Time Series Analysis & Forecasting

Class 4

Arnab Bose, Ph.D.

MSc Analytics

University of Chicago

Model Relative Quality

Measure the **relative** quality of models given a dataset

- Akaike Information Criterion $AIC = 2k - 2\ln(L)$

  where

  - $k -$ # of parameters in the model
  - L – maximum value of the likelihood estimator

- AIC has bias for small sample size that is corrected with Akaike Information Criterion

$$AICc = AIC + \frac{2k(k+1)}{n-k-1}$$, n is the sample size

- Bayesian Information Criterion $BIC = -2\ln(L) + k\ln(n)$

Model Diagnostic – Residual Analysis

- Define residual = actual – estimate

- Autocorrelation – use Durbin Watson

$$d = 2(1 - r)$$

r => 1st lag sample autocorrelation of the residual

d = 2 => no autocorrelation ($0 \le d \le 4$)

H_0 = no autocorrelation i.e. $r = 0$

H_a= autocorrelation i.e. $r \neq 0$

Ljung-Box Test

1. Let sample autocorrelation function of the residuals be denoted as $\widehat{r_k}$

2. Box-Pierce statistic

$$Q = n \sum_k \widehat{r_i^2}$$

3. For ARMA(p,q) for large n, Q has approximate χ^2 (chi-squared) distribution

4. Ljung-Box noted that even with large n, n=100, the approximation to χ^2 distribution is not satisfactory

5. Ljung-Box modified Q

$$Q = n(n+2) \sum_k \widehat{r_i^2}$$

H_0 = data is independent distributed, i.e. no autocorrelation

H_a= data is not independently distributed, i.e. autocorrelation

Portmanteau Test

Box-Pierce and Ljung-Box tests are Portmanteau test – null hypothesis H_0 is well-defined but alternative hypothesis H_a is loosely defined.

Good for tests where a model has multiple ways to depart from H_0 .

Model Diagnostics

$$residual = actual - estimate$$

- Autocorrelation
  - *Durbin Watson*
  - *Ljung Box (https://robjhyndman.com/hyndsight/ljung-box-test/)*
- Normality
  - *Shapiro Wilk*
  - *Kolmogorov Smirnov*
  - *Jarque Bera*
- Heteroscedasticity
  - *Breusch Pagan*
  - *McLead Li*

Time Series Decompositions – Trend, Seasonal and Cyclical

- Trend adjustments
  - Fit a regression model and subtract from TS to get residuals with no trend
    - Assumes constant trend historically and continuing into (immediate) future
  - Difference the TS
    - No parameter estimation (simpler)

- Seasonal adjustments
  - To eliminate seasonality say with lag = D

$$\nabla^D \, Y_t = (1 - B^D) Y_t = Y_t - Y_{t-D}$$

- Cyclical adjustments
  - Adjust data ups and downs that have no fixed period with curve fitting

R code – Regular and Seasonal Differencing

```
library(datasets)
ts.plot(AirPassengers)
ts.plot(diff(AirPassengers, 12))
ts.plot(diff(AirPassengers))
ts.plot(diff(diff(AirPassengers, 12)))
ts.plot(AirPassengers)
```

Seasonal ARIMA

- Mathematically using the backward shift operator

$$B^s z^t = z_{t-s}, s \Rightarrow seasonal\ period$$

- TS represented as

$$\Phi(B^s)Y_t = \Theta(B^s)e_t$$

- Also, since errors are autocorrelated as in regular ARIMA, we also have

$$\Phi(B)Y_t = \Theta(B)e_t$$

- So multiplicative seasonal ARIMA, SARIMA $(p, d, q) \times (P, D, Q)_S$ is represented as

$$\Phi(B)\Phi(B^s)\nabla^d \nabla_s^D\ Y_t = \Theta(B)\Theta(B^s)e^t$$

Time Series Additive & Multiplicative Decompositions

- Additive Model

$$y_t = S_t + T_t + E_t$$

y_t - data at period t

S_t - seasonal component at period t

T_t - trend-cycle component at period t

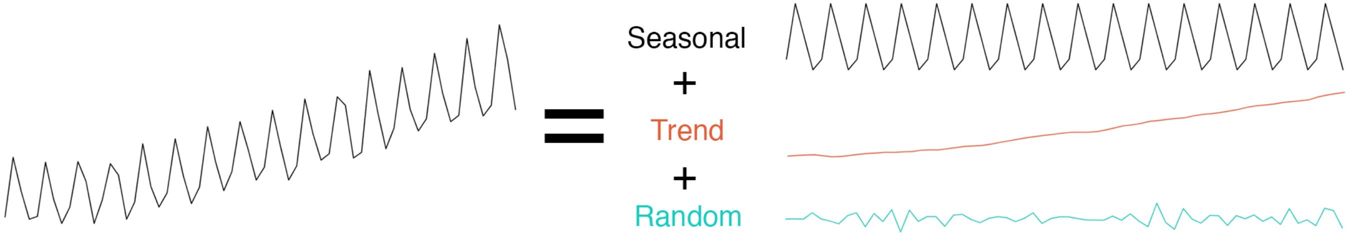
E_t - error component at period t

- Multiplicative Model

$$y_t = S_t \times T_t \times E_t$$

STL Decomposition

1. Seasonal and Trend decomposition using Loess (Local weighted regression)
2. Can handle any type of seasonality (not just monthly or daily)
3. Useful for business cycles where every cycle may not have the same period
4. Seasonal component can change over time
5. Robust to outliers- occasional unusual observations do not affect the estimates
6. Can handle non-stationary data
7. TS repeated patterns
    1. Seasonal – fixed known period that is associated with calendar (seasonal ARMA models)
    2. Cyclic – data has ups and downs with no fixed period (ARMA models)



https://anomaly.io/seasonal-trend-decomposition-in-r/
http://www.gardner.fyi/blog/STL-Part-II/

STL Parameters

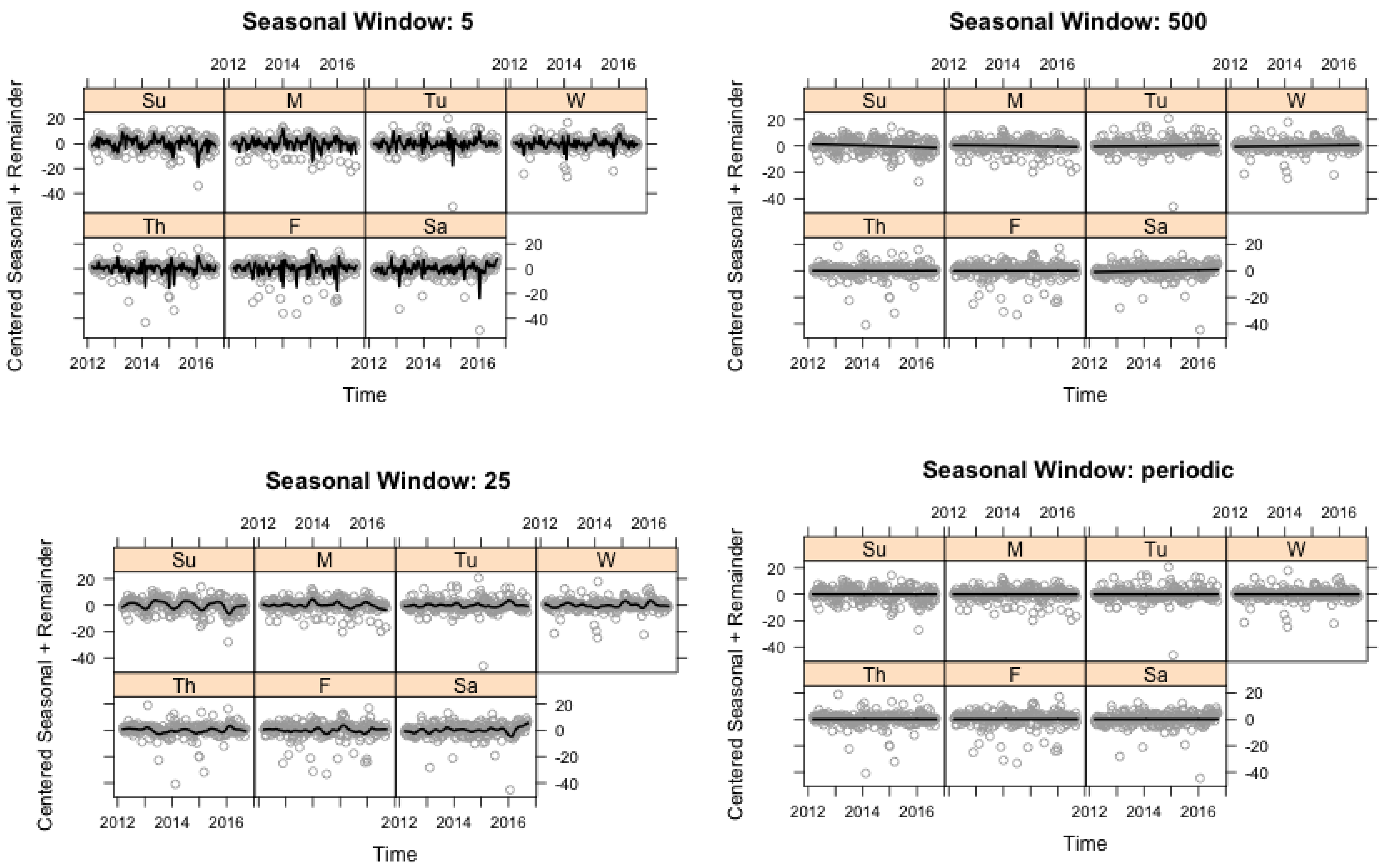
1. `t.window` aka trend-cycle window – controls how rapidly the trend-cycle can change. It is the number of consecutive observations to be used when estimating the trend-cycle. As this increases, the trend is increasingly smoothed. Has a default value, check R help for details.

2. `s.window` aka seasonal window – controls how rapidly the seasonal component can change. It is the number of consecutive years to be used in estimating each value in the seasonal component. User must specify as there is no default. Setting it to be infinite is equivalent to forcing the seasonal component to be periodic (i.e., identical across years).

3. Both settings should be odd numbers.

4. Smaller values allow for more rapid changes.

STL s.window



http://www.gardner.fyi/blog/STL-Part-II/

R code – Seasonal Arima and STL

```
library(datasets)
library(forecast)
a <- auto.arima(AirPassengers)
plot(forecast(a, h = 12))
auto.arima(diff(AirPassengers, 12))
auto.arima(diff(AirPassengers))
s.p <- stl(AirPassengers, s.window = "periodic")
plot(s.p)
plot(forecast(s.p, h = 12))
s.7 <- stl(AirPassengers, s.window = 7)
plot(s.7)
plot(forecast(s.7, h = 12))
```

Regression with ARIMA errors aka Dynamic Regression

- Regression assumption that errors are not autocorrelated is violated

$$y_t = \beta_0 + \beta_1 x_t + u_i$$

where u_i is autocorrelated

- So u_i can be modeled as

$$\phi(B)u_i = \theta(B)e_t$$

where e_t is white noise

- ARMA can be considered as a special type of regression model => predictors are lags of the dependent variable and/or lags of the forecast error

Dynamic Regression – Cochrane Orchutt

- Refer to PDF in Reading

R code – Regression with ARIMA errors

```
x <- 1:100
e <- arima.sim(model=list(ar=0.3, ma=0.9), n=100)
y <- 1 + 2*x + e
fit1 <- lm (y~x)
summary(fit1)
Plot(fit1)
acf(fit1$residuals, lag=100)
qqnorm(fit1$residuals)
qqline(fit1$residuals)
fit2 <- auto.arima(y, xreg=x)
summary(fit2)
qqnorm(fit2$residuals)
qqline(fit2$residuals)
par(mfrow=c(1,2))
acf(fit2$residuals)
acf(fit1$residuals, lag=100)
```

ARIMAX

- Different than Regression with ARIMA errors – refer to PDF in Reading
- Read The ARIMAX model muddle

ARFIMA – Definition

- Fractionally Integrated ARMA model => ARFIMA
- Mathematically represented as

$$\Phi(B)(1-B)^d\, Y_t = \Theta(B)e_t$$

where $-0.5 < d < 0.5$

ARFIMA – Properties

1. For $0 < d < 0.5$ the TS $\{y_t\}$ has positive autocorrelations such that they decay at hyperbolic rate => aka “Long Memory” process.
2. For $-0.5 < d < 0$ the TS $\{y_t\}$ has absolute value of the autocorrelations converge to a constant => aka “antipersistent” or “intermediate memory” process.
3. For $d = 0$ $\{y_t\}$ is ARMA => aka “short memory” process.
4. For $0.5 \leq d < 1$ the TS $\{y_t\}$ is non-stationary since variance is not finite, but interestingly mean-reverting.
5. For $d < -0.5$, the TS $\{y_t\}$ is not invertible.
6. Long memory ARFIMA TS $\{y_t\}$ lies hafway between stationay I(0) and non-stationary I(1) processes.
7. Mandelbrot fractals – objects with fractional dimension that exhibit self-similarity and the dimensions decreases as *d* increases, i.e. TS $\{y_t\}$ with lower *d* exhibits more fractal behavior than higher *d*.

ARFIMA – Detection

1. Use the Hurst Exponent to determine long memory of TS $\{y_t\}$.
2. It uses the rescaled range – a statistical measure of the variability of a series of numbers introduced by the British hydrologist Harold Edwin Hurst (1880 – 1978).
3. The rescaled range is calculated by dividing the range of TS $\{y_t\}$ with the standard deviation of the TS.
4. For a TS $\{y_t\}$ the slope of the plot of the rescaled range vs the logarithm of number of observations gives the Hurst exponent H.
5. $0 \leq H \leq 1$, and $0.5 < H$ indicates long memory TS.
6. Brownian motion (aka random walk) has $H = 0.5$.
7. For example, the height of the Nile river measured annually gives $H \approx 0.7$*.

* https://blog.revolutionanalytics.com/2014/09/intro-to-long-memory-herodotus-hurst-and-h.html

R code – ARFIMA

```
library("arfima")
data("SeriesJ")
acf(SeriesJ$YJ, lag=40)
library("forecast")

d <- fracdiff::fracdiff(SeriesJ$YJ)    #get the fractional d
st <- fracdiff::diffseries(SeriesJ$YJ,d$d)   # do the fractional diff
acf(st, lag=40)
m1 <- auto.arima(st)  # now the TS is stationary, run ARIMA
AIC(m1)

# does the above (fractional difference + ARIMA) in 1 step
m2 <- forecast::arfima(SeriesJ$YJ)
AIC(m2)  # you can compare the models and choose the one with lower AIC
```

Textbook Chapters

Materials covered available in book chapters:

FPP: 3 – 5, 9, 10

PTS: 2, 6

Thank You


IF I HAD IT TO DO AGAIN, I'D ASK MORE QUESTIONS AND INTERRUPT FEWER ANSWERS.
ROBERT BRAULT