# EF4822 Financial Econometrics Problem Set 2

Please submit in groups and write out the names of each groupmate. If R programming is used, please attach the R commands. The data files needed in this problem set are uploaded into the same Canvas folder as this file.

Note on online submission: Any file format is accepted as long as it is clear to read. PDF and Word file format are preferred. You could scan, take photos, or type.

1. Suppose that the simple return of a monthly bond index follows the model

*rt* = 0*.*02 + *at* + 0*.*2*at*−2*,*

where *at* is a white noise series with mean zero and standard deviation *σa* = 0*.*025. What are the mean and variance of the return series *rt*? Compute the lag-1 and lag-2 autocorrelations of *rt*. Assume that *a*100 = 0*.*01 and *a*99 = −0*.*02. Compute the 1-step-ahead and 2-stepahead forecasts of the return series at the forecast origin *t* = 100. Compute the associated forecast errors and the standard deviations of the forecast errors.

1. Suppose that the daily log return of a security follows the model

*rt* = 0*.*01 + 0*.*2*rt*−2 + *at,*

where *at* is a white noise series with mean zero and variance 02. What are the mean and variance of the return series *rt*? Compute the lag-1 and lag-2 autocorrelations of *rt*. Assume that *r*100 = −0*.*01, and *r*99 = 0*.*02. Compute the 1-step-ahead and 2-step-ahead forecasts of the return series at the forecast origin *t* = 100. Compute the associated forecast errors and the standard deviations of the forecast errors.

data("m.deciles08")

#This tells you that the data series is in a time series format

is.ts(m.deciles08)

## [1] FALSE

#We change the data to time series format

#STEP 2:

# Now that we know that the data is time series we should do some data exploration. Functions print() and summary() are used to get the overview of the data. The start() and end() functions return the time index of the first and last observations, respectively. The time() function calculates a vector of time indices, with one element for each time index on which the series was observed. Finally, the frequency() function returns the number of observations per unit time.

#This will give us the structure of our data

print(m.deciles08)

summary(m.deciles08)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 104.0 180.0 265.5 280.3 360.5 622.0

#Starting index, end index

start(m.deciles08)

## [1] 1949 1

end(m.deciles08)

time(m.deciles08)

frequency(m.deciles08)

#Step 3:

# It is essential to analyze the trends prior to building any kind of time series model. The details we are interested in pertains to any kind of trend, seasonality or random behaviour in the series. what better way to do so than visualize the Time Series.

#This will plot the time series

ts.plot(AirPassengers, xlab="v1", ylab="V3", main="Monthly totals of international airline passengers, 1949-1960")

# This will fit in a line

abline(reg=lm(m.deciles08~time(m.deciles08)))

#Auto correlation matrixx

acf(m.deciles08)

#Fit the AR model to the dataset

AR <- arima(m.deciles08, order = c(0,1,0))

print(AR)

#Plotting the AR model

ts.plot(m.deciles08)

#Fitting the model

AR\_fit <- m.deciles08 - residuals(AR)

points(AR\_fit, type = "l", col = 2, lty = 2)

#Using predict() to make a 1-step forecast

predict\_AR <- predict(AR)

#Obtaining the 1-step forecast using $pred[1]

predict\_AR$pred[1]

#ALternatively Using predict to make 1-step through 10-step forecasts

predict(AR, n.ahead = 10)

#plotting the data series plus the forecast and 95% prediction intervals

ts.plot(m.deciles08, xlim = c(1949, 1961))

AR\_forecast <- predict(AR, n.ahead = 10)$pred

AR\_forecast\_se <- predict(AR, n.ahead = 10)$se

points(AR\_forecast, type = "l", col = 2)

points(AR\_forecast - 2\*AR\_forecast\_se, type = "l", col = 2, lty = 2)

points(AR\_forecast + 2\*AR\_forecast\_se, type = "l", col = 2, lty = 2)

#Fitting the MA model to the dataset

MA <- arima(m.deciles08, order = c(0,0,1))

print(MA)

#plotting the series along with the MA fitted values

ts.plot(m.deciles08)

MA\_fit <- m.deciles08 - resid(MA)

points(MA\_fit, type = "l", col = 2, lty = 2)

#Making a 1-step forecast based on MA

predict\_MA <- predict(MA)

#Obtaining the 1-step forecast using $pred[1]

predict\_MA$pred[1]

#Alternately Making a 1-step through 10-step forecast based on MA

predict(MA,n.ahead=10)

#Plotting the m.deciles08 series plus the forecast and 95% prediction intervals

ts.plot(m.deciles08, xlim = c(1949, 1961))

MA\_forecasts <- predict(MA, n.ahead = 10)$pred

MA\_forecast\_se <- predict(MA, n.ahead = 10)$se

points(MA\_forecasts, type = "l", col = 2)

points(MA\_forecasts - 2\*MA\_forecast\_se, type = "l", col = 2, lty = 2)

points(MA\_forecasts + 2\*MA\_forecast\_se, type = "l", col = 2, lty = 2)

#Choosing AR or MA: Exploiting ACF plots

# Find correlation between AR\_fit and MA\_fit

cor(AR\_fit, MA\_fit)

# Find AIC of AR

AIC(AR)

# Find AIC of MA

AIC(MA)

# Find BIC of AR

BIC(AR)

# Find BIC of MA

BIC(MA)