Class 2: Moving averages and ARMA

Andrew Parnell
andrew.parnell@ucd.ie

Learning outcomes

- Recognise and understand the basic theory behind MA(1) and MA(q) models
- Understand the basic ARMA(p,q) formulation
- Know the basics of using the `forecast` package
- Understand the limitations of ARMA forecasting

Reminder: 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**

Reminder: AR models

- An Autoregressive (AR) model works by making the current data point dependent on the previous value, dampened by a parameter
- The usual likelihood used is:

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

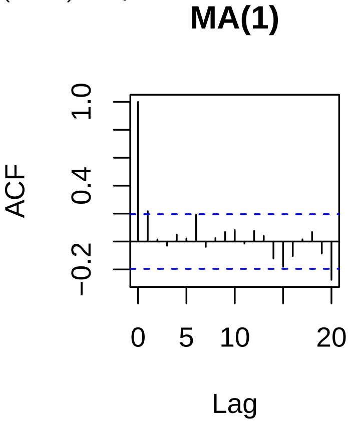
- β is usually constrained (naturally via the data) to lie between -1 and 1. Outside that range the process blows up
- The sample PACF is often a good way of diagnosing if an AR model might be appropriate

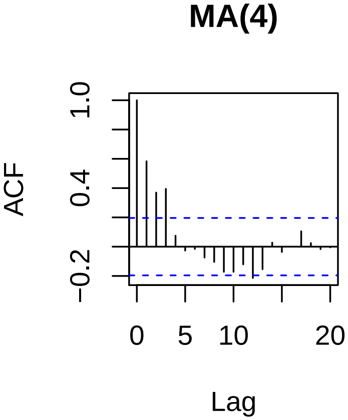
Intro to Moving Average Models

- Moving Average (MA) models are similar to AR models but they depend on the previous residual of the series rather than the value itself
- The previous residual is made up of how well we forecasted the last value of the series
- If the previous residual was large (i.e. our forecast was bad) then we want to make a big change to the next prediction
- If the previous residual was small (i.e. our forecast was good) then we might not want to make much of a change

Moving average models and the ACF/PACF

- Recall that the sample partial autocorrelation function (PACF) can be used to diagnose whether an AR model is appropriate (and also suggest the order p)
- For the MA model, it is the sample autocorrelation function (ACF) helps determine the order of the model





Example 1: MA(1)

- The MA(1) model is defined as:

$$y_t = \alpha + \theta\epsilon_{t-1} + \epsilon_t$$

  where $\epsilon_t \sim N(0, \sigma^2)$ as usual
- Parameter α represents the overall mean, whilst θ controls the amount of weight placed on previous residuals
- Like the AR model the values of θ are not expected to be outside (-1, 1), and negative values can sometimes be physically unrealistic
- The likelihood version of the model is:

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

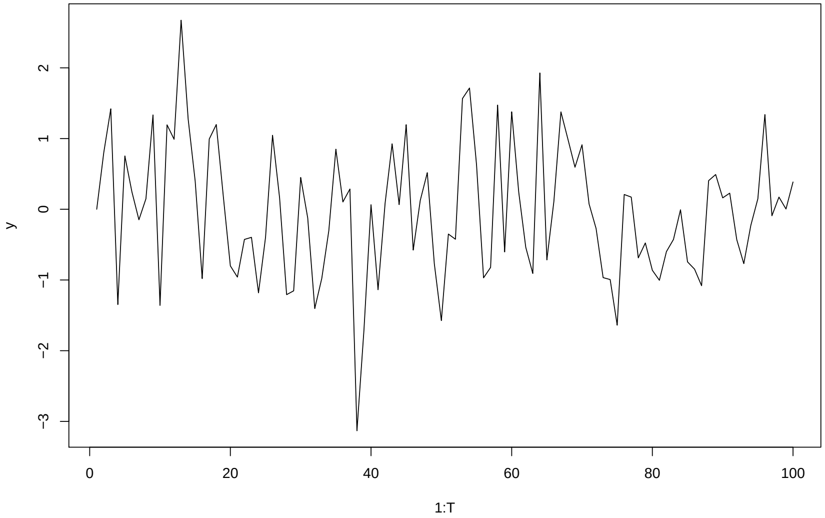
Simulating from the MA(1) process

Below is some simple code to simulate from an MA(1) process.
Note that the first values of `y` and `eps` need to be initialised

```
T = 100 # Number of observations
sigma = 1 # Residual sd
alpha = 0 # Mean
theta = runif(1) # Choose a positive value
y = eps = rep(NA,T)
y[1] = alpha
eps[1] = 0
for(t in 2:T) {
  y[t] = rnorm(1, mean = alpha + theta * eps[t-1],
                  sd = sigma)
  eps[t] = y[t] - alpha - theta * eps[t-1]
}
```

Time series plot

```
plot(1:T,y,type='l')
```



Fitting MA(1) models

- We can fit an MA(1) model with the `forecast` package like before

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

```
## Series: y
## ARIMA(0,0,1) with non-zero mean
##
## Coefficients:
##          ma1     mean
##       0.2118  -0.0325
## s.e.  0.0924   0.1120
##
## sigma^2 estimated as 0.8747:  log likelihood=-134.21
## AIC=274.43   AICc=274.68   BIC=282.24
```

Extending to MA(q)

- It's reasonably straightforward to extend this model to have the current value of y depending on more than one previous residual
- The model becomes an MA(q) model with:

$$y_t \sim N(\alpha + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q}, \sigma^2)$$

- The parameters are as before, except there are now q values of θ.
- Usually when estimated they will decrease with q; the older residuals matter less

Fitting an MA(q) model

```r
Arima(y, order = c(0, 0, 3))
```

```
## Series: y
## ARIMA(0,0,3) with non-zero mean
##
## Coefficients:
##          ma1     ma2      ma3     mean
##       0.2269  0.0170  -0.0413  -0.0338
## s.e.  0.1005  0.0981   0.0876   0.1111
##
## sigma^2 estimated as 0.8897:  log likelihood=-134.04
## AIC=278.07   AICc=278.71   BIC=291.1
```

- Compare the AIC of this model with the previous MA(1) version

Forecasting an MA value

- You can create a one step ahead forecast for an MA(1) model by:

$$\hat{y}_{t+1} = \alpha + \theta\epsilon_t$$

- Forecasts of more than one step ahead will be pretty boring, as every future prediction of $\hat{\epsilon_t}$ will be 0
- Thus MA(q) models are only really informative if you are forecasting $q - 1$ steps ahead

Combining AR and MA into ARMA

- There is no reason why we have to use just AR or MA on their own
- It's possible to combine them together, for example:

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

  This is an *Autoregressive Moving Average* (ARMA) model
- It's often written as ARMA(p,q) where p is the number of AR terms (here 1) and q the number of MA terms (here also 1)
- ARMA models can deal with a very wide variety of flexible time series behaviour, though they remain stationary
- The likelihood format is:

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

Fitting an ARMA(1, 1) model

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

```
## Series: y
## ARIMA(1,0,1) with non-zero mean
##
## Coefficients:
##          ar1     ma1     mean
##       0.1119  0.1085  -0.0323
## s.e.  0.3743  0.3703   0.1153
##
## sigma^2 estimated as 0.8829:  log likelihood=-134.17
## AIC=276.33   AICc=276.76   BIC=286.76
```

- Compare again with previous models

The general ARMA(p, q) framework

- The general equation for an ARMA(p, q) model is:

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

- The values of β have to be tightly controlled to get a series that is stationary, though this is only really a problem if we want to simulate the time series
- Occasionally you will run into problems with `Arima` because it doesn't use maximum likelihood (by default) to fit the models. It uses something faster and more approximate instead

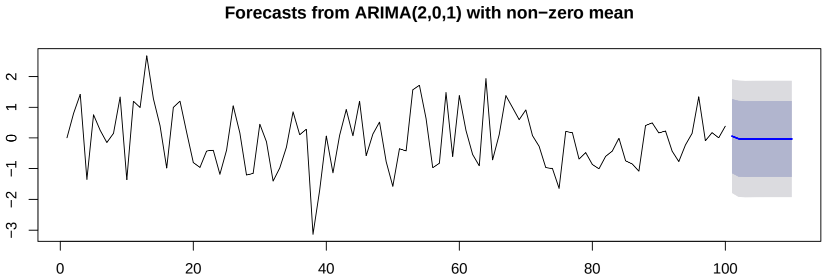
Predicting the future with ARMA

- The `forecast` package contains methods to predict into the future
- First create a model (here ARMA(2, 1))

```
my_model = Arima(y, order = c(2, 0, 1))
```

- ...then forecast...

```
plot(forecast(my_model,h = 10))
```



A real-world example

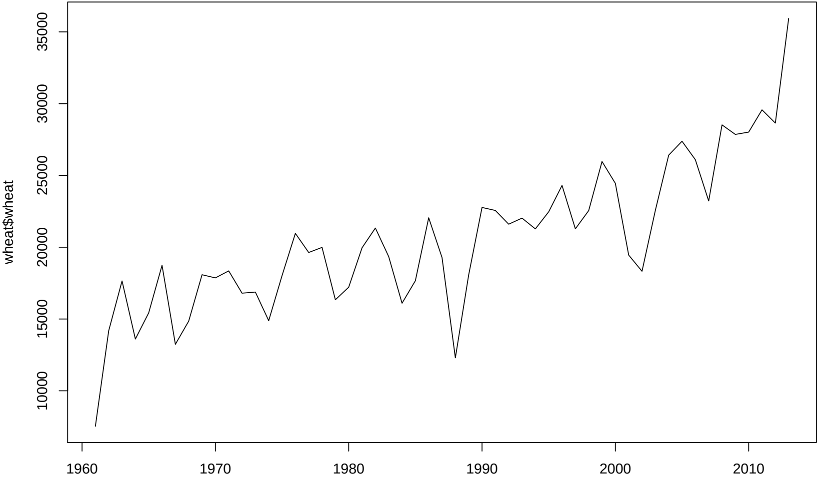
Steps in a time series analysis

1. Plot the data and the ACF/PACF
2. Decide if the data look stationary or not. If not, perform a suitable transformation and return to 1
3. Guess at a suitable p and q for an ARMA(p, q) model
4. Fit the model
5. Try a few models around it by increasing/decreasing p and q and checking the AIC (or others)
6. Check the residuals
7. Forecast into the future

A real example: wheat data

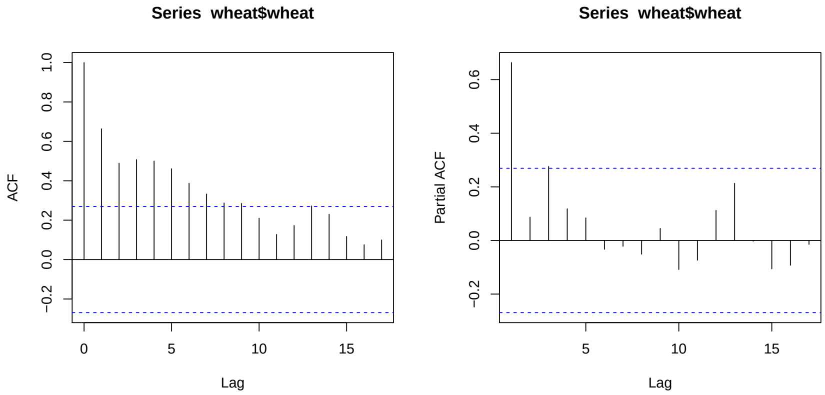
- Let's follow the steps for the wheat data:

```
wheat = read.csv('../../data/wheat.csv')
plot(wheat$year, wheat$wheat, type = 'l')
```



ACF and PACF

```r
par(mfrow = c(1, 2))
acf(wheat$wheat)
pacf(wheat$wheat)
```



- Suggest starting with AR(1) or AR(3)?

First model

```r
Arima(wheat$wheat, order = c(1, 0, 0))
```

```
## Series: wheat$wheat
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1       mean
##       0.8972  20849.522
## s.e.  0.0826   3615.699
##
## sigma^2 estimated as 10079564:  log likelihood=-502.34
## AIC=1010.68   AICc=1011.17   BIC=1016.59
```

Next models

- Try AR(2), ARMA(1, 1), and ARMA(2, 1)

```r
Arima(wheat$wheat, order = c(2, 0, 0))$aic
```

```
## [1] 1012.683
```

```r
Arima(wheat$wheat, order = c(1, 0, 1))$aic
```

```
## [1] 1011.36
```

```r
Arima(wheat$wheat, order = c(2, 0, 1))$aic
```

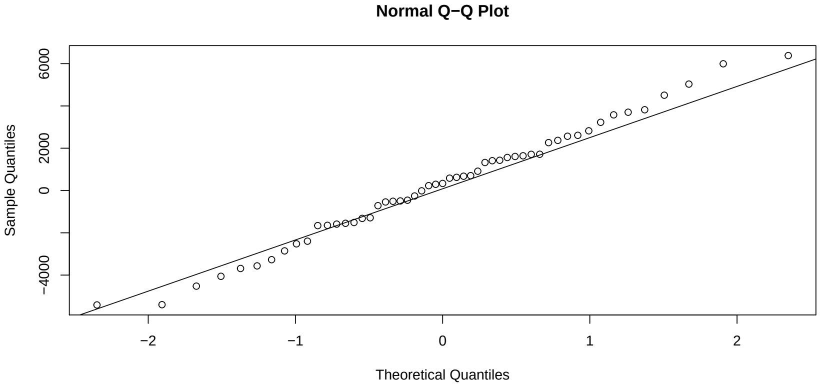
```
## [1] 1004.125
```

- Best one seems to be ARMA(2, 1). (could also try others)

Check residuals

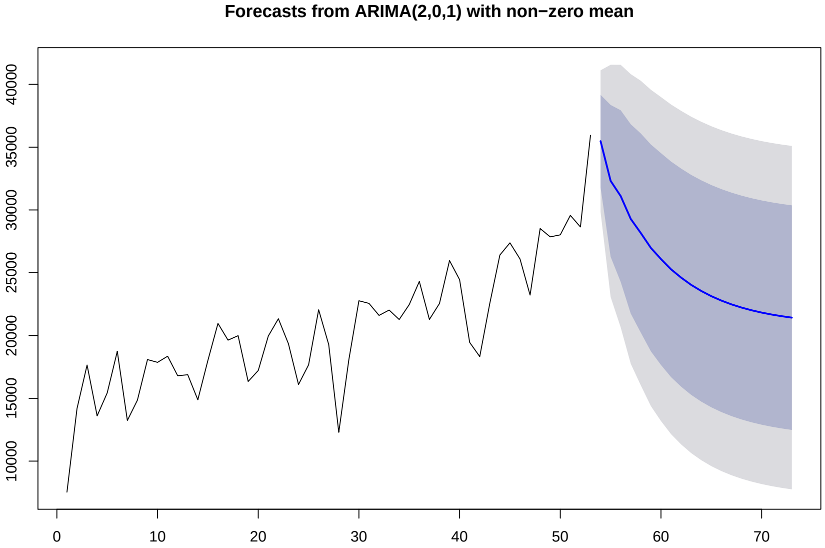
- Check the residuals of this model

```
my_model_ARMA21 = Arima(wheat$wheat, order = c(2, 0, 1))
qqnorm(my_model_ARMA21$residuals)
qqline(my_model_ARMA21$residuals)
```



Forecast into the future

```
plot(forecast(my_model_ARMA21,h=20))
```



What happened to the forecasts here?

- Why did the series diverge rapidly away from what you might have expected?
- The answer is that we have fitted a *stationary model*, i.e. one with constant mean and variance
- The model will just slowly reverts back to that mean over time. The speed at which it reverts will depend on the amount of autocorrelation in the series
- The solution to this lies in better identification of the trend. See the next lecture!

Summary

- MA(q) models are used to create future forecasts based on the error in the previous forecasts
- ARMA models combine AR and MA ideas together
- The `forecast` package allows us to fit all of these models
- We need to be a bit careful with forecasts that assume stationarity - they will mean-revert