

Tidy Time Series & Forecasting in R



2. Time series graphics

Outline

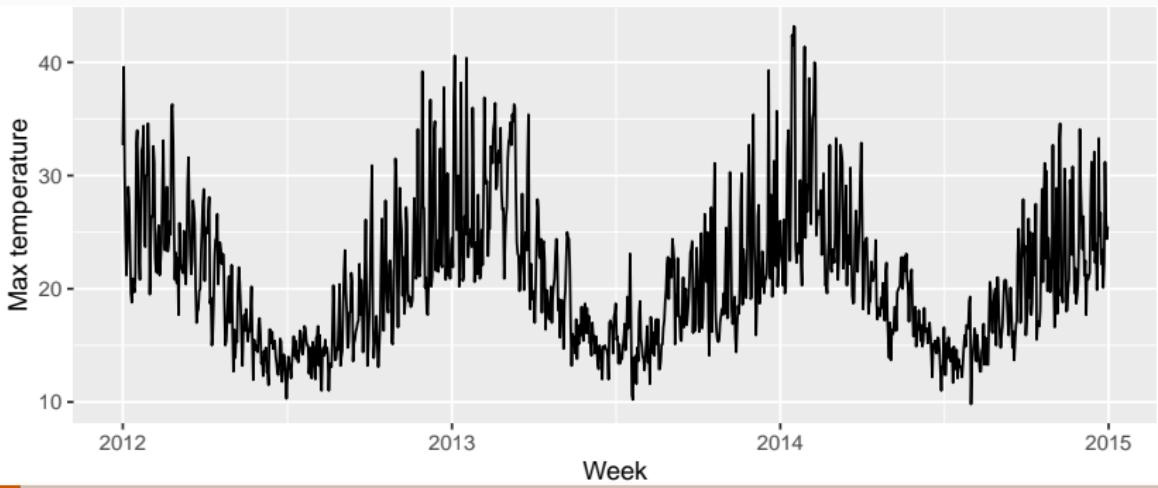
- 1 Time plots
- 2 Lab Session 2
- 3 Seasonal plots
- 4 Lab Session 3
- 5 Decompositions
- 6 Lab Session 4

Outline

- 1 Time plots
- 2 Lab Session 2
- 3 Seasonal plots
- 4 Lab Session 3
- 5 Decompositions
- 6 Lab Session 4

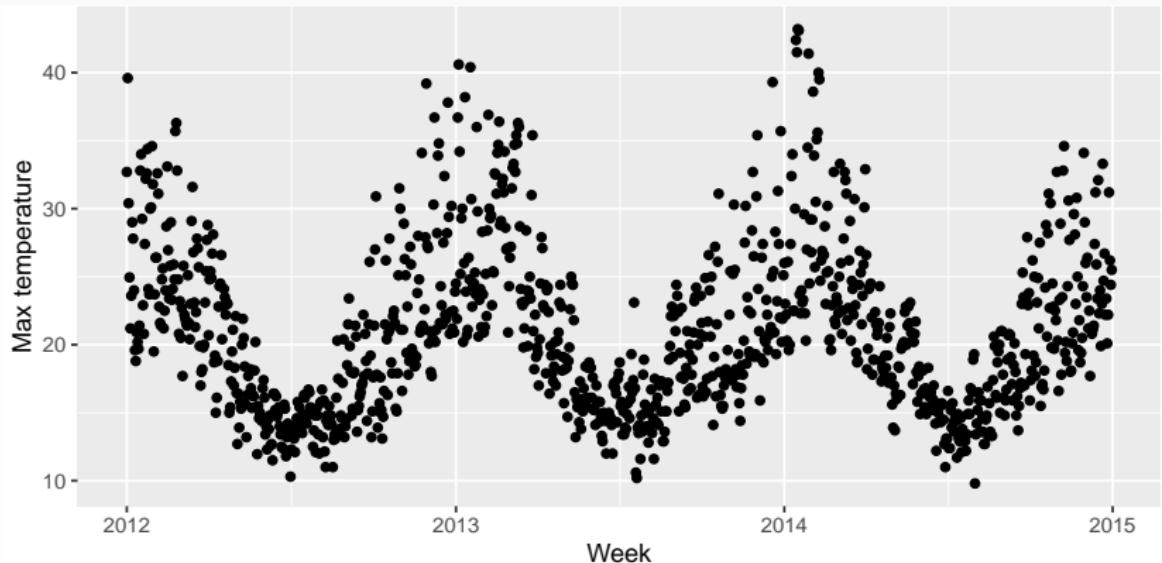
Are line plots best?

```
maxtemp <- vic_elec %>%  
  index_by(Day = date(Time)) %>%  
  summarise(Temperature = max(Temperature))  
maxtemp %>%  
  autoplot(Temperature) +  
  xlab("Week") + ylab("Max temperature")
```



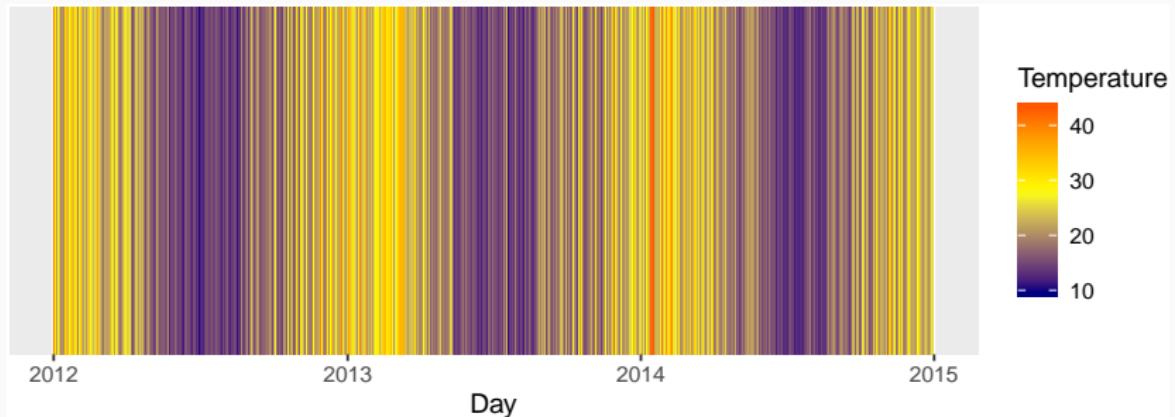
Are line plots best?

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

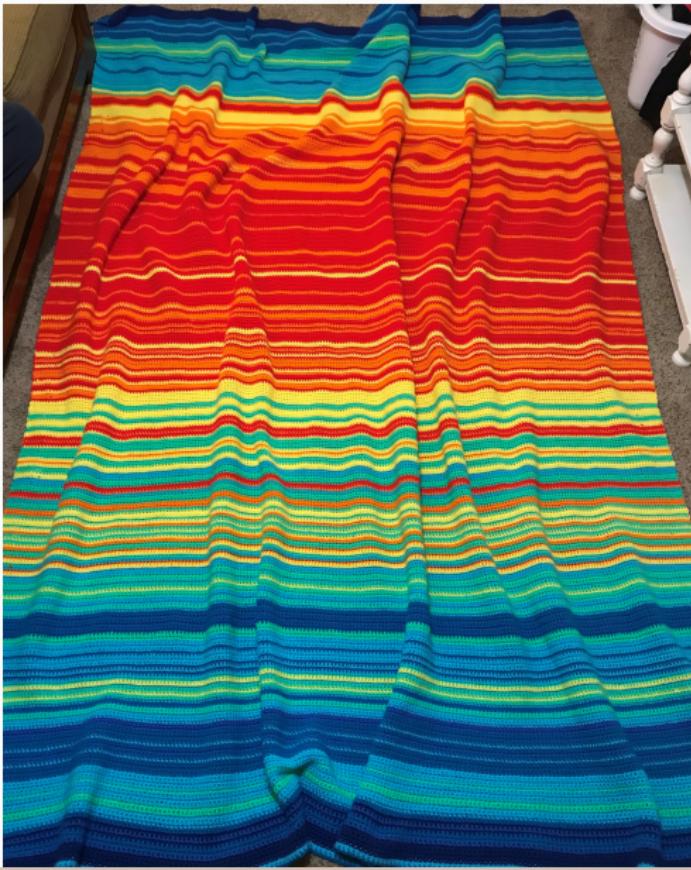


Are line plots best?

```
maxtemp %>%
  ggplot(aes(x = Day, y = 1)) +
  geom_tile(aes(fill = Temperature)) +
  scale_fill_gradient2(low = "navy", mid = "yellow",
                        high = "red", midpoint=28) +
  ylab("") + scale_y_discrete(expand=c(0,0))
```



Are line plots best?



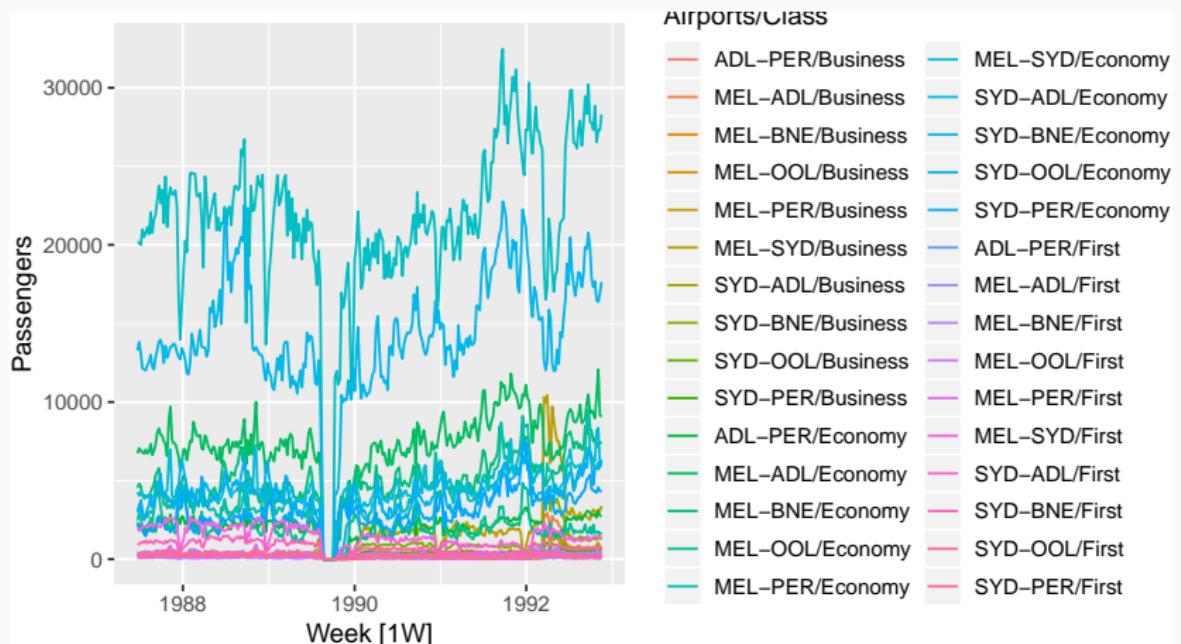
Ansett airlines



Ansett airlines

ansett %>%

autoplott(Passengers)

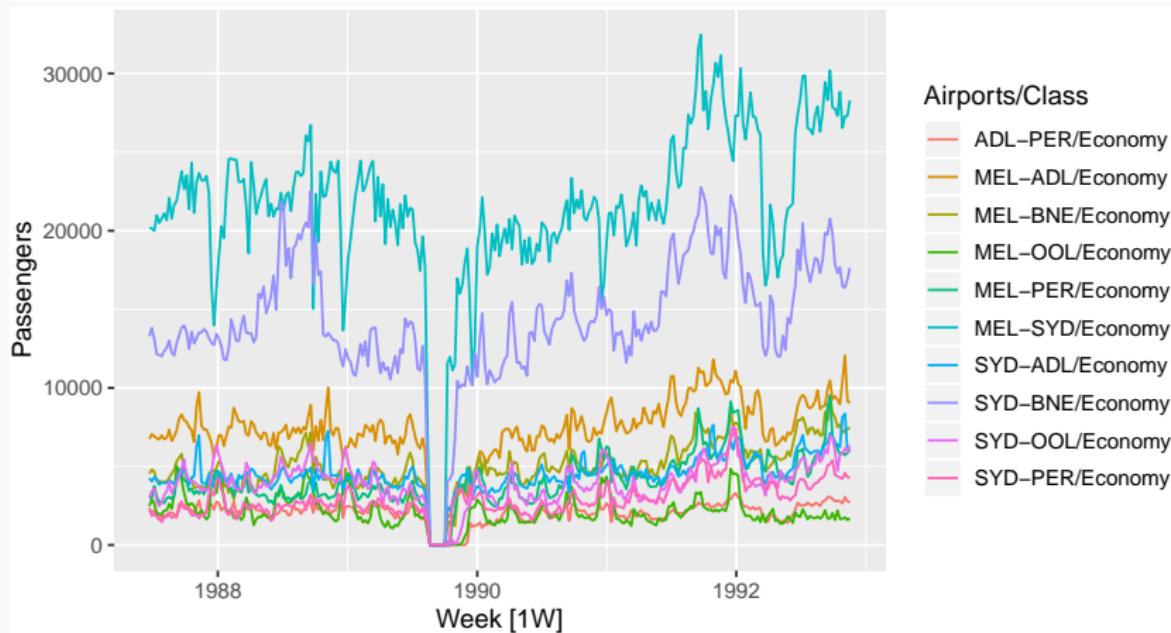


Ansett airlines

```
ansett %>%
```

```
  filter(Class=="Economy") %>%
```

```
  autoplot(Passengers)
```

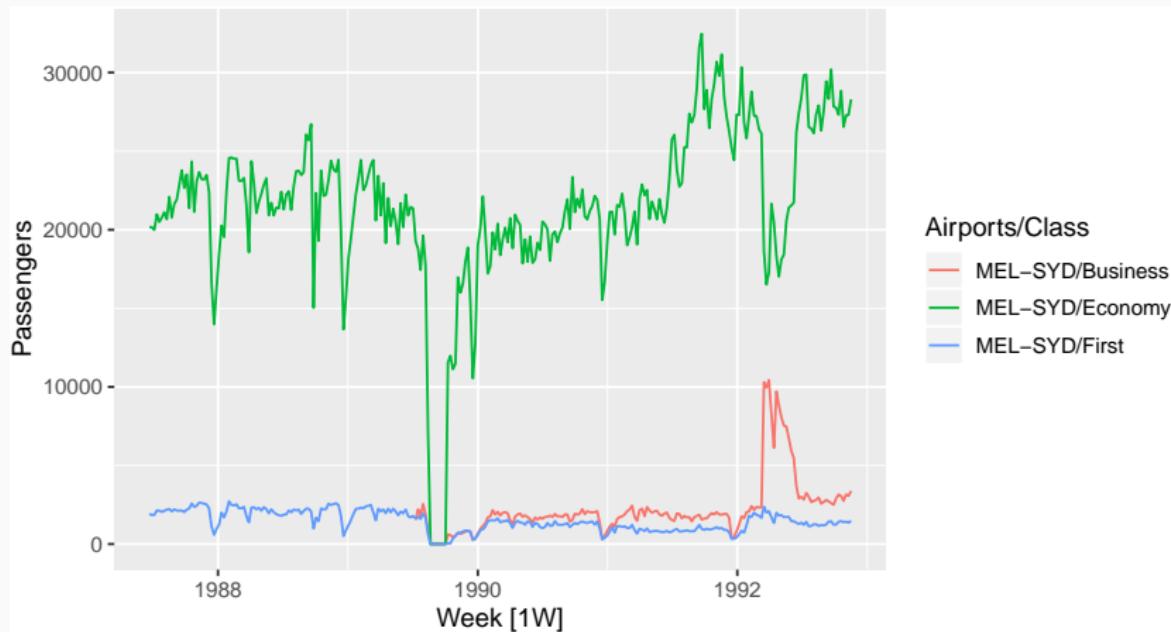


Ansett airlines

```
ansett %>%
```

```
  filter(Airports=="MEL-SYD") %>%
```

```
  autoplot(Passengers)
```

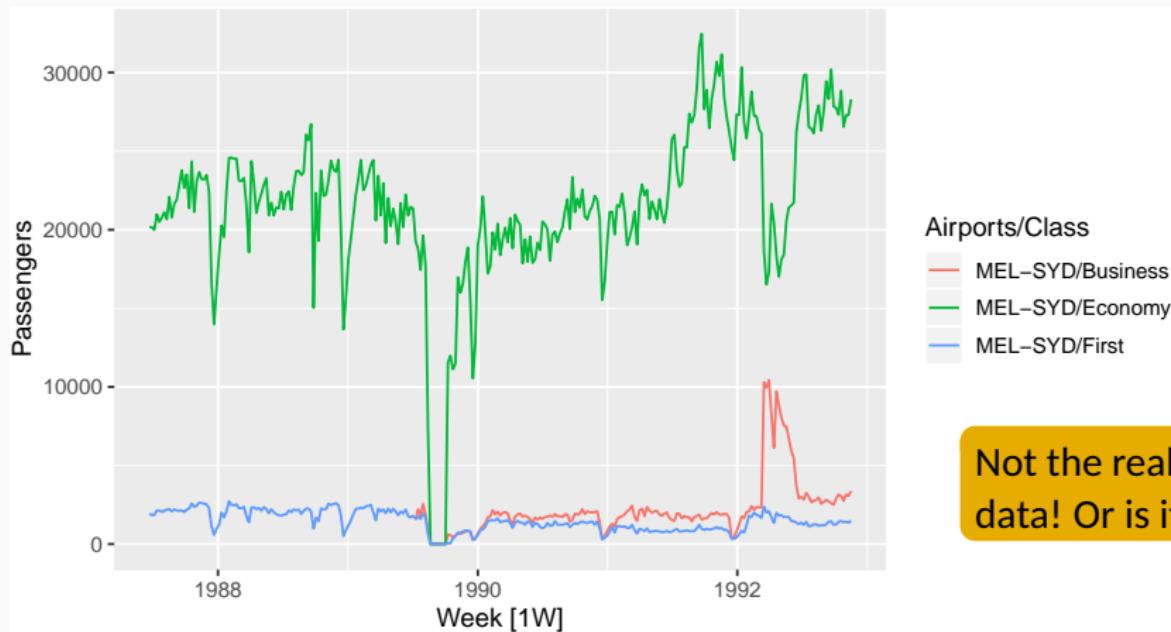


Ansett airlines

```
ansett %>%
```

```
  filter(Airports=="MEL-SYD") %>%
```

```
  autoplot(Passengers)
```



Outline

- 1 Time plots
- 2 Lab Session 2
- 3 Seasonal plots
- 4 Lab Session 3
- 5 Decompositions
- 6 Lab Session 4

Lab Session 2

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

Outline

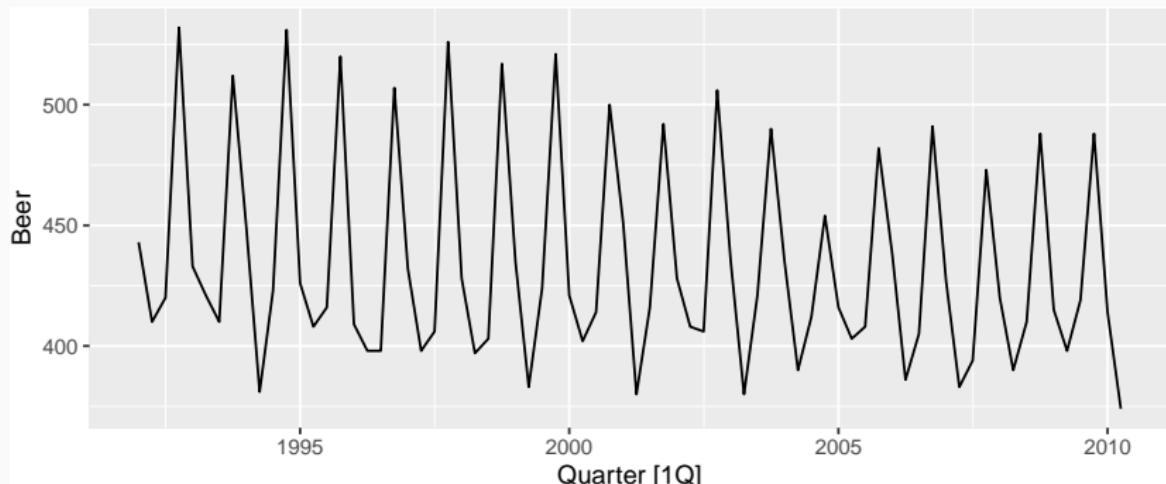
- 1 Time plots
- 2 Lab Session 2
- 3 Seasonal plots
- 4 Lab Session 3
- 5 Decompositions
- 6 Lab Session 4

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()`

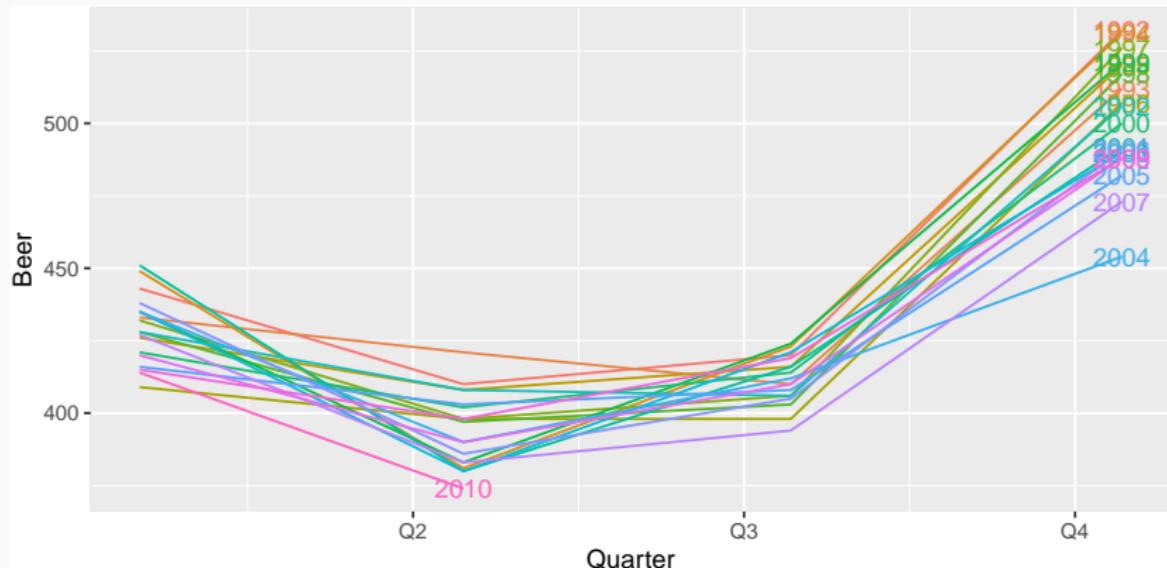
Quarterly Australian Beer Production

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



Quarterly Australian Beer Production

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



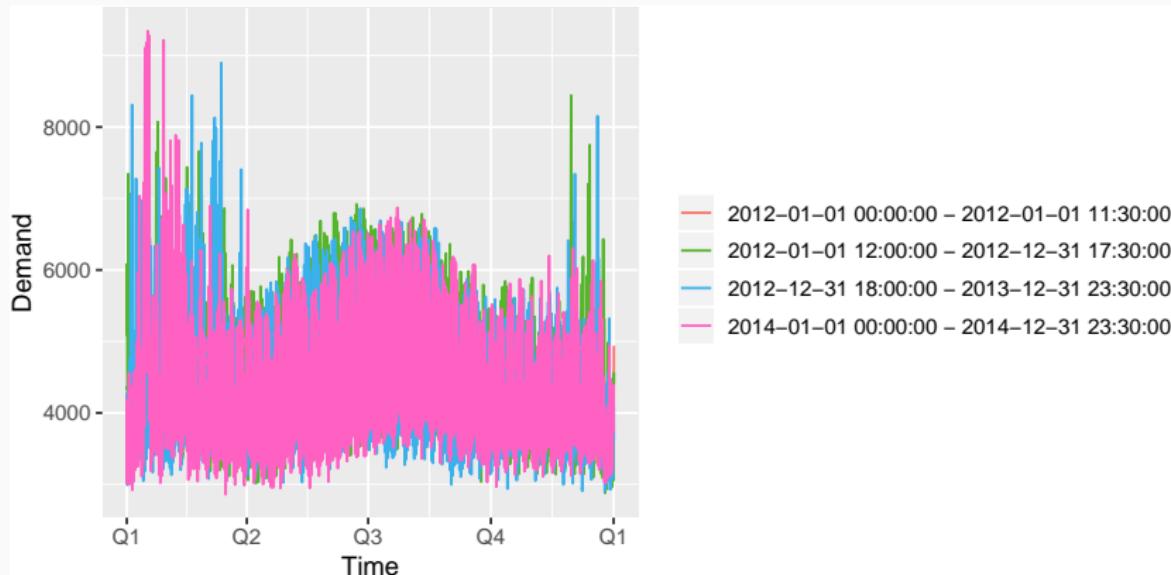
Multiple seasonal periods

```
vic_elec
```

```
## # A tsibble: 52,608 x 5 [30m] <UTC>
##   Time           Demand Temperature Date     Holiday
##   <dttm>        <dbl>      <dbl> <date>    <lgl>
## 1 2012-01-01 00:00:00 4263.      21.0 2012-01-01 TRUE
## 2 2012-01-01 00:30:00 4049.      20.7 2012-01-01 TRUE
## 3 2012-01-01 01:00:00 3878.      20.6 2012-01-01 TRUE
## 4 2012-01-01 01:30:00 4036.      20.4 2012-01-01 TRUE
## 5 2012-01-01 02:00:00 3866.      20.2 2012-01-01 TRUE
## 6 2012-01-01 02:30:00 3694.      20.1 2012-01-01 TRUE
## 7 2012-01-01 03:00:00 3562.      19.6 2012-01-01 TRUE
## 8 2012-01-01 03:30:00 3433.      19.1 2012-01-01 TRUE
## 9 2012-01-01 04:00:00 3359.      19.0 2012-01-01 TRUE
## 10 2012-01-01 04:30:00 3331.      18.8 2012-01-01 TRUE
## # ... with 52,598 more rows
```

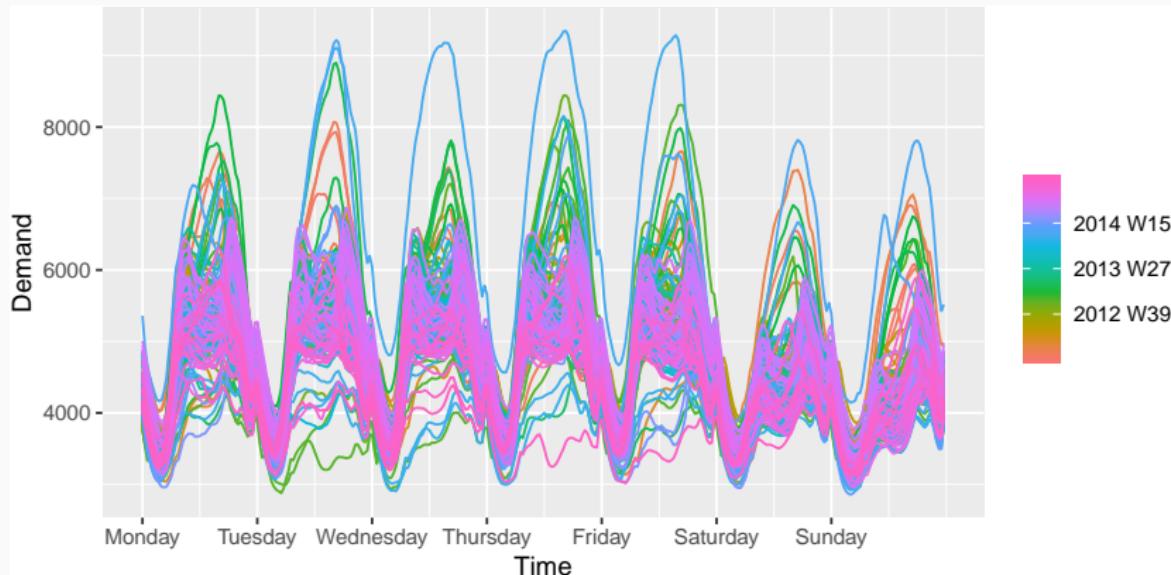
Multiple seasonal periods

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



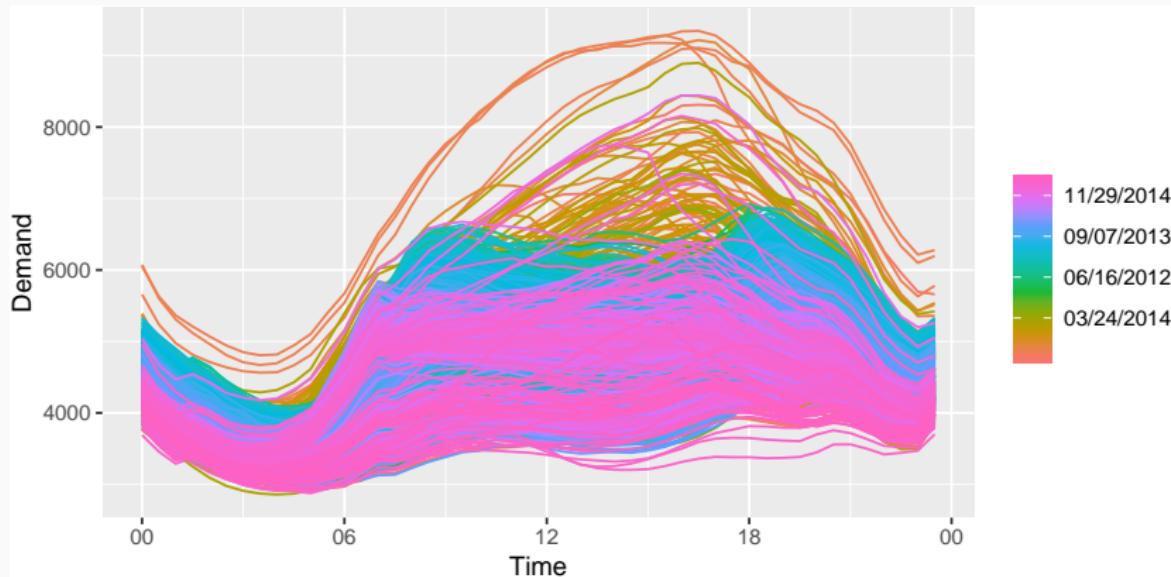
Multiple seasonal periods

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



Multiple seasonal periods

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

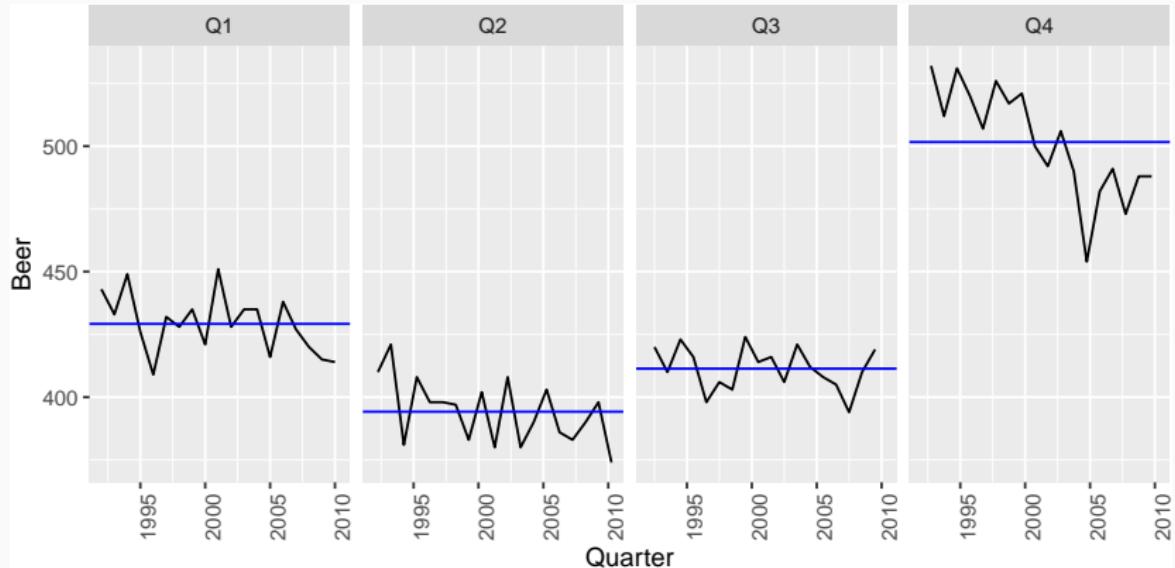


Seasonal subseries plots

- 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

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



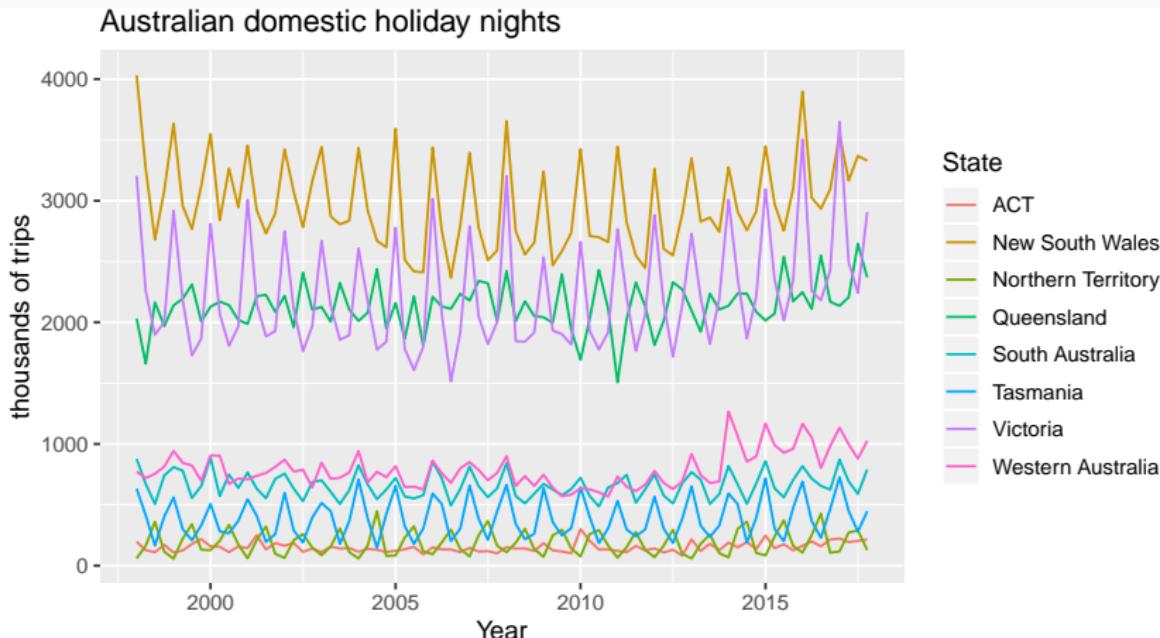
Australian holidays

```
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.
```

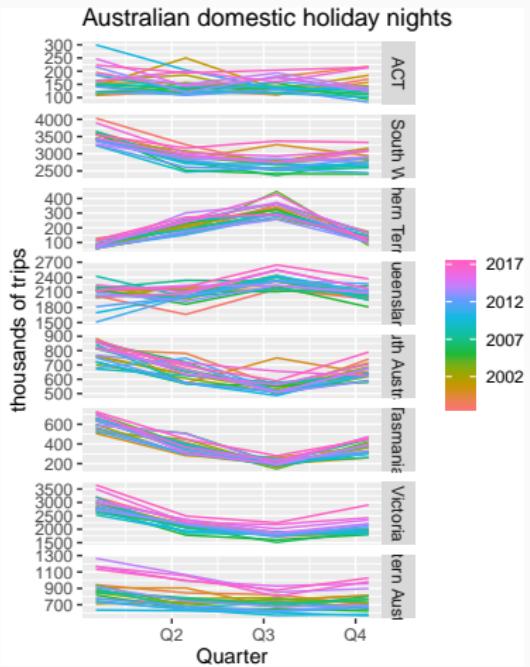
Australian holidays

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



Seasonal plots

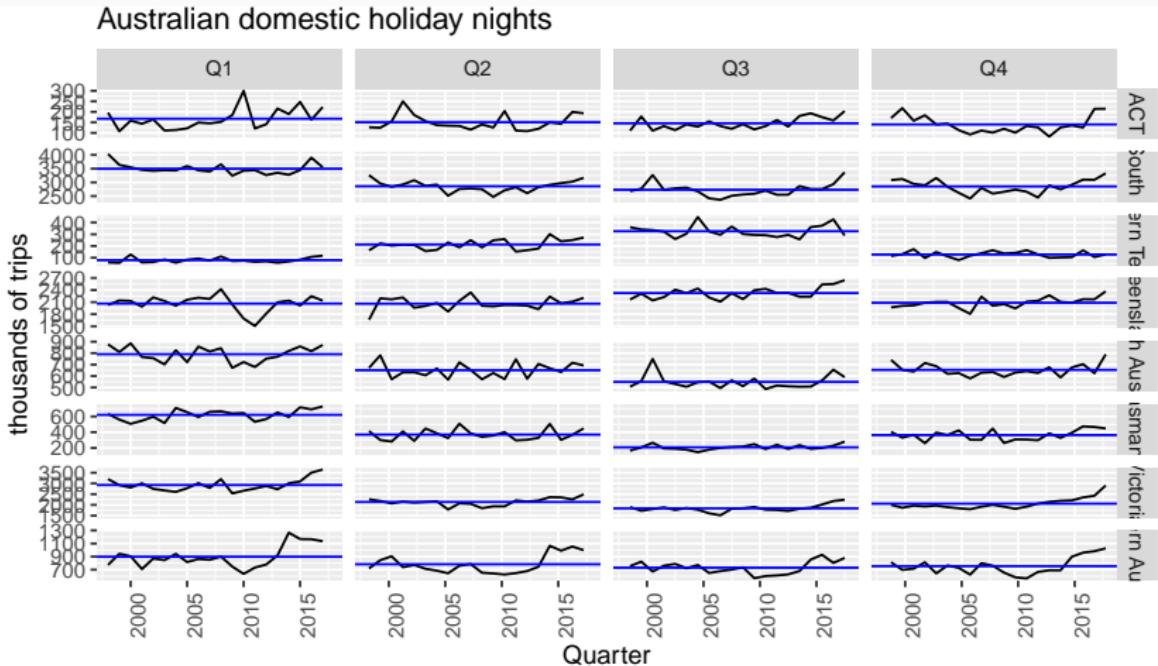
```
holidays %>% gg_season(Trips) +  
  ylab("thousands of trips") +  
  ggtitle("Australian domestic holiday nights")
```



Seasonal subseries plots

holidays %>%

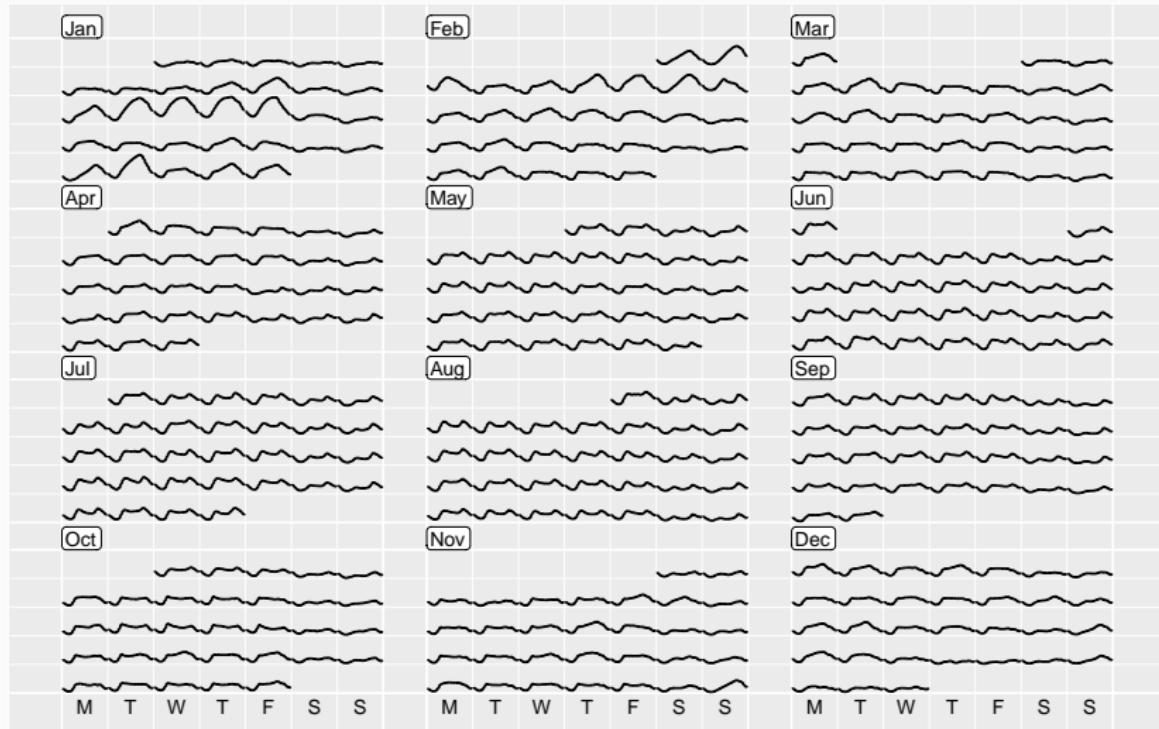
```
gg_subseries(Trips) + ylab("thousands of trips") +  
  ggtitle("Australian domestic holiday nights")
```



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
  pretty(p1, size = 3,
         label.padding = unit(0.15, "lines"))
```

Calendar plots



Outline

- 1 Time plots
- 2 Lab Session 2
- 3 Seasonal plots
- 4 Lab Session 3
- 5 Decompositions
- 6 Lab Session 4

Lab Session 3

Look at the quarterly tourism data for the Snowy Mountains

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

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

Outline

- 1 Time plots
- 2 Lab Session 2
- 3 Seasonal plots
- 4 Lab Session 3
- 5 Decompositions
- 6 Lab Session 4

Time series decomposition

Trend-Cycle aperiodic changes in level over time.

Seasonal (almost) periodic changes in level due to seasonal factors (e.g., the quarter of the year, the month, or day of the week).

Additive decomposition

$$y_t = S_t + T_t + R_t$$

where y_t = data at period t

T_t = trend-cycle component at period t

S_t = seasonal component at period t

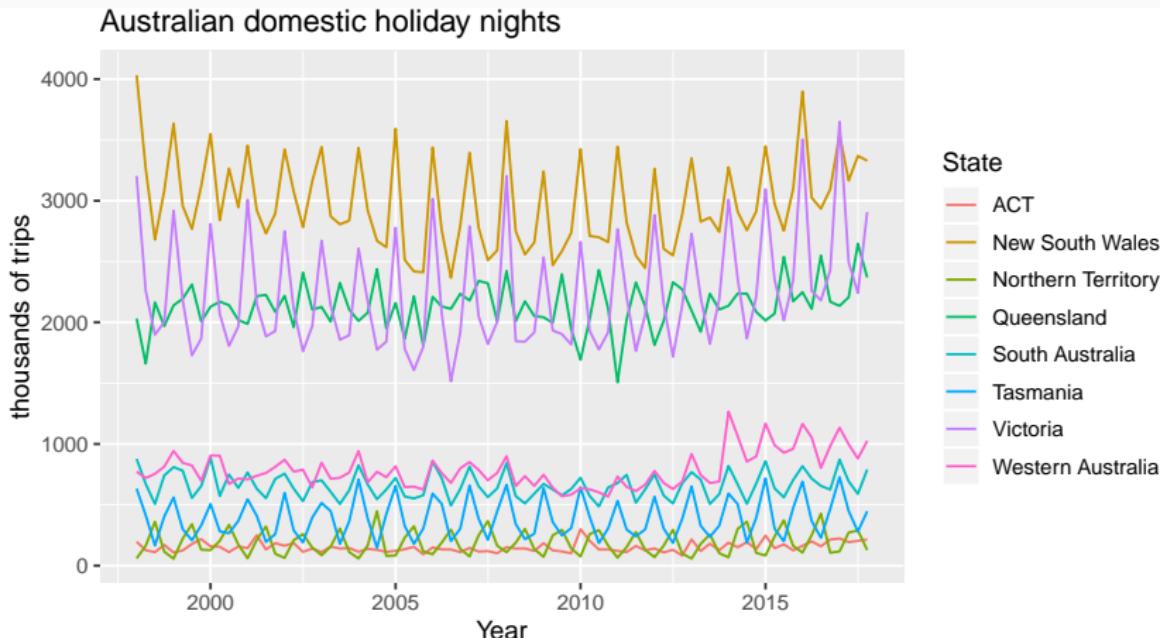
R_t = remainder component at period t

STL decomposition

- STL: “Seasonal and Trend decomposition using Loess”
- Very versatile and robust.
- Seasonal component allowed to change over time, and rate of change controlled by user.
- Smoothness of trend-cycle also controlled by user.
- Optionally robust to outliers
- Not trading day or calendar adjustments.
- Only additive.
- Take logs to get multiplicative decomposition.
- Use Box-Cox transformations to get other decompositions.

Australian holidays

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



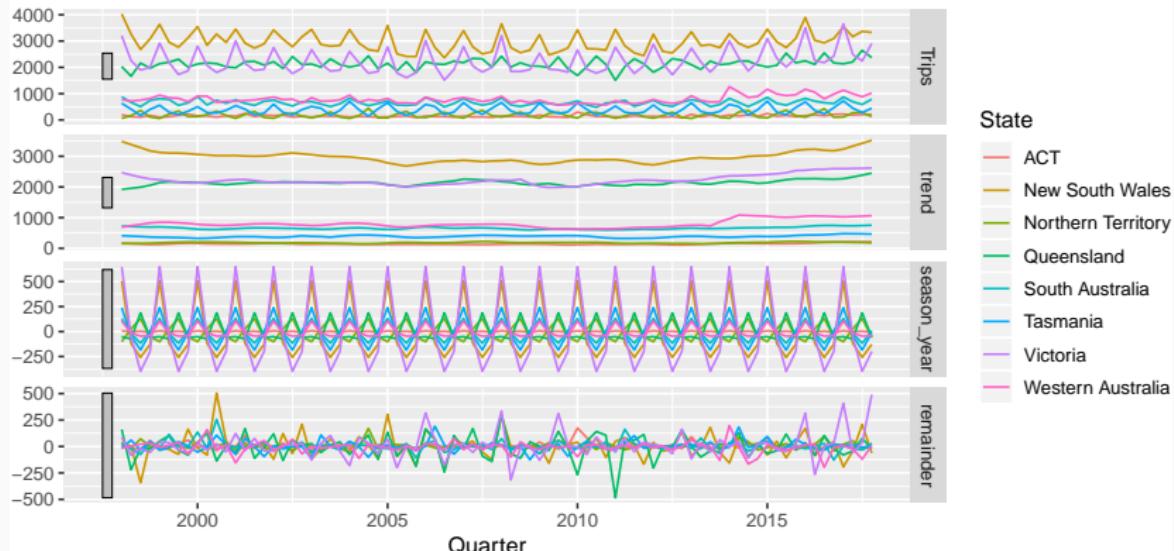
Holidays decomposition

```
holidays %>%
```

```
  STL(Trips ~ season(window="periodic"), robust=TRUE) %>%
  autoplot()
```

STL decomposition

Trips = trend + season_year + remainder



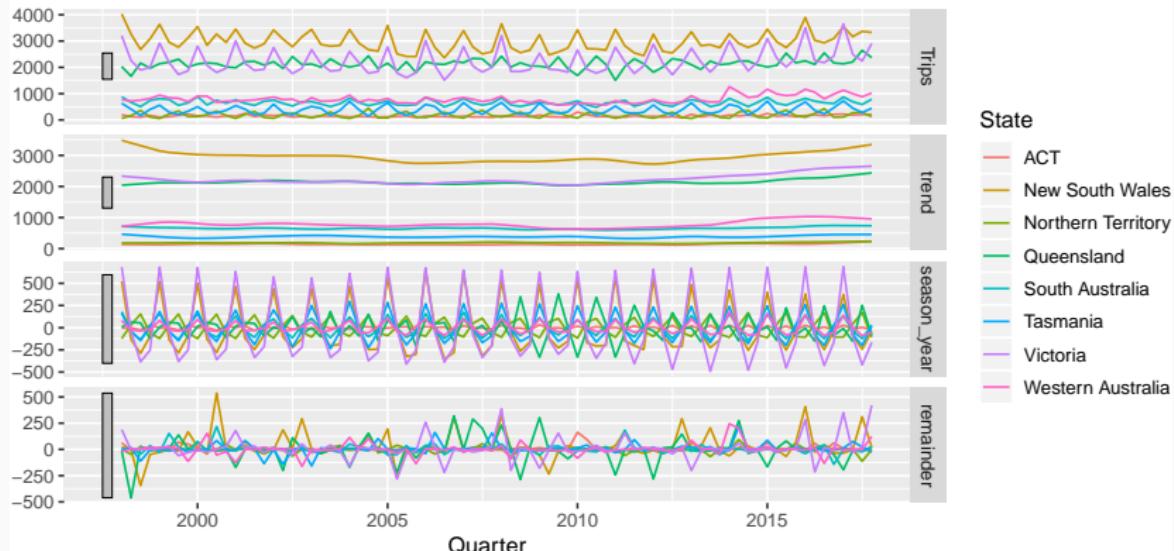
Holidays decomposition

```
holidays %>%
```

```
  STL(Trips ~ season(window = 5), robust = TRUE) %>%  
  autoplot()
```

STL decomposition

Trips = trend + season_year + remainder



STL decomposition

```
holidays %>%
  STL(Trips ~ trend(window=15) + season(window=13),
    robust = TRUE)
```

- `trend(window = ?)` controls wiggliness of trend component.
- `season(window = ?)` controls variation on seasonal component.
- `STL()` chooses `season(window=13)` by default
- A large seasonal window is equivalent to setting `window="periodic"`.
- Odd numbers should be used for symmetry.

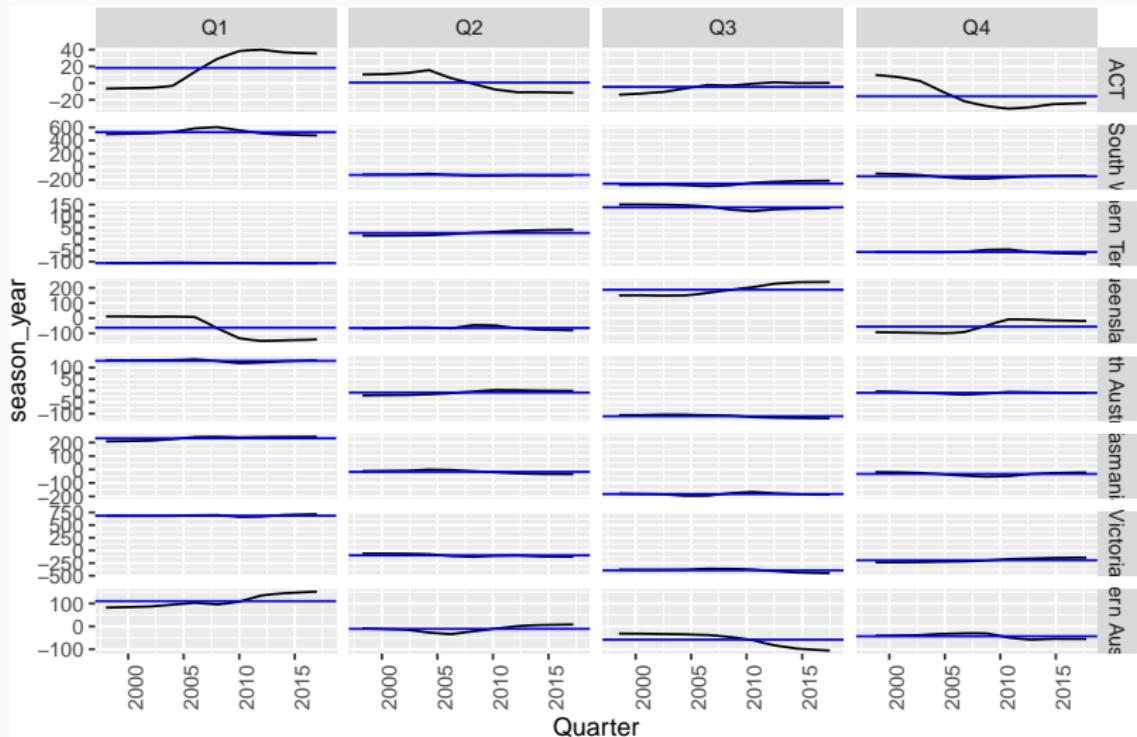
Holidays decomposition

```
dcmp <- holidays %>% STL(Trips)  
dcmp
```

```
## # A dable:          640 x 7 [1Q]  
## # Key:             State [8]  
## # STL Decomposition: Trips = trend + season_year +  
## #   remainder  
## #   State    Quarter Trips trend season_year remainder  
## #   <chr>    <qtr>  <dbl>  <dbl>     <dbl>    <dbl>  
## # 1 ACT      1998 Q1    196.  171.     -6.60    32.3  
## # 2 ACT      1998 Q2    127.  156.      10.3    -39.7  
## # 3 ACT      1998 Q3    111.  142.     -13.9    -17.2  
## # 4 ACT      1998 Q4    170.  130.      9.76    30.3  
## # 5 ACT      1999 Q1    108.  135.     -6.35    -20.7  
## # 6 ACT      1999 Q2    125.  148.      10.5    -33.9  
## # 7 ACT      1999 Q3    178.  166.     -13.2    25.5  
## # 8 ACT      1999 Q4    218.  177.      8.56    32.0  
## # 9 ACT      2000 Q1    158.  169.     -6.09    -4.74  
## # 10 ACT     2000 Q2    155.  151.      10.7    -7.00
```

Holidays decomposition

```
dcmp %>% gg_subseries(season_year)
```

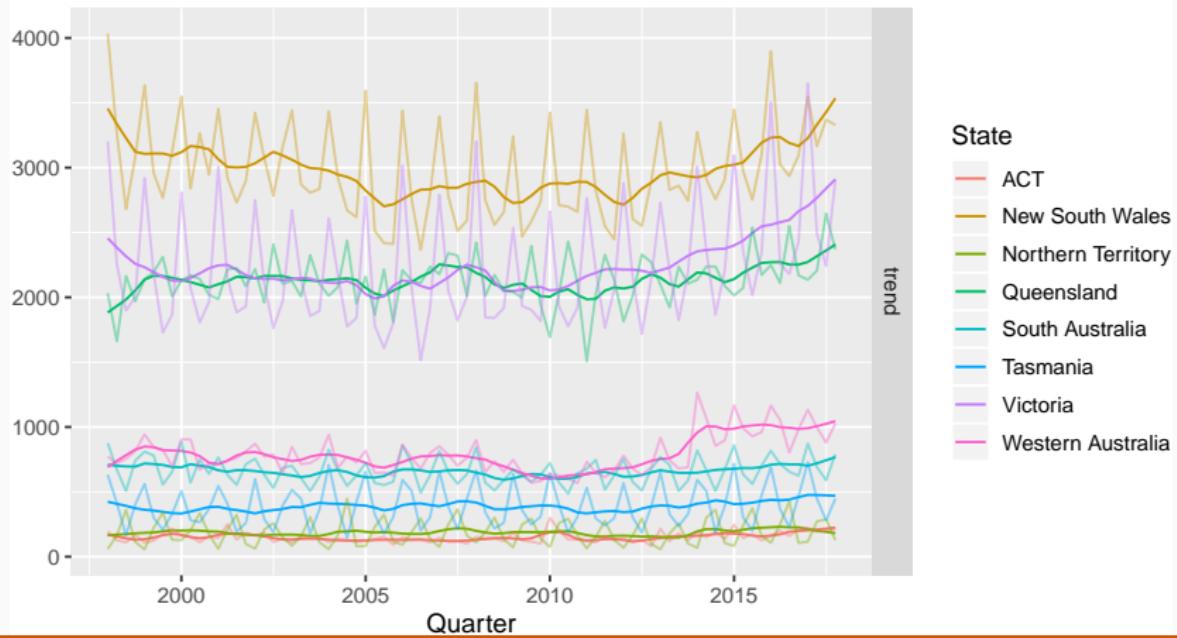


Holidays decomposition

```
autoplot(dcmp, trend, scale_bars=FALSE) +  
  autolayer(holidays, alpha=0.4)
```

STL decomposition

trend



Outline

- 1 Time plots
- 2 Lab Session 2
- 3 Seasonal plots
- 4 Lab Session 3
- 5 Decompositions
- 6 Lab Session 4

Lab Session 4

Repeat the decomposition using

```
holidays %>%  
  STL(Trips ~ season(window=7) + trend(window=11)) %>%  
  autoplot()
```

What happens as you change `season(window = ???)` and `trend(window = ???)`?