Project 1

Zijing Gao

November 4, 2019

```
# Here, I need to import some useful packages in case I need them
library(TSA)

##
## Attaching package: 'TSA'

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar

library(lmtest)

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

library(psych)
library(ggplot2)

##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha

library(pracma)

##
## Attaching package: 'pracma'

## The following objects are masked from 'package:psych':
##
##     logit, polar

library(forecast)
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.Arima       TSA
##   fitted.fracdiff    fracdiff
##   plot.Arima         TSA
##   residuals.fracdiff fracdiff
```

```r
library(yarrr)
```

```
## Loading required package: jpeg

## Loading required package: BayesFactor

## Loading required package: coda

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:pracma':
##
##     expm, lu, tril, triu

## ************
## Welcome to BayesFactor 0.9.12-4.2. If you have questions, please contact R
ichard Morey (richarddmorey@gmail.com).
##
## Type BFManual() to open the manual.
## ************

## Loading required package: circlize

## ========================================
## circlize version 0.4.8
## CRAN page: https://cran.r-project.org/package=circlize
## Github page: https://github.com/jokergoo/circlize
## Documentation: http://jokergoo.github.io/circlize_book/book/
##
## If you use it in published research, please cite:
## Gu, Z. circlize implements and enhances circular visualization
##   in R. Bioinformatics 2014.
## ========================================
```

```
## yarrr v0.1.5. Citation info at citation('yarrr'). Package guide at yarrr.g
uide()

## Email me at Nathaniel.D.Phillips.is@gmail.com

##
## Attaching package: 'yarrr'

## The following object is masked from 'package:ggplot2':
##
##     diamonds
```

```r
library(DAAG)
```

```
## Loading required package: lattice

##
## Attaching package: 'DAAG'

## The following object is masked from 'package:psych':
##
##     cities
```

Preparation

When I get the data, I think the first step is to clean and manage the data.

(1) read the data

```r
# explainary

# Labor force participation (monthly) start = 1976
LFP = read.csv("LBSSA08.csv",header = T)

# Unemployment Rate in Colorado (monthly) start = 1976
COUR = read.csv("COURN.csv",header = T)

# Dividends, Interest and Rent in Colorado (quarterly) start = 1948
DIR = read.csv("COODIV.csv",header = T)

# response

# New Private Housing Units Authorized by Building Permits for Colorado (mont
hly) start = 1988
COBP = read.csv("COBPPRIV.csv",header = T)

# New Private Housing Units Authorized by Building Permits for US (monthly) 1
959
PERMIT = read.csv("PERMIT.csv",header = T)
```

(2) data cleansing and fetching

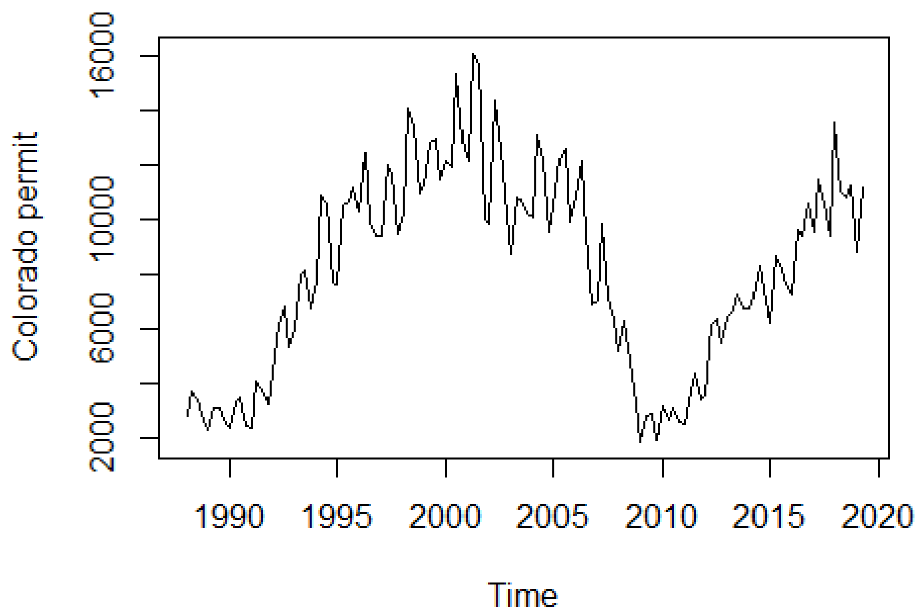
```r
# data fetching

# at first, I change all of the datasets to be ts.


LFP_ts = ts(LFP$LBSSA08,start = c(1976,1),frequency = 12)
COUR_ts = ts(COUR$COUR,start = c(1976,1),frequency = 12)
DIR_ts = ts(DIR$COODIV,start = c(1948,1),frequency = 4)

COBP_ts = ts(COBP$COBPPRIV,start = c(1988,1),frequency = 12)
PER_ts = ts(PERMIT$PERMIT,start = c(1960,1),frequency = 12)

# I transform monthly data to quarterly data

# I set the start date to be 1988 Jan

# filter the data to make it start at 1988 Jan
# Then, transform monthly to quarterly by summing the data of every 3 months
to 1 qtr.
LFP_ts = window(LFP_ts,start = 1988)
LFP_ts <- aggregate(LFP_ts, nfrequency = 4)
LFP_ts = window(LFP_ts,end = 2019.25)

COUR_ts = window(COUR_ts,start = 1988)
COUR_ts <- aggregate(COUR_ts, nfrequency = 4)
COUR_ts = window(COUR_ts,end = 2019.25)

DIR_ts = tail(DIR_ts,(2019-1988)*4+2)


COBP_ts = window(COBP_ts,start = 1988)
COBP_ts <- aggregate(COBP_ts, nfrequency = 4)
COBP_ts = window(COBP_ts,end = 2019.25)

PER_ts = window(PER_ts,start = 1988)
PER_ts <- aggregate(PER_ts, nfrequency = 4)
PER_ts = window(PER_ts,end = 2019.25)

explainary = list("LFP" = LFP_ts,
                  "COUR" = COUR_ts,
                  "DIR" = DIR_ts)

response = list("COBP" = COBP_ts,
                "PER" = PER_ts)
```

Question 1

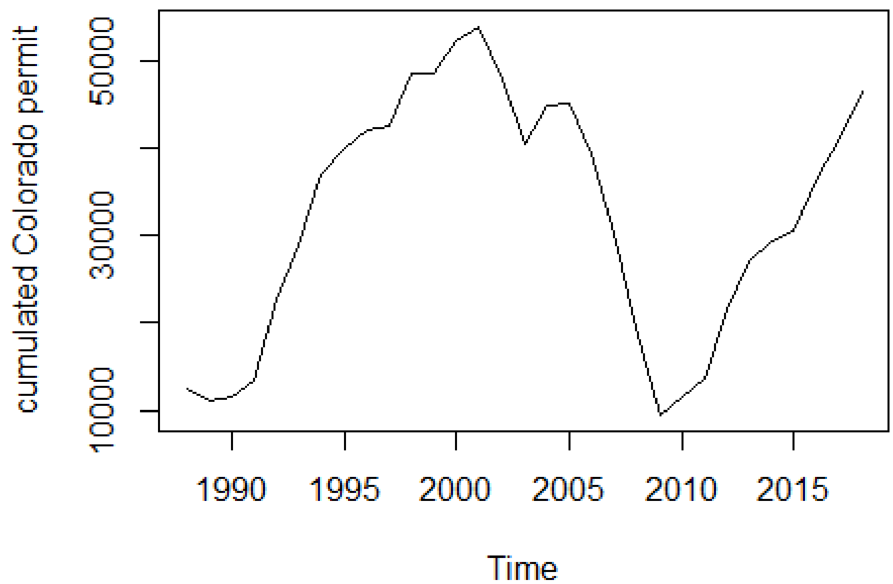
Describe the attributes of New Private Housing Units Authorized by Building Permits for the state you are assigned to. (Colorado)

To describe a dataset, I first want to plot it and detect the features from the plot at the first sight and do some regular calculation on it.

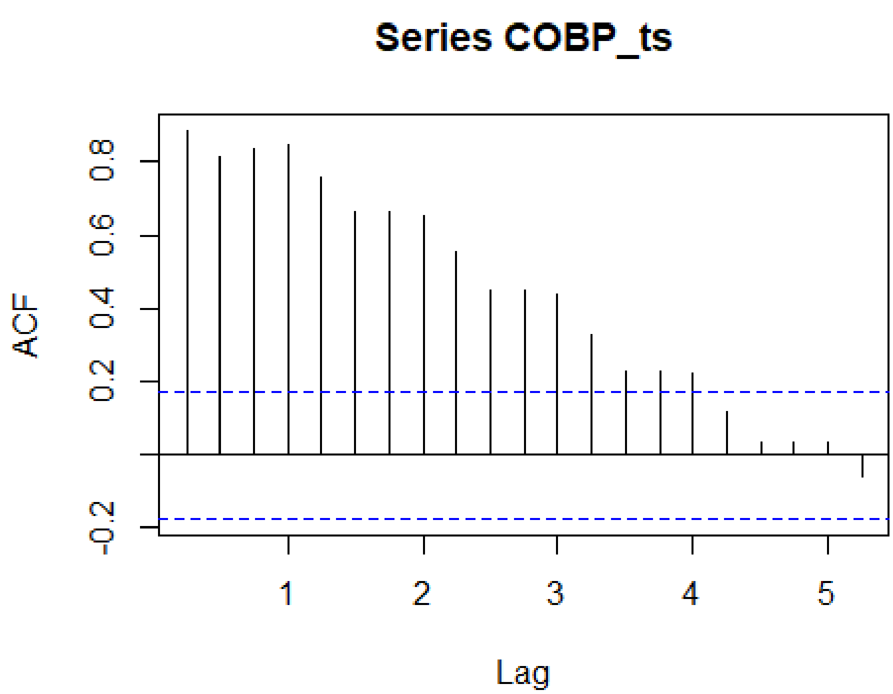
```
# plot the COBP_TS
plot(COBP_ts,ylab = "Colorado permit")
```




```
# from the plot, I find that it is seasonal
# the cycle seems to be 30 years, which contain 2 seasonal trends

describe(COBP_ts)

##    vars   n   mean      sd median trimmed    mad  min   max range  skew
## X1    1 126 8094.7 3666.13 8475.5 8079.72 3868.1 1846 16068 14222 -0.08
##    kurtosis     se
## X1    -1.07 326.61

# Here I find the basical statistical features.

# cumulated COBP
plot(aggregate(COBP_ts), ylab="cumulated Colorado permit")
```

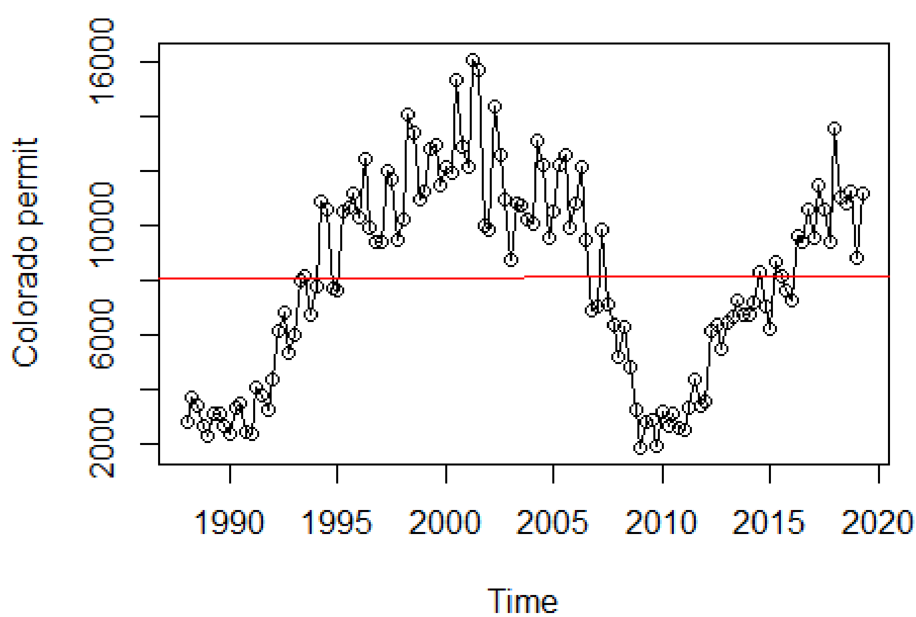



```r
# acf
acf(COBP_ts)
```



Then, I want to describe the attribute of the data with respect to time shifts. Let us see what will happen if I do the regression analysis.

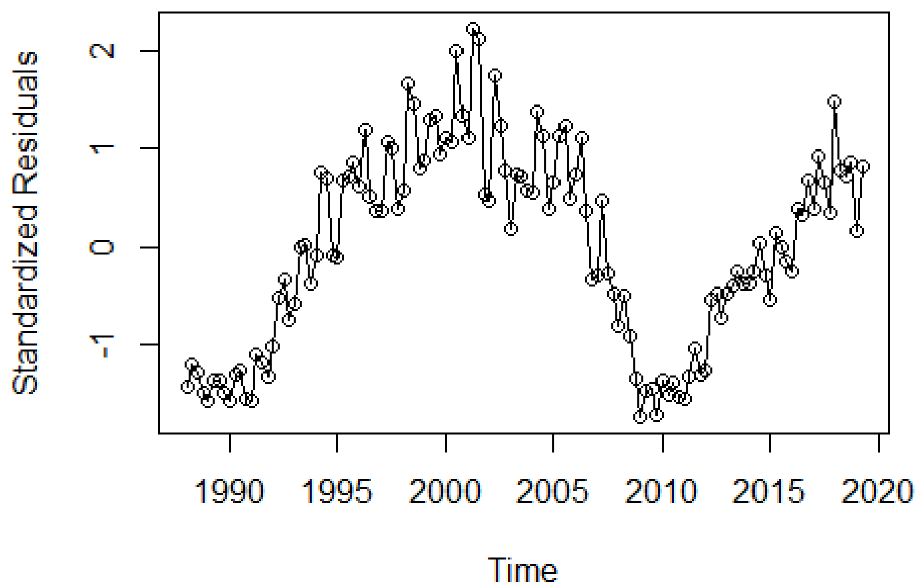
```
fit_COBP = lm(COBP_ts~time(COBP_ts)-1)

summary(fit_COBP)

##
## Call:
## lm(formula = COBP_ts ~ time(COBP_ts) - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6272.4 -3204.4   333.4  2778.3  7980.9
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## time(COBP_ts)   4.0410     0.1628   24.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3662 on 125 degrees of freedom
## Multiple R-squared:  0.8313, Adjusted R-squared:   0.83
## F-statistic: 616.1 on 1 and 125 DF,  p-value: < 2.2e-16

plot(COBP_ts,ylab = "Colorado permit",type = "o")
abline(fit_COBP,col=2)
```



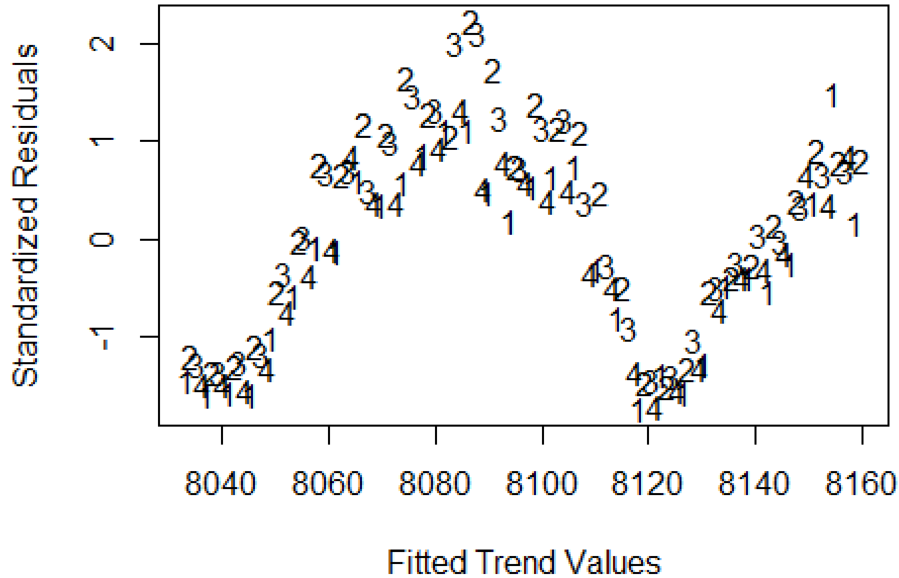
```
# from the plot, we can see that it has a seasonal trend

# And the linear regression does not fit the data well, since the Multiple R-
squared is vert small.
```

Now, let us do the residual analysis.

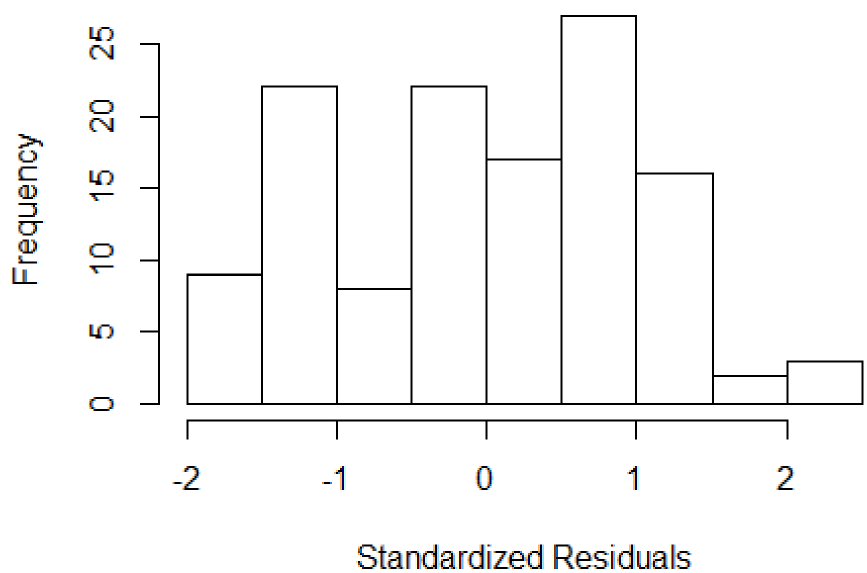
```
#  resdiuals vs. fitted values
plot(y=rstudent(fit_COBP),x=as.vector(time(COBP_ts)),xlab='Time', ylab='Stand
ardized Residuals',type='o')
```



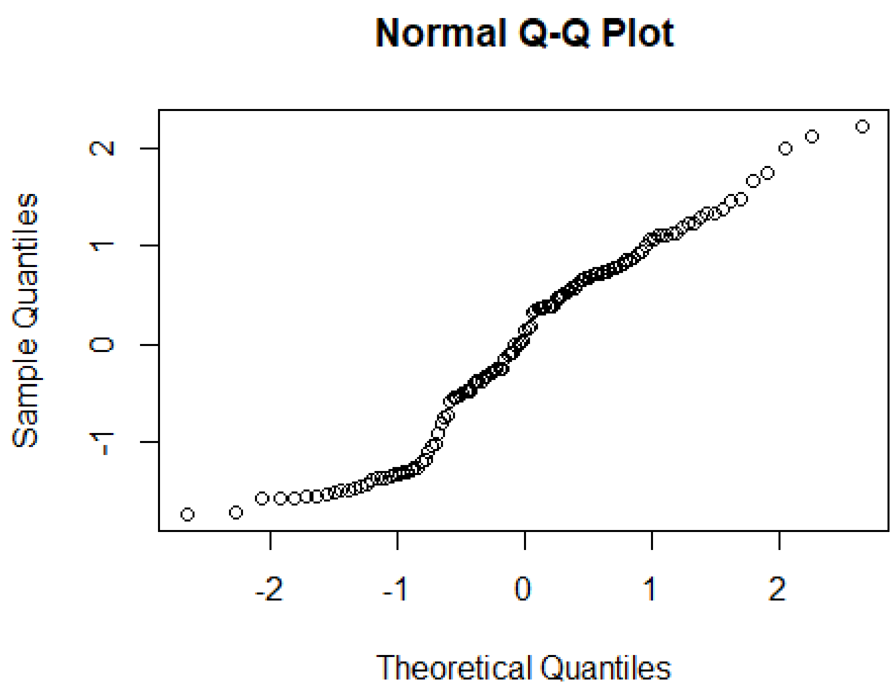

```
#resdiuals vs. fitted values
plot(y=rstudent(fit_COBP),x=as.vector(fitted(fit_COBP)),xlab='Fitted Trend Va
lues',
ylab='Standardized Residuals',type="n")
points(y=rstudent(fit_COBP),x=as.vector(fitted(fit_COBP)),pch=as.vector(seaso
n(COBP_ts)))
```



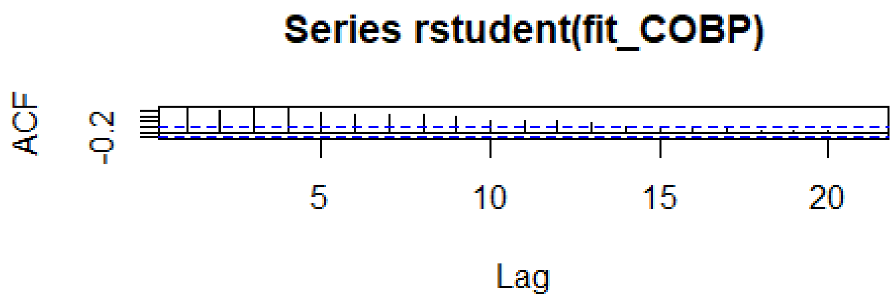

```r
#residual histogram and qqplot
hist(rstudent(fit_COBP),xlab='Standardized Residuals',main='')
```

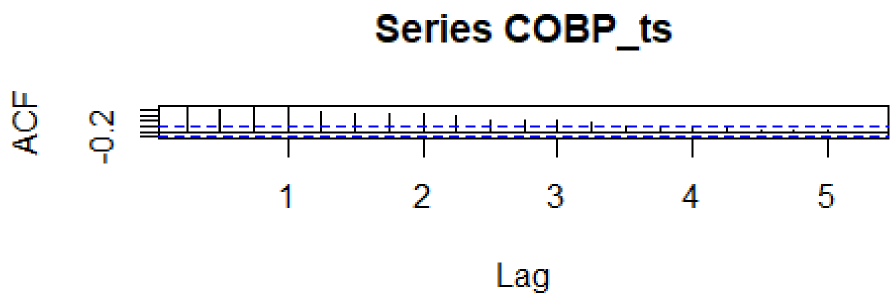


```r
qqnorm(rstudent(fit_COBP))
```




```r
# compare ACFs
par(mfrow=c(2,1))
acf(rstudent(fit_COBP))
acf(COBP_ts)
```



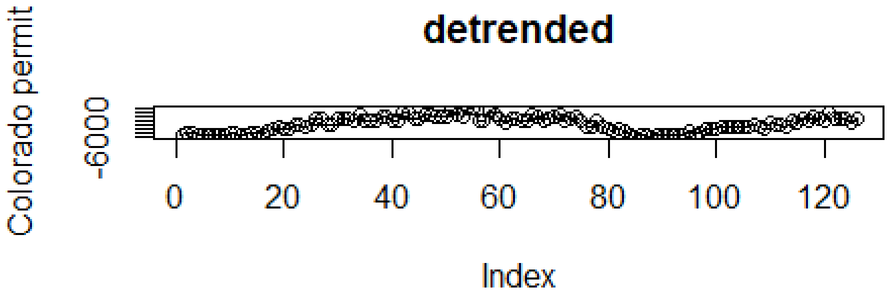


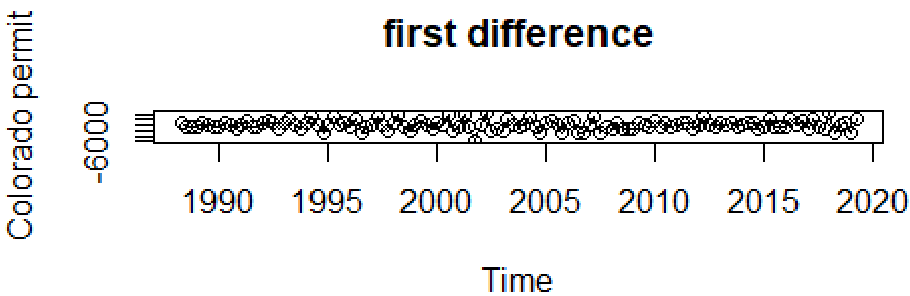
Durbin-Watson Test of Autocorrelation

```
# two-sided alternative
dwtest(fit_COBP)

##
##  Durbin-Watson test
##
## data:  fit_COBP
## DW = 0.20658, p-value < 2.2e-16
## alternative hypothesis: true autocorrelation is greater than 0

# Here, I use Durbin-Watson Test to test autocorrelation parameter
```

Now, let us do the detrending, differencing and Smoothing

```
# Detrending and Differencing

#  plot both detrended and first differenced series

par(mfrow=c(2,1))
plot(resid(fit_COBP), type="o", main="detrended",ylab = "Colorado permit")
plot(diff(COBP_ts), type="o", main="first difference",ylab = "Colorado permit
")
```

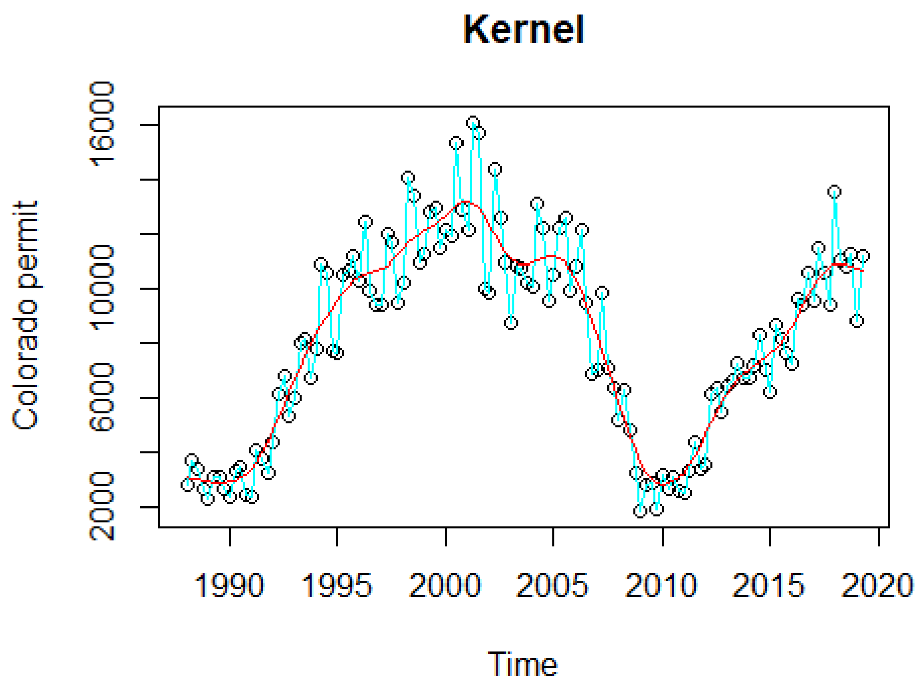




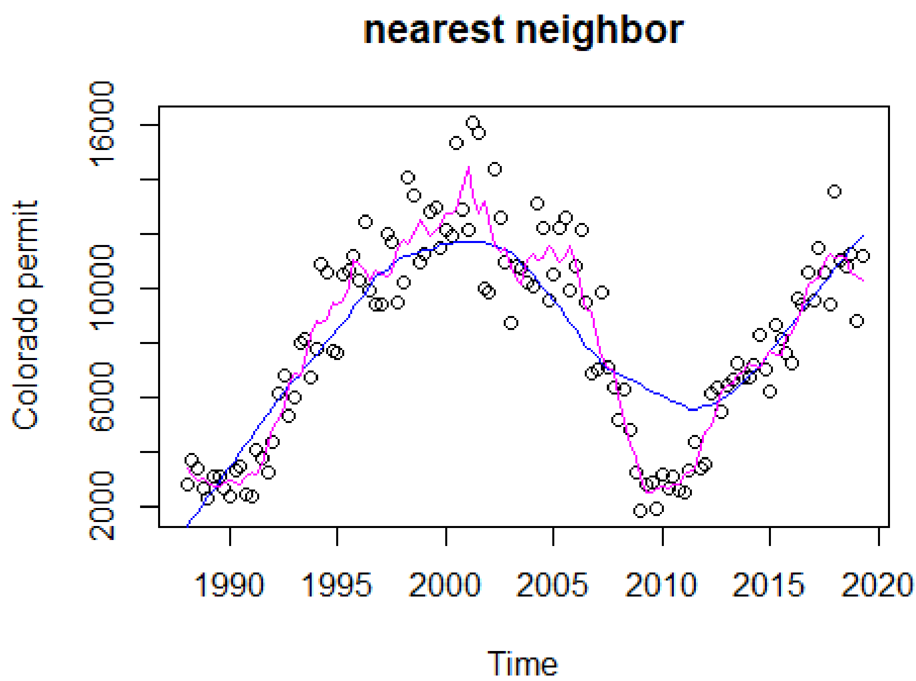

```
# I think differencing better detrend the data.
```

Now, let us use 2 kinds of smoothers to smooth the Colorado permit

```
# Kernel Smoother
plot(COBP_ts, type="p", ylab="Colorado permit",main="Kernel")
lines(ksmooth(time(COBP_ts), COBP_ts, "normal", bandwidth=5/52),col = 5)
lines(ksmooth(time(COBP_ts), COBP_ts, "normal", bandwidth=2),col = 2)
```




```
# Nearest Neighbor Regression
plot(COBP_ts, type="p", ylab="Colorado permit", main="nearest neighbor")
lines(supsmu(time(COBP_ts), COBP_ts, span=.3),col = 100)
lines(supsmu(time(COBP_ts), COBP_ts, span=.01),col = 6)
```

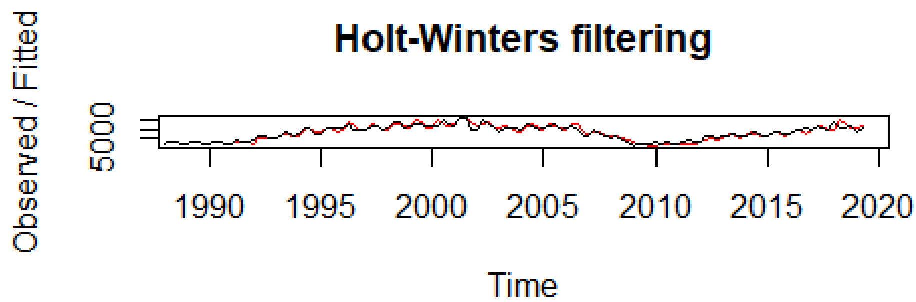


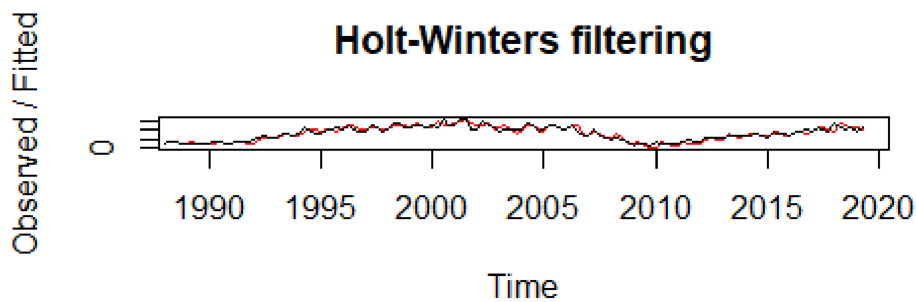
Holt_Winters Method

Holt-Winters exponential smoothing estimates the level, slope and seasonal component at the current time point. Smoothing is controlled by three parameters: alpha, beta, and gamma, for the estimates of the level, slope b of the trend component, and the seasonal component, respectively, at the current time point.

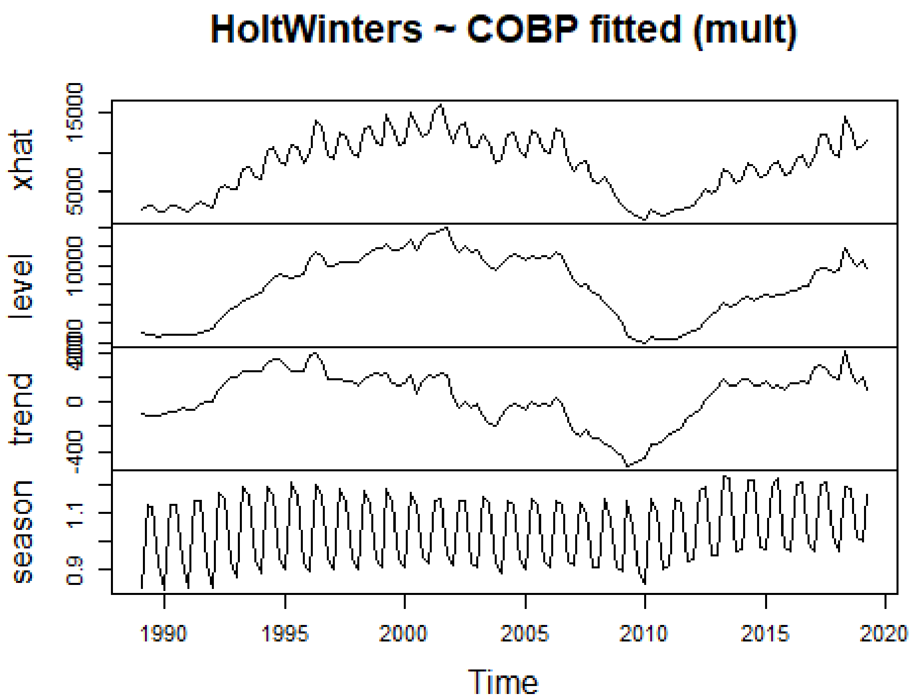
```
COBP.hw = HoltWinters(COBP_ts, seasonal="mult")
COBP.hw

## Holt-Winters exponential smoothing with trend and multiplicative seasonal
component.
##
## Call:
## HoltWinters(x = COBP_ts, seasonal = "mult")
##
## Smoothing parameters:
##  alpha: 0.5091094
##  beta : 0.09889523
##  gamma: 0.2083016
##
## Coefficients:
##              [,1]
## a   9763.6585677
## b     85.3576635
## s1     1.1630783
## s2     1.0225148
```

```
## s3    0.9768007
## s4    1.1548384

COBP.hw2 = HoltWinters(COBP_ts, seasonal="addit")
COBP.hw2

## Holt-Winters exponential smoothing with trend and additive seasonal compon
ent.
##
## Call:
## HoltWinters(x = COBP_ts, seasonal = "addit")
##
## Smoothing parameters:
##  alpha: 0.4945359
##  beta : 0.1320332
##  gamma: 0.3038037
##
## Coefficients:
##            [,1]
## a  10421.76521
## b     63.25495
## s1   389.72690
## s2  -134.34902
## s3  -445.16285
## s4   568.26182

par(mfrow=c(2,1))
plot(COBP.hw)
plot(COBP.hw2)
```

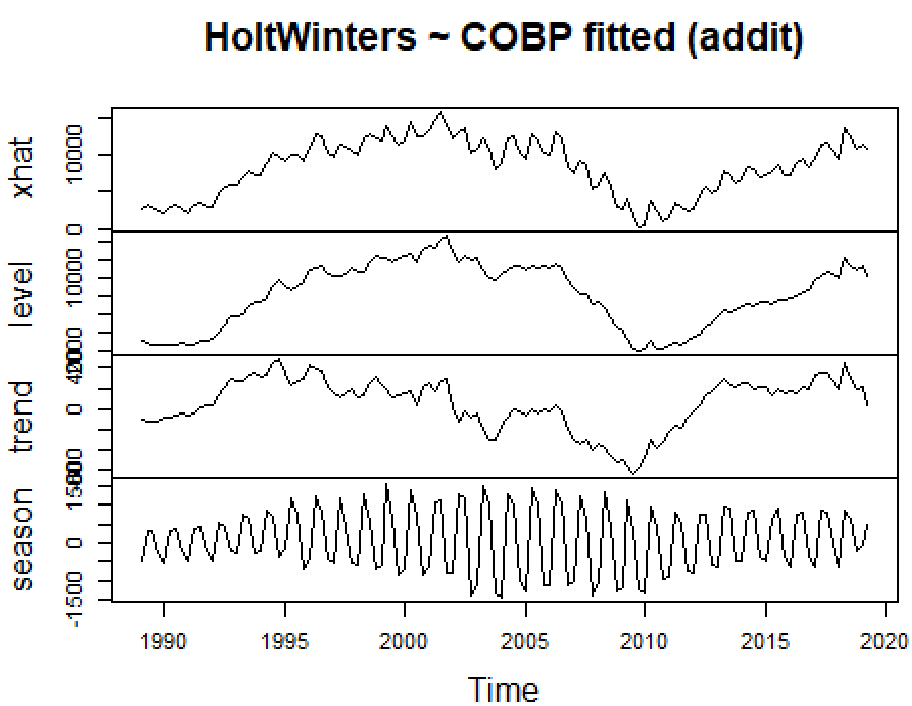





```
# From the plot and the coefficents, I find that the change of the result is
merely omitted when I change the seasonal parameter.

plot(COBP.hw$fitted,main = "HoltWinters ~ COBP fitted (mult)")
```




```r
plot(COBP.hw2$fitted,main = "HoltWinters ~ COBP fitted (addit)")
```



```
# from the alpha = 0.4945359, it means that the current prediction is capable
 to balance the recent and forward observations

# from the beta = 0.1320332 which is super close to 0, it means that the slop
e of the trending part is relatively constant in this series.
```

Now, let us forecast the data in the following 12 months.

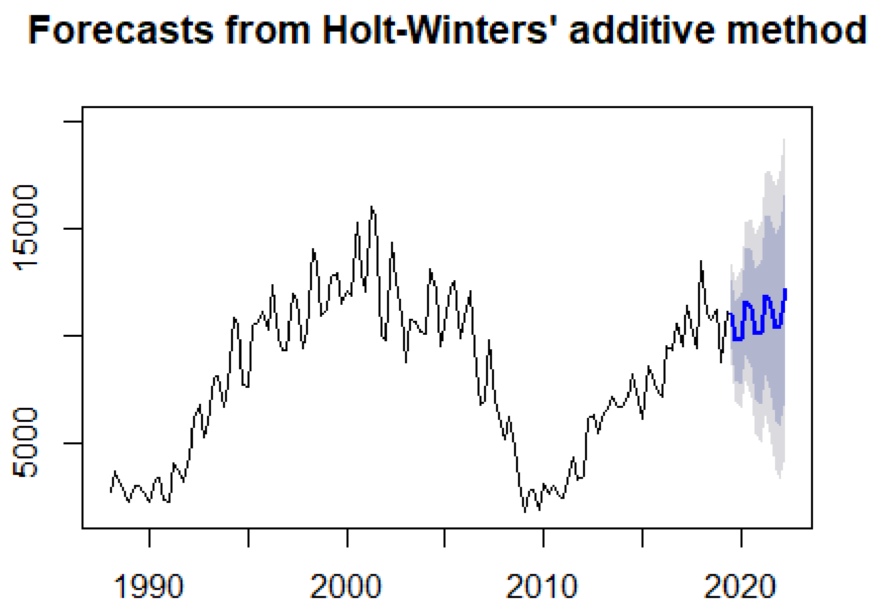
```
# Forecasting
COBP_forecast = forecast::hw(COBP_ts,h=12)
summary(COBP_forecast)

## 
## Forecast method: Holt-Winters' additive method
## 
## Model Information:
## Holt-Winters' additive method 
## 
## Call:
##  forecast::hw(y = COBP_ts, h = 12) 
## 
##   Smoothing parameters:
##     alpha = 0.5649 
##     beta  = 0.0566 
##     gamma = 1e-04 
## 
##   Initial states:
##     l = 3216.626 
##     b = 266.4756 
##     s = -752.4153 595.2534 921.9472 -764.7853
## 
##   sigma:  1219.762
## 
##      AIC     AICc      BIC 
## 2409.922 2411.474 2435.448 
## 
## Error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -26.63638 1180.404 877.1237 0.1439582 13.40506 0.6215172
##                    ACF1
## Training set 0.03467581
## 
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2019 Q3      11070.787 9507.599 12633.97 8680.097 13461.48
## 2019 Q4       9799.473 7958.927 11640.02 6984.600 12614.35
## 2020 Q1       9863.413 7739.372 11987.45 6614.972 13111.85
## 2020 Q2      11626.562 9211.803 14041.32 7933.507 15319.62
## 2020 Q3      11376.335 8662.999 14089.67 7226.646 15526.02
## 2020 Q4      10105.022 7085.111 13124.93 5486.467 14723.58
## 2021 Q1      10168.962 6834.277 13503.65 5069.002 15268.92
```

```
## 2021 Q2       11932.110 8274.426 15589.79 6338.164 17526.06
## 2021 Q3       11681.884 7692.944 15670.82 5581.327 17782.44
## 2021 Q4       10410.570 6082.319 14738.82 3791.082 17030.06
## 2022 Q1       10474.510 5798.917 15150.10 3323.807 17625.21
## 2022 Q2       12237.659 7206.796 17268.52 4543.619 19931.70
```

```
COBP_forecast$mean
```

```
##           Qtr1      Qtr2      Qtr3      Qtr4
## 2019                     11070.787  9799.473
## 2020  9863.413 11626.562 11376.335 10105.022
## 2021 10168.962 11932.110 11681.884 10410.570
## 2022 10474.510 12237.659
```

```
plot(COBP_forecast)
```



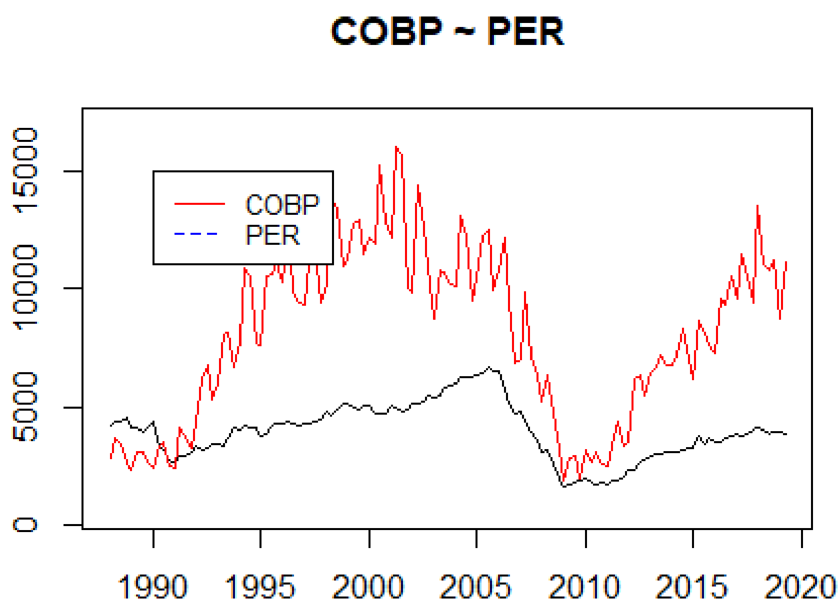

```
# from the result, we can find that the Colorado permit in the following 12 m
onths are the result from COBP_forecast$mean.
```

Question 2

How does the response variable for the state you are studying compare to the same variable at the national level?

At first, I want to plot these 2 datasets

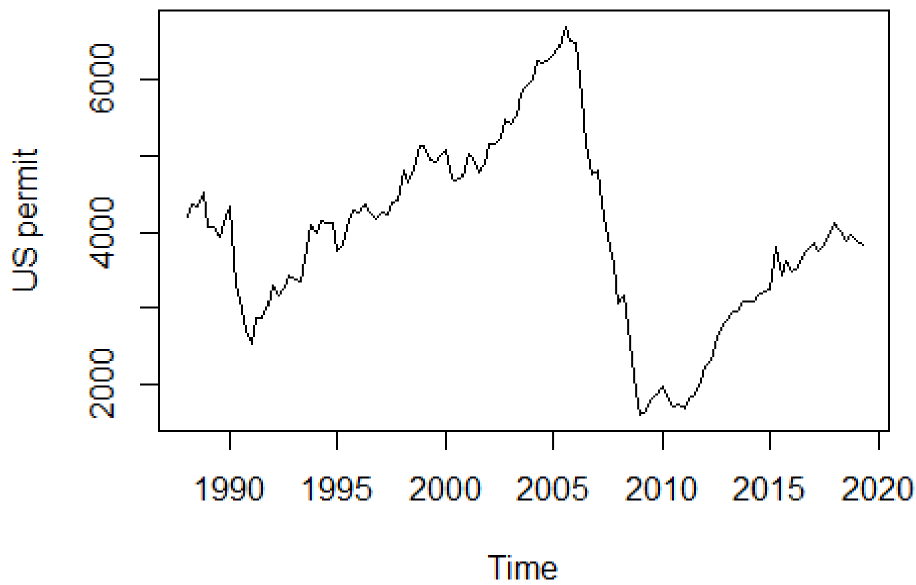
```
plot(PER_ts,ylim = c(500,17000),main = "COBP ~ PER",ylab = "",xlab = "")
lines(COBP_ts,col=2)
legend(1990, 15000, legend=c("COBP", "PER"),
       col=c("red", "blue"),lty = 1:2, cex=0.8)
```




```
# So, the data in Colorado is above national average except for 1990 around.
But, the plot shows me that the data is mostly greater than the same one on a
 national level.
```

Now, let us do the similar things to the PER_ts

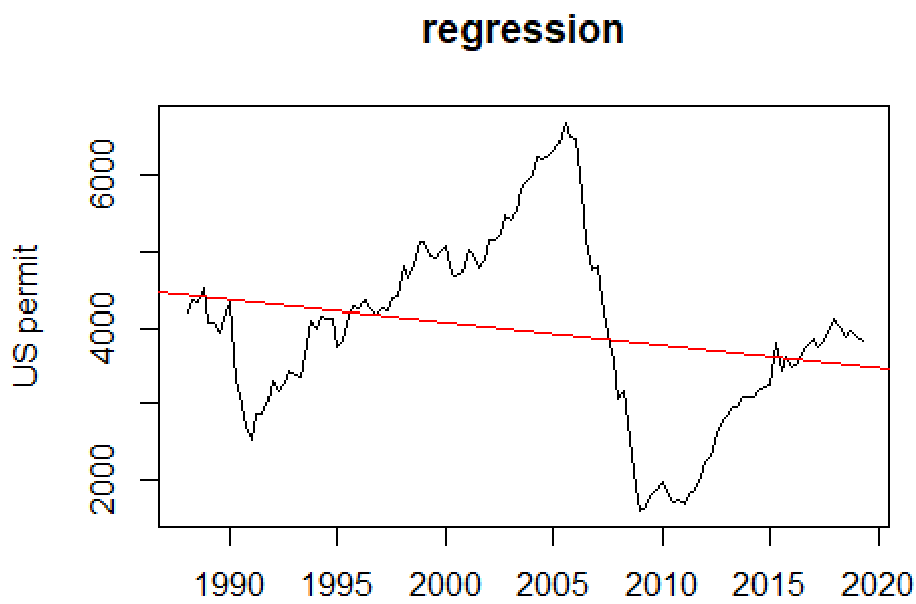
```
# plot the PER_ts
plot(PER_ts,ylab = "US permit")
```



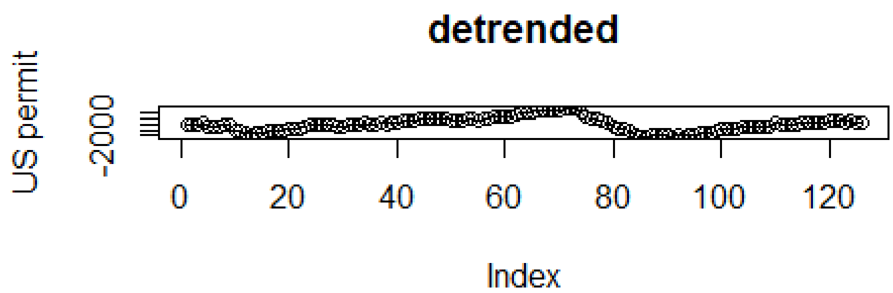

```
# It seems that the New Private Housing Units Authorized by Building Permits
are getting down in these years constantly.

# summary
describe(PER_ts)

##    vars   n    mean      sd median trimmed     mad  min  max range skew
## X1    1 126 3958.71 1210.31 3989.5 3947.63 1203.87 1616 6685  5069 0.06
##    kurtosis     se
## X1    -0.41 107.82
```

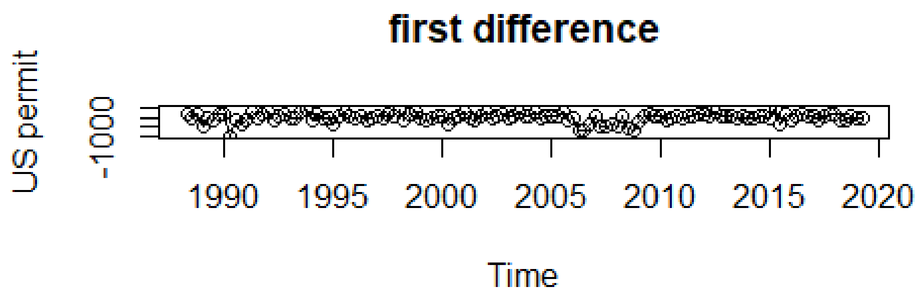
let us do the detrending, differencing and Smoothing.

```
fit2 = lm(PER_ts~time(PER_ts))

plot(PER_ts,ylab = "US permit",xlab = "",main = "regression")
abline(fit2,col=2) # add regression line to the plot
```




```r
#  plot both detrended and first differenced series

par(mfrow=c(2,1))
plot(resid(fit2), type="o", main="detrended",ylab = "US permit")
plot(diff(PER_ts), type="o", main="first difference",ylab = "US permit")
```





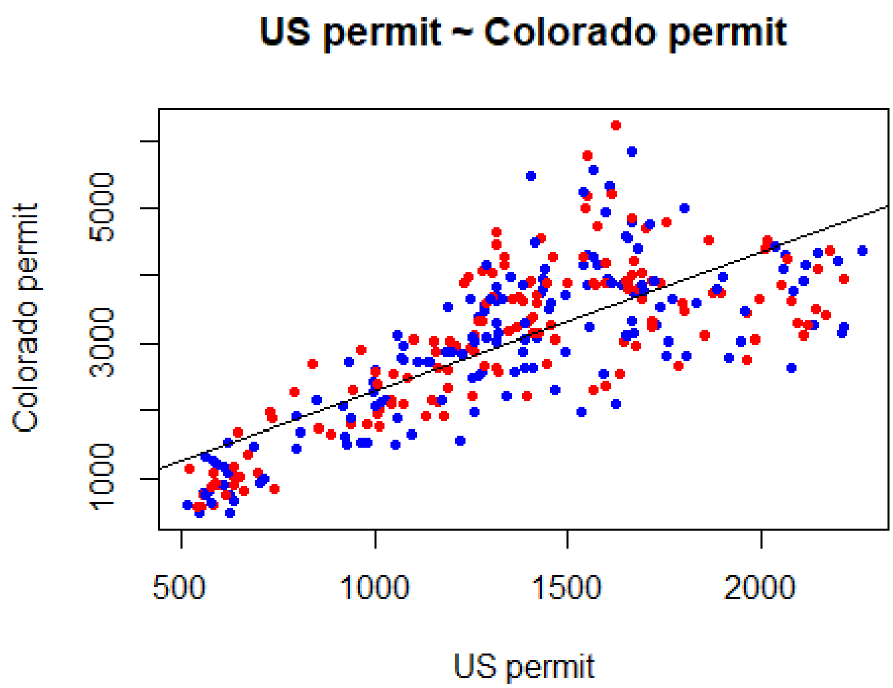

```
# I think differencing better detrend the data.
```

Regression with Autoregressive Errors

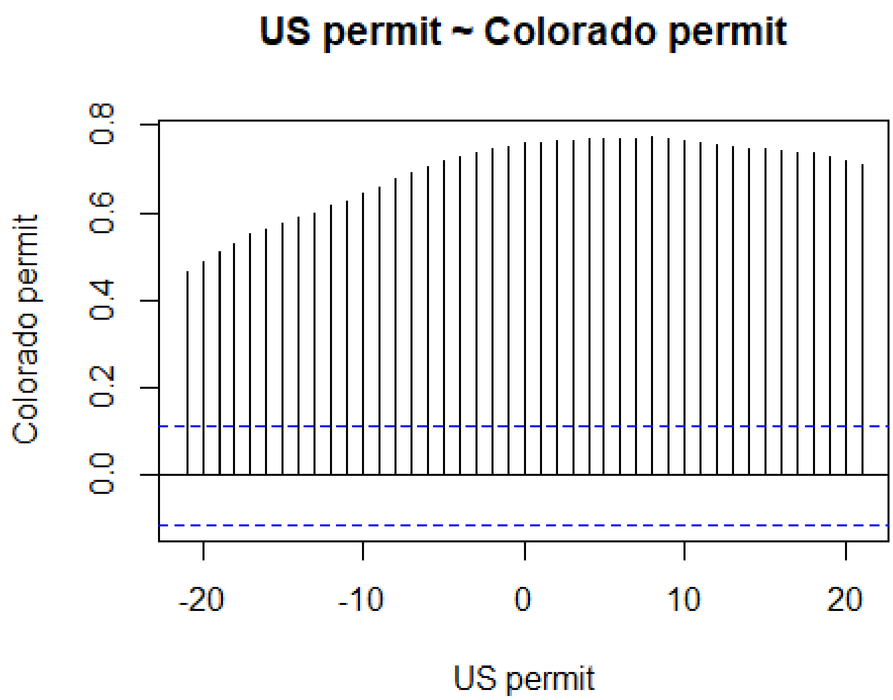
```
# Now, I consider the entire dataset of New Private Housing Units Authorized
by Building Permits for Colorado
# I extract the last 300 rows of data

x = tail(PERMIT$PERMIT,300)
y = tail(COBP$COBPPRIV,300)
fit.xy = lm(y~x)
summary(fit.xy)

##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1874.0  -535.2   -44.5   454.0  2652.3
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 216.3261   144.9357   1.493    0.137
## x             2.0667     0.1022  20.214   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 774.3 on 298 degrees of freedom
## Multiple R-squared:  0.5783, Adjusted R-squared:  0.5768
## F-statistic: 408.6 on 1 and 298 DF,  p-value: < 2.2e-16
```

```
# plot the data

plot(x,y,
     xlab = "US permit",
      ylab = "Colorado permit",col = c("red","blue"), pch = 20, main = "US pe
rmit ~ Colorado permit")

# I find that the cross relationship between them gives me a positive relatio
n.
abline(fit.xy)
```



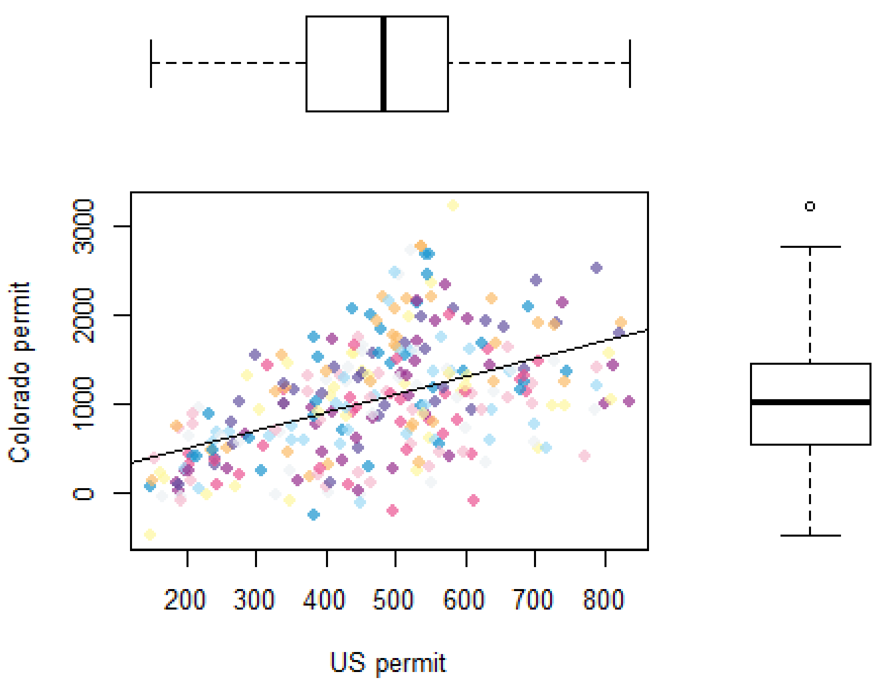

```
# cross relationship
ccf(x,y,
    xlab = "US permit",
    ylab = "Colorado permit",
    main = "US permit ~ Colorado permit")
```




```
# now, I compute the rho and transform the data
res<-fit.xy$residuals
n = 300
rnum<-0
rdenom<-0
for(i in 2:n){
rnum<-rnum+res[i]*res[i-1]
rdenom<-rdenom+res[i-1]^2
}
rhat<- rnum/rdenom # compute rho
rhat<-as.numeric(rhat)

yprime<-rep(0,n-1)
xprime<-rep(0,n-1)

#transform data
for (i in 1:n-1){
yprime[i]<-y[i+1]-rhat*y[i]
xprime[i]<-x[i+1]-rhat*x[i]}

# transformed fit.xy
fit.xy.trans<-lm(yprime~xprime)
summary(fit.xy.trans)

##
## Call:
```

```
## lm(formula = yprime ~ xprime)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1408.10  -387.27   -63.76   361.01  1952.34
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 110.7493   106.0752   1.044    0.297
## xprime        1.9925     0.2125   9.378   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 589.7 on 297 degrees of freedom
## Multiple R-squared:  0.2285, Adjusted R-squared:  0.2259
## F-statistic: 87.95 on 1 and 297 DF,  p-value: < 2.2e-16

#check if autocorrelation is still a problem
dwtest(fit.xy.trans)

##
##  Durbin-Watson test
##
## data:  fit.xy.trans
## DW = 2.3209, p-value = 0.9969
## alternative hypothesis: true autocorrelation is greater than 0

#transform back
b0<-fit.xy.trans$coefficients[1]/(1-rhat)
b1<-fit.xy.trans$coefficients[2]
b0;b1

## (Intercept)
##    315.8447

##   xprime
## 1.992542

# plot the transformed data

# Set plot layout
layout(mat = matrix(c(2, 1, 0, 3),
                        nrow = 2,
                        ncol = 2),
       heights = c(1, 2),    # Heights of the two rows
       widths = c(2, 1))     # Widths of the two columns

# Plot 1: Scatterplot
par(mar = c(5, 4, 0, 0))
plot(x = xprime,
     y = yprime,
```

```r
     xlab = "US permit",
     ylab = "Colorado permit",
     pch = 16,
     col = yarrr::piratepal("pony",trans = 0.3))
abline(fit.xy.trans)

# Plot 2: Top (height) boxplot
par(mar = c(0, 4, 0, 0))
boxplot(xprime, xaxt = "n",
        yaxt = "n", bty = "n", yaxt = "n",
        col = "white", frame = FALSE, horizontal = TRUE)

## Warning in bxp(list(stats = structure(c(147.503516736951,
## 371.384377266796, : Duplicated argument yaxt = "n" is disregarded

# Plot 3: Right (weight) boxplot
par(mar = c(5, 0, 0, 0))
boxplot(yprime, xaxt = "n",
        yaxt = "n", bty = "n", yaxt = "n",
        col = "white", frame = F)
```




```r
# we see that the transformed data show us autocorrelation is not a problem a
nd we can clearly detect the trend and relative description of the transforme
d data.
```

```r
# forecasting
x[n+1]=1500
e300=y[n]-(b0+b1*x[n])
y[n+1]= b0+b1*x[n+1]
# difference in results compared to notes is due to round-off
f301=y[n+1]+rhat*e300
xf=x[n+1]-rhat*x[n]
# get MSE value from ANOVA table
anova(fit.xy.trans)
```

```
## Analysis of Variance Table
##
## Response: yprime
##             Df    Sum Sq  Mean Sq F value    Pr(>F)
## xprime       1  30586180 30586180  87.953 < 2.2e-16 ***
## Residuals 297 103283320   347755
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# alpha = 0.05
mse=347755
spred=sqrt(mse*(1+(1/(n-1))+((xf-mean(xprime))^2)/(sum((xprime-mean(xprime))^
2)) ))
tvalue=qt(1-.025,n-3)
f_lower=f301-tvalue*spred
as.numeric(f_lower)
```

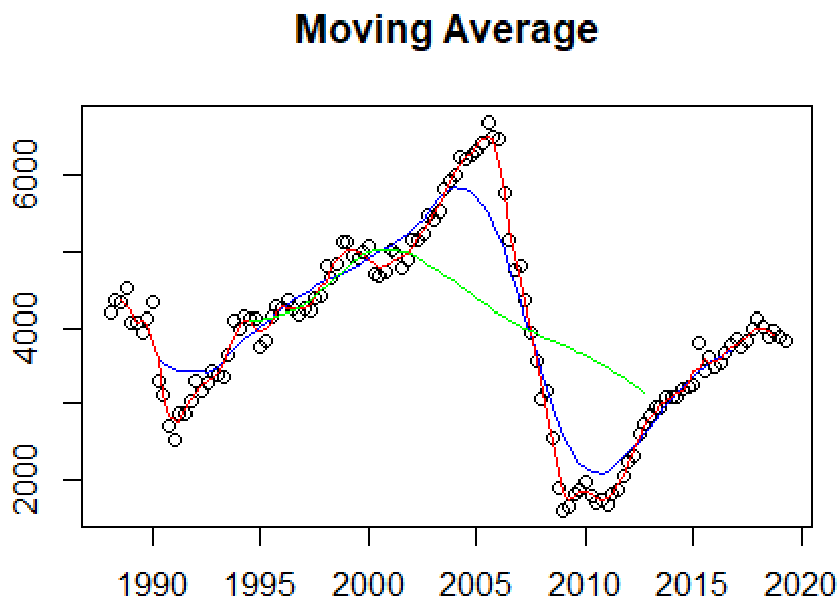
```
## [1] 2278.386
```

```r
f_upper=f301+tvalue*spred
as.numeric(f_upper)
```

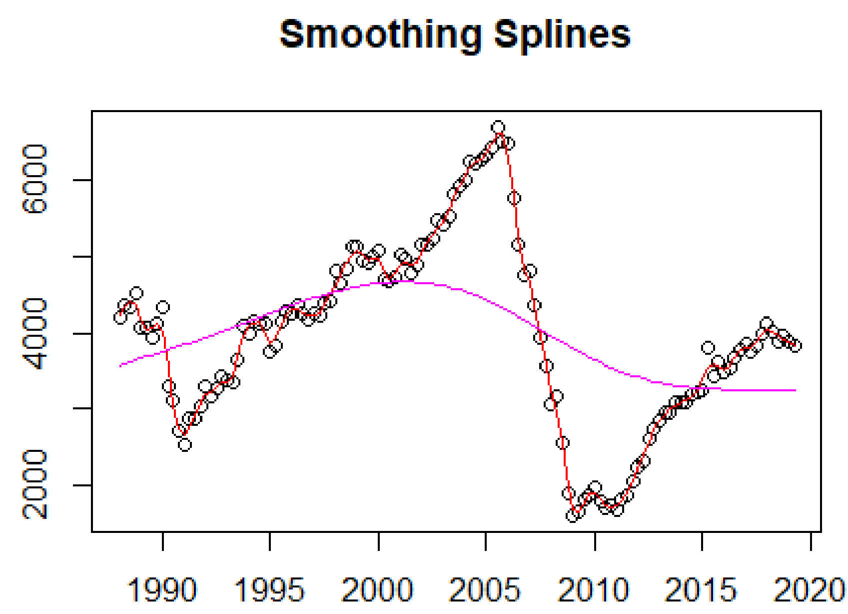
```
## [1] 4605.645
```

Now, let us use 2 kinds of smoothers to smooth the US permit

```r
# Moving Average Smoother

# 20-point moving average
ma20 = filter(PER_ts, sides=2, rep(1,20)/20)
# unequal weights
ma5u = filter(PER_ts, sides=2, c(.5/5, 1.5/5, 1/5, 1.5/5, .5/5))
#53-point moving average
ma53 = filter(PER_ts, sides=2, rep(1,53)/53)
plot(PER_ts, type="p", ylab="",xlab = "",main = "Moving Average")
lines(ma20, col="blue")
lines(ma5u,col="red")
lines(ma53,col="green")
```




```
# Smoothing Splines
plot(PER_ts, type="p", ylab="",xlab = "",main = "Smoothing Splines")
lines(smooth.spline(time(PER_ts), PER_ts),col = 2)
lines(smooth.spline(time(PER_ts), PER_ts, spar=1),col = 6)
```

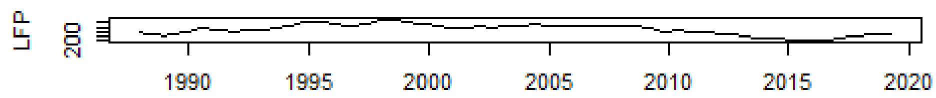


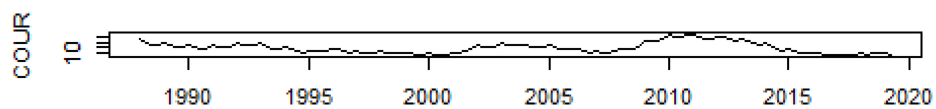
Question 3

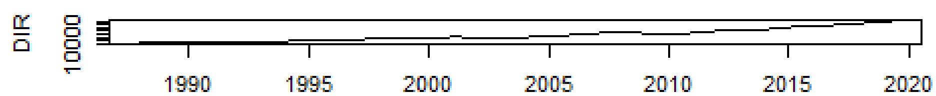
Describe the attributes of the data for each of the explanatory variables you considered.

Labor force participation

```
# LFP

# plot the data for each of the explanatory variables

par(mfrow = c(3,1))
plot(LFP_ts,ylab = "LFP",xlab = "")
plot(COUR_ts,ylab = "COUR",xlab = "")
plot(DIR_ts,ylab = "DIR",xlab = "")
```







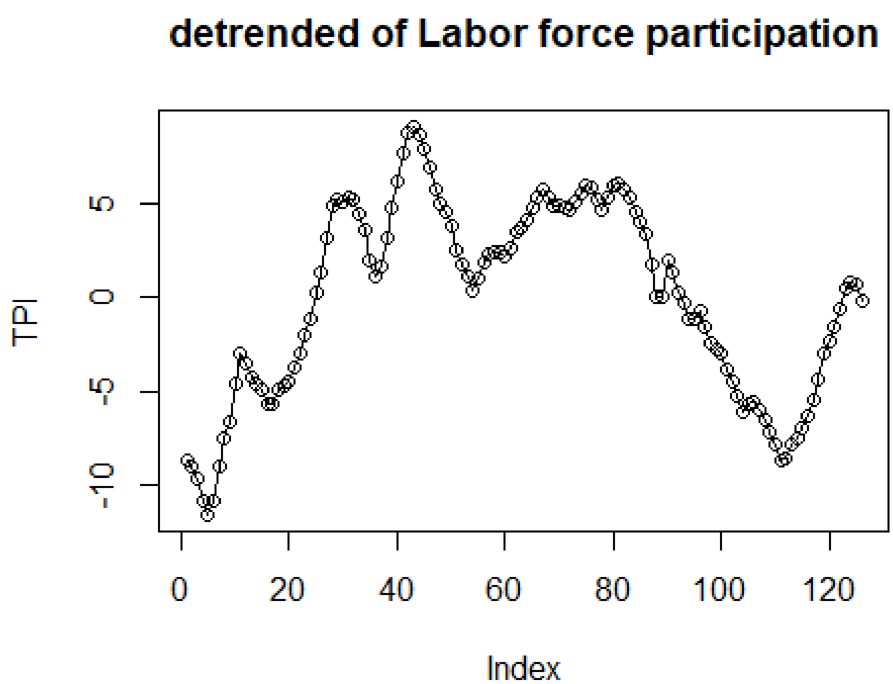

```r
# From these 3 plots, I find that LFP and COUR are fluctuated, and the Divide
nds, Interest and Rent in Colorado seems to be increasing over time.

# describe the explainary variables
for(data in explainary){
  print(describe(data))
}

##    vars   n  mean   sd median trimmed  mad   min   max range  skew
## X1    1 126 212.5 6.03  213.9  212.76 5.71 199.8 223.4  23.6 -0.38
##    kurtosis   se
## X1    -0.84 0.54
##    vars   n  mean   sd median trimmed  mad min  max range skew kurtosis
## X1    1 126 14.94 4.99  14.35   14.52 5.26 7.5 27.2  19.7 0.64     -0.4
##      se
## X1 0.44
##    vars   n     mean       sd   median  trimmed      mad     min     max
## X1    1 126 36521.65 18180.85 30921.45 34913.94 20253.21 12489.3 78401.6
##      range skew kurtosis      se
## X1 65912.3 0.63    -0.61 1619.68
```

Now, let us do the detrending, differencing and Smoothing

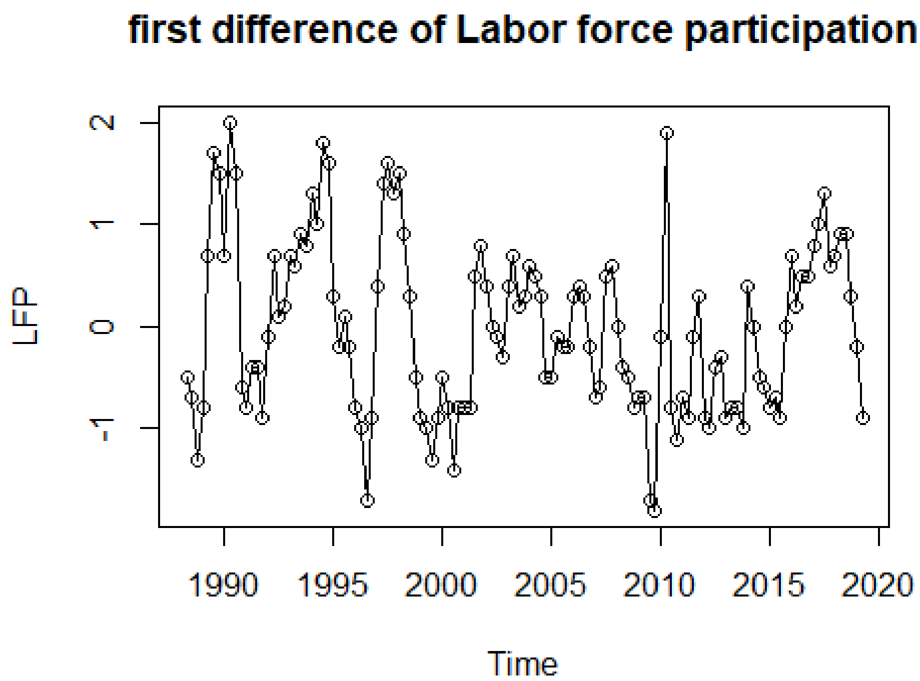
```r
fit_LFP = lm(LFP_ts~time(LFP_ts));summary(fit_LFP)

##
## Call:
## lm(formula = LFP_ts ~ time(LFP_ts))
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.6477  -4.5297   0.8967   4.8000   9.1652
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  890.11659  102.02452   8.725 1.48e-14 ***
## time(LFP_ts)  -0.33819    0.05092  -6.642 8.72e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.197 on 124 degrees of freedom
## Multiple R-squared:  0.2624, Adjusted R-squared:  0.2565
## F-statistic: 44.11 on 1 and 124 DF,  p-value: 8.72e-10
```

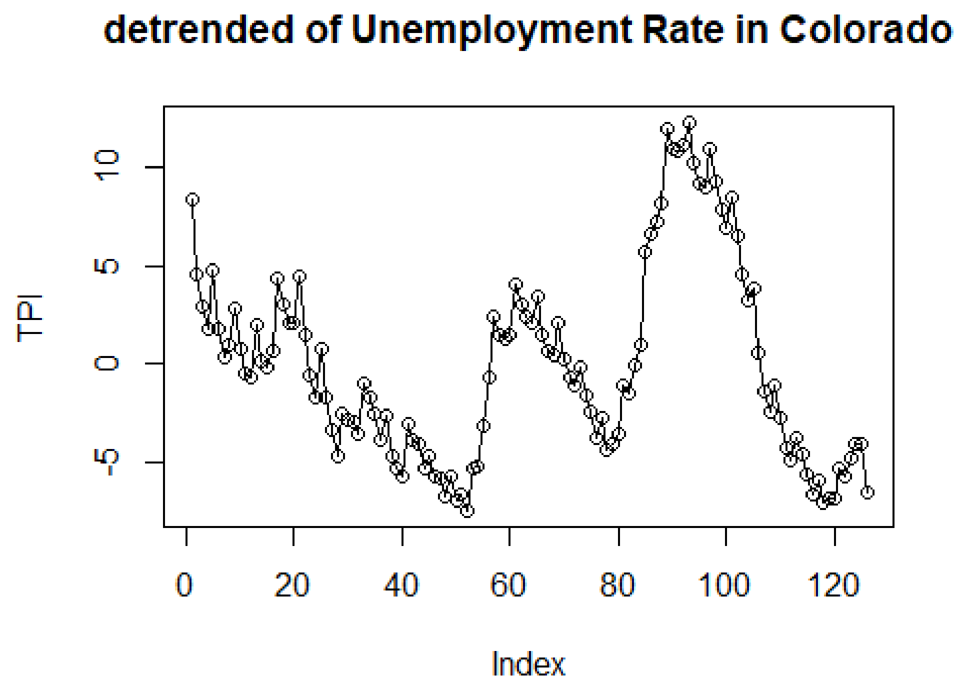
```r
plot(resid(fit_LFP), type="o", ylab = "TPI", main="detrended of Labor force p
articipation")
```



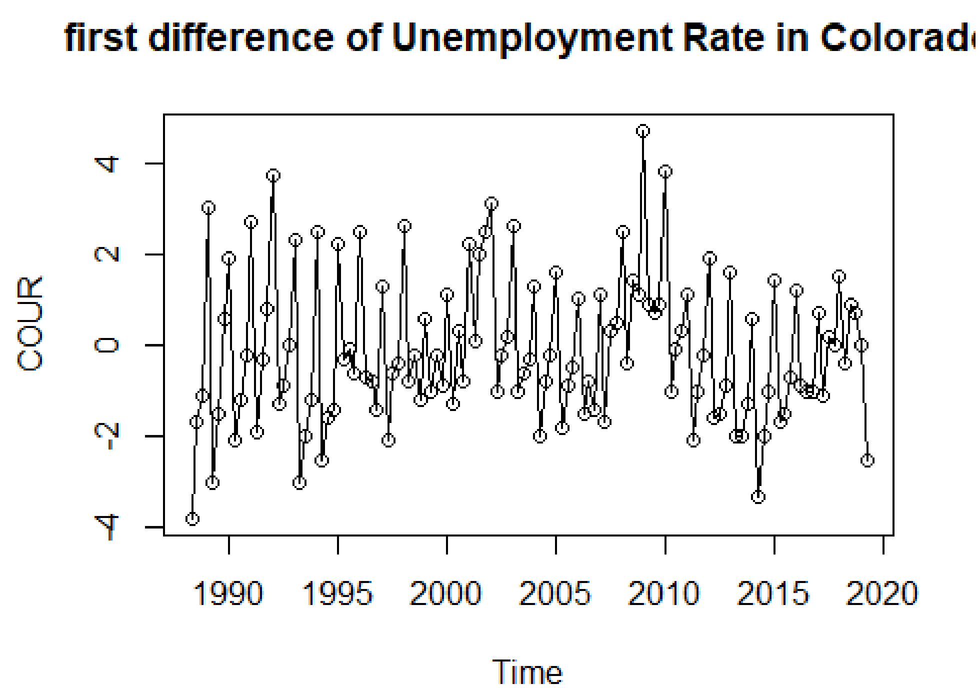

```r
plot(diff(LFP_ts), type="o", ylab = "LFP", main="first difference of Labor fo
rce participation")
```



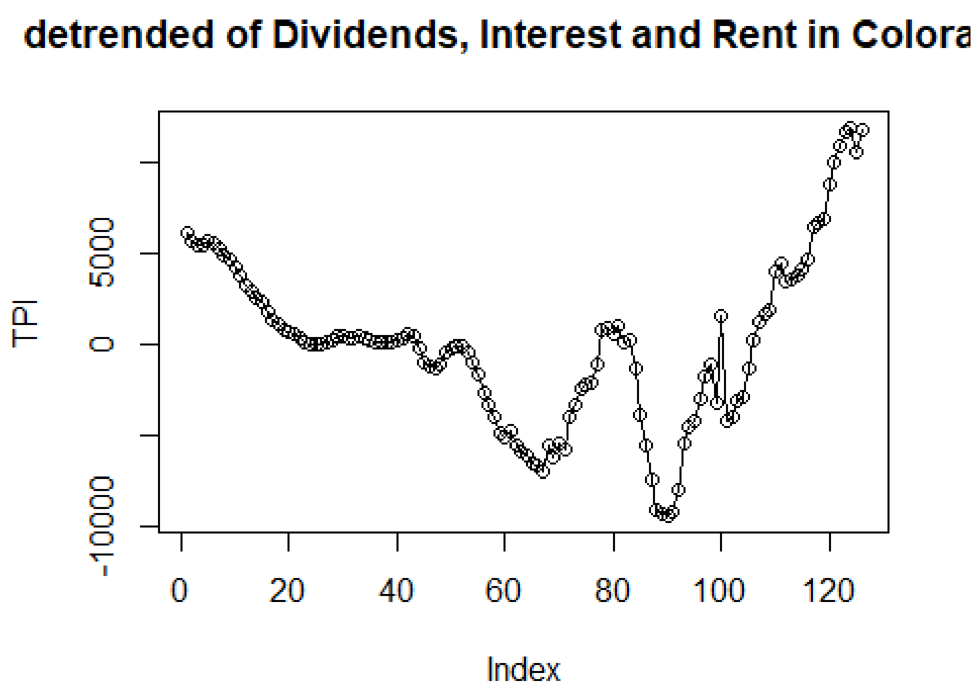

```
fit_COUR = lm(COUR_ts~time(COUR_ts));summary(fit_COUR)

##
## Call:
## lm(formula = COUR_ts ~ time(COUR_ts))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4648 -4.0967 -0.6384  2.7439 12.3275
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   32.966856  98.411593   0.335    0.738
## time(COUR_ts) -0.008998   0.049116  -0.183    0.855
##
## Residual standard error: 5.013 on 124 degrees of freedom
## Multiple R-squared:  0.0002706,	Adjusted R-squared:  -0.007792
## F-statistic: 0.03356 on 1 and 124 DF,  p-value: 0.8549

plot(resid(fit_COUR), type="o", ylab = "TPI", main="detrended of Unemployment
 Rate in Colorado")
```



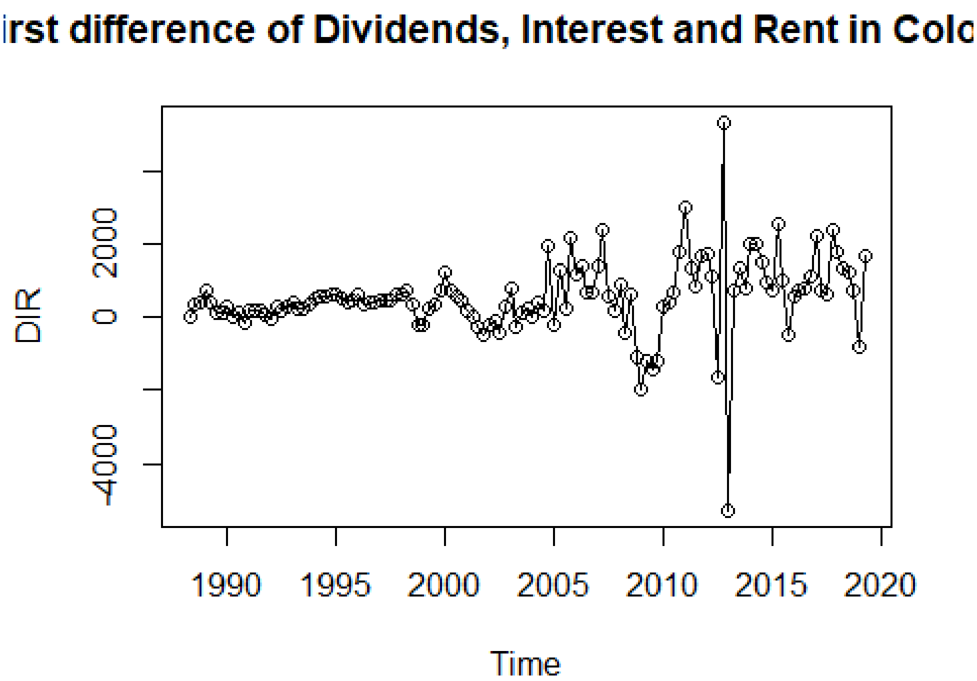

```r
plot(diff(COUR_ts), type="o", ylab = "COUR", main="first difference of Unemployment Rate in Colorado")
```



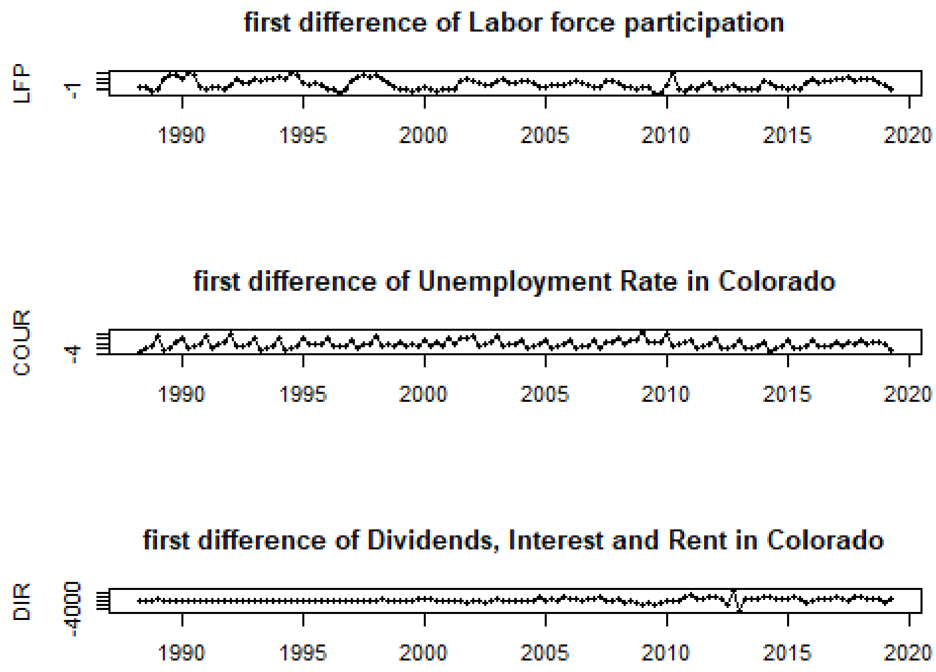
```
fit_DIR = lm(DIR_ts~time(DIR_ts));summary(fit_DIR)

##
## Call:
## lm(formula = DIR_ts ~ time(DIR_ts))
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -9425  -3112    126   2171  11883
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -3.826e+06  8.999e+04  -42.52   <2e-16 ***
## time(DIR_ts)  1.928e+03  4.491e+01   42.92   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4584 on 124 degrees of freedom
## Multiple R-squared:  0.9369, Adjusted R-squared:  0.9364
## F-statistic:  1842 on 1 and 124 DF,  p-value: < 2.2e-16

plot(resid(fit_DIR), type="o", ylab = "TPI", main="detrended of Dividends, In
terest and Rent in Colorado")
```




```
plot(diff(DIR_ts), type="o", ylab = "DIR", main="first difference of Dividend
s, Interest and Rent in Colorado")
```

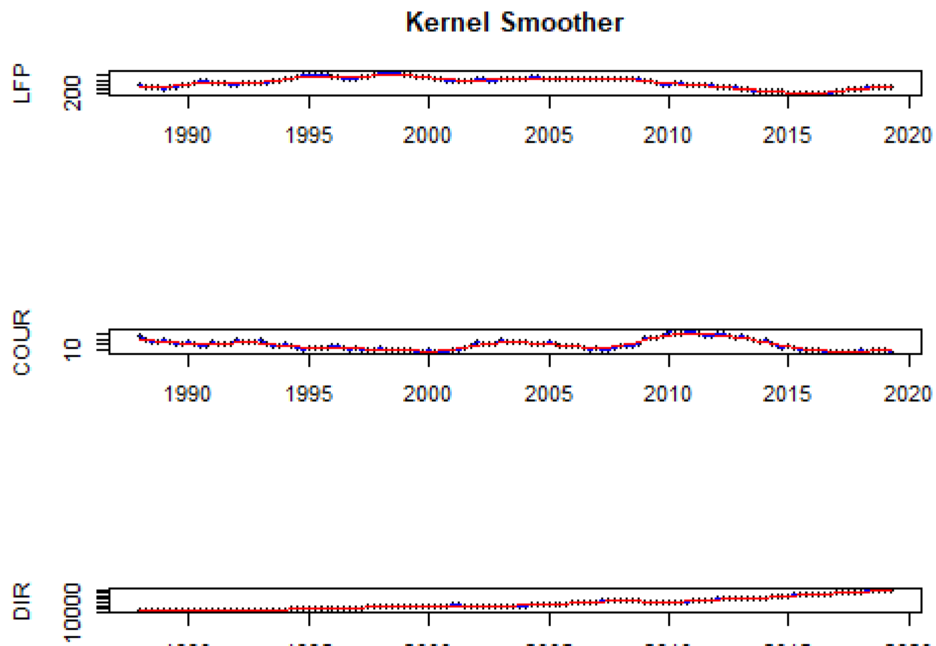



```
par(mfrow=c(3,1))
plot(diff(LFP_ts), type="o", xlab = "",ylab = "LFP", main="first difference o
f Labor force participation")
plot(diff(COUR_ts), type="o", xlab = "",ylab = "COUR", main="first difference
 of Unemployment Rate in Colorado")
plot(diff(DIR_ts), type="o", xlab = "",ylab = "DIR", main="first difference o
f Dividends, Interest and Rent in Colorado")
```



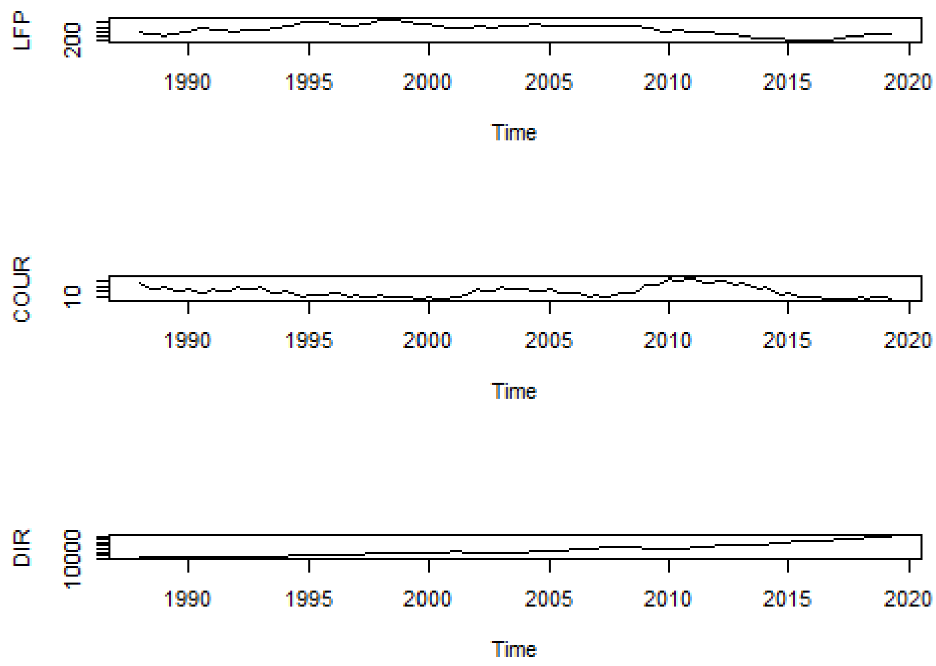
Now, let us smooth these 3 explainary variables

```
# Kernel Smoother

par(mfrow=c(3,1))
plot(LFP_ts, type="p", xlab = "", ylab="LFP",main = "Kernel Smoother")
lines(ksmooth(time(LFP_ts), LFP_ts, "normal", bandwidth=5/52),col = 4)
lines(ksmooth(time(LFP_ts), LFP_ts, "normal", bandwidth=2),col = 2)


plot(COUR_ts, type="p", xlab = "", ylab="COUR")
lines(ksmooth(time(COUR_ts), COUR_ts, "normal", bandwidth=5/52),col = 4)
lines(ksmooth(time(COUR_ts), COUR_ts, "normal", bandwidth=2),col = 2)


plot(DIR_ts, type="p", xlab = "", ylab="DIR")
lines(ksmooth(time(DIR_ts), DIR_ts, "normal", bandwidth=5/52),col = 4)
lines(ksmooth(time(DIR_ts), DIR_ts, "normal", bandwidth=2),col = 2)
```

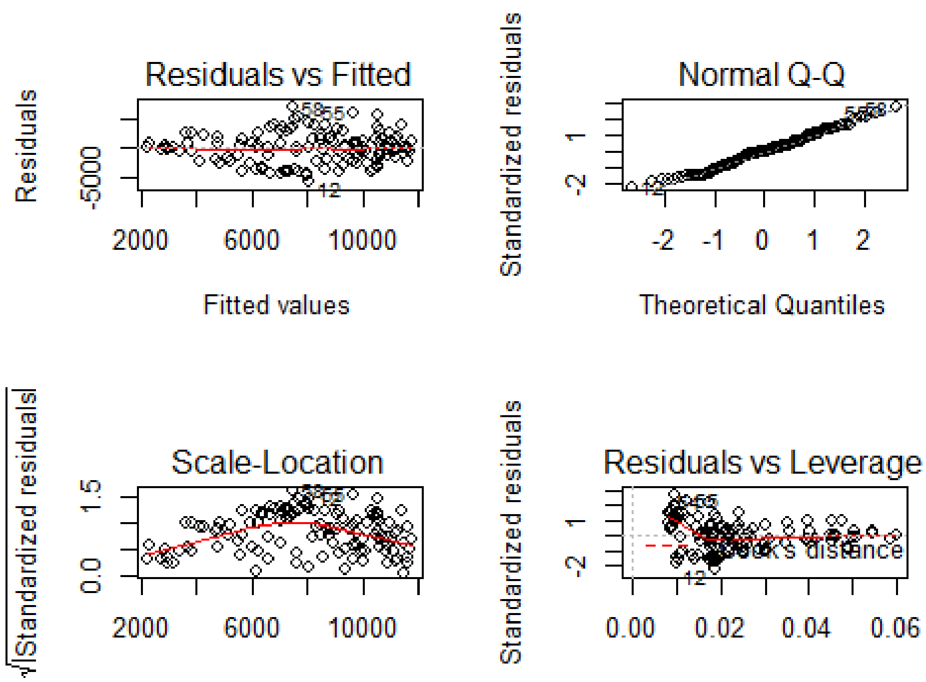


Question 4

```
mydata = data.frame(y = COBP_ts,
                    x1 = LFP_ts,
                    x2 = COUR_ts,
                    x3 = DIR_ts)

par(mfrow = c(3,1))
plot(LFP_ts,ylab = "LFP")
plot(COUR_ts,ylab = "COUR")
plot(DIR_ts,ylab = "DIR")
```




```r
multi_fit = lm(y ~ x1 + x2 + x3 -1, data = mydata)
summary(multi_fit)

##
## Call:
## lm(formula = y ~ x1 + x2 + x3 - 1, data = mydata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5606.2 -1963.7    36.2  1529.6  6883.0
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## x1   68.49141    4.15938  16.467   <2e-16 ***
## x2 -475.57152   46.19196 -10.296   <2e-16 ***
## x3    0.01818    0.01250   1.454    0.149
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2622 on 123 degrees of freedom
## Multiple R-squared:  0.9149, Adjusted R-squared:  0.9128
## F-statistic: 440.9 on 3 and 123 DF,  p-value: < 2.2e-16

par(mfrow = c(2,2))
plot(multi_fit)
```




```r
# from the Multiple R-squared, we know that 91.49% of variation in permit can
 be explained by Labor force participation, Unemployment Rate in Colorado, Di
vidends, Interest and Rent in Colorado.

# from the slope of each varible, I see that permit is positively related to
LFP, the slope is 68.49.

# permit is negatively related to COUR sharply which really makes sense, sinc
e the unemployment rate affects permit very much, the slope is -475.57

# permit is nearly not related to DIR for the slope is close to 0.

# let us check the correlation between there 3 varibles
cor(LFP_ts,COUR_ts,method = "pearson")

## [1] -0.1912881

cor(LFP_ts,DIR_ts,method = "pearson")

## [1] -0.5931632

cor(DIR_ts,COUR_ts,method = "pearson")

## [1] -0.1476877

confint(multi_fit, conf.level = 0.95)
```

```
##            2.5 %        97.5 %
## x1  6.025817e+01   76.72465118
## x2 -5.670057e+02 -384.13736154
## x3 -6.573854e-03    0.04292501
```

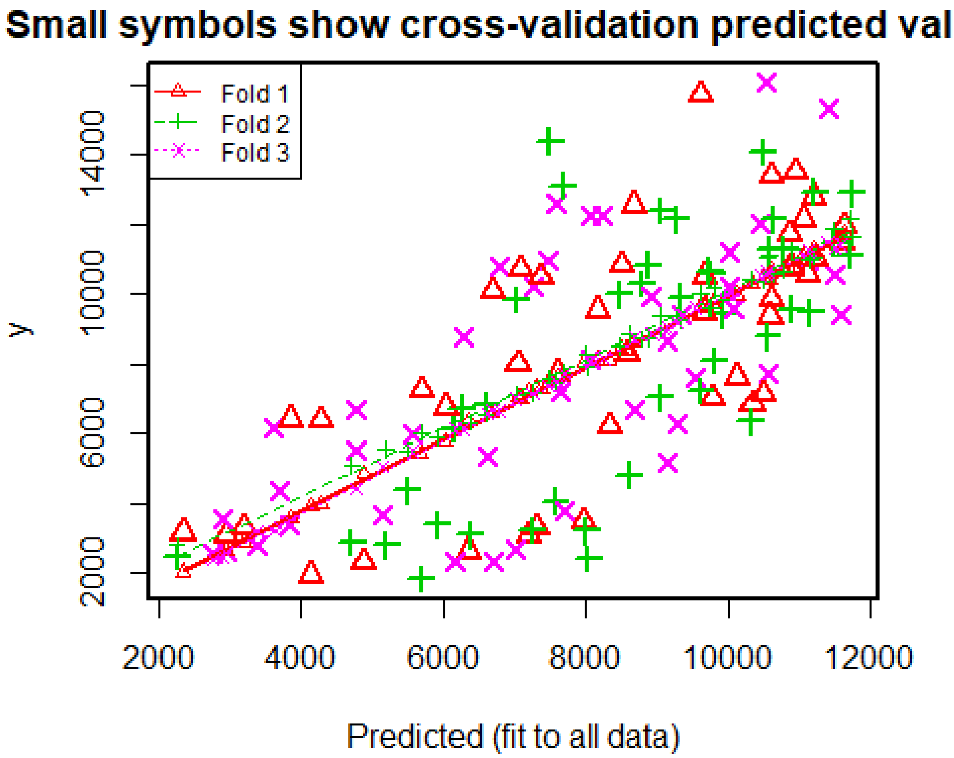
```r
anova(multi_fit)
```

```
## Analysis of Variance Table
##
## Response: y
##            Df     Sum Sq    Mean Sq   F value Pr(>F)
## x1          1 8327115504 8327115504 1211.5455 <2e-16 ***
## x2          1  749074313  749074313  108.9858 <2e-16 ***
## x3          1   14523958   14523958    2.1131 0.1486
## Residuals 123  845395604    6873135
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Cross Validation
DAAG::cv.lm(data = mydata, multi_fit, m=3)
```

```
## Analysis of Variance Table
##
## Response: y
##            Df   Sum Sq  Mean Sq F value Pr(>F)
## x1          1 8.33e+09 8.33e+09 1211.55 <2e-16 ***
## x2          1 7.49e+08 7.49e+08  108.99 <2e-16 ***
## x3          1 1.45e+07 1.45e+07    2.11   0.15
## Residuals 123 8.45e+08 6.87e+06
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Warning in DAAG::cv.lm(data = mydata, multi_fit, m = 3):
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
##
## fold 1
## Observations in test set: 42
##                  4     5     7    10    11   22   25    30    39    43
## Predicted     6354  4886  7200  7304  7980 7048 7606  9678 10865 10609
## cvpred        6369  4846  7236  7339  8036 7061 7631  9755 10960 10679
## y             2654  2311  3082  3317  3479 8012 7743 10522 11719 13384
## CV residual -3715 -2535 -4154 -4022 -4557  951  112   767   759  2705
##                 46    48    49    50    55    63    65   68    69    71
## Predicted    11202 11593 11058 11634  9617  7093  6684 8176  7377  8667
## cvpred       11296 11696 11135 11727  9637  7025  6599 8128  7303  8630
## y            12779 11495 12150 11935 15699 10716 10092 9547 10488 12556
## CV residual  1483  -201  1015   208  6062  3691  3493 1419  3185  3926
##                 73   75    76    77    78    79    88   89   91   94   99
## Predicted     8497 9676 10340  9795 10606 10508  4144 2335 2953 3211 4277
## cvpred        8434 9641 10325  9752 10578 10472  3925 2051 2684 2917 3998
## y            10834 9474  6848  7015  9848  7131  1935 3152 3082 3319 6368
## CV residual  2400 -167 -3477 -2737  -730 -3341 -1990 1101  398  402 2370
##               101  103  105  107   109   112   115   116   121     122
## Predicted    3862 5692 6034 8583  8328 10128 10594 11124 10948 11224.0
## cvpred       3570 5452 5790 8408  8134  9980 10449 10990 10759 11036.5
## y            6420 7237 6755 8287  6193  7616  9372 10584 13525 11021.0
## CV residual 2850 1785  965 -121 -1941 -2364 -1077  -406  2766   -15.5
##                  123
## Predicted    10879.9
## cvpred       10672.5
## y            10769.0
```

```
## CV residual    96.5
##
## Sum of squares = 2.56e+08    Mean square = 6096332    n = 42
##
## fold 2
## Observations in test set: 42
##                  3     6    12    14    16    17      18   19    26    27
## Predicted    5912  6368  8031  7575  7252  5485 6156.58 6595  8874  9768
## cvpred       5878  6328  7913  7485  7181  5506 6146.54 6564  8758  9613
## y            3381  3092  2425  4065  3240  4366 6143.00 6807 10854 10570
## CV residual -2497 -3236 -5488 -3420 -3941 -1140   -3.54  243  2096   957
##                 31    33    34   35   36    40    42    45    47    52
## Predicted    9739  8787  9058 9329 9941 11155 10487 10791 11214 11740
## cvpred       9612  8723  8983 9244 9828 11004 10389 10676 11082 11623
## y           10665 10307 12398 9925 9417  9470 14071 11275 12928 12908
## CV residual  1053  1584  3415  681 -411 -1534  3682   599  1846  1285
##                 53    56   57    58    66    74    80    83    84    85
## Predicted   10642  8474 7023  7497  7676  9263 10314  8608  8010  5691
## cvpred      10584  8522 7145  7592  7777  9379 10447  8843  8262  6041
## y           12143 10004 9838 14380 13113 12141  6352  4821  3214  1846
## CV residual  1559  1482 2693  6788  5336  2762 -4095 -4022 -5048 -4195
##                 86    87   93  104   108   111   113   117   118   124
## Predicted    5192  4717 2251 6256  9035  9804  9615 10886 11491 10581
## cvpred       5554  5085 2807 6694  9404 10183 10006 11269 11852 11091
## y            2813  2889 2476 6736  7052  8110  7240  9565 11491 11268
## CV residual -2741 -2196 -331   42 -2352 -2073 -2766 -1704  -361   177
##                125   126
## Predicted   10552 11710
## cvpred      11054 12170
## y            8780 11148
## CV residual -2274 -1022
##
## Sum of squares = 3.27e+08    Mean square = 7792038    n = 42
##
## fold 3
## Observations in test set: 42
##                  1     2     8     9    13    15    20   21   23    24    28
## Predicted    3373  5146  7020  6169  6696  7694  6614 5575 8065  8697 10555
## cvpred       3217  5071  7015  6123  6670  7712  6578 5489 8087  8744 10666
## y            2822  3698  2672  2347  2349  3764  5328 5961 8114  6699  7724
## CV residual -395 -1373 -4343 -3776 -4321 -3948 -1250  472   27 -2045 -2942
##                 29    32     37    38    41    44    51    54    59    60
## Predicted    9540 10020 9358.5 10461 10032 11142 11406 10539  7578  7467
## cvpred       9601 10094 9392.1 10540 10076 11230 11490 10583  7495  7378
## y            7635 11185 9382.0 11991 10200 10981 15304 16068 12599 10968
## CV residual -1966  1091  -10.1  1451   124  -249  3814  5485  5104  3590
##                 61    62    64    67    70   72    81    82     90   92   95
## Predicted    6272  6791  7260  8080  8248 8930  9142  9297 2948.0 2767 3696
## cvpred       6124  6667  7154  8006  8164 8862  9021  9186 2591.4 2388 3327
## y            8764 10806 10214 12222 12239 9925  5183  6299 2650.0 2565 4388
```

```
## CV residual 2640  4139  3060  4216  4075 1063 -3838 -2887   58.6  177 1061
##                 96   97   98  100  102  106  110   114   119   120
## Predicted     3842 2908 3621 4781 4772 7640 9135 10070 11497 11581
## cvpred        3469 2483 3221 4411 4433 7393 8921  9888 11339 11411
## y             3367 3527 6165 5498 6687 7187 8664  9587 10581  9431
## CV residual -102 1044 2944 1087 2254 -206 -257  -301  -758 -1980
##
## Sum of squares = 2.89e+08    Mean square = 6875447    n = 42
##
## Overall (Sum over all 42 folds)
##      ms
## 6921272
```

```r
# Stepwise Regression
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:DAAG':
##
##     hills
```

```r
fit2 <- lm(y~x1+x2+x3-1,data=mydata)
step <- stepAIC(fit2, direction="both")
```

```
## Start:  AIC=1987
## y ~ x1 + x2 + x3 - 1
##
##        Df Sum of Sq      RSS  AIC
## <none>              8.45e+08 1987
## - x3    1  1.45e+07 8.60e+08 1987
## - x2    1  7.29e+08 1.57e+09 2063
## - x1    1  1.86e+09 2.71e+09 2131
```

```r
step$anova # display results
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## y ~ x1 + x2 + x3 - 1
##
## Final Model:
## y ~ x1 + x2 + x3 - 1
##
##
##   Step Df Deviance Resid. Df Resid. Dev  AIC
## 1                        123   8.45e+08 1987
```