{ITSMR CHEAT SHEET}

R PACKAGE
(YANQING WANG, QINQI ZHANG)

INTRODUCTION:

- Time Series and Forecasting:

- A time series is a set of observations with each one being recorded at a specific time.
- Forecasting time series have different methods, such as recursive prediction algorithms, Durbin-Levinson algorithm and innovations algorithms.

- Description of ITSMR:

- ITSMR contains functions for modeling and forecasting time series data and uses innovations algorithms for forecasting variance.
- ITSMR gives users without ITSM2000 similar operations with R language.

MAIN CONCEPTS:

- Weak Stationarity:

- If both mean function and covariance function are independent to time, this time series is (weak) stationary.
- Stationary processes have autocovariance (**ACVF**) and Autocorrelation (**ACF**).

- Innovations Algorithms:

- Innovations algorithm is applicable when the process is nonstationary.
- It predicts by recursively computing coefficient of innovations and corresponding MSE with ACVF, where one-step predictor is a linear combination of innovations.

- ARMA(p,q) Process:

- $\phi(B)X_t = \theta(B)Z_t, \quad Z_t \sim WN(0, \sigma^2)$
- **Causality:** $\phi(z) \neq 0 \; \forall \; |z| \leq 1$
- **Invertibility:** $\theta(z) \neq 0 \; \forall \; |z| \leq 1$
- all MA(q) are causal, all AR(p) are invertible.
- Partial autocorrelation function(**PACF**) can be used for identifying AR(p) process using visualization.

COMMON FUNCTIONS:

- aacvf(a, h):

- Computes ACVF for only ARMA process
- Inputs a=ARMA model with ϕ values vector, θ values vector and σ^2 of Z_t and h=lag
- Returns a vector with length h+1

- acvf(x, h):

- Inputs x=time series data and h=lag
- Computes ACVF for any time series data

- ar.inf(a, n = 50):

- Compute AR infinity coefficients
- Inputs a=ARMA model, n=Order
- Returns a vector of length n+1 to accomodate coefficient 0 at index 1

- ma.inf(a, n = 50):

- Compute MA infinity coefficients
- Inputs a=ARMA model and n=Order
- Returns a vector of length n+1 to accomodate coefficient 0 at index 1.

- arar(y, h = 10, opt = 2):

- Forecast using ARAR algorithm
- Inputs y=time series data, h=steps ahead, opt=display option
- Returns predicted values, standard errors, lower bounds (95% CI), and upper bounds

- autofit(x, p = 0:5, q = 0:5):

- Find the best model from a range of possible ARMA models
- Inputs x=time series data, p=range of AR orders, q=range of MA order
- Returns an ARMA model

- Resid(x, M = NULL, a = NULL):

- Inputs x=Time series data, M=Data model, a=ARMA model
- Returns a vector of residuals the same length as x, and Compute residuals

MODELLING AND FORECASTING:

- Estimating ARMA Parameters

function.	input	method
yw	Time series data, AR order	Yule-Walker
burg	Time series data, AR order	Burg
ia	Time series data, MA order, Recursion level	Innovations Algorithm
hannan	Time series data, AR order, MA order	Hannan-Rissanen
arma	Time series data, AR order, MA order	Maximum Likelihood

- Test and Forecast

Function.	Input	Description
check	ARMA model	Check for causality and invertibility
test	Time series data(residuals)	Test stationarity and randomness
selftest	Null	Run a self test
trend	Time series data, Polynomial order	Estimate trend component
forecast	Time series data, Data model, ARMA model, Steps ahead, Display option, α	Forecast future values

DATASETS AND VISUALIZATION:

- Time Series Datasets

airpass	deaths	dowj	lake	strikes	Sunspots	wine

- Plot Functions

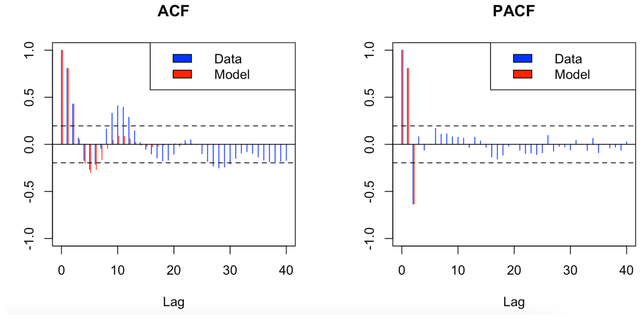
Function	Input	Description
periodogram	Time series data, MA filter order, Plot option	Plot a periodogram
plota	Data and/or ARMA model, Maximum lag	Plot ACF and PACF of input
plotc	2 Data vectors	Plot one or two time series
plots	Data vector or an ARMA model	Plot spectrum of input

- Smooth Filters

Function.	Input	Method
smooth.exp	Time series data, Smoothness setting	an exponential filter
smooth.fft	Time series data, Cut-off frequency	a low pass filter
smooth.ma	Time series data, Filter order	a moving average filter
smooth.rank	Time series data, Number of frequencies	a spectral filter

- Example

> plota(Sunspots,yw(Sunspots,2))



> plotc(strikes,smooth.ma(strikes,2))