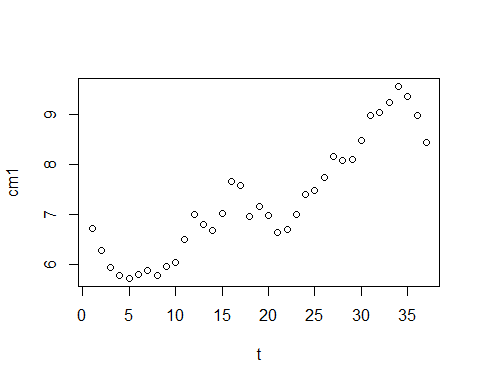
Exp 8-5 Edit

Hao Li

2019-02-14

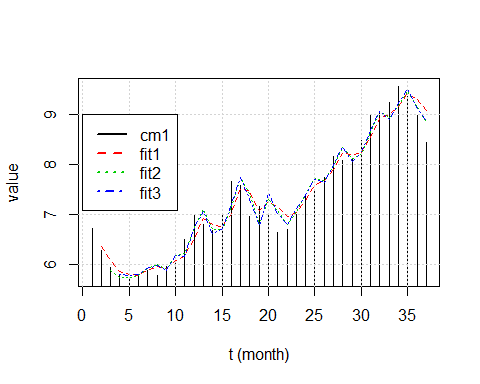
# Example 8.6 ARM problem  
  
#1. Read and preview  
# read data (change path for your computer)  
cm1=scan("cm1.txt")  
n=length(cm1)  
t=seq(1,n,1)  
# plot data  
#par(mfcol=c(2,2))  
plot(t,cm1)



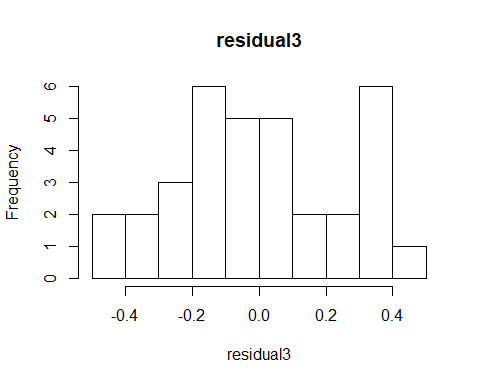
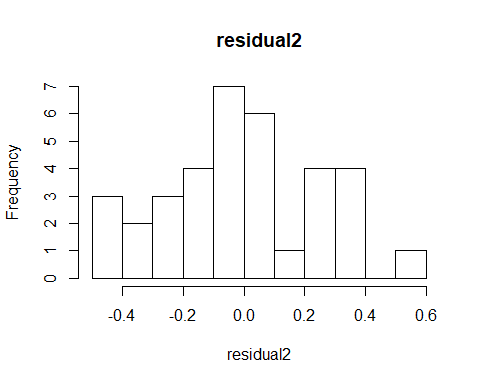
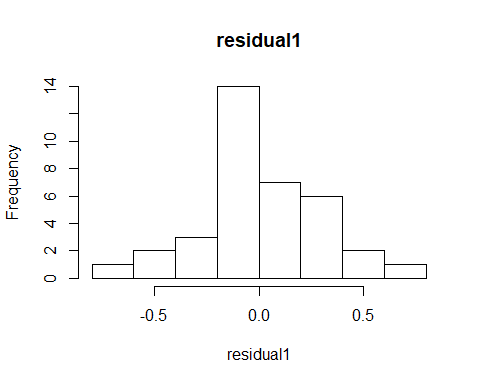
#2. Data manipulation  
#repeat for Autoregression involving 1:3 previous terms  
#(to avoid copying and pasting lag1=...,fit1...)  
  
cm1df = data.frame(t=t,cm1=cm1)  
cm1df = within(cm1df,{  
 for(i in 1:3) assign(paste0('lag',i),c(rep(NA,i),cm1[1:(n-i)]))#assign lag1:lag3  
 remove(i)  
})  
#for(i in 1:3) assign(paste0('lag',i),c(rep(NA,i),cm1[1:(n-i)]))#assign lag1:lag3  
  
fit1 = lm(cm1~t+lag1,data=cm1df)  
fit2 = lm(cm1~t+lag1+lag2,data = cm1df)  
fit3 = lm(cm1~t+lag1+lag2+lag3,data = cm1df)  
  
cm1df = within(cm1df,{  
 for(i in 1:3) assign(paste0('fit',i),c(rep(NA,i),predict.lm(eval(parse(text = paste0('fit',i))))))#assign lag1:lag3  
 remove(i)  
})  
cm1df =within(cm1df,{  
 for(i in 1:3) assign(paste0('res',i),cm1 - get(paste0('fit',i)))#assign lag1:lag3  
 remove(i)  
})  
#cm1df  
head(cm1df)

## t cm1 lag3 lag2 lag1 fit3 fit2 fit1 res3  
## 1 1 6.73 NA NA NA NA NA NA NA  
## 2 2 6.27 NA NA 6.73 NA NA 6.359942 NA  
## 3 3 5.93 NA 6.73 6.27 NA 5.875437 6.072003 NA  
## 4 4 5.77 6.73 6.27 5.93 5.788715 5.733346 5.867784 -0.018714877  
## 5 5 5.72 6.27 5.93 5.77 5.775334 5.737694 5.789145 -0.055333863  
## 6 6 5.80 5.93 5.77 5.72 5.794679 5.785128 5.787249 0.005321005  
## res2 res1  
## 1 NA NA  
## 2 NA -0.08994206  
## 3 0.05456274 -0.14200271  
## 4 0.03665432 -0.09778354  
## 5 -0.01769442 -0.06914464  
## 6 0.01487194 0.01275077

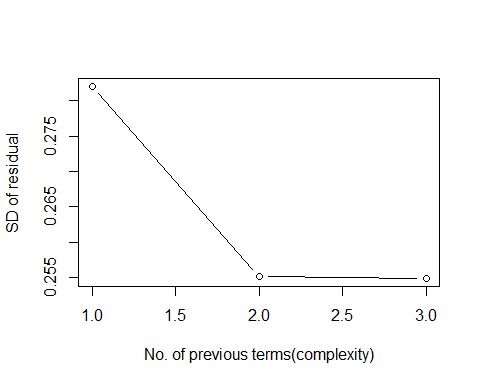
layout(1)  
cm1pred = c('cm1','fit1','fit2','fit3')  
matplot(x=cm1df[,'t'],y=cm1df[,cm1pred],type = c('h','l','l','l'),lty = 1:4,col =1:4,  
 xlab = 't (month)',ylab = 'value')  
grid()  
legend(0,9,legend = cm1pred,lwd=2,col = 1:4,lty = 1:4)



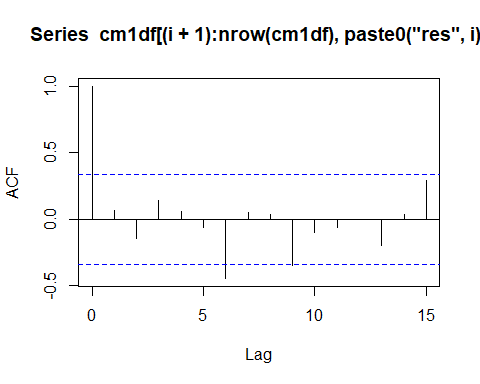
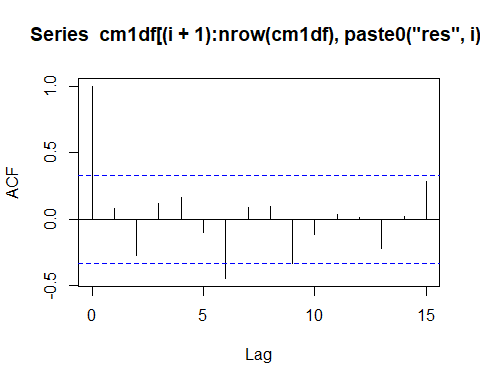
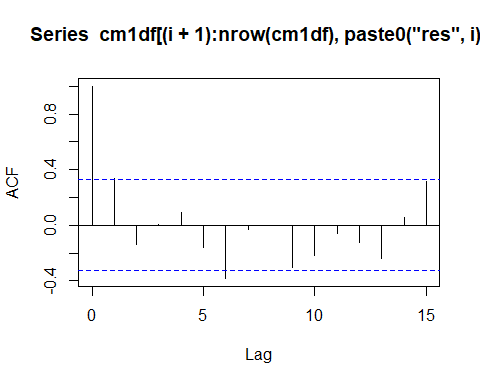
#layout(1:3)  
for(i in 1:3) hist(cm1df[,as.character(paste0('res',i))],xlab = paste0('residual',i),main = paste0('residual',i))



layout(1)  
sd\_error=numeric(3)  
for(i in 1:3) sd\_error[i] = sd(cm1df[,as.character(paste0('res',i))],na.rm =T)  
plot(c(1:3), sd\_error,  
 xlab = 'No. of previous terms(complexity)',ylab = 'SD of residual',type = 'b')



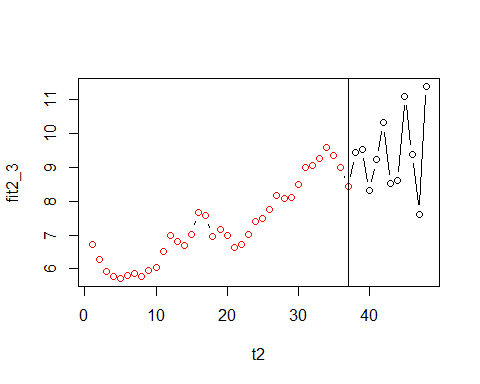
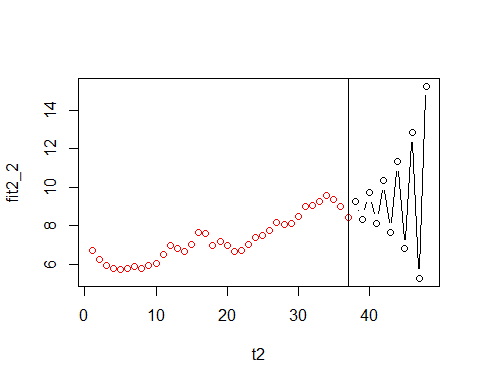
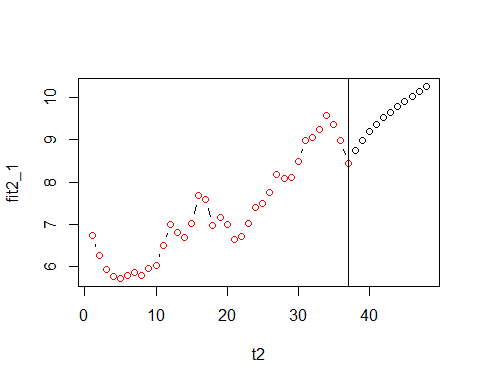
#plot shows how residuals get lowered at the expense of greater complexity  
  
#uncomment the code below to avoid copying and pasting  
#!!!Note that this is usually not encouraged because it is unintuitive  
#(to avoid copying and pasting lag1=...,fit1...)  
#for(i in 1:3){  
# indp = paste0('lag',1:i)  
# sapply(indp,paste,sep = "+")  
# assign(paste0('fit',i),lm(as.formula(paste('cm1 ~ t +',indp))))  
#}  
# for(i in 1:3) print(summary(eval(parse(text = paste0('fit',i)))))  
  
  
# plot autocovariance function for these residuals  
#require(graphics)  
#layout(1:3)  
for(i in 1:3) acf(cm1df[(i+1):nrow(cm1df),paste0('res',i)])



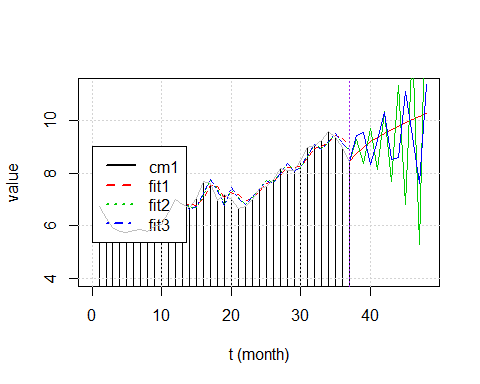
#acf(e1)  
  
  
# forecast cm1 for May 1990  
for(i in 1:3) {  
 assign(paste0('coef',i),coef(get(paste0('fit',i))))  
 print(get(paste0('coef',i)))  
 }

## (Intercept) t lag1   
## 1.5986595 0.0329880 0.6976681   
## (Intercept) t lag1 lag2   
## 1.78482453 0.03300389 1.10359733 -0.43506006   
## (Intercept) t lag1 lag2 lag3   
## 1.63315690 0.03188948 1.17110722 -0.64645712 0.16888861

nf=11#n of forcast  
t2=seq(1,n+nf,1)  
for(i in 1:3){  
 #for every model  
 cm1f = c(cm1,numeric(nf))  
 #for(k in 1:i){  
 # #for every lag  
 # coeflag = c(coeflag,get(paste0('coef',k))[paste0('lag',k)])  
 #}  
 for(j in n+1:nf){  
 #for every ts term  
   
  
 #cm1flag = NULL  
 #for(k in 1:i){  
 #for every lag  
 cm1flag= cm1f[(j-i):(j-1)]  
 cm1f[j] = sum(get(paste0('coef',i))\*c(1,t2[j],cm1flag))  
 }  
 assign(paste0('cm1f',i),cm1f)  
 #fit2 = numeric(length(cm1f))  
 #for(j in n+i:nf){  
 # fit2[j] = sum(get(paste0('coef',i))\*c(1,t2[j],cm1f[(j-i+1):(j)]))  
 #}  
 assign(paste0('fit2\_',i),fit2)  
}  
#layout(1:3)  
for(i in 1:3){  
 plot(t2,get(paste0('cm1f',i)),ylab=paste0('fit2\_',i), type ='b')  
 points(t,cm1,col = 'red')  
 abline(v = n)  
}



layout(1)  
cm1pred = c('cm1','fit1','fit2','fit3')  
matplot(x=cm1df[,'t'],y=cm1df[,cm1pred],type = c('h','l','l','l'),lty = 1:4,col =1:4,  
 xlab = 't (month)',ylab = 'value',  
 xlim = c(0,n+nf),  
 ylim = c(min(cm1) - 1.5\*sd(cm1),  
 (max(cm1)+1.5\*sd(cm1))))  
grid()  
legend(0,9,legend = cm1pred,lwd=2,col = 1:4,lty = 1:4)  
for(i in 1:3){  
 lines(t2,get(paste0('cm1f',i)),ylab=paste0('fit2\_',i), type ='l',col = i+1)  
 #points(t,cm1,col = 'red')  
 abline(v = n,col = 'purple',lty = 3)  
}  
lines(t,cm1,col = 'grey')



for(i in 1:3) {  
 print(i)  
 print(get(paste0('cm1f',i)))  
}

## [1] 1  
## [1] 6.730000 6.270000 5.930000 5.770000 5.720000 5.800000 5.870000  
## [8] 5.780000 5.960000 6.030000 6.500000 7.000000 6.800000 6.680000  
## [15] 7.030000 7.670000 7.590000 6.960000 7.170000 6.990000 6.640000  
## [22] 6.710000 7.010000 7.400000 7.490000 7.750000 8.170000 8.090000  
## [29] 8.110000 8.480000 8.990000 9.050000 9.250000 9.570000 9.360000  
## [36] 8.980000 8.440000 8.740522 8.983175 9.185455 9.359566 9.514027  
## [43] 9.654777 9.785961 9.910473 10.030328 10.146936 10.261277  
## [1] 2  
## [1] 6.730000 6.270000 5.930000 5.770000 5.720000 5.800000 5.870000  
## [8] 5.780000 5.960000 6.030000 6.500000 7.000000 6.800000 6.680000  
## [15] 7.030000 7.670000 7.590000 6.960000 7.170000 6.990000 6.640000  
## [22] 6.710000 7.010000 7.400000 7.490000 7.750000 8.170000 8.090000  
## [29] 8.110000 8.480000 8.990000 9.050000 9.250000 9.570000 9.360000  
## [36] 8.980000 8.440000 9.277369 8.350125 9.710655 8.128442 10.351280  
## [43] 7.671090 11.323256 6.809497 12.836778 5.266181 15.244541  
## [1] 3  
## [1] 6.730000 6.270000 5.930000 5.770000 5.720000 5.800000 5.870000  
## [8] 5.780000 5.960000 6.030000 6.500000 7.000000 6.800000 6.680000  
## [15] 7.030000 7.670000 7.590000 6.960000 7.170000 6.990000 6.640000  
## [22] 6.710000 7.010000 7.400000 7.490000 7.750000 8.170000 8.090000  
## [29] 8.110000 8.480000 8.990000 9.050000 9.250000 9.570000 9.360000  
## [36] 8.980000 8.440000 9.426756 9.529363 8.308289 9.223218 10.319165  
## [43] 8.514676 8.604805 11.101913 9.384021 7.617056 11.385450