Study Guide

MSBA 6430: Advanced Issues in Business Analytics

Danny Moncada (monca016)

April 19, 2020

Time Series Analysis

Lecture 2: Stationarity

A. Mean function: $\mu_t = \mathrm{E}(Y_t)$ for any t (any point).

B. Auto-covariance: $\gamma_{t,s} = \mathrm{Cov}(Y_t, Y_s)$ for any t, s (between any two points).

C. Auto-correlation: $\rho_{t,s} = \mathrm{Corr}(Y_t, Y_s)$ for any t, s (between any two points). The correlation of a signal with a delayed copy of itself as a function of delay. Informally, it is the similarity between observations as a function of the time lag between them.

Stationarity:

Why is stationarity so important?

One sample path can infer the stochastic process. Stationarity implies that the process is at an equilibrium. It has a well-defined constant mean level and just jumps around for eternity in the same way (deviating approximately at the same scale, and having the same level of correlation with previous observations.) A process Y_t is stationary if:
1. Mean function $\mathrm{E}(Y_t) = \mathrm{c}$ is constant over time
2. Auto-covariance function $\mathrm{Cov}(Y_t, Y_{t-k}) = \gamma_t$

Stochastic process vs. sample path

1. *Stochastic process* is a sequence of random vairables $\{Y_t|\mathrm{t} = 0, 1, 2, \dots\}$. Y_1 is coin 1, Y_2 is coin 2.

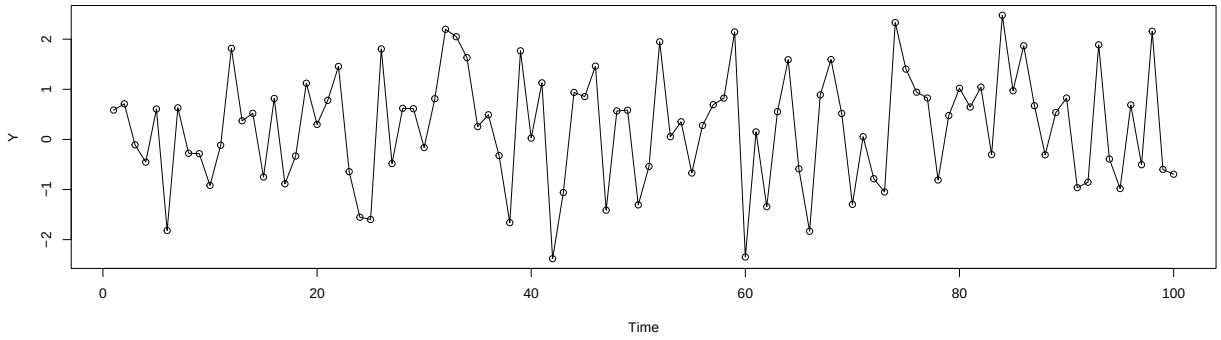
2. *Sample path* of a stochastic process is just ONE sample (realization) of that stochastic process (sequence of random variables). E.g. {H, H, T, H, T, T, ... } result of flipping coins.

3. One stocahstic process can generate many sample paths (infinitely many).

White noise

1. $Y_t = e_t$
2. Called "white noise" because it looks like white light on the spectrometers.
3. Part of variation beyond human's control / understanding.
4. Yes, it is stationary because the mean function $\mathrm{E}(Y_t) = 0$, and the auto-covariance function $\mathrm{Cov}(Y_t, Y_s) = 0$ (every Y_t is independent of every Y_s).

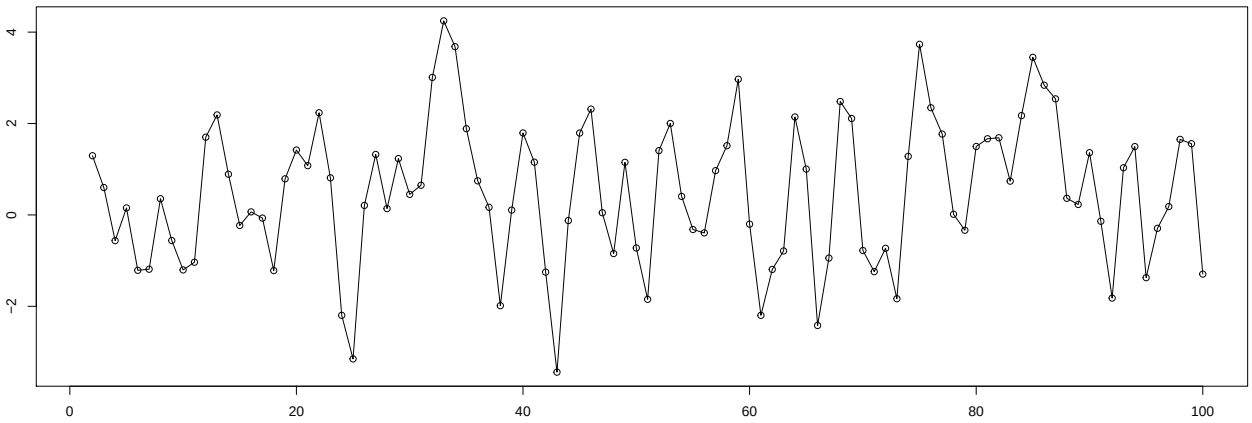
```r
set.seed(12345)
e <- rnorm(100)
Y <- ts(e)

plot(Y, type = 'o')
```



Moving Average (never forget e_t!)

1. $Y_t = e_t + e_{t-1}$
2. Only formulates first order moving dependency but you cannot predict more than n time points ahead.
3. You can do short term forecasting (and why we need AR models for long term).
4. Yes, it is stationary because mean is constant zero and auto correlation is indepedent of t (terms are uncorrelated).

```r
set.seed(12345)
e <- rnorm(100)
Y <- ts((e + zlag(e)))
par(mar=c(2,2,0.2,0.2)); plot(Y, type = 'o')
```

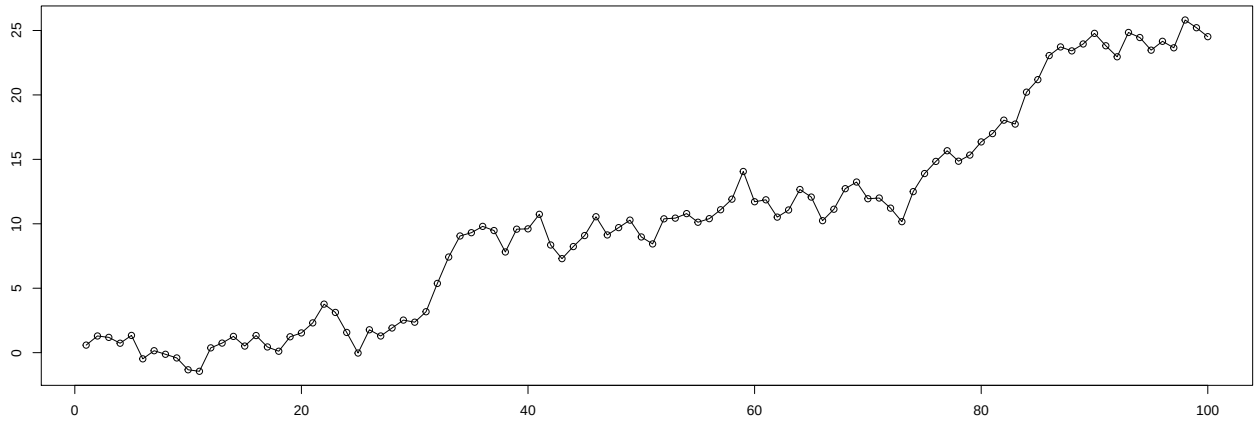


Random Walk

1. $Y_t = e_1 + e_2 + e_3 \ldots + e_k$

2. No, it is **not** stationary; you accumulate new knowledge with no attrition (you never forget). The mean is always the same during random walk (**0**) but the variance increases expontentially, because all past values are being included.

```
set.seed(12345)
e <- rnorm(100)
Y <- ts(cumsum(e))
par(mar=c(2,2,0.2,0.2)); plot(Y, type = 'o')
```



$$\mathrm{Var}(Y_1) = \mathrm{Var}(e_1) = 1$$

$$\mathrm{Var}(Y_2) = \mathrm{Var}(e_1 + e_2) = \mathrm{Var}(e_1) + \mathrm{Var}(e_2) = 2$$

$$\mathrm{Var}(Y_3) = \mathrm{Var}(e_1 + e_2 + e_3) = 3$$

$$\mathrm{Cov}(Y_2, Y_3) = \mathrm{Cov}(e_1+e_2, e_1+e_2+e_3) = \mathrm{Cov}(e_1, e_1)+\mathrm{Cov}(e_2, e_2) = 2$$

Lecture 4: ARMA

MA - Moving Average

1. Y_t is a function of only white noise e_{t-1}, $e_{t-2}\ldots$

MA(1)

1. $Y_t = e_t$ - $\theta_1 e_{t-1}$ (some constant)
2. First-Order MAP
3. Stationarity: Yes.

$$\begin{cases} E[Y_t] = 0 \\ \text{Var}[Y_t] = 1 + \theta_1^2 \\ \text{Cov}[Y_t, Y_{t-1}] = -\theta_1 \\ \text{Cov}[Y_t, Y_{t-2}] = 0 \\ \text{Cov}[Y_t, Y_{t-3}] = 0 \\ \text{Corr}[Y_t, Y_{t-1}] = -\frac{\theta_1}{1+\theta_1^2} \end{cases}$$

Figure 1: Mean, Variance, lag-k auto-covariance for MA(1) process

MA(2)

1. $Y_t = e_t$ - $\theta_1 e_{t-1}$ - $\theta_2 e_{t-2}$ (some constants)
2. Second-Order MAP
3. Stationarity: Yes.

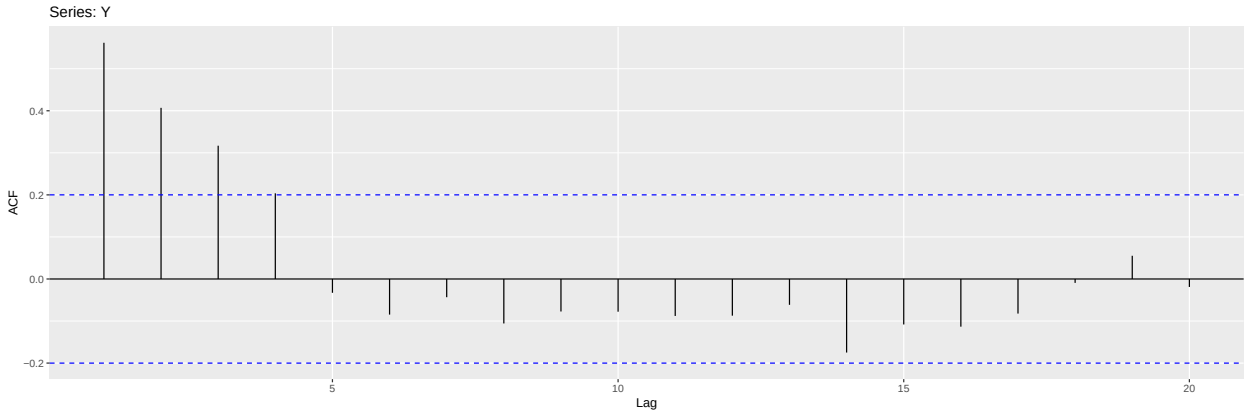
$$\begin{cases} E[Y_t] = 0 \\ \text{Var}[Y_t] = 1 + \theta_1^2 + \theta_2^2 \\ \text{Cov}[Y_t, Y_{t-1}] = -\theta_1 + \theta_1\theta_2 \\ \text{Cov}[Y_t, Y_{t-2}] = -\theta_2 \\ \text{Cov}[Y_t, Y_{t-3}] = 0 \end{cases}$$

Figure 2: Mean, Variance, lag-k auto-covariance for MA(2) process

MA(q)

1. $Y_t = e_t$ - $\theta_1 e_{t-1}$ - $\theta_2 e_{t-2}$-. . .-$\theta_q e_{t-q}$ (some constants)
2. qth-Order MAP
3. Stationarity: Yes.

```
set.seed(12345)
e <- rnorm(100)
Y <- ts(e + 0.5*zlag(e) + 0.5*zlag(e, 2) + 0.5*zlag(e, 3) + 0.5*zlag(e, 4))
ggAcf(Y)
```



AR - Autoregressive model

1. Y_t is a function of its past Y_{t-1}, $Y_{t-2}\ldots$

AR(1)

1. $Y_t = \phi_1 Y_{t-1} + e_t$ (where ϕ_1 is a constant who absolute value is less than 1; if it is equal to 1 then it is a random walk).
2. First-Order ARP
3. Backshift operator: (1 - ϕB)$Y_t = e_t$

$$\begin{cases} E[Y_t] = 0 \\ \mathrm{Var}[Y_t] = \frac{1}{1-\phi^2} \\ \mathrm{Cov}[Y_t, Y_{t-1}] = \frac{\phi}{1-\phi^2} \\ \mathrm{Cov}[Y_t, Y_{t-2}] = \frac{\phi^2}{1-\phi^2} \\ \mathrm{Cov}[Y_t, Y_{t-3}] = \frac{\phi^3}{1-\phi^2} \\ \mathrm{Corr}[Y_t, Y_{t-1}] = \phi \\ \mathrm{Corr}[Y_t, Y_{t-2}] = \phi^2 \\ \mathrm{Corr}[Y_t, Y_{t-3}] = \phi^3 \end{cases}$$

Figure 3: Mean, Variance, lag-k auto-covariance for AR(1) process

AR(2)

1. $Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + e_t$ (where ϕ_1, ϕ_2 are constants).
2. Second-Order ARP
3. Backshift operator:

AR(p)

1. $Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \phi_p Y_{t-p} + e_t$
2. pth-Order ARP
3. Backshift operator:

What is the general definition of ARMA(p, q) (never forget e_t!)?

ARMA(p,q) is Auto-Regressive Moving Average Process of orders p and q.

General formula:

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_p Y_{t-p} + e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2} - ... - \theta_q e_{t-q}$$

Backshift operator:

$$Y_t - \phi_1 Y_{t-1} - \phi_2 Y_{t-2} - ... - \phi_p Y_{t-p} = e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2} - ... - \theta_q e_{t-q}$$

This is why we use "minus" sign in definition of MA. $\Phi(B)Y_t = \Theta(B)e_t$

$$e_t = Y_t - \phi_1 Y_{t-1} - \phi_2 Y_{t-2} - \cdots - \phi_p Y_{t-p} =$$

$$= \underbrace{\left(1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_q B^p\right)}_{\Phi(B)} Y_t = \Phi(B) Y_t$$

Is ARMA(p,q) stationary?

The MA part is ALWAYS stationary. Thus, the stationarity of ARMA(p,q) only relies on AR part, so if AR(p) is stationary, whole process is stationary.

How do you check stationarity of AR(p) or ARMA(p,q) if the model is known? Also, how to draw a conclusion if the roots of the characteristic polynomial is given?

1. We have to check the characteristic polynomial.
2. AR(p) is stationary if $\Phi = 0$ has all roots >= 1 or <= -1. We can check this using *polyroot()*.
3. $Y_t = 0.8Y_{t-1} + 0.6Y_{t-2} + e_t$

Φ(x) = 1 - 0.8x - 0.6x^2

```
polyroot(c(1, -0.8, -0.6))
```

```
## [1]  0.7862996+0i -2.1196330+0i
```

Not stationary, because 0.7 is not greater than 1.

$Y_t = 2.2Y_{t-1}$ - $1.57Y_{t-2} + 0.36Y_{t-3} + e_t$

```r
polyroot(c(1, -2.2, 1.57, -0.36))
```

```
## [1] 1.111111-0i 1.250000+0i 2.000000-0i
```

Stationary, all roots >1.

Why do we need both AR and MA models? (Which one has short/long memory?)

MA models short-term dependence. AR models long-term dependence. ARMA is as *fundamental as linear regression*. It is the **best "linear approximation" to any arbitrary stationary process**.

Lecture 5: SARIMA

Differencing

Why do we need to take differences?

Stationarity is what grants us magic power to discover the full process from *one* sample path, and without this, we cannot say anything about the full process. Real world data is rarely stationary, so we can difference to coerce the process to a stationary one.

Demonstrate that differencing makes these processes stationary.

1. Random walk: $Y_t = Y_{t-1} + e_t$ If we define a new process, $Z_t = Y_t$ - $Y_{t-1} = e_t$, this Z_t is white-noise and stationary (taking a first difference).
2. Stationary time series with linear trend: $Y_t = t + e_t$ is not stationary.

$$Zt = \Delta Y_t = t + e_t - [(t-1) + e_{t-1}] = 1 + e_t - e_{t-1}$$

IS STATIONARY.

$Y_t = t^2 + e_t$ is not stationary.

$$Zt = \Delta Y_t[t^2 + e_t] - [(t-1)^2 + e_{t-1}] = 2t - 1 + e_t - e_{t-1}$$

3. Simple ARIMA (up to ARIMA (1,1,2))

- **Def.** Y_t follows an ARIMA(p,d,q) if
  - $\nabla^d Y_t$ follows an ARMA(p,q)
- For example,
  - If ∇Y_t is ARMA(p,q) then Y_t is ARIMA(p,1,q)
- **Intuition:** Y_t is not stationary and could be something complex but the process $Y_t - Y_{t-1}$ is a nice ARMA model
- In practice, we consider $d = 0$ or 1 most of the time, occassionally $d = 2$

Write the d-th order differencing with backshift operator.

In general, $Y_t = p_d(\text{t}) + e_t$ is stationary after dth-order polynomial, e.g. $p_4(t) = 2.7t^4 + 3.6t^3 + 1.2t^2 + t + 0.5$

Write simple ARIMA (up to ARIMA(1,1,2)) with backshift operator.

- ARIMA(1,1,1) is just

$$\underbrace{(1-\phi_1 B)}_{\text{AR(1)}}\underbrace{(1-B)}_{\text{I(1)}}Y_t = \underbrace{(1-\theta_1 B)}_{\text{MA(1)}}e_t$$

Figure 4: ARIMA(1,1,1) using backshift operator

$$(1-\phi_1 B)(1-B)Y_t = (1-\theta_1 B)e_t - (1-\theta_2 B)^2 e_{t-2}$$

Read order and estimated coefficients of ARIMA(p,1,q) from R, and recover underlying time series model.

NEVER FORGET e_T on any model - free points!

$$Y_t - Y_{t-1} = 0.83(Y_{t-1} - Y_{t-2}) + e_t + 0.71 \cdot e_{t-1} - 0.22 \cdot e_{t-2})$$

Seasonality

What does each order of SARIMA(p,d,q)(P,D,Q)[S] mean? (p is degree of non-seaonsal AR part, Q is the degree of the seasonal MA part)

- Non-seasonal
  - p: degree of non-seasonal AR part
  - d: degree of differencing
  - q: degree of non-seasonal MA part
- Seasonal
  - P: degree of seasonal AR part
  - D: degree of seasonal differencing
  - Q: degree of seasonal MA part
- S: period

Multiplicative Seasonal ARIMA

- **Def.** Y_t is a *multiplicative seasonal ARIMA* of orders $(1,1,1) \times (1,1,1)$ with period S if it could be expressed as

$$\underbrace{(1-B)(1-\phi_1 B)}_{\text{non-seasonal ARI(1,1)}} \underbrace{(1-B^S)(1-\Phi_1 B^S)}_{\text{seasonal ARI(1,1)}} Y_t =$$

$$= \underbrace{\alpha_0}_{\text{drift}} + \underbrace{(1-\theta_1 B)}_{\text{non-seasonal MA(1)}} \cdot \underbrace{(1-\Theta_1 B^S)}_{\text{seasonal MA(1)}} e_t$$

- **Def.** Y_t is a *multiplicative seasonal ARIMA* of orders $(p,d,q) \times (P,D,Q)$ with period S is defined by extending the above

Figure 5: Multiplicative Seasonal ARIMA

Write Seasonal AR(p) and Seasonal MA(q)

1. Seasonal AR(p) or SAR(1)[12]

$$Y_t = \phi Y_{t-12} + \phi Y_{t-24} + ... + \phi Y_{t-p} + e_t$$

2. Seasonal MA(q) or SMA(1)[12]

$$Y_t = e_t - \theta_1 e_{t-12} - \theta_2 e_{t-24} - ... - \theta_q e_{t-q}$$

Read the order and estimated coefficients of SARIMA (up to SARIMA(1,1,1)(1,1,1)[S] from R)

```
# This example is a *multiplicative* seasonal ARIMA!!
set.seed(1234)
data(co2)
arima_est <- auto.arima(co2)
arima_est
```

```
## Series: co2
## ARIMA(1,0,1)(0,1,1)[12] with drift
##
## Coefficients:
##          ar1      ma1     sma1   drift
##       0.8349  -0.4630  -0.8487  0.1520
## s.e.  0.0819   0.1246   0.1274  0.0052
##
## sigma^2 estimated as 0.5288:  log likelihood=-136.09
## AIC=282.18   AICc=282.7   BIC=296.11
```

Full functional form (rounded to two digits):

$$(1 - B^{12})(1 - 0.83B)Y_t = 0.15 + (1 - 0.46B)(1 - 0.85B^{12})e_t$$

where $Var[e_t] = 0.5288$.

Lecture 6: Model Selection

Model Selection Tools:

- To distinguish AR or MA from ARMA:
  - ACF determines q for MA (already seen in the previous class)
  - PACF determines p for AR
  - Both reliable and theoretically sound
- To determine (p, q) in ARMA:
  - EACF (practical, not 100% reliable)
- To distinguish ARMA from ARIMA:
  - ADF test (practical, not 100% reliable)
- To compare a few arbitrary models
  - AIC, BIC (practical, and reliable)
- At the end of the day, we can still have *auto.arima()* as a no-brainer
  - Fast, easy, but not optimal

Figure 6: Model Selection

Unit Roots:

How to find unit roots?

- How should we test for presence of unit roots?
- One way we could do that is to create a statistical test on ϕ:

$$Y_t = \phi Y_{t-1} + e_t$$

and verify if $\phi = 1$ or not.

- Null hypothesis: $H_0 : \phi = 1$
  - (that is, H_0 assumes the process has a unit root!)
- Alternative hypothesis: $H_1 : \phi < 1$

- There is a big problem however.
  - Both Y_t and Y_{t-1} are non-stationary if H_0 is true.
  - Can't use linear regression to test it: *spurious regressions* may result

Figure 7: Unit Roots

What is the test to identify random walk against stationary time series?

The ADF (Augmented Dickey-Fuller) test.

The intuition behind ADF is:

- if the process has unit roots then its past value Y_{t-1} can't predict the change (it behaves like a random walk) so $(\phi - 1) = 0$.
- if the process is stationary then $(\phi - 1) < 0$ since *stationary processes revert to the mean!*
  - if Y_{t-1} is large positive then ∇Y_t is more likely to be negative
  - if Y_{t-1} is large negative then ∇Y_t is more likely to be positive

Figure 8: ADF Explained

How do you read R results?

1. Null hypothesis: not stationary
2. Alternative hypothesis: stationary
3. p-value < 0.05, reject the null hypothesis and conclude process is stationary.

```
#      Augmented Dickey-Fuller Test

# data: y
# Dickey-Fuller = -2.4591, Lag order = 21, p-value = 0.3839
# alternative hypothesis: stationary

# The p-value is not low enough to reject the null hypothesis, so we cannot conclude that
# the process is stationary.  We would have to conclude that this process is a random walk.
```

```
set.seed(12345)
rand_walk <- cumsum(rnorm(100))
adf.test(rand_walk)
```

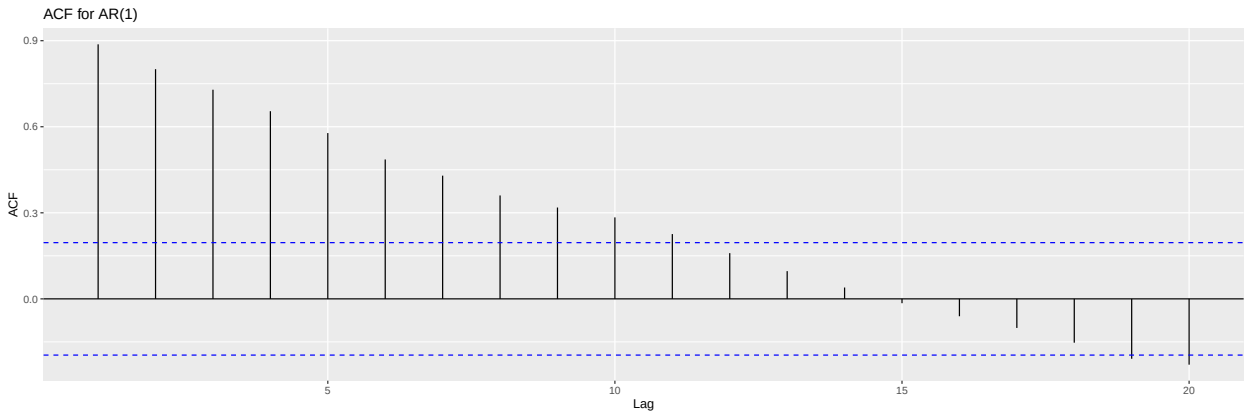
```
##
##  Augmented Dickey-Fuller Test
##
## data:  rand_walk
## Dickey-Fuller = -2.3527, Lag order = 4, p-value = 0.4305
## alternative hypothesis: stationary
```

p value is high so can't reject the null hypothesis H_0 in favor of H_1 + H0: unit root + H1: stationary

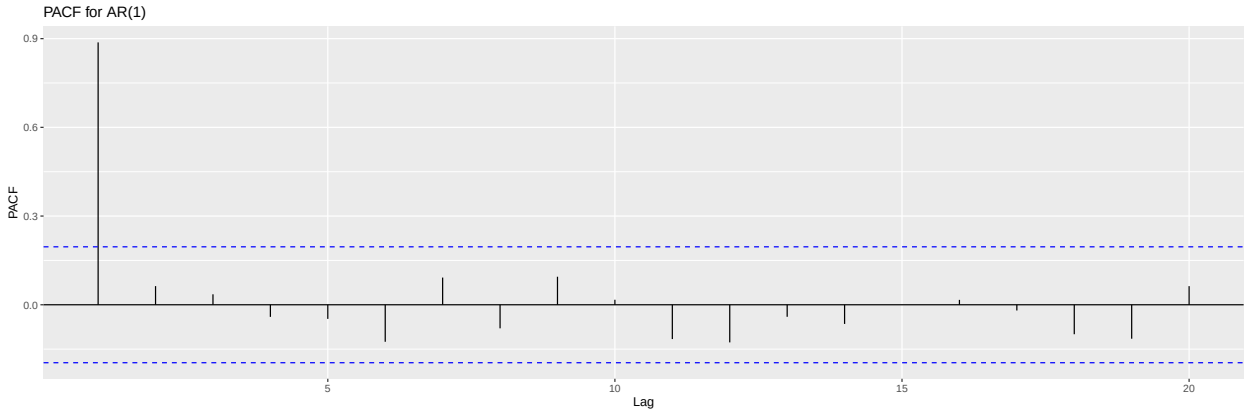
Determine orders using ACF and PACF plots

AR(p):

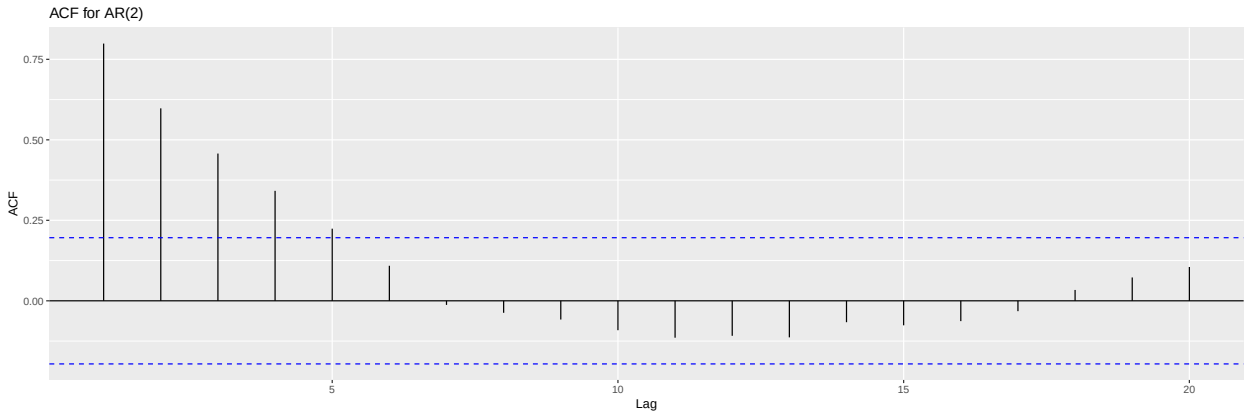
```
set.seed(12345)
Y = arima.sim(model=list(ar=0.9), sd=1, n = 100)
Y2 = arima.sim(model=list(ar=0.9,0.5), sd=1, n=100)

ggAcf(Y) + labs(title = "ACF for AR(1)")
```



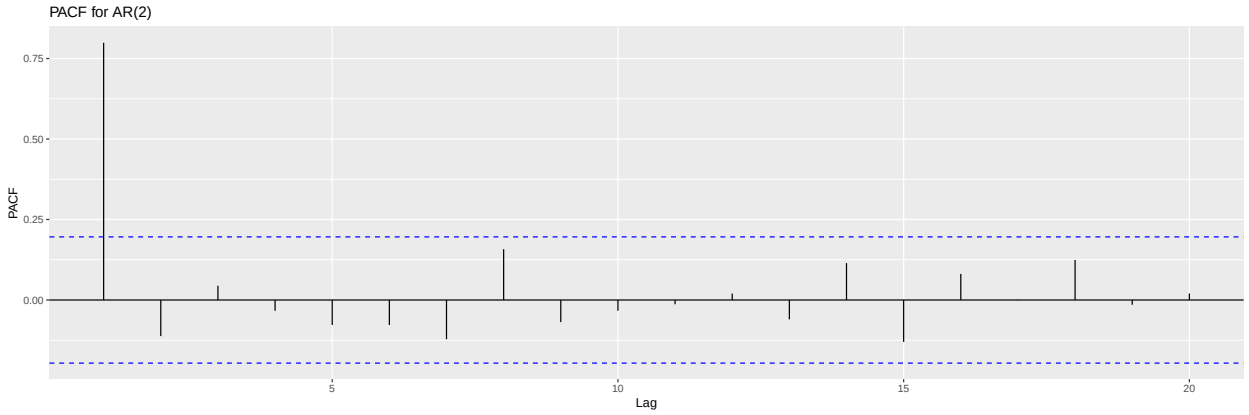
```r
ggPacf(Y) + labs(title = "PACF for AR(1)")
```




```r
ggAcf(Y2) + labs(title = "ACF for AR(2)")
```



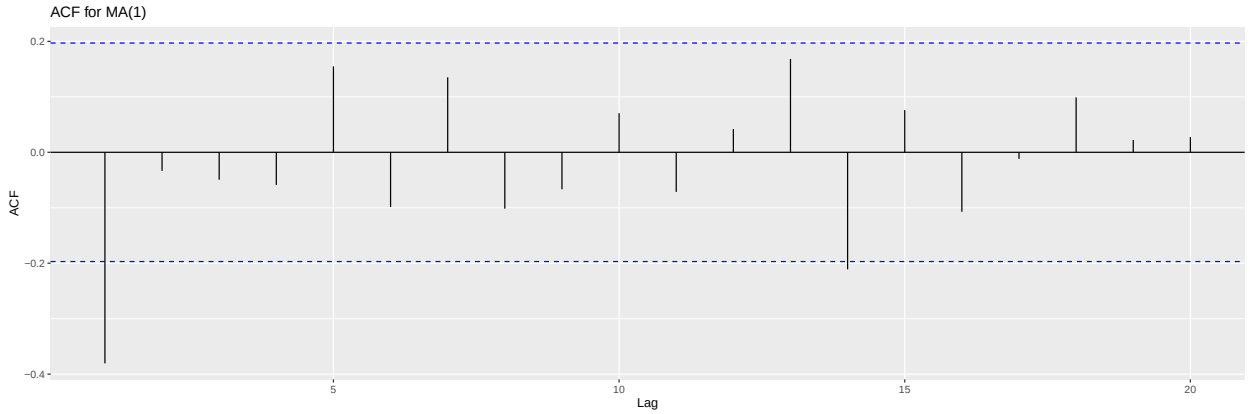

```r
ggPacf(Y2) + labs(title = "PACF for AR(2)")
```



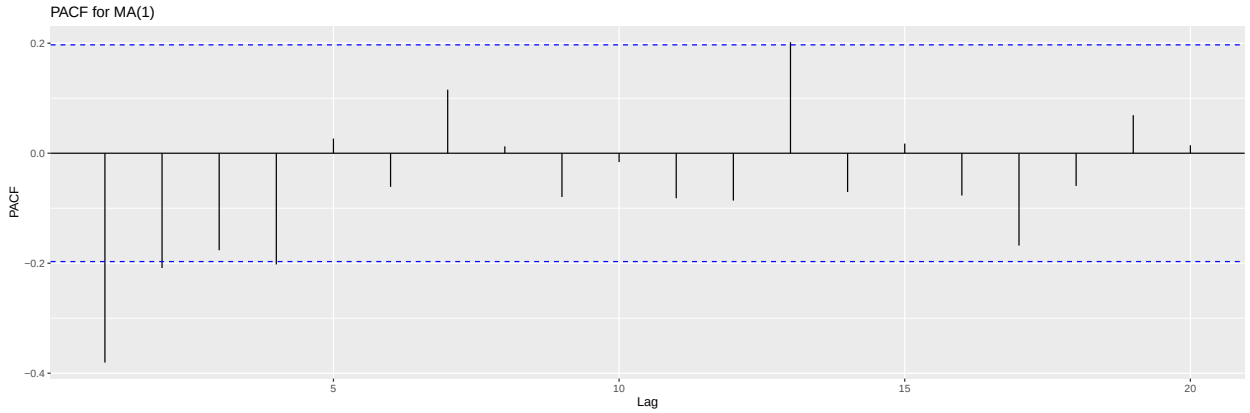
MA(q):

```r
set.seed(12345)
e <- rnorm(100)
```

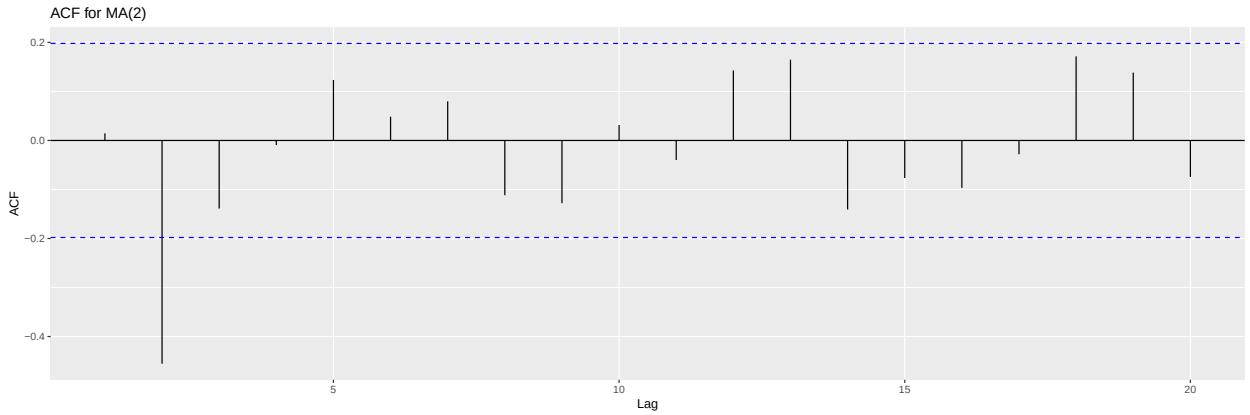
```r
Y <- ts((e - 0.5*zlag(e)))
Y2 <- ts((e - 0.5*zlag(e) - 0.9*zlag(e, 2)))

ggAcf(Y) + labs(title = "ACF for MA(1)")
```



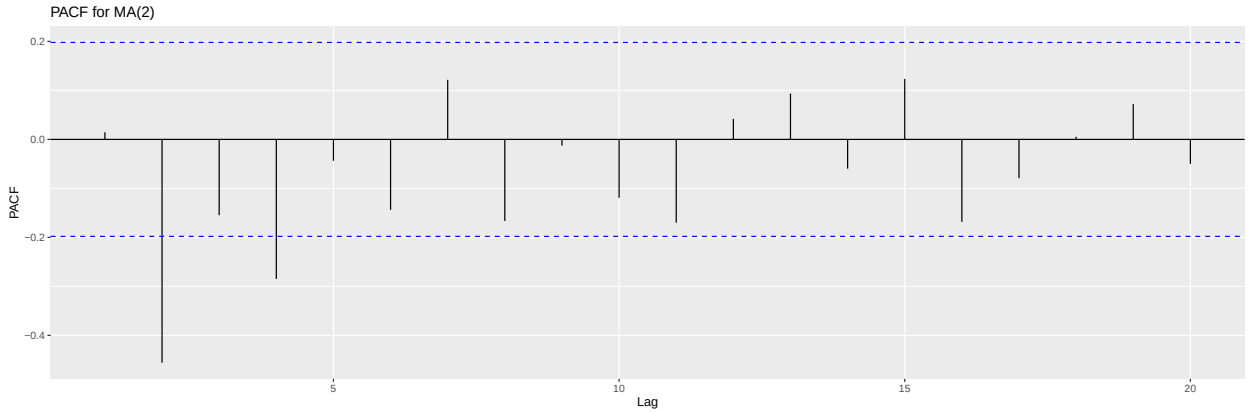

```r
ggPacf(Y) + labs(title = "PACF for MA(1)")
```




```r
ggAcf(Y2) + labs(title = "ACF for MA(2)")
```



```
ggPacf(Y2) + labs(title = "PACF for MA(2)")
```



How do you read ACF/PACF plots? What does the value (height of black bar) at each time lag mean?

Can we determine the order of ARMA(p,q) by ACF and PACF?

ACF and PACF for ARMA models

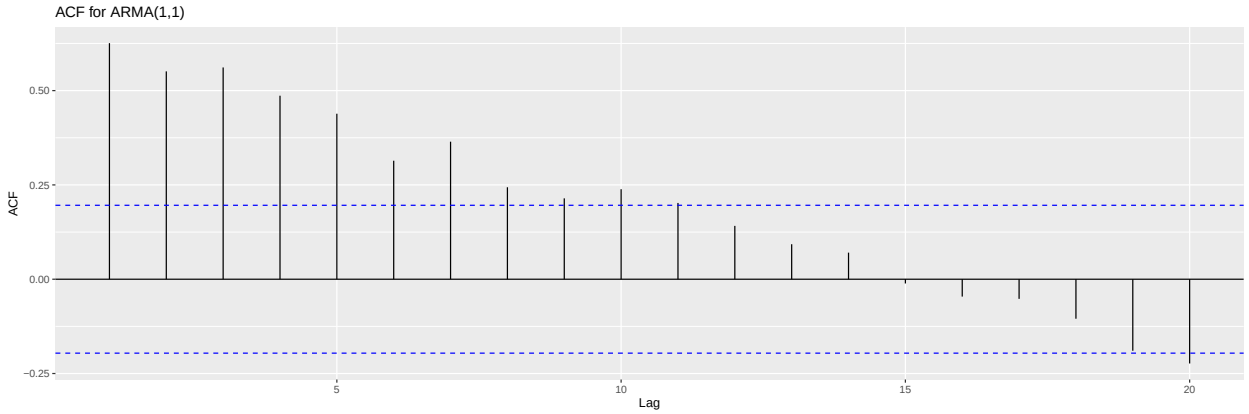
Table 1: ACF and PACF for ARMA models

Function	AR(p)	MA(q)	ARMA(p, q)
ACF	Tails off	Cuts at lag q	Tails off
PACF	Cuts at lag p	Tails off	Tails off

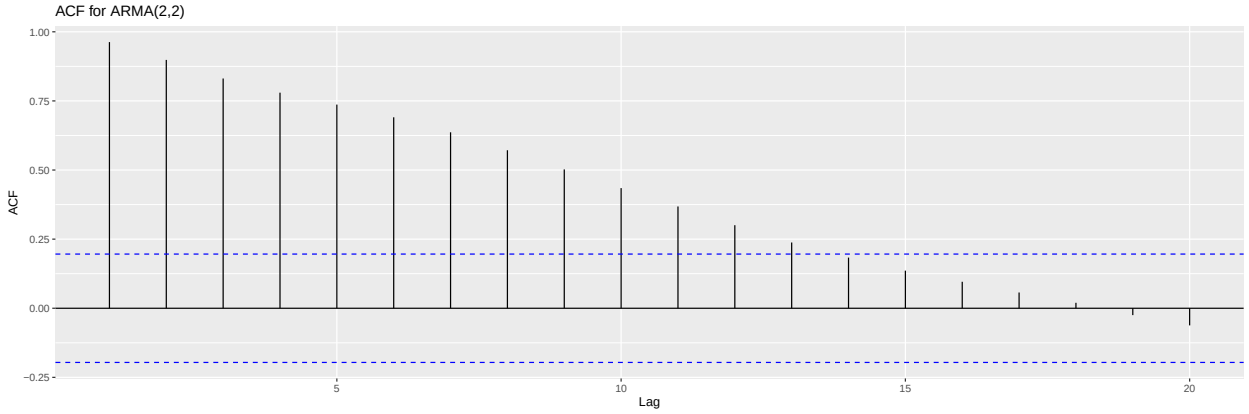
- Ok, we can use the above to find the order of AR or MA if we have one
- How do we find the order if it is ARMA?

Figure 9: ACF and PACF for ARMA Models

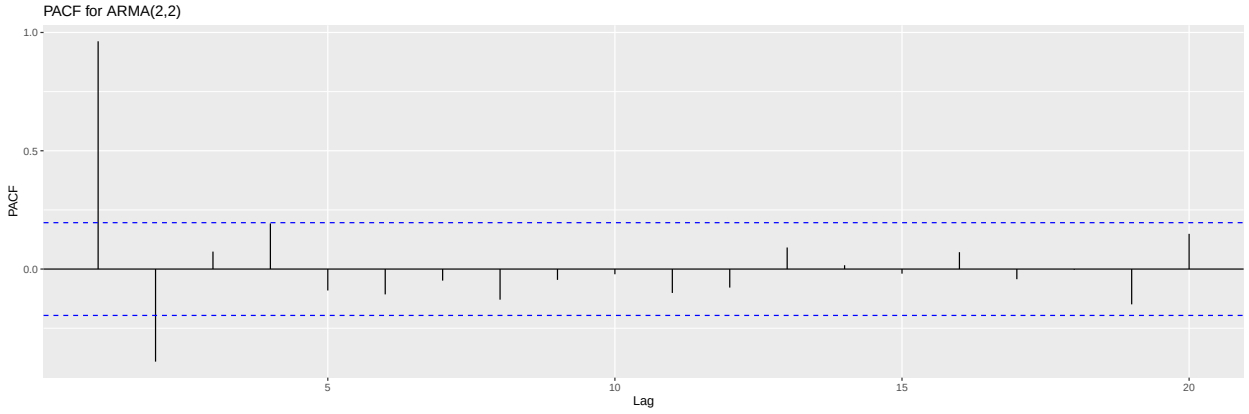
```r
set.seed(12345)
Y = arima.sim(model=list(ar=0.9, ma=-0.5), sd=1, n = 100)
ggAcf(Y) + labs(title = "ACF for ARMA(1,1)")
```




```r
set.seed(88)
Y = arima.sim(model=list(order=c(2,0,2),ar=c(0.5,0.45), ma=c(0.8,0.5)), sd=1, n = 100)
ggAcf(Y) + labs(title = "ACF for ARMA(2,2)")
```




```r
ggPacf(Y) + labs(title = "PACF for ARMA(2,2)")
```



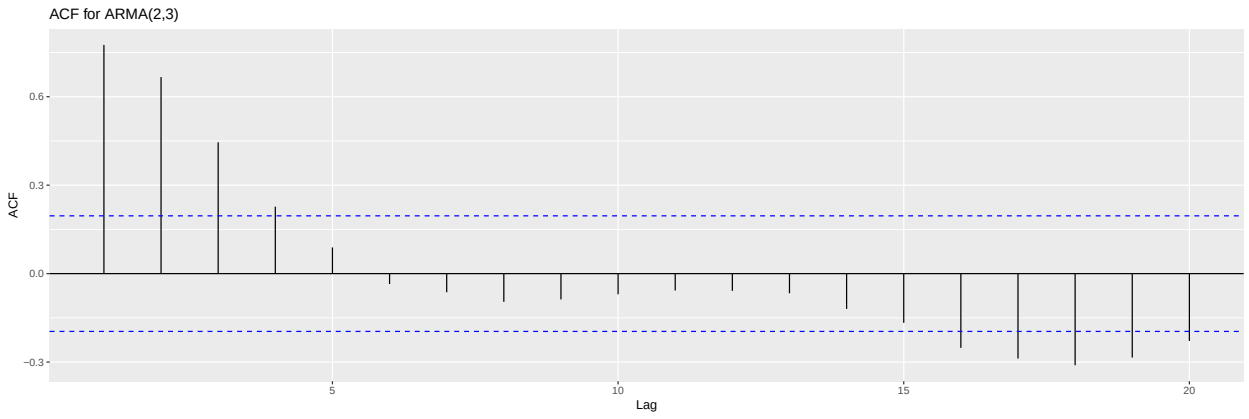
How to read EACF result?

Extended Auto-Correlation (EACF)

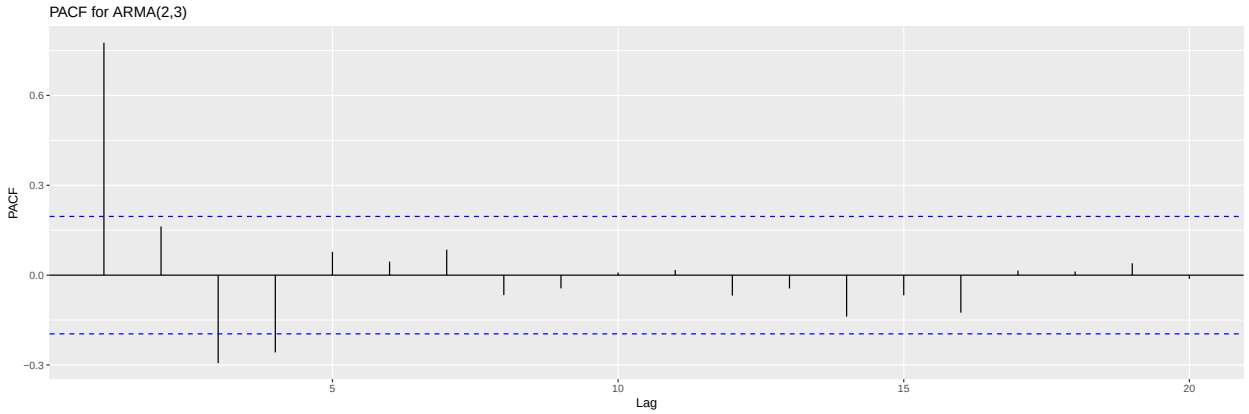
- Consider ARMA(p,q) with unknown p and q
- The task:
  - you need to find orders p and q
- The intuition for solution:
  - assume you know p already, then you can remove AR-component and the rest is just an MA(q) that should look nice on ACF
  - if your candidate p is too small then the rest is **not** an MA(q) and would not look nice on ACF
  - so just try different p and see where it gets you

Figure 10: EACF

```
set.seed(88)
Y = arima.sim(model=list(order=c(2,0,3),ar=c(0.3,0.2), ma=c(0.2,0.3, 0.2)), sd=1, n = 100)
ggAcf(Y) + labs(title = "ACF for ARMA(2,3)")
```




```
ggPacf(Y) + labs(title = "PACF for ARMA(2,3)")
```




```
eacf(Y)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x o o o o o o o  o  o  o
## 1 x x x o o o o o o o o  o  o  o
## 2 x x o o o o o o o o o  o  o  o
## 3 x x o o o o o o o o o  o  o  o
## 4 x o x o o o o o o o o  o  o  o
## 5 x o x o x o o o o o o  o  o  o
## 6 x o x o o o o o o o o  o  o  o
## 7 x x x o o o o o o o o  o  o  o
```

Pick the first 'o' in the upper right corner.

Akaike Information Criterion (AiC)

$$AIC = -2log(maxlikelihood) + 2k$$

Bayesian Information Criterion (BiC)

$$BIC = -2log(maxlikelihood) + klog(n)$$

+ BiC has the bigger penalty in general. + BIC penalizes the extra model parameters more heavily than AIC. + as n approaches infinity, BIC would tend to converge to a specific model in the set of models; AIC wouldn't – it may pick more and more complex models

How to compare different models by using AiC or BiC?

The model that has the *lowest* is generally the best model.

```
#AR (2) : Likelihood = 100, k = 2
#ARMA (2,2): Likelihood = 100, k = 4

#AIC ( AR(2) ) = -2 log100 + 2 * 2 = C + 4
#AIC ( AR(4) ) = -2 * log100 + 2 * 4 = C + 8
```

```
#AIC ( AR(2) ) = -2 * 99 + 2 * 2 = -195
#AIC ( ARMA(2, 2) ) = -2 * 100 + 2 * 4 = -192
```

Box-Jenkings Approach to Time Series

1. Determine stationarity and seasonality
   - Stationary should show constant mean and scale of deviation
   - Seasonality could be seen from ACF and domain knowledge
2. If non-stationary, use first-differences ∇ to achieve stationarity
3. Identify p and q of ARMA(p, q)
   - Use ACF, PACF, EACF as above
4. Estimate model parameters
   - Use `Arima` or `auto.arima`
5. Check the model fit
   - Residuals
6. Forecasting – our next step...

Figure 11: Box-Jenkings Approach to Time Series

Lecture 7: Forecasting

Simple Exponential Smoothing

$$\hat{Y}_{t+1} = \alpha Y_t + \alpha(1-a)Y_{t-1} + \alpha(1-a)^2 Y_{t-2} + ...$$

where

$$0 \leq x \leq 1$$

SES Smoothing:

$$(1-B)Y_t = (1 - B + \alpha B)e_t = (1 - \theta B)e_t$$

which is clearly describing an ARIMA(0,1,1).

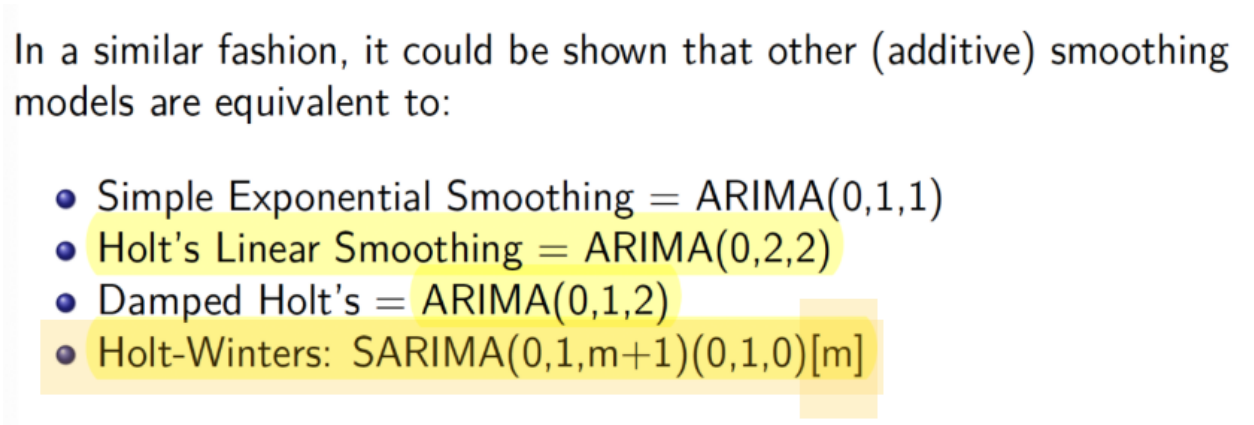


Figure 12: Other Smoothing Methods

Given Y_1, Y2, ... Yt, how to forecast Y_{t+1}, Y_{t+2}, ..., Y_{t+h} with AR(p)? p<=2, h<=3

General form ($e_t = 0$, non-zero mean):

$$\hat{Y}_{101} - intercept = coeff \cdot (Y_{100} - intercept) + coeff \cdot (Y_{99} - intercept)$$

Given Y_1, Y2, ... Yt, how to forecast Y_{t+1}, Y_{t+2}, ..., Y_{t+h} with MA(q)? q<=2, h<=3

General form ($e_t = 0$, non-zero mean):

$$\hat{Y}_{101} = coeff \cdot \hat{e}_{100}$$

$$\hat{Y}_{101} - intercept = coeff \cdot (Y_{100} - \hat{Y}_{100})$$

Given Y_1, Y2, ... Yt, how to forecast Y_{t+1}, Y_{t+2}, ..., Y_{t+h} with ARMA(p,q)? p<=2, q<=2, h<=3.

General form ($e_t = 0$), non-zero mean:

$$\hat{Y}_{101} - intercept = coeff \cdot (Y_{100} - intercept) + coeff \cdot (Y_{99} - intercept) - coeff \cdot e_{t-1}$$

$$\hat{Y}_{102} - intercept = coeff \cdot (\hat{Y}_{101} - intercept) + coeff \cdot (Y_{100} - intercept)$$

AR(1) Forecasting

- $\hat{Y}_{T+1} = \hat{\phi} Y_T$
- Compare with the (hypothetically) true value $Y_{T+1} = \phi Y_T + e_{T+1}$
- Why don't we have e_{T+1}?
- e_{T+1} is white noise: error beyond human understanding.
- We substitute e_{T+1} with its mean $E(e_{T+1}) = 0$

Figure 13: AR(1) Forecasting

Forecasting with non-zero mean

- In the above case, $T = 100$, so

$$\hat{Y}_{101} = \hat{\mu} + \hat{\phi}(\hat{Y}_T - \hat{\mu}) = -0.0701 + 0.7623(\hat{Y}_{100} + 0.0701)$$

```
> model1

Call:
arima(x = Y, order = c(1, 0, 0), method = "ML")

Coefficients:
         ar1  intercept
      0.7623    -0.0701
s.e.  0.0627     0.2117

sigma^2 estimated as 0.2695:  log likelihood = -76.76,  aic = 157.52
> mu=model1$coef[2]       #intercept
> phi=model1$coef[1]
> Y101=mu+phi*(Y[100]-mu) #\hat{Y}_{T+1}=\mu+\phi*(Y_T-\mu)
> as.numeric(Y101)
[1] -0.2502956
> Yhat=predict(model1,n.ahead=1)$pred
> as.numeric(Yhat)
[1] -0.2502956
```

- Here Yhat is $\hat{Y}_{101}$ provided by *arima predict*, while Y101 is $\hat{Y}_{101}$ calculated by hand

Figure 14: AR(1) with Non Zero Mean Forecasting

AR(2) Forecasting

- $\hat{Y}_{T+1} = \hat{\phi}_1 Y_T + \hat{\phi}_2 Y_{T-1}$
- $\hat{Y}_{T+2} = \hat{\phi}_1 \hat{Y}_{T+1} + \hat{\phi}_2 Y_T$
- $\hat{Y}_{T+3} = \hat{\phi}_1 \hat{Y}_{T+2} + \hat{\phi}_2 \hat{Y}_{T+1}$

Figure 15: AR(2) Forecasting

MA(1) Forecasting

- The same, we replace e_T by $\hat{e}_T$
- Where do we get $\hat{e}_T$?
- $\hat{e}_T = Y_T - \hat{Y}_T$, the estimation residual (provided by *arima*)
- That is,

$$\hat{Y}_{T+1} = -\hat{\theta}(Y_T - \hat{Y}_T)$$

We substitue e_{t+1} with its means $\mathrm{E}(e_{t+1}) = 0$

MA(1) Forecasting

- Then what about $\hat{Y}_{T+2}$?
- $\hat{Y}_{T+2} = 0$. Surprising?!
- Two ways to explain it:
- First, $\hat{Y}_{T+2} = -\hat{\theta}(Y_{T+1} - \hat{Y}_{T+1})$, since we don't have Y_{T+1}, we estimate it by $\hat{Y}_{T+1}$
- Second, $Y_{T+2} = e_{T+2} - \theta e_{T+1}$. Both e_{T+2} and e_{T+1} are not observed and hence are replaced by 0

Because MA(1) is a short order model and only carries one term forward the covariance of Y100 and Y102 is 0, which leads to Y102 being 0.

MA(1) Forecasting with non-zero mean

- Similar to AR(p)
- With mean $\hat{\mu}$ we have,

$$\hat{Y}_{T+1} - \hat{\mu} = -\hat{\theta}[(Y_T - \hat{\mu}) - (\hat{Y}_T - \hat{\mu})]$$

- Hence

$$\hat{Y}_{T+1} = \hat{\mu} - \hat{\theta}(Y_T - \hat{Y}_T)$$

- $\hat{Y}_{T+h} = \hat{\mu}$, $h \geq 2$

Subtract the means, just like the AR model, so it comes down to 0 by subtracting mean.

MA(q) Forecasting

- For MA(2) with zero mean,

$$Y_{T+3} = e_{T+3} - \theta_1 e_{T+2} - \theta_2 e_{T+1}$$

  hence $\hat{Y}_{T+3} = 0$

- For MA(q) with zero mean,

$$Y_{T+q+1} = e_{T+q+1} - \theta_1 e_{T+q} - \cdots - \theta_q e_{T+1}$$

  hence $\hat{Y}_{T+q+1} = 0$

- **A strong conclusion:** MA(q) cannot forecast more than q points ahead!

Figure 16: MA(q) Forecasting

ARMA(p,q) Forecasting

- For ARMA(p,q), any idea?

$$\begin{aligned} Y_t &= (\phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p}) + e_t \\ &\quad -(\theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots \theta_q e_{t-q}) \end{aligned}$$

- ϕ_i replaced by $\hat{\phi}_i$, $i = 1, \ldots, p$
- θ_j replaced by $\hat{\theta}_j$, $j = 1 \ldots, q$
- e_t replaced by 0
- use Y_{t-i} or replace it by $\hat{Y}_{t-i}$ depending on the availability of Y_{t-i}, as in AR(p)
- replace e_{t-j} by $\hat{e}_{t-j} = Y_{t-j} - \hat{Y}_{t-j}$ or 0 depending on the availability of Y_{t-j}, as in MA(q)

Figure 17: ARMA(p,q) Forecasting

What is a prediction interval? How do you interpret the result?

Users can have an expectation (of lower bound and upper bound) ahead of time. In a business setting, you want to be able to plan for best and worst case scenarios - so there is some flexibility in your planning and you can pivot accordingly.

Commonly used prediction intervals

- 95% prediction interval

$$\hat{Y}_t \pm 1.96\hat{\sigma}_t$$

where $\pm$ 1.96 is the 2.5% and 97.5% quantile of N(0,1).

- 80% prediction interval

$$\hat{Y}_t \pm 1.28\hat{\sigma}_t$$

where $\pm$ 1.28 is the 10% and 90% quantile of N(0,1)

$\hat{\sigma}_t$ is the estimated standard deviation at t.

- In general, if you need a $(1-\alpha)\times 100\%$ prediction interval
- You check the quantile table of N(0,1), looking for

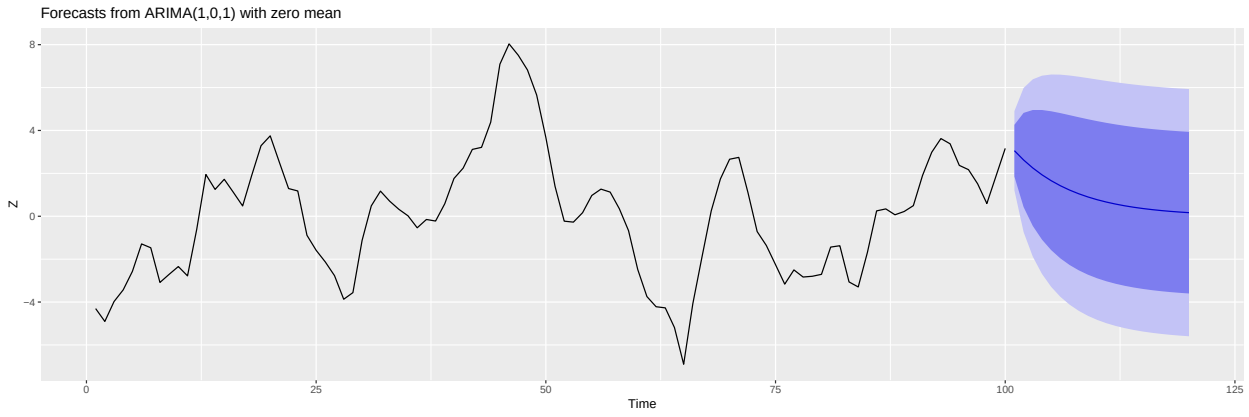
$$\alpha/2 \text{ or } 1-\alpha/2$$

- e.g., $\{95\%\text{ interval}\} \Rightarrow \{\alpha = 0.05\} \Rightarrow \{1-\alpha/2 = 0.975\}$

```
> qnorm(0.975)
[1] 1.959964
> qnorm(0.9)
[1] 1.281552
```

Figure 18: Quantile

```
set.seed(88)
Z = arima.sim(model=list(order=c(1,0,1),ar=0.8, ma=0.6), sd=1, n = 100)
model3 = auto.arima(Z,d=0,seasonal=F)
autoplot(forecast(model3, h=20))
```



Light blue is 80% c.i., dark blue is 90%.

What is an ARMAX model?

An ARMAX model is ARMA (or ARIMA) with covariates.

$$Y_t = \beta X_t + (\phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_p Y_{t-p}) + e_t - (\theta_1 e_{t-1} + \theta_2 e_{t-2} + ... + \theta_q e_{t-q})$$

Where X_t is covariate at time t and β is its coefficent. X_t (S&P 500) could be another time series like Y_t (Dow Jones). X_t can also be "static" variable, like gender, holiday/season indicator, location.

Is B (beta) the effect on Y_t as it is in linear regression?

No, the actual actual effect is $\beta/\Theta(\mathrm{B})$ rather than β.

- Second, β is not the effect on Y_t when the X_t is increased by 1 (as it is in linear regression)
- To see that, we re-write the model with backshift operators:

$$\Phi(B)Y_t = \beta X_t + \Theta(B)e_t$$

- Then

$$Y_t = \frac{\beta}{\Phi(B)}X_t + \frac{\Theta(B)}{\Phi(B)}e_t$$

The actual effect is $\beta/\Phi(B)$ rather than β

Solution: Fit regression ($Y_t \sim \beta X_t$) first, so that β has regular interpretation.

Fit residual ($Y_t \sim \hat{\beta} X_t$) with ARIMA models.

Lecture 8: Trend

Definitions:

1. Constant trend:

$$Y_t = \mu + Z_t$$

where μ does not depend on t, $E(Z_t) = 0$ and Z_t is stationary.

- Yes, it is stationary! $E(Y_t) = \mu$
- For example,
- AR(1) with μ: $Y_t - \mu = \phi(Y_{t-1} - \mu) + e_t$
- MA(1) with μ: $Y_t - \mu = e_t - \theta e_{t-1}$

Figure 19: Stationary?

- Then how to estimate μ?
- One approach is to use the average:

$$\hat{\mu} = \frac{1}{n}\sum_{t=1}^{n} Y_t$$

- **Note**: This is a *time average* over one sample path!

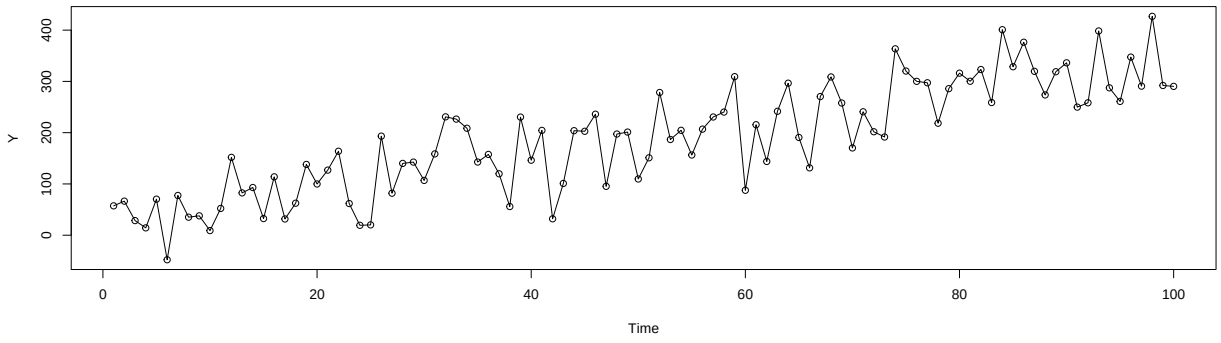
All the data in a time series is highly correlated, so random slection from the data will lead to HIGH bias.

2. Linear trend:

$$Y_t = \beta_0 + \beta_1 t + Z_t$$

This is not stationary; Z_t is 0 but the mean is increasing every few steps.

```
set.seed(12345);
e <- rnorm(100, 0, 50)
t <- 1:100
Y <- ts(25.0 + 3.0*t + e)
plot(Y, type = 'o')
```



3. Seasonal trend:

$$Y_t = \mu + Z_t$$

where $\mu = \mu_{t-k}$;

Seasonal trend in monthly data:

$$\mu_1 = \mu_1 3 = \mu_2 5 = ...$$
$$\mu_2 = \mu_1 4 = \mu_2 6 = ...$$

This is not stationary. Seasonality implies consistent differences in the means of sample paths at different times: $E(Y_{Jan2018}) \neq E(Y_{June2018})$

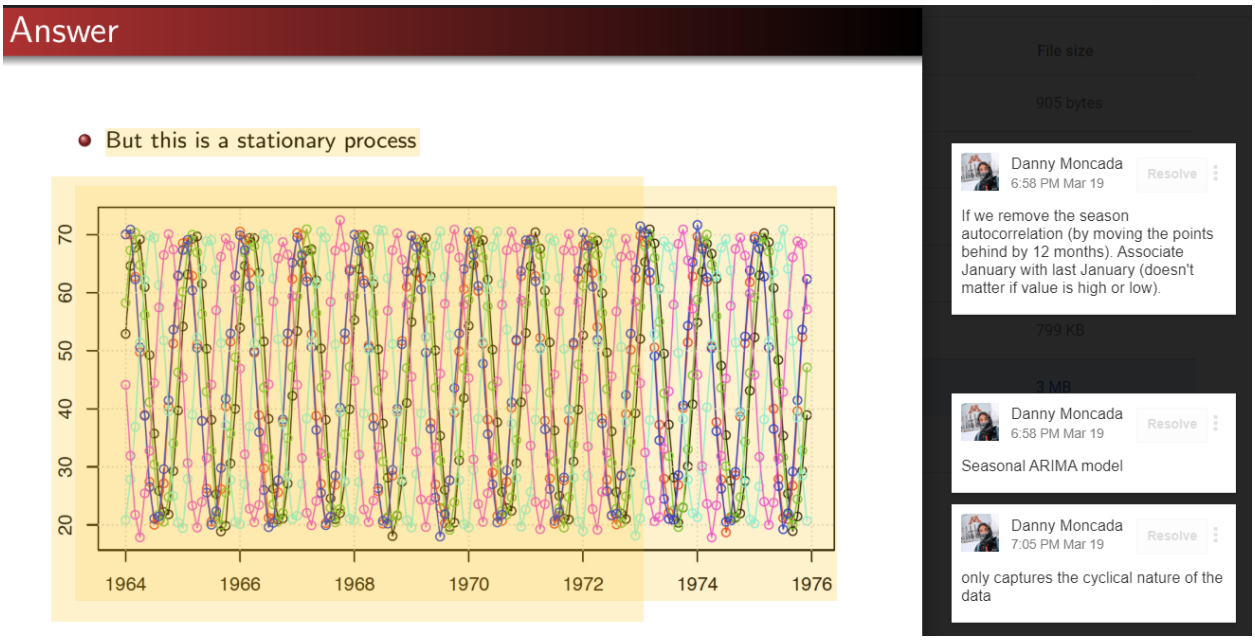


Figure 20: Seasonal Trend - Stationary

4. Any combination of three (linear + seasonal trend): hybrid trend. Split and model separately (which we did in Workday 2).

How do you conduct sequential modeling?

Sequential Modeling: Fitting

- Suppose we want to fit

$$Y_t = X_t\beta + Z_t,$$

  any suggestions?
- Sequential Fitting:
  1. Fit $Y_t \sim X_t\beta$ and get $\hat{\beta}$ (without considering Z_t)
  2. Define the model residual $Z_t \leftarrow Y_t - X_t\hat{\beta}$
  3. Fit Z_t with ARIMA(p,d,q), and get $\hat{Z}_t$
  4. Get $\hat{Y}_t \leftarrow X_t\hat{\beta} + \hat{Z}_t$
- Then how do we forecst?

Suppose you want to fit this, a linear model and time series model. In order to do this, even though you don't know the linear part, you can just fit the linear part first, get the estimate/model residual, and then add the two parts together. If you scale back to the original model, then it will do it after the transformation.

Sequential Modeling: Forecasting

- Suppose we want to forecast

$$Y_{T+1} = X_{T+1}\beta + Z_{T+1}$$

- Forecasting:
  1. For the first part, we have: $X_{T+1}\hat{\beta}$
  2. For the second part, use ARIMA forecasting to get $\hat{Z}_{T+1}$
     - (Or consider using lag: Z_t for Y_{t+1}, $t = 1, \ldots, T$)
  3. Then we get $\hat{Y}_{T+1} \leftarrow X_{T+1}\hat{\beta} + \hat{Z}_{T+1}$

Sequential Modeling

- Sequential modeling is a very important practical skill
- since we can combine the advantages of multiple models
- Later on you could possibly:
- replace the linear model by
  - e.g., Multilayer perceptron (MLP, but not my little pony)
- or replace ARIMA by
  - e.g., LSTM
- or replace both by your favorite ones
- or add a third model
  - e.g., a logistic regression
  - Model fitting $\Rightarrow$ forecasting $\Rightarrow$ decision making

Does auto-correlation of Y_t's affect the estimation of the trend?

Yes, read through the slides below.

What's the difference between a linear trend and a random walk?

- Remember that for a linear model

$$Y_t = \beta_0 + \beta_1 t + e_t,$$

we require e_t's being independent, and hence

$$Cov(Y_t, Y_{t-1}) = 0$$

- But for a random walk,

$$Cov(Y_t, Y_{t-1}) = t - 1$$

- Model assumption violated

Figure 21: Linear Differences 1

- Remember linear regression assumptions:

$$Y_t = \beta_0 + \beta_1 t + \varepsilon_t$$

  where $\varepsilon_t \sim N(0, \sigma^2)$ is i.i.d
- However, let's look at left- and right-hand sides:
  - $\mathrm{Var}[\beta_0 + \beta_1 t + \varepsilon_t] = \mathrm{Var}[\varepsilon_t] = \sigma^2$ and thus, is a constant
  - Yet $\mathrm{Var}[Y_t] = t$ is growing infinitely with t
- Also:
  - $\mathrm{Cov}[\beta_0 + \beta_1 t + \varepsilon_t, \quad \beta_0 + \beta_1(t-1) + \varepsilon_{t-1}] = 0$
  - Yet $\mathrm{Cov}[Y_t, Y_{t-1}] = t - 1$

Figure 22: Linear Differences 2

- The random walk Y_t does not follow the required linear regression assumptions. Hence the problems!

$$\underbrace{Y_t}_{\text{Var grows}} \quad = \quad \underbrace{\beta_0 + \beta_1 t + \varepsilon_t}_{\text{Var is assumed const}}$$

- And:

$$\underbrace{Y_t}_{\text{is corr with } Y_{t-1}} \quad = \quad \underbrace{\beta_0 + \beta_1 t + \varepsilon_t}_{\text{not corr with } \beta_0 + \beta_1(t-1) + \varepsilon_{t-1}}$$

Figure 23: Linear Differences 3

What's the consequence of fitting a random walk with a linear model?

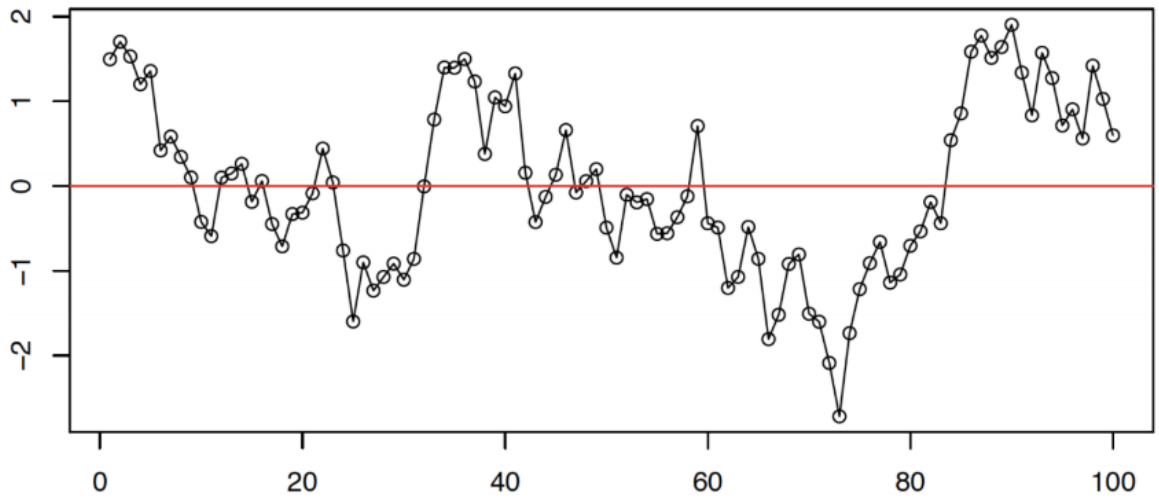
Fit a trend which does not exist. Predicted values keep increasing with t, while random walk has mean 0 and can decrease anytime.

How to check if a regression is spurious, i.e. a random walk fit with linear trend

1. Regression diagnostics to check residual correlation

Residual Plot

- Residual plot: plot model residuals against time
- We can clearly see dependency among consecutive residuals: NOT white noise

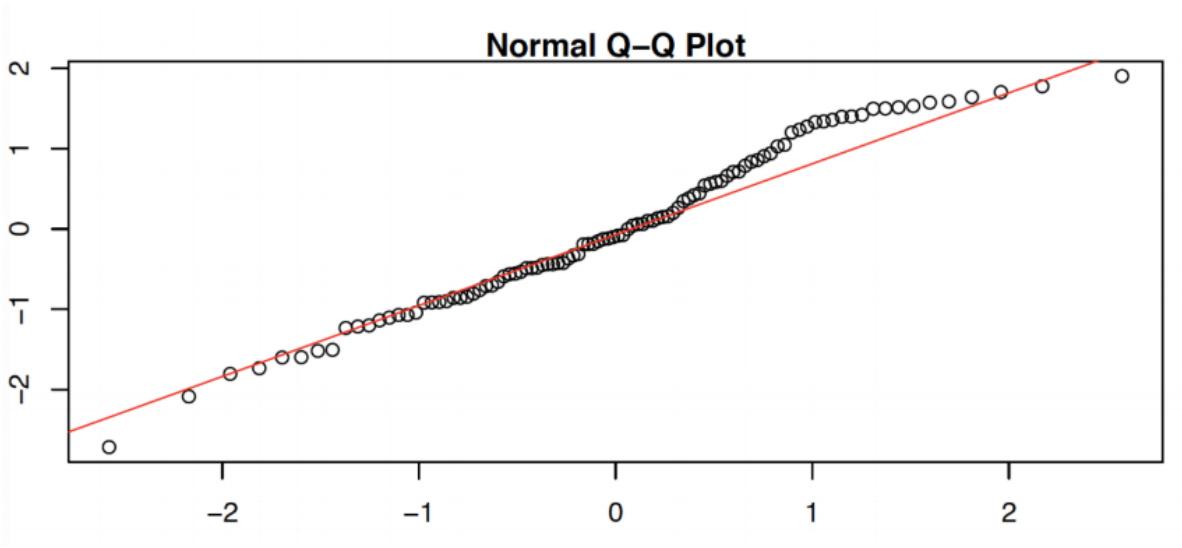


Clearly they are not independent variables.

2. QQ plot

QQ plot

- Points should fall on the straight line if the model residuals are normally distributed (Clearly not this case)
- function *qqnorm()* in R



If it perfectly normal, then all of these points should align perfectly on the red line.

3. ADF test

Lecture 10 & 11: Social Network Analysis

Definitions

- **Def.** Graph $G = (V, E)$ is a collection of
  - vertices $v \in V$
  - edges $e \in E$ where each edge is a pair of vertices (v_1, v_2)

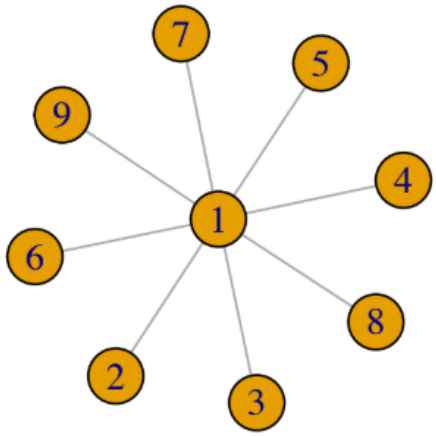


Figure 24: Graphs

A. Vertex:

In mathematics, and more specifically in graph theory, a **vertex** (plural **vertices**) or **node** is the fundamental unit of which graphs are formed: an undirected graph consists of a set of vertices and a set of edges (unordered pairs of vertices), while a directed graph consists of a set of vertices and a set of arcs (ordered pairs of vertices). In a diagram of a graph, a vertex is usually represented by a circle with a label, and an edge is represented by a line or arrow extending from one vertex to another.

Figure 25: Vertex

B. Edge: each edge is a pair of vertices (v_1, v_2)

C. Degree: number of edges a vertex has

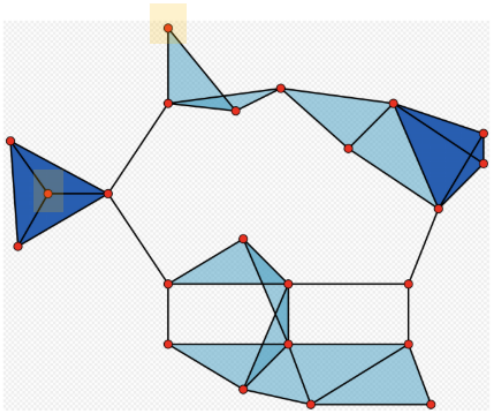


Figure: from Wikipedia.org

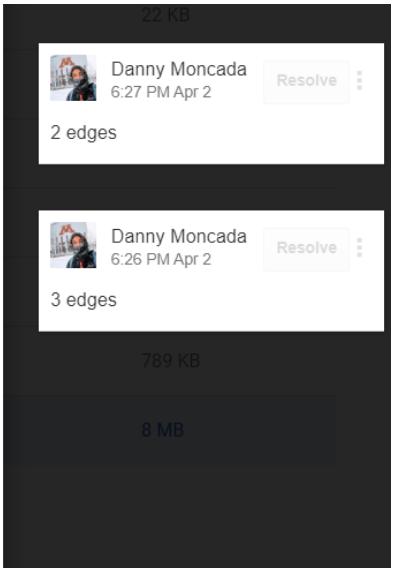


Figure 26: Degree

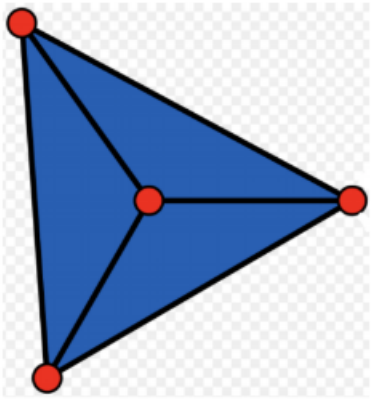
D. Density:

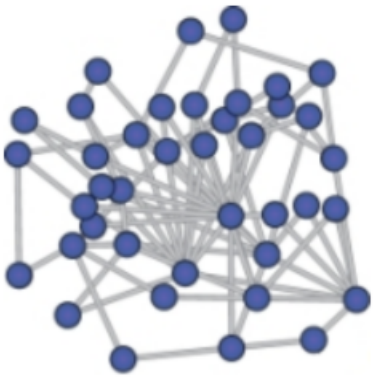
$$\frac{number of edges}{number of total possible edges}$$

- Density for undirected networks:

$$\frac{\text{Number of edges}}{\text{Number of total possible edges}}$$

  - Which one has higher density?





- Smaller networks tend to have a higher density
- Density for directed networks: replace *edges* by *arrows*

The plot is denser. All possible edges are connected. The network is much smaller than the other one.

E. Clique: Cliques are subsets of vertices, all adjacent to each other, which is also called subgraphs.

- Clique: A sub-network in which every two vertices are adjacent (all blue shades)
- Maximum clique: A clique of maximum possible size (dark blue shades)

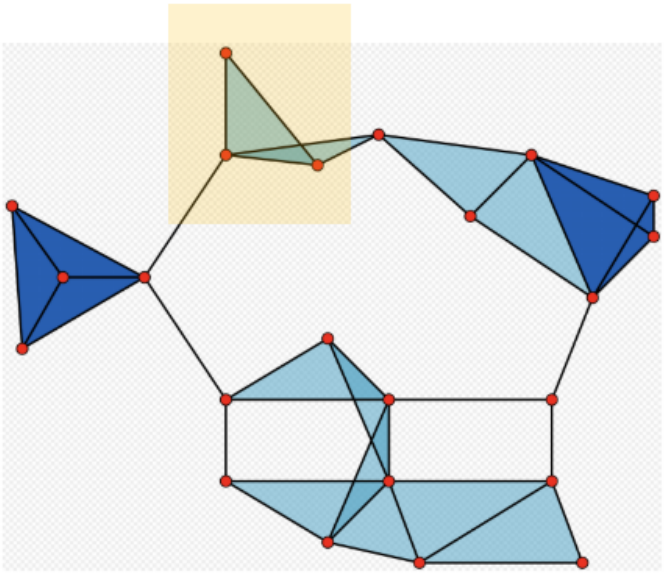


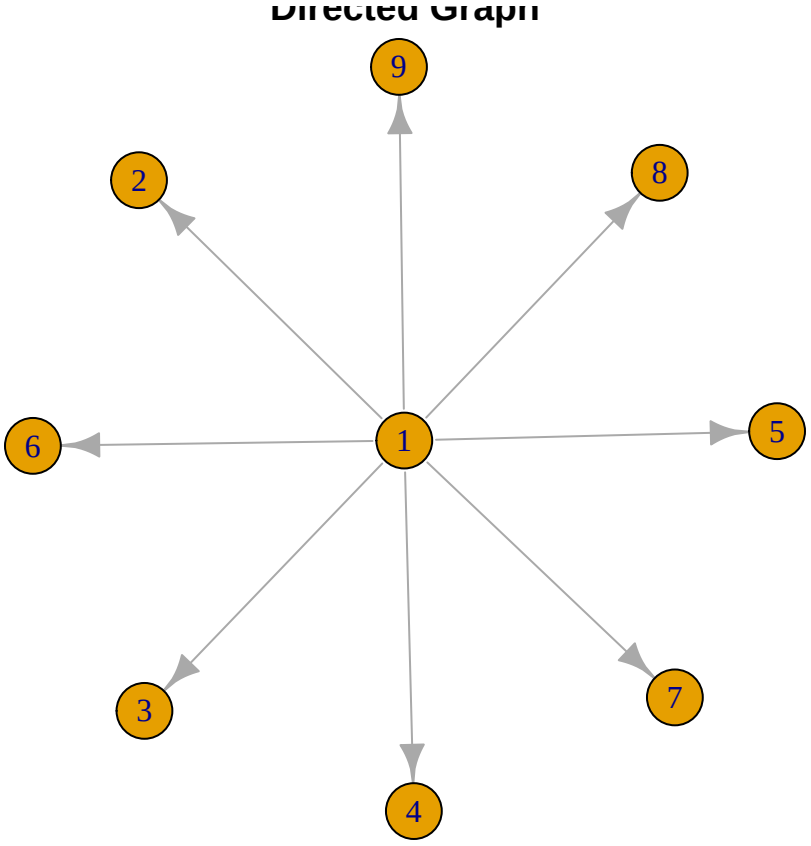
Figure: from Wikipedia.org

Every triangle is a clique, and for maximum clique, no one else can be included in this clique.

F.

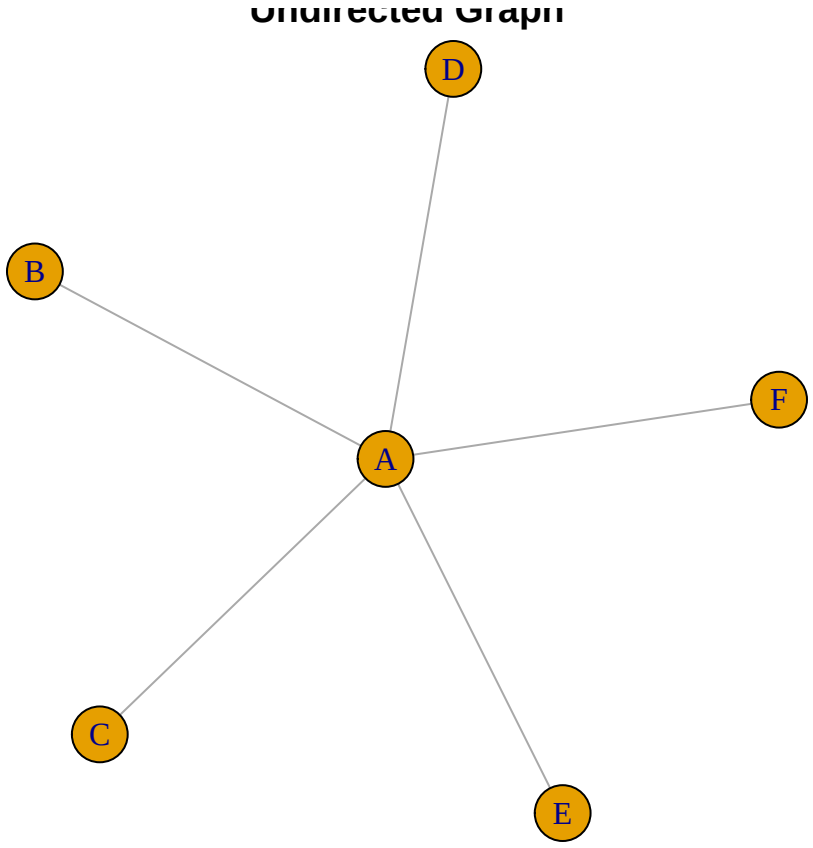
1. Directed network: $(v_1, v_2) \neq (v_2, v_1)$
2. undirected networks: $(v_1, v_2) = (v_2, v_1)$

```
E <- matrix(c(1,2,1,3,1,4,1,5,1,6,1,7,1,8,1,9), ncol = 2, byrow = TRUE)

g1 <- graph_from_edgelist(E, directed=TRUE)
par(mar=c(0,0,0,0));
plot(g1) + title(main = "Directed Graph")
```




```
## integer(0)
```

```
g2 <- graph_from_literal(A--B, A--C, A--D, A--E, A--F)
par(mar=c(0,0,0,0));
plot(g2) + title(main = "Undirected Graph")
```

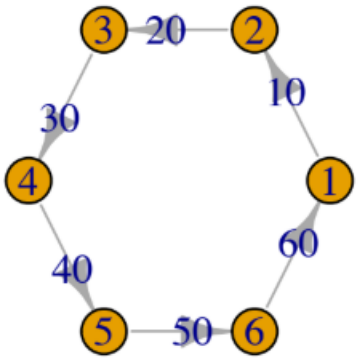



```
## integer(0)
```

G. Weighted/unweighted networks

Weighted Graphs: example

- In directed networks:
  - Say, a circular network of bus routes



- The costs:
  - going from city 1 to city 2 is $10
  - going from city 2 to city 3 is $20
  - and so on

Figure 27: Weighted Graphs

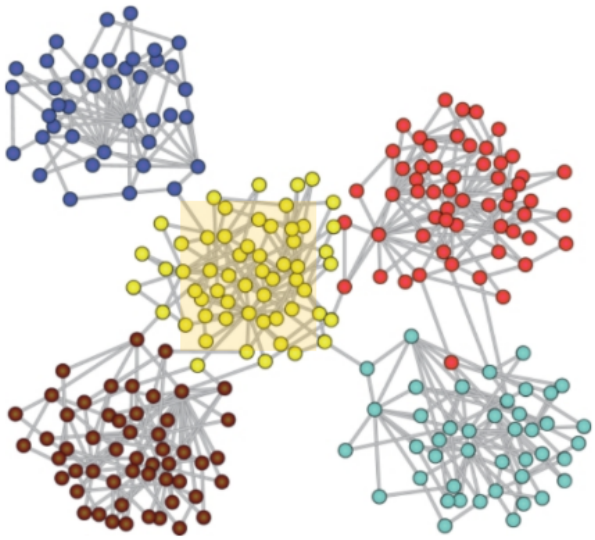
$$e = (v_1, v_2, w)$$

Weight usually represents capacity, cost or distance for that link.

Community Detection

Community Detection

- Community structure: A group of vertices that are more densely connected internally than with the rest of the network



Influence other groups from her - how do you find the people that bridge the gap?#

Basic principles

Edge Betweenness

- The number of shortest paths going through an edge; the edge with higher betweenness tends to be the bridge between two communities.
- Starts from 1 community
- How does it split? Cutting on edges - look up HW write-up

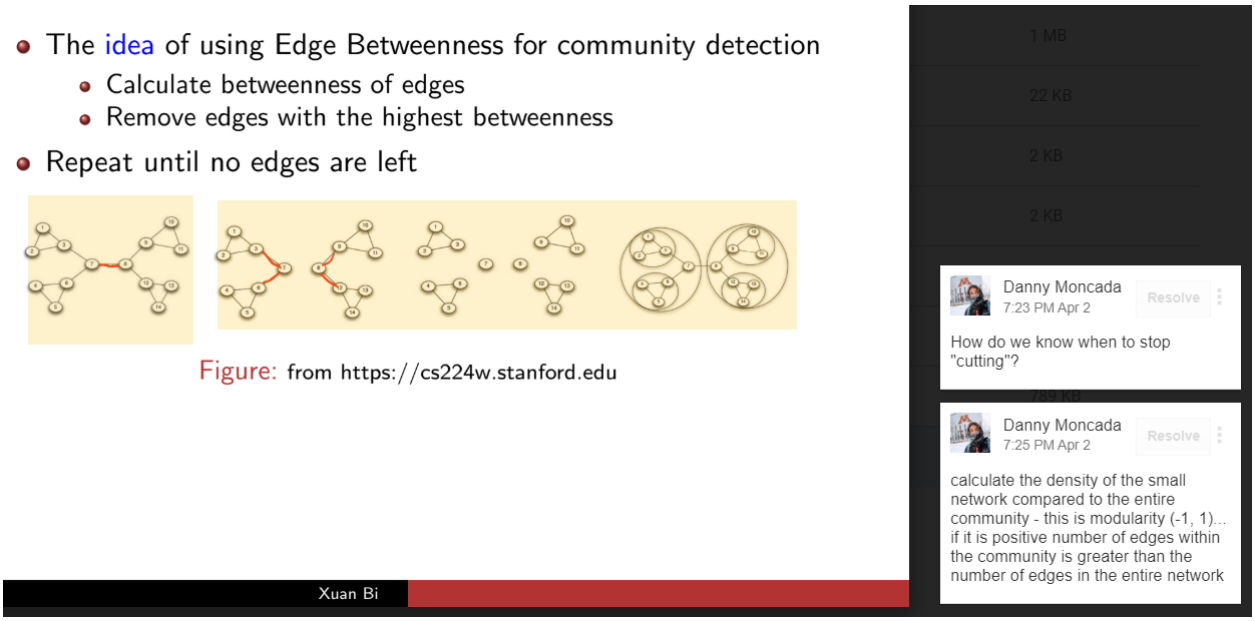


Figure 28: Edge Betweenness

Walk Trap

- Starts from n communities and merges vertices. It assumes that everyone is their own community, and brings them together.
- How does it merge?

- More specifically, it defines a distance r between a vertex i and a community C

$$r_{iC} = \sqrt{\sum_{l=1}^{n} \frac{(P_{il}^t - P_{Cl}^t)^2}{k_l}}$$

such that r is small when the vertex i is in the community C

- The distance is derived from probabilities of random walks (where the name *Walk Trap* stems from)

Figure 29: Walk Trap

Modularity:

- Definition:

$$Q = (\#of edges within community) - (Expected \#within community)$$

This is the stopping criteria, when this is maximized.

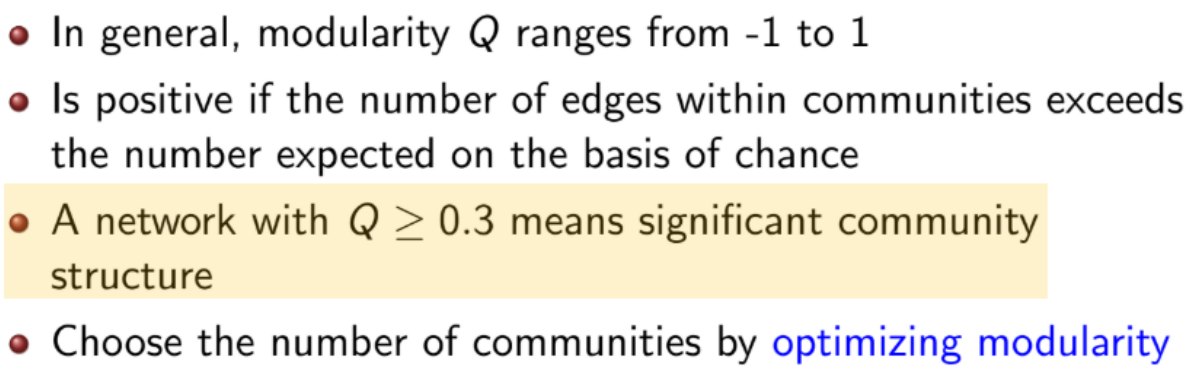
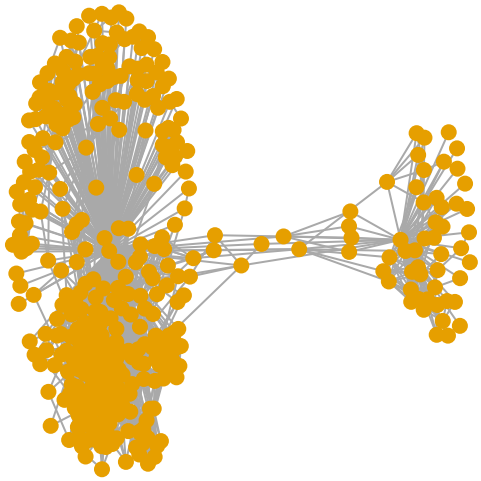


Figure 30: Modularity

How to compare two community structures

```
Esub=read.table("C:/Users/danny/Downloads/fb400.txt",header=F)
Esub=as.matrix(Esub)
#Graph illustration
par(mfrow=c(1,1))
g <- graph_from_edgelist(Esub, directed=FALSE)

set.seed(66)
plot(g,vertex.label=NA,
vertex.frame.color=NA, vertex.size=7,
edge.arrow.size=0.5, edge.arrow.width=0.5)
```



```
#Edge betweenness
t0=proc.time() #Also count the computational time
EB=edge.betweenness.community(g, directed = FALSE) #modularity=0.42
proc.time()-t0 #8.8s
```

```
##    user  system elapsed
##   10.66    0.00   10.69
```

```
memberEB=membership(EB)
max(memberEB) #41 clusters
```

```
## [1] 41
```

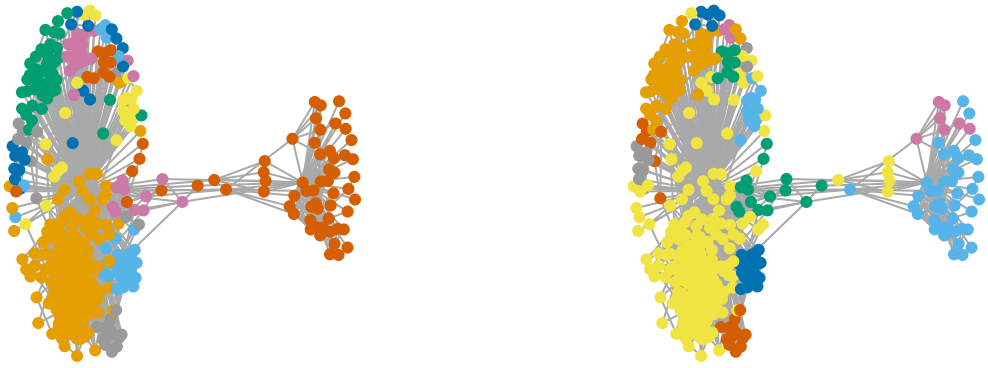
```
#WalkTrap
t0=proc.time()
WT=walktrap.community(g) #modularity=0.41
proc.time()-t0 #0.008s
```

```
##    user  system elapsed
##    0.01    0.00    0.01
```

```
memberWT=membership(WT)

max(memberWT) #18 clusters
```

```
## [1] 18
```

```
#Illustrate and compare the community detection results
par(mfrow=c(1,2))
set.seed(66)
plot(g,vertex.label=NA, vertex.color=memberEB,
vertex.frame.color=NA, vertex.size=7,
edge.arrow.size=0.5, edge.arrow.width=0.5)
set.seed(66)

plot(g,vertex.label=NA, vertex.color=memberWT,
vertex.frame.color=NA, vertex.size=7,
edge.arrow.size=0.5, edge.arrow.width=0.5)
```



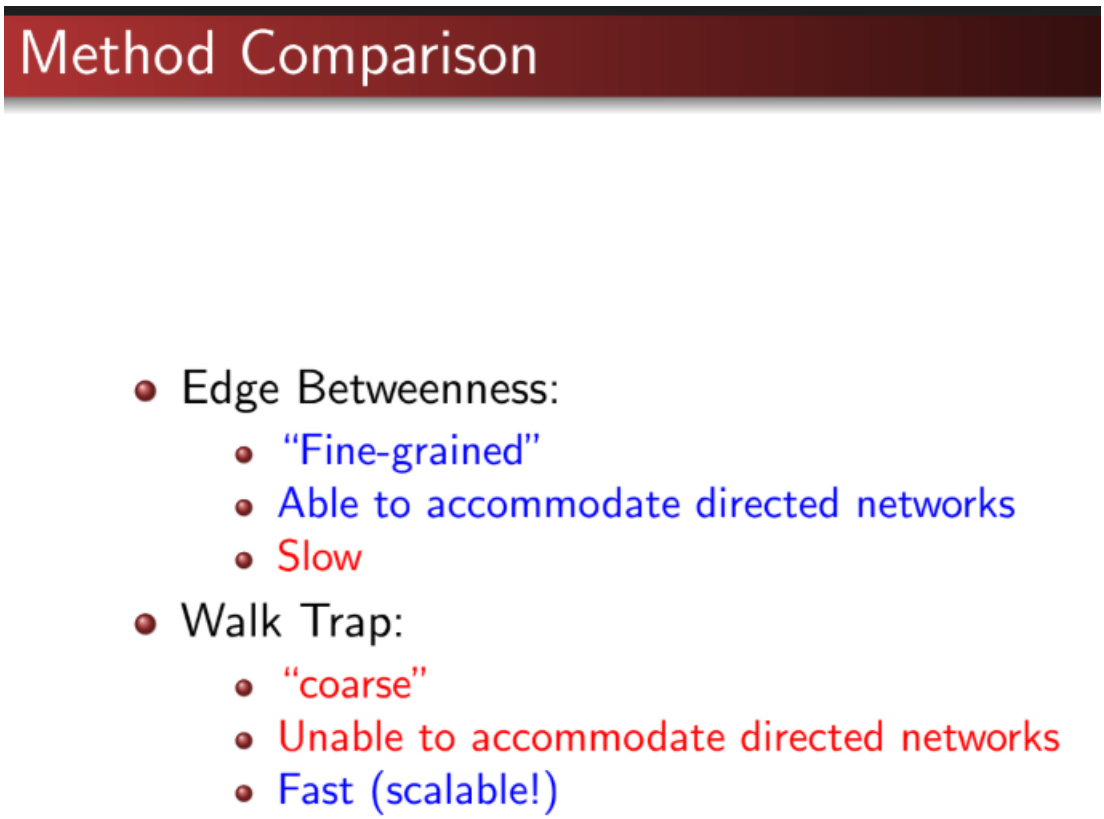


Figure 31: Method Comparison

Personalization

Latent Factor Models in Social Networks

- Representation of latent factors
- How are latent factors utilized in social networks?
  - Undirected networks: *node* (vertex)
  - Directed networks: *sender*, *receiver*
    - each entity may have two latent factors
    - one represents its sender attributes
    - the other represents its receiver attributes
- But how do we formulate latent factors?

Figure 32: Latent Factor Models

Adjacency matrix

Defintion: An important representation of a graph. The elements of the matrix represent the edges of the graph. It is applicable both to directed and weighted graphs.

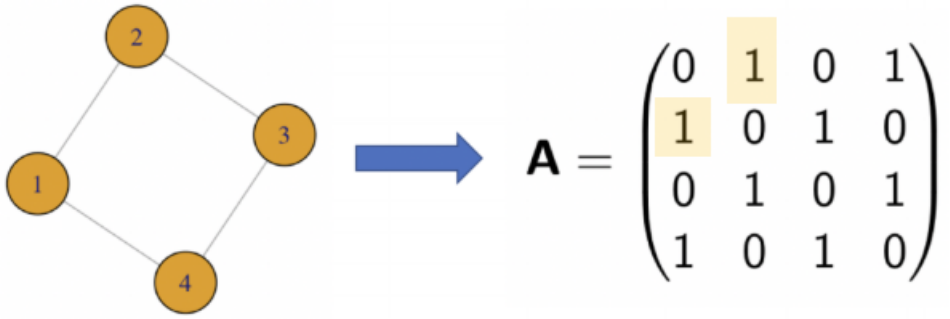
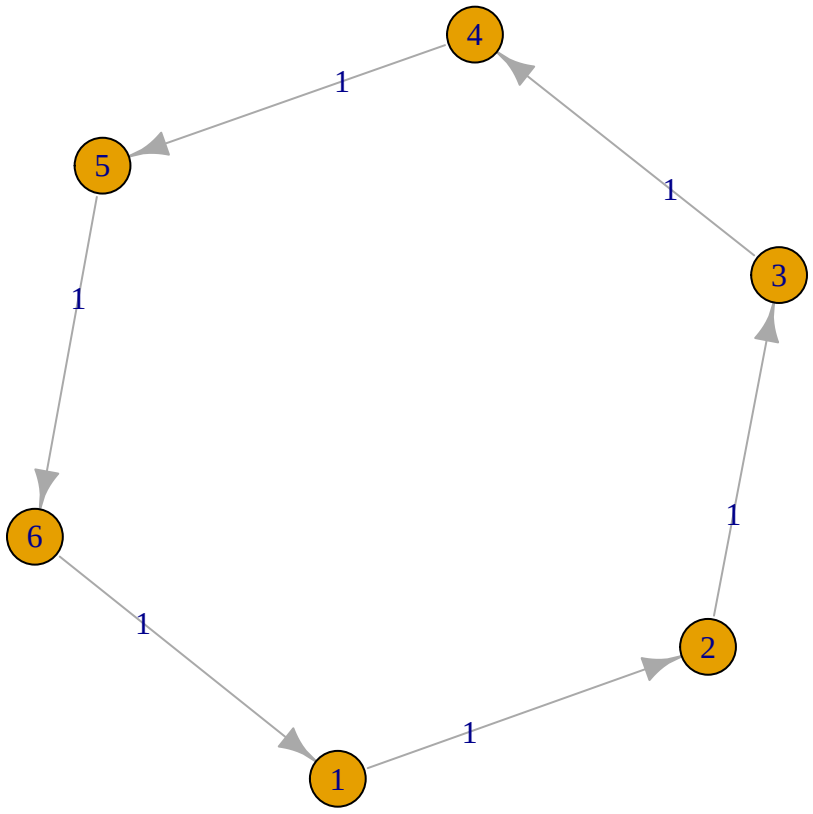
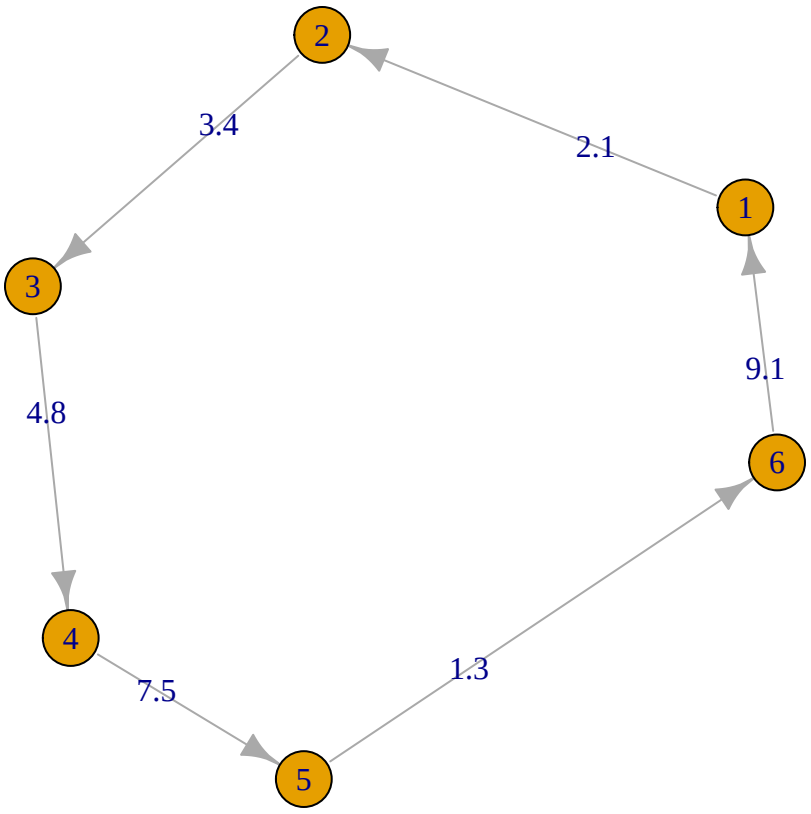


Figure 33: Adjacency Matrix

```
A <- matrix(c(0,1,0,0,0,0,
              0,0,1,0,0,0,
              0,0,0,1,0,0,
              0,0,0,0,1,0,
              0,0,0,0,0,1,
              1,0,0,0,0,0),
              ncol = 6, byrow = TRUE)

g <- graph_from_adjacency_matrix(A, weighted = TRUE)
par(mar=c(0,0,0,0)); plot(g, edge.label = E(g)$weight)
```




```
A <- matrix(c(0,2.1,0,0,0,0,
              0,0,3.4,0,0,0,
              0,0,0,4.8,0,0,
              0,0,0,0,7.5,0,
              0,0,0,0,0,1.3,
              9.1,0,0,0,0,0),
        ncol = 6, byrow = TRUE)

g <- graph_from_adjacency_matrix(A, weighted = TRUE)
par(mar=c(0,0,0,0)); plot(g, edge.label = E(g)$weight)
```



Latent factors (send and receiver attributes) P: p_i sender-specific latent factor Q: q_j receiver-specific latent factor

Missing Data

- But what if

$$\mathbf{A} = \begin{pmatrix} 0 & 2.1 & 0 & 0 & ? & 0 \\ ? & 0 & 3.4 & 0 & 0 & 0 \\ ? & 0 & 0 & 4.8 & 0 & 0 \\ 0 & 0 & ? & 0 & 7.5 & 0 \\ 0 & ? & ? & 0 & 0 & 1.3 \\ 9.1 & 0 & 0 & 0 & ? & 0 \end{pmatrix}$$

- Sometimes it's hard to distinguish 0 from "?"
- Not connected $\Rightarrow$ "not know" or "don't want to connect"

Figure 34: Missing Data

Latent factors

- Similar to the recommender systems
- We use a latent factor model to fill in missing values
- We assume the adjacency matrix

$$\mathbf{A}_{n\times n} = \mathbf{P}_{n\times k}(\mathbf{Q}_{n\times k})'$$

- For each a_{ij}, we have

$$a_{ij} = \mathbf{p}_i'\mathbf{q}_j$$

- Here $\mathbf{A}_{n\times n}$ is a square matrix
  (as opposed to the user-item matrix in RecSys)

Figure 35: Latent Factors 1

Missing Data

- $\mathbf{p}_i$ and $\mathbf{q}_j$ are estimated iteratively in algorithm
- If the estimated $\mathbf{p}_i$ is close to $\mathbf{q}_j$, then
- **Interpretation:** i has strong inclination to connect with j
  - Here we call $\mathbf{p}_i$ the sender-specific latent factor
  - And $\mathbf{q}_j$ the receiver-specific latent factor
- Question: How to choose the number of latent factors k?
- Practically challenging

Figure 36: Latent Factors 2

How to predict weights (probability) of a link based on P and Q

- Matrix multiplication

Suppose node 2's sender-specific latent factor is p2 = (0, 0.4, 0.6) and node 1's receiver-specific latent factor is q1 = (0.7, 0.1, 0). How likely will node 2 send a signal invitation to node 1?

- Likelihood equation:
  - (0 x 0.7) + (0.4 x 0.1) + (0.6 x 0) = 0.04
  - 0.04 is your probability.

Undirected Graph

- Recall: What does the adjacent matrix of an undirected graph look like?
- Symmetry! For all (i, j)'s, $a_{ij} = a_{ji}$
- Then how do we estimate the probability of each link?
- Simply let $\mathbf{P} = \mathbf{Q}$

Figure 37: Probability 1

Undirected Graph

- That is,

$$prob_{ij} = \mathbf{p}_i' \mathbf{p}_j$$

- Each individual is now both sender and receiver
- $\mathbf{p}_i$ describes i's all characteristics
- High probability of connection, if i's characteristics are similar to j's

Figure 38: Probability 2

Optional Reading: Unweighted Graphs

- For example, in logistic regression:
- Instead of using $a_{ij} = \mathbf{p}_i'\mathbf{q}_j$
- We now assume

$$a_{ij} \sim Bin(prob_{ij})$$

- with

$$prob_{ij} = \frac{\exp(\mathbf{p}_i'\mathbf{q}_j)}{1 + \exp(\mathbf{p}_i'\mathbf{q}_j)}$$

- In R, this can be done by changing the *loss()* option into *log* or *hinge*

Figure 39: Probability 3

If you made it all the way here, then you know everything there is to know about time series data, estimation, forecasting, and social network analysis. And you should be extremely proud.