The ARMA Model

Full Name:

The **A**uto**r**egressive **M**oving **A**verage Model

Mathematical Notation:

$$r_t = c + \varphi_1 r_{t-1} + \theta_1 \varepsilon_{t-1} + \varepsilon_t$$

r_t, r_{t-1}	Values in the current period and 1 period ago respectively
ε_t, ε_{t-1}	Error terms for the same two periods
c	Baseline constant factor
φ_1	What part of the value last period is relevant in explaining the current one
θ_1	What part of the error last period is relevant in explaining the current value

Description:

The name comes from combining the names of the two simpler models it incorporates – the AR and the MA.

The ARMA incorporates both past values (like the AR) and past errors (like the MA). By including both, we should improve our estimates. This is because we are enabling our AR model to calibrate (by including how far off our predictions were) and also giving a benchmark (different from the constant) to the MA model, which should severely decrease the variation in the residuals.

Picking the correct order for such a model could be tricky, since including or removing AR and MA orders can have wildly different effects on the accuracy.

The ARMA Model

Implementation of the Simple Model in Python:

The library the *ARMA* method comes from

The method we are importing

```
from statsmodels.tsa.arima_model import ARMA
```

```
model_ar_1_ma_1 = ARMA(df.market_value, order=(1,1))
```

The variable storing the model characteristics that we will fit later

The time series we wish to analyse

The order of the model

*For an ARMA(p,q) model, simply change the order from (1,1) to (p,q).