Class 1: Auto-regressive models and random walks

Andrew Parnell (with some extra code by Doug McNeall)
andrew.parnell@ucd.ie

Learning outcomes

- Understand the key parts that make up a stationary time series
- Know what an Autocorrelation (ACF) and Partial Autocorrelation (PACF) function is
- Understand how a random walk process is generated
- Understand the AR(p) process

The most important slide in the course

Almost all of time series is based on two ideas:

1. Base your future predictions on previous values of the data
2. Base your future predictions on how wrong you were in your past predictions

The remainder of the course is focussed on applying these two ideas in increasingly complex situations

For today we only discuss *discrete time series*,
i.e. where $t = 1, 2, 3, \ldots$

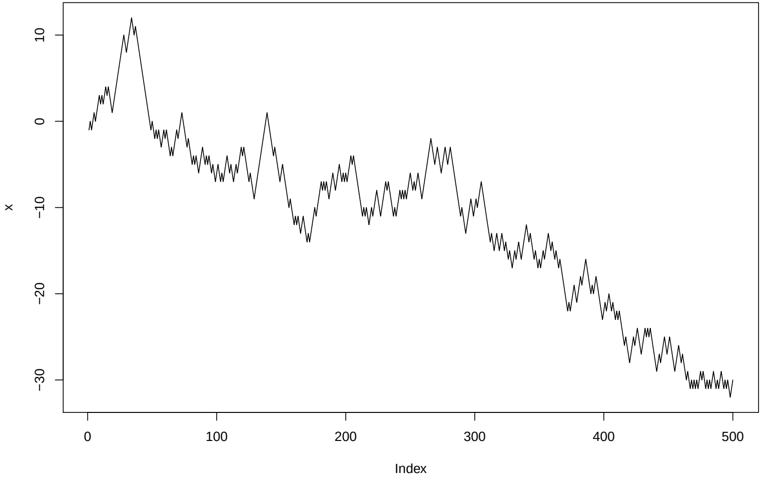
Recall: decomposing time series

- We decompose time series commonly as:

$$y_t = \text{trend}_t + \text{seasonality}_t + \text{error}_t$$

- . . . but sometimes it is not easy to separate these into different parts
- The concept of *stationarity* helps us decompose the time series

A time series with a big trend?

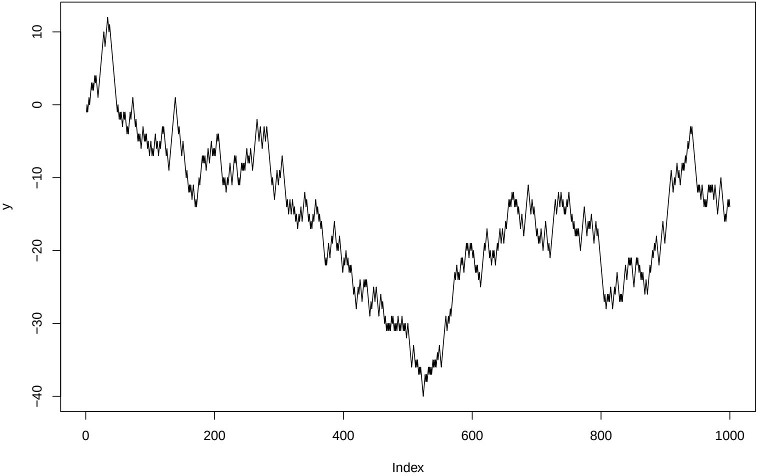


Generating the series

```r
set.seed(123)
y <- cumsum(sample(c(-1, 1), size=1000, replace=TRUE))
```

- The `sample` command just produces a set of 1000 values either -1 or 1
- `cumsum` just cumulatively adds them up
- This is a *random walk* series

And what happens when we continue the series?



Generating a one-dimensional random walk

1. Start at zero
2. Flip a coin and move (+1) for heads (-1) for tails
3. Repeat

Where do you end up? - Expected mean is 0 as number of steps gets large

$$\hat{Y}_t = Y_{t-1}$$

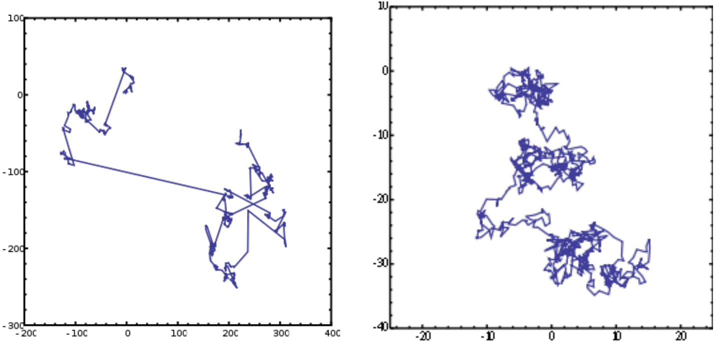
- The variance of the series gets larger as you move into the future
- But in an infinite series, every point (including zero) is crossed an infinite number of times. This causes the **gambler's ruin**

Random walk applications

The steps don't have to be the same size.

- A step size with a Gaussian distribution is useful for modelling stock markets
- If the step size probability distribution is heavy tailed, we have *Levy flight*
- They also don't have to be in 1 dimension. We can simulate random walks in 2, 3, . . . dimensions by adding multivariate steps

Levy flight vs Brownian motion in 2D



There is evidence that animals such as sharks follow a levy flight pattern (left) when foraging for food - they had previously been thought to approximate Brownian motion (right)

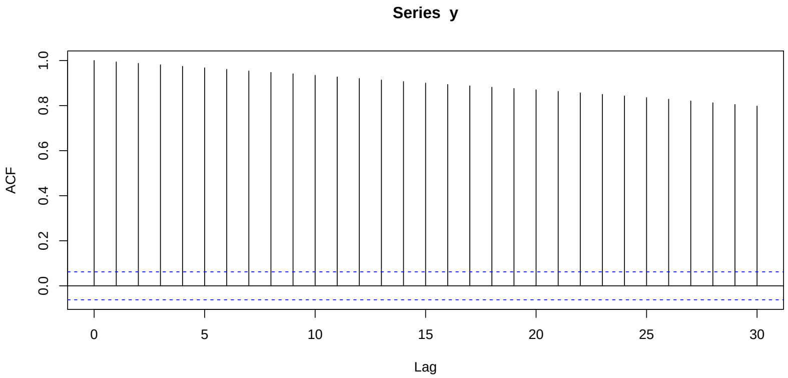
Diagnosing time series behaviour

- A useful way to characterise the behaviour of a time series is the *autocorrelation* function
- (*auto* = *self*, i.e. correlation of the series with itself)
- We calculate the correlation of the series with itself shifted by 1 time point
- The shifted data set is known as the *lagged* time series
- We repeat the autocorrelation calculation for 1 lag, 2 lags, 3 lags, etc

Plotting the acf

- R has a function `acf` which will plot this for you

```
acf(y)
```

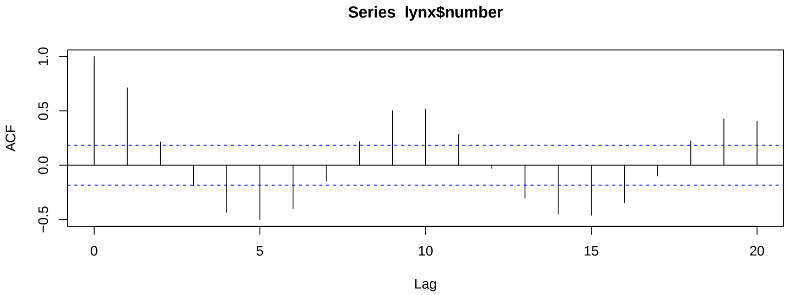


This random walk series has very high autocorrelation

A real acf

- With seasonal data, you might see patterns in the ACF

```
lynx = read.csv('../../data/lynx.csv')
acf(lynx$number)
```



- R automatically shows significance levels for the ACF plot

Autoregressive models

Autoregressive (AR) models

- Autoregressive models literally perform a linear regression of the time series against the previous lag of the series
- For example, an AR(1) process can be written as:

$$y_t = \alpha + \beta y_{t-1} + \epsilon_t$$

- where $\epsilon_t \sim N(0, \sigma^2)$ just like a linear regression.
- In a probability distribution format, we might write:

$$y_t \sim N(\alpha + \beta y_{t-1}, \sigma^2)$$

. . . and maximise the likelihood as normal

Interpretation of the AR parameters

- α is an estimate of the stable mean of the process
- β is interesting:
  - Values close to 1 indicate that the series is almost like a random walk.
  - Values close to 0 indicate that the series is almost completely composed of random normally-distributed error terms
  - Values less than 0 indicate that the series is 'repulsive'
  - Values greater than 1 (or less than -1) indicate that the series is chaotic

Simulating from an AR(1) process

- We can simulate from an AR(1) process with:

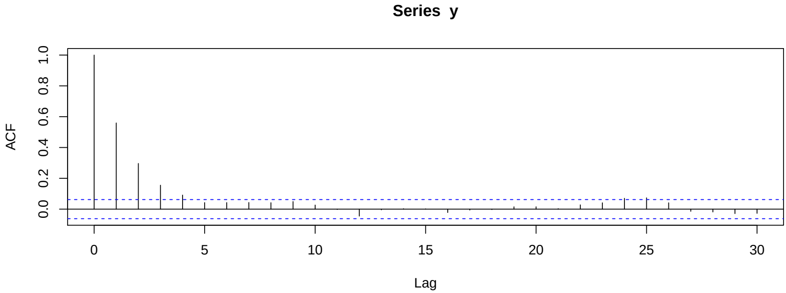
```r
set.seed(123)
T = 1000
t_seq = 1:T
sigma = 0.4
alpha = 1
beta = 0.6
y = rep(NA,T)
y[1] = rnorm(1,0,sigma)
for(t in 2:T) y[t] = rnorm(1,alpha + beta * y[t-1],
                           sigma)
```

- It's fun to play around with values of beta and see what happens to the model

Some features of an AR(1) process

- For an AR(1) process, the ACF decays exponentially if $\beta < 1$
- We can see this if we look at the acf

```
acf(y)
```



Extending the AR model

- An AR(2) process is written as:

$$y_t = \alpha + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \epsilon_t$$

- An AR(p) process can be written as:

$$y_t = \alpha + \sum_{i=1}^{p} \beta_i y_{t-i} + \epsilon_t$$

- The restrictions on the values of β_i get much more complicated as p increases to avoid a chaotic series

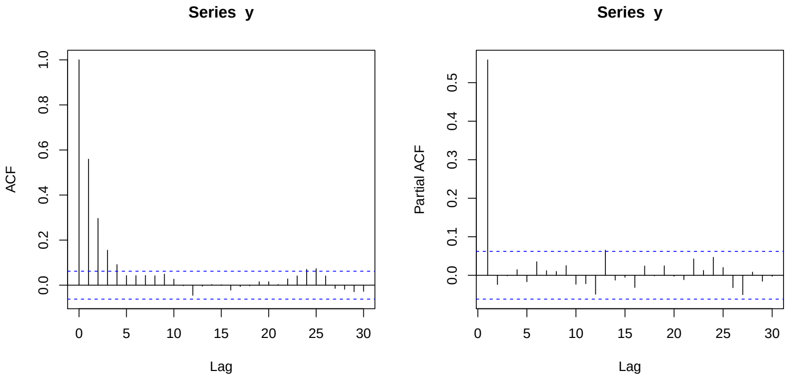
The partial autocorrelation function

- Another commonly plotted summary statistic from a time series is the *partial autocorrelation function* or PACF
- The PACF at lag k is created by:
  1. Creating an AR(k-1) model
  2. Getting the residuals $\hat{\epsilon}_t$
  3. Calculating the autocorrelation function between the data and the residuals
- You can think of it as being the autocorrelation function for lag k having *removed* the effect of the lower level lags

A plot of the PACF

- The ACF and PACF for the AR(1) process we generated

```
par(mfrow=c(1, 2))
acf(y)
pacf(y)
```



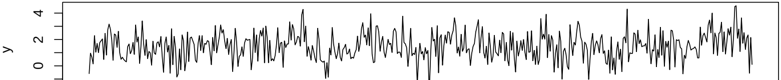
Some rules about ACF and PACF

- In general for an AR(p) process:
  - There will be exponential decay in the ACF
  - There will be p sticking out ('significant') lags in the PACF

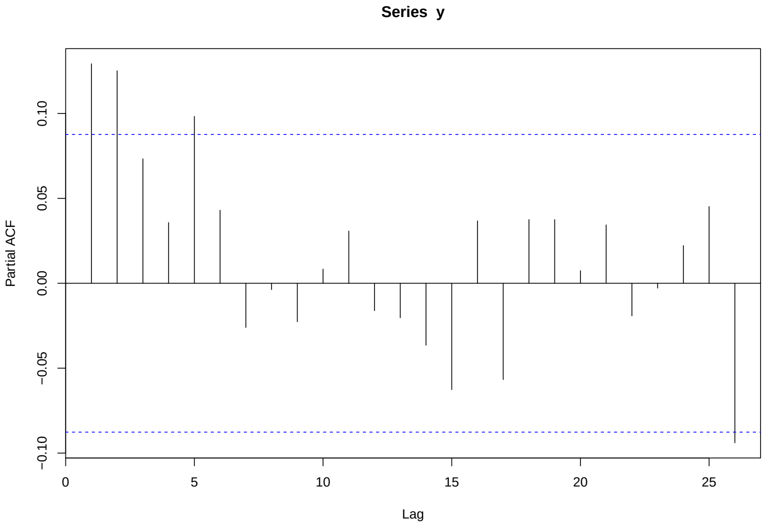
Thus you can have a good initial guess at identifying p in an AR(p) model from the ACF and PACF

Simulating from an AR(p) process

```r
set.seed(100)
T = 500
t_seq = 1:T
sigma = alpha = 1
p = 3
beta = sort(runif(p, 0, 0.2), decreasing = TRUE)
y = rep(NA,T)
y[1:p] = rnorm(p, mean = alpha, sd = sigma)
for(t in (p+1):T)
  y[t] = rnorm(1,
               alpha + sum( beta * y[(t-1):(t-p)] ),
               sigma)
plot(t_seq,y,type='l')
```



PACF for our example AR3 process



Predicting the future

- Once we have our model we have an easy way to predict the future:
$$\hat{y}_{t+1} = \hat{\alpha} + \hat{\beta} y_t$$
- If we want to forecast further ahead we have to start using the forecast values, e.g.
$$\hat{y}_{t+2} = \hat{\alpha} + \hat{\beta} \hat{y}_{t+1}$$
- We can create a confidence interval for our predictions by using the standard errors of $\hat{\alpha}$ and $\hat{\beta}$
- We can create a *prediction interval* by also including the values $\hat{\sigma}$ and simulating from the full probability distribution

Stationarity

- One of the key concepts in time series is *stationarity*
- A time series is said to be weakly stationary if:
  1. The mean is stable
  2. The variance is stable
  3. The autocorrelation doesn't depend on where you are in the series
- The first can be checked from plots of the time series, but the last one is quite tricky
- All AR models are stationary provided the restrictions on β are met

What if a series isn't stationary?

- If a time series isn't stationary we might:
  - stabilise the mean by removing the trend or seasonality (by e.g. using linear regression)
  - make the variance stable by performing a transformation such as log or Box-Cox
  - Fit a more advanced non-stationary model (see later in the course)
- Quite often, when a series is not mean or variance stationary, people will *difference* the time series, and fit models only to the differences
- You can also *seasonally difference* the data

Fitting models in R

- The R function `Arima` in the `forecast` package will fit AR models for you
- To fit an AR(1) model to data y we just type:

```
Arima(y, order = c(1, 0, 0))
```

```
## Series: y
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1    mean
##       0.1305  1.4878
## s.e.  0.0446  0.0526
##
## sigma^2 estimated as 1.053:  log likelihood=-721.3
## AIC=1448.6   AICc=1448.65   BIC=1461.24
```

Summary

- The ACF and PACF are two really useful tools to run on every time series (as well as a plot of the time series itself)
- The AR(p) framework is a class of models for stationary time series which regresses the current values of the series on past values
- A time series is stationary if it has stable mean and standard deviation, and an ACF which doesn't change over time
- There are lots of ways to make a series stationary. The most common one is differencing