STAT481/581: Introduction to Time Series Analysis

Ch2. Time series graphics

OTexts.org/fpp3/

Outline

Outline

Class packages

```r
# Data manipulation and plotting functions
library(tidyverse)
# Forecasting functions
library(fable)
# Time series manipulation
library(tsibble)
# Time series graphics and statistics
library(feasts)
# Functions to work with date-times
library(lubridate)
# Tidy time series data
library(tsibbledata)
```

tsibbledata datasets

1. anasett: Passenger numbers on Anasett airline flights
2. aus_livestock: Meat production in Australia for human consumption from Q3 1965 to Q4 2018.
3. aus_production: Quarterly estimates of manufacturing production of selected commodities in Australia.
4. aus_retail: Australian retail trade turnover (total value of retail traded).
5. gafa_stock: GAFA stock prices.
6. global_economy: Global economic indicators.

tsibbledata datasets

7. `hh_budget`: Household budget characteristics.
8. `nyc_bikes`: A sample from NYC Citi Bike usage of 10 bikes throughout 2018.
9. `olympic_running`: Fastest running times for Olympic races.
10. `PBS`: Monthly Medicare Australia prescription data.
11. `pelt`: Pelt trading records.
12. `vic_elec`: Half-hourly electricity demand for Victoria, Australia

`tsibble` objects

A `tsibble` allows storage and manipulation of time series in R.
A tsibble is a data- and model-oriented object.

It contains:

- Measured variable(s): numbers of interest
- Key variable(s): identifiers for each series
- An index: time information about the observation
- A tsibble is sorted by its key first and then index
- Key variable(s) together with the index uniquely identifies each record

tsibble objects example

```r
#Creating a tsibble object
library(tsibble)
e1 <- tsibble(year = 2012:2016, x = c(1,2,1,2,2),
  y = c(123,39,78,52,110), index = year, key = x)
e1
```

```
## # A tsibble: 5 x 3 [1Y]
## # Key:       x [2]
##    year     x     y
##   <int> <dbl> <dbl>
## 1  2012     1   123
## 2  2014     1    78
## 3  2013     2    39
```

tsibble objects example

```
# yearquarter returns a numeric value that can be a
e2<- tsibble(
  qtr = rep(yearquarter("2010 Q1") + 0:9, 3),
  group = rep(c("x", "y", "z"), each = 10),
  value = rnorm(30),key = group, index = qtr)
e2
```

```
## # A tsibble: 30 x 3 [1Q]
## # Key:       group [3]
##        qtr group  value
##      <qtr> <chr>  <dbl>
##  1 2010 Q1 x      0.449
##  2 2010 Q2 x      0.702
```

The tsibble index

Common time index variables can be created with these functions:

Frequency	Function
Annual	start year:end year
Quarterly	yearquarter()
Monthly	yearmonth()
Weekly	yearweek()
Daily	as_Date(), ymd()
Sub-daily	as_datetime()

Example for time index variables

```r
2015:2020
```

```
## [1] 2015 2016 2017 2018 2019 2020
```

```r
yearquarter("2010 Q1")+0:3
```

```
## [1] "2010 Q1" "2010 Q2" "2010 Q3" "2010 Q4"
```

```r
yearmonth("2010 1")+0:3
```

```
## [1] "2010 Jan" "2010 Feb" "2010 Mar" "2010 Apr"
```

Example for time index variables

```r
yearweek("2010 1")+0:3
```

```
## [1] "2009 W53" "2010 W01" "2010 W02" "2010 W03"
```

```r
as.Date("2020-01-22") + 0:3
```

```
## [1] "2020-01-22" "2020-01-23" "2020-01-24"
## [4] "2020-01-25"
```

Example for time index variables

```r
ymd("2020-01-22")+0:3
```

```
## [1] "2020-01-22" "2020-01-23" "2020-01-24"
## [4] "2020-01-25"
```

```r
as_datetime("2020-01-22 00:50:50")+0:3
```

```
## [1] "2020-01-22 00:50:50 UTC"
## [2] "2020-01-22 00:50:51 UTC"
## [3] "2020-01-22 00:50:52 UTC"
## [4] "2020-01-22 00:50:53 UTC"
```

Coerce a dataset to be an tsibbledata object

```r
olympic_running %>% as_tsibble(
  key = c(Length, Sex), index = Year)
```

```
## # A tsibble: 312 x 4 [4Y]
## # Key:         Length, Sex [14]
##      Year Length Sex    Time
##     <dbl> <fct>  <chr> <dbl>
##   1  1896 100m   men     12
##   2  1900 100m   men     11
##   3  1904 100m   men     11
```

The key to many time series

```
tsibbledata::olympic_running %>%
  group_by_key() %>%
  slice(1) %>%
  head(6) %>%
  knitr::kable(booktabs=TRUE)
```

The key to many time series

Year	Length	Sex	Time
1896	100m	men	12.0
1928	100m	women	12.2
1900	200m	men	22.2
1948	200m	women	24.4
1896	400m	men	54.2
1964	400m	women	52.0

Australian GDP

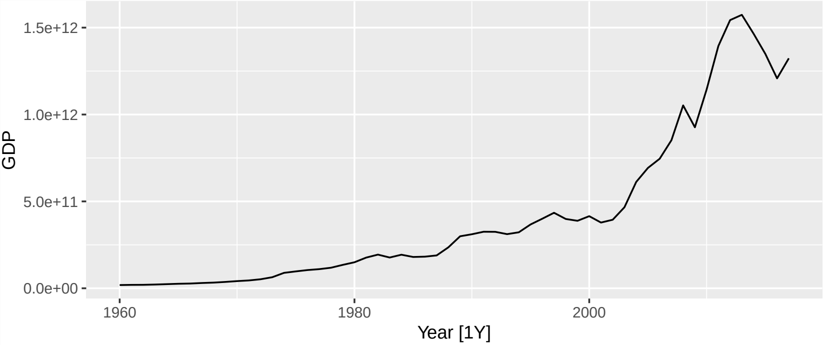
```r
#filter is to select a subset in rows
aus_economy <- global_economy %>%
  filter(Code == "AUS")
```

```
## # A tsibble: 58 x 9 [1Y]
## # Key:       Country [1]
##     Country Code   Year     GDP Growth   CPI
##     <fct>   <fct> <dbl>   <dbl>  <dbl> <dbl>
##   1 Austra~ AUS    1960 1.86e10  NA     7.96
##   2 Austra~ AUS    1961 1.96e10   2.49  8.14
##   3 Austra~ AUS    1962 1.99e10   1.30  8.12
##   4 Austra~ AUS    1963 2.15e10   6.21  8.17
##   5 Austra~ AUS    1964 2.38e10   6.98  8.40
##   6 Austra~ AUS    1965 2.59e10   5.98  8.69
```

Outline

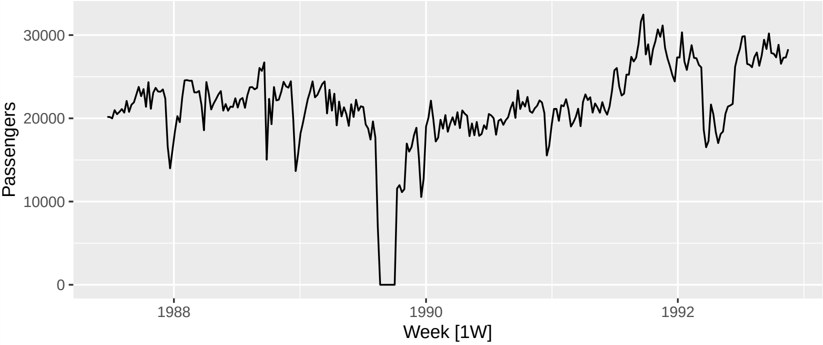
Australian GDP

```r
aus_economy %>% autoplot(GDP)
```



Time plots

```
ansett %>%
  filter(Airports=="MEL-SYD", Class=="Economy") %>%
  autoplot(Passengers)
```



Time plots

```
# Taking a subset of the time series according to time
a10.subset <- PBS %>% filter(ATC2 == "A10")%>%
  filter_index("2008 Jan")
a10.subset
```

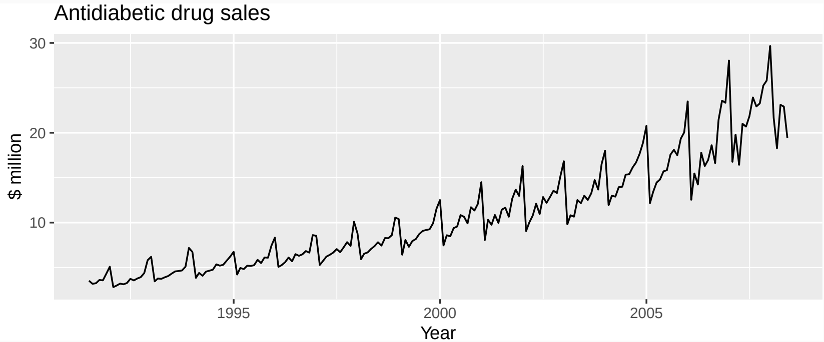
```
## # A tsibble: 4 x 9 [1M]
## # Key:       Concession, Type, ATC1, ATC2 [4]
##        Month Concession Type  ATC1  ATC1_desc
##        <mth> <chr>      <chr> <chr> <chr>
## 1   2008 Jan Concessio~ Co-p~ A     Alimenta~
## 2   2008 Jan Concessio~ Safe~ A     Alimenta~
## 3   2008 Jan General    Co-p~ A     Alimenta~
## 4   2008 Jan General    Safe~ A     Alimenta~
## # ... with 4 more variables: ATC2 <chr>,
```

Time plots

```
a10 <- PBS %>%
  filter(ATC2 == "A10") %>%
  summarise(Cost = sum(Cost)/1e6)
```

Time plots

```
a10 %>% autoplot(Cost) +
  ylab("$ million") + xlab("Year") +
  ggtitle("Antidiabetic drug sales")
```

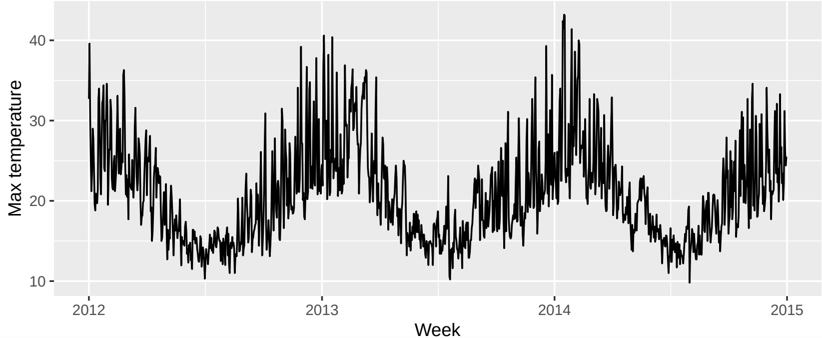


Your turn

- Create plots of the following time series: Bricks from `aus_production`, Lynx from `pelt`, Google from `gafa_stock`
- Use `help()` to find out about the data in each series.
- For the last plot, modify the axis labels and title.

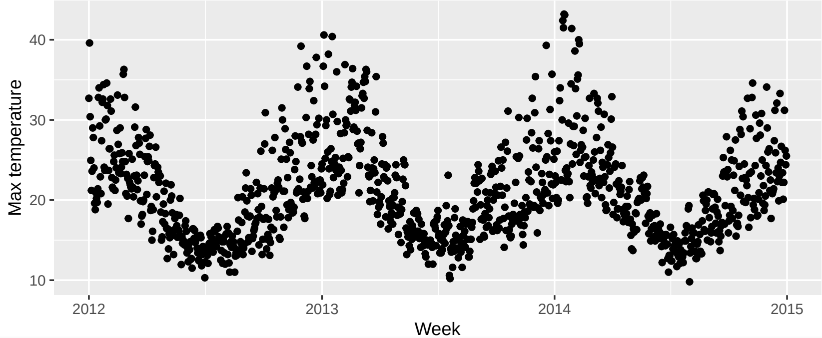
Are time plots the best?

```
maxtemp %>%
  autoplot(Temperature) +
  xlab("Week") + ylab("Max temperature")
```

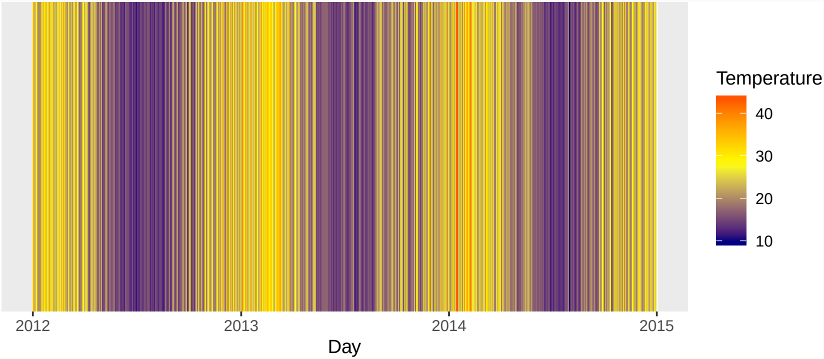


Are time plots the best?

```
maxtemp %>%
  ggplot(aes(x = Day, y = Temperature)) + geom_point()
  xlab("Week") + ylab("Max temperature")
```



Are time plots the best?

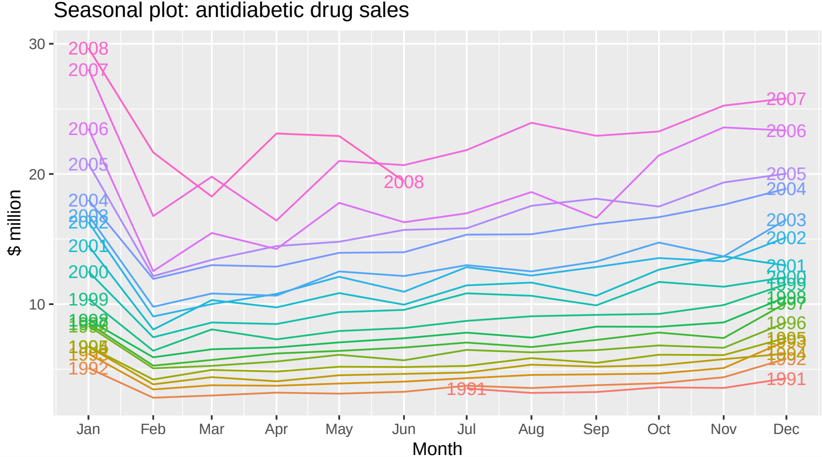


Are time plots the best?

Outline

Seasonal plots

```
a10 %>% gg_season(Cost, labels = "both") +
  ylab("$ million") + ggtitle("Seasonal plot: antidiabetic drug sa
```

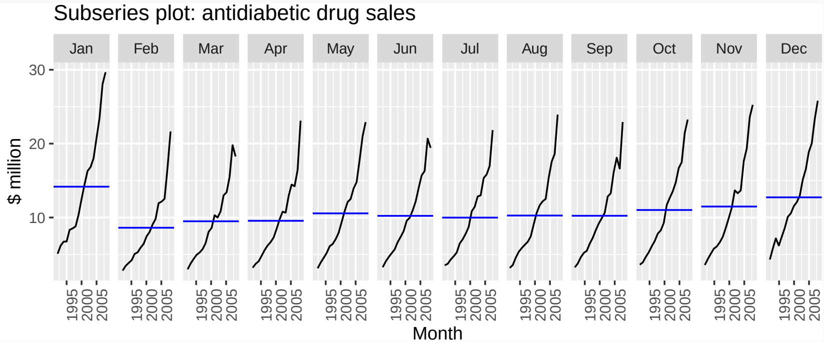


Seasonal plots

- Data plotted against the individual “seasons” in which the data were observed. (In this case a “season” is a month.)
- Something like a time plot except that the data from each season are overlapped.
- Enables the underlying seasonal pattern to be seen more clearly, and also allows any substantial departures from the seasonal pattern to be easily identified.
- In R: `gg_season()`

Seasonal subseries plots

```r
a10 %>%
  gg_subseries(Cost) + ylab("$ million") +
  ggtitle("Subseries plot: antidiabetic drug sales")
```

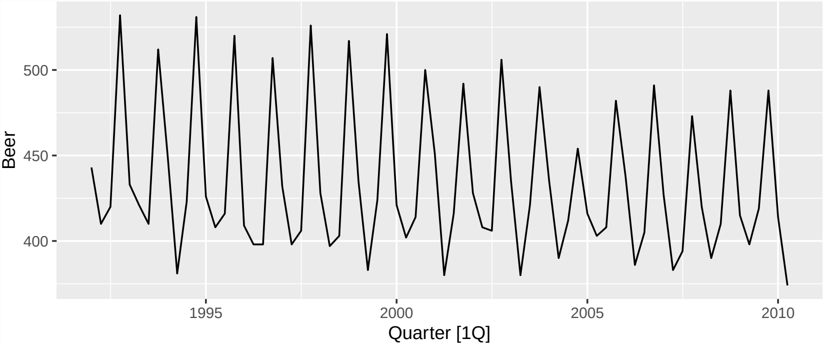


Seasonal subseries plots

- Data for each season collected together in time plot as separate time series.
- Enables the underlying seasonal pattern to be seen clearly, and changes in seasonality over time to be visualized.
- In R: `gg_subseries()`

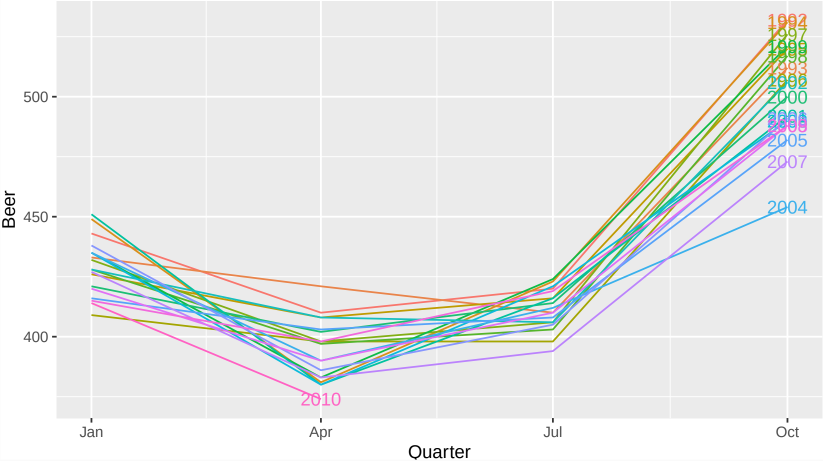
Quarterly Australian Beer Production

```
beer <- aus_production %>%
  select(Quarter, Beer) %>%
  filter(year(Quarter) >= 1992)
beer %>% autoplot(Beer)
```



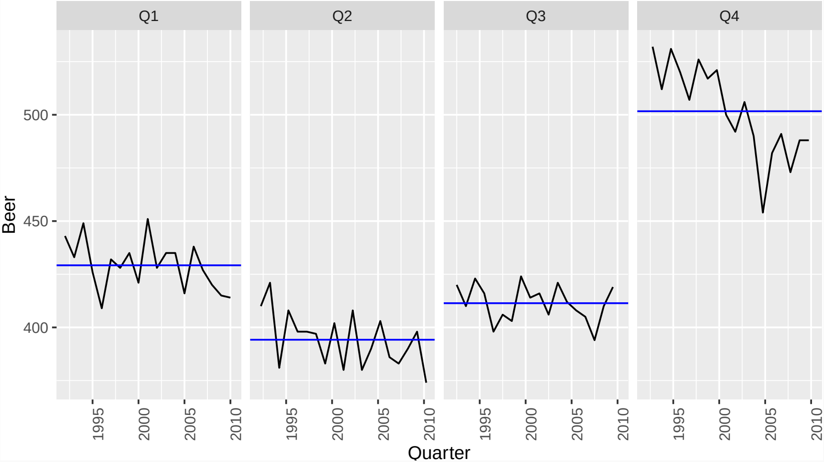
Quarterly Australian Beer Production

```r
beer %>% gg_season(Beer, labels="right")
```



Quarterly Australian Beer Production

```r
beer %>% gg_subseries(Beer)
```



Your turn

Look at the quarterly tourism data for the Snowy Mountains

```
snowy <- filter(tourism,
  Region == "Snowy Mountains",
  Purpose == "Holiday")
```

- Use `autoplot()`, `gg_season()` and `gg_subseries()` to explore the data.
- What do you learn?

Outline

Time series patterns

Trend pattern exists when there is a long-term increase or decrease in the data.

Seasonal pattern exists when a series is influenced by seasonal factors (e.g., the quarter of the year, the month, or day of the week).

Cyclic pattern exists when data exhibit rises and falls that are *not of fixed period*.

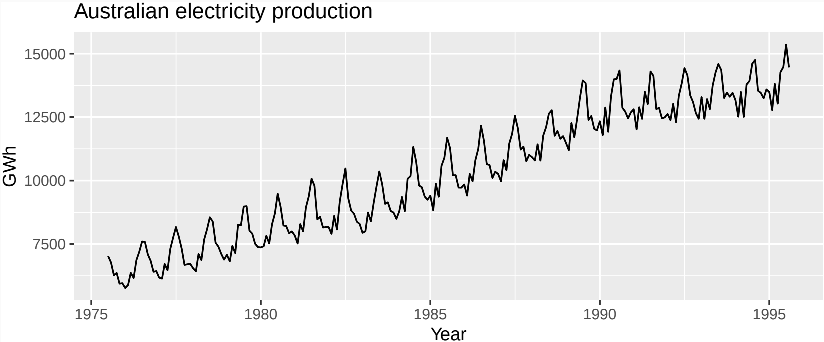
Time series components

Differences between seasonal and cyclic patterns:

- seasonal pattern constant length; cyclic pattern variable length
- average length of cycle longer than length of seasonal pattern
- magnitude of cycle more variable than magnitude of seasonal pattern

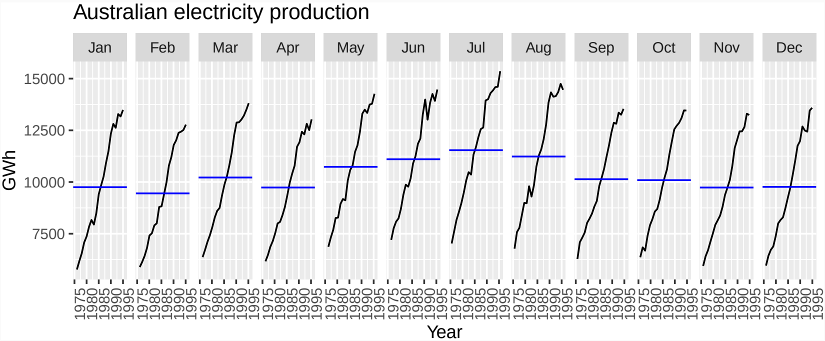
Time series patterns

```r
as_tsibble(fma::elec) %>%
  filter(index >= 1980) %>% # or filter_index("1980
  autoplot(value) + xlab("Year") + ylab("GWh") +
  ggtitle("Australian electricity production")
```



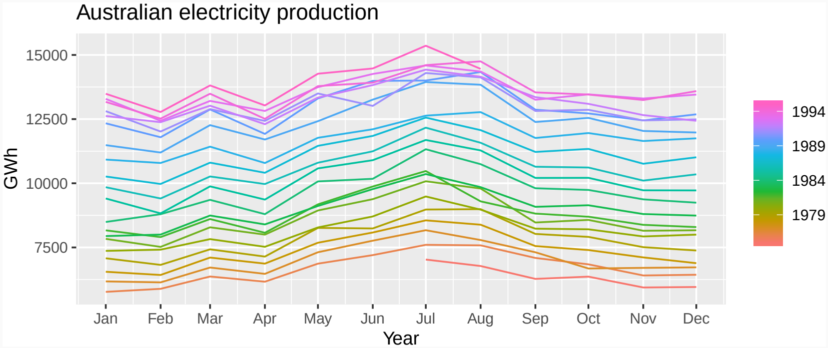
Time series patterns

```r
as_tsibble(fma::elec) %>%
  filter(index >= 1980) %>% gg_subseries(value
  xlab("Year") + ylab("GWh") +ggtitle("Austral
```



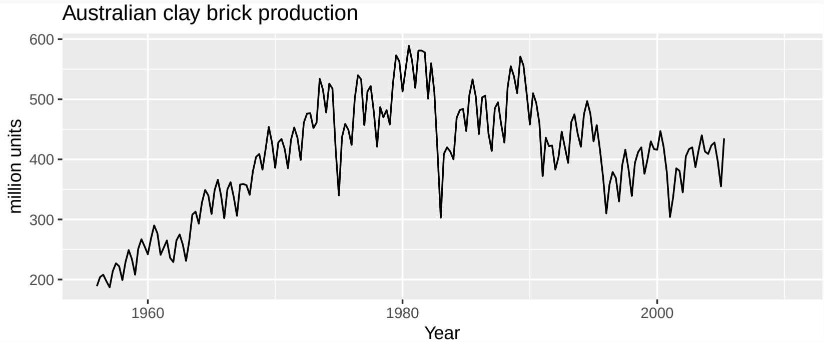
Time series patterns

```
as_tsibble(fma::elec) %>%
  filter(index >= 1980) %>% gg_season(value)+
  xlab("Year") + ylab("GWh") +ggtitle("Austral
```



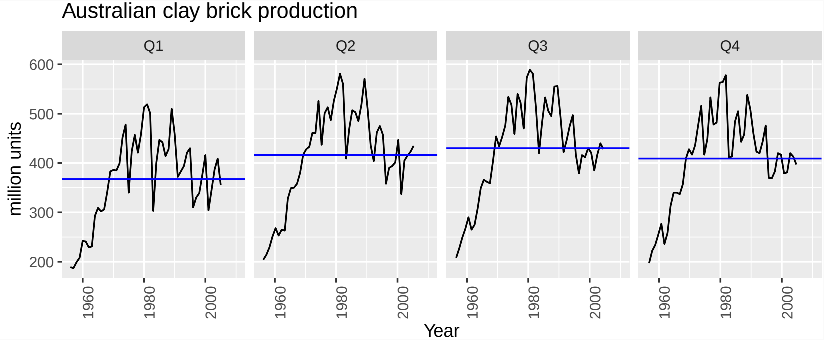
Time series patterns

```
aus_production %>%
  autoplot(Bricks) +
  ggtitle("Australian clay brick production") +
  xlab("Year") + ylab("million units")
```



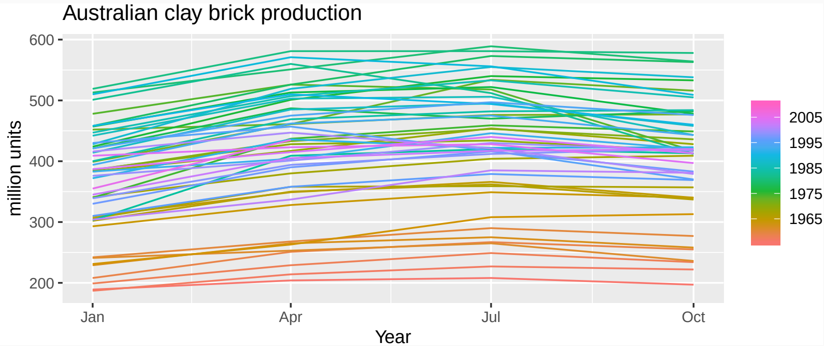
Time series patterns

```
aus_production %>%
  gg_subseries(Bricks) +
  ggtitle("Australian clay brick production") +
  xlab("Year") + ylab("million units")
```



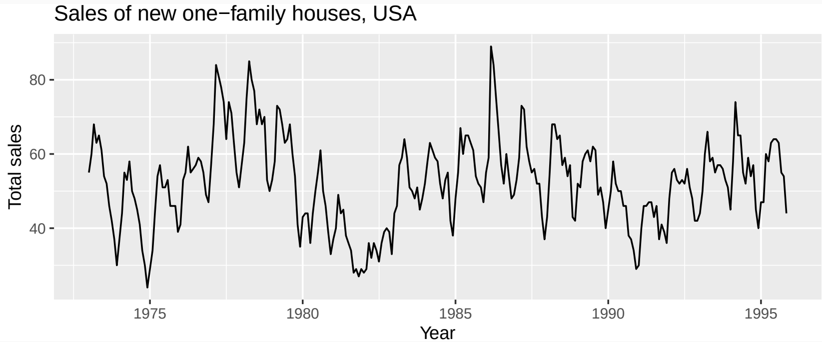
Time series patterns

```
aus_production %>%
  gg_season(Bricks) +
  ggtitle("Australian clay brick production") +
  xlab("Year") + ylab("million units")
```



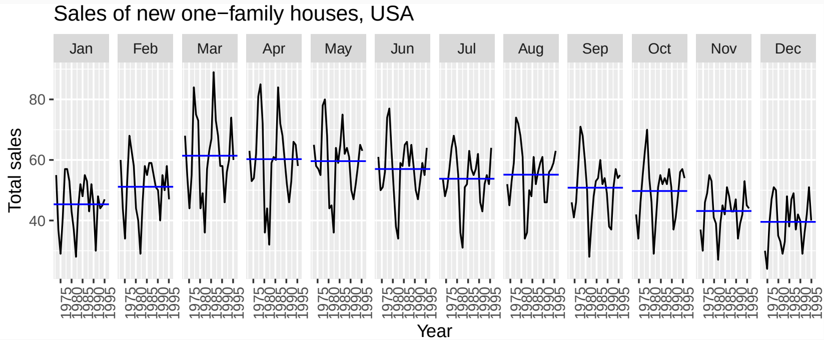
Time series patterns

```r
as_tsibble(fma::hsales) %>%
  autoplot(value) +
  ggtitle("Sales of new one-family houses, USA") +
  xlab("Year") + ylab("Total sales")
```



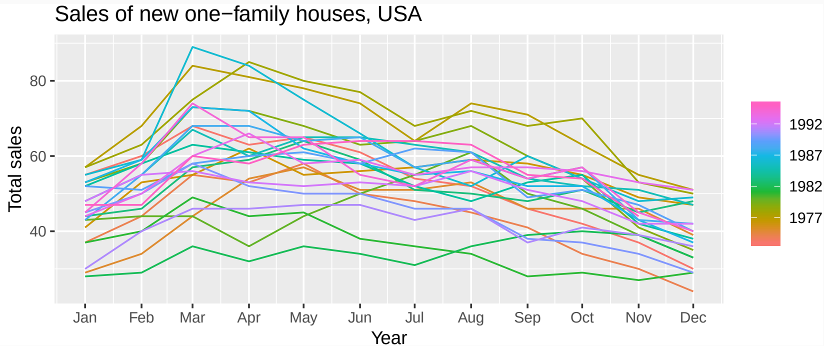
Time series patterns

```
as_tsibble(fma::hsales) %>%
  gg_subseries(value) +
  ggtitle("Sales of new one-family houses, USA") +
  xlab("Year") + ylab("Total sales")
```



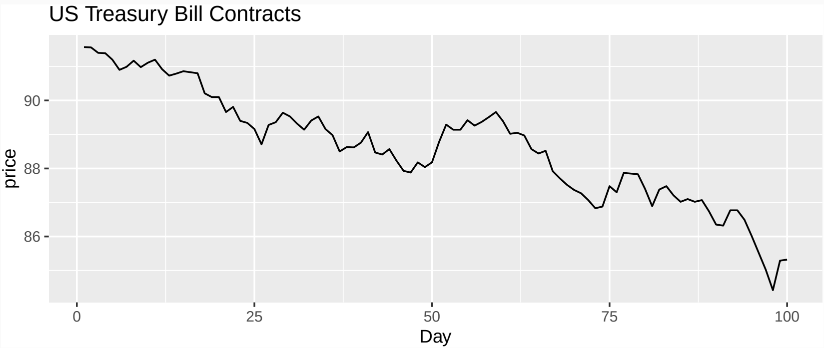
Time series patterns

```
as_tsibble(fma::hsales) %>%
  gg_season(value) +
  ggtitle("Sales of new one-family houses, USA") +
  xlab("Year") + ylab("Total sales")
```



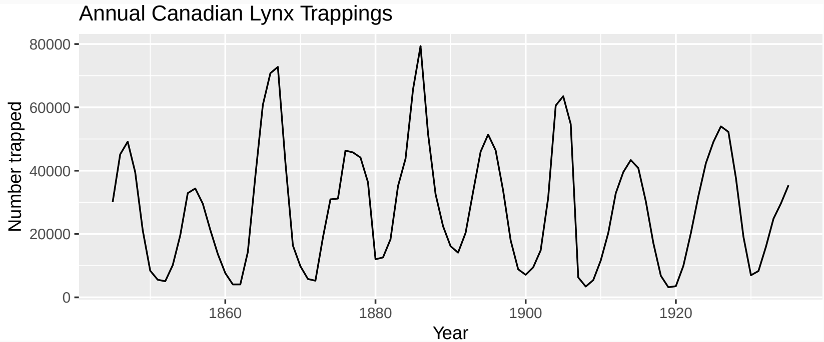
Time series patterns

```
as_tsibble(fma::ustreas) %>%
  autoplot(value) +
  ggtitle("US Treasury Bill Contracts") +
  xlab("Day") + ylab("price")
```



Time series patterns

```
pelt %>%
  autoplot(Lynx) +
  ggtitle("Annual Canadian Lynx Trappings") +
  xlab("Year") + ylab("Number trapped")
```



Seasonal or cyclic?

Differences between seasonal and cyclic patterns:

- seasonal pattern constant length; cyclic pattern variable length
- average length of cycle longer than length of seasonal pattern
- magnitude of cycle more variable than magnitude of seasonal pattern

Seasonal or cyclic?

Differences between seasonal and cyclic patterns:

- seasonal pattern constant length; cyclic pattern variable length
- average length of cycle longer than length of seasonal pattern
- magnitude of cycle more variable than magnitude of seasonal pattern

The timing of peaks and troughs is predictable with seasonal data, but unpredictable in the long term with cyclic data.

Outline

Example: Beer production

```
new_production <- aus_production %>%
  filter(year(Quarter) >= 1992) # or filter_index("
new_production
```

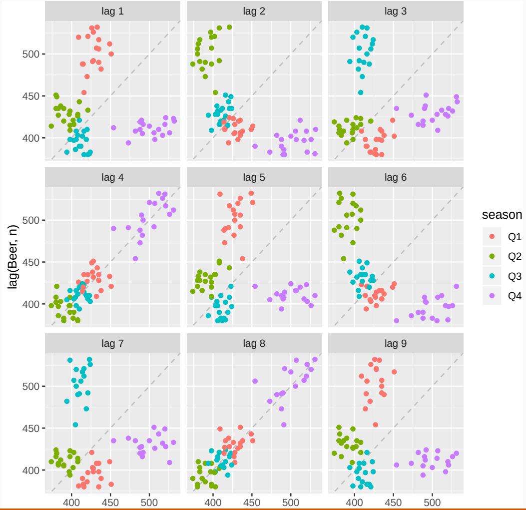
```
## # A tsibble: 74 x 7 [1Q]
##         Quarter  Beer Tobacco Bricks Cement
##           <qtr> <dbl>   <dbl>  <dbl>  <dbl>
##  1      1992 Q1   443    5777    383   1289
##  2      1992 Q2   410    5853    404   1501
##  3      1992 Q3   420    6416    446   1539
##  4      1992 Q4   532    5825    420   1568
##  5      1993 Q1   433    5724    394   1450
##  6      1993 Q2   421    6036    462   1668
```

Lagged scatterplots

- Each graph shows y_t plotted against y_{t-k} for different values of k.
- Vertical axis: lagged observation
- Horizontal axis: current observation
- Colors: points are colored by the current quarter

Example: Beer production

```r
new_production %>% gg_lag(Beer, geom='point')
```



Lagged scatterplots

- The autocorrelations are the correlations associated with these scatterplots.

Autocorrelation

Covariance and **correlation**: measure extent of **linear relationship** between two variables (y and X).

Autocorrelation

Covariance and **correlation**: measure extent of **linear relationship** between two variables (y and X).

Autocovariance and **autocorrelation**: measure linear relationship between **lagged values** of a time series y.

Autocorrelation

Covariance and **correlation**: measure extent of **linear relationship** between two variables (y and X).

Autocovariance and **autocorrelation**: measure linear relationship between **lagged values** of a time series y.

We measure the relationship between:

- y_t and y_{t-1}
- y_t and y_{t-2}
- y_t and y_{t-3}
- etc.

Autocorrelation

We denote the sample autocovariance at lag k by c_k and the sample autocorrelation at lag k by r_k. Then define

$$c_k = \frac{1}{T} \sum_{t=k+1}^{T} (y_t - \bar{y})(y_{t-k} - \bar{y})$$

$$\text{and} \qquad r_k = c_k / c_0$$

Autocorrelation

We denote the sample autocovariance at lag k by c_k and the sample autocorrelation at lag k by r_k. Then define

$$c_k = \frac{1}{T} \sum_{t=k+1}^{T} (y_t - \bar{y})(y_{t-k} - \bar{y})$$

and $$r_k = c_k / c_0$$

- r_1 indicates how successive values of y relate to each other
- r_2 indicates how y values two periods apart relate to each other
- r_k is *almost* the same as the sample correlation between y_t and y_{t-k}.

Autocorrelation

Results for first 9 lags for beer data:

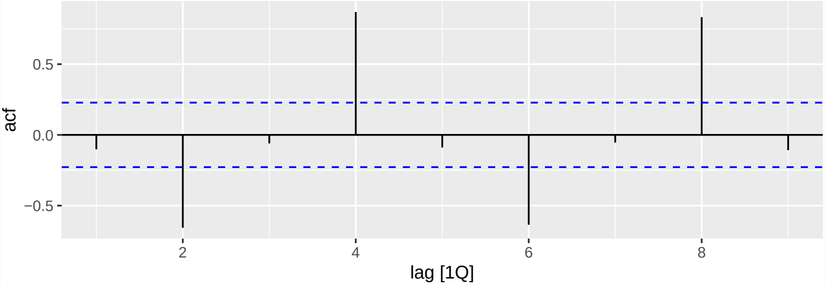
```
new_production %>% ACF(Beer, lag_max = 9)
```

```
## # A tsibble: 9 x 2 [1Q]
##      lag     acf
##    <lag>   <dbl>
## 1     1Q -0.102
## 2     2Q -0.657
## 3     3Q -0.0603
## 4     4Q  0.869
## 5     5Q -0.0892
## 6     6Q -0.635
## 7     7Q -0.0542
## 8     8Q  0.832
```

Autocorrelation

Results for first 9 lags for beer data:

```
new_production %>% ACF(Beer, lag_max = 9) %>% autoplot()
```

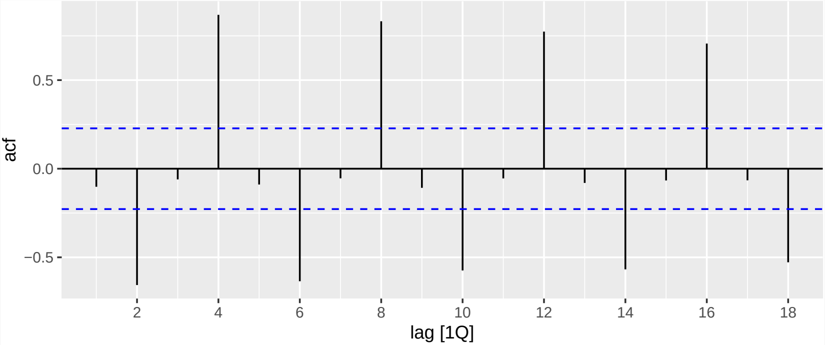


Autocorrelation

- r_4 higher than for the other lags. This is due to **the seasonal pattern in the data**: the peaks tend to be **4 quarters** apart and the troughs tend to be **2 quarters** apart.
- r_2 is more negative than for the other lags because troughs tend to be 2 quarters behind peaks.
- Together, the autocorrelations at lags 1, 2, . . . , make up the *autocorrelation* or ACF.
- The plot is known as a **correlogram**

ACF

```
new_production %>% ACF(Beer) %>% autoplot()
```

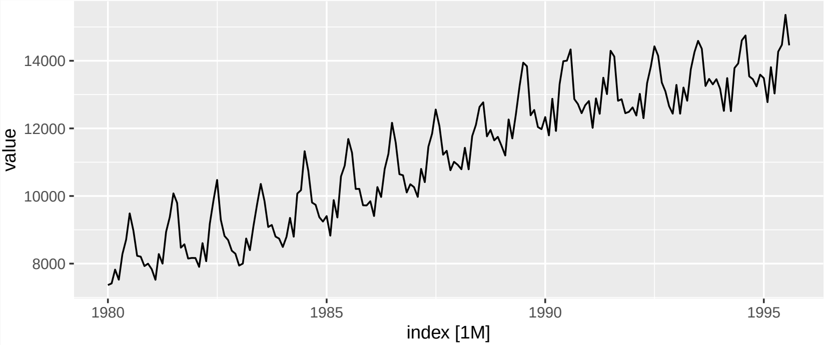


Trend and seasonality in ACF plots

- When data have a trend, the autocorrelations for small lags tend to be large and positive.
- When data are seasonal, the autocorrelations will be larger at the seasonal lags (i.e., at multiples of the seasonal frequency)
- When data are trended and seasonal, you see a combination of these effects.

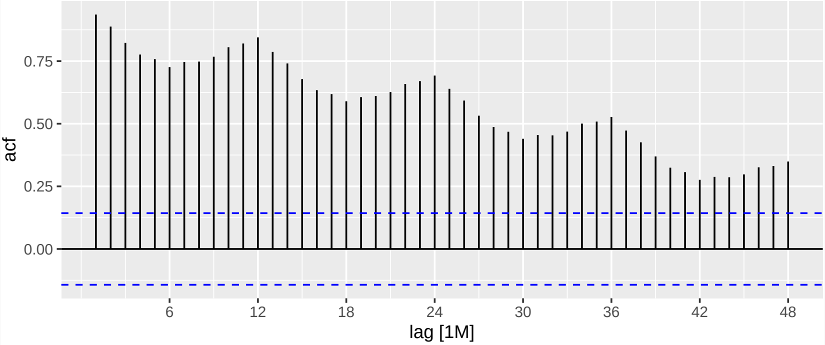
Aus monthly electricity production

```r
elec2 <- as_tsibble(fma::elec) %>%
  filter(year(index) >= 1980)
elec2 %>% autoplot(value)
```



Aus monthly electricity production

```
elec2 %>% ACF(value, lag_max=48) %>%
  autoplot()
```



Aus monthly electricity production

Time plot shows clear trend and seasonality.

The same features are reflected in the ACF.

- The slowly decaying ACF indicates trend.
- The ACF peaks at lags 12, 24, 36, . . . , indicate seasonality of length 12.

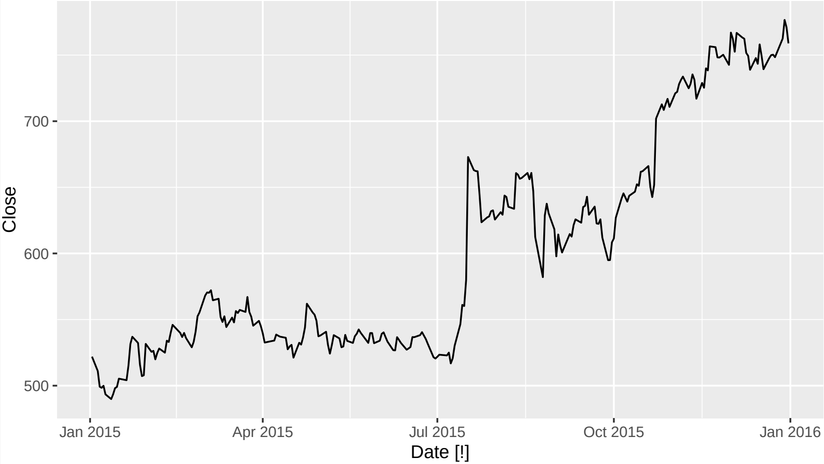
Google stock price

```
google_2015 <- gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2015) %>%
  select(Date, Close)
google_2015
```

```
## # A tsibble: 252 x 2 [!]
##    Date       Close
##    <date>     <dbl>
##  1 2015-01-02  522.
##  2 2015-01-05  511.
##  3 2015-01-06  499.
##  4 2015-01-07  498.
##  5 2015-01-08  500.
##  6 2015-01-09  493.
```

Google stock price

```
google_2015 %>% autoplot(Close)
```



Google stock price

```
google_2015 %>%
  ACF(Close, lag_max=100)
# Error: Can't handle tsibble of irregular interval.
```

Google stock price

```
google_2015 %>%
  ACF(Close, lag_max=100)
# Error: Can't handle tsibble of irregular interval.
```

```
google_2015
```

```
## # A tsibble: 252 x 2 [!]
##    Date        Close
##    <date>      <dbl>
##  1 2015-01-02  522.
##  2 2015-01-05  511.
##  3 2015-01-06  499.
```

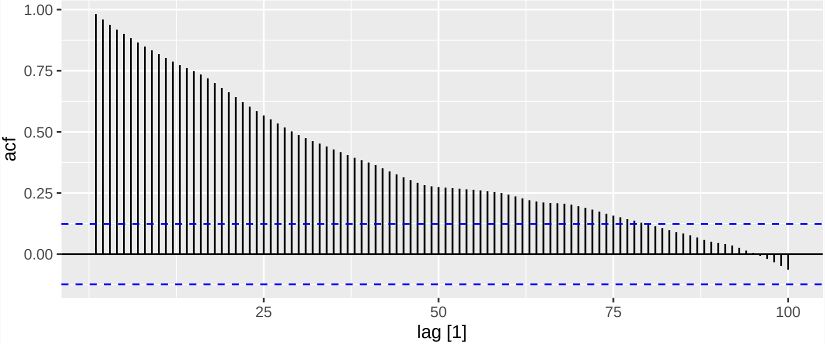
Google stock price

```
#mutate is to create a new variable
google_2015 <- google_2015 %>%
  mutate(trading_day = row_number()) %>%
  update_tsibble(index=trading_day, regular=TRUE)
google_2015
```

```
## # A tsibble: 252 x 3 [1]
##    Date       Close trading_day
##    <date>     <dbl>       <int>
##  1 2015-01-02  522.           1
##  2 2015-01-05  511.           2
##  3 2015-01-06  499.           3
##  4 2015-01-07  498.           4
##  5 2015-01-08  500.           5
```

Google stock price

```
google_2015 %>%ACF(Close, lag_max=100) %>% aut
```



Your turn

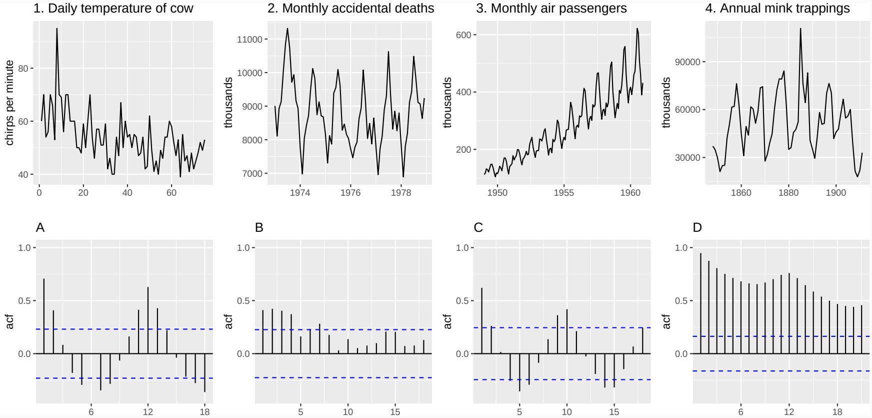
We have introduced the following functions:

- `gg_lag`
- `ACF`

Explore the following time series using these functions. Can you spot any seasonality, cyclicity and trend? What do you learn about the series?

- Bricks from `aus_production`
- Lynx from `pelt`
- Victorian Electricity Demand from `aus_elec`

Which is which?

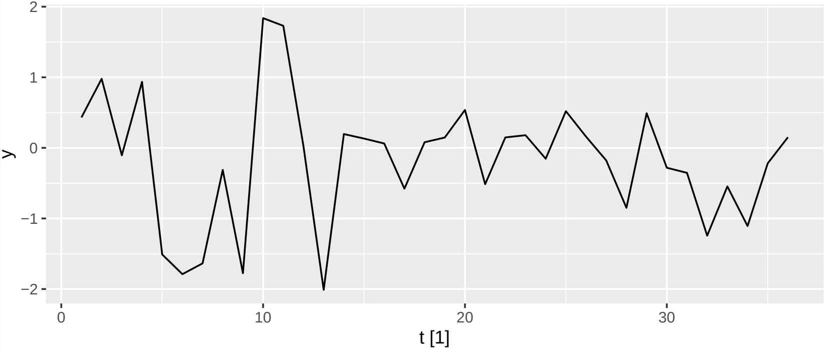


Outline

1. Time series in R
2. Time plots
3. Seasonal plots
4. Seasonal or cyclic?
5. Lag plots and autocorrelation
6. White noise

Example: White noise

```
wn <- tsibble(t = seq_len(36), y = rnorm(36),
              index = t)
wn %>% autoplot(y)
```

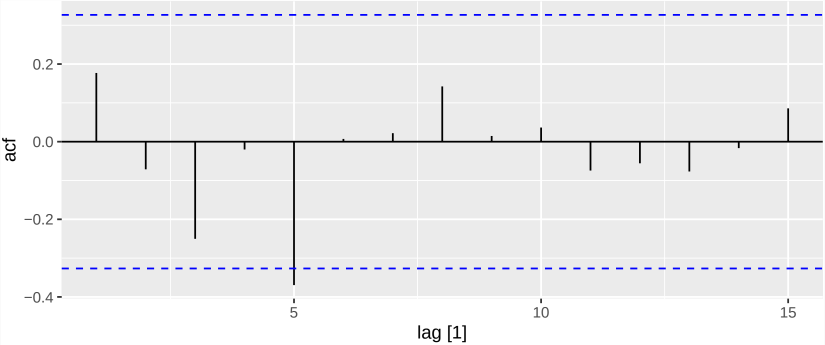


Example: White noise

```
wn %>% ACF(y, lag_max = 10) %>%
  as_tibble() %>%
  tidyr::spread(lag, acf) %>%
  rename_all(function(x){paste("$r_{",x,"}$",sep="")}) %>%
  knitr::kable(booktabs=TRUE,
               escape=FALSE, align="c", digits=3,
               format.args=list(nsmall=3))
```

r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}
0.177	-0.071	-0.250	-0.020	-0.370	0.007	0.022	0.142	0.015	0.036

Example: White noise



Sampling distribution of autocorrelations

Sampling distribution of r_k for white noise data is asymptotically N$(0{,}1/T)$.

Sampling distribution of autocorrelations

Sampling distribution of r_k for white noise data is asymptotically N(0,1/T).

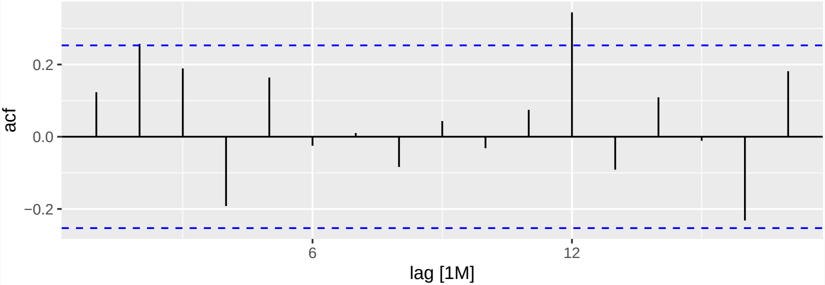
- 95% of all r_k for white noise must lie within $\pm 1.96/\sqrt{T}$.
- If this is not the case, the series is probably not WN.
- Common to plot lines at $\pm 1.96/\sqrt{T}$ when plotting ACF. These are the critical values.

Example: Pigs slaughtered

```
pigs <- aus_livestock %>%
  filter(State == "Victoria", Animal == "Pigs",
         year(Month) >= 2014)
pigs %>% autoplot(Count/1e3) +
  xlab("Year") + ylab("Thousands") +
  ggtitle("Number of pigs slaughtered in Victoria")
```

Example: Pigs slaughtered

```
pigs %>% ACF(Count) %>% autoplot()
```



Example: Pigs slaughtered

Monthly total number of pigs slaughtered in the state of Victoria, Australia, from January 2014 through December 2018 (Source: Australian Bureau of Statistics.)

Example: Pigs slaughtered

Monthly total number of pigs slaughtered in the state of Victoria, Australia, from January 2014 through December 2018 (Source: Australian Bureau of Statistics.)

- Difficult to detect pattern in time plot.
- ACF shows significant autocorrelation for lag 2 and 12.
- Indicate some slight seasonality.

Example: Pigs slaughtered

Monthly total number of pigs slaughtered in the state of Victoria, Australia, from January 2014 through December 2018 (Source: Australian Bureau of Statistics.)

- Difficult to detect pattern in time plot.
- ACF shows significant autocorrelation for lag 2 and 12.
- Indicate some slight seasonality.

These show the series is **not a white noise series**.

Your turn

You can compute the daily changes in the Google stock price in 2018 using

```
dgoog <- gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) >= 2018) %>%
  mutate(trading_day = row_number()) %>%
  update_tsibble(index=trading_day, regular=TRUE) %>%
  mutate(diff = difference(Close))
```

Does `diff` look like white noise?