temperatura (°C)", xlab = "Tempo")
legend("topleft", legend = "Temperatura", col = "blue", lty = 1, bty = "n")
text(2024, max(ts_temperatura), "1 variável ao longo do tempo", col = "blue")

# Série multivariada
plot(ts_multi, main = "Série Temporal Multivariada",
     col = c("blue", "darkgreen"))
legend("topright", legend = c("Temperatura", "Umidade"),
       col = c("blue", "darkgreen"), lty = 1, bty = "n")

par(mfrow = c(1,1))  # Reset layout
```

Univariada x Multivariada

```r
# Dataset real disponível no R
data("airquality")

# Preprocessamento: remover NAs e criar um vetor de datas
dados <- na.omit(airquality)  # remove linhas com NA
tempo <- seq.Date(from = as.Date("1973-05-01"), by = "day", length.out = nrow(dados))

# Criar série univariada (apenas Ozone)
ozone_ts <- ts(dados$Ozone, start = c(1973, 5), frequency = 30)  # aprox. diário

# Criar série multivariada com 4 variáveis
multi_ts <- ts(cbind(Ozone = dados$Ozone,
                     Solar = dados$Solar.R,
                     Wind = dados$Wind,
                     Temp = dados$Temp),
               start = c(1973, 5), frequency = 30)

# Gráficos lado a lado
par(mfrow = c(1, 2))

# Série Univariada
```

Univariada x Multivariada

```r
# Série Univariada
plot(ozone_ts, col = "blue", lwd = 2,
     main = "Série Temporal Univariada",
     ylab = "Ozone (ppb)", xlab = "Dias desde Maio/1973")
legend("topright", legend = "Ozone", col = "blue", lty = 1, bty = "n")
text(1973.3, max(ozone_ts), "1 variável: Ozone", col = "blue")

# Série Multivariada
plot(multi_ts, main = "Série Temporal Multivariada",
     col = c("blue", "orange", "darkgreen", "red"))
legend("topright", legend = c("Ozone", "Solar.R", "Wind", "Temp"),
       col = c("blue", "orange", "darkgreen", "red"), lty = 1, bty = "n")

par(mfrow = c(1, 1))  # Reset layout
```

Correlação

•É possível observar **picos simultâneos** ou **comportamentos opostos** (ex: dias quentes = menos vento e mais ozônio?)
•Use as funções cor() ou pairs() para explorar relações entre variáveis

```r
# Garantir dados sem NA
dados <- na.omit(airquality)

# Extrair séries
ozone <- dados$Ozone
temp <- dados$Temp

# Correlação cruzada: Ozone vs Temp
ccf(temp, ozone, lag.max = 15, main = "Correlação Cruzada: Temp x Ozone")
```

Predição em séries temporais

Previsão (ou predição) em séries temporais consiste em utilizar dados passados para **estimar valores futuros** de uma variável que varia ao longo do tempo

Objetivos principais:

- Identificar padrões (tendência, sazonalidade, ciclos)
- Gerar projeções futuras com base nesses padrões
- Auxiliar decisões em áreas como economia, logística, finanças, saúde, etc.

Principais Bibliotecas em R para Predição

Biblioteca	Descrição	Uso típico
forecast	Biblioteca clássica com modelos ARIMA e ETS	Projetos estatísticos tradicionais
prophet	Desenvolvida pelo Facebook para séries com sazonalidade	Séries com feriados e eventos
fable	Abordagem moderna baseada em tidyverse	Modelos ARIMA, ETS, NNETAR
tsibble	Estrutura de dados tidy para séries temporais	Manipulação e organização

Modelos

1.ARIMA (AutoRegressive Integrated Moving Average)

- Captura tendência e autocorrelação
- Muito usado quando a série **não tem sazonalidade forte**
- Função: auto.arima() ou ARIMA() no fable

2.ETS (Exponential Smoothing)

- Usa suavização exponencial para tendência e sazonalidade
- Bom para **dados com padrões claros e suaves**
- Função: ets() ou ETS() no fable

3.Prophet

- Permite decomposição aditiva/multiplicativa com **sazonalidade flexível**
- Ideal para séries com **feriados, promoções, picos de eventos**
- Função: prophet()

Modelos

```r
library(forecast)
modelo <- auto.arima(AirPassengers)
previsao <- forecast(modelo, h = 12)
plot(previsao)
```

```r
# Transformar AirPassengers em data.frame com data explícita
df_air <- data.frame(
  ds = seq(as.Date("1949-01-01"), by = "month", length.out = length(AirPassengers)),
  y = as.numeric(AirPassengers)
)

# Verificar as primeiras linhas
head(df_air)
```

```r
install.packages("prophet")
library(prophet)

modelo <- prophet(df_air)  # Ajuste do modelo
futuro <- make_future_dataframe(modelo, periods = 12, freq = "month")  # 12 meses à frente
previsao <- predict(modelo, futuro)

# Plot da previsão
plot(modelo, previsao)
```

Modelos

```
# Instalar pacotes, se necessário:
#install.packages(c("forecast", "fpp2", "fpp3","prophet", "tsibble",
"fable", "lubridate", "tibble"))
# -------------------------
# 1. forecast::auto.arima() e forecast()
# -------------------------
library(forecast)
data(AirPassengers)
serie <- AirPassengers
modelo_arima <- auto.arima(serie)
previsao_arima <- forecast(modelo_arima, h = 12)
plot(previsao_arima, main = "ARIMA com forecast (AirPassengers)")
# -------------------------
# 2. forecast::ets() - Suavização Exponencial
# -------------------------
modelo_ets <- ets(serie)
previsao_ets <- forecast(modelo_ets, h = 12)
plot(previsao_ets, main = "ETS com forecast (AirPassengers)")
```

```r
# ---------------------------
# 3. prophet::Prophet
# ---------------------------
library(prophet)
library(tibble)
# Convertendo série mensal para formato de dataframe
df_prophet <- data.frame(
  ds = seq(as.Date("1949-01-01"), by = "month",
length.out = length(serie)),
  y = as.numeric(serie)
)
modelo_prophet <- prophet(df_prophet)
futuro <- make_future_dataframe(modelo_prophet, periods
= 12, freq = "month")
previsao <- predict(modelo_prophet, futuro)
plot(modelo_prophet, previsao)
```

```
# -------------------------
# 4. fable::ARIMA e ETS com tsibble
# -------------------------
library(fpp3)
library(tsibble)
library(fable)
library(lubridate)
# Usar a base "aus_production" que já é tsibble
aus_production <-aus_production
# Selecionar série de produção de cerveja
cerveja <- aus_production %>% filter(Quarter >= yearquarter("2000 Q1")) %>% select(Quarter, Beer)
# Modelo ARIMA com fable
modelo_arima_fable <- cerveja %>%
  model(ARIMA = ARIMA(Beer))
previsao_arima_fable <- forecast(modelo_arima_fable, h = "2 years")
autoplot(previsao_arima_fable)
# Modelo ETS com fable
modelo_ets_fable <- cerveja %>%
  model(ETS = ETS(Beer))
previsao_ets_fable <- forecast(modelo_ets_fable, h = "2 years")
autoplot(previsao_ets_fable)
```