

STAT 436 / 536 - Lecture 13: Key

ARMA Process

- Thus far we have fit either AR(p) processes

$$x_t = \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \dots + \alpha_p x_{t-p} + w_t$$

or MA(q) processes

$$x_t = w_t + \beta_1 w_{t-1} + \dots + \beta_q w_{t-q}$$

- However, it would be reasonable to want to fit both at the same time. This results an ARMA model.

$$x_t = \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \dots + \alpha_p x_{t-p} + w_t + \beta_1 w_{t-1} + \dots + \beta_q w_{t-q}$$

- ARMA models can be written using the polynomial expression with the characteristic equations:

$$\theta_p(B)x_t = \phi_q(B)w_t$$

There are several key points about an ARMA(p,q) process:

- a. The process is stationary when the absolute value of the roots of θ all exceed 1.
- b. The process is invertible when the absolute value of the roots of ϕ all exceed 1.
- c. An AR(p) model is a special case of an ARMA model, specifically ARMA(p,0).
- d. Similarly, the MA(q) model is a special case of an ARMA(0,q) model.
- e. When fitting to data, an ARMA model will often be more parameter efficient than a single AR or MA model.
- f. When θ and ϕ share a common factor, a stationary model can be simplified. For example, with

$$(1 - \frac{1}{2}B)(1 - \frac{1}{3}B)x_t = (1 - \frac{1}{2}B)w_t$$

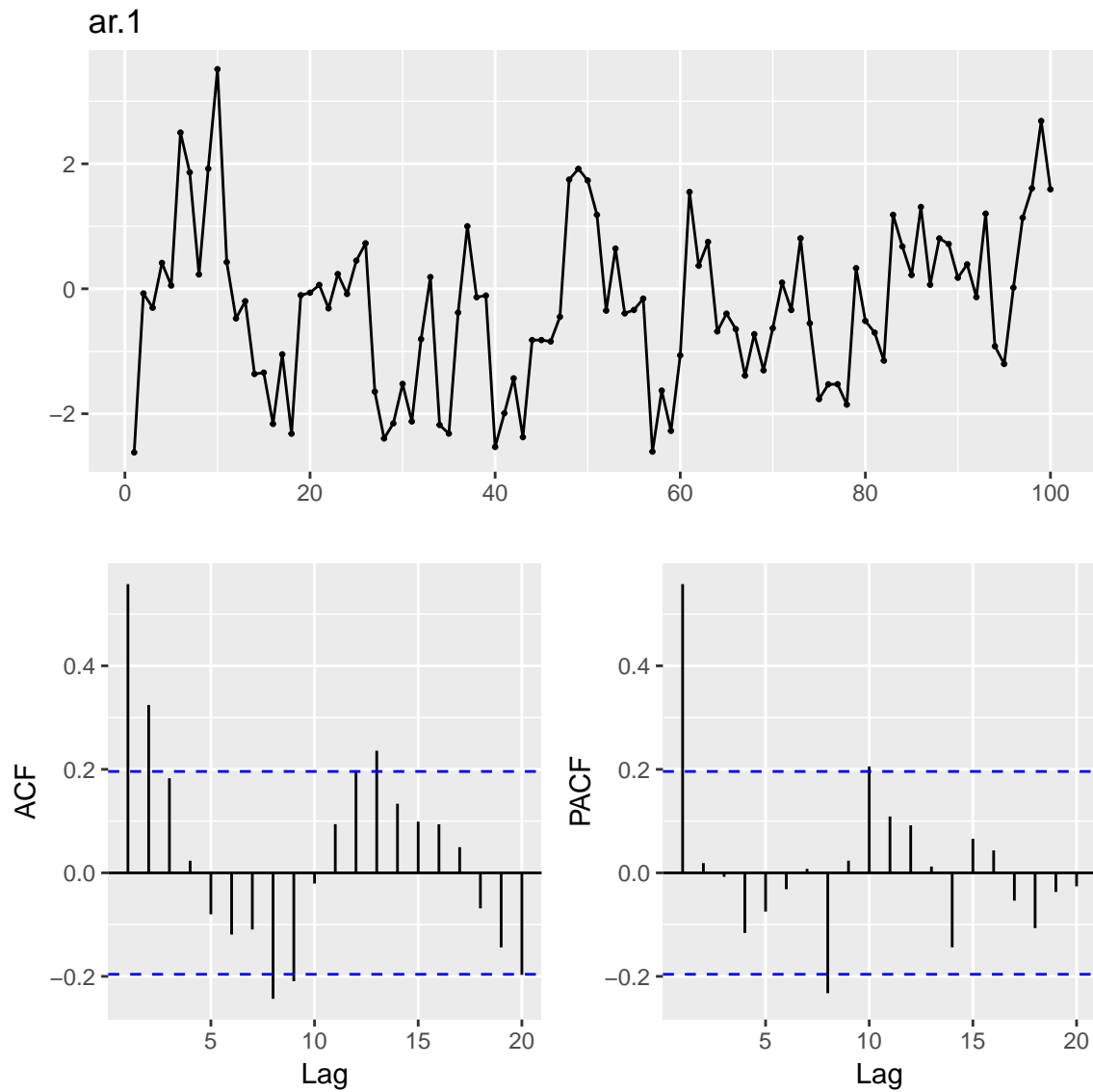
can be simplified to

$$(1 - \frac{1}{3}B)x_t = w_t.$$

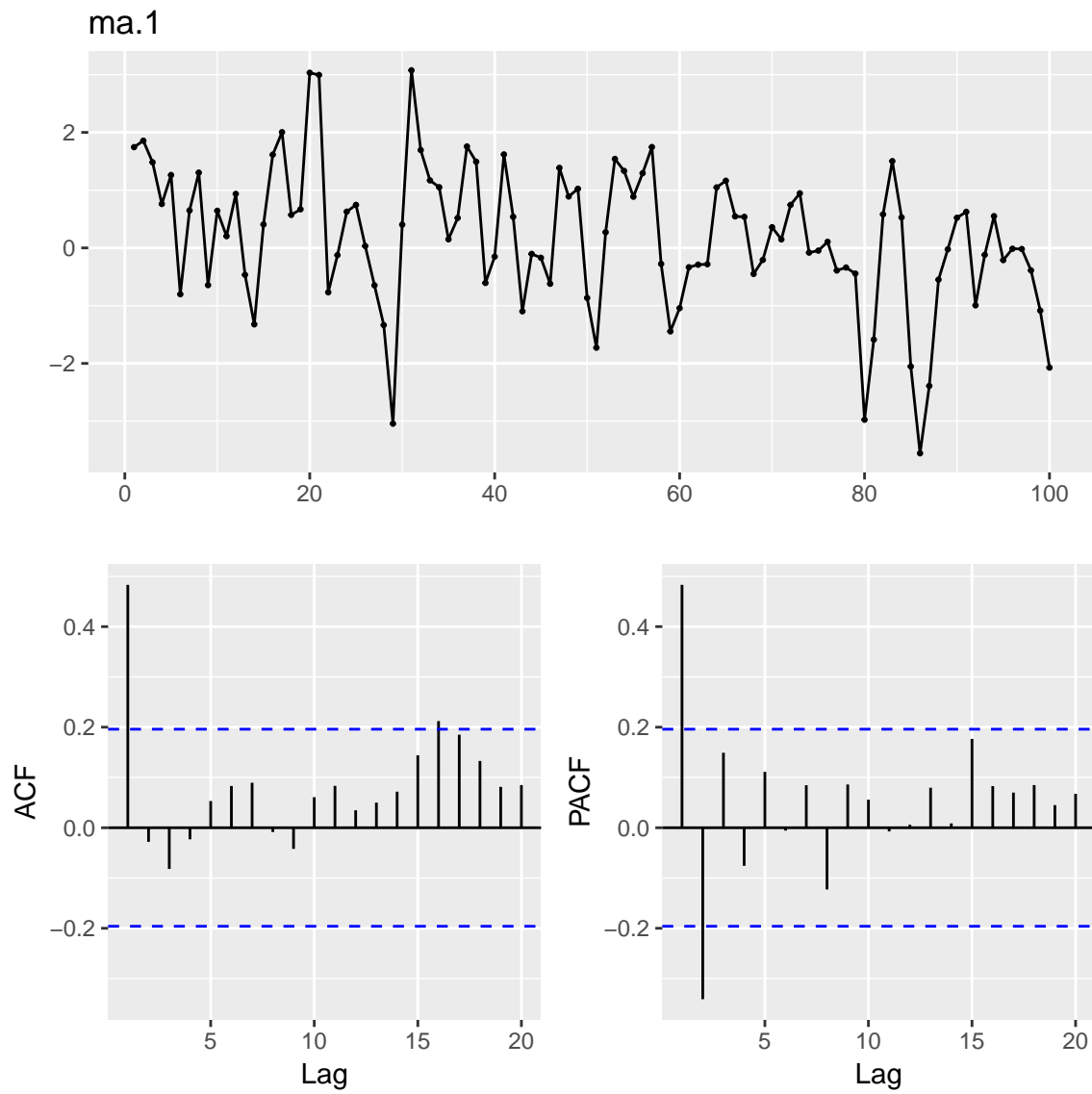
Simulation and Model Fitting

- The R function `arima.sim` allows simulation from ARMA processes.

```
set.seed(10312018)
#install.packages('fpp')
library(fpp)
ar.1 <- arima.sim(n = 100, model = list(ar = .7))
ggtsdisplay(ar.1)
```



```
#install.packages('fpp')
library(fpp)
ma.1 <- arima.sim(n = 100, model = list(ma = .7))
ggtsdisplay(ma.1)
```



- Consider fitting the `arma.1` series that is simulated from an AR(1) process using both an AR model and a MA model. How should we choose?

```
arima(ar.1, order = c(1,0,0))
```

```
##
## Call:
## arima(x = ar.1, order = c(1, 0, 0))
##
## Coefficients:
##      ar1  intercept
## 0.5834   -0.2841
## s.e. 0.0833    0.2505
##
## sigma^2 estimated as 1.12:  log likelihood = -147.76,  aic = 301.51
```

```
arima(ar.1, order = c(0,0,1))
```

```
##
## Call:
## arima(x = ar.1, order = c(0, 0, 1))
##
## Coefficients:
##          ma1  intercept
##          0.5054   -0.2871
## s.e.  0.0815    0.1664
##
## sigma^2 estimated as 1.229:  log likelihood = -152.36,  aic = 310.73
```

```
arima(ma.1, order = c(1,0,0))
```

```
##
## Call:
## arima(x = ma.1, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##          0.5036    0.1840
## s.e.  0.0883    0.2113
##
## sigma^2 estimated as 1.122:  log likelihood = -147.81,  aic = 301.62
```

```
arima(ma.1, order = c(0,0,1))
```

```
##
## Call:
## arima(x = ma.1, order = c(0, 0, 1))
##
## Coefficients:
##          ma1  intercept
##          0.7903    0.1884
## s.e.  0.0655    0.1705
##
## sigma^2 estimated as 0.9151:  log likelihood = -137.95,  aic = 281.9
```

- We have seen the predictive framework, but AIC is often used as well. In fact the `auto.arima` function automatically selects the best model based on AIC.

```
auto.arima(ar.1, max.d = 0)
```

```
## Series: ar.1
## ARIMA(1,0,0) with zero mean
##
## Coefficients:
##          ar1
##          0.6026
## s.e.  0.0818
##
## sigma^2 estimated as 1.145:  log likelihood=-148.37
## AIC=300.74  AICc=300.86  BIC=305.95
```

```
auto.arima(ma.1, max.d = 0)
```

```
## Series: ma.1
## ARIMA(0,0,1) with zero mean
##
## Coefficients:
##          ma1
##          0.7930
## s.e.    0.0646
##
## sigma^2 estimated as 0.9355: log likelihood=-138.55
## AIC=281.11   AICc=281.23   BIC=286.32
```