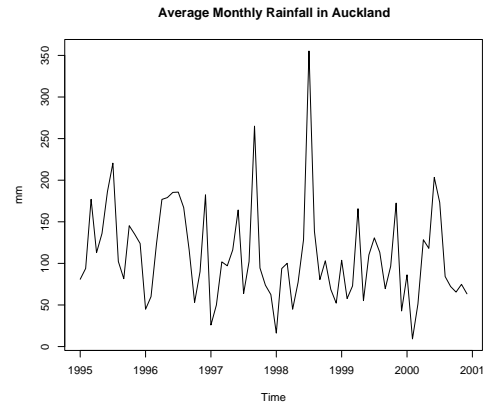


Statistics 120 Displaying Time Series Data

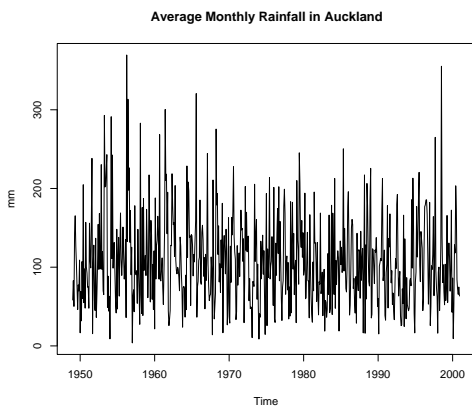


Time Series

- A *time series* is a set of observations made at equally spaced points in time.
- Time series observations are usually numerical measurements, but occasionally categorical time series are encountered.
- Time series observations are typically not (statistically) independent.
- This means that the time order the observations is crucial to their analysis.

Time Series in R

- The function `ts` can be used to turn an ordinary vector into a special time series object.
- It does this by specifying parameters which describe when the observations were made.
- The parameter `frequency` describes how many observations are made per unit time.
- The parameter `start` describes when sampling started.



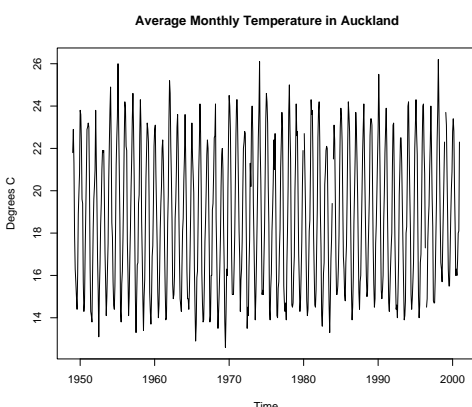
Example – Creating the Rain Series

- Suppose that the Auckland rainfall values have been read into a vector called `rainvalues`.
- The values are monthly values with the first value sampled in January 1949.

```
> rain = ts(rainvalues, frequency = 12,
            start = c(1949, 1))
```

- R interprets the value 1949 as “the start of 1949” so we could use the simpler form.

```
> rain = ts(rainvalues, frequency = 12,
            start = 1949)
```



Simple Operations on Time Series

- Arithmetic operations can be carried out on time series just as you might expect.

```
> lograin = log(rain)
```

- Subsetting is done by focusing on the values of a time series which fall within a given time *window*.

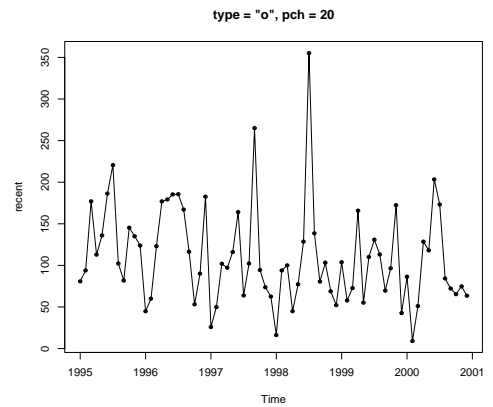
```
> rain2000 = window(rain,
                    start = c(2000, 1),
                    end   = c(2000, 12))
```

Printing Time Series

- Time series are printed in a special way.

```
> window(rain, start = c(1999, 1),
         end = c(2000, 12))
      Jan Feb Mar Apr May Jun
1999 103.8 57.7 72.9 165.7 55.2 110.2
2000  86.2  9.2 51.2 128.4 118.1 203.4
      Jul Aug Sep Oct Nov Dec
1999 130.7 113.2 69.6 96.6 172.5 42.8
2000 173.2  84.2 72.2 65.5 74.8 63.4
```

- The frequency values 12 and 4 are recognised as special and taken to correspond to monthly and quarterly observations.



Time Series Plots

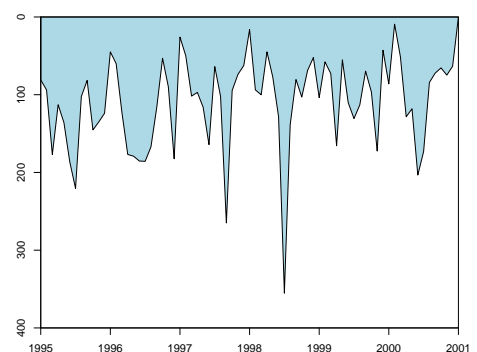
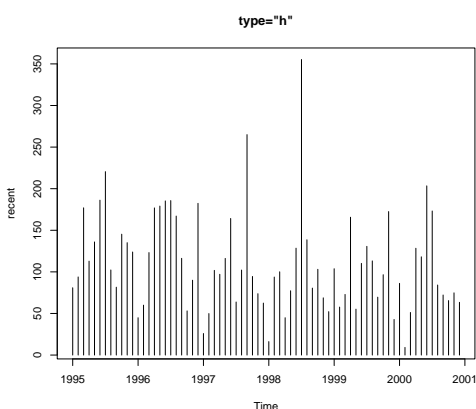
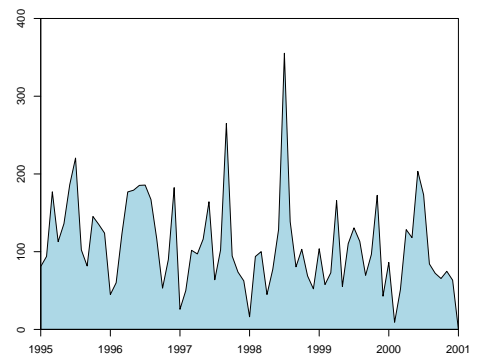
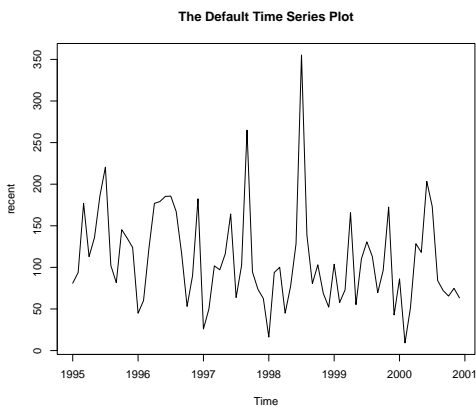
- The plot function recognises time series and plots them in an appropriate way.
- The default plotting method is to "join up the dots," but this and other aspects of the plot can be customised.

```
> recent = window(rain, start = c(1995, 1),
                 end = c(2000, 12))
> plot(recent)
> plot(recent, type = "h")
> plot(recent, type = "o", pch=20)
```

Time Series Plots

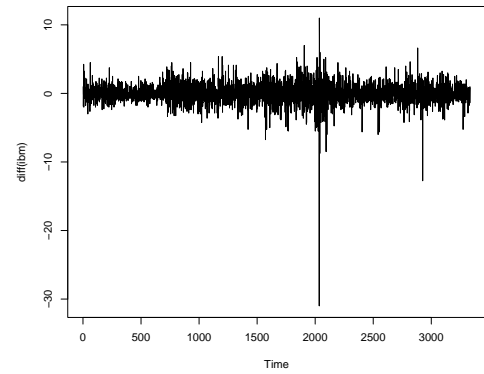
- We saw earlier in the course that it is easy to produce filled time series plots using polygon.

```
> plot.new()
> plot.window(c(1995, 2001), xaxs = "i",
             c(0,400), yaxs = "i")
> x = c(1995, time(recent), 2001)
> y = c(0, recent, 0)
> polygon(x, y, col = "lightblue")
> axis(1); axis(2); box()
```



A Horizon Effect?

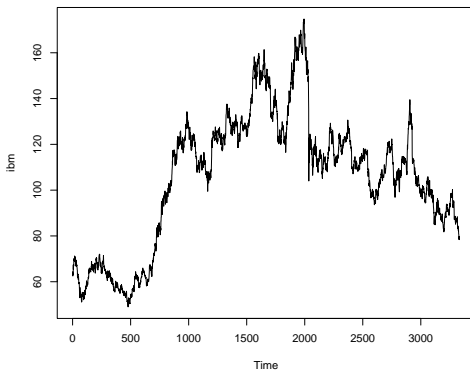
- There appears to be a difference between these two plots.
- My conjecture is that there is wiring in the brain which means that we notice peaks rather than troughs in plots.
- This is especially true for plots which are divided horizontally by a colour horizon.
- Because of this effect, my recommendation is that you avoid this kind of plot.



Example: Stock Prices

- In this example we will look at the closing price for IBM stock, daily from Jan 1, 1980 to Oct. 8, 1992.
- This is a typical pattern for any stock.

```
> plot(ibm)
```



Time Series Decomposition

- It can be useful to regard many real world series as being composed of several independent components.
- A particularly useful model is the trend plus seasonal plus irregular component model.

$$x_t = T_t + S_t + I_t$$

where

- T_t = a slowly varying trend model
- S_t = a periodic seasonal component
- I_t = a set of random irregular "shocks"

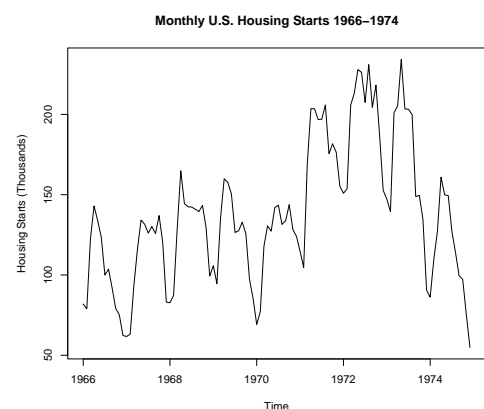
Example – U.S. Housing Starts

- The number of housing starts in any given month is an important leading economic indicator.
- Houses are only built when there are clearly economic "good times" ahead.
- This example shows the United States housing start series from 1966 to 1974.

Stock Prices and Efficient Markets

- Theory says that in an efficient market stock prices should behave as random walks.
- This means that on any given day the price of a stock will go up or down with equal probability.
- There are a variety of reasons why the New Zealand market cannot be considered efficient.
- We can check the theory with the IBM stock by examining the first differences in the series – i.e. each day's value minus the day before.

```
> plot(diff(ibm))
```



Interpretation

- The series clearly shows:
 - a regular seasonal variation with a peak in housing starts in summer and a trough in winter.
 - a long term (cyclical) trend.
 - short term irregularities which are not explained by the other two components.
- This is typical of monthly or quarterly economic series.

Interpretation

- The procedure has done a good job of decomposing the original series into interpretable subseries.
- The seasonal subseries is remarkably stable over time.
- The irregular component is far from being a “random” series.

Seasonal Decomposition

- There are statistical techniques which can be used to decompose a time series into trend plus seasonal plus irregular components.
- We will use a technique called STL which uses the lowess smoother as follows.
 - A long term trend is estimated using a lowess smooth and then subtracted from the series.
 - Each month (or quarter) is smoothed separately and this seasonal effect is subtracted.
 - The remainder of the series is taken to be the irregular component.

Seasonal Adjustment

- Seasonal effects tend to obscure the trends and short term variation present in a time series.
- A technique called seasonal adjustment is used to remove seasonal variation from a time series.
- A seasonally adjusted series can be obtained from the results produced by `stl`, by adding the trend and irregular components.

```
> plot(sd$time.series[,2]+sd$time.series[,3],
      main="Seasonally Adjusted Housing Starts",
      ylab="Housing Starts (Thousands)")

> monthplot(sd$time.series[,1],
           ylab="Seasonal Effect")
```

Seasonal Decomposition in R

- Assuming that we have the housing start series stored in `hstart`, here is how we carry out a seasonal decomposition and display the result graphically.

```
> library(ts)
> sd = stl(hstart, s.window=10, t.window=10)
> plot(sd)
```

- The STL procedure can be tuned in a variety of ways.
- The values of `s.window` and `t.window` determine the amount of smoothing used to determine the seasonal pattern and the trend (larger values produce more smoothing).

