MONASH University

MONASH BUSINESS SCHOOL

Forecasting: principles and practice

Rob J Hyndman

1.2 Time series graphics

Outline

`ts` objects and `ts` function

A time series is stored in a `ts` object in R:

- a list of numbers
- information about times those numbers were recorded.

Example

Year	Observation
2012	123
2013	39
2014	78
2015	52
2016	110

```
y <- ts(c(123,39,78,52,110), start=2012)
```

`ts` objects and `ts` function

For observations that are more frequent than once per year, add a `frequency` argument.

E.g., monthly data stored as a numerical vector `z`:

```
y <- ts(z, frequency=12, start=c(2003, 1))
```

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual		
Quarterly		
Monthly		
Daily		
Weekly		
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	
Quarterly		
Monthly		
Daily		
Weekly		
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly		
Monthly		
Daily		
Weekly		
Hourly		
Half-hourly		

ts objects and ts function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	
Monthly		
Daily		
Weekly		
Hourly		
Half-hourly		

`ts` objects and `ts` function

```
ts(data, frequency, start)
```

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly		
Daily		
Weekly		
Hourly		
Half-hourly		

ts objects and ts function

ts(data, frequency, start)

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	
Daily		
Weekly		
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily		
Weekly		
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	
Weekly		
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	1 *or* c(1995,234)
Weekly		
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	1 *or* c(1995,234)
Weekly	52.18	
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	1 *or* c(1995,234)
Weekly	52.18	c(1995,23)
Hourly		
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	1 *or* c(1995,234)
Weekly	52.18	c(1995,23)
Hourly	24 *or* 168 *or* 8,766	
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	1 *or* c(1995,234)
Weekly	52.18	c(1995,23)
Hourly	24 *or* 168 *or* 8,766	1
Half-hourly		

`ts` objects and `ts` function

`ts(data, frequency, start)`

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	1 *or* c(1995,234)
Weekly	52.18	c(1995,23)
Hourly	24 *or* 168 *or* 8,766	1
Half-hourly	48 *or* 336 *or* 17,532	

`ts` objects and `ts` function

```
ts(data, frequency, start)
```

Type of data	frequency	start example
Annual	1	1995
Quarterly	4	c(1995,2)
Monthly	12	c(1995,9)
Daily	7 *or* 365.25	1 *or* c(1995,234)
Weekly	52.18	c(1995,23)
Hourly	24 *or* 168 *or* 8,766	1
Half-hourly	48 *or* 336 *or* 17,532	1

Australian GDP

```r
ausgdp <- ts(x, frequency=4, start=c(1971,3))
```

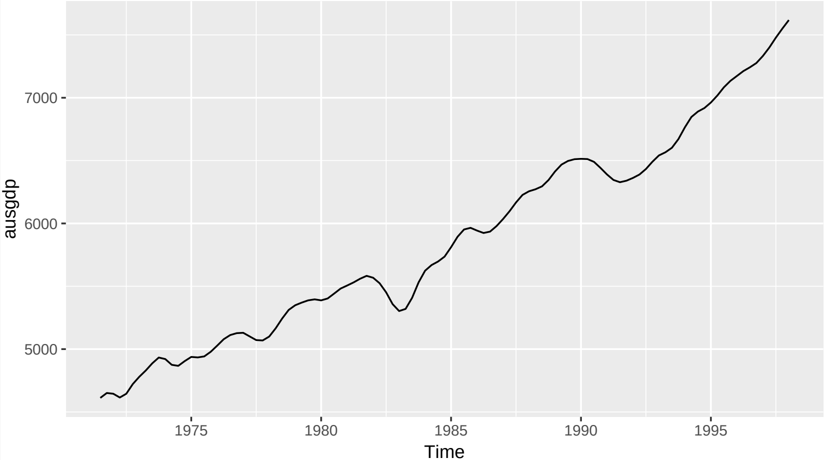
- Class: "ts"
- Print and plotting methods available.

```r
ausgdp
```

```
##      Qtr1 Qtr2 Qtr3 Qtr4
## 1971           4612 4651
## 1972 4645 4615 4645 4722
## 1973 4780 4830 4887 4933
## 1974 4921 4875 4867 4905
## 1975 4938 4934 4942 4979
## 1976 5028 5079 5112 5127
```

Australian GDP

```r
autoplot(ausgdp)
```



Residential electricity sales

```
elecsales
```

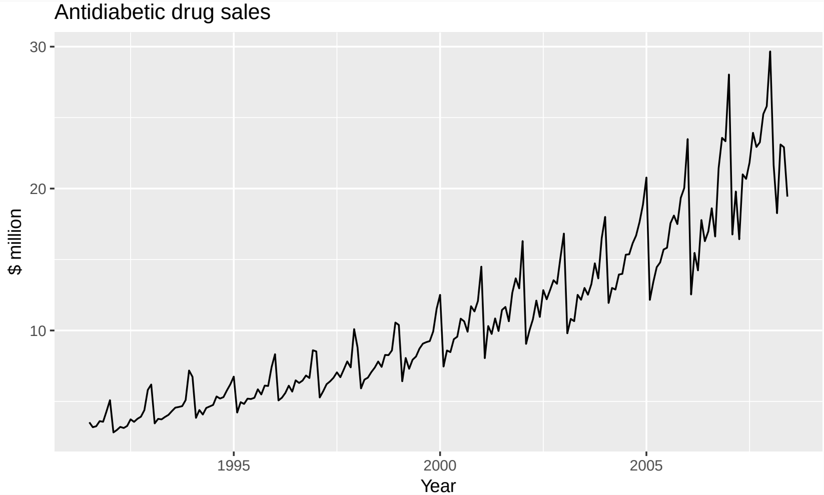
```
## Time Series:
## Start = 1989
## End = 2008
## Frequency = 1
##  [1] 2354.34 2379.71 2318.52 2468.99 2386.0
##  [6] 2569.47 2575.72 2762.72 2844.50 3000.7
## [11] 3108.10 3357.50 3075.70 3180.60 3221.6
## [16] 3176.20 3430.60 3527.48 3637.89 3655.0
```

Time plots

```
autoplot(melsyd[,"Economy.Class"])
```

Time plots

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



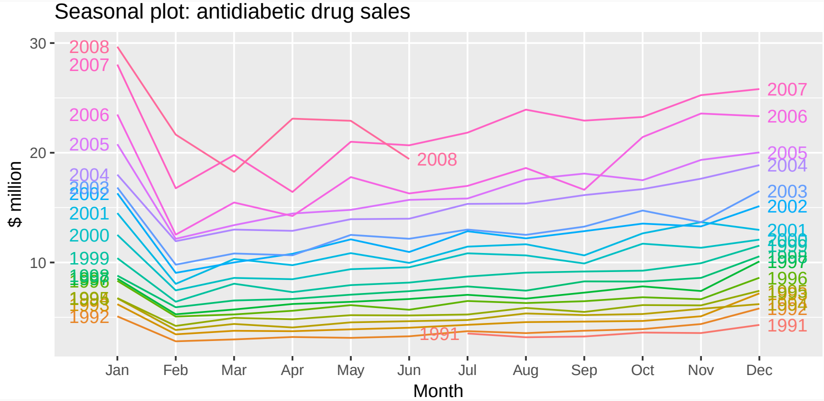
Outline

Lab Session 1

Outline

Seasonal plots

```
ggseasonplot(a10, ylab="$ million",
  year.labels=TRUE, year.labels.left=TRUE) +
  ggtitle("Seasonal plot: antidiabetic drug sales")
```

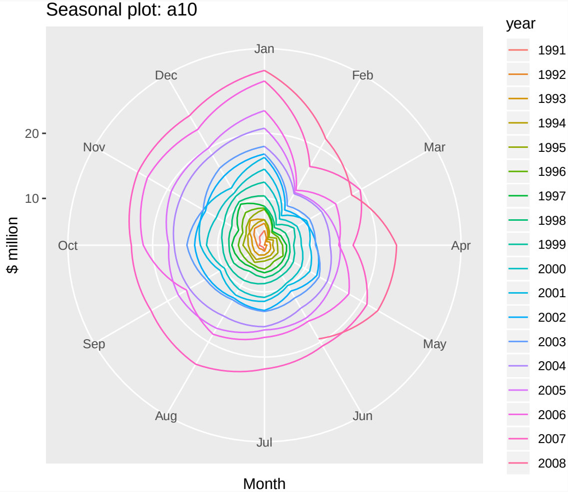


Seasonal plots

- 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: `ggseasonplot`

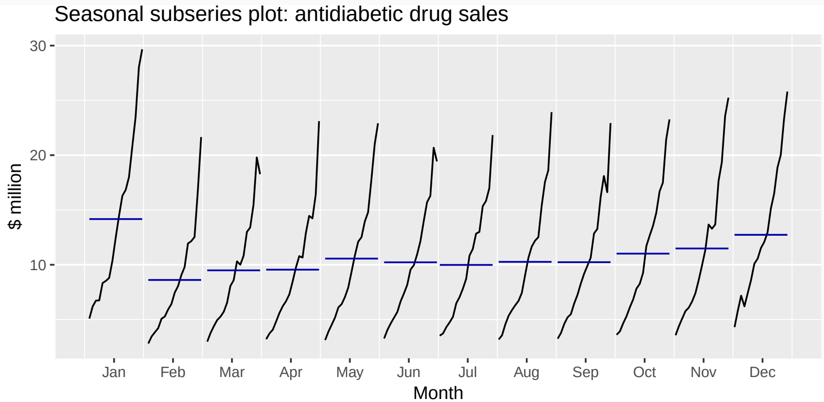
Seasonal polar plots

```r
ggseasonplot(a10, polar=TRUE) + ylab("$ million")
```



Seasonal subseries plots

```
ggsubseriesplot(a10) + ylab("$ million") +
  ggtitle("Seasonal subseries plot: antidiabetic drug sales")
```

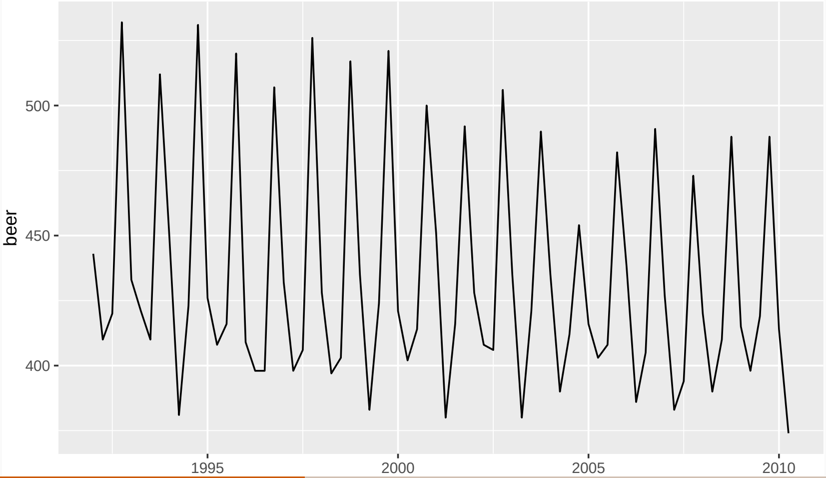


Seasonal subseries plots

- 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: `ggsubseriesplot`

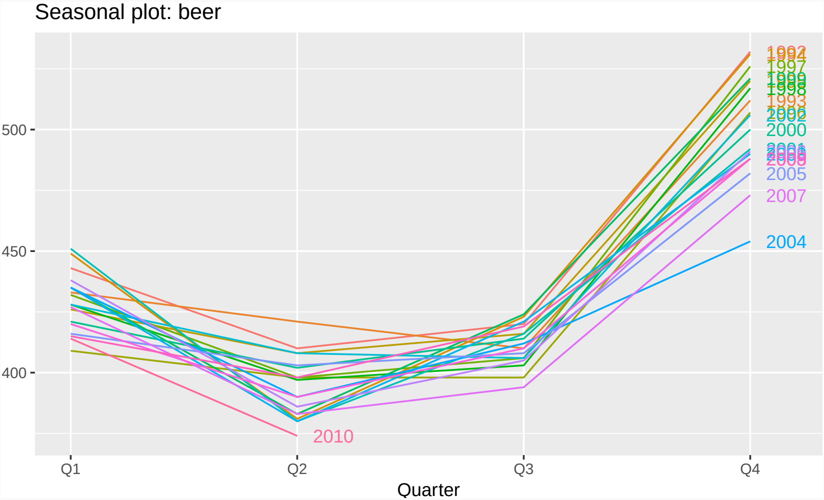
Quarterly Australian Beer Production

```r
beer <- window(ausbeer,start=1992)
autoplot(beer)
```



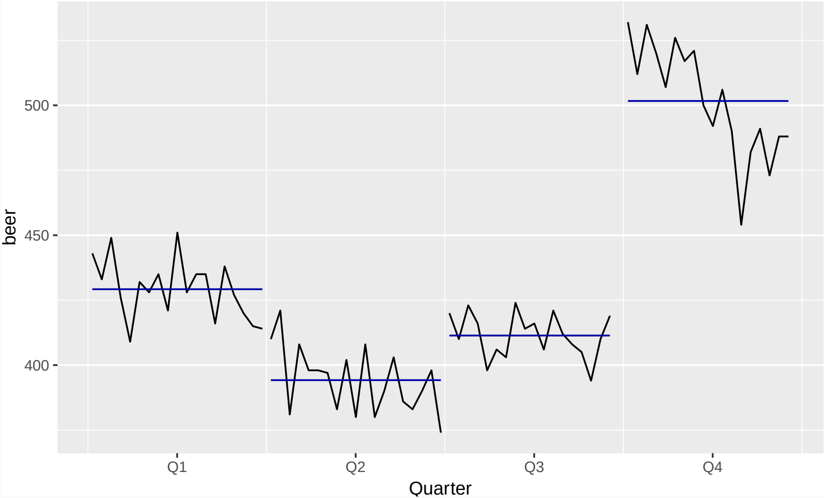
Quarterly Australian Beer Production

```r
ggseasonplot(beer,year.labels=TRUE)
```



Quarterly Australian Beer Production

```
ggsubseriesplot(beer)
```



Outline

Lab Session 2

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).

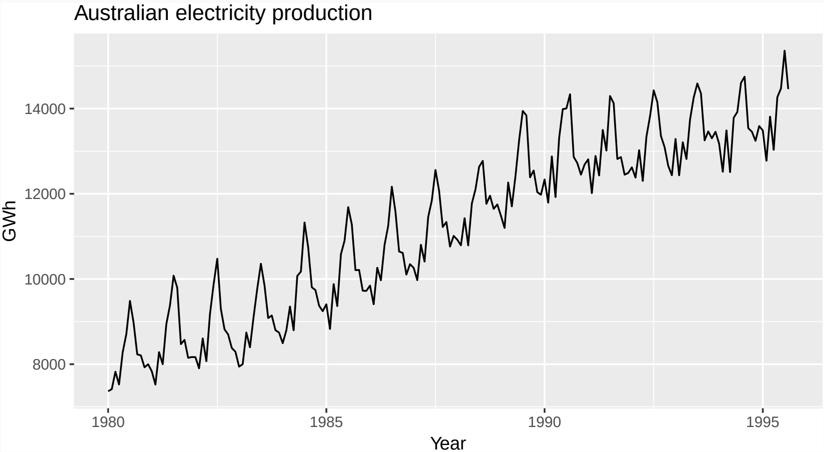
Time series components

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

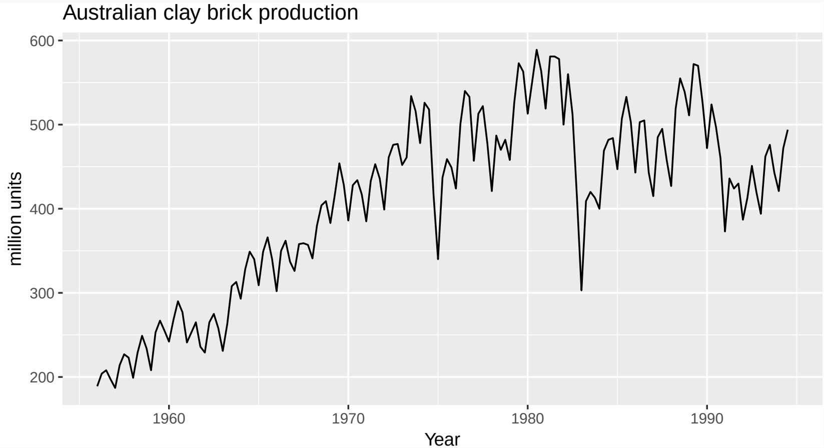
Time series patterns

```r
autoplot(window(elec, start=1980)) +
  ggtitle("Australian electricity production") +
  xlab("Year") + ylab("GWh")
```



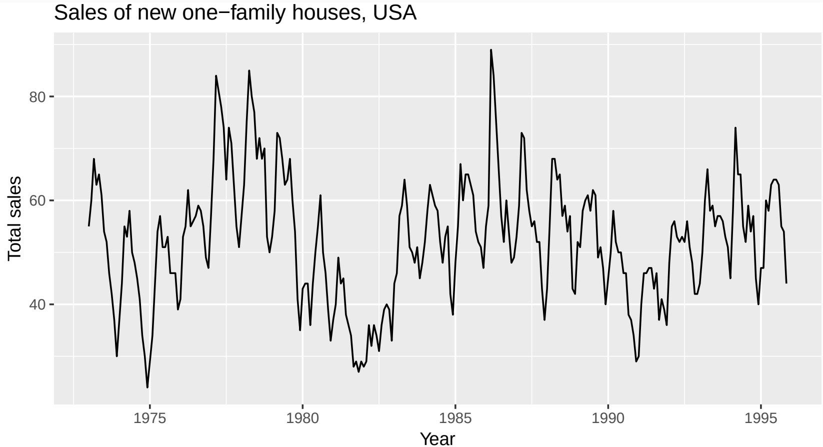
Time series patterns

```r
autoplot(bricksq) +
  ggtitle("Australian clay brick production") +
  xlab("Year") + ylab("million units")
```



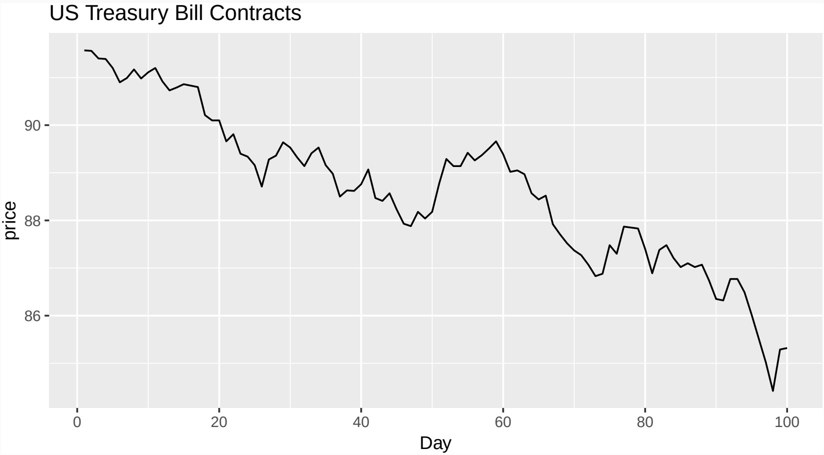
Time series patterns

```r
autoplot(hsales) +
  ggtitle("Sales of new one-family houses, USA") +
  xlab("Year") + ylab("Total sales")
```



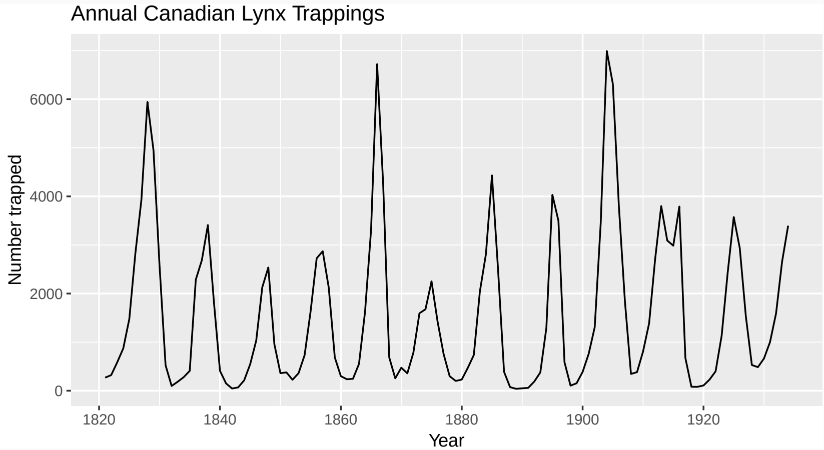
Time series patterns

```r
autoplot(ustreas) +
  ggtitle("US Treasury Bill Contracts") +
  xlab("Day") + ylab("price")
```



Time series patterns

```r
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

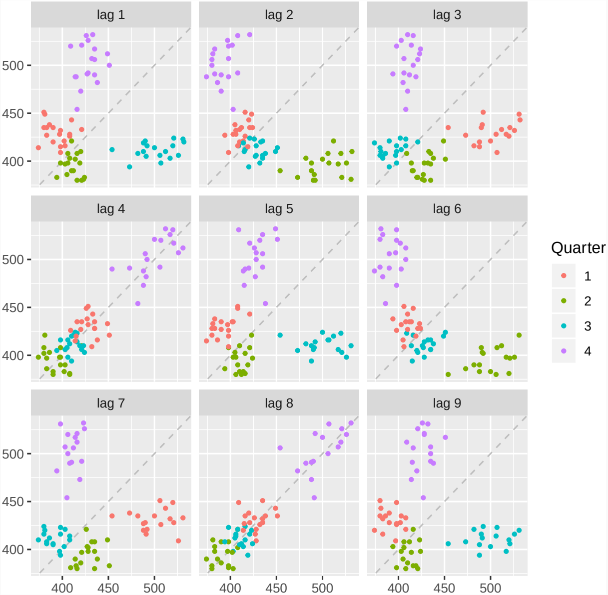
The timing of peaks and troughs is predictable with seasonal data, but unpredictable in the long term with cyclic data.

Outline

Example: Beer production

```r
beer <- window(ausbeer, start=1992)
gglagplot(beer, lags=9, do.lines=FALSE,
  continuous=FALSE)
```

Example: Beer production



Lagged scatterplots

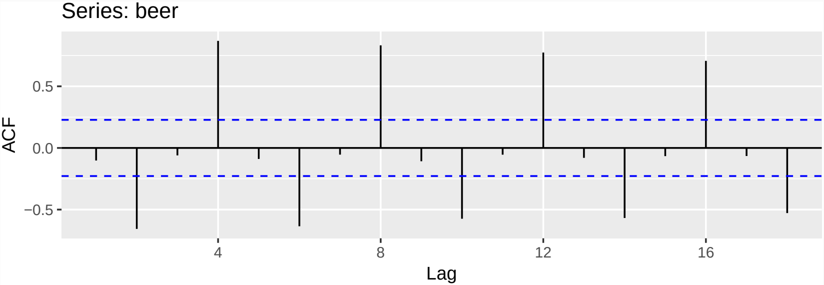
- Each graph shows y_t plotted against y_{t-k} for different values of k.
- The autocorrelations are the correlations associated with these scatterplots.
- ACF (autocorrelation function):
    - r_1 = Correlation(y_t, y_{t-1})
    - r_2 = Correlation(y_t, y_{t-2})
    - r_3 = Correlation(y_t, y_{t-3})
    - 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:

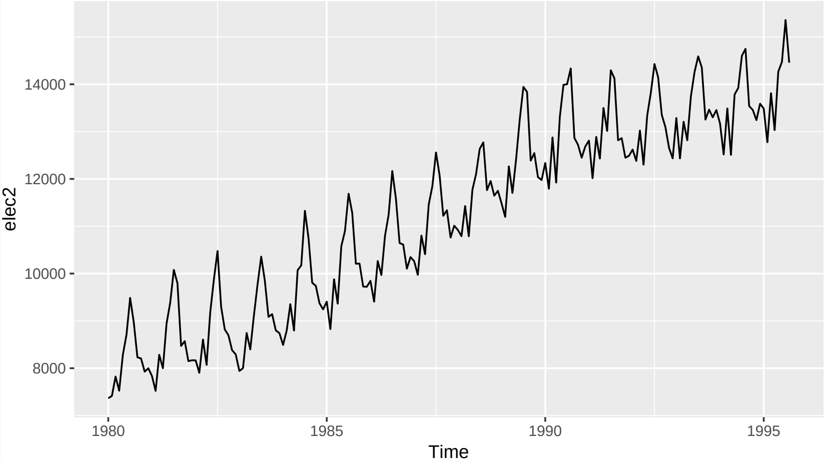
r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9
-0.102	-0.657	-0.060	0.869	-0.089	-0.635	-0.054	0.832	-0.108

```r
ggAcf(beer)
```



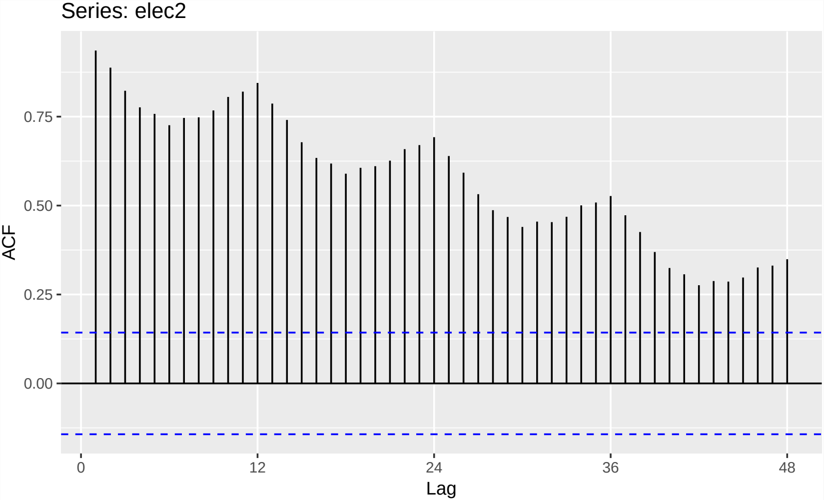
Aus monthly electricity production

```
elec2 <- window(elec, start=1980)
autoplot(elec2)
```



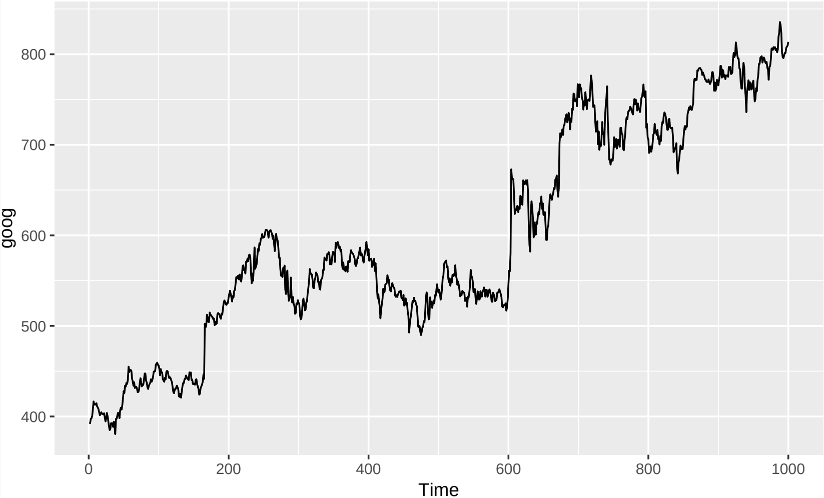
Aus monthly electricity production

```r
ggAcf(elec2, lag.max=48)
```



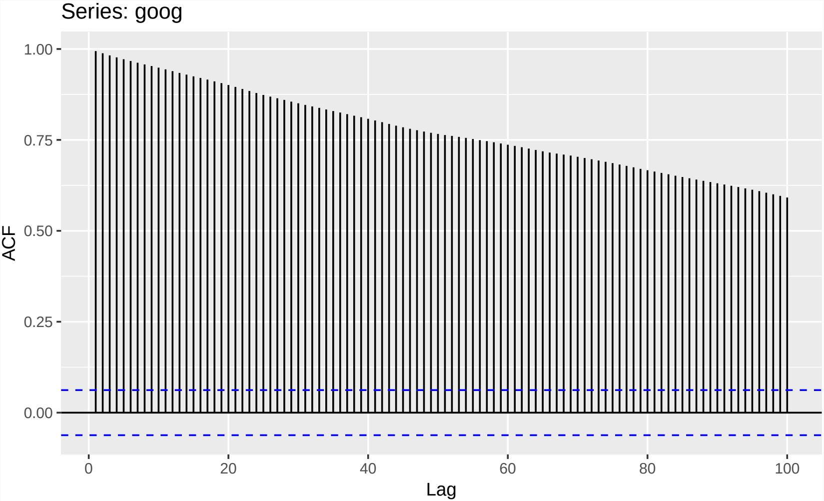
Google stock price

```
autoplot(goog)
```

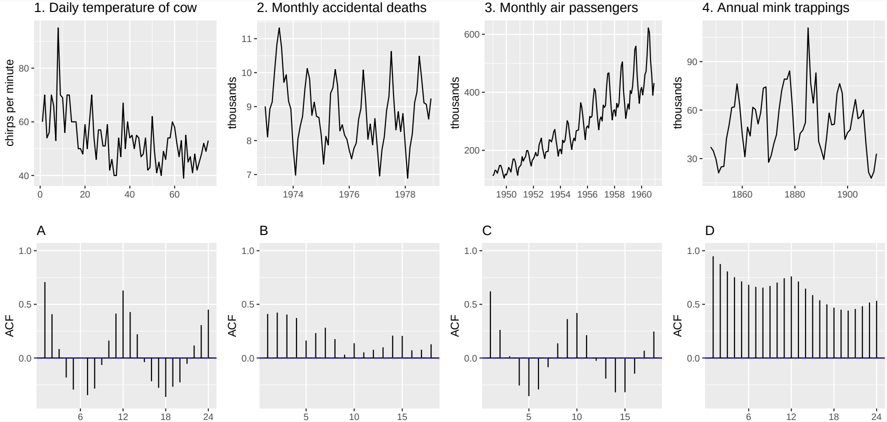


Google stock price

```
ggAcf(goog, lag.max=100)
```



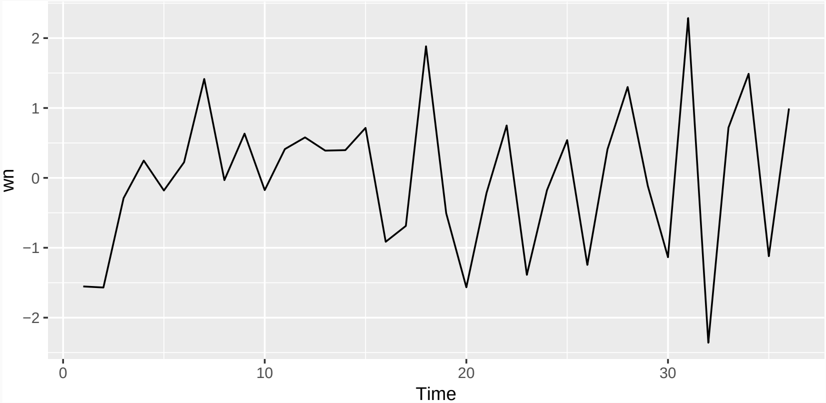
Which is which?



Outline

Example: White noise

```r
wn <- ts(rnorm(36))
autoplot(wn)
```



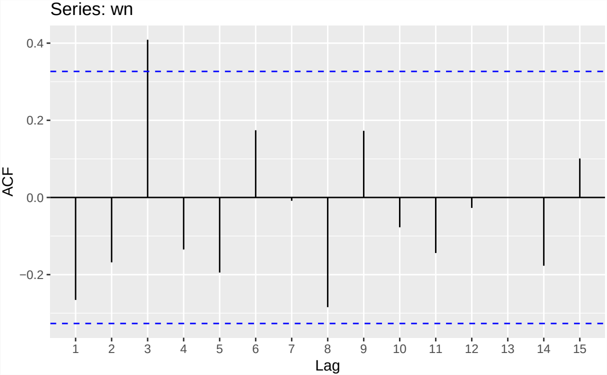
Example: White noise

```r
wn <- ts(rnorm(36))
autoplot(wn)
```

White noise data is uncorrelated across time with zero mean and constant variance.
(Technically, we require independence as well7)

Example: White noise

r_1	-0.27
r_2	-0.17
r_3	0.41
r_4	-0.13
r_5	-0.19
r_6	0.17
r_7	-0.01
r_8	-0.28
r_9	0.17
r_{10}	-0.08



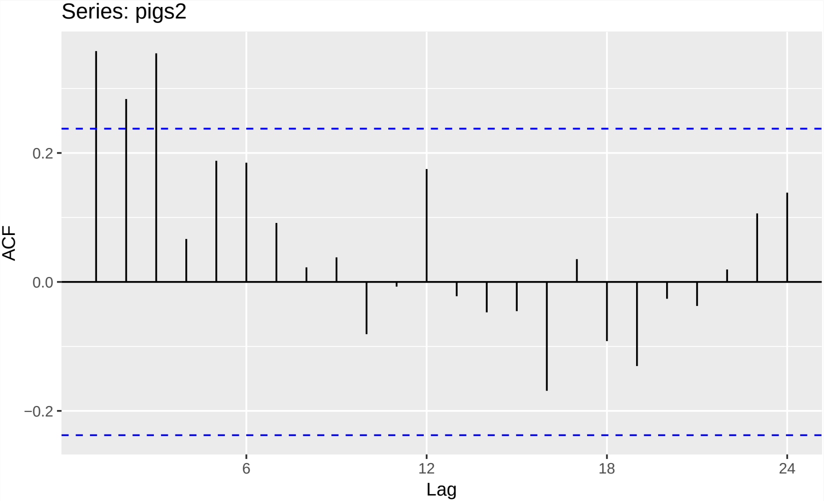
- Sample autocorrelations for white noise series.
- Expect each autocorrelation to be close to zero.
- Blue lines show 95% critical values.

Example: Pigs slaughtered

```
pigs2 <- window(pigs, start=1990)
autoplot(pigs2) +
  xlab("Year") + ylab("thousands") +
  ggtitle("Number of pigs slaughtered in Victoria")
```

Example: Pigs slaughtered

```r
ggAcf(pigs2)
```



Outline

Lab Session 3