Chapter 8 Autocorrelation Assignment

Petra Guy

October 22, 2017

1 Introduction

I wrote three scripts (so far!). Initially I wrote a python-ish functions script, but I wanted to make this run using Rscript, which I couldnt. I therefore rewrote I long ugly code, thinking that if I ran that from Rscript it would simple run through line by line. But the load command would not work. I also checked the results of my calculations against the R acf() function, and they were the same, so I wrote another script using that.

2 Graphs

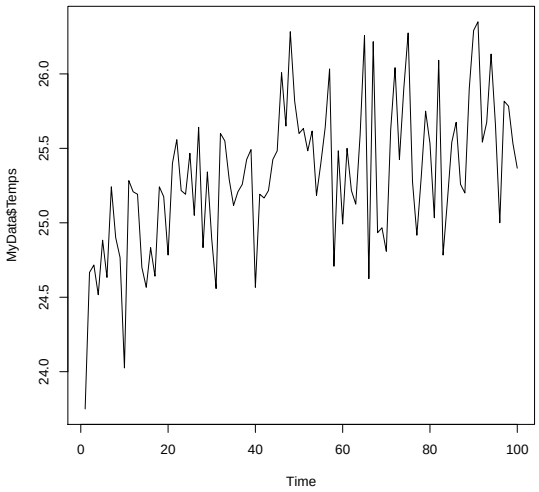

Figure 1: Time series plot.

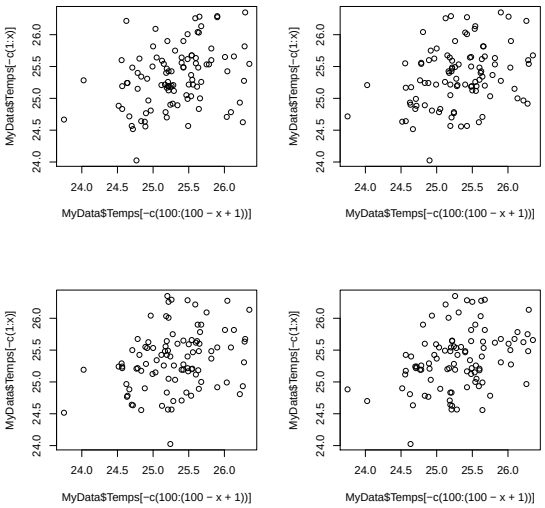

Figure 2: Scatter plots of years for lags of 1 to 4 years.

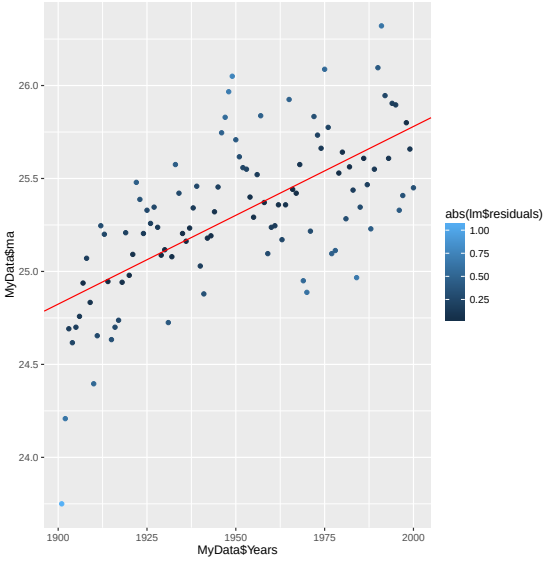

Figure 3: Plot of moving averages and fit.

3 Code

```
#!/usr/bin/env Rscript
#Chapter 8.8 Final Practical

rm(list = ls())
library(stats) # for plot.ts
library(ggplot2)
library(pracma) # for movavg

# get the data and make two vectors coz easier for loops

load("../Data/KeyWestAnnualMeanTemperature.RData")
Years = ats[[1]]
Temps = ats[[2]]
MyData = as.data.frame(cbind(Years,Temps))


#Examine = function(Data){
  # plot simple time seriesgraphics.off()

pdf("../Results/TAutocorrtimeseries1.pdf")

plot.ts(MyData$Temps)
  #plot terms with lag of 1 to 4 years against each other M Crawley p787
dev.off()

pdf("../Results/TAutocorrtimeseries2.pdf")
par(mfrow = c(2,2))
sapply(1:4, function(x) plot(MyData$Temps[-c(100:(100-x+1))], MyData$Temps[-
c(1:x)]))
dev.off()

#autocorrelation coef is Sum(Y[i+1]-AveY)(Y[i] -AveY)/sum(sqr(Y[i]-AveY))


# calculate sum[(Yi+1 - ave)(Yi - ave)] -- numerator of autocorr coef
#Calc_numerator = function(avector){
num = vector("numeric",99)
for (i in seq_along((Temps))) {
  if (i <100) {
  num[i] = as.vector( Temps[i+1] - mean(Temps) ) * ( Temps[i] - mean(Temps) )
  }
 else
 totalnum = sum(num)}



#Calc_denom = function(avector){
#  calcuate sum[(Yi - ave)^2] -- denom of auto corr coef
denom = vector("numeric",99)
for (i in seq_along((Temps))) {
  if (i <100) {
    denom[i] = as.vector(( Temps[i] - mean(Temps) )^2)
  }
  else
    totaldenom = sum(denom)}


#Calc_acf = function(avector){
   autocorrcoef = totalnum/totaldenom
    print("autocorrelatoin coefficient for lag 1 is ")
    print(autocorrcoef)
```

Figure 4: Longwinded and ugly without functions.

```
#!/usr/bin/env Rscript
#Chapter 8.8 Final Practial


prepare_workspace = function(){
  rm(list = ls())
  library(stats) # for plot.ts
  library(ggplot2)
  library(pracma) # for movavg
  graphics.off()
}

# get the data and make two vectors coz easier for loops

GetData = function(){
  load("../Data/KeyWestAnnualMeanTemperature.RData")
  Years = ats[[1]]
  Temps = ats[[2]]
  MyData = as.data.frame(cbind(Years,Temps))
  return(MyData)
}

Examine = function(Data){
  # plot simple time series
  plot.ts(Data)
  #plot terms with lag of 1 to 4 years against each other M Crawley p787
  par(mfrow = c(2,2))
  sapply(1:4, function(x) plot(Data[-c(100:(100-x+1))], Data[-c(1:x)]))
}

#autocorrelation coef is Sum(Y[i+1]-AveY)(Y[i] -AveY)/sum(sqr(Y[i]-AveY))


# calculate sum[(Yi+1 - ave)(Yi - ave)] -- numerator of autocorr coef
Calc_numerator = function(avector){
num = vector("numeric",99)
for (i in seq_along((avector))) {
  if (i <100) {
  num[i] = as.vector( avector[i+1] - mean(avector) ) * ( avector[i] -
mean(avector) )
  }
 else
 totalnum = sum(num)
}
return(totalnum)}


Calc_denom = function(avector){
#  calcuate sum[(Yi - ave)^2] -- denom of auto corr coef
denom = vector("numeric",99)
for (i in seq_along((avector))) {
  if (i <100) {
    denom[i] = as.vector(( avector[i] - mean(avector) )^2)
  }
  else
    totaldenom = sum(denom)
}
return(totaldenom)}


Calc_acf = function(avector){
  num = Calc_numerator(avector)
  denom = Calc_denom(avector)
  autocorrcoef = num/denom
```

Figure 5: With functions.

```
#!/usr/bin/env Rscript
#Chapter 8.8 Final Practial


prepare_workspace = function(){
  rm(list = ls())
   graphics.off()
   library(dplyr)
}

# get the data and make two vectors coz easier for loops


  load("../Data/KeyWestAnnualMeanTemperature.RData")
  Years = ats[[1]]
  Temps = ats[[2]]
  MyData = as.data.frame(cbind(Years,Temps))


  # plot simple time series

  autocorr = acf(Temps,1)[[1]][2]

  print("autocorrelation coefficient for lag of 1 is ")
  print(autocorr)

  # repeating with the 1000 samples in a for loop
  acfs = vector("numeric",1000)
  for (i in 1:1000){
    acfs[i] = acf(sample(Temps,100))[[1]][2]
  }


  # reapeting using piping - but since acf gives a list, the answer is messh=y,
hnce unlist and select out alternate values
  acfs2 =vector("numeric",1000)
  for (i in 1:1000){
    acfs2[i] <- Temps %>% sample(.,100) %>% acf(.,1)
  }
acfs2 = unlist(acfs2[c(FALSE,TRUE)])
```

Figure 6: Using acf() and pipes.