# Homework 3

*Nikhil Guruji*

*November 11, 2017*

1-

To get the coefficient of P2 in the multi regression (Sales vs P1 and P2), we will need to get the part of P2 that is not correlated to P1. That can be obtained by the following expression:

$P2 = a_0 + a_1 * P1 + e_2$

Where $e_2$ is the part of P2 that is uncorrelated to P1 and $a_1$ is the part of P2 that is correlated to P1. So, we will need to perform two single regressions in order to get the coefficient of P2 on Sales in a multi regression:

I) Regress P1 on P2 and get the residual term
II) Regress the residual term on Sales and get the coefficient

```
library(DataAnalytics)
data("multi")
p1_on_p2<-lm(multi$p2~multi$p1)
e2<-p1_on_p2$residuals
res_on_sales<-lm(multi$Sales~e2)
p2_net_effect<-res_on_sales$coefficient[2]
names(p2_net_effect)<-c("P2 coefficient in multiregression")
p2_net_effect
```

```
## P2 coefficient in multiregression
##                          108.7999
```

2-

Using the formulae:

$b = (X'X)^{-1} * (X'Y)$

and

$Var(b) = \sigma^2 * (X'X)^{-1}$

```
data("countryret")
y=countryret$usa
X=cbind(rep(1,length(y)),countryret$canada,countryret$uk,countryret$australia,countryret$france,countryr
x_prim_x_inv<-chol2inv(chol(crossprod(X)))

#calculating b

b=x_prim_x_inv%*%crossprod(X,y)

#calculating standard errors
e<- y - X%*%b
ssq<-sum(e*e)/(length(y)-ncol(X))
varb<-ssq*x_prim_x_inv
se<-sqrt(diag(varb))
names(se)<-c("Intercept","CAN","UK","AUS","FRA","GER","JPN")

cbind(b,se)
```

```
##                               se
## Intercept  0.006135614 0.00230897
## CAN        0.444362109 0.06958673
## UK         0.225690196 0.06491489
## AUS       -0.056688434 0.05036627
## FRA        0.166742081 0.06133779
## GER       -0.064792831 0.05723881
## JPN       -0.051027942 0.03461495
```

3-

a)

As per lecture notes, the prediction interval is:

$b_0 + b_1 X_f \pm t^*_{N-1,\alpha/2} s_{pred}$

where

$s_{pred} = s\left[1 + \frac{1}{N} + \frac{(X_f - \bar{X})^2}{(N-1)s_x^2}\right]^{0.5}$

```
library(reshape2)
data("marketRf")
data("Vanguard")
Van=Vanguard[,c(1,2,5)]
V_reshaped=dcast(Van,date~ticker,value.var="mret")
Van_mkt=merge(V_reshaped,marketRf,by="date")
x<-Van_mkt$vwretd
out<-lm(VWNFX ~ vwretd,data=Van_mkt)
#for standard error of regression, s
anovatable<-anova(out)
sse<-anovatable[2,2]
N<-length(Van_mkt$vwretd)
s<-round(sqrt(sse/(N-2)),3)
xf<-0.05
t_90<-qt(0.95,df=N-2)
#calculating spred
spred<-s*((1+(1/N) + ((xf-mean(x))^2)/((N-1)*(sd(x))^2))^0.5)
b_0<-out$coefficients[1]
b_1<-out$coefficients[2]
#calculating the prediction interval
pred_interval<-c(b_0+(b_1*xf)-t_90*spred,b_0+(b_1*xf)+t_90*spred)
names(pred_interval)<-c("Lower Limit","Upper Limit")
pred_interval
```

```
## Lower Limit Upper Limit
##  0.01742059  0.07364052
```

```
predict(out,new=data.frame(vwretd=.05),int="prediction",level=0.90)
```

```
##          fit        lwr        upr
## 1 0.04553055 0.01744646 0.07361465
```

4-

```
#part a

mu<-matrix(c(0.01,0.015,0.025),ncol=1)
vc_mat<-matrix(c(0.0016,0.001,0.0015,0.0010,0.0020,0.0019,0.0015,0.0019,0.0042),nrow=3,ncol=3)
```

```
sigma<-sqrt(diag(vc_mat))
corr_mat<-matrix(c(rep(0,9)),nrow=3,ncol=3)
for(i in 1:3){
  for(j in 1:3){
  corr_mat[i,j]=vc_mat[i,j]/(sigma[i]*sigma[j])
  }
}
corr_mat
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.5590170 0.5786376
## [2,] 0.5590170 1.0000000 0.6555623
## [3,] 0.5786376 0.6555623 1.0000000
```

```
#part b

weights<-c(0.3,0.4,0.3)
portfolio_mean<-weights%*%mu
portfolio_sd<-(weights%*%vc_mat)%*%(weights)
portfolio_mean
```

```
##        [,1]
## [1,] 0.0165
```

```
sqrt(portfolio_sd)
```

```
##            [,1]
## [1,] 0.04252058
```

5-

```
mkt_up=ifelse(Van_mkt$vwretd>0,1,0)
Van_mkt$upvw=mkt_up*Van_mkt$vwretd
Van_mkt$dwnvw=(1-mkt_up)*Van_mkt$vwretd
mkt_timing=lm(VWNFX~upvw+dwnvw,data=Van_mkt)
R=matrix(c(0,1,-1),byrow=TRUE,nrow=1)
r=c(0)
X=cbind(c(rep(1,nrow(Van_mkt))),Van_mkt$upvw,Van_mkt$dwnvw)
b=as.vector(mkt_timing$coef)
QFmat=chol2inv(chol(crossprod(X)))
QFmat=R%*%QFmat%*%t(R)
Violation=R%*%b-matrix(r,ncol=1)
fnum=t(Violation)%*%chol2inv(chol(QFmat))%*%Violation
n_minus_k = nrow(Van_mkt)-length(b)
fdenom=nrow(R)*sum(mkt_timing$resid**2)/n_minus_k
f=fnum/fdenom
f
```

```
##           [,1]
## [1,] 0.1708104
```

```
pvalue=1-pf(f,df1=nrow(R),df2=n_minus_k)
pvalue
```

```
##           [,1]
## [1,] 0.6796486
```

As p value is higher than 5% or even 10%, we fail to reject the null that $\beta_{up} = \beta_{down}$

Hence, we conclude that $\beta_{up} = \beta_{down}$

6-

```
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Version 0.4-0 included new data defaults. See ?getSymbols.
```

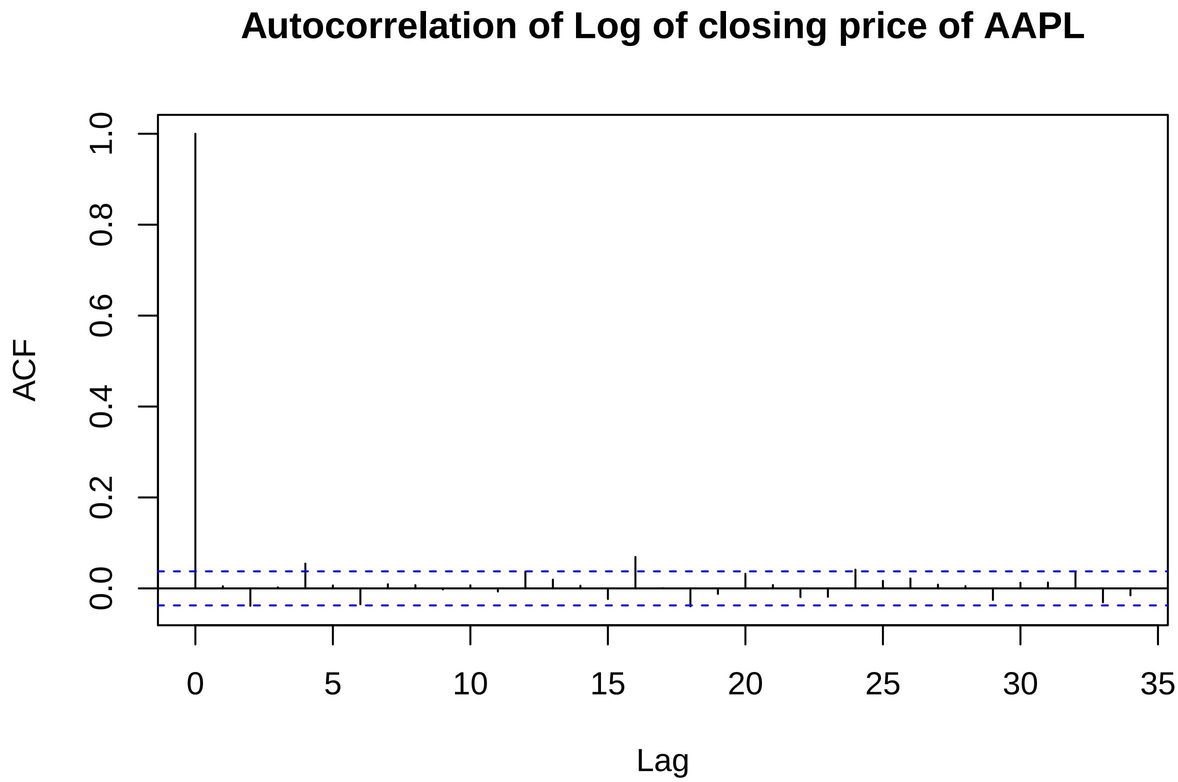
```
getSymbols("AAPL")
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.

##
## WARNING: There have been significant changes to Yahoo Finance data.
## Please see the Warning section of '?getSymbols.yahoo' for details.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.yahoo.warning"=FALSE).

## [1] "AAPL"
```

```
plot(acf(diff(log(AAPL$AAPL.Close)),na.action=na.omit,plot=FALSE),main="Autocorrelation of Log of closi
```


Autocorrelation of Log of closing price of AAPL
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
35
Lag