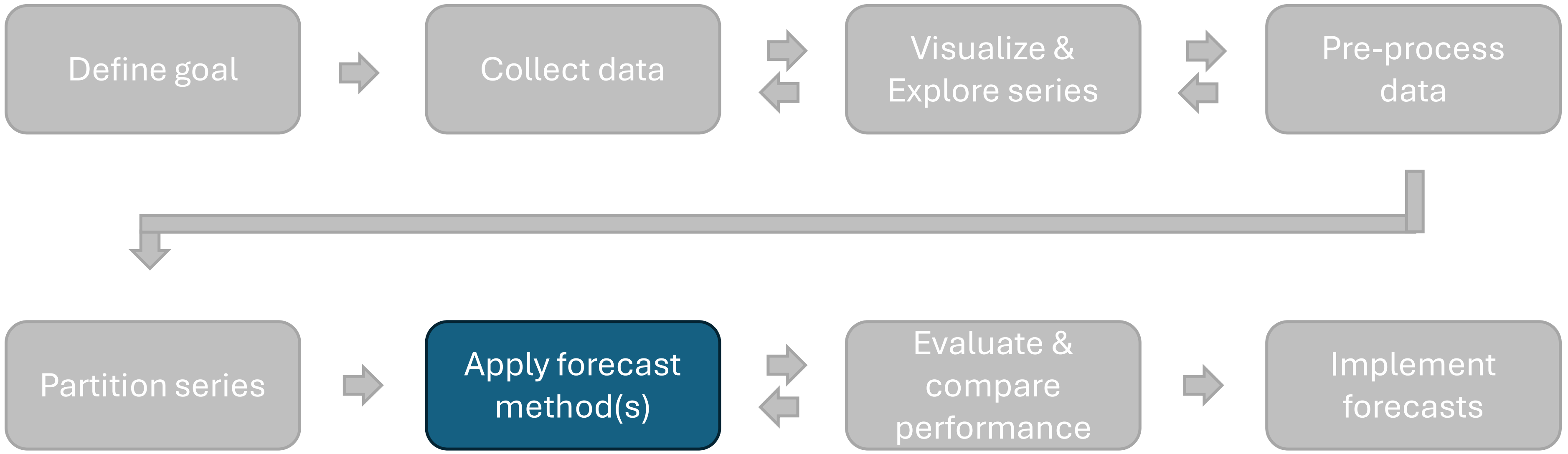
Time series Forecasting

Part 4: ARIMA

The forecasting process



We will cover

- Autocorrelation
- Autoregressive models
- Two-layer models

Autocorrelation

Autocorrelation measures how strong the values of a time series are related to their own past values.
i.e., compute the **correlation** between values in different periods.

- Lag(1) autocorrelation = correlation between (y_1, y_2, …, y_{t-1}) and (y_2,y_3,…, y_t)

$$\begin{tikzcd}[column sep=small]
(y_t, \arrow[r, bend right] & y_{t-1}, \arrow[r, bend right] & y_{t-2}, \arrow[r, bend right] & y_{t-3}, & \cdots, & y_3, \arrow[r, bend right] & y_2, \arrow[r, bend right] & y_1)
\end{tikzcd}$$

- Lag(k) autocorrelation = correlation between (y_1, y_2, …, y_{t-k}) and (y_{k+1},y_{k+2},…,y_t)

$$\begin{tikzcd}[column sep=small]
(y_t, \arrow[rrrrr, bend right] & y_{t-1}, \arrow[rrrrr, bend right] & y_{t-2}, \arrow[rrrrr, bend right] & y_{t-3}, \arrow[rrrrr, bend right] & \cdots, & y_{t-k}, & y_{t-k-1}, & y_{t-k-2}, & y_{t-k-3}, & \cdots, & y_3, & y_2, & y_1)
\end{tikzcd}$$

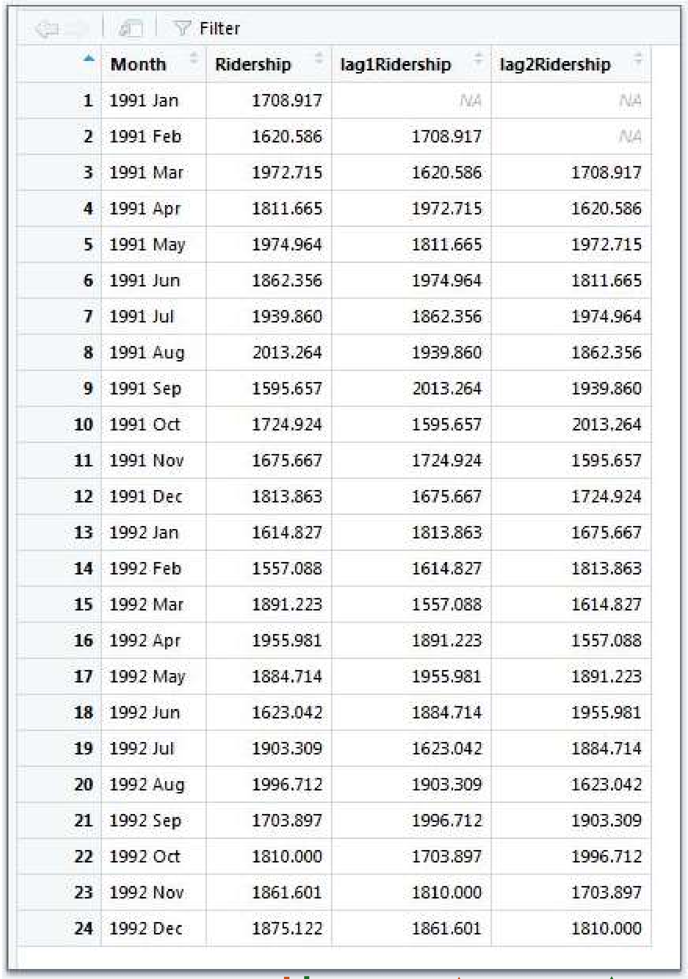
Note: autocorrelation measures **linear** relationship

Autocorrelation

Uses

- Check forecast errors for independence
- Model remaining information
- Evaluate predictability

Autocorrelation



	Month	Ridership	lag1Ridership	lag2Ridership
1	1991 Jan	1708.917	NA	NA
2	1991 Feb	1620.586	1708.917	NA
3	1991 Mar	1972.715	1620.586	1708.917
4	1991 Apr	1811.665	1972.715	1620.586
5	1991 May	1974.964	1811.665	1972.715
6	1991 Jun	1862.356	1974.964	1811.665
7	1991 Jul	1939.860	1862.356	1974.964
8	1991 Aug	2013.264	1939.860	1862.356
9	1991 Sep	1595.657	2013.264	1939.860
10	1991 Oct	1724.924	1595.657	2013.264
11	1991 Nov	1675.667	1724.924	1595.657
12	1991 Dec	1813.863	1675.667	1724.924
13	1992 Jan	1614.827	1813.863	1675.667
14	1992 Feb	1557.088	1614.827	1813.863
15	1992 Mar	1891.223	1557.088	1614.827
16	1992 Apr	1955.981	1891.223	1557.088
17	1992 May	1884.714	1955.981	1891.223
18	1992 Jun	1623.042	1884.714	1955.981
19	1992 Jul	1903.309	1623.042	1884.714
20	1992 Aug	1996.712	1903.309	1623.042
21	1992 Sep	1703.897	1996.712	1903.309
22	1992 Oct	1810.000	1703.897	1996.712
23	1992 Nov	1861.601	1810.000	1703.897
24	1992 Dec	1875.122	1861.601	1810.000

First 24 months of Amtrak ridership series, lag-1 series, and lag-2 series

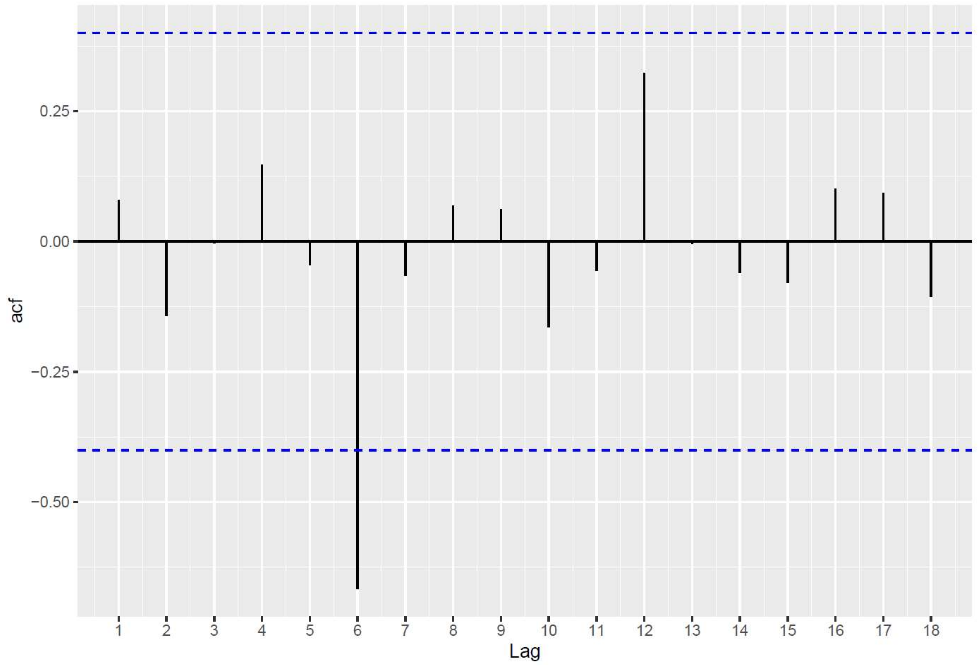
Correlation between the original series and the lag-1 series (e.g., via the R function ***cor()***) to be 0.08.

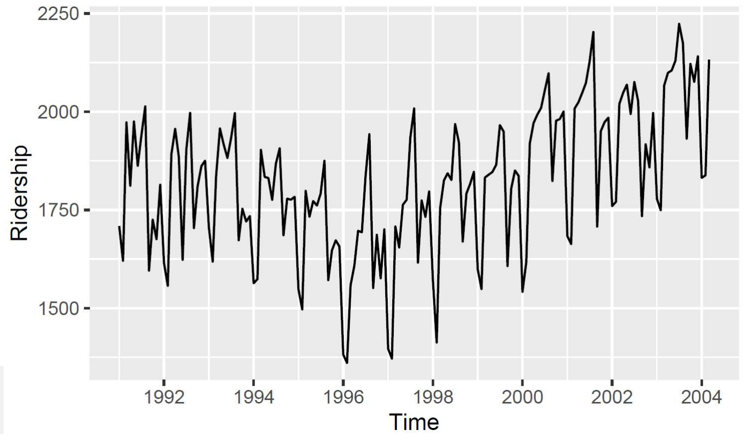
The correlation between the original series and the lag-2 (two time periods apart) series is -0.15.

Autocorrelation

```
ridership.24 <- ridership |> filter_index(~ "1992 Dec")

ridership.24 |> ACF(Ridership, lag_max = 18) |>
          autoplot() +
          xlab("Lag") +
          scale_x_continuous(breaks = seq(0, 18, by = 1))
```

We use *scale_x_continuous()* to add x-axis tick marks at each lag (the default displays fewer lag values)

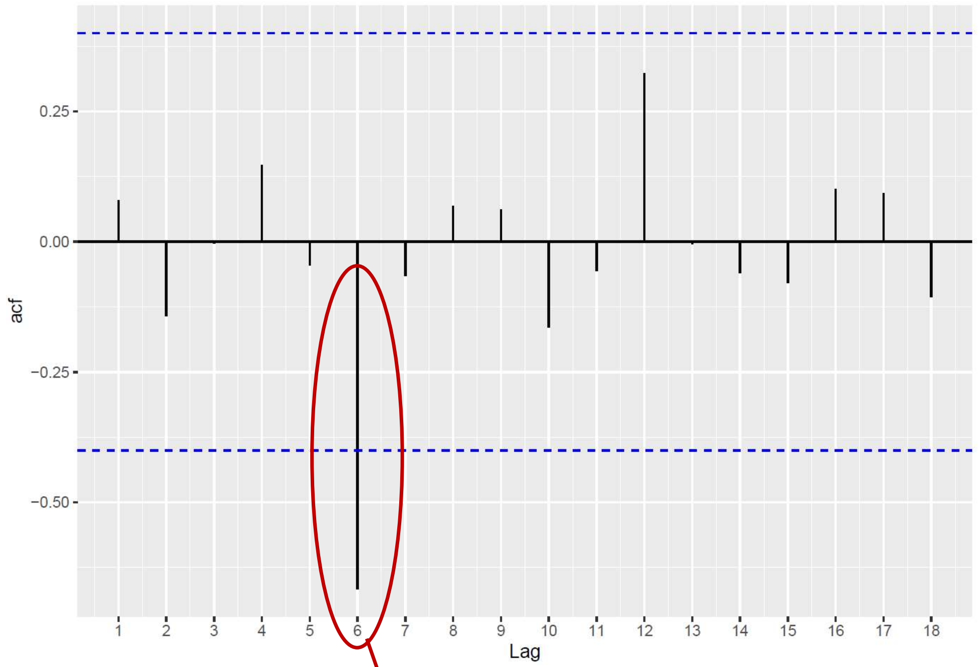


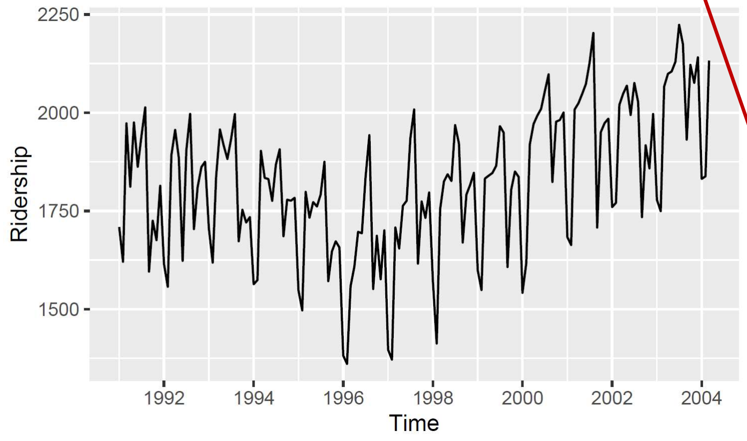


```
ridership.24 |> autoplot()
```

Autocorrelation

```
ridership.24 <- ridership |> filter_index(~ "1992 Dec")

ridership.24 |> ACF(Ridership, lag_max = 18) |>
          autoplot() +
          xlab("Lag") +
          scale_x_continuous(breaks = seq(0, 18, by = 1))
```



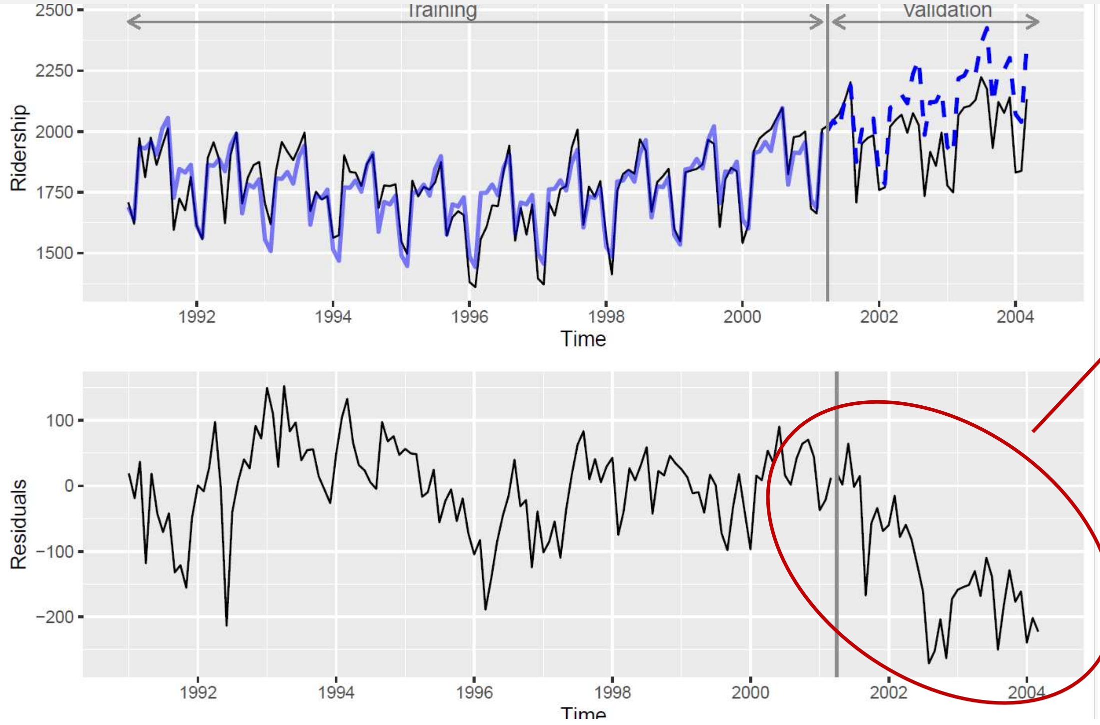


This indicates a biannual pattern in ridership, with 6-month switches from high to low ridership.
A look at the time plot confirms the high-summer low-winter pattern.

What about the residuals

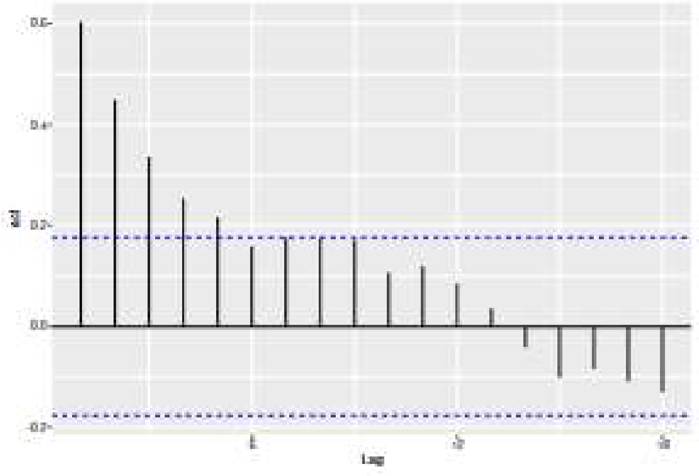
- Let’s examine the **residuals** from Model 5 (ESM additive seasonality + quadratic trend)

```
> train.lm.trend.season <- train.ridership |>
                              model(TSLM(Ridership ~ trend() + I(trend()^2) + season()))
```



If we have adequately modelled the original time series (captured seasonality, e.g.), the residual series should show no autocorrelation at the season's lag.

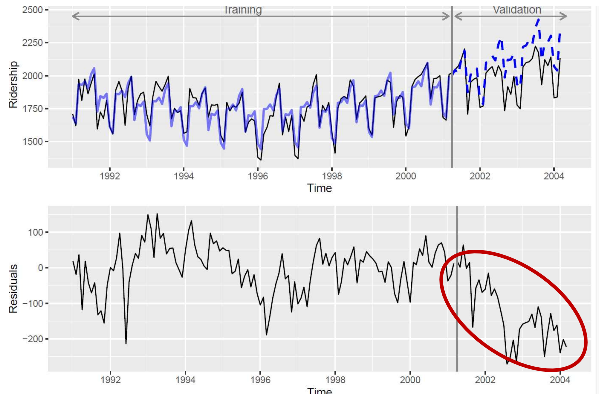
But here we can see a clear trend!



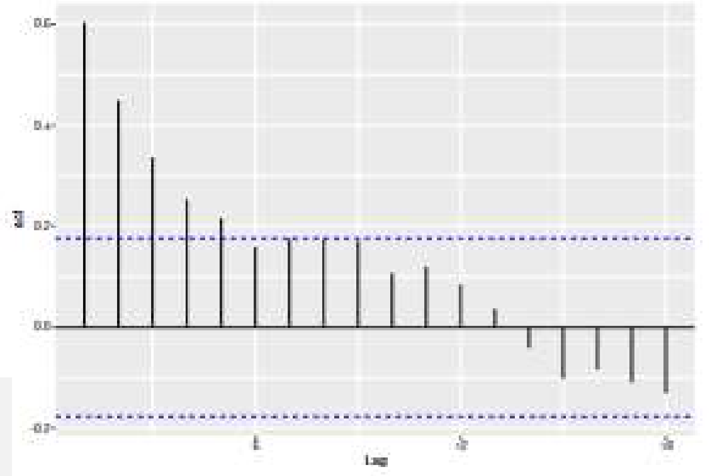
What about the residuals

- Let’s examine the **residuals** from Model 5 (ESM additive seasonality + quadratic trend)

```
> train.lm.trend.season <- train.ridership |>
                                              model(TSLM(Ridership ~ trend() + I(trend()^2) + season()))
```



Let’s look at the autocorrelation plot of the residuals.



```
train.lm.trend.season |> residuals() |>
                ACF(.resid, lag_max = 18) |>
                autoplot() +    xlab("Lag")
```

It is clear that the 6-month (and 12-month) cyclical behaviour no longer dominates the series of residuals, indicating that the regression model adequately captured it. **But,** we can also see a strong positive autocorrelation from lag-1, indicating a positive relationship between neighbouring residuals.

This is valuable information, which can be used to improve forecasting.

Typical autocorrelation behavior

- Positive lag-1 autocorrelation (**“stickiness”)**:
  high values usually immediately follow **high values**, and
  low values usually immediately follow **low values**

- Negative lag-1 autocorrelation (**“swings”**):
  high values usually immediately follow **low values** and
  low values usually immediately follow **high values**

- High positive autocorrelation at multiples of a certain lag (e.g. lags 4, 8, 12…) indicates **seasonality**.

How to model?

Forecasting with 2-level models (manually)

Level 1:	Model /method applied to raw data produces forecasts + forecast errors.
Level 2:	AR applied to forecast errors, produces errors-of-forecast-errors.

ARIMA models (AutoRegressive Integrated Moving Average)

ARMA

- Directly models the autocorrelation of the series values and the autocorrelations of the forecast errors.

$$y_t = \underbrace{\alpha}_{\boxed{\text{Level}}} + \underbrace{\beta_1\, y_{t-1} + \beta_2\, y_{t-2} + \cdots + \beta_p\, y_{t-p}}_{\boxed{\begin{array}{l}\text{AR(P), autoregressive part}\\ \text{Captures the autocorrelations of the}\\ \text{series values at lags } 1,2,\cdots,\boldsymbol{p}.\end{array}}} + \underbrace{\varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q}}_{\boxed{\begin{array}{l}\text{MA(q), moving average part}\\ \text{autocorrelation terms for the forecast}\\ \text{errors (at error lags } 1,2,\cdots,\boldsymbol{q}\text{).}\end{array}}}$$

AR and ARMA models can only be fitted to data without trend or seasonality!

ARIMA

- The ARIMA model incorporates a preliminary step of differencing, which removes trend.
- **I** is the differencing operation (Integrated) in ARIMA.
- The order of differencing is denoted by parameter *d*.
- *d* indicates how many rounds of lag-1 differencing are performed
  - *d=0* no differencing, which is suitable if the series lacks a trend
  - *d=1* differencing the series once (at lag-1), remove a linear trend
  - *d=2* differencing the series twice (each time at lag-1), remove a quadratic trend
- Similarly, a seasonal-ARIMA model incorporates a step of differencing to remove seasonality before applying ARIMA model.

ARIMA Method

The Idea:
Add lags of the series and/or lags of the forecast errors to capture all forms of autocorrelation

Uses:
Capture/model autocorrelation directly.

Advantages:
Large set of more flexible models

Disadvantages:
Require stationary series = data with no patterns (trend, seasonality)
Requires much statistical expertise

Key concept:
Using past values of the original data and past errors

Estimating ARIMA model

- ARIMA models require selecting the values of *p,d,q* and then the software estimates the β and θ parameters.
- The ARIMA() function in the *fable package* allows the user to specify *p,d,q* values (large values will result in long-running time).
- If *p,d,q* values are left blank , the software will search for the p,d,q that results in the model with the best AIC, AICc, or BIC.
- Note, ARIMA() function can also fit seasonal-ARIMA models.
- Seasonal ARIMA models contain an extra set of parameters *P,D,Q* that specify
    - the number of seasonal autoregressive lags (*P*),
    - seasonal differencing order (*D*), and
    - number of seasonal moving average lags (*Q*).

Currently the limit for ARIMA is $p + q + P + Q \leq 6$ and constant $d + D \leq 2$

Estimating ARIMA model: Souvenir Sales Example

- SouvenirSales.xls contains monthly sales for a souvenir shop at a beach resort town in Queensland, Australia, between 1995 and 2001.
- Aim: forecast sales for the next 12 months (year 2002)

1. Split the data into training and validation periods (year 2001)
2. Using the training period, fit the automated ARIMA model on sales and *log(sales)*.
3. What are the values of p, d, q and P, D, Q?
4. Use the selected models to forecast sales. What were the sales in February 2001 as predicted by each model?
5. Compare the predictive performance of the two auto-selected ARIMA models by comparing performance metrics and by examining performance charts. Is one of the models preferable to the other? If so, which one and how have you determined this?

Estimating ARIMA model: Souvenir Sales Example

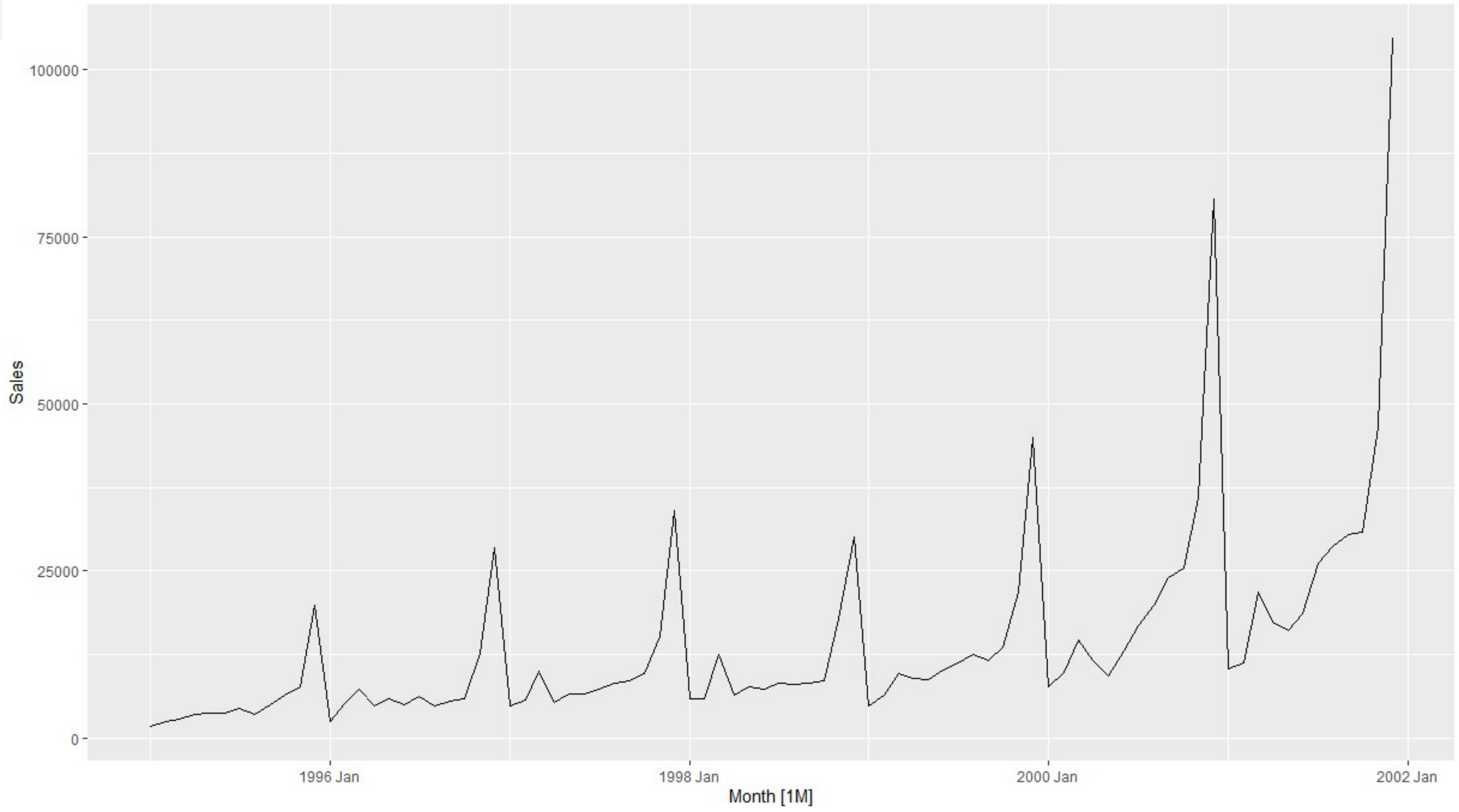
- SouvenirSales.xls contains monthly sales for a souvenir shop at a beach resort town in Queensland, Australia, between 1995 and 2001.
- Aim: forecast sales for the next 12 months (year 2002)

1. Split the data into training and validation periods (year 2001)

```
SouvenirSales <- read.csv("C:/…/Data/SouvenirSales.csv")

SouvenirSales <- SouvenirSales |>
        dplyr::mutate(Month = yearmonth(as.character(SouvenirSales$Month)) ) |>
        as_tsibble(index = Month)

SouvenirSales.train <- SouvenirSales |> filter_index(~ "2000 Dec")
SouvenirSales.valid  <- SouvenirSales |> filter_index("2001 Jan" ~ .)
```

Estimating ARIMA model: Souvenir Sales Example

```
SouvenirSales |> autoplot()
```



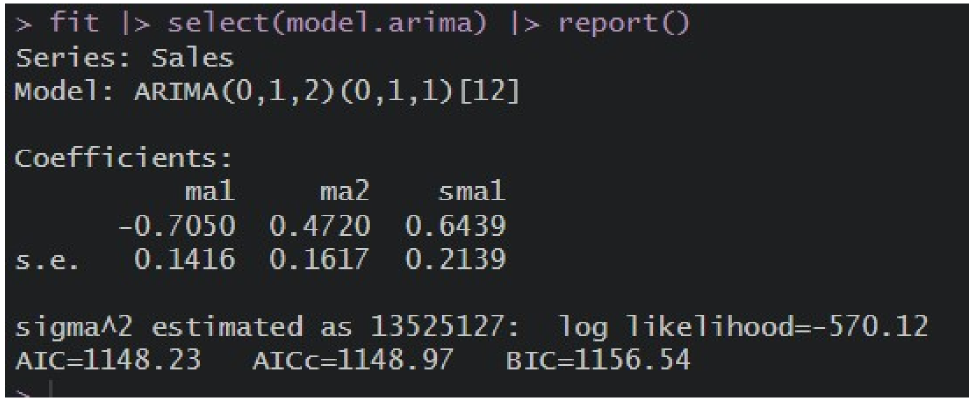
Estimating ARIMA model: Souvenir Sales Example

Using the training period, fit the automated ARIMA model on sales and *log(sales)*.

```
fit <- SouvenirSales.train |>
            model(
                        model.arima = ARIMA(Sales),
                        model.arima.log = ARIMA(log(Sales))
                      )
```

Estimating ARIMA model: Souvenir Sales Example

What are the values of p, d, q and P, D, Q?
ARIMA on **Seles**

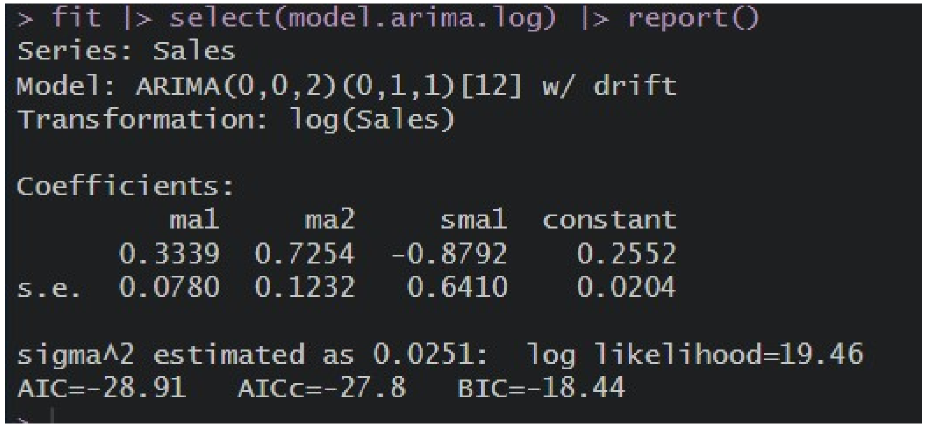


```
> fit |> select(model.arima) |> report()
Series: Sales
Model: ARIMA(0,1,2)(0,1,1)[12]

Coefficients:
          ma1     ma2    sma1
      -0.7050  0.4720  0.6439
s.e.   0.1416  0.1617  0.2139

sigma^2 estimated as 13525127:  log likelihood=-570.12
AIC=1148.23   AICc=1148.97   BIC=1156.54
>
```

It means MA(2) with
lag-1 differencing,
seasonal differencing, lag-12, and
a seasonal MA(1).
This model does not have a constant!

Estimating ARIMA model: Souvenir Sales Example

What are the values of p, d, q and P, D, Q?
ARIMA on **log(Seles)**



```
> fit |> select(model.arima.log) |> report()
Series: Sales
Model: ARIMA(0,0,2)(0,1,1)[12] w/ drift
Transformation: log(Sales)

Coefficients:
         ma1     ma2     sma1  constant
      0.3339  0.7254  -0.8792    0.2552
s.e.  0.0780  0.1232   0.6410    0.0204

sigma^2 estimated as 0.0251:  log likelihood=19.46
AIC=-28.91   AICc=-27.8   BIC=-18.44
>
```

It means MA(2) with
~~lag-1 differencing~~,
seasonal differencing, lag-12, and
a seasonal MA(1).
This model ~~does not~~ have a constant!

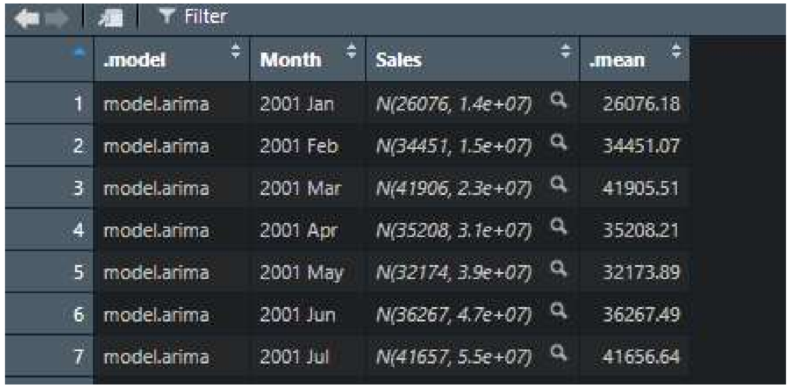
Estimating ARIMA model: Souvenir Sales Example

Use the selected models to forecast sales. What were the sales in February 2001 as predicted by each model?

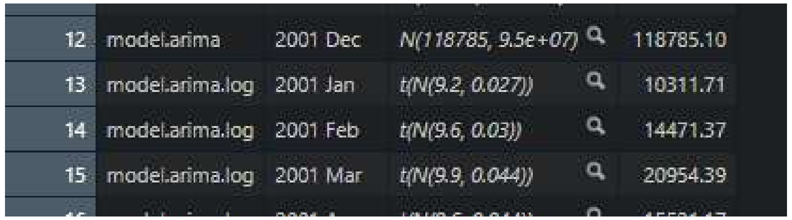
```
fc <- fit |> forecast(h=12)
fc |> View()
```

The forecast for Feb 2001 is AUD 34,451

The forecast for Feb 2001 is AUD 14,471. This is substantially lower than the first forecast (60% lower).



	.model	Month	Sales	.mean
1	model.arima	2001 Jan	N(26076, 1.4e+07)	26076.18
2	model.arima	2001 Feb	N(34451, 1.5e+07)	34451.07
3	model.arima	2001 Mar	N(41906, 2.3e+07)	41905.51
4	model.arima	2001 Apr	N(35208, 3.1e+07)	35208.21
5	model.arima	2001 May	N(32174, 3.9e+07)	32173.89
6	model.arima	2001 Jun	N(36267, 4.7e+07)	36267.49
7	model.arima	2001 Jul	N(41657, 5.5e+07)	41656.64

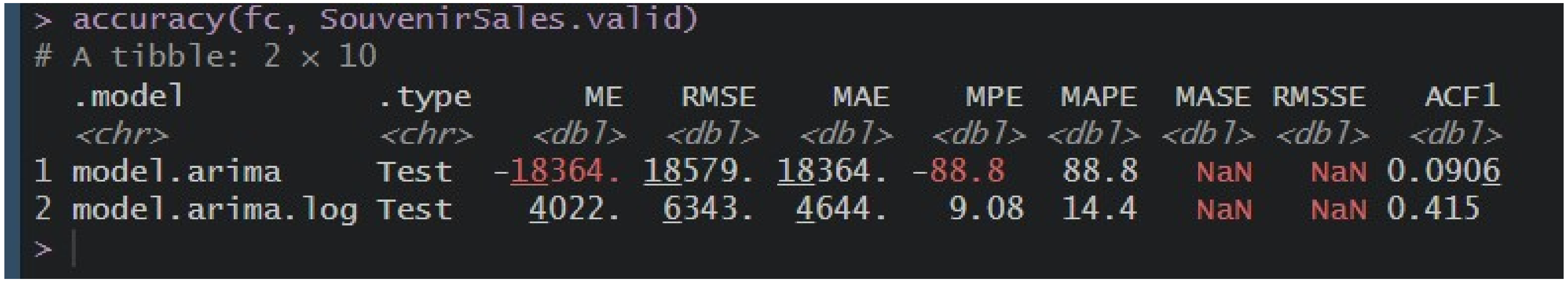


12	model.arima	2001 Dec	N(118785, 9.5e+07)	118785.10
13	model.arima.log	2001 Jan	t(N(9.2, 0.027))	10311.71
14	model.arima.log	2001 Feb	t(N(9.6, 0.03))	14471.37
15	model.arima.log	2001 Mar	t(N(9.9, 0.044))	20954.39

Estimating ARIMA model: Souvenir Sales Example

- Compare the predictive performance of the two auto-selected ARIMA models, by comparing performance metrics and by examining performance charts. Is one of
- the models preferable to the other? If so, which one and how have you determined this?

```
# accuracy (on validation period)
accuracy(fc, SouvenirSales.valid)
```



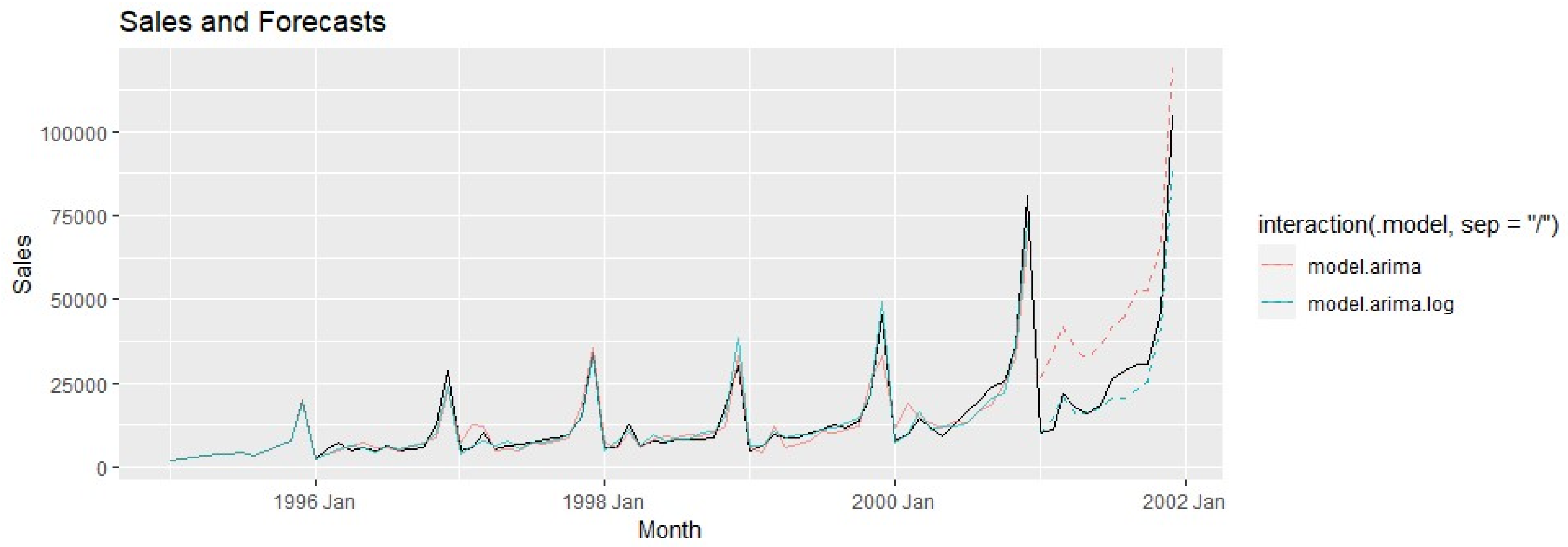
```
> accuracy(fc, SouvenirSales.valid)
# A tibble: 2 × 10
  .model           .type      ME   RMSE    MAE    MPE  MAPE  MASE RMSSE   ACF1
  <chr>            <chr>   <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl>  <dbl>
1 model.arima      Test  -18364. 18579. 18364. -88.8   88.8   NaN   NaN 0.0906
2 model.arima.log  Test    4022.  6343.  4644.   9.08  14.4   NaN   NaN 0.415
>
```

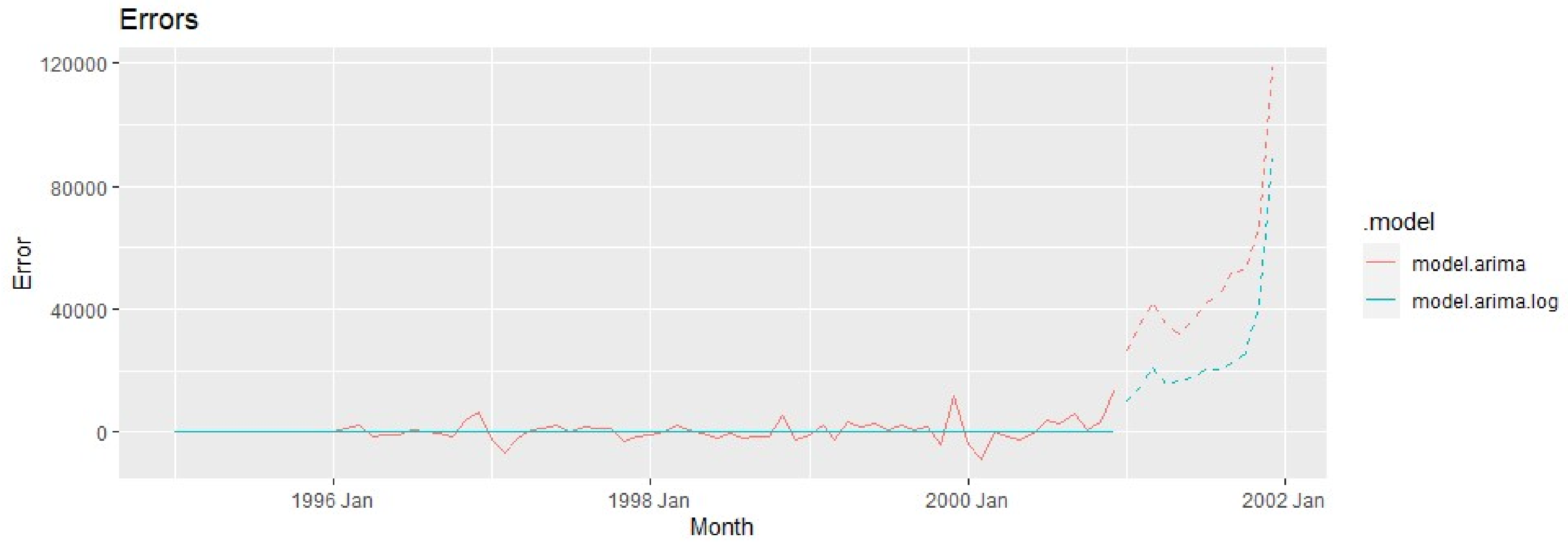
RMSE (root mean squared error),
MAE (mean absolute error) and
MAPE (mean absolute percentage error)
Are smaller for the log(sale) model => log(sale) model performs better.

Estimating ARIMA model: Souvenir Sales Example

```
fc2 <- fc |> group_by(.model) |>
        left_join(SouvenirSales.valid, by = "Month") |>
        mutate(fc.error = Sales.y - .mean)

# Plot 1: actual and forecasts
p1 <- autoplot(SouvenirSales, Sales) +
        autolayer(fitted(fit), .fitted, alpha = 0.7) +
        autolayer(fc2, .mean, linetype = "dashed", level = NULL) +
        labs(title = "Sales and Forecasts", x = "Month", y = "Sales")

# Plot 2: errors
p2 <- autoplot(fc2, series="fc.error", linetype = "dashed") +
        autolayer(resid(fit), .resid) +
        labs(title = "Errors", x = "Month", y = "Error")

grid.arrange(p1, p2 , nrow = 2)
```

ARIMA


Sales and Forecasts
Sales
100000
75000
50000
25000
0
1996 Jan
1998 Jan
2000 Jan
2002 Jan
Month
interaction(.model, sep = "/")
model.arima
model.arima.log


Errors
Error
120000
80000
40000
0
1996 Jan
1998 Jan
2000 Jan
2002 Jan
Month
.model
model.arima
model.arima.log