Applied Economic Forecasting

7. ARIMA Models

Section 1

Stationarity and Differencing

Stationarity

Definition

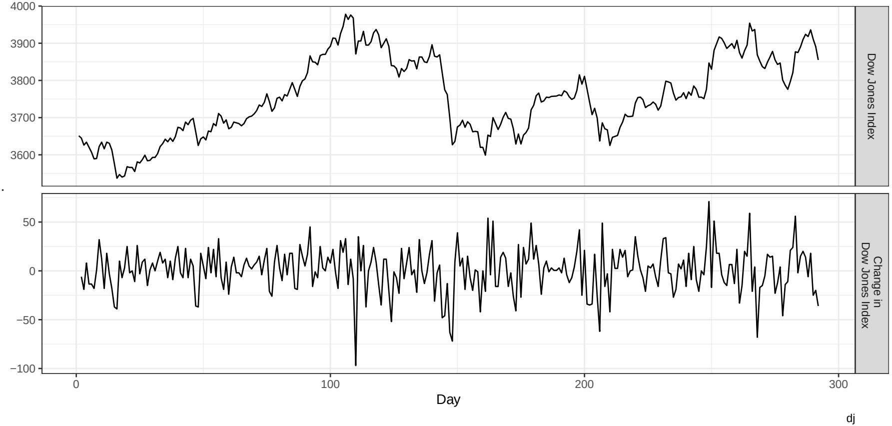
If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \dots, y_{t+s})$ does not depend on t.

A **stationary series** is:

- constant mean
- constant variance
- no patterns predictable in the long-term

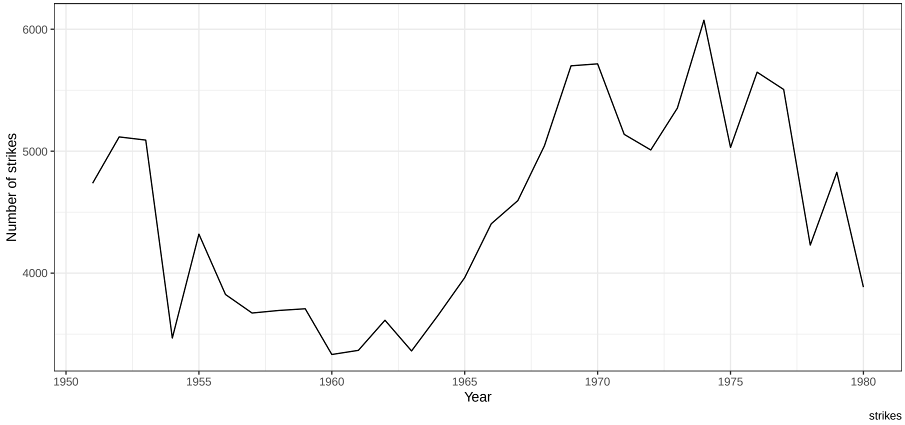
Stationary?

```r
cbind("Dow Jones Index" = dj,
      "Change in \n Dow Jones Index" = diff(dj)) %>%
  autoplot(facets= TRUE) + labs(x = "Day", caption = "dj")
```



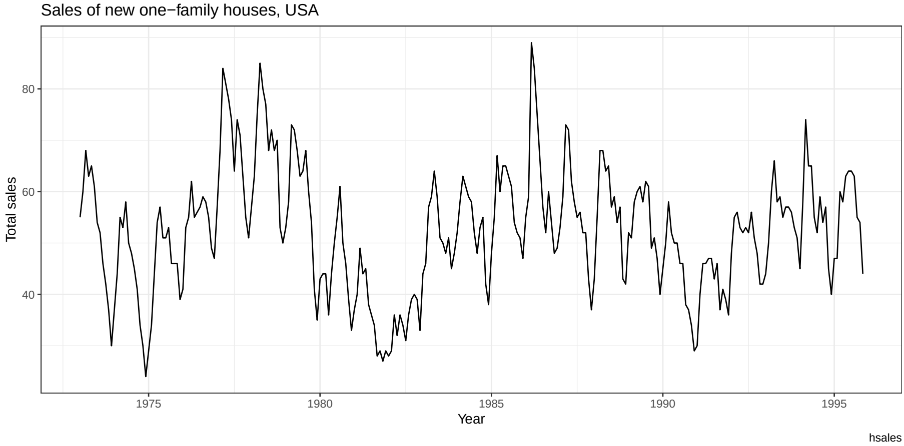
Stationary?

```
autoplot(strikes) + labs(y = "Number of strikes", x = "Year",
        caption = "strikes")
```



Stationary?

```
autoplot(hsales) + labs(title = "Sales of new one-family houses, USA",
  x = "Year", y ="Total sales", caption = "hsales")
```

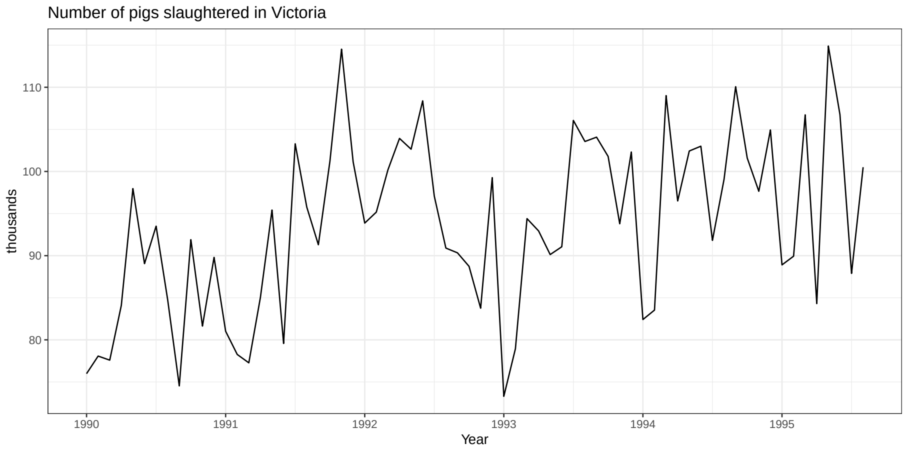


Stationary?

```r
autoplot(eggs) + labs(title ="Price of a dozen eggs in 1993 dollars",
  xlab = "Year", y = "$",caption = "eggs")
```

Stationary?

```r
autoplot(window(pigs/1e3, start=1990)) +
  labs(x = "Year", y = "thousands",
       title = "Number of pigs slaughtered in Victoria")
```

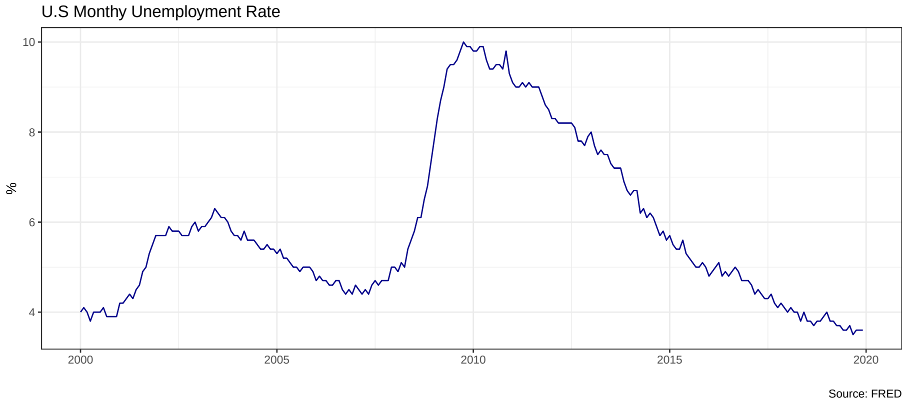


Stationary?

```r
quantmod::getSymbols(Symbols = "UNRATE", src = "FRED")
```

```
## [1] "UNRATE"
```

```r
UNRATE["2000::2019"] %>% ts(start = c(2000,1), frequency = 12) %>%
  autoplot(color= "blue4") + labs(x = "", y = "%",
    title = "U.S Monthy Unemployment Rate", caption = "Source: FRED")
```



Stationarity

Definition

If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \dots, y_{t+s})$ does not depend on t.

Transformations help to **stabilize the variance**.

For ARIMA modeling, we also need to **stabilize the mean**.

Non-stationarity in the mean

Identifying non-stationary series

- time plot.
- The ACF of stationary data drops to zero relatively quickly
- The ACF of non-stationary data decreases slowly.
- For non-stationary data, the value of r_1 is often large and positive.

Differencing (Recap)

- Differencing helps to **stabilize the mean**.
- The differenced series is the *change* between each observation in the original series: $\Delta y_t = y_t - y_{t-1}$.
- The differenced series will have only $T - 1$ values since it is not possible to calculate a difference Δy_1 for the first observation.

Second-order differencing

Occasionally the differenced data will not appear stationary and it may be necessary to difference the data a second time:

$$\begin{aligned}\Delta^{(2)} y_t &= \Delta y_t - \Delta y_{t-1} \\ &= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \\ &= y_t - 2y_{t-1} + y_{t-2}.\end{aligned}$$

- $\Delta^{(2)} y_t$ will have $T-2$ values.
- In practice, it is almost never necessary to go beyond second-order differences.

Seasonal differencing (Recap)

A seasonal difference is the difference between an observation and the corresponding observation from the previous year.

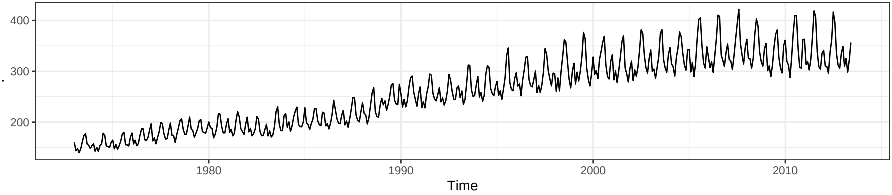
$$\Delta_m y_t = y_t - y_{t-m}$$

where $m =$ number of seasons.

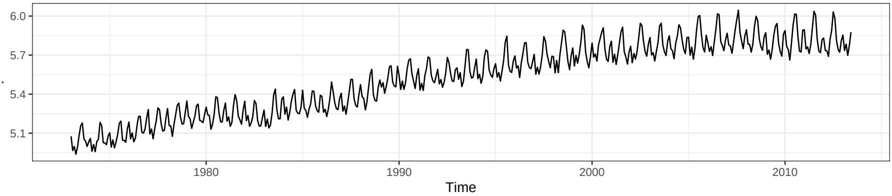
- For monthly data $m = 12$.
- For quarterly data $m = 4$.

Electricity production

```r
usmelec %>% autoplot()
```

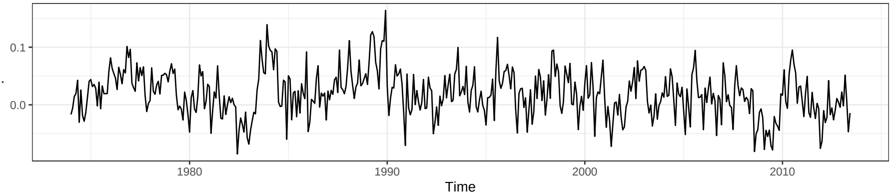



```r
usmelec %>% log() %>% autoplot()
```

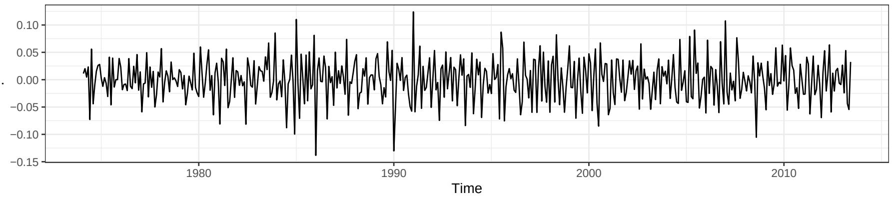


Electricity production

```r
usmelec %>% log() %>% diff(lag=12) %>% autoplot()
```




```r
usmelec %>% log() %>% diff(lag=12) %>% diff(lag=1) %>%
  autoplot()
```



Electricity production

- Seasonally differenced series is closer to being stationary.
- Remaining non-stationarity can be removed with further first difference.

If $\Delta_m y_t = y_t - y_{t-12}$ denotes seasonally differenced series, then twice-differenced series `i`

$$
\begin{aligned}
y_t^* &= \Delta_m y_t - \Delta_m^{(2)} y_{t-1} \\
&= (y_t - y_{t-12}) - (y_{t-1} - y_{t-13}) \\
&= y_t - y_{t-1} - y_{t-12} + y_{t-13}\,. \\
\text{Generic Form } y_t^* &= y_t - y_{t-1} - y_{t-m} + y_{t-m-1}
\end{aligned}
$$

Seasonal differencing (Recap)

When both seasonal and first differences are applied...

- it makes no difference which is done first—the result will be the same.
- If seasonality is strong, we recommend that seasonal differencing be done first because sometimes the resulting series will be stationary and there will be no need for further first difference.

It is important that if differencing is used, the differences are interpretable.

Interpretation of differencing

- first differences are the change between **one observation and the next**;
- seasonal differences are the change between **one year to the next**.

But taking lag 3 differences for yearly data, for example, results in a model which cannot be sensibly interpreted.

Unit root tests

Statistical tests to determine the required order of differencing.

1. Augmented Dickey Fuller (ADF) test **null hypothesis is that the data are non-stationary and non-seasonal.**
2. Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test **null hypothesis is that the data are stationary and non-seasonal.**
3. Other tests available for seasonal data.

ADF test

General Specification

$$\Delta Y_t = a_0 + \gamma Y_{t-1} + \phi t + \sum_{i=2}^{p} \beta_i \Delta Y_{t-(i-1)} + \varepsilon_t$$

where $\gamma = -\left(1 - \sum_{i=1}^{p} a_i\right)$ and $\beta_i = -\sum_{j=1}^{p} a_j$

```
ur.df(y,type = "trend") %>% summary()
# with a drift and trend
ur.df(y,type = "drift") %>% summary()
# with a drift
ur.df(y,type = "none") %>% summary()
# without a drift and trend
```

ADF test

Decision Rules

Model	H_0	Test Statistic	Crit. Value (95% & 99%)	Decision Rule
$\Delta y_t = a_0 + \gamma y_{t-1} + \phi t + \varepsilon_t$	$\gamma = 0$	τ_3	-3.42; -3.13	Reject H_0 if $t_{stat} < t_{crit}$
	$a_0 = \gamma = 0$	ϕ_2	4.71; 4.05	Reject H_0 if $t_{stat} > t_{crit}$
	$a_0 = \gamma = a_2 = 0$	ϕ_3	6.30; 5.36	Reject H_0 if $t_{stat} > t_{crit}$
$\Delta y_t = a_0 + \gamma y_{t-1} + \varepsilon_t$	$\gamma = 0$	τ_2	-2.87; -2.57	Reject H_0 if $t_{stat} < t_{crit}$
	$a_0 = \gamma = 0$	ϕ_1	4.61; 3.79	Reject H_0 if $t_{stat} > t_{crit}$
$\Delta y_t = \gamma y_{t-1} + \varepsilon_t$	$\gamma = 0$	τ	-1.95; -1.62	Reject H_0 if $t_{stat} < t_{crit}$

KPSS test

```r
ur.kpss(goog) %>% summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 7 lags.
##
## Value of test-statistic is: 10.72
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

```r
ndiffs(goog)
```

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

Automatically selecting differences

STL decomposition: $y_t = T_t + S_t + R_t$

Seasonal strength $F_s = \max\left(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(S_t + R_t)}\right)$

If $F_s > 0.64$, do one seasonal difference.

```
usmelec %>% log() %>% nsdiffs()
```

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

```
usmelec %>% log() %>% diff(lag=12) %>% ndiffs()
```

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

Your turn

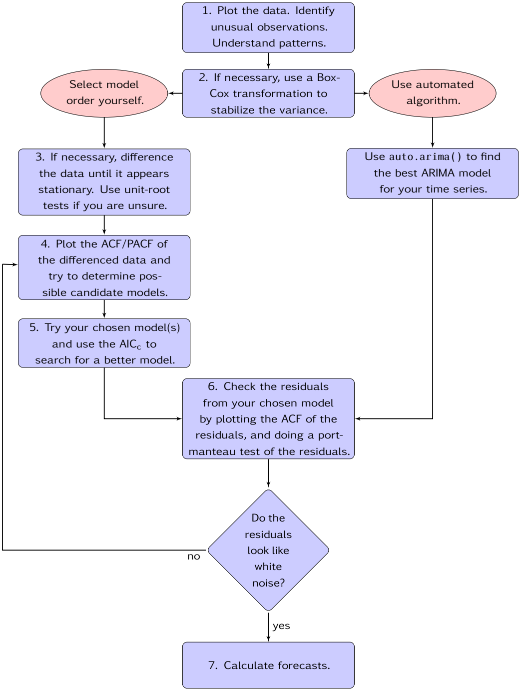
For the `visitors` series, find an appropriate differencing (after transformation if necessary) to obtain stationary data.

Section 2

ARIMA Modeling

Box-Jenkins Methodology

The Box-Jenkins methodology refers to a set of procedures for identifying, fitting, and checking ARIMA models with time series data. Forecasts follow directly from the form of the fitted model.


1. Plot the data. Identify unusual observations. Understand patterns.
2. If necessary, use a Box-Cox transformation to stabilize the variance.
Select model order yourself.
Use automated algorithm.
3. If necessary, difference the data until it appears stationary. Use unit-root tests if you are unsure.
Use auto.arima() to find the best ARIMA model for your time series.
4. Plot the ACF/PACF of the differenced data and try to determine possible candidate models.
5. Try your chosen model(s) and use the AIC$_c$ to search for a better model.
6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals.
Do the residuals look like white noise?
no
yes
7. Calculate forecasts.

Box-Jenkins Methodology

- Autoregressive integrated moving average (ARIMA) models
  - a class of linear models that are capable of representing ***stationary*** as well as ***nonstationary*** time series.
  - it generates forecasts based on a description of ***historical patterns in the data itself,***

$$Y_t = \beta_0 + \underbrace{\beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_k Y_{t-p}}_{\text{autoregressive terms}}$$

$$+ \varepsilon_t \underbrace{-\omega_1 \varepsilon_{t-1} - \omega_2 \varepsilon_{t-2} - \cdots - \omega_q \varepsilon_{t-q}}_{\text{moving average terms}}$$

Nonstationary time series data

1. The mean of the data is function of time

$$E(Y_t) = a + b \cdot t$$

so-called first moment nonstationary.

- A unit root (random walk)

$$Y_t = Y_{t-1} + \varepsilon_t$$

Stationary time series data

- The mean of the data is independent of time (i.e., constant),

$$E\left(Y_t\right) = a$$

so-called first moment stationary.

- The initial selection of an ARIMA model is based on

  1. Autocorrelation function (ACF) ==> **Will help us to determine the moving average component.**

$$r_k = \frac{cov\left(Y_t, Y_{t-k}\right)}{\sqrt{var\left(Y_t\right)}\sqrt{var\left(Y_{t-k}\right)}}$$

2. Partial autocorrelation function (PACF) ==> **Will help us to determine the autoregressive component.**

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_k Y_{t-k} + \varepsilon_t$$

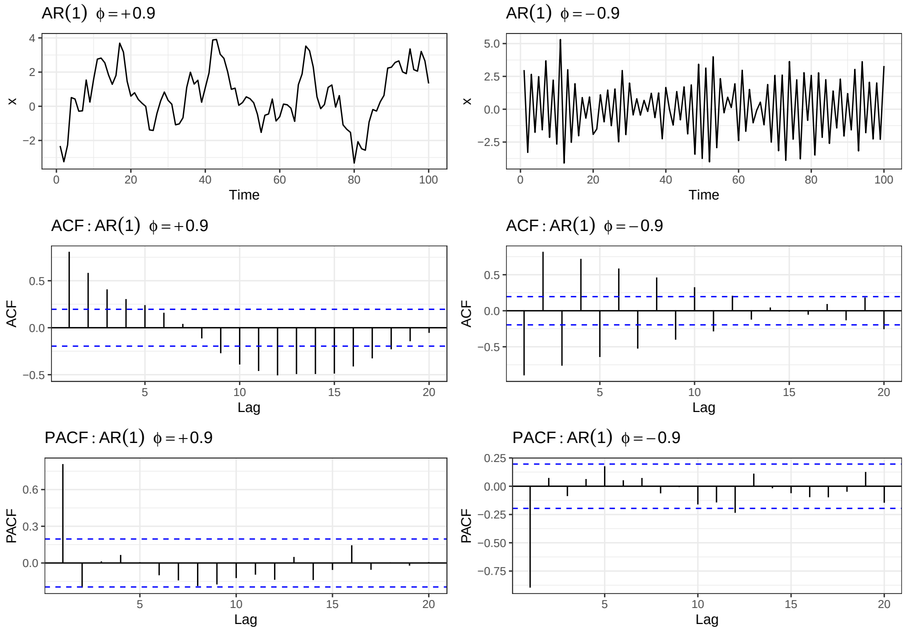
such that

$$r_k = \beta_k$$

Box-Jenkins Methodology

Distance k	Regression	Partial Autocorrelation coefficient r_k
1	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \varepsilon_t$	β_1
2	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \varepsilon_t$	β_2
3	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \beta_3 Y_{t-3} + \varepsilon_t$	β_3
...	...	...
k	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_k Y_{t-k} + \varepsilon_t$	β_k

Sample of an AR(1) Process

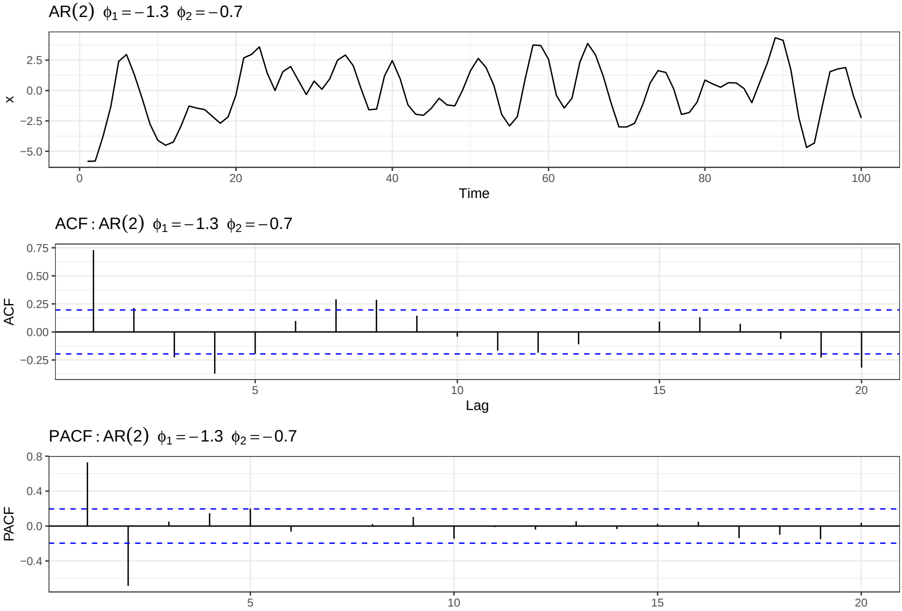


AR(1) model

$$y_t = c + \phi_1 y_{t-1} + \varepsilon_t$$

- When $\phi_1 = 0$, y_t is **equivalent to WN**
- When $\phi_1 = 1$ and $c = 0$, y_t is **equivalent to a RW**
- When $\phi_1 = 1$ and $c \neq 0$, y_t is **equivalent to a RW with drift**
- When $\phi_1 < 0$, y_t tends to **oscillate between positive and negative values**.

Sample of an AR(2) Process



Stationarity conditions

We normally restrict autoregressive models to stationary data, and then some constraints on the values of the parameters are required.

General condition for stationarity

Complex roots of

$$1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$$

lie outside the unit circle on the complex plane.

- For $p = 1$: $-1 < \phi_1 < 1$.
- For $p = 2$:

$$-1 < \phi_2 < 1 \qquad \phi_2 + \phi_1 < 1 \qquad \phi_2 - \phi_1 < 1.$$

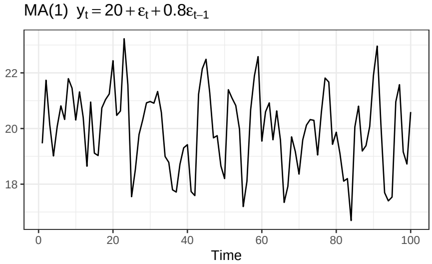
- More complicated conditions hold for $p \geq 3$.
- Estimation software takes care of this.

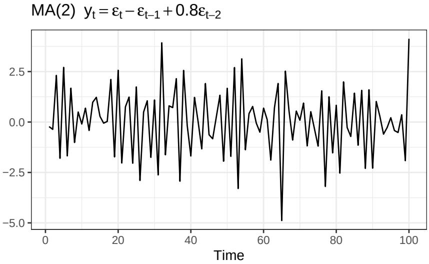
Moving Average (MA) models:

$$y_t = c + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q},$$

where ε_t is white noise. This is a multiple regression with **past *errors*** as predictors. *Don't confuse this with moving average smoothing!*

```r
set.seed(2)
p1 <- autoplot(20 + arima.sim(list(ma = 0.8), n = 100)) +
  labs(y = "", title = quote("MA(1)"~~y[t] == 20 + epsilon[t] + 0.8*epsilon[t-1]))
p2 <- autoplot(arima.sim(list(ma = c(-1, +0.8)), n = 100)) +
  labs(y = "", title=quote("MA(2)"~~y[t] == epsilon[t] - epsilon[t-1] + 0.8*epsilon[t-2]))
gridExtra::grid.arrange(p1,p2,nrow=1)
```





MA(∞) models

It is possible to write any stationary AR(p) process as an MA(∞) process.

Example: AR(1)

$$
\begin{aligned}
y_t &= \phi_1 y_{t-1} + \varepsilon_t \\
&= \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t \\
&= \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\
&= \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\
&\dots
\end{aligned}
$$

Provided $-1 < \phi_1 < 1$:

$$y_t = \varepsilon_t + \phi_1 \varepsilon_{t-1} + \phi_1^2 \varepsilon_{t-2} + \phi_1^3 \varepsilon_{t-3} + \cdots$$

Invertibility

- Any MA(q) process can be written as an AR(∞) process if we impose some constraints on the MA parameters.
- Then the MA model is called "invertible".
- Invertible models have some mathematical properties that make them easier to use in practice.
- Invertibility of an ARIMA model is equivalent to forecastability of an ETS model.

Invertibility

General condition for invertibility

Complex roots of $1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$ lie outside the unit circle on the complex plane.

- For $q = 1$: $-1 < \theta_1 < 1$.
- For $q = 2$:
  $-1 < \theta_2 < 1 \qquad \theta_2 + \theta_1 > -1 \qquad \theta_1 - \theta_2 < 1.$
- More complicated conditions hold for $q \geq 3$.
- Estimation software takes care of this.

Identifying AR and MA Models

	Autocorrelations	*Partial Autocorrelations*
MA(q)	Cut off after the order q of the process	Die out
AR(p)	Die out	Cut off after the order p of the process
ARMA(p, q)	Die out	Die out

Let us explore some simulations and test our theory!

ARIMA models

Autoregressive Moving Average models:

$$y_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} \\ + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t.$$

- Predictors include both **lagged values of y_t and lagged errors.**
- Conditions on coefficients ensure stationarity.
- Conditions on coefficients ensure invertibility.

Autoregressive Integrated Moving Average models

- Combine ARMA model with **differencing**.

ARIMA models

Autoregressive Integrated Moving Average models

ARIMA(p, d, q) model

AR:	$p =$ order of the autoregressive part
I:	$d =$ degree of first differencing involved
MA:	$q =$ order of the moving average part.

- White noise model: ARIMA(0,0,0)
- Random walk: ARIMA(0,1,0) with no constant
- Random walk with drift: ARIMA(0,1,0) with const.
- AR(p): ARIMA(p,0,0)
- MA(q): ARIMA(0,0,q)

Implementing the ARIMA Model

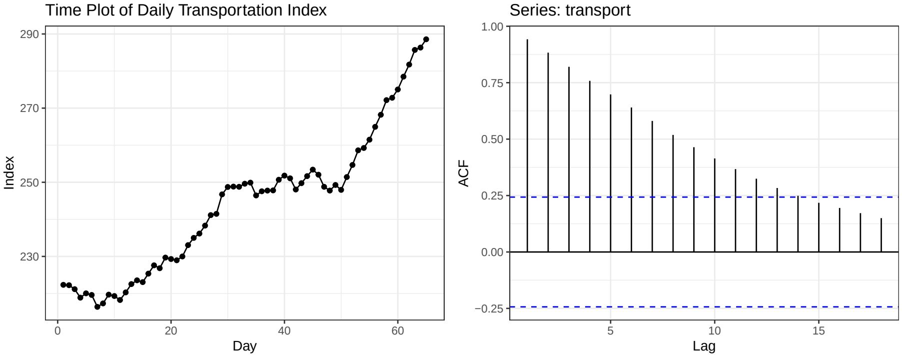
Steps for the Box-Jenkins Method:

1. Selecting an initial model (model identification).
2. Estimating the model coefficients (parameter estimation).
3. Analyzing the residuals (Model Checking).
4. Modifying the model if necessary.
5. Repeating (1)-(4) until no further modification.
6. Using the final model for forecasting.

Forecasting with the Model (Transport Data)

Step 1: Plot the data

```
transport <- ts(readxl::read_xls("Transport.xls"))
q1 <-  autoplot(transport) + labs(y = "Index", x = "Day",
  title = "Time Plot of Daily Transportation Index") + geom_point()
q2 <- ggAcf(transport)
gridExtra::grid.arrange(q1,q2,ncol =2)
```



Forecasting with the Model (Transport Data)

Step 1: Plot the data

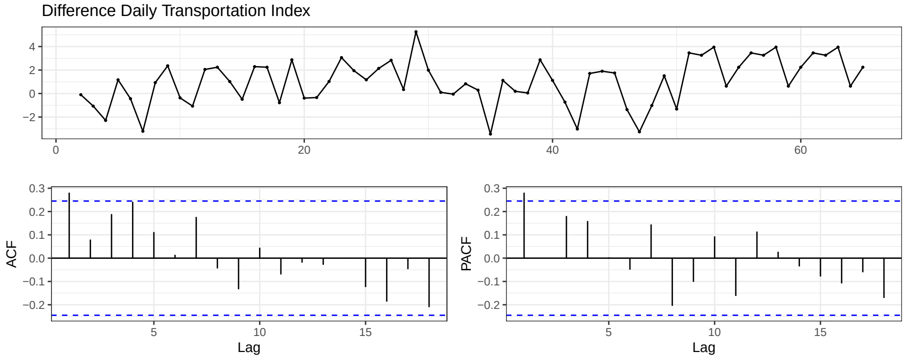
The initial observations from the graphs are:

1. The data has an upward trend.
2. The sample autocorrelations are persistently large and decay to zero slowly. You therefore conclude that the data is nonstationary with nonconstant mean.

Forecasting with the Model (Transport Data)

Step 2: You decide to difference the data in hopes of making it stationary.

```
transport %>% diff() -> dtransport
dtransport %>% ggtsdisplay(main = "Difference Daily Transportation Index")
```



Forecasting with the Model (Transport Data)

The observations from the graphs of the differenced series are:

1. The data fluctuate around a constant mean.
2. ACF shows that only the lag 1 autocorrelation is significant.
3. The ACFs appear to cut off after lag 1.
4. PACF also shows that only the lag 1 partial autocorrelation is significant.
5. The PACFs appear to cut off after lag 1.
6. Neither ACFs nor PACFs appear to die out in a declining manner.

Forecasting with the Model (Transport Data)

Competing Models

We can conclude that

1. the observations from 2 and 3 indicates MA(1).
2. the observations from 4 and 5 indicates AR(1).
3. ARIMA(1,1,0), ARIMA(0,1,1) and ARIMA(1,1,1) can be considered as the initially identified models.

$$ARIMA(1,1,0): \ \Delta Y_t = \phi_0 + \phi_1 \Delta Y_{t-1} + \varepsilon_t \tag{1}$$

$$ARIMA(0,1,1): \ \Delta Y_t = \mu + \varepsilon_t - \omega_1 \varepsilon_{t-1} \tag{2}$$

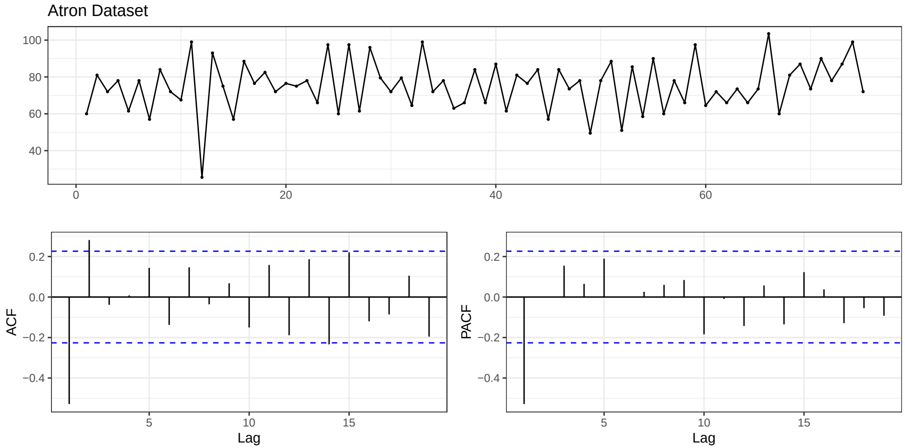
$$ARIMA(1,1,1): \ \Delta Y_t = \phi_0 + \phi_1 \Delta Y_{t-1} + \varepsilon_t - \omega_1 \varepsilon_{t-1} \tag{3}$$

Let's estimate and compare these three competing models.

```
Arima(y, order = c(p,d,q)) %>% summary()
```

Let us revisit the Atron Dataset.

```
atron <- ts(read.csv("Atron.csv"))
ggtsdisplay(atron, main = "Atron Dataset")
```



Forecasting with the Model (Atron Data)

To determine lag orders for AR and MA,

1. The PACF graph shows that the first lag is significant with the coefficient value, -0.53.
2. After the lag 1, the rest of lags are all zero (insignificant).
3. So, the PACF observations suggest an AR(1) model
4. The ACF graph shows that the first two lags are significant.
5. The rest of lags are also insignificant.
6. So, the ACF observations suggest a MA(2) model.

Competing Models

We can consider ARIMA(1,0,0), ARIMA(0,0,2), and ARIMA(1,0,2) as possible contenders.

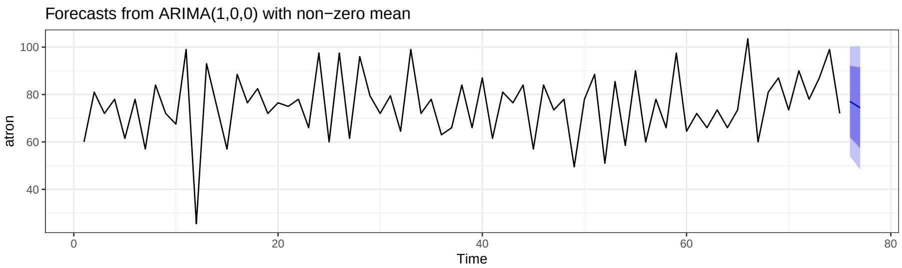
Forecasting with the Model (Atron Data)

Two step ahead forecast with the ARIMA(1,0,0) model

```
(Arima(atron, order = c(1,0,0)) %>% forecast(2))$mean
```

```
## Time Series:
## Start = 76
## End = 77
## Frequency = 1
## [1] 77.09 74.39
```

```
Arima(atron, order = c(1,0,0)) %>% forecast(2) %>% autoplot()
```



Section 3

ARIMA Models for Seasonal Data

Seasonal ARIMA Models

- Seasonal pattern in a year
  - Seasonal pattern year after year
  - Seasonal differences for nonstationary data
- The notation for a seasonal ARIMA model:

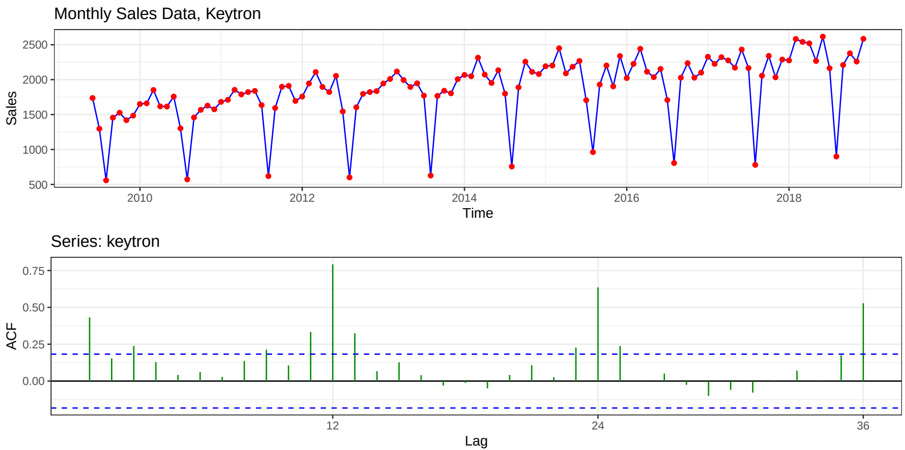
$$\text{ARIMA}(p, d, q)(P, D, Q)_m$$

  where

- p is regular AR terms
- d is regular differences
- q is regular MA terms
- P is seasonal autoregressive terms at lag m
- D is seasonal difference at lag m
- Q is seasonal MA terms at lag m.

Seasonal ARIMA Models (Keytron)

```
keytron <- ts(read.csv("Keytron.csv"),end = c(2018,12), frequency = 12)
k1 <- autoplot(keytron, colour = "blue") +
  labs(title = "Monthly Sales Data, Keytron", y= "Sales") +  geom_point(col
k2 <- ggAcf(keytron, colour = "green4", lag.max = 36)
gridExtra::grid.arrange(k1,k2,nrow = 2)
```

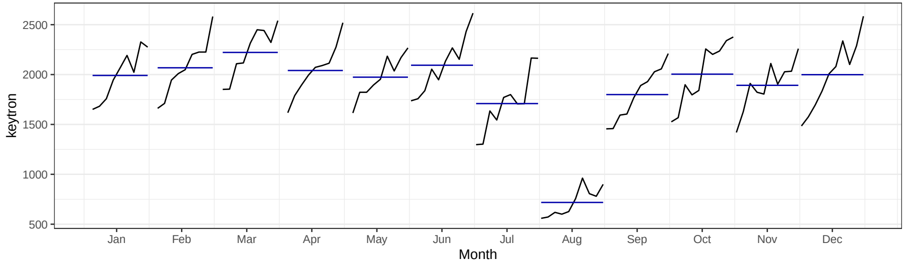


Seasonal ARIMA Models (Keytron)

The initial observation are:

- The data have an upward trend (nonstationary).
- The plot shows a pronounced seasonal pattern. We could throw in a `ggsubseriesplot` here as well.

```
ggsubseriesplot(keytron)
```



Therefore, you decide to start with a seasonal ARIMA model.

Seasonal ARIMA Models (Keytron)

From the ACF earlier, we note that:

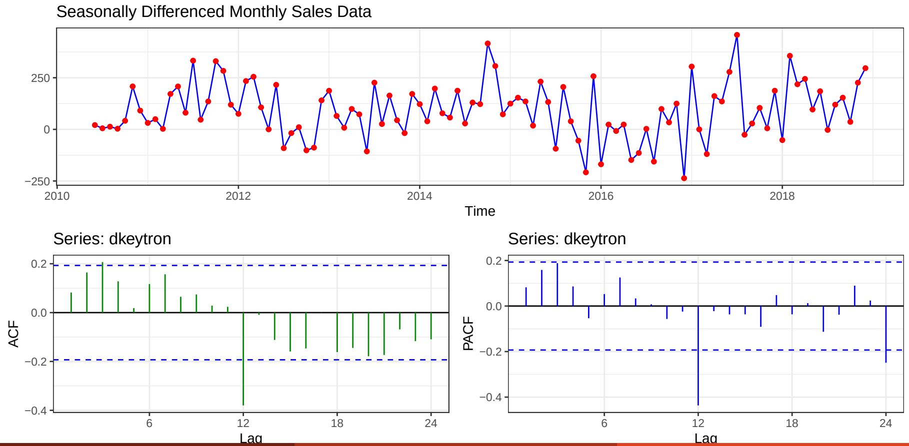
- The first lag is significant
- The ACFs appeared to cut off after lag 1.
- The ACFs are large and significant at the seasonal lags - 12, 24 and 36.

Take care of the seasonality first:

$$\Delta Y_{t,m=12} = Y_t - Y_{t-12}$$

```r
keytron %>% diff(12) -> dkeytron
k3 <- autoplot(dkeytron, colour = "blue") +
  labs(title = "Seasonally Differenced Monthly Sales Data", y = "") +
  geom_point(colour = "red")
k4 <- ggAcf(dkeytron, colour = "green4")
k5 <- ggPacf(dkeytron, colour = "blue")

grid.arrange(arrangeGrob(k3,k4,k5,layout_matrix= rbind(c(1,1), c(2,3))))
```



Seasonal ARIMA Models (Keytron)

The observations from seasonally differenced sales:

- The seasonally differenced sales are stationary and vary around a constant mean.
- The ACF plot has one significant spike at lag 12 (cuts off)
- The PACF plot has signficant spikes at 12 and 24 lags that get progressively smaller (die out).
- Overall, the plots suggest an AR(1) and MA(1) term at the seasonal lag 12.

Seasonal ARIMA Models (Keytron)

Based on those observations, we identified a model of

$$\text{ARIMA}(0,0,0)(0,1,1)_{12}$$

where $p = 0$ regular autoregressive terms.

$d = 0$ regular differences

$q = 0$ regular moving average terms

$P = 1$ seasonal autoregressive terms

$D = 1$ seasonal difference at lag 12

$Q = 1$ seasonal moving average term at 12

Seasonal ARIMA Models (Keytron)

In a regression formulation, an ARIMA$(0, 0, 0)(0, 1, 1)_{12}$ model takes the form

$$Y_t - Y_{t-12} = \phi_0 + \phi_1 \Delta Y_{t-1,S=12} + \varepsilon_t - \omega_1 \varepsilon_{t-12}$$

where ω_1 is the seasonal MA parameter (coefficient) to be estimated.

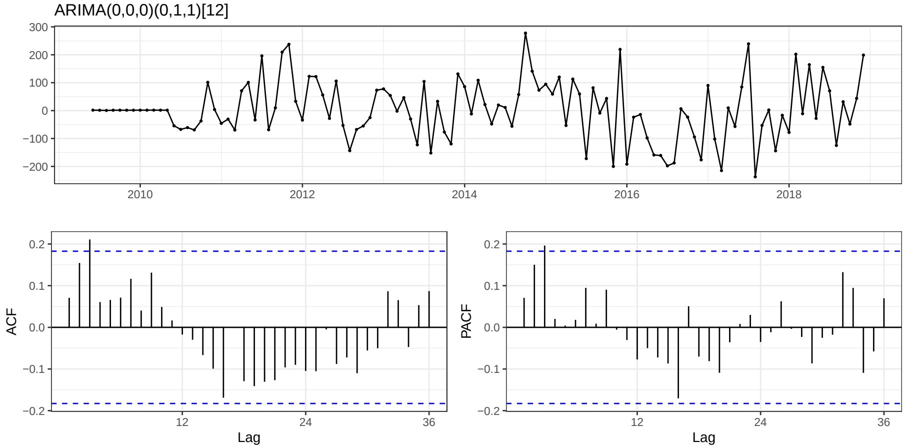
Estimating in R:

```r
keytron %>% Arima(order=c(0,0,0), seasonal = c(0,1,1),
    include.constant = TRUE) %>% summary()
```

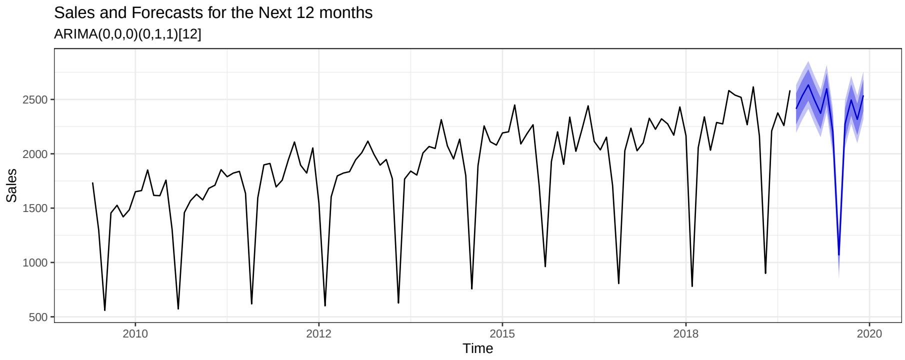
```
## Series: .
## ARIMA(0,0,0)(0,1,1)[12] with drift
##
## Coefficients:
##          sma1  drift
##        -0.668  7.183
## s.e.    0.097  0.409
##
## sigma^2 estimated as 12610:  log likelihood=-635
## AIC=1276   AICc=1276   BIC=1284
##
## Training set error measures:
##                 ME  RMSE  MAE     MPE  MAPE   MASE   ACF1
## Training set 3.094 105.2 80.3 -0.7729 4.828 0.6342 0.0708
```

Seasonal ARIMA Models (Keytron)

```
Arima(keytron,order=c(0,0,0),seasonal = c(0,1,1), include.constant = TRUE)
   residuals() %>% ggtsdisplay(main = "ARIMA(0,0,0)(0,1,1)[12]")
```



```r
Arima(keytron,order=c(0,0,0),seasonal = c(0,1,1),
  include.constant = TRUE) %>% forecast(12) %>% autoplot() +
  labs(title = "Sales and Forecasts for the Next 12 months",
  subtitle = "ARIMA(0,0,0)(0,1,1)[12]", y= "Sales") + theme_bw()
```



```r
(Arima(keytron,order=c(0,0,0),seasonal = c(0,1,1),
      include.constant = TRUE) %>% forecast(12))$mean -> fsales
fsales.m <- matrix(fsales, dimnames = list(month.abb,"Forecast"))
knitr::kable(fsales.m,
             format.args = list(big.mark = ","),
             caption = "Forecasted Sales Values for 2020")
```

Table 1: Forecasted Sales Values for 2020

	Forecast
Jan	2,413
Feb	2,533
Mar	2,634
Apr	2,496
May	2,372
Jun	2,598
Jul	2,202
Aug	1,069
Sep	2,275
Oct	2,495
Nov	2,316
Dec	2,538

Advantages and Disadvantages of ARIMA Models

- Advantages of ARIMA Models.

  – powerful and flexible for short-range forecasts

  – Formal procedures for testing model adequacy are available

  – Forecast intervals can be constructed from the fitted model

- Disadvantages of ARIMA Models

  – require a relatively large amount of data

  – no easy ways to update the parameters as new data arrive

  – Model construction and maintenance require a large investment of time and other resources.