Tidy Time Series & Forecasting in R

2. Time series graphics

bit.ly/fable202

Outline

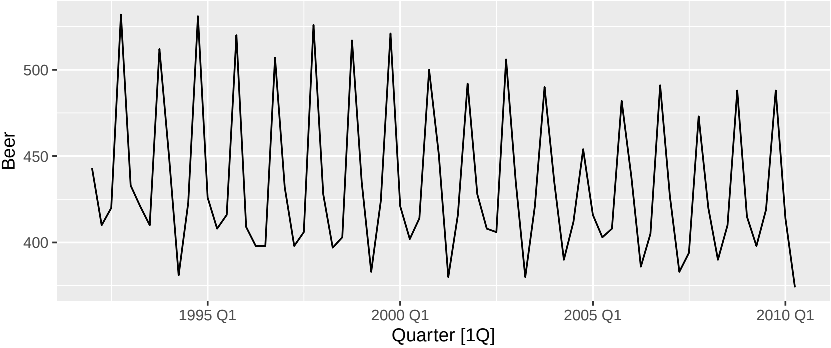
Outline

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()`

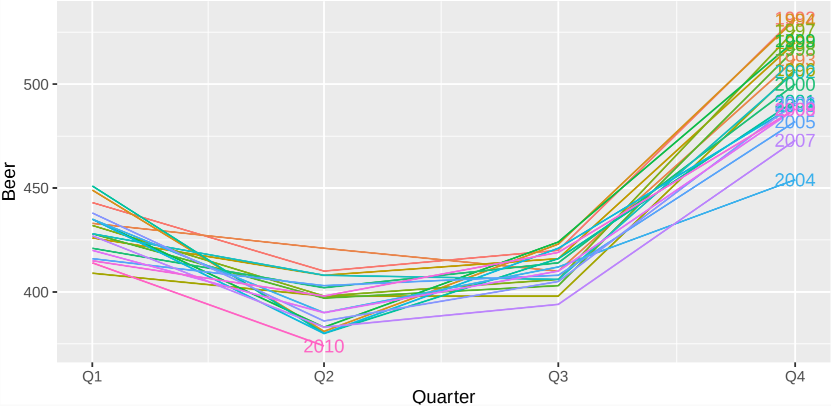
Quarterly Australian Beer Production

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



Quarterly Australian Beer Production

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



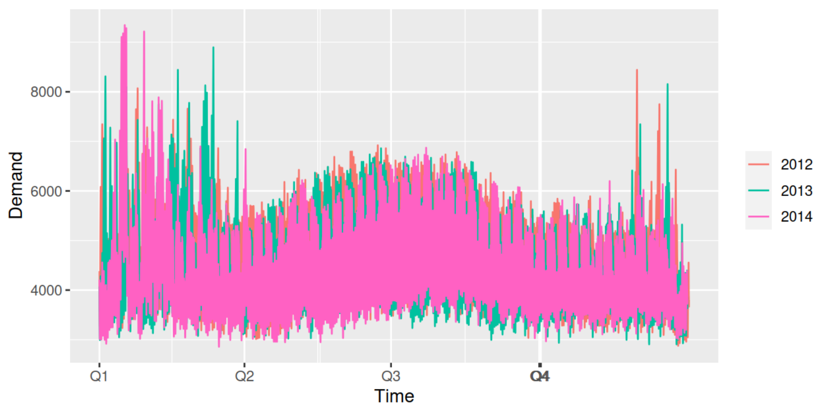
Multiple seasonal periods

```
vic_elec
```

```
## # A tsibble: 52,608 x 5 [30m] <Australia/Melbourne>
##    Time                Demand Temperature Date       Holiday
##    <dttm>               <dbl>       <dbl> <date>     <lgl>
##  1 2012-01-01 00:00:00  4383.        21.4 2012-01-01 TRUE
##  2 2012-01-01 00:30:00  4263.        21.0 2012-01-01 TRUE
##  3 2012-01-01 01:00:00  4049.        20.7 2012-01-01 TRUE
##  4 2012-01-01 01:30:00  3878.        20.6 2012-01-01 TRUE
##  5 2012-01-01 02:00:00  4036.        20.4 2012-01-01 TRUE
##  6 2012-01-01 02:30:00  3866.        20.2 2012-01-01 TRUE
##  7 2012-01-01 03:00:00  3694.        20.1 2012-01-01 TRUE
##  8 2012-01-01 03:30:00  3562.        19.6 2012-01-01 TRUE
##  9 2012-01-01 04:00:00  3433.        19.1 2012-01-01 TRUE
## 10 2012-01-01 04:30:00  3359.        19.0 2012-01-01 TRUE
## # ... with 52,598 more rows
```

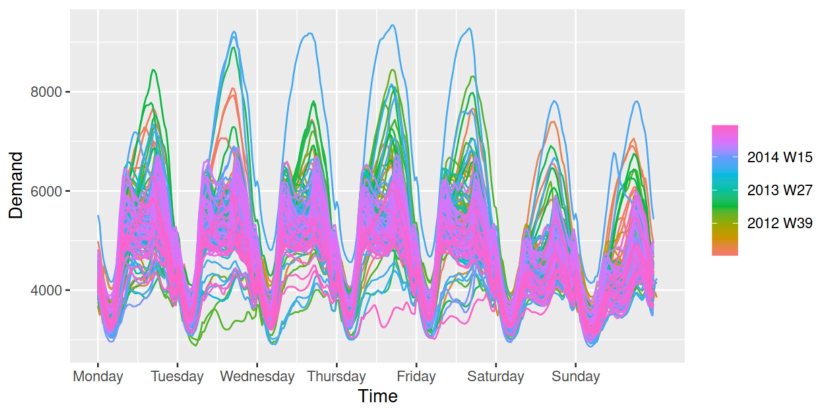
Multiple seasonal periods

```r
vic_elec %>% gg_season(Demand)
```



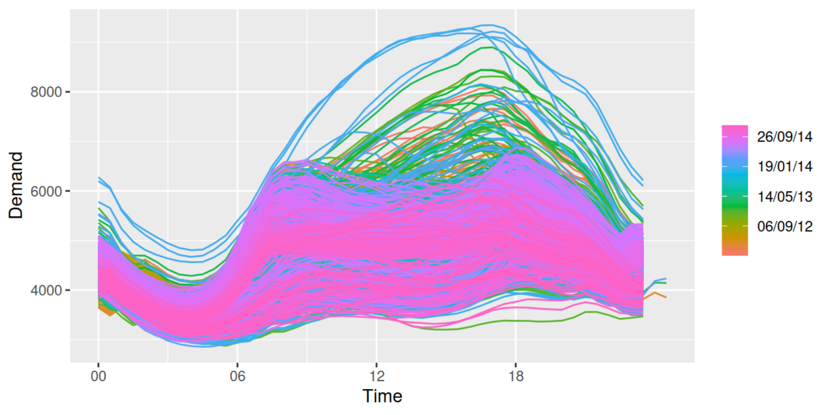
Multiple seasonal periods

```r
vic_elec %>% gg_season(Demand, period = "week")
```



Multiple seasonal periods

```
vic_elec %>% gg_season(Demand, period = "day")
```

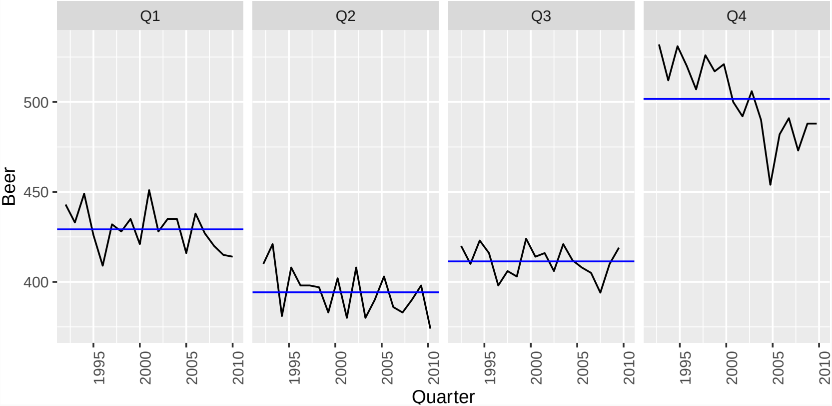


Seasonal subseries plots

- 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

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



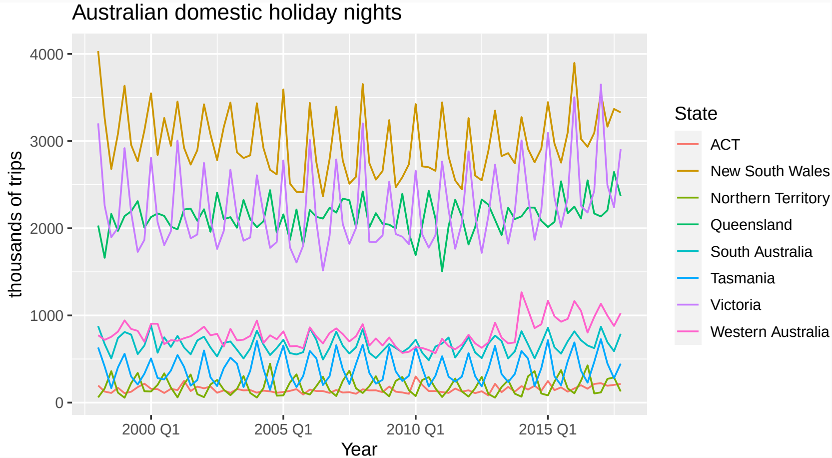
Australian holidays

```r
holidays <- tourism %>%
  filter(Purpose == "Holiday") %>%
  group_by(State) %>%
  summarise(Trips = sum(Trips))
```

```
## # A tsibble: 640 x 3 [1Q]
## # Key:       State [8]
##    State Quarter Trips
##    <chr>   <qtr> <dbl>
##  1 ACT   1998 Q1  196.
##  2 ACT   1998 Q2  127.
##  3 ACT   1998 Q3  111.
##  4 ACT   1998 Q4  170.
##  5 ACT   1999 Q1  108.
##  6 ACT   1999 Q2  125.
##  7 ACT   1999 Q3  178.
##  8 ACT   1999 Q4  218.
##  9 ACT   2000 Q1  158.
## 10 ACT   2000 Q2  155.
## # ... with 630 more rows
```

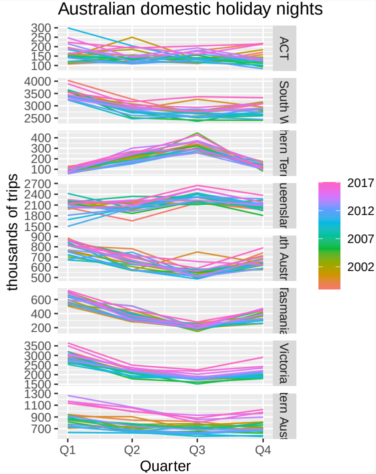
Australian holidays

```r
holidays %>% autoplot(Trips) +
  ylab("thousands of trips") + xlab("Year") +
  ggtitle("Australian domestic holiday nights")
```



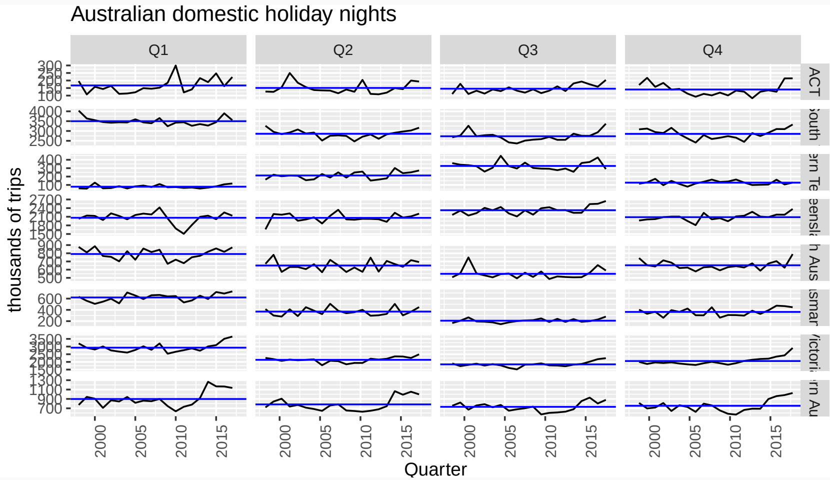
Seasonal plots

```
holidays %>% gg_season(Trips) +
  ylab("thousands of trips") +
  ggtitle("Australian domestic holiday nights")
```



Seasonal subseries plots

```
holidays %>%
  gg_subseries(Trips) + ylab("thousands of trips") +
  ggtitle("Australian domestic holiday nights")
```

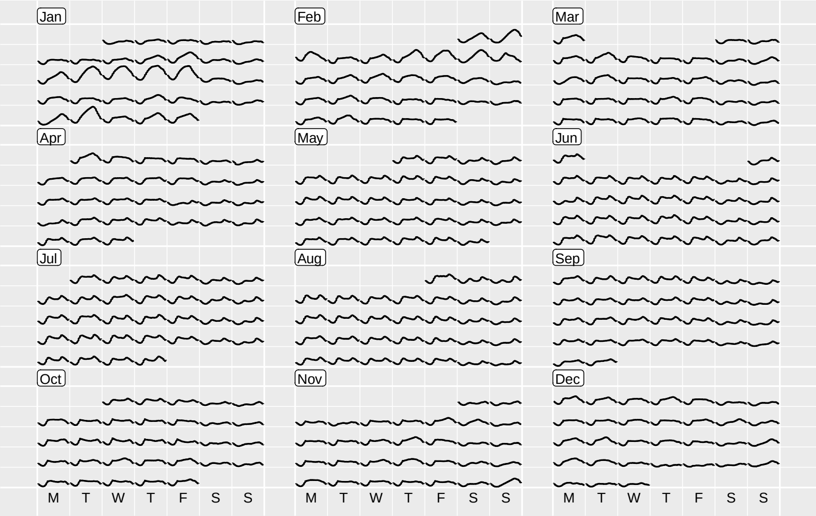


Calendar plots

```
library(sugrrants)
vic_elec %>%
  filter(year(Date) == 2014) %>%
  mutate(Hour = hour(Time)) %>%
  frame_calendar(
    x = Hour, y = Demand, date = Date,
    nrow = 4
  ) %>%
  ggplot(aes(x = .Hour, y = .Demand, group = Date)) +
  geom_line() -> p1
prettify(p1,
  size = 3,
  label.padding = unit(0.15, "lines")
)
```

`frame_calendar()` makes a compact calendar plot, `facet_calendar()` provides an easier ggplot2 integration.

Calendar plots



Outline

Lab Session 3

1. Look at the quarterly tourism data for the Snowy Mountains

```
snowy <- tourism %>%
  filter(Region == "Snowy Mountains")
```

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

2. Produce a calendar plot for the `pedestrian` data from one location and one year.

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* (duration usually of at least 2 years).

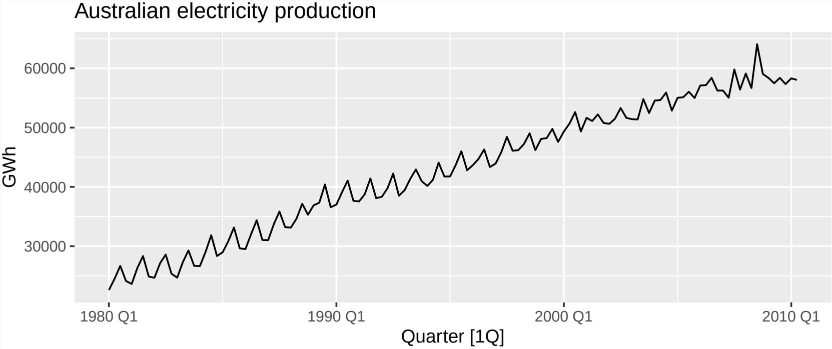
Time series components

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

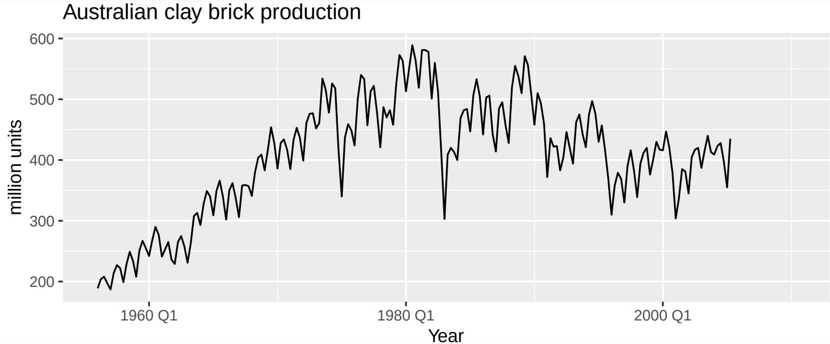
Time series patterns

```
aus_production %>%
  filter(year(Quarter) >= 1980) %>%
  autoplot(Electricity) + ylab("GWh") +
  ggtitle("Australian electricity production")
```



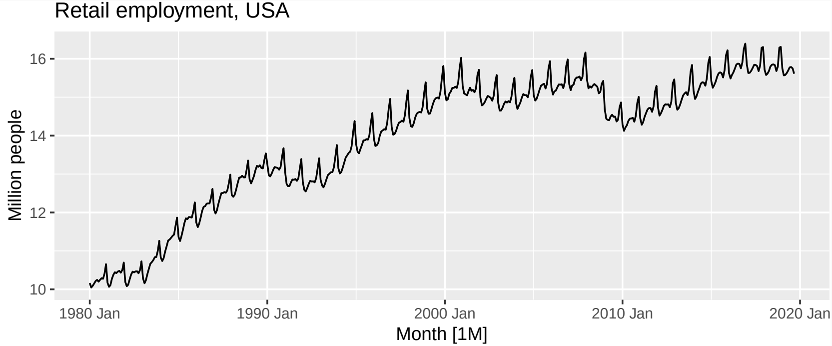
Time series patterns

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



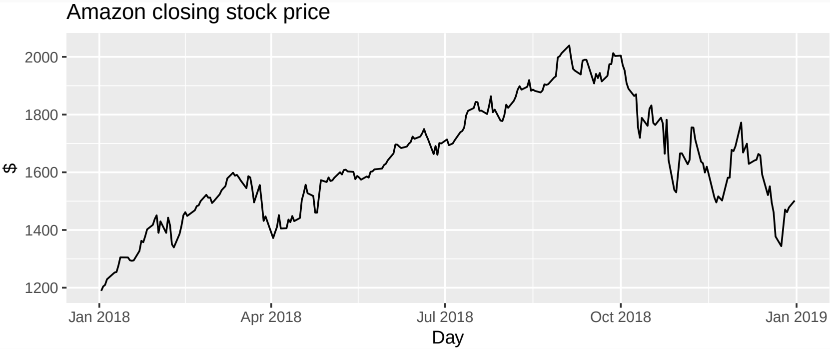
Time series patterns

```r
us_employment %>%
  filter(Title == "Retail Trade", year(Month) >= 1980) %>%
  autoplot(Employed / 1e3) +
  ggtitle("Retail employment, USA") + ylab("Million people")
```



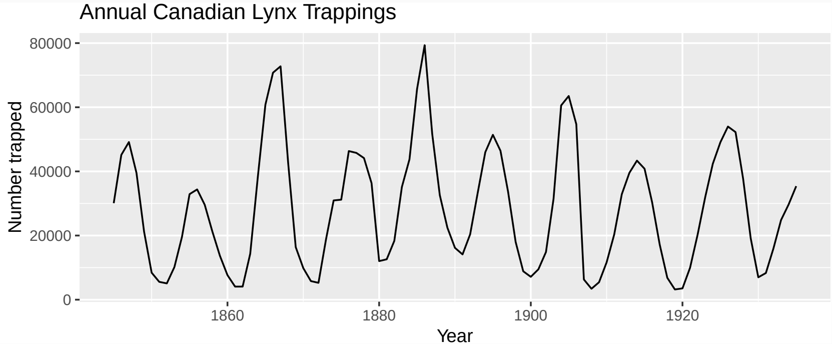
Time series patterns

```
gafa_stock %>%
  filter(Symbol == "AMZN", year(Date) >= 2018) %>%
  autoplot(Close) +
  ggtitle("Amazon closing stock price") +
  xlab("Day") + ylab("$")
```



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

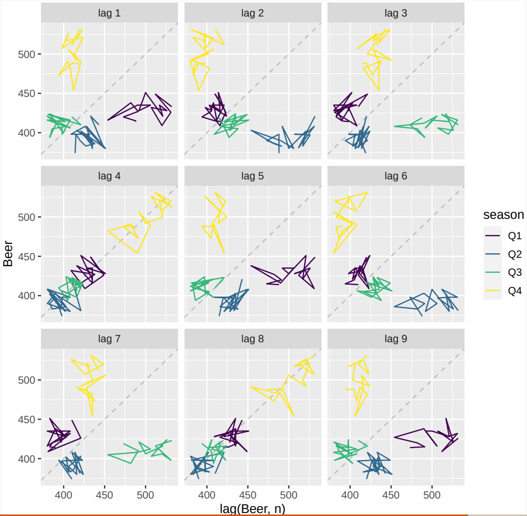
Example: Beer production

```r
new_production <- aus_production %>%
  filter(year(Quarter) >= 1992)
new_production
```

```
## # A tsibble: 74 x 7 [1Q]
##    Quarter  Beer Tobacco Bricks Cement Electricity   Gas
##      <qtr> <dbl>   <dbl>  <dbl>  <dbl>       <dbl> <dbl>
##  1 1992 Q1   443    5777    383   1289       38332   117
##  2 1992 Q2   410    5853    404   1501       39774   151
##  3 1992 Q3   420    6416    446   1539       42246   175
##  4 1992 Q4   532    5825    420   1568       38498   129
##  5 1993 Q1   433    5724    394   1450       39460   116
##  6 1993 Q2   421    6036    462   1668       41356   149
##  7 1993 Q3   410    6570    475   1648       42949   163
##  8 1993 Q4   512    5675    443   1863       40974   138
##  9 1994 Q1   449    5311    421   1468       40162   127
## 10 1994 Q2   381    5717    475   1755       41199   159
## # ... with 64 more rows
```

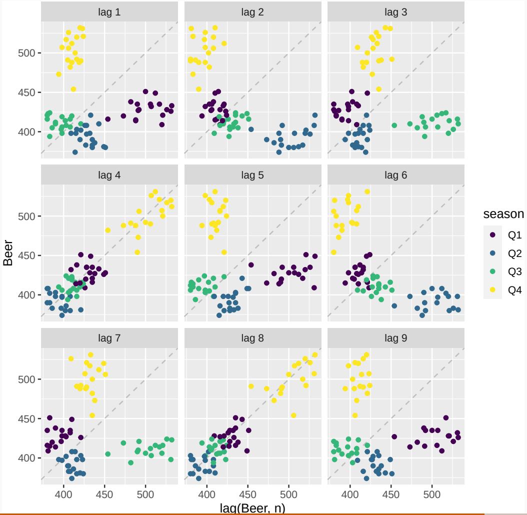
Example: Beer production

```
new_production %>% gg_lag(Beer)
```



Example: Beer production

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



Lagged scatterplots

- Each graph shows y_t plotted against y_{t-k} for different values of k.
- The autocorrelations are the correlations associated with these scatterplots.
- ACF (autocorrelation function):
  - $r_1 = \text{Correlation}(y_t, y_{t-1})$
  - $r_2 = \text{Correlation}(y_t, y_{t-2})$
  - $r_3 = \text{Correlation}(y_t, y_{t-3})$
  - etc.
- If there is **seasonality**, the ACF at the seasonal lag (e.g., 12 for monthly data) will be **large and positive**.

Autocorrelation

Results for first 9 lags for beer data:

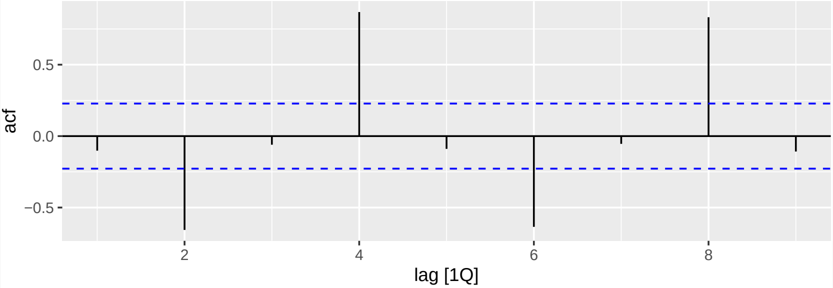
```
new_production %>% ACF(Beer, lag_max = 9)
```

```
## # 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
## 9     9Q -0.108
```

Autocorrelation

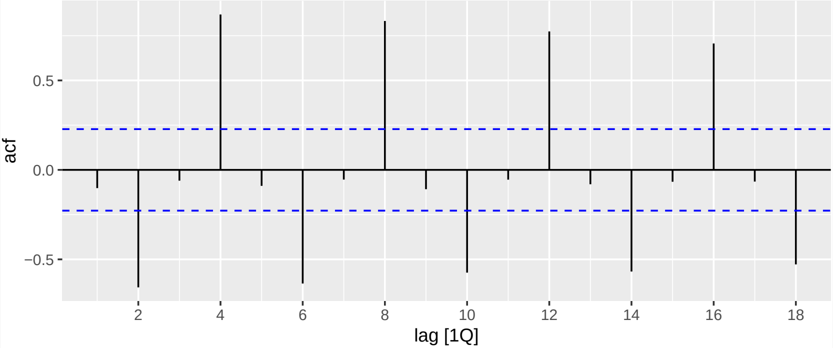
Results for first 9 lags for beer data:

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



ACF

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



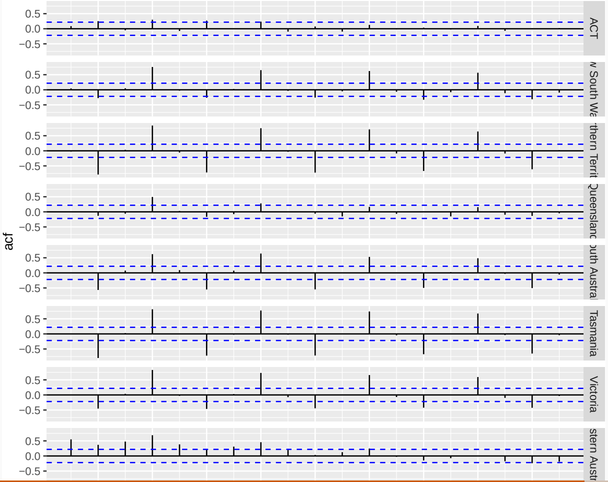
Australian holidays

```
holidays %>% ACF(Trips)
```

```
## # A tsibble: 152 x 3 [1Q]
## # Key:       State [8]
##    State   lag      acf
##    <chr> <lag>    <dbl>
##  1 ACT      1Q  0.0877
##  2 ACT      2Q  0.252
##  3 ACT      3Q -0.0496
##  4 ACT      4Q  0.300
##  5 ACT      5Q -0.0741
##  6 ACT      6Q  0.269
##  7 ACT      7Q -0.00504
##  8 ACT      8Q  0.236
##  9 ACT      9Q -0.0953
## 10 ACT     10Q  0.0750
## # ... with 142 more rows
```

Australian holidays

```r
holidays %>%
  ACF(Trips) %>%
  autoplot()
```

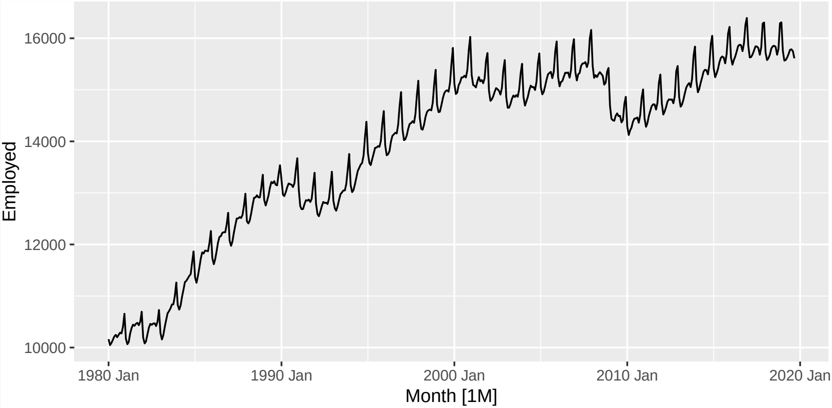


Trend and seasonality in ACF plots

- 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.

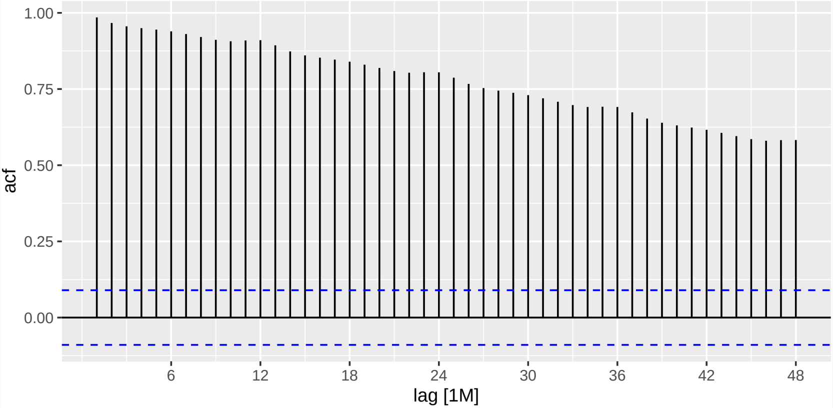
US retail trade employment

```
retail <- us_employment %>%
  filter(Title == "Retail Trade", year(Month) >= 1980)
retail %>% autoplot(Employed)
```



US retail trade employment

```r
retail %>%
  ACF(Employed, lag_max = 48) %>%
  autoplot()
```



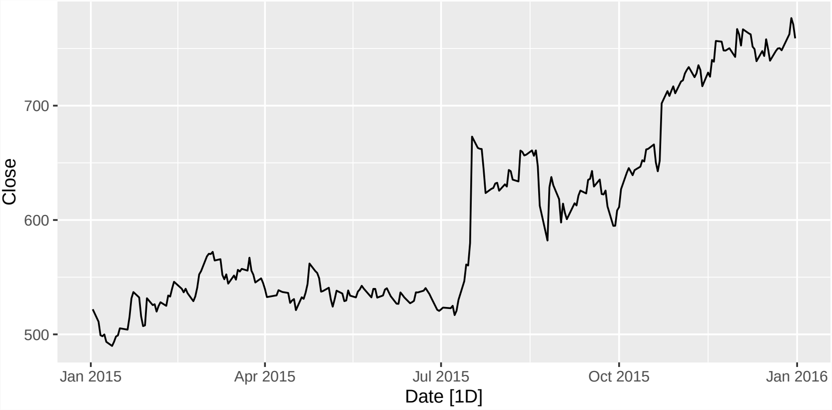
Google stock price

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

```
## # A tsibble: 252 x 2 [1D]
##    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.
##  7 2015-01-12  490.
```

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 [1D]
##    Date       Close
##    <date>     <dbl>
##  1 2015-01-02  522.
##  2 2015-01-05  511.
##  3 2015-01-06  499.
##  4 2015-01-07  498.
```

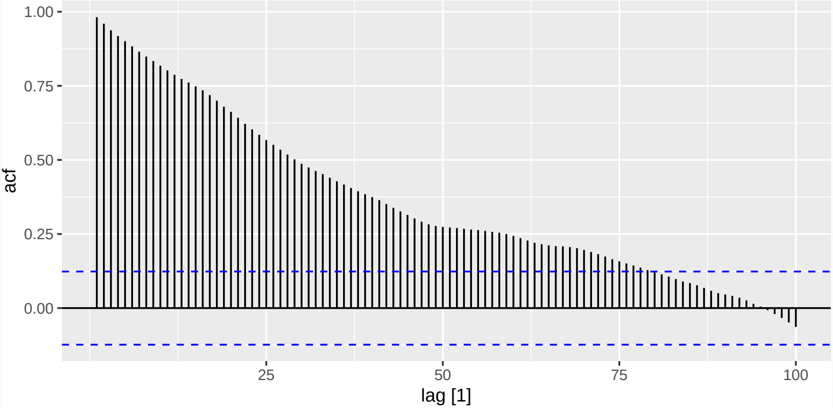
Google stock price

```
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
##  6 2015-01-09  493.           6
##  7 2015-01-12  490.           7
##  8 2015-01-13  493.           8
```

Google stock price

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

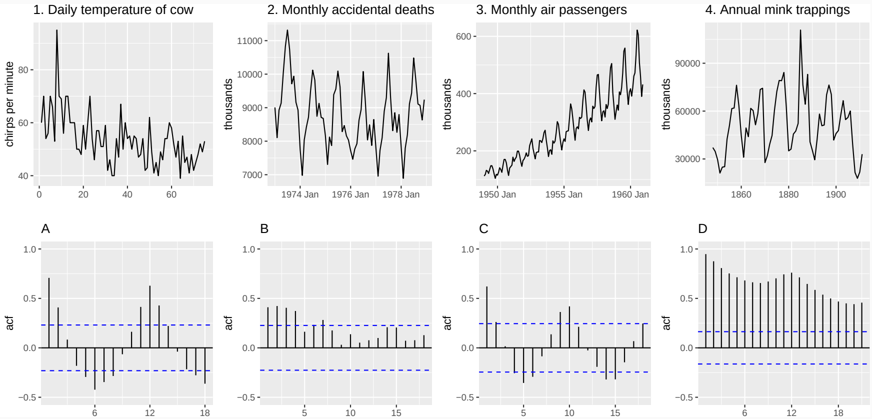


Outline

Lab Session 4

We have introduced the following functions: `gg_lag` and `ACF`. Use these functions to explore the four time series: `Bricks` from `aus_production`, `Lynx` from `pelt`, `Close` price of Amazon from `gafa_stock`, `Demand` from `vic_elec`. Can you spot any seasonality, cyclicity and trend? What do you learn about the series?

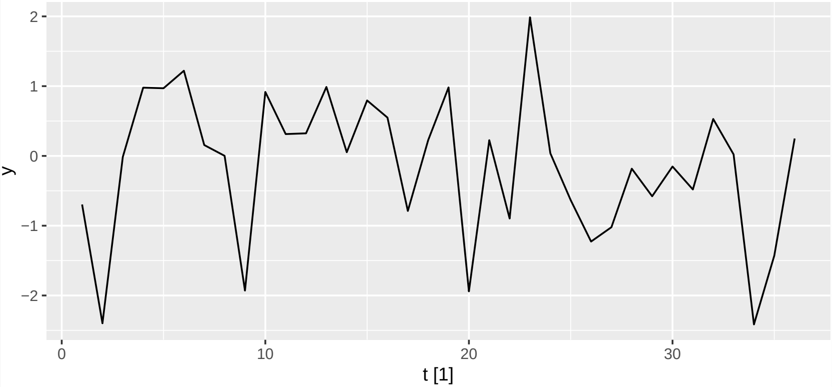
Which is which?



Outline

Example: White noise

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



Example: White noise

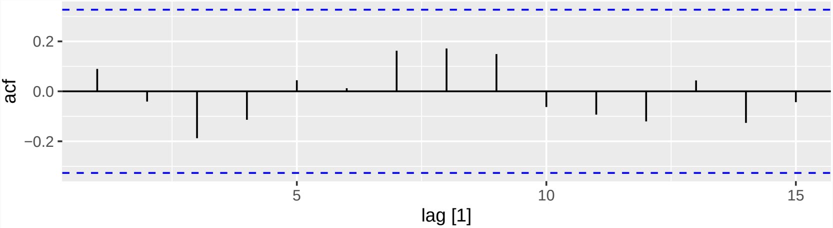
```r
wn <- tsibble(t = seq(36), y = rnorm(36), index = t)
wn %>% autoplot(y)
```

White noise data is uncorrelated across time with zero mean and constant variance. (Technically, we require independence as well.)

Example: White noise

```
wn %>% ACF(y)
```

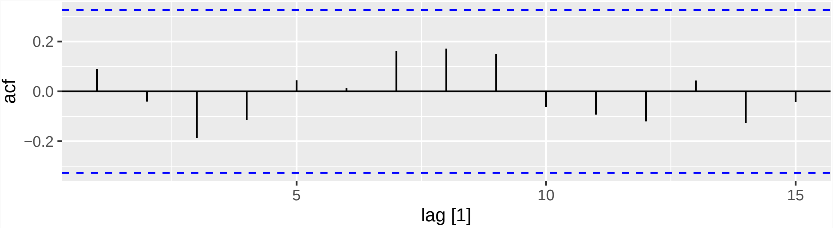
r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}
0.090	-0.041	-0.188	-0.114	0.044	0.012	0.162	0.172	0.149	-0.063



Example: White noise

```r
wn %>% ACF(y)
```

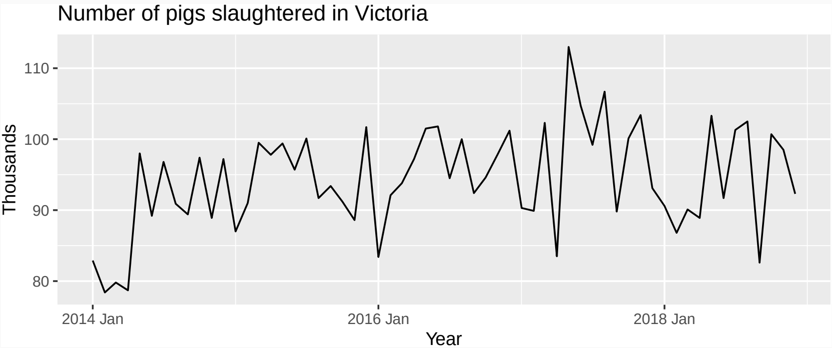
r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}
0.090	-0.041	-0.188	-0.114	0.044	0.012	0.162	0.172	0.149	-0.063



- Sample autocorrelations for white noise series.
- Expect each autocorrelation to be close to zero.
- Blue lines show 95% critical values.

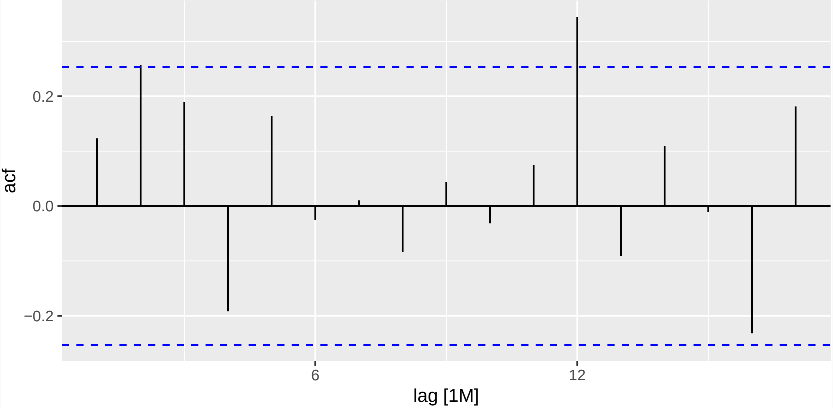
Example: Pigs slaughtered

```
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**.

Outline

Lab Session 5

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?