Applied Time Series Analysis
Time Series in R and Plots

Prof. Dr. Stephan Trahasch

Source: OTexts.org/fpp2/

Outline

- Introduction
- Time Series in R
- Time plots
- Seasonal plots
- Seasonal or cyclic?
- Lag plots and autocorrelation
- White noise

Applied Time Series: A First Example

In 2006, Singapore Airlines decided to place an order for new aircraft. It contained the following jets:

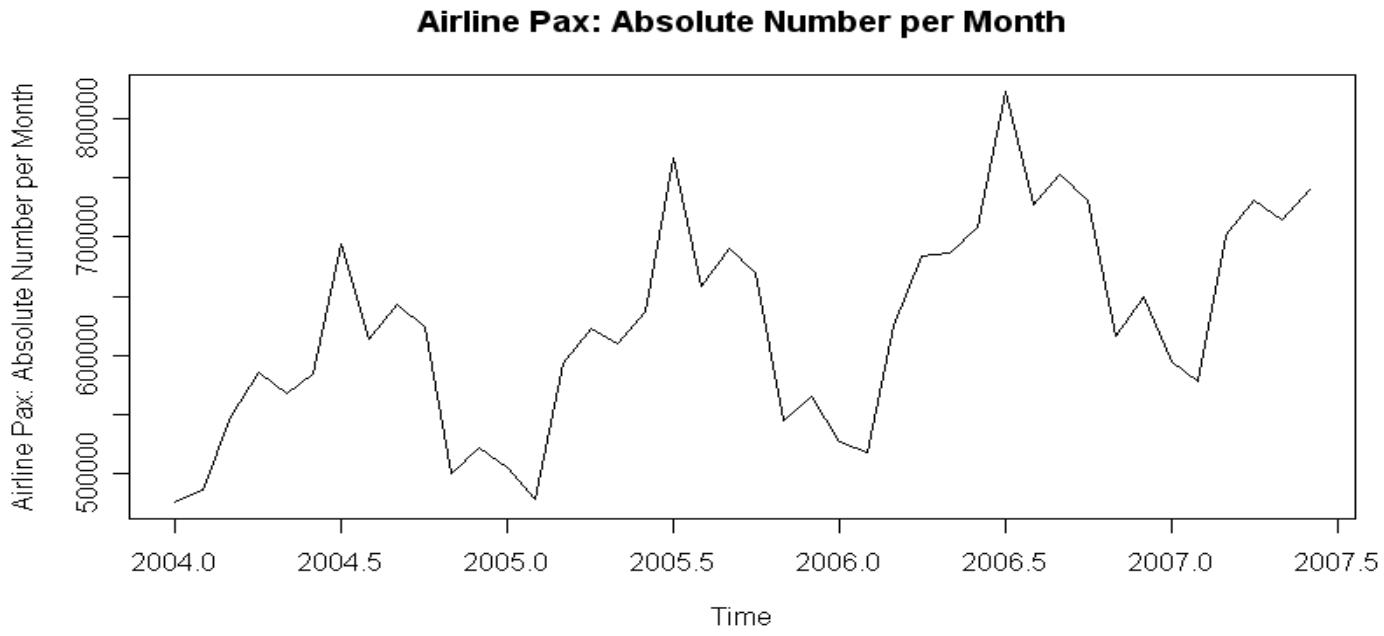
- 20 Boeing 787
- 20 Airbus A350
- 9 Airbus A380

How was this decision taken?

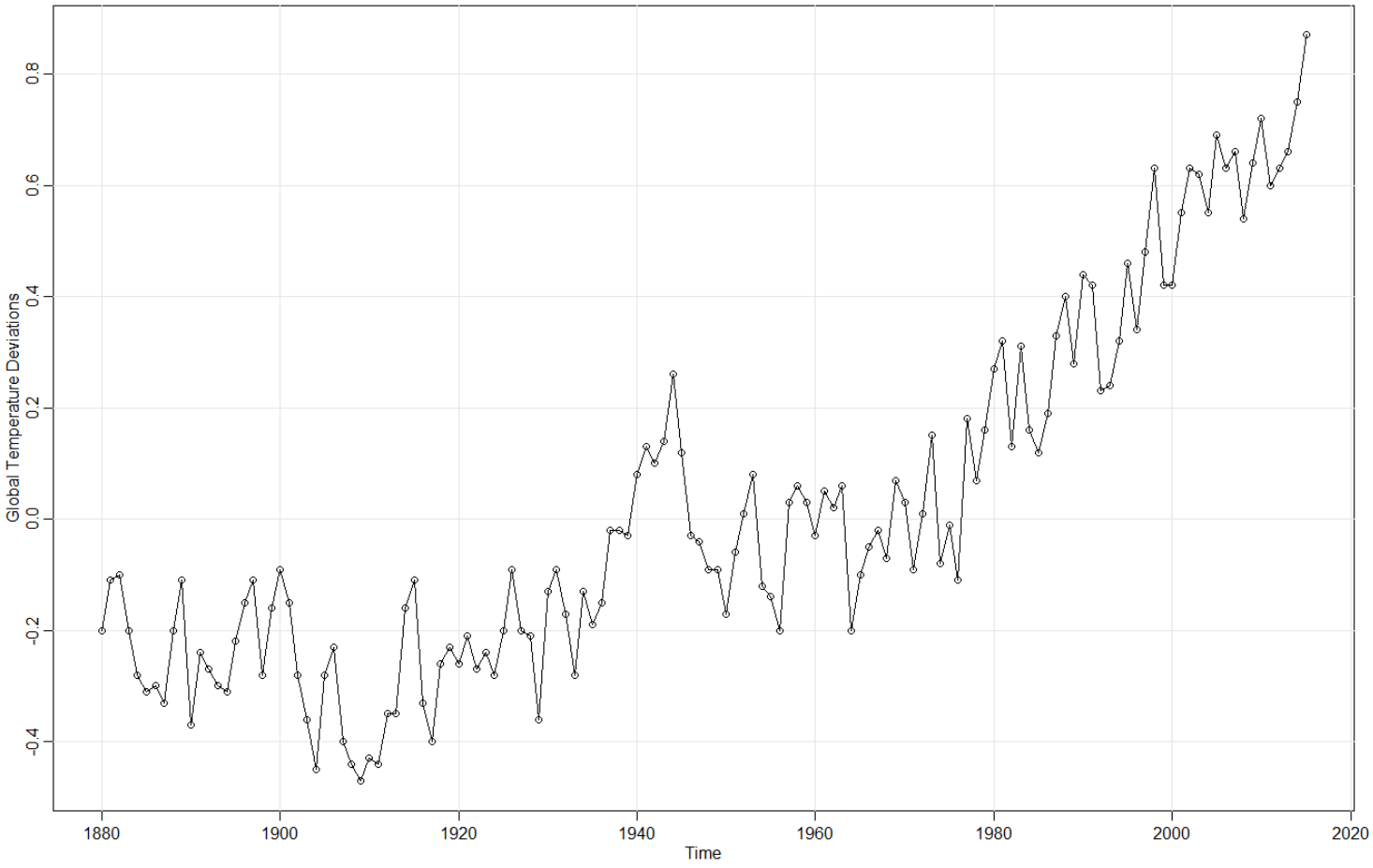
It was based on a combination of time series analysis on airline passenger trends, plus knowing the corporate plans for maintaining or increasing the market share.

Applied Time Series: A Second Example

Airline business: # of checked-in passengers per month



Example: Global Temperature Deviation



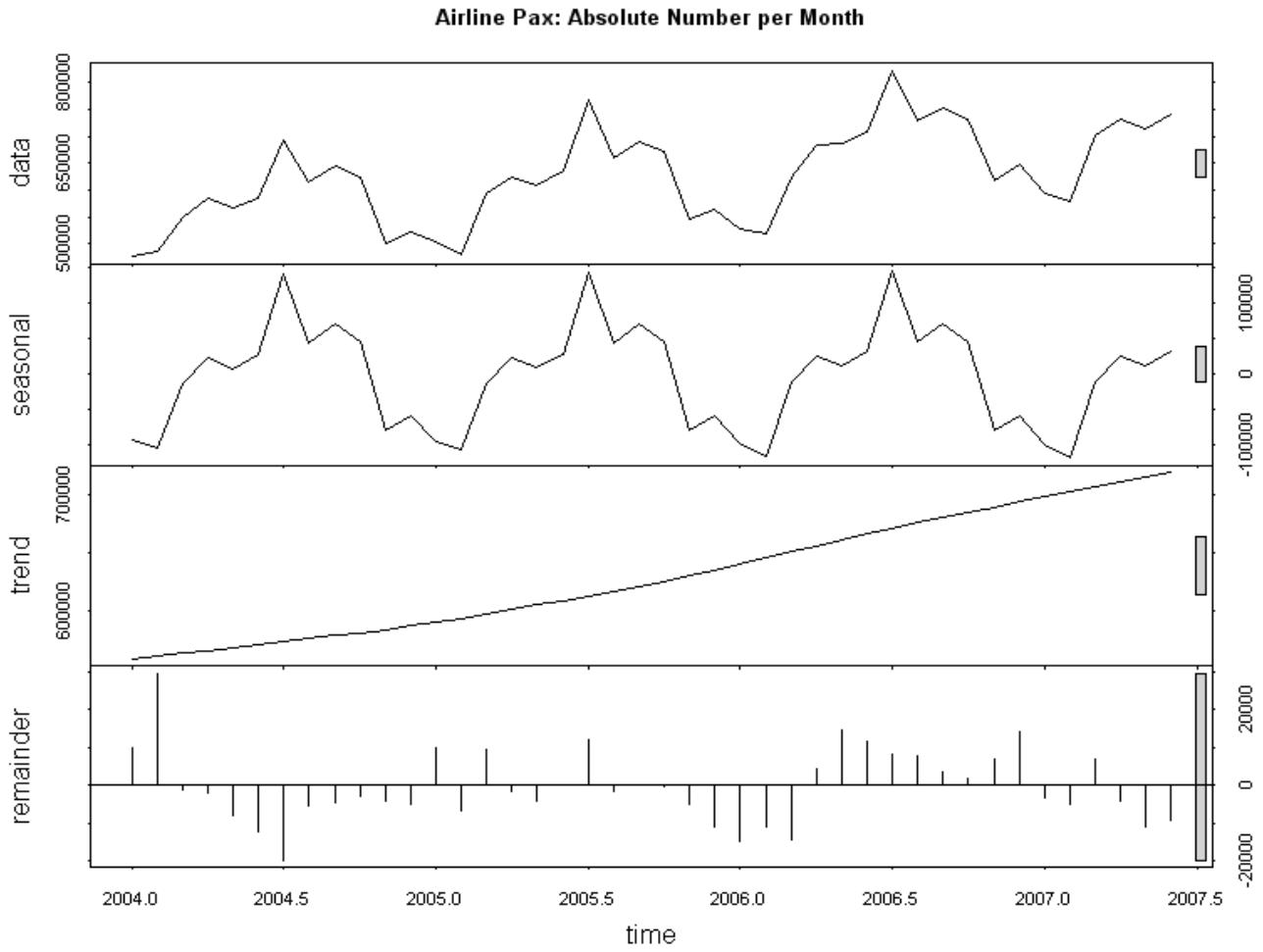
Applied Time Series

Some Properties of the Series

- Increasing trend (i.e. generally more passengers)
- Very prominent seasonal pattern (i.e. peaks/valleys)
- Hard to see details beyond the obvious

Goals of the Project

- Visualize, or better, extract trend and seasonal pattern
- Quantify the amount of random variation/uncertainty
- Provide the basis for a man-made forecast after mid-2007
- Forecast (extrapolation) from mid-2007 until end of 2008
- How can we better organize/collect data?


Airline Pax: Absolute Number per Month
data
seasonal
trend
remainder
500000
650000
800000
-100000
0
100000
600000
700000
-20000
0
20000
2004.0
2004.5
2005.0
2005.5
2006.0
2006.5
2007.0
2007.5
time

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 (duration usually of at least 2 years).

Introduction: What is a Time Series?

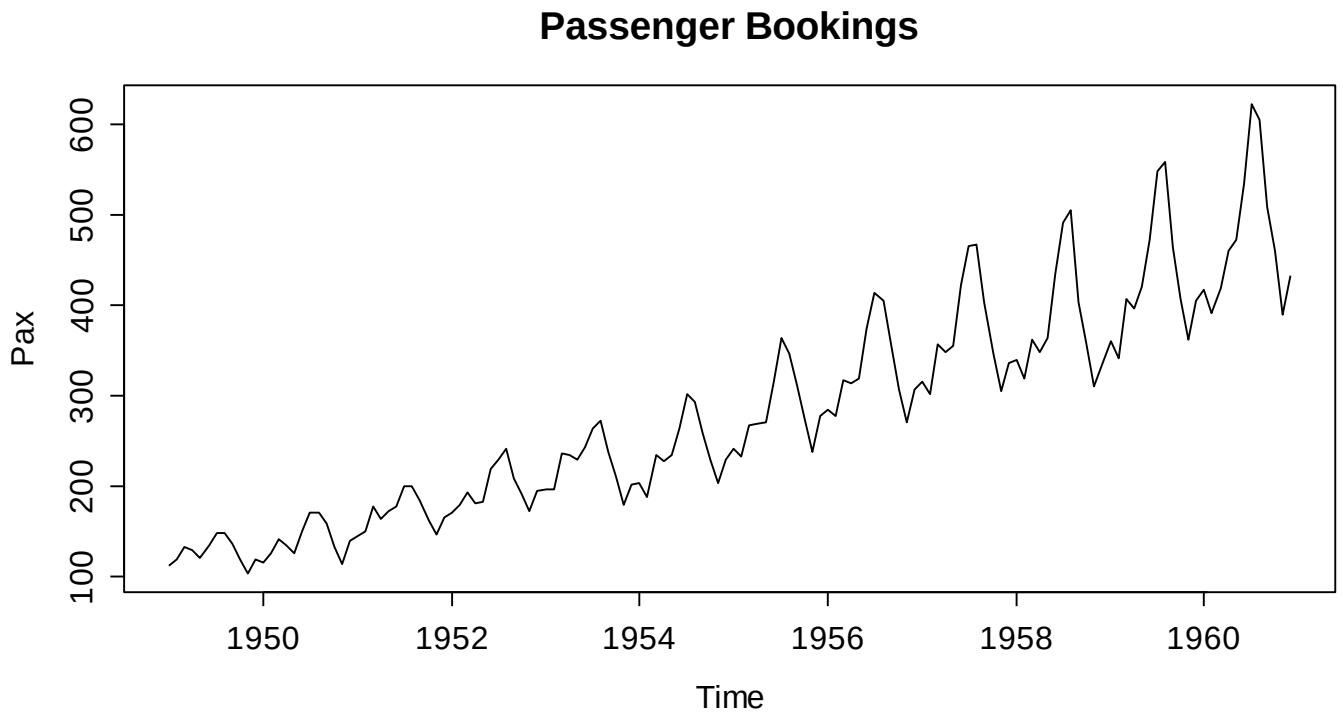
- A time series is a set of observations x_t , where each of the observations was made at a specific time t.
- the set of times T is discrete and finite
- observations were made at fixed time intervals
- continuous and irregularly spaced time series are not covered
- Rationale behind time series analysis:
- The rationale in time series analysis is to understand the past of a series, and to be able to predict the future well.

Example 1: Air Passenger Bookings

```
> data(AirPassengers)

> AirPassengers
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1949 112 118 132 129 121 135 148 148 136 119 104 118
1950 115 126 141 135 125 149 170 170 158 133 114 140
1951 145 150 178 163 172 178 199 199 184 162 146 166
1952 171 180 193 181 183 218 230 242 209 191 172 194
1953 196 196 236 235 229 243 264 272 237 211 180 201
1954 204 188 235 227 234 264 302 293 259 229 203 229
1955 242 233 267 269 270 315 364 347 312 274 237 278
1956 284 277 317 313 318 374 413 405 355 306 271 306
1957 315 301 356 348 355 422 465 467 404 347 305 336
1958 340 318 362 348 363 435 491 505 404 359 310 337
1959 360 342 406 396 420 472 548 559 463 407 362 405
1960 417 391 419 461 472 535 622 606 508 461 390 432
```

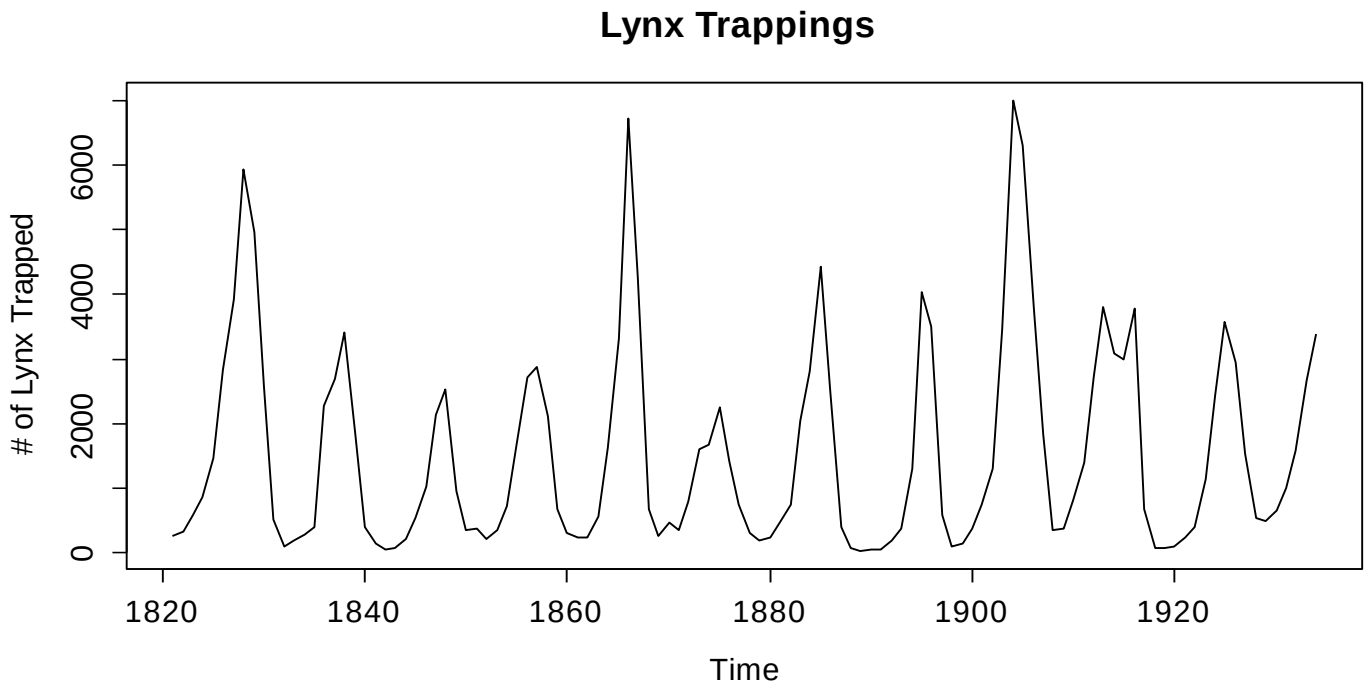
Example 1: Air Passenger Bookings

```
> plot(AirPassengers, ylab="Pax", main="Pax Bookings")
```



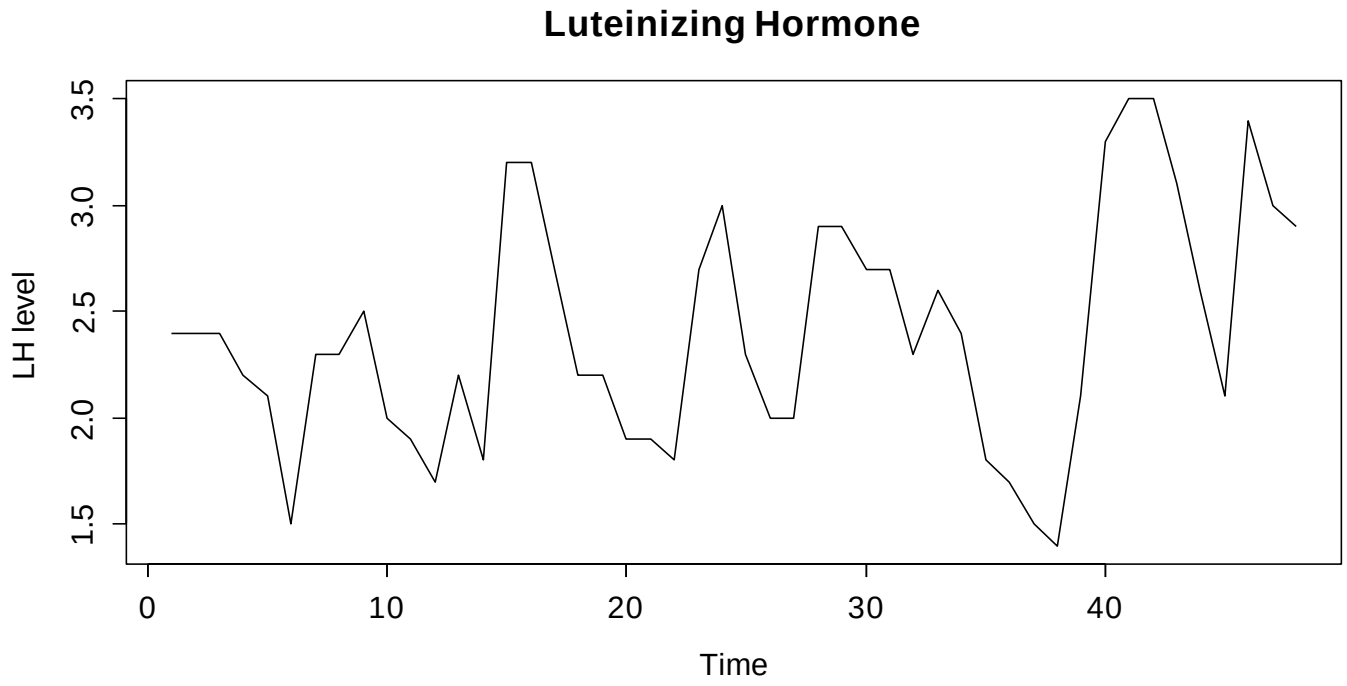
Example 2: Lynx Trappings

```
> data(lynx)
> plot(lynx, ylab="# of Lynx", main="Lynx Trappings")
```



Example 3: Luteinizing Hormone

```
> data(lh)
> plot(lh, ylab="LH level", main="Luteinizing Hormone")
```



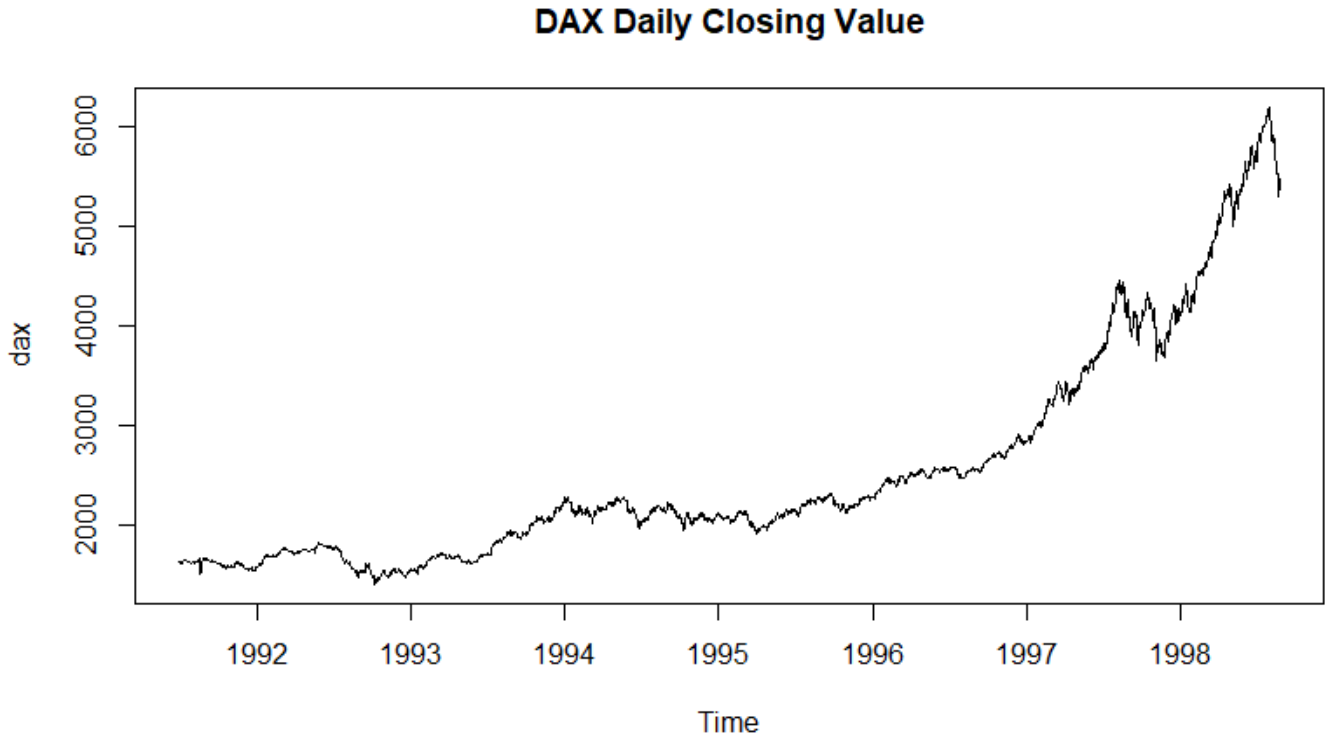
Example 4: DAX

We have a multiple time series object:

```
> data(EuStockMarkets)

> EuStockMarkets
Time Series:
Start = c(1991, 130)
End = c(1998, 169)

Frequency = 260

              DAX    SMI    CAC   FTSE
1991.496 1628.75 1678.1 1772.8 2443.6
1991.500 1613.63 1688.5 1750.5 2460.2
1991.504 1606.51 1678.6 1718.0 2448.2
1991.508 1621.04 1684.1 1708.1 2470.4
1991.512 1618.16 1686.6 1723.1 2484.7
1991.515 1610.61 1671.6 1714.3 2466.8
```

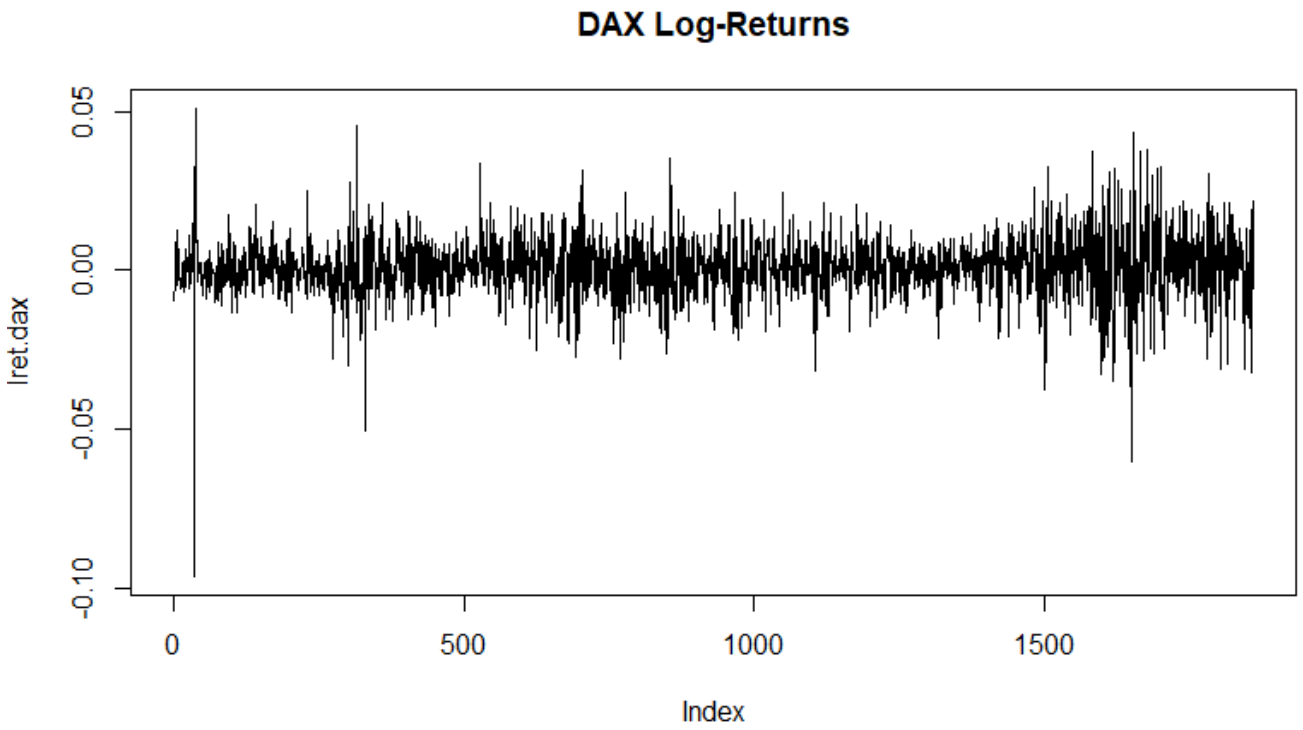
Example 4: DAX

```
> esm <- EuStockMarkets
> tmp <- EuStockMarkets[,1]
> dax <- ts(tmp, start=start(esm), freq=frequency(esm))
> plot(dax, main="DAX Daily Closing Value")
```



Example 4: DAX

```
> lret.dax <- log(dax[2:1860]/dax[1:1859])
> plot(lret.dax, main="DAX Log-Returns", tyep="l")
```



Which is easiest to forecast?

1. daily electricity demand in 3 days time
2. timing of next Halley's comet appearance
3. time of sunrise this day next year
4. Google stock price tomorrow
5. Google stock price in 6 months time
6. maximum temperature tomorrow
7. exchange rate of $US/AUS next week
8. total sales of drugs in Australian pharmacies next month

How do we measure "easiest"?

What makes something easy/difficult to forecast?

Goals in Time Series Analysis

Exploratory Analysis

Visualization of the properties of the series

- time series plot
- decomposition into trend/seasonal pattern/random error
- correlogram for understanding the dependency structure

Modeling

Fitting a stochastic model to the data that represents and reflects the most important properties of the series

- done exploratory or with previous knowledge
- model choice and parameter estimation is crucial
- inference: how well does the model fit the data?

Factors affecting forecastability

Something is easier to forecast if:

- we have a good understanding of the factors that contribute to it
- there is lots of data available;
- the forecasts cannot affect the thing we are trying to forecast.
- there is relatively low natural/unexplainable random variation.
- the future is somewhat similar to the past

Goals in Time Series Analysis

Forecasting

Prediction of future observations with measure of uncertainty

- mostly model based, uses dependency and past data
- is an extrapolation, thus often to take with a grain of salt
- similar to driving a car by looking in the rear window mirror

Process Control

The output of a (physical) process defines a time series

- a stochastic model is fitted to observed data
- this allows understanding both signal and noise
- it is feasible to monitor normal/abnormal fluctuations

Goals in Time Series Analysis

Time Series Regression

Modeling response time series using 1 or more input series

$$Y_t = \beta_0 + \beta_1 u_t + \beta_2 v_t + E_t$$

where E_t is independent of u_t and v_t, but not i.i.d.

Example: $(\text{Ozone})_t = (\text{Wind})_t + (\text{Temperature})_t + E_t$

- Fitting this model under i.i.d error assumption:
- leads to unbiased estimates, but...
- often grossly wrong standard errors
- thus, confidence intervals and tests are misleading

Outline

- Introduction
- Time Series in R
- Time plots
- Seasonal plots
- Seasonal or cyclic?
- Lag plots and autocorrelation
- White noise

Time Series in R

- In R, there are objects, which are organized in a large number of classes. These classes e.g. include vectors, data frames, model output, functions, and many more. Not surprisingly, there are also several classes for time series.
- We focus on ts, the basic class for regularly spaced time series in R. This class is comparably simple, as it can only represent time series with fixed interval records, and only uses numeric time stamps, i.e. enumerates the index set.
- For defining a ts object, we have to supply the data, but also the starting time (as argument start), and the frequency of measurements as argument frequency.

Time Series in R: Example

Data:

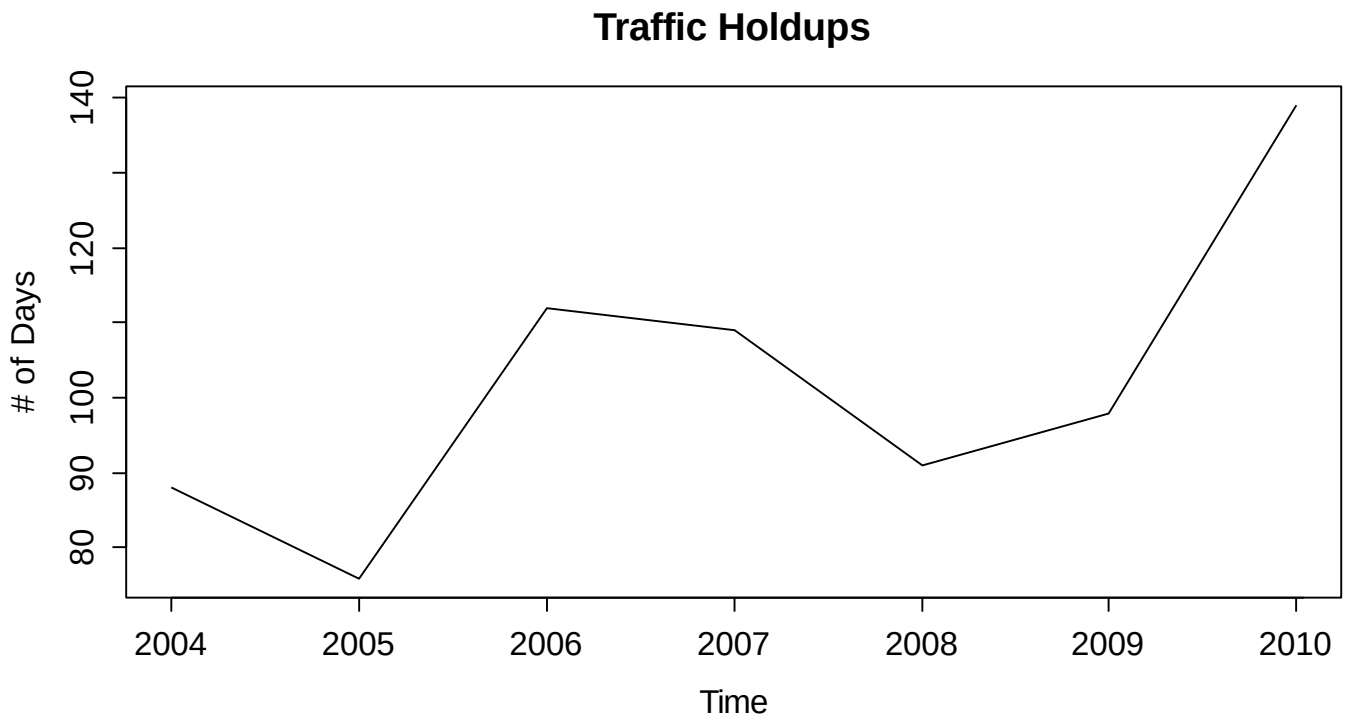
number of days per year with traffic holdups in front of the Gotthard road tunnel north entrance in Switzerland.

2004	2005	2006	2007	2008	2009	2010
88	76	112	109	91	98	139

```
> rawdat <- c(88, 76, 112, 109, 91, 98, 139)
> ts.dat <- ts(rawdat, start=2004, freq=1)

> ts.dat
Time Series: Start = 2004
End = 2010; Frequency = 1
[1]  88  76 112 109  91  98 139
```

Time Series in R: Example

```
> plot(ts.dat, ylab="# of Days", main="Traffic Holdups")
```



Time Series in R

`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(data, frequency, start)`

Typoe of Data	Frequency	Start
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

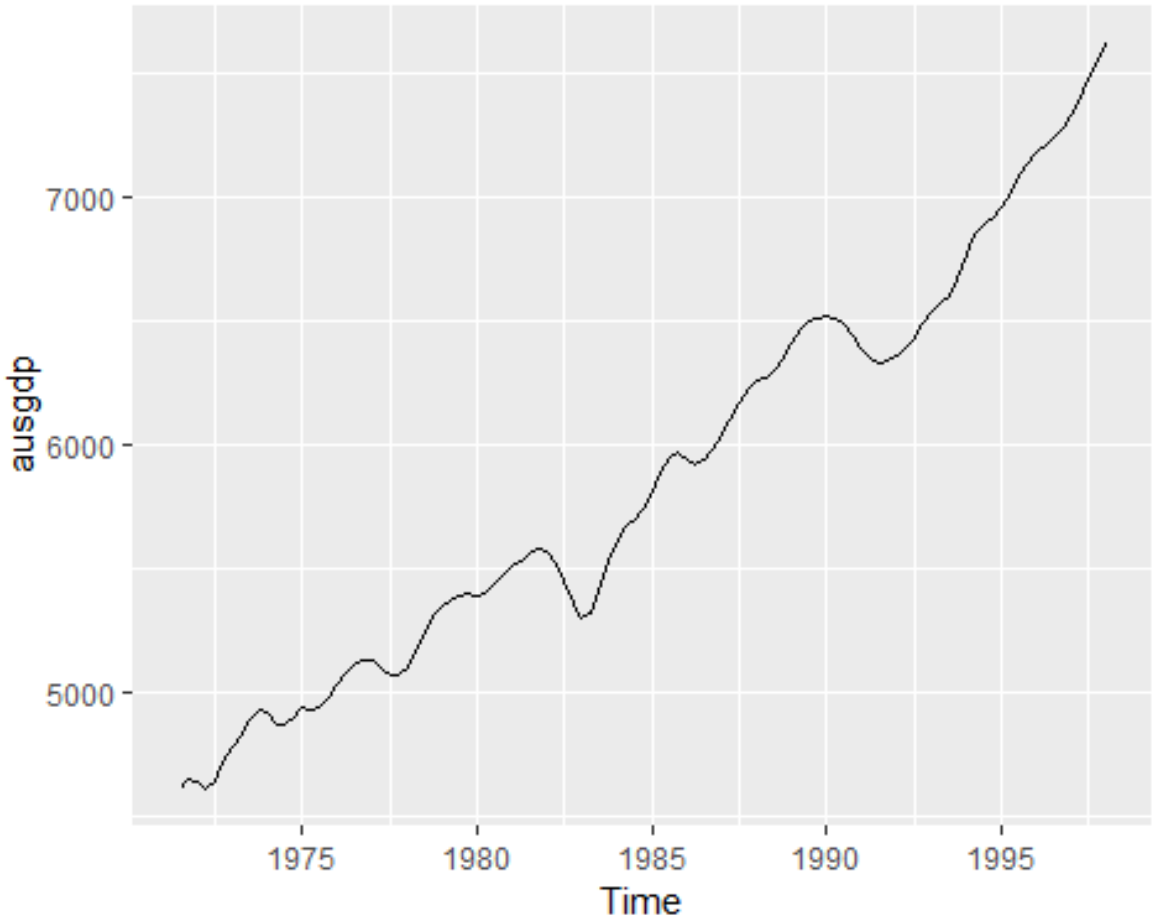
```
ausgdp <- ts(x, frequency=4, start=c(1971,3))
```

Class: “ts”

Print and plotting methods available.

```
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
## 1977 5130 5101 5072 5069
## 1978 5100 5166 5244 5312
## 1979 5349 5370 5388 5396
## 1980 5388 5403 5442 5482
## 1981 5506 5531 5560 5583
## 1982 5568 5524 5452 5358
## 1983 5303 5320 5408 5531
## 1984 5624 5669 5697 5736
## 1985 5811 5894 5952 5965
## 1986 5943 5924 5935 5979
## 1987 6035 6097 6167 6227
## 1988 6256 6272 6295 6345
## 1989 6413 6468 6497 6511
## 1990 6514 6512 6490 6442
## 1991 6390 6346 6328 6340
## 1992 6362 6389 6433 6491
## 1993 6541 6566 6602 6671
…
```

autoplot(ausgdp)



Residential electricity sales

```
elecsales
## Time Series:
## Start = 1989
## End = 2008
## Frequency = 1
##  [1] 2354.34 2379.71 2318.52 2468.99 2386.09
##  [6] 2569.47 2575.72 2762.72 2844.50 3000.70
## [11] 3108.10 3357.50 3075.70 3180.60 3221.60
## [16] 3176.20 3430.60 3527.48 3637.89 3655.00
```

Time Series Packages

forecast: Forecasting Functions for Time Series and Linear Models

Methods and tools for displaying and analysing univariate time series forecasts including exponential smoothing via state space models and automatic ARIMA modelling.

Rob J Hyndman, Professor of Statistics, Monash University

https://cran.r-project.org/web/packages/forecast/index.html

Overview of Time Series related packages

https://cran.r-project.org/web/views/TimeSeries.html

Class package

```
> library(fpp2)
```

This loads:

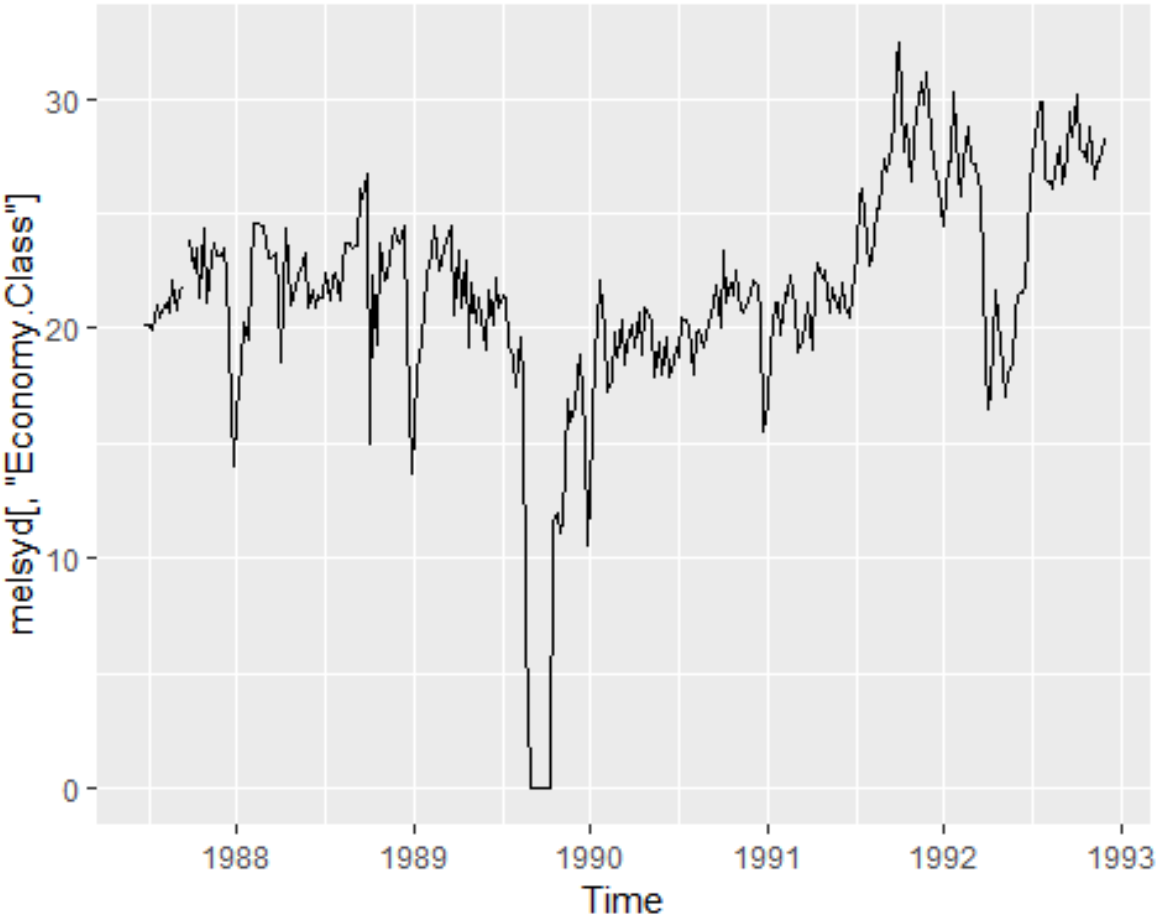
- some data for use in examples and exercises
- **forecast** package (for forecasting functions)
- **ggplot2** package (for graphics functions)
- **fma** package (for lots of time series data)
- **expsmooth** package (for more time series data)

Outline

- Introduction
- Time Series in R
- Time plots
- Seasonal plots
- Seasonal or cyclic?
- Lag plots and autocorrelation
- White noise

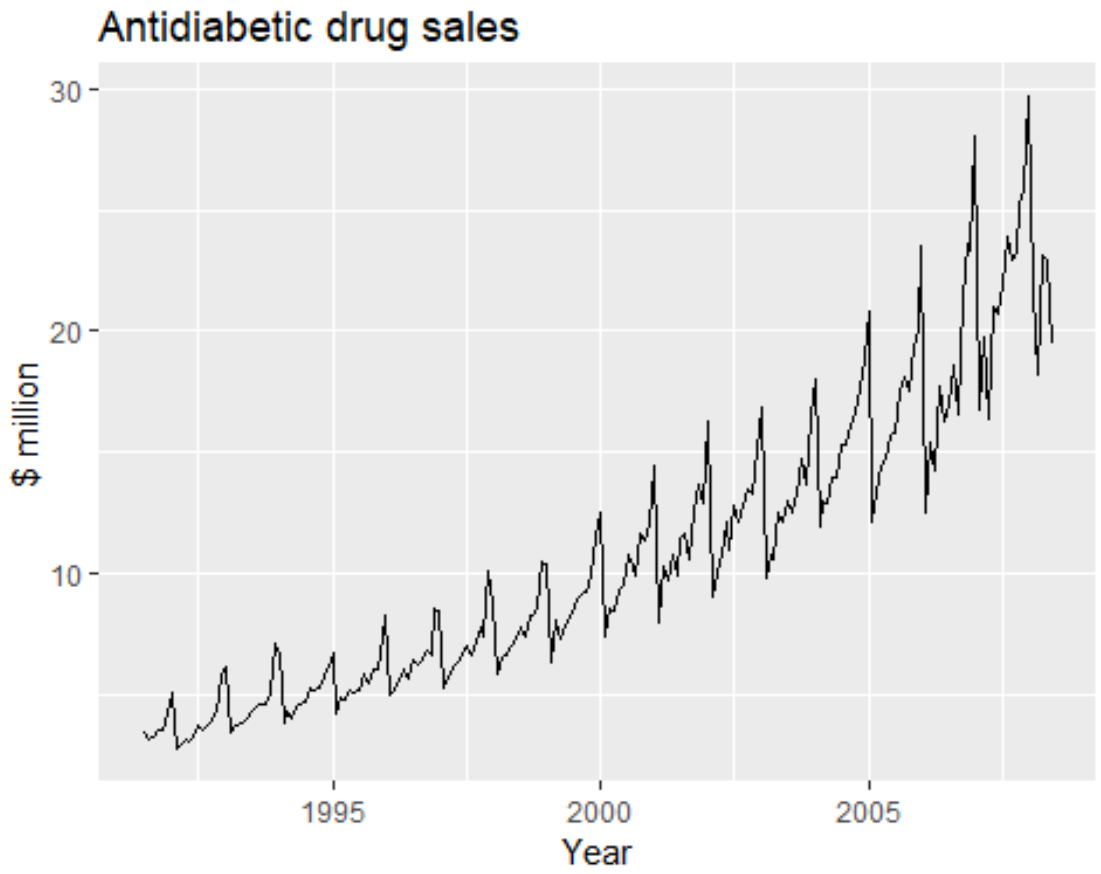
Time plots

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



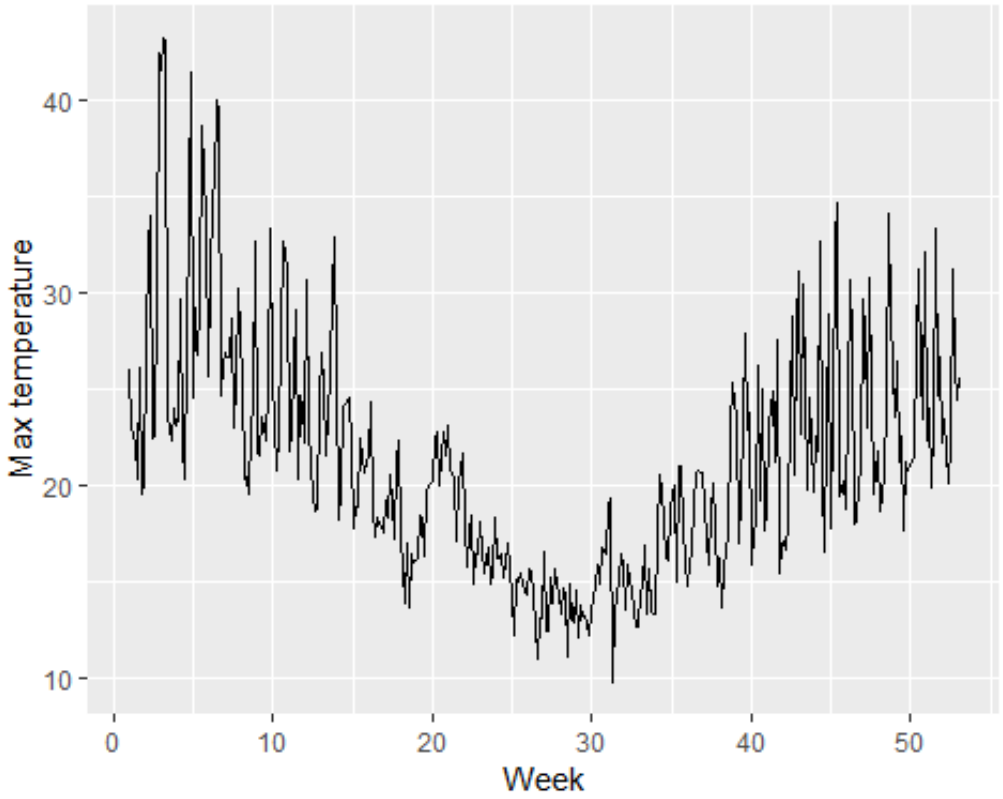
Time plots

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



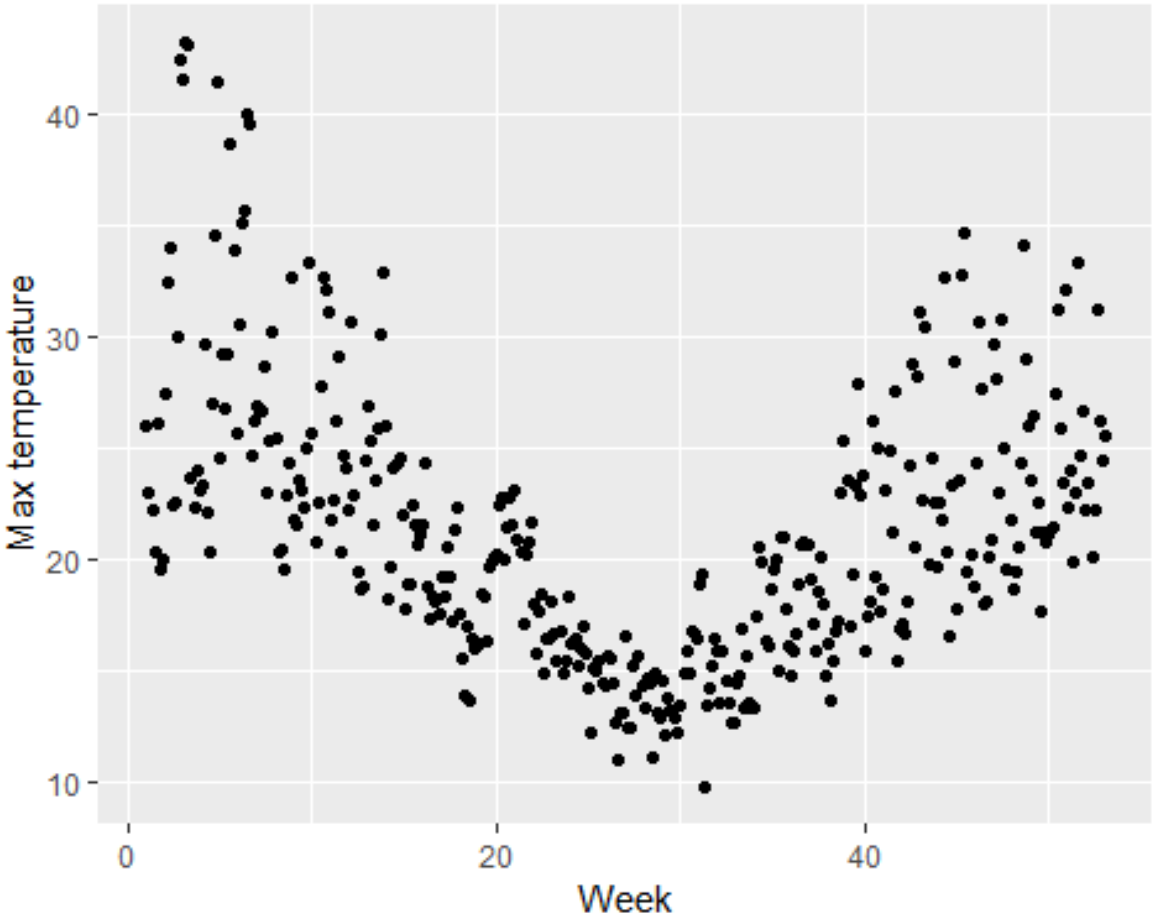
Are time plots best?

```
autoplot(elecdaily[,"Temperature"]) +
  xlab("Week") + ylab("Max temperature")
```

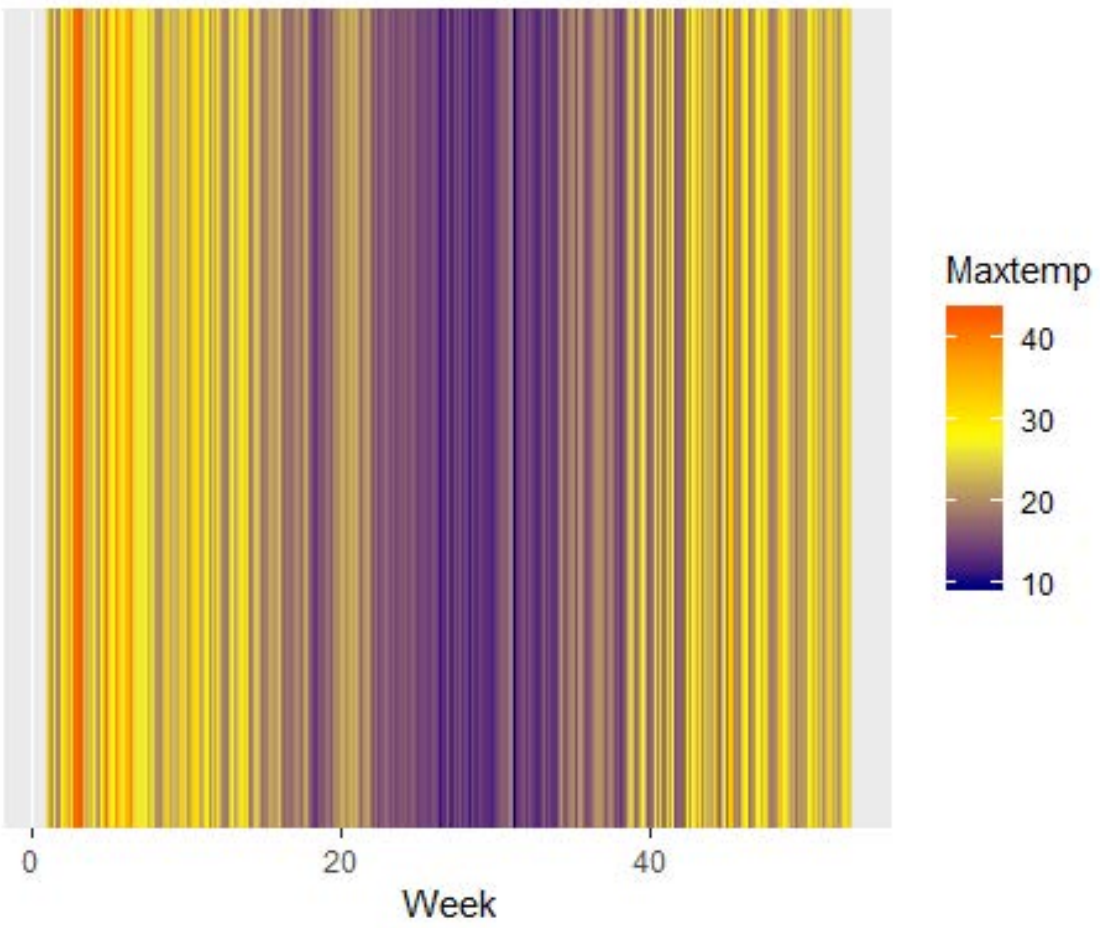


Are time plots best?

```
qplot(time(elecdaily), elecdaily[,"Temperature"]) +
  xlab("Week") + ylab("Max temperature")
```



Are time plots best?



Outline

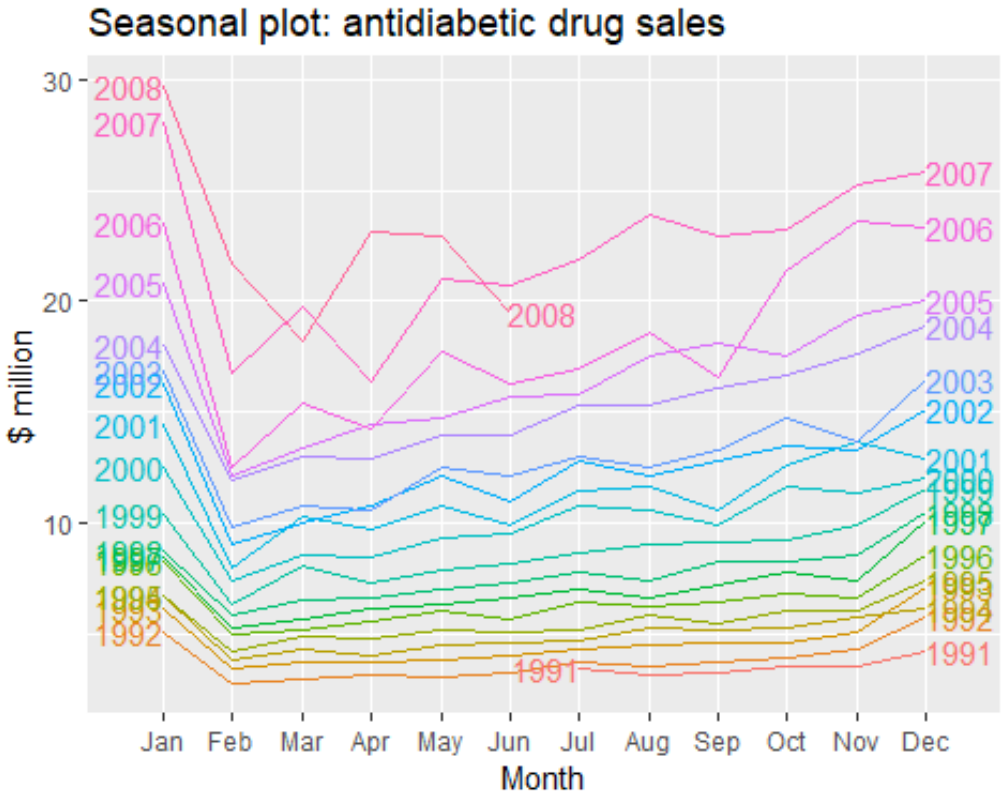
- Introduction
- Time Series in R
- Time plots
- Seasonal plots
- Seasonal or cyclic?
- Lag plots and autocorrelation
- White noise

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()`

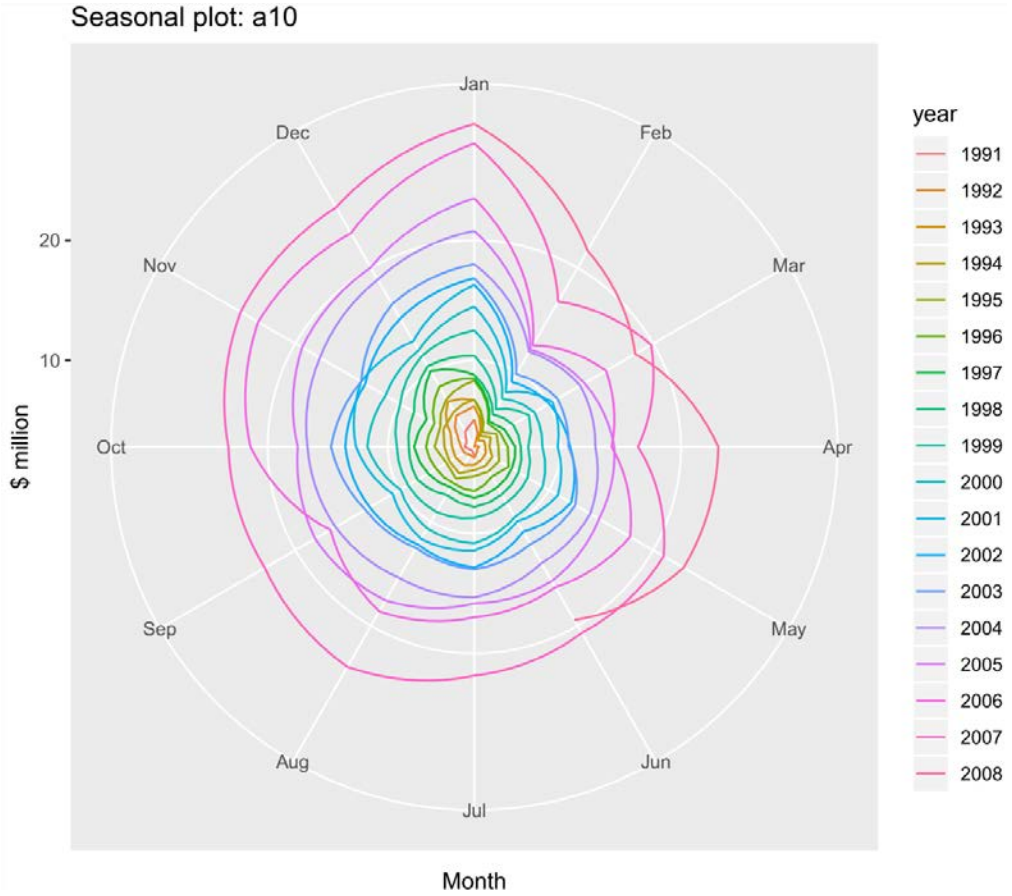
Seasonal plots

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



Seasonal polar plots

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

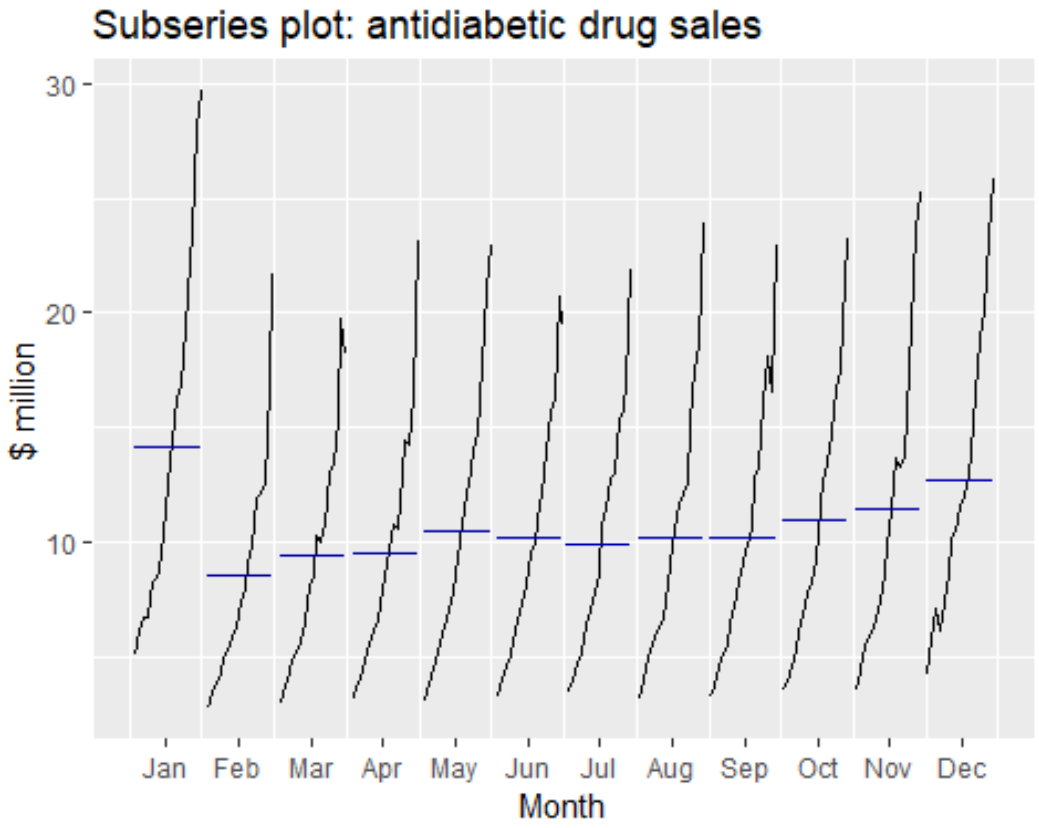


Seasonal subseries plots

- 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()`

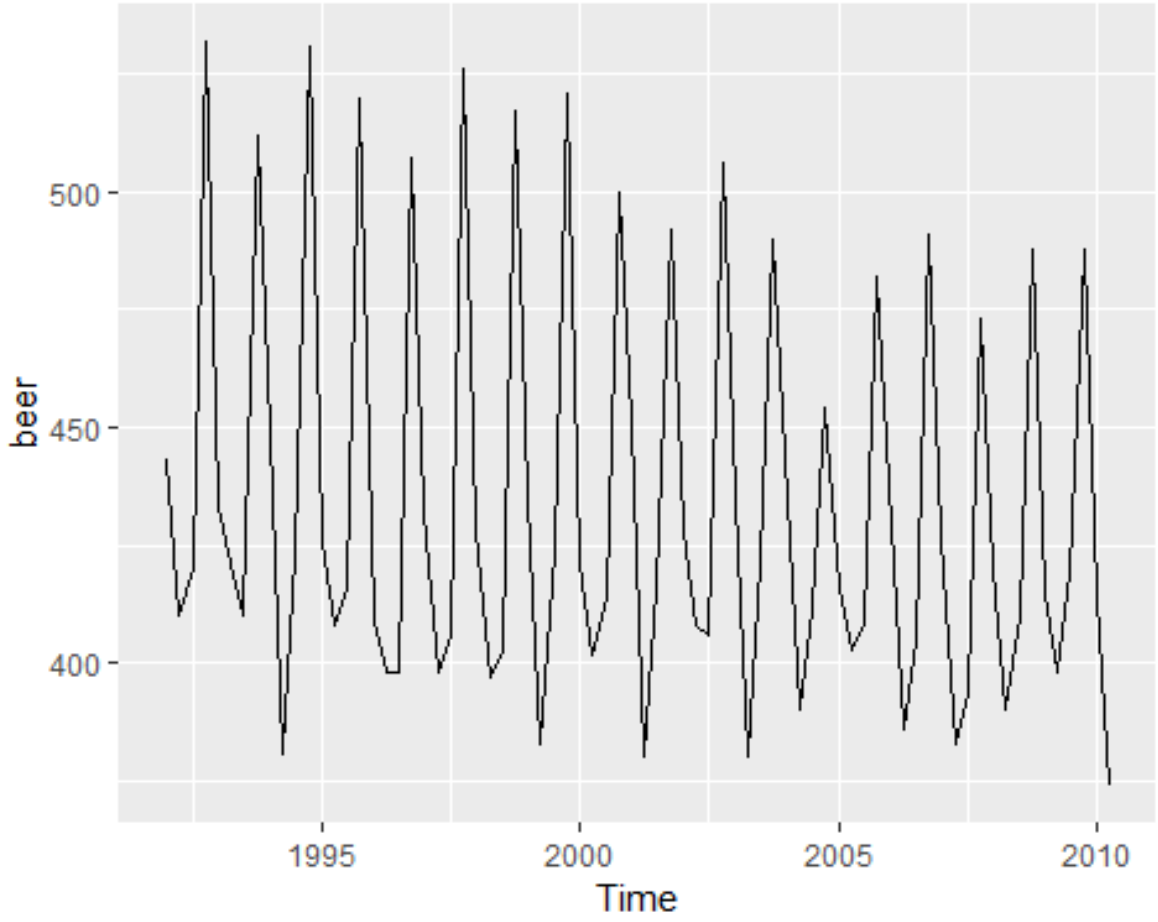
Seasonal subseries plots

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



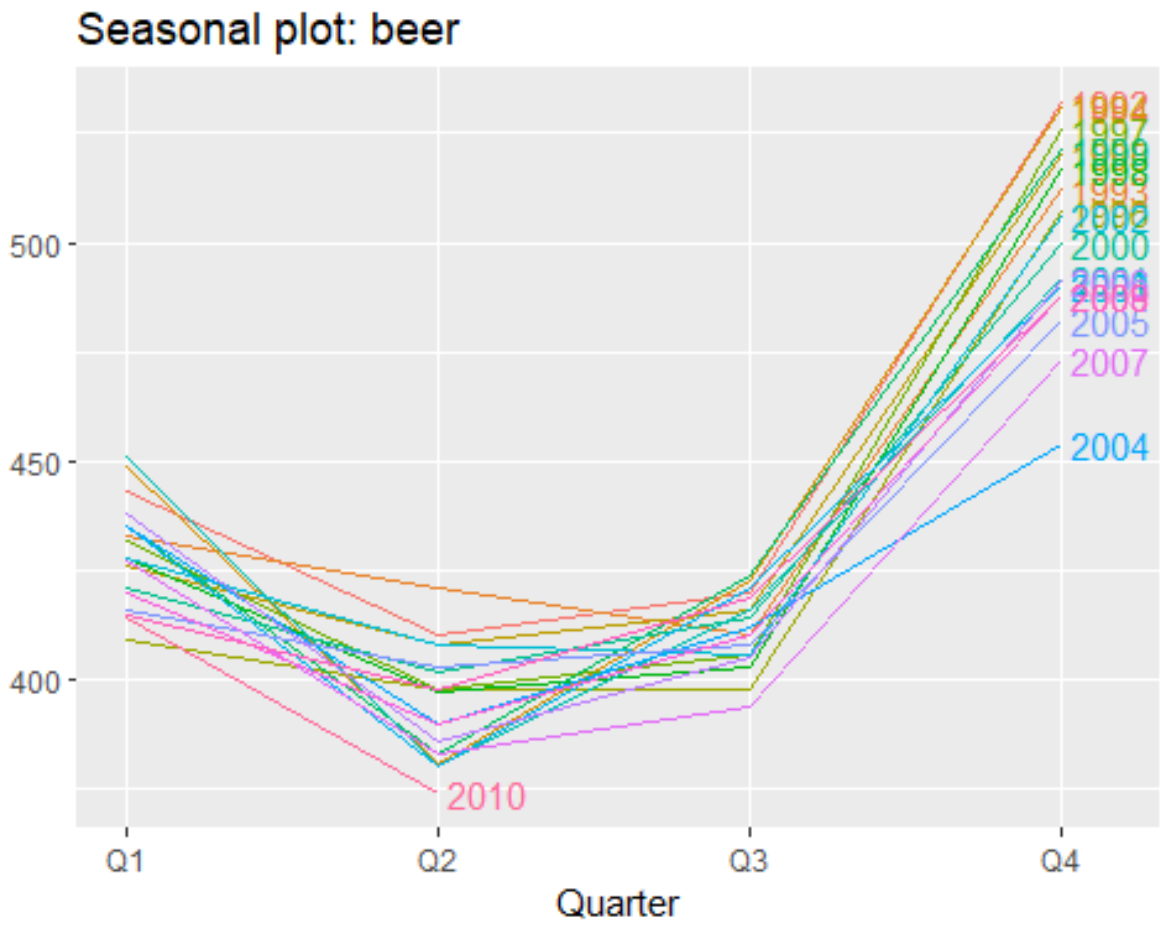
Quarterly Australian Beer Production

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



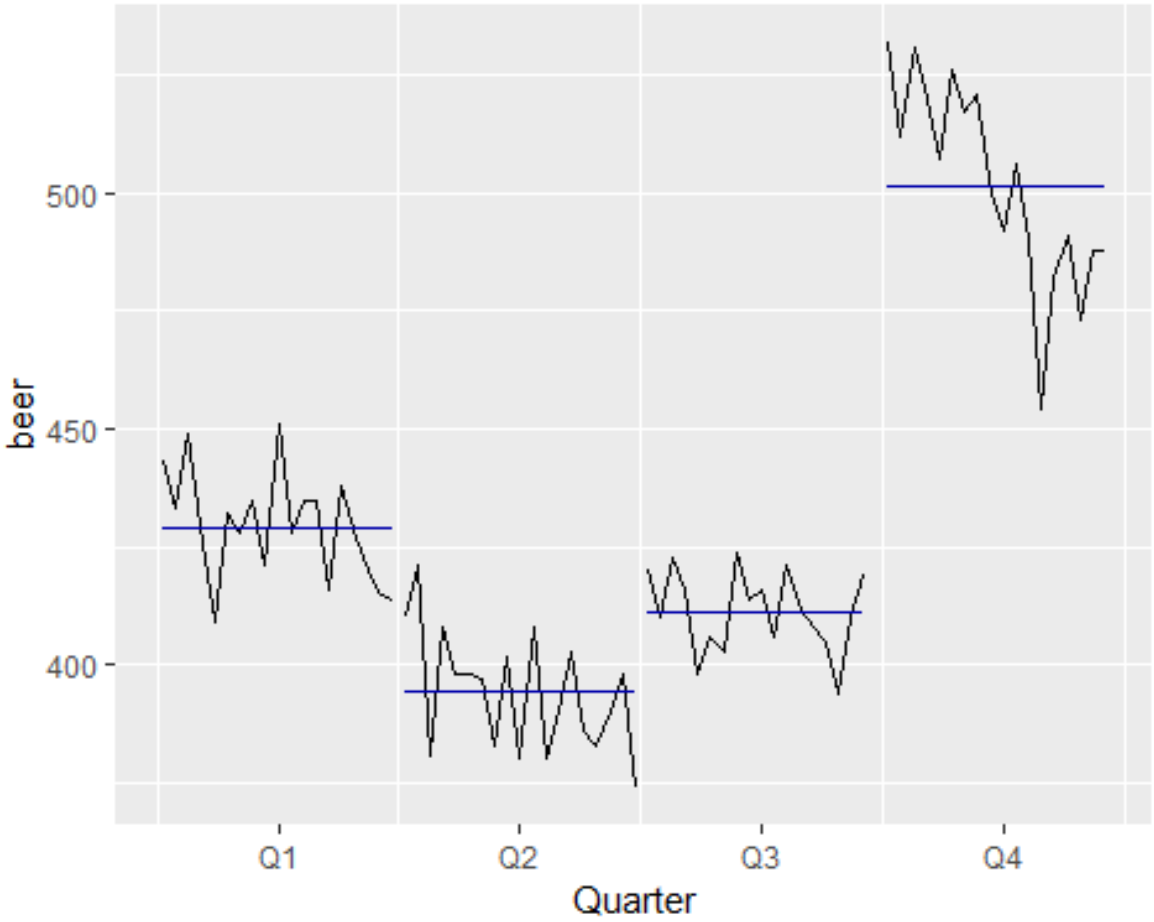
Quarterly Australian Beer Production

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



Quarterly Australian Beer Production

```
ggsubseriesplot(beer)
```



Outline

- Introduction
- Time Series in R
- Time plots
- Seasonal plots
- Seasonal or cyclic?
- Lag plots and autocorrelation
- White noise

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 (duration usually of at least 2 years).

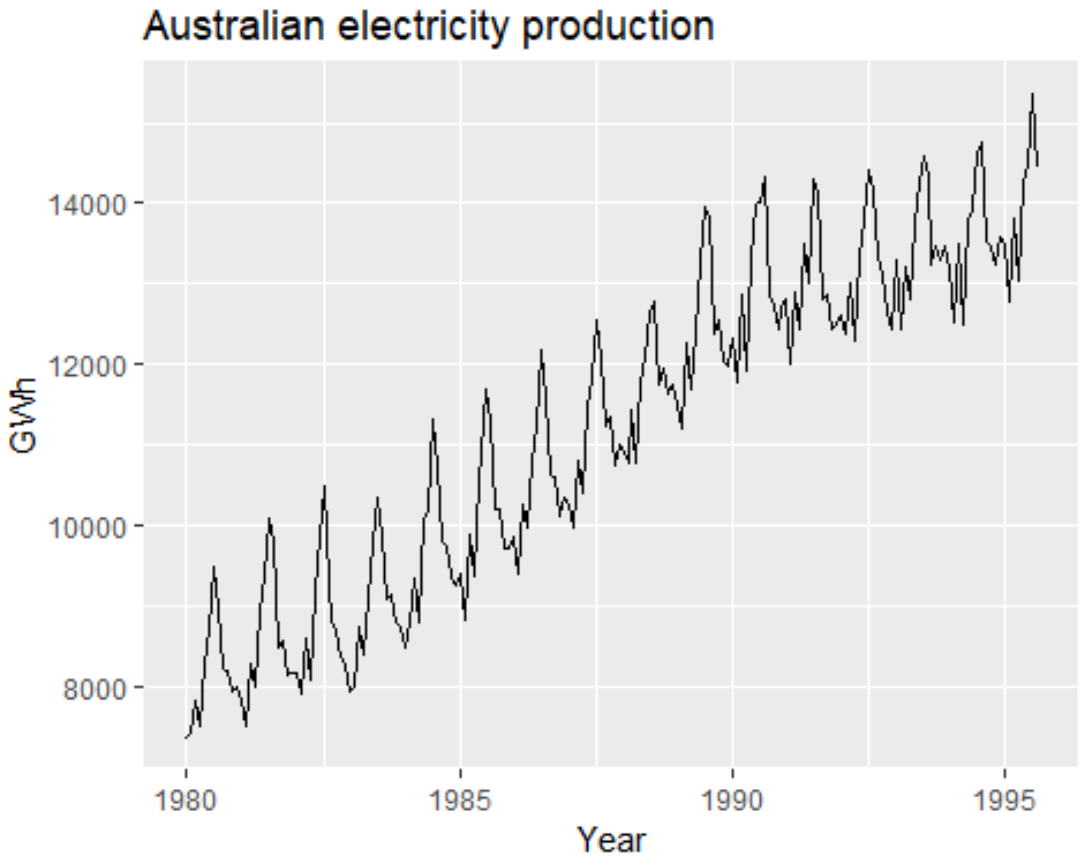
Time Series Components

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

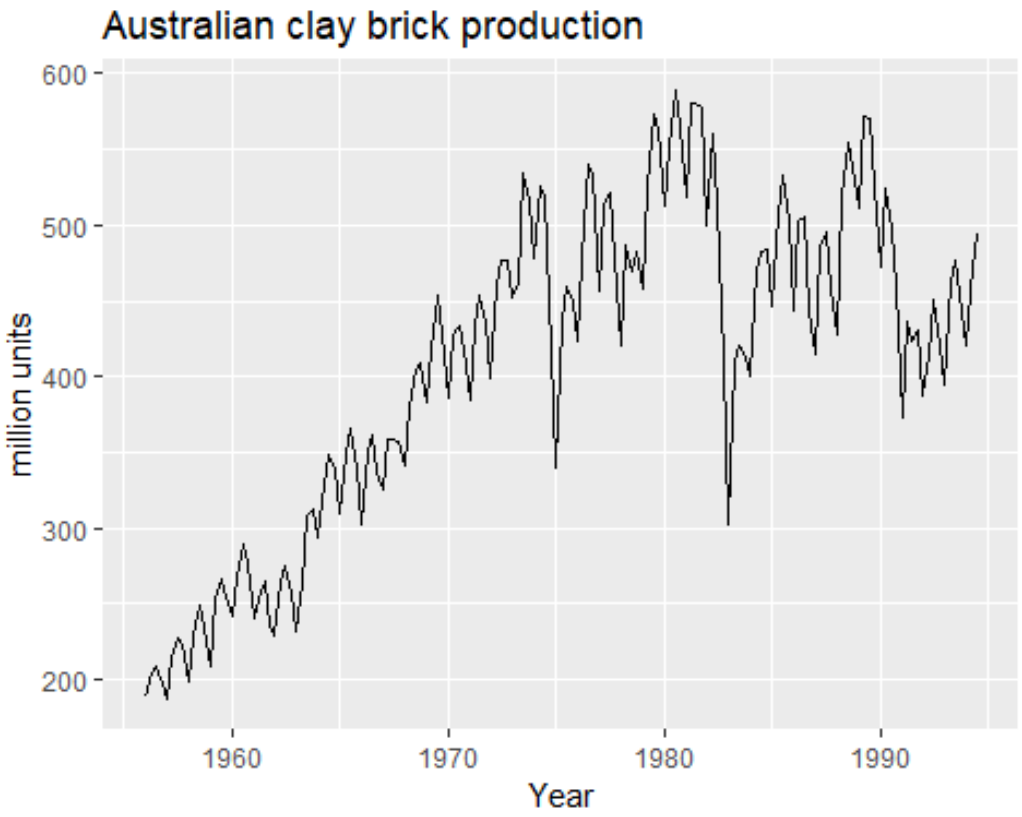
Time Series Patterns

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



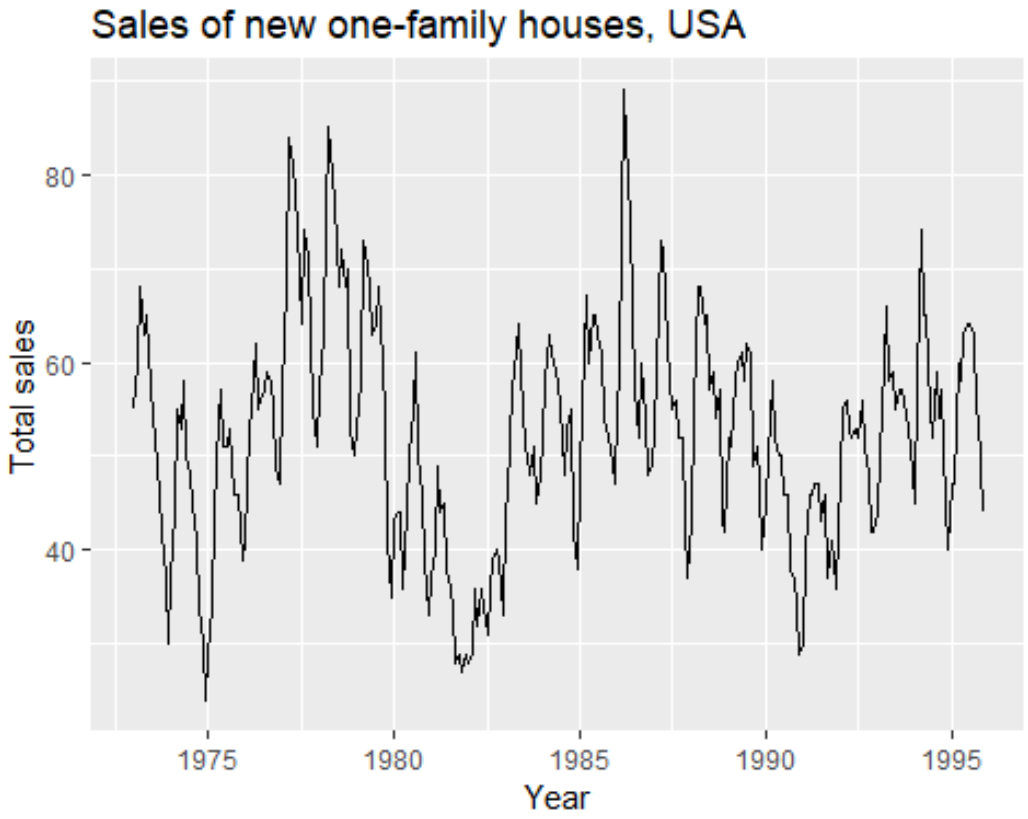
Time Series Patterns

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



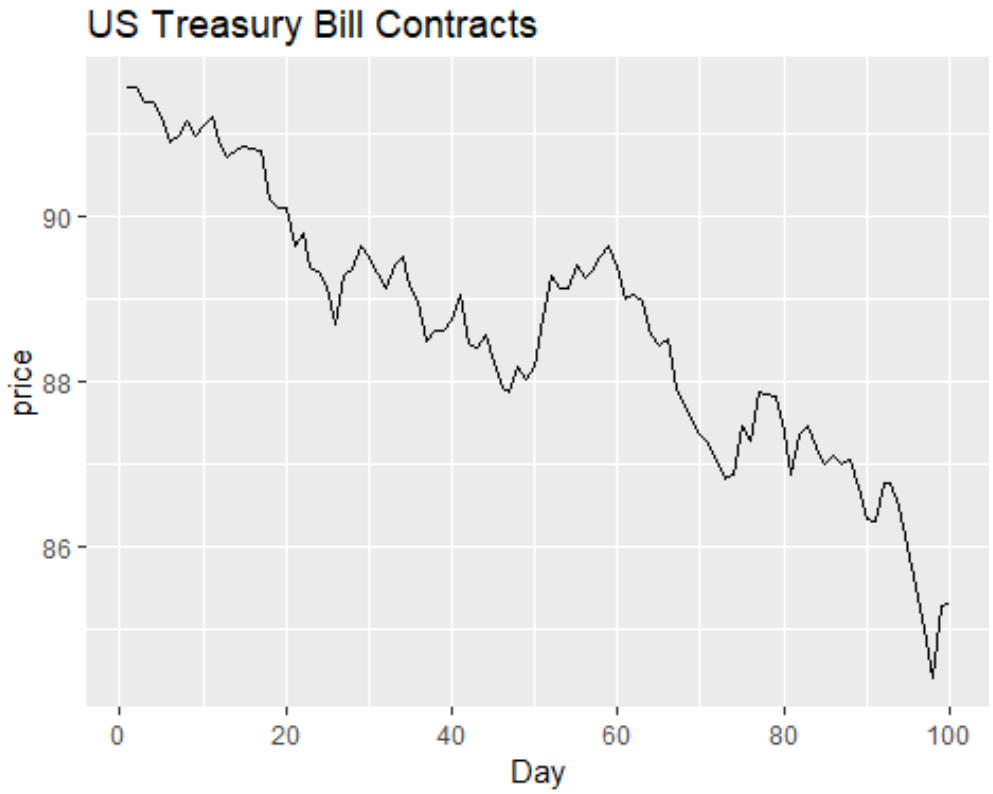
Time Series Patterns

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



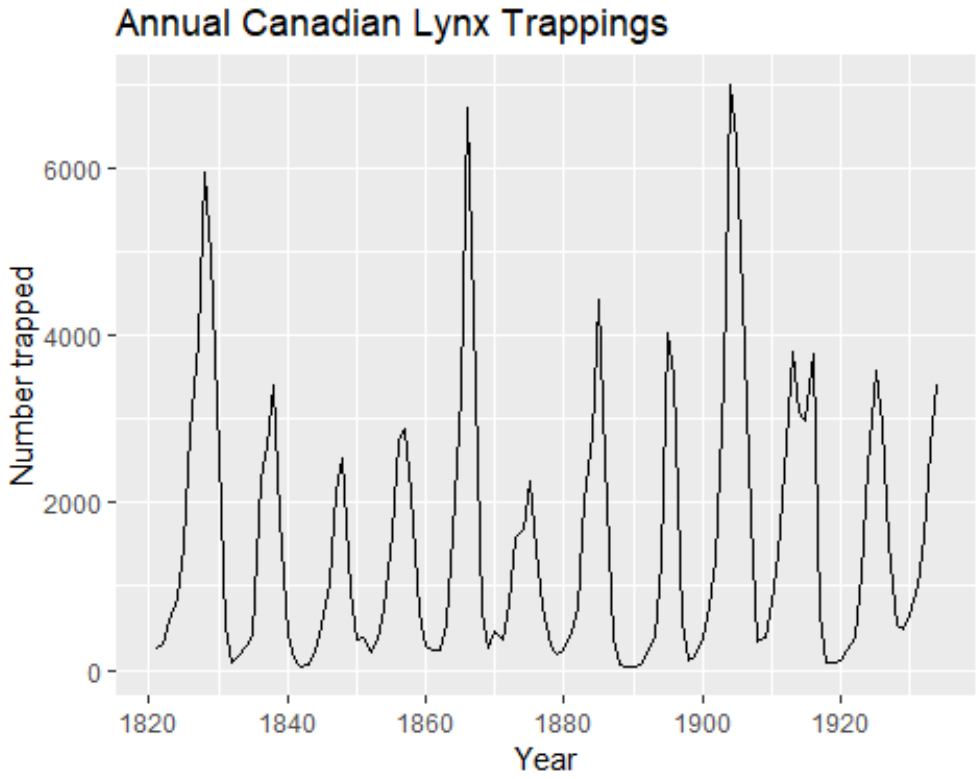
Time Series Patterns

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



Time Series Patterns

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



Seasonal or cyclic?

- 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.

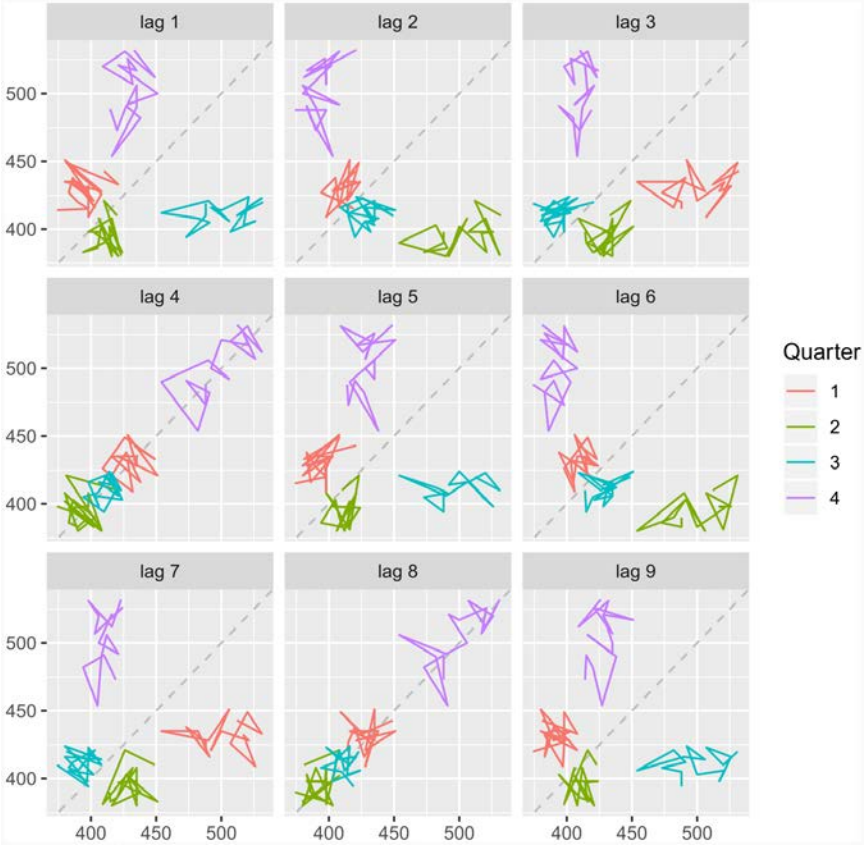
Outline

- Introduction
- Time Series in R
- Time plots
- Seasonal plots
- Seasonal or cyclic?
- Lag plots and autocorrelation
- White noise

Lag plots and autocorrelation

Example: Beer production

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



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.

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

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

$$r_k = \frac{c_k}{c_0}$$

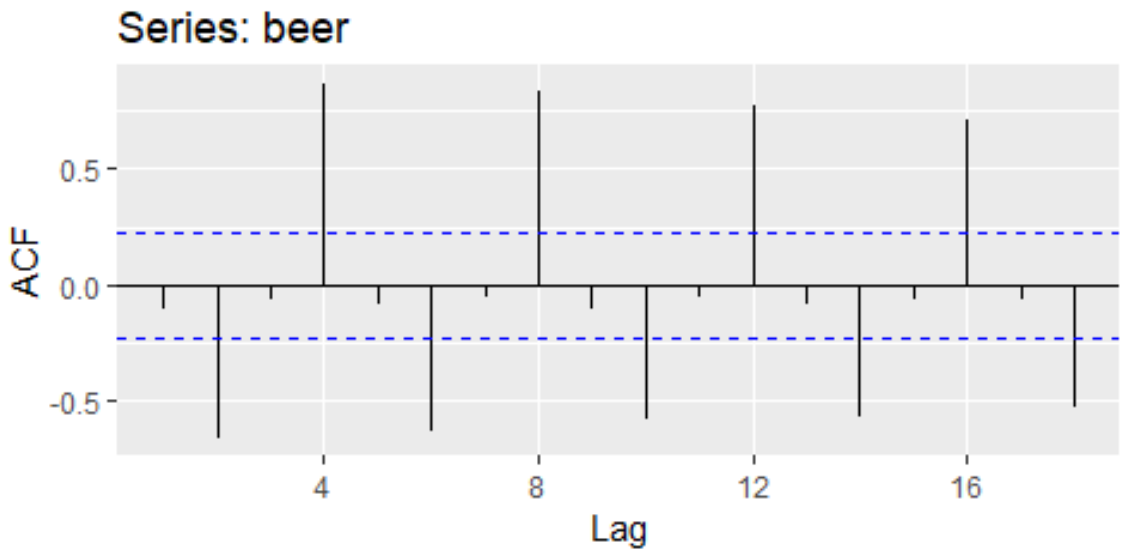
- r_1 indicates how successive values of y relate to each other
- r_2 indicates how y values two periods apart relate to each other
- r_k is the same as the sample correlation between y_t and y_{t-k}.

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

```
ggAcf(beer)
```

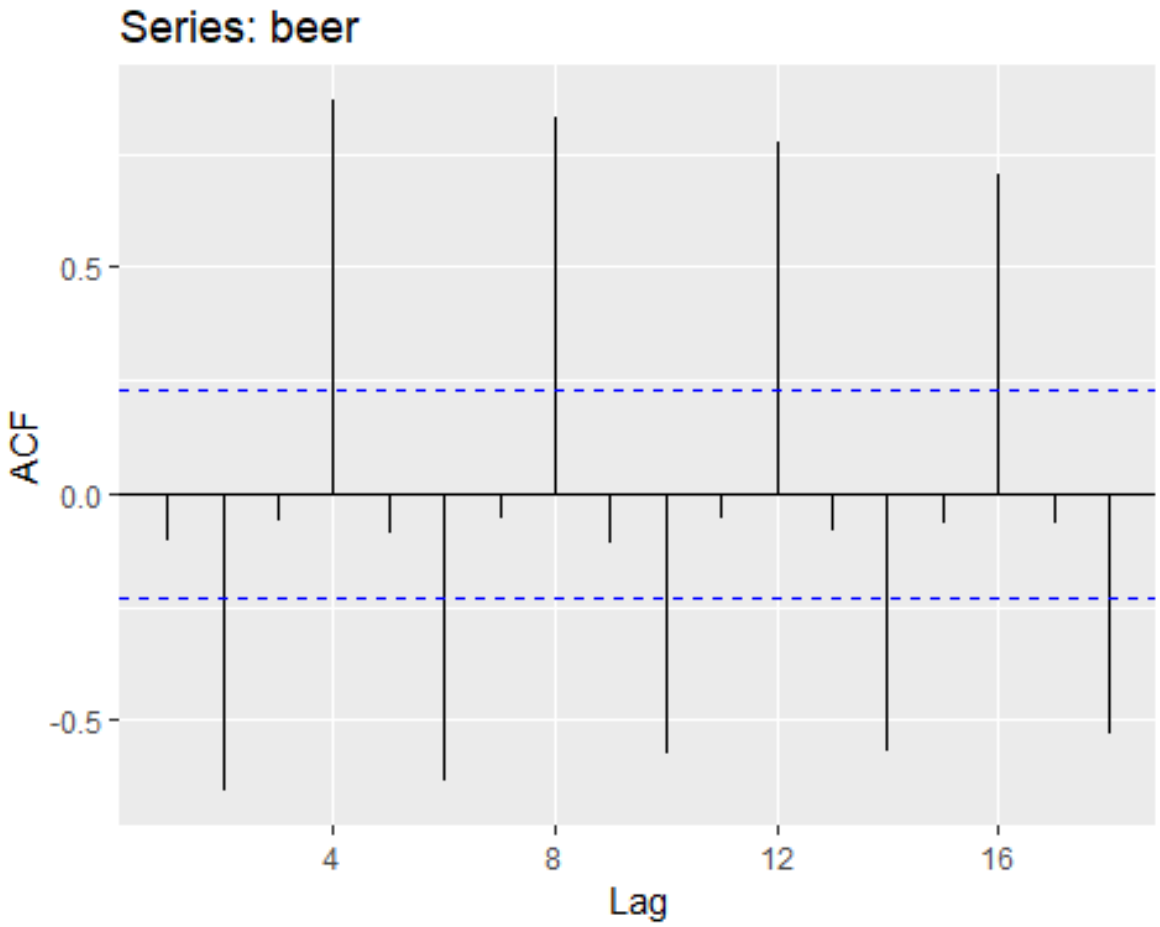


Autocorrelation

- 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 or ACF.
- The plot is known as a **correlogram**

ACF

```
ggAcf(beer)
```

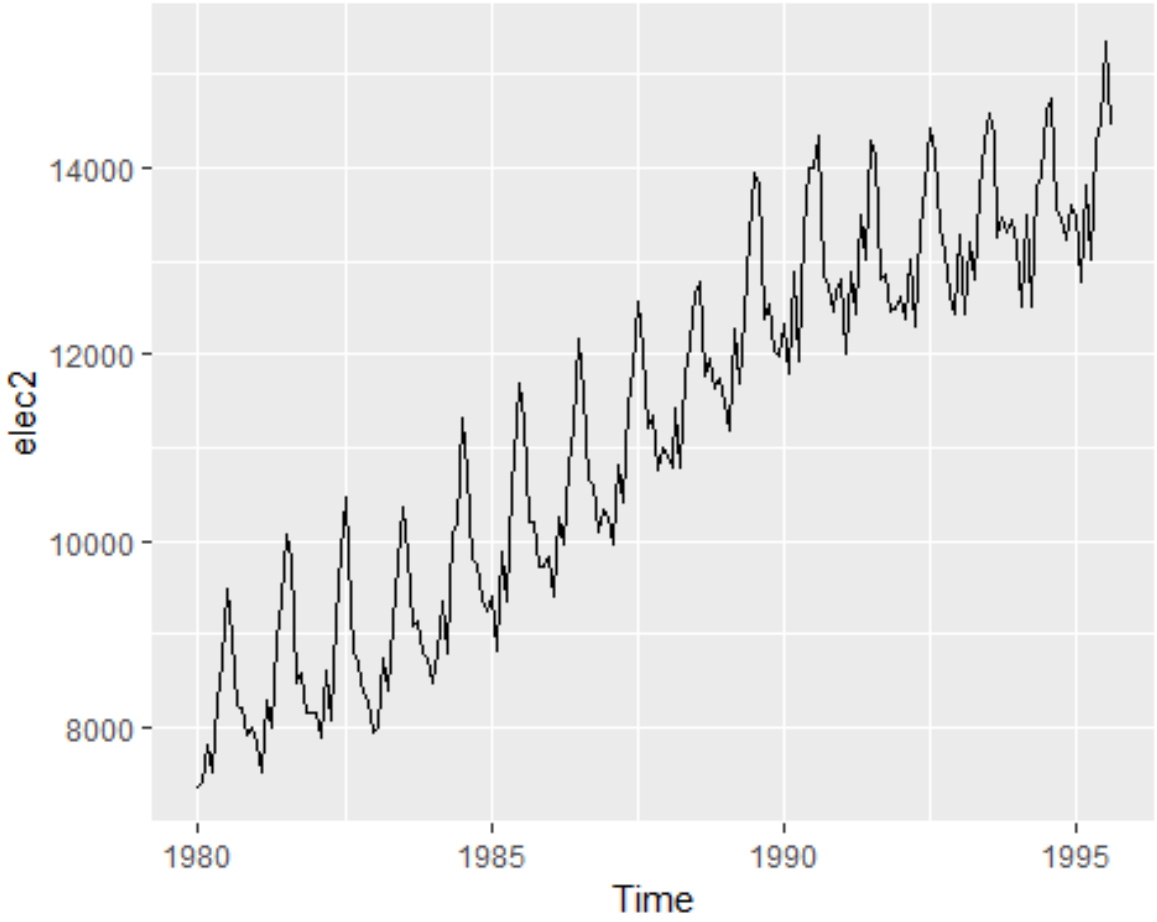


Trend and seasonality in ACF plots

- 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.

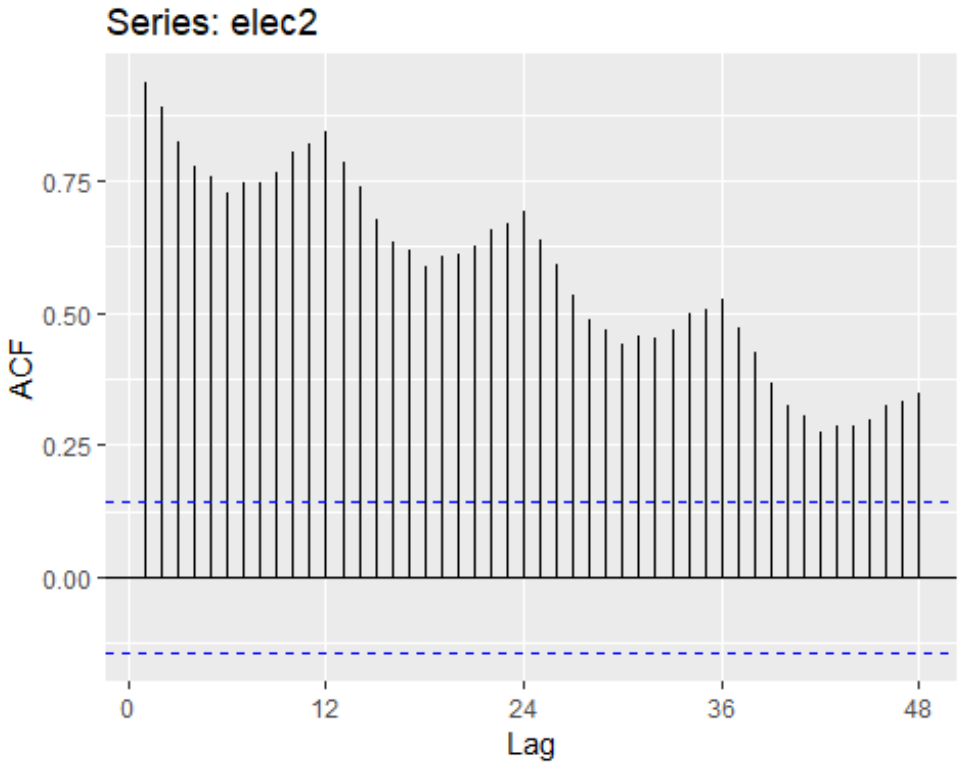
Aus monthly electricity production

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



Aus monthly electricity production

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



Time plot shows clear trend and seasonality.

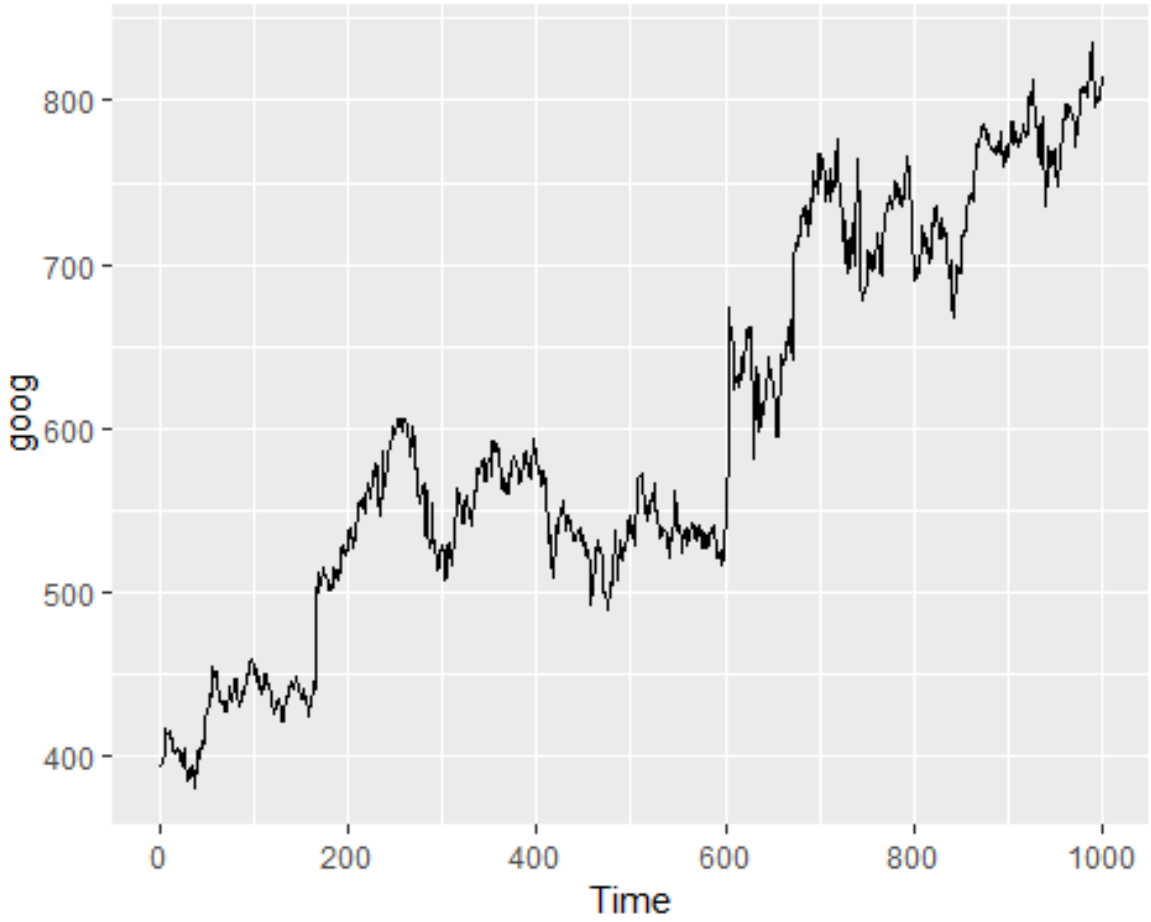
The same features are reflected in the ACF.

The slowly decaying ACF indicates trend.

The ACF peaks at lags 12, 24, 36, , indicate seasonality of length 12.

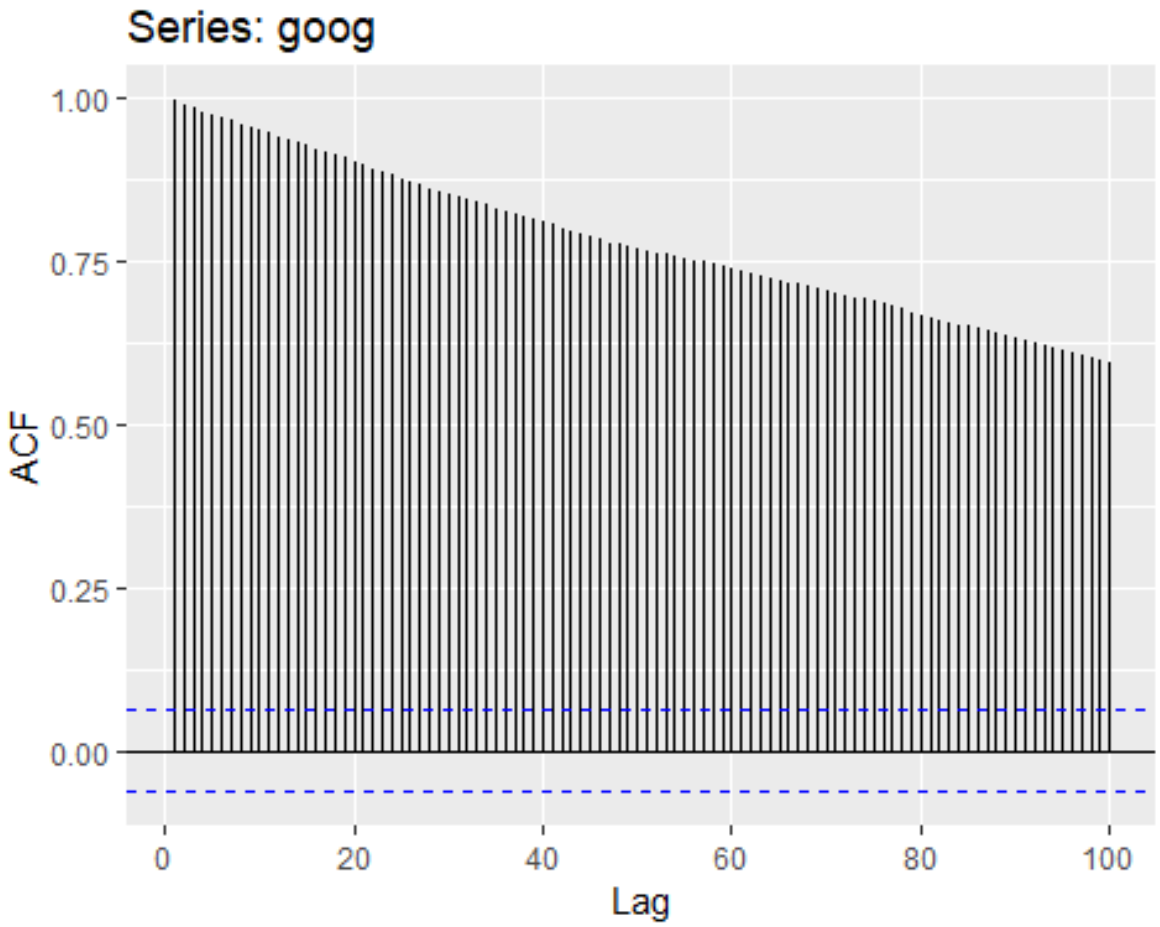
Google stock price

```
autoplot(goog)
```



Google stock price

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



Your turn

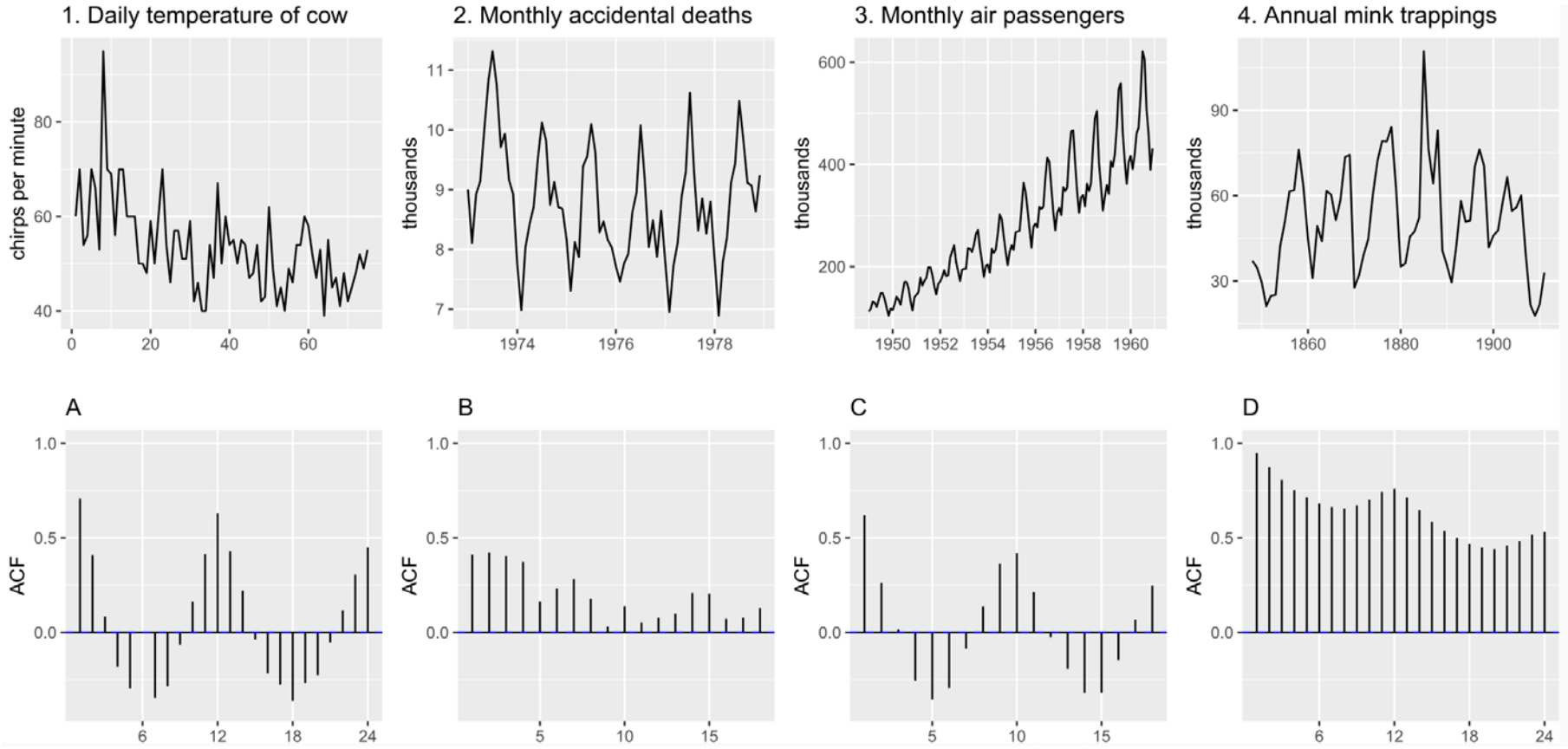
We have introduced the following graphics functions:

- `gglagplot`
- `ggAcf`

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

- `hsales`
- `usdeaths`
- `bricksq`
- `sunspotarea`
- `gasoline`

Which is which?

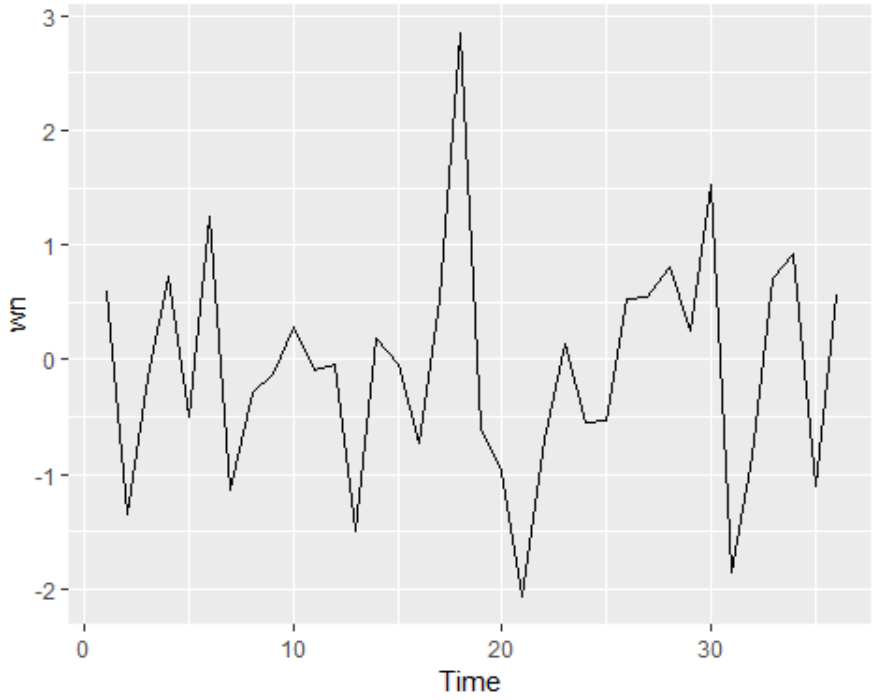


Outline

- Introduction
- Time Series in R
- Time plots
- Seasonal plots
- Seasonal or cyclic?
- Lag plots and autocorrelation
- White noise

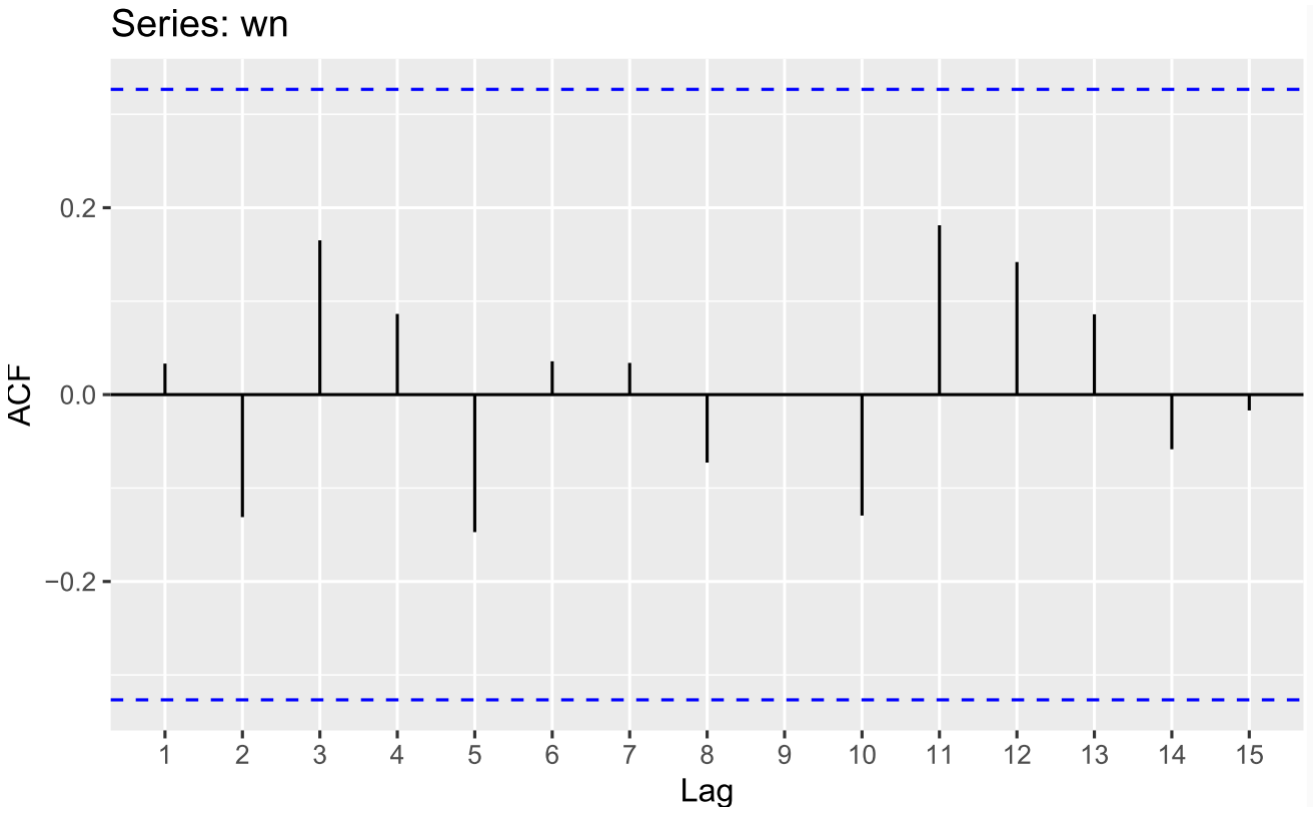
Example: White noise

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



Example: White noise

r_1 0.03
r_2 -0.13
r_3 0.17
r_4 0.09
r_5 -0.15
r_6 0.04
r_7 0.03
r_8 -0.07
r_9 0.00
r_{10} -0.13



Sample autocorrelations for white noise series.

We expect each autocorrelation to be close to zero.

Sampling distribution of autocorrelations

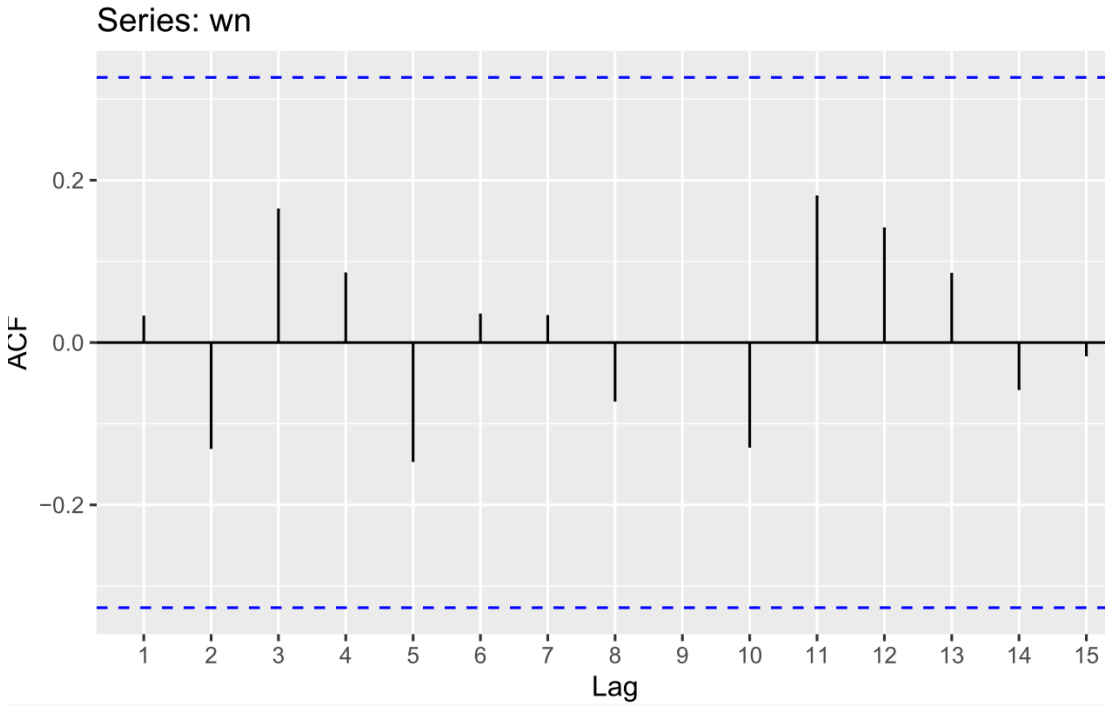
Sampling distribution of r_k for white noise data is asymptotically N(0,$1/T$).

- 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 Autocorrelation

T = 36 and so critical values at $\pm 1.96/\sqrt{36} = \pm 0.327$.

All autocorrelation coefficients lie within these limits, confirming that the data are white noise. More precisely, the data cannot be distinguished from white noise.

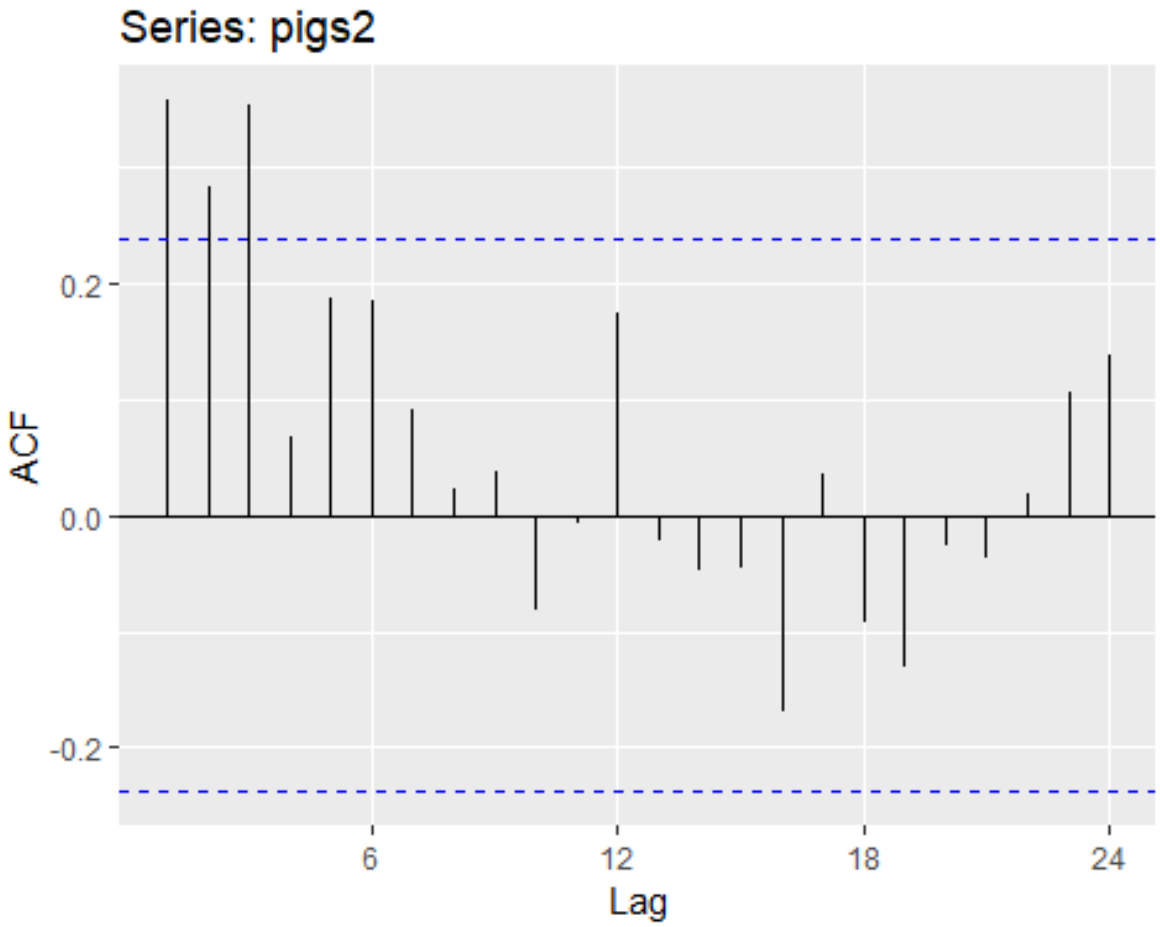


Autocorrelation

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

Example: Pigs slaughtered

```
ggAcf(pigs2)
```



Example: Pigs slaughtered

Monthly total number of pigs slaughtered in the state of Victoria, Australia, from January 1990 through August 1995. (Source: Australian Bureau of Statistics.)

- Difficult to detect pattern in time plot.
- ACF shows some significant autocorrelation at lags 1, 2, and 3.
- r_{12} relatively large although not significant. This may indicate some slight seasonality.

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

Summary