Assignment3last.R

sergazy

Thu Dec 13 02:00:29 2018

```
#setwd("/Users/sergazy/Downloads/Fall 2018 courses and all files/Eric's class/12:05:2018 Time series")
load("Time_series_data.Rdata")
library('randomForest')
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(car)# to use qqPlot package
```

```
## Loading required package: carData
```

```
library(expm)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'expm'
```

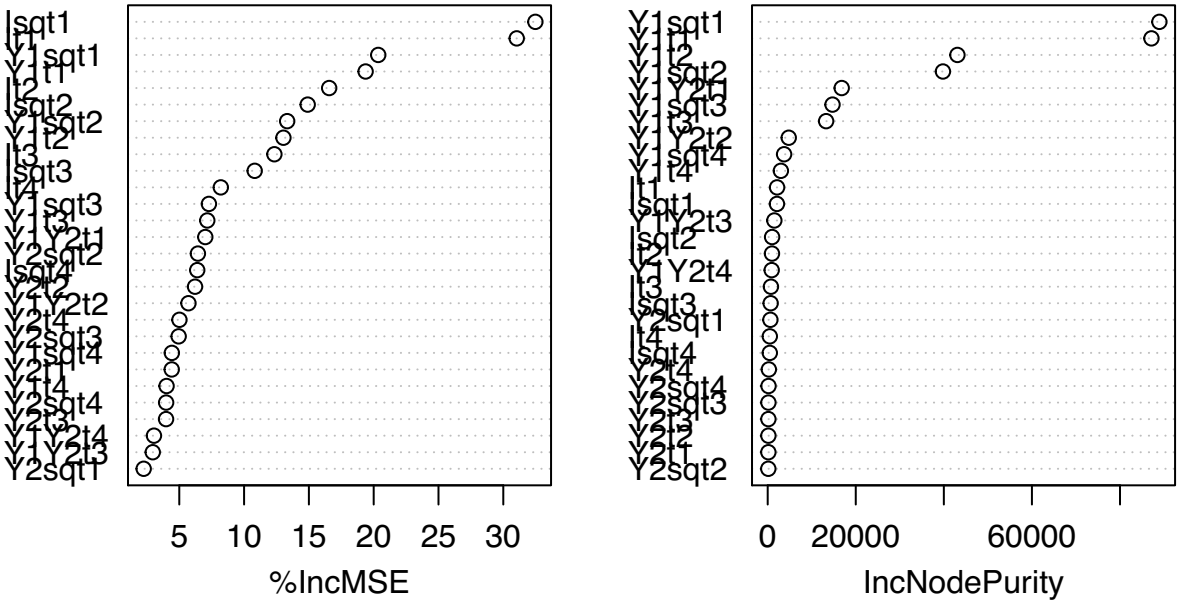
```
## The following object is masked from 'package:Matrix':
##
##     expm
```

```
# Set random seed to make results reproducible:
set.seed(17)

# Assign the data to squares
Y1square=log(Y1^2) #we are taking logarith here to make MSE smaller
Y2square=log(Y2^2) #we are taking logarith here to make MSE smaller
Isquare=I^2
# and interactions
Y1interactY2=log(Y1*Y2) #we are taking logarith here to make MSE smaller
#Y1interactI=Y1*I
#Y2interactI=log(Y2*I)

#test1=Y1[1001:1200]
#test2=Y2[1001:1200]
Y1inc=diff(Y1[1:1200])
Y2inc=diff(Y2[1:1200])



# Assign the data to the covariates sets
covariates=cbind(Y1[1:996],Y1[2:997],Y1[3:998],Y1[4:999],
                 Y2[1:996],Y2[2:997],Y2[3:998],Y2[4:999],
                 I[1:996],I[2:997],I[3:998],I[4:999],
                 Y1square[1:996],Y1square[2:997],Y1square[3:998],Y1square[4:999],
                 Y2square[1:996],Y2square[2:997],Y2square[3:998],Y2square[4:999],
                 Isquare[1:996],Isquare[2:997],Isquare[3:998],Isquare[4:999],
                 Y1interactY2[1:996],Y1interactY2[2:997],Y1interactY2[3:998], Y1interactY2[4:999])
#Y1interactI[1:996],Y1interactI[2:997],Y1interactI[3:998], Y1interactI[4:999],
```

```
#Y2interactI[1:996],Y2interactI[2:997],Y2interactI[3:998], Y1interactI[4:999])
#we are giving column names here. Note Y1t4 means Y1(t-4) and Y1t3=Y1(t-3) etc. The rest is
# similar idea. It will be easy to read the important variable when we use varImpPlot function.
colnames(covariates)=c("Y1t4","Y1t3","Y1t2","Y1t1",
                       "Y2t4","Y2t3","Y2t2","Y2t1",
                       "It4","It3","It2","It1",
                       "Y1sqt4","Y1sqt3","Y1sqt2","Y1sqt1",
                       "Y2sqt4","Y2sqt3","Y2sqt2","Y2sqt1",
                       "Isqt4","Isqt3","Isqt2","Isqt1",
                       "Y1Y2t4","Y1Y2t3","Y1Y2t2","Y1Y2t1")
#                       "Y1It4","Y1It3","Y1It2","Y1It1",
#                       "Y2It4","Y2It3","Y2It2","Y2It1")
# Assign the data to the covariate test sets
#covariates_test=cbind(Y1[997:1196],Y1[998:1197],Y1[999:1198],Y1[1000:1199],
#                      Y2[997:1196],Y2[998:1197],Y2[999:1198],Y2[1000:1199],
#                     I[997:1196],I[998:1197],I[999:1198],I[1000:1199],
#                   Y1inc[997:1196],Y1inc[998:1197],Y1inc[999:1198],
#                  Y2inc[997:1196],Y2inc[998:1197],Y2inc[999:1198])

covariates_test=cbind(Y1[997:1196],Y1[998:1197],Y1[999:1198],Y1[1000:1199],
                      Y2[997:1196],Y2[998:1197],Y2[999:1198],Y2[1000:1199],
                      I[997:1196],I[998:1197],I[999:1198],I[1000:1199],

                      Y1square[997:1196],Y1square[998:1197],Y1square[999:1198],Y1square[1000:1199],
                      Y2square[997:1196],Y2square[998:1197],Y2square[999:1198],Y2square[1000:1199],
                      Isquare[997:1196],Isquare[998:1197],Isquare[999:1198],Isquare[1000:1199],

                      Y1interactY2[997:1196],Y1interactY2[998:1197],
                      Y1interactY2[999:1198],Y1interactY2[1000:1199])

#  Y1interactI[997:1196],Y1interactI[998:1197],
# Y1interactI[999:1198],Y1interactI[1000:1199],
#Y2interactI[997:1196],Y2interactI[998:1197],
#Y2interactI[999:1198],Y2interactI[1000:1199])
colnames(covariates_test)=c("Y1t4","Y1t3","Y1t2","Y1t1",
                            "Y2t4","Y2t3","Y2t2","Y2t1",
                            "It4","It3","It2","It1",
                            "Y1sqt4","Y1sqt3","Y1sqt2","Y1sqt1",
                            "Y2sqt4","Y2sqt3","Y2sqt2","Y2sqt1",
                            "Isqt4","Isqt3","Isqt2","Isqt1",
                            "Y1Y2t4","Y1Y2t3","Y1Y2t2","Y1Y2t1")
#"Y1It4","Y1It3","Y1It2","Y1It1",
#"Y2It4","Y2It3","Y2It2","Y2It1")

############################################ part a

#############################################################################################
#First, creating random forests for Y1 and Y2
# Create a Random Forest model  for Y1 now with default parameters
model1_a = randomForest(Y1[5:1000] ~ ., data = covariates, importance=TRUE)
varImpPlot(model1_a) #this tells me that Isquare and Y1 interecation with I variables are important.
```

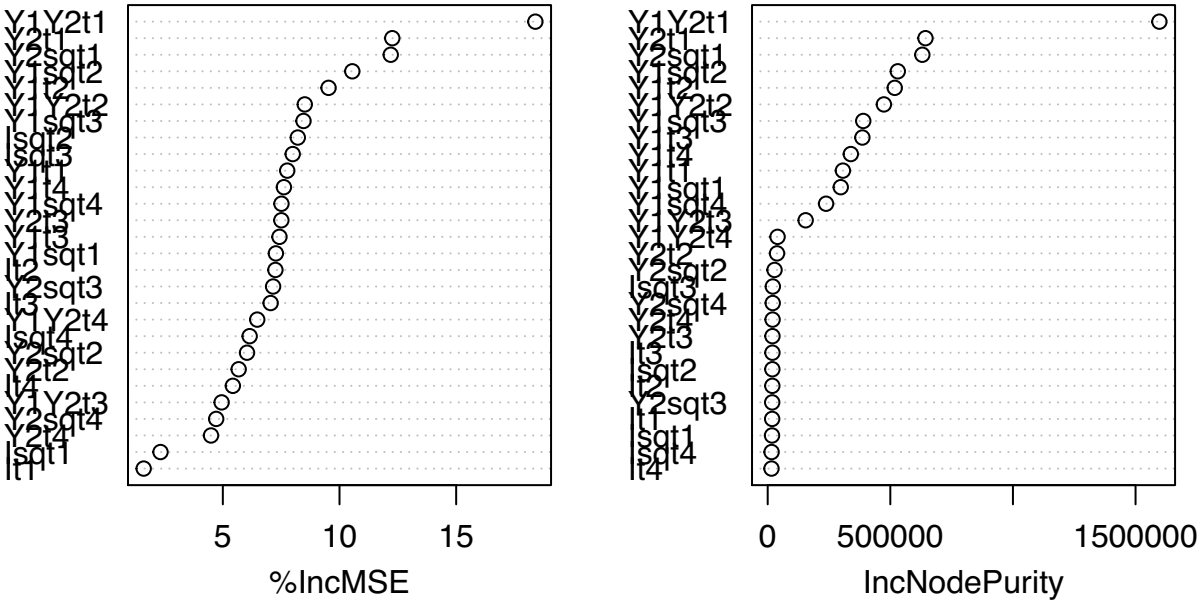
I are more important.

model1_a



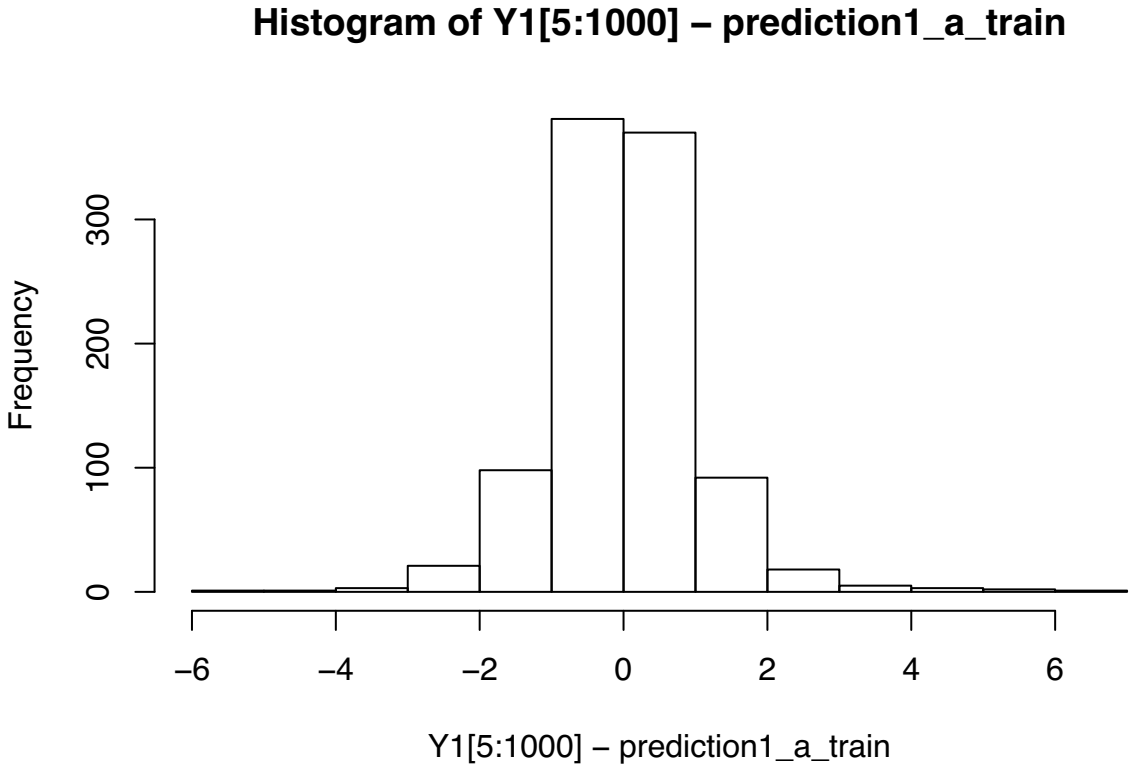

```
# Create a Random Forest model with default parameters for Y2 now
model2_a = randomForest(Y2[5:1000] ~ ., data = covariates, importance=TRUE)
varImpPlot(model2_a)# here we see that Y1Y2(t-1) is the most important variable.
```

model2_a



```
########################################  Done, creating models for Y1 and Y2
```

```
#########################################################################################
#first lets check, How our model is doing on train set

#I will do prediction on the train set for Y1
prediction1_a_train= predict(model1_a,covariates)
#I will check the histogram of  residuals and mean square error.
hist(Y1[5:1000]- prediction1_a_train ) #it looks normal, So that is good.
```




```
MSE1_a_train= 1/996 * sum((Y1[5:1000]- prediction1_a_train)^2)
MSE1_a_train #this is pretty small which is expected because we are testing on train set. This is for Y
```

```
## [1] 1.097738
```

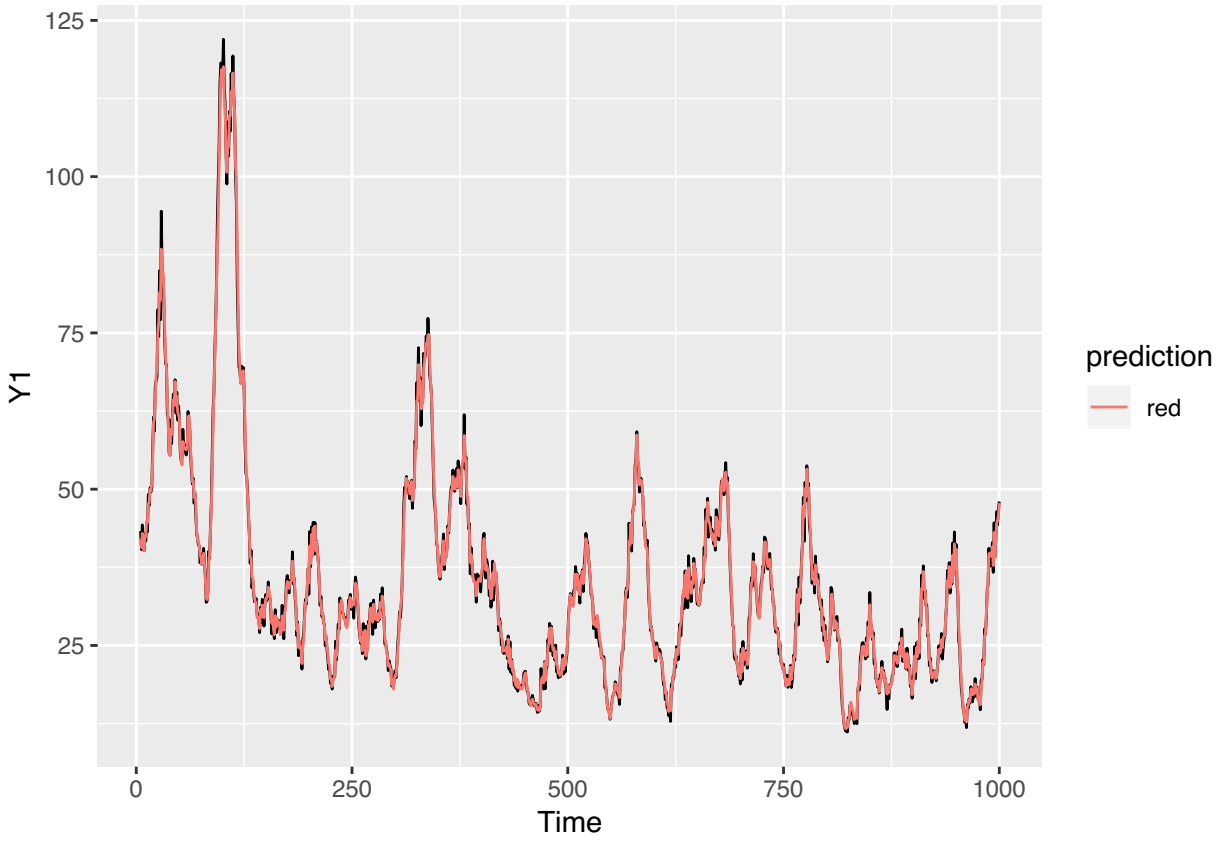
```
#Now let us plot and see  how it looks.
library("ggplot2")
```

```
##
## Attaching package: 'ggplot2'
```

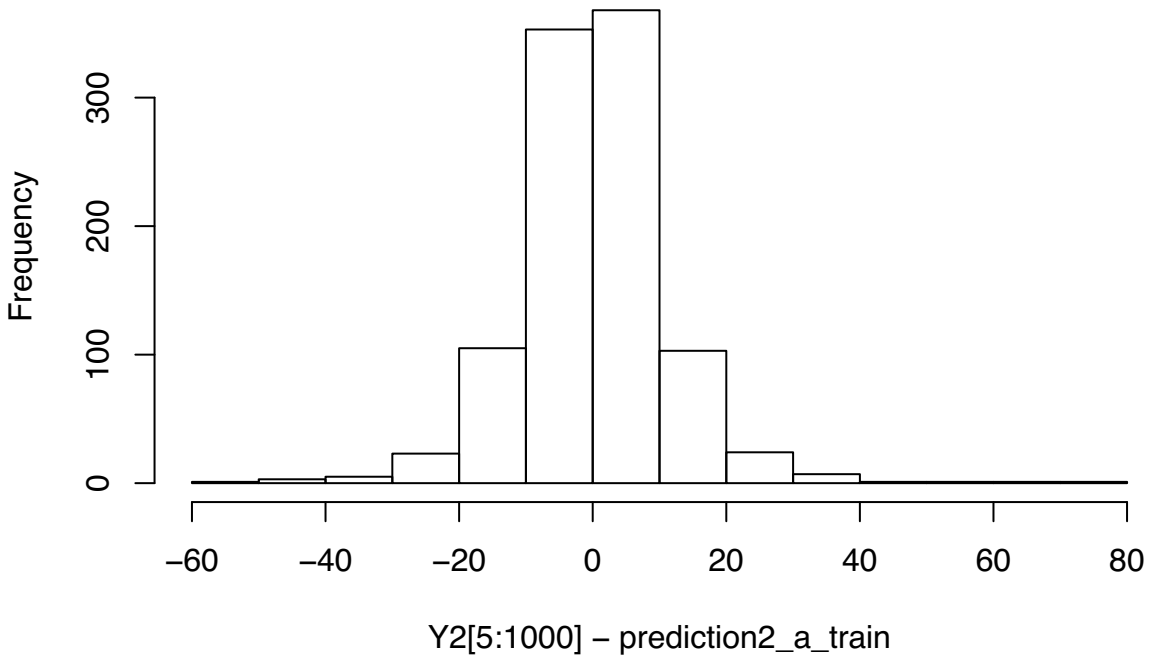
```
## The following object is masked from 'package:randomForest':
##
##     margin
```

```
plot1_train_a=ggplot(data=NULL,aes(Time, Y1, colour = prediction))+
  geom_line(aes(x=c(5:1000), y=Y1[5:1000]), color="black")+
  geom_line(aes(x=c(5:1000), y=prediction1_a_train, color="red"))
```

```
print(plot1_train_a) #this looks so good as we expected
```




```
#now let us do same calculations for Y2.
#I will do prediction on the train set for Y2
prediction2_a_train= predict(model2_a,covariates)
#I will check the histogram of  residuals and mean square error.
hist(Y2[5:1000]- prediction2_a_train ) #it looks normal, So that is good.
```

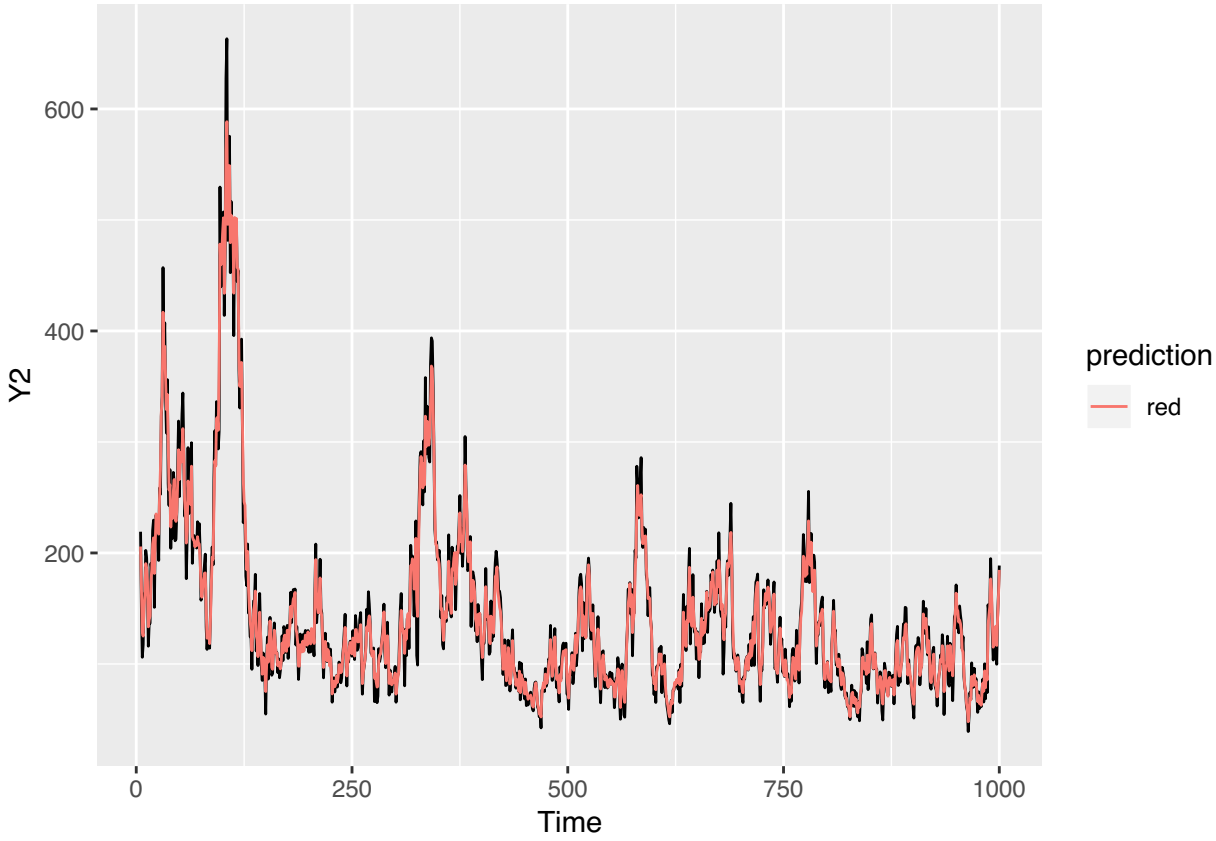



```
MSE2_a_train= 1/996 * sum((Y2[5:1000]- prediction2_a_train)^2)
MSE2_a_train #this is pretty small which is expected because we are testing on train set. This is for Y
```

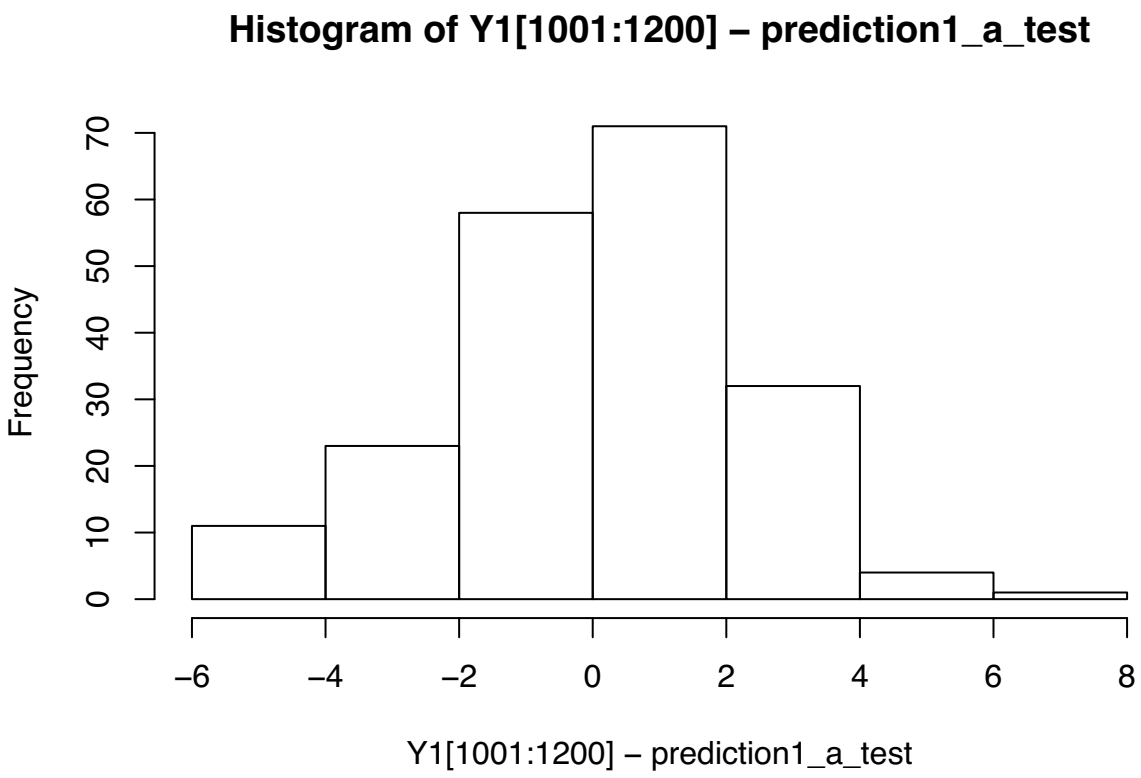
This is for Y2.

```
## [1] 122.7025
```

```
#Now let us plot and see  how it looks.
library("ggplot2")
plot2_train_a=ggplot(data=NULL,aes(Time, Y2, colour = prediction))+
  geom_line(aes(x=c(5:1000), y=Y2[5:1000]), color="black")+
  geom_line(aes(x=c(5:1000), y=prediction2_a_train, color="red"))
print(plot2_train_a) #this looks so good as we expected.
```



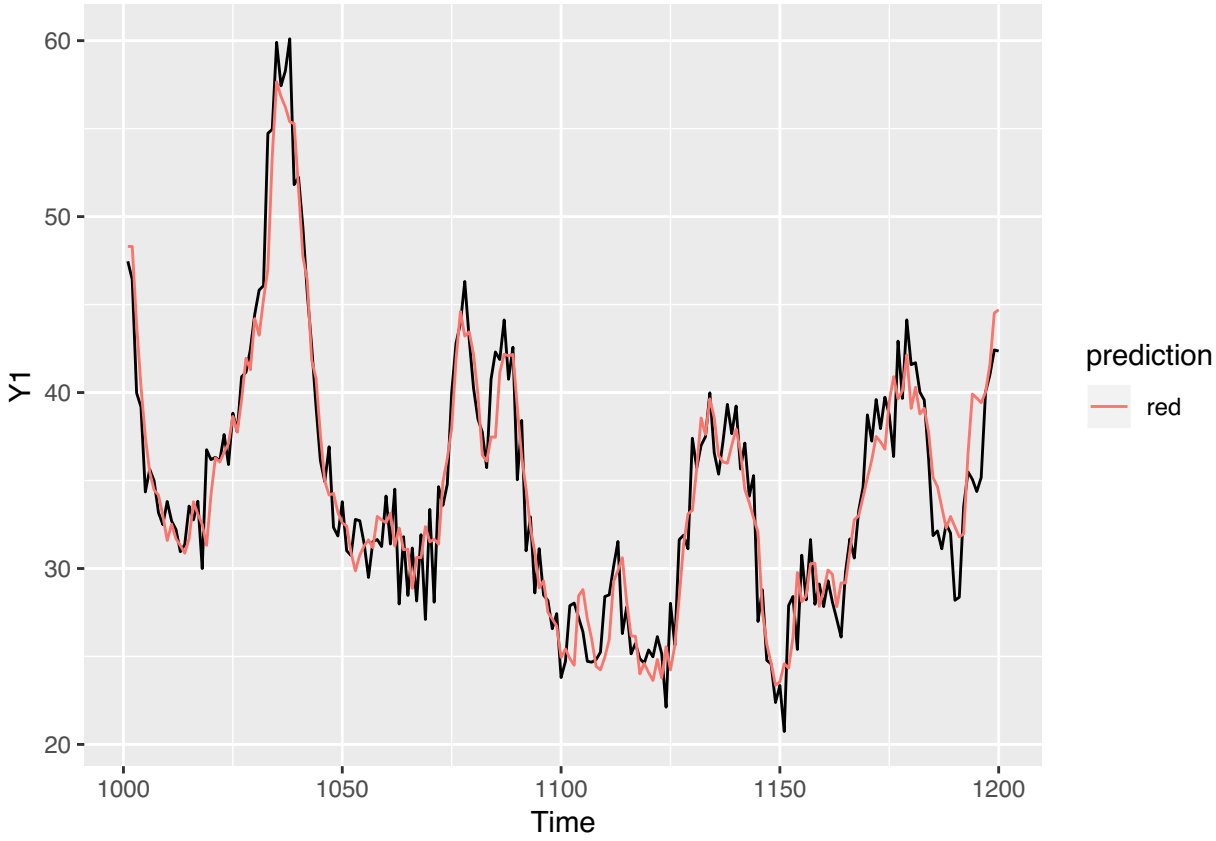

```
############################################ finishing, How our model is doing on train set



#############################################################################################
#Now, it is time to check on  a test set
#I will do prediction on the test set for Y1
prediction1_a_test= predict(model1_a, covariates_test)
#I will check the histogram of  residuals and mean square error for Y1
hist(Y1[1001:1200]- prediction1_a_test ) #it looks normal, So that is good.
```



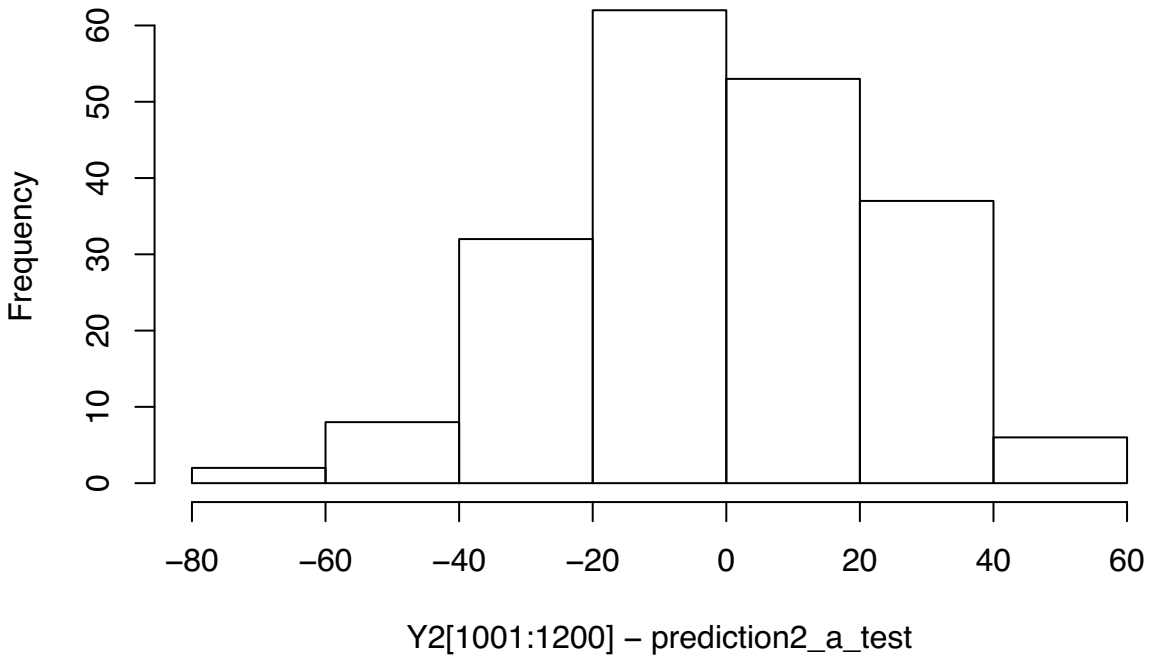

```
MSE1_a_test= 1/200 * sum((Y1[1001:1200]- prediction1_a_test)^2)
MSE1_a_test
```

```
## [1] 4.918447
```

```
library("ggplot2")
plot1_a_test=ggplot(data=NULL,aes(Time, Y1, colour = prediction))+
  geom_line(aes(x=c(1001:1200), y=Y1[1001:1200]), color="black")+
  geom_line(aes(x=c(1001:1200), y=prediction1_a_test, color="red"))
print(plot1_a_test)
```



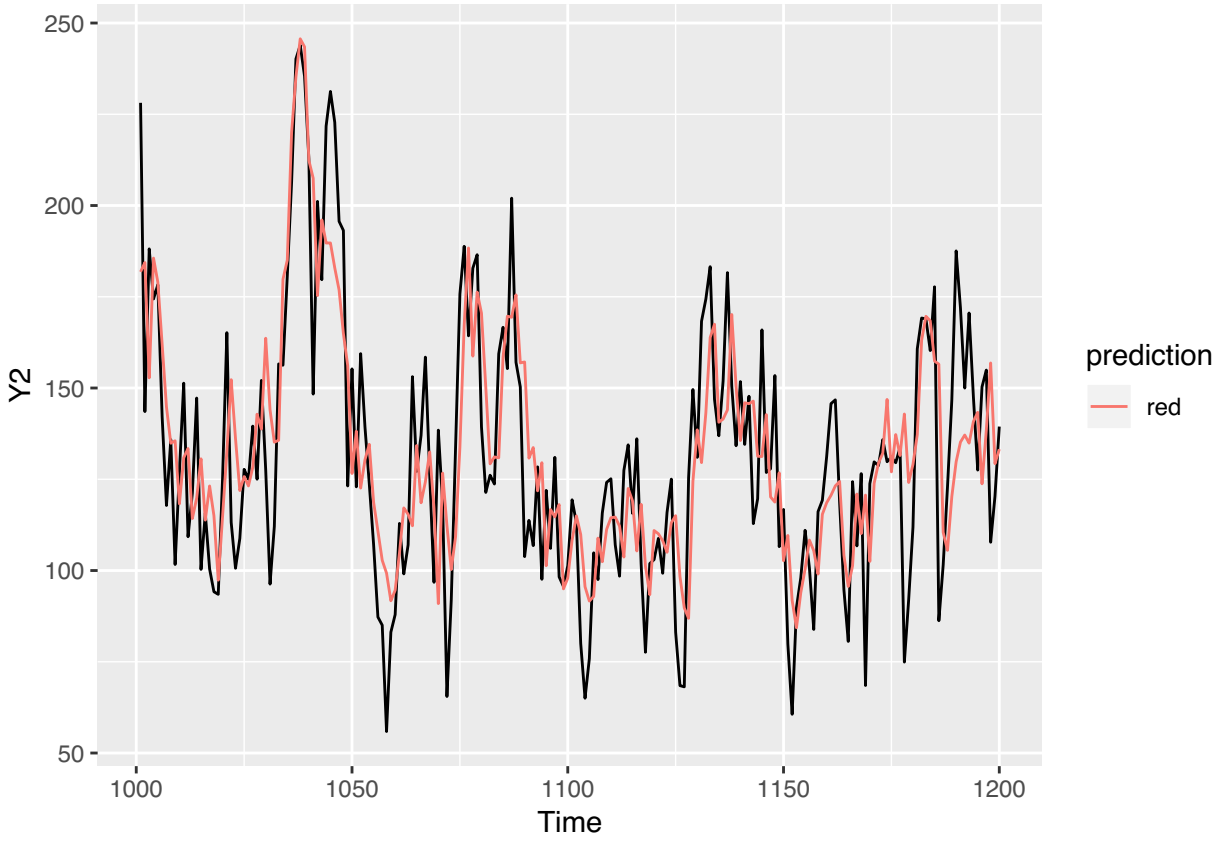

```
#I will do prediction on the test set for Y2
prediction2_a_test= predict(model2_a, covariates_test)
#I will check the histogram of  residuals and mean square error for Y2
hist(Y2[1001:1200]    - prediction2_a_test ) #it looks normal also it is good.
```




```
MSE2_a_test = 1/200 * sum((Y2[1001:1200]   - prediction2_a_test )^2)
MSE2_a_test
```

```
## [1] 572.9075
```

```
plot2_a_test=ggplot(data=NULL,aes(Time, Y2, colour = prediction))+
  geom_line(aes(x=c(1001:1200), y=Y2[1001:1200]), color="black")+
  geom_line(aes(x=c(1001:1200), y=prediction2_a_test, color="red"))
print(plot2_a_test)
```



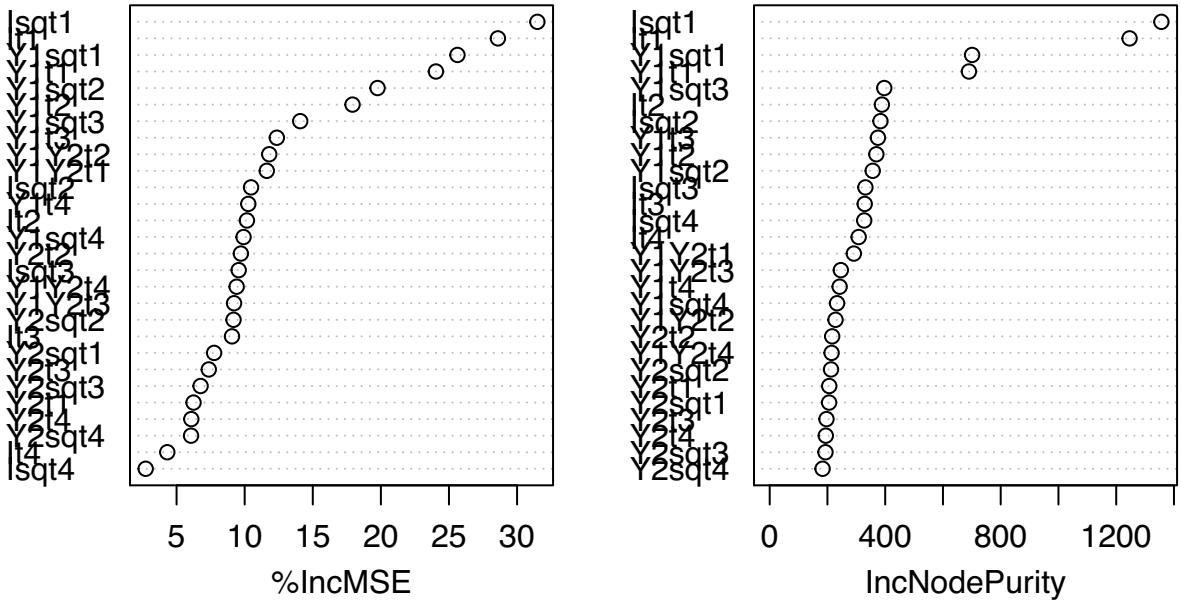

```
############################################ finishing, testing on a test set


#Our conclusion as you can see from MSE and plot is:
#prediction for Y1 is  far better for prediction for Y2.
```

Part a finishes here.

```
############################################## part b

# Now let's create models on increments
#################################################################################################
# Create a Random Forest model with default parameters for Y1inc now
model1_b = randomForest(Y1inc[4:999] ~ ., data = covariates, importance=TRUE)
varImpPlot(model1_b) #this tells me that I(t-1) is the most important.
```

I sq(t-1)

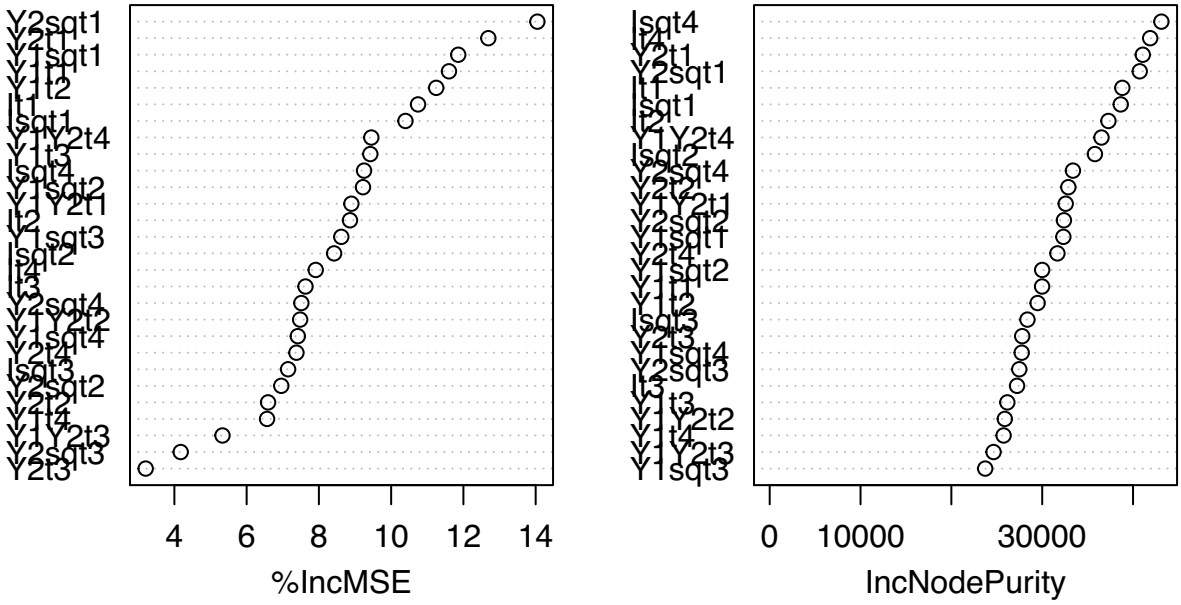
model1_b




```
# Create a Random Forest model with default parameters for Y2inc now
model2_b = randomForest(Y2inc[5:1000] ~ ., data = covariates,importance=TRUE)
varImpPlot(model2_b) #this tells me that Y2(t-1) is the most important.
```

Y2 square (t-1)

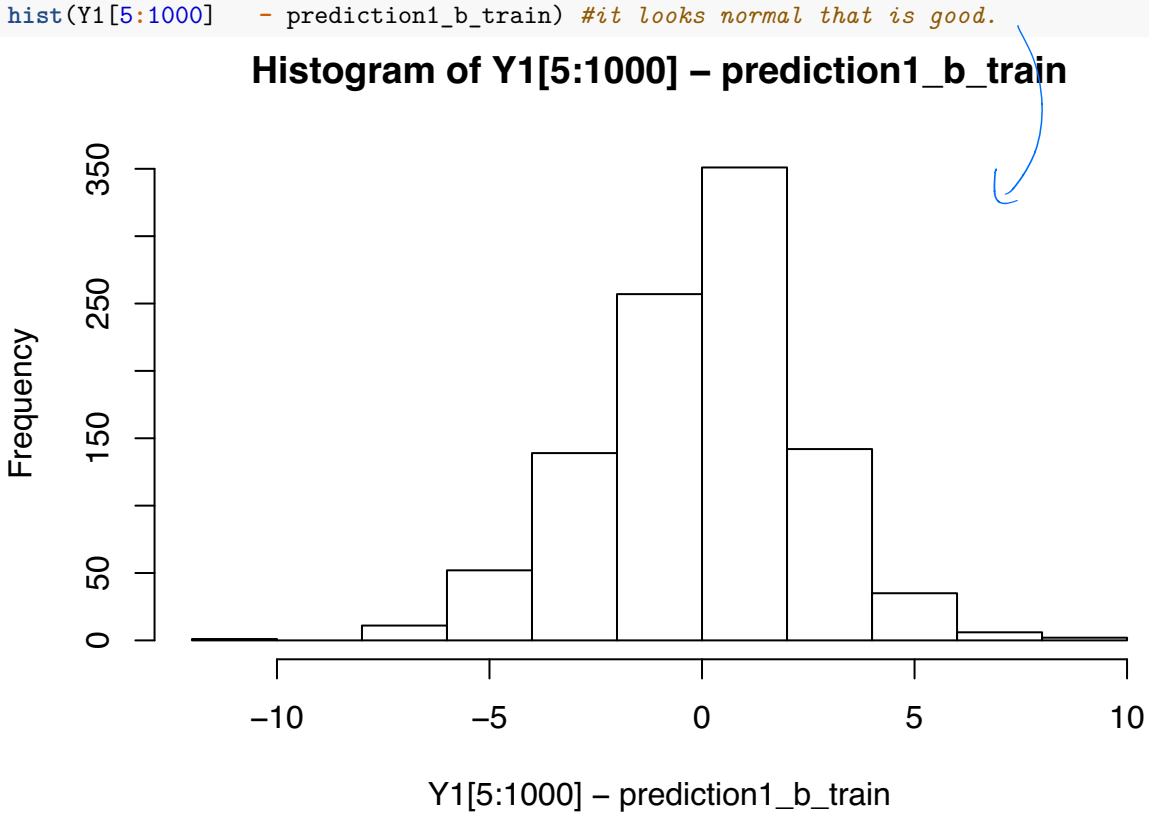
model2_b



```
#############################################################################################
#First, lets  check on training set how well we are doing.
#I will do prediction for Y1inc on the train set
prediction1_b_inc = predict(model1_b,covariates)
prediction1_b_train=Y1[5:1000] + prediction1_b_inc #we added the original one to get prediction on Y1
#prediction1_b
head(prediction1_b_train)
```

```
##        1        2        3        4        5        6
## 40.42699 37.91030 46.31380 39.63054 38.46564 45.22874
```

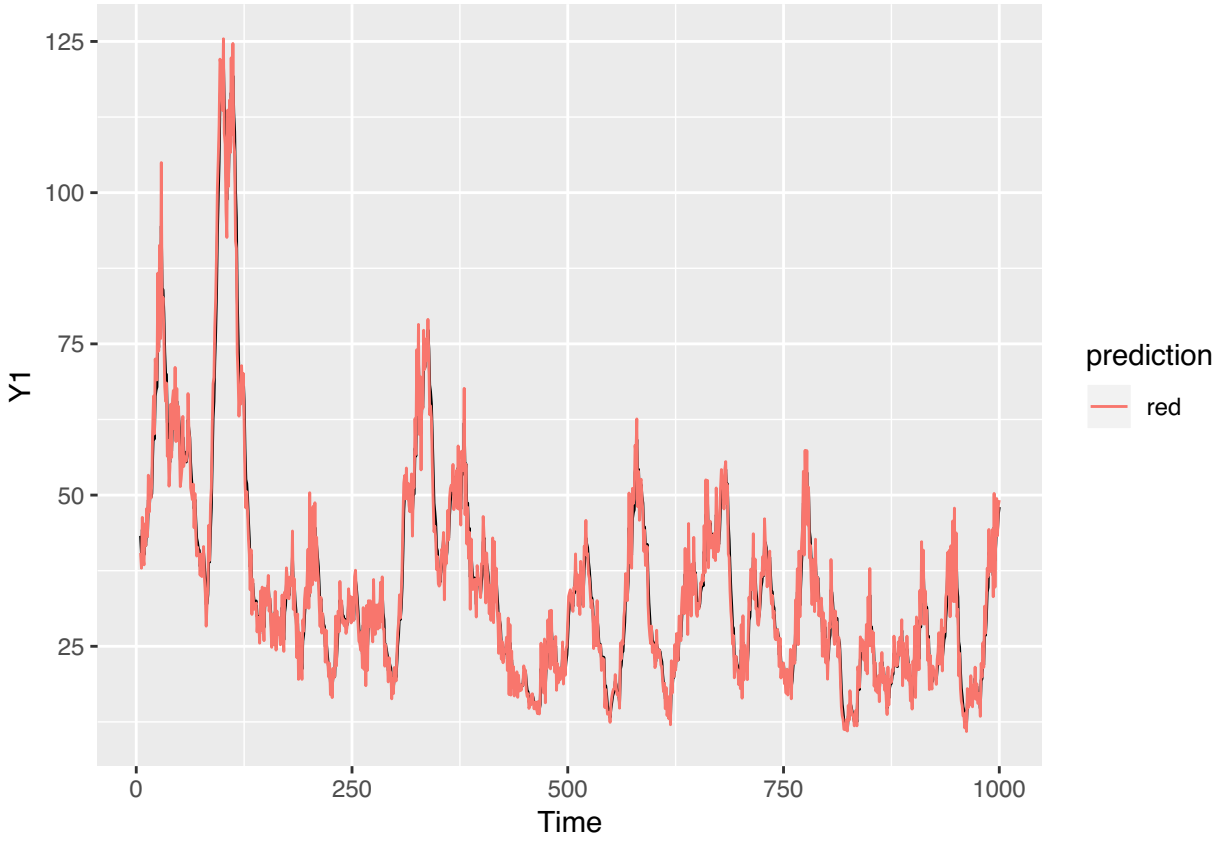
```
hist(Y1[5:1000]   - prediction1_b_train) #it looks normal that is good.
```




```
MSE1_b_train= 1/996 * sum((Y1[5:1000]   - prediction1_b_train)^2)
MSE1_b_train#this is mean square error for increments Y1inc
```

```
## [1] 6.173792
```

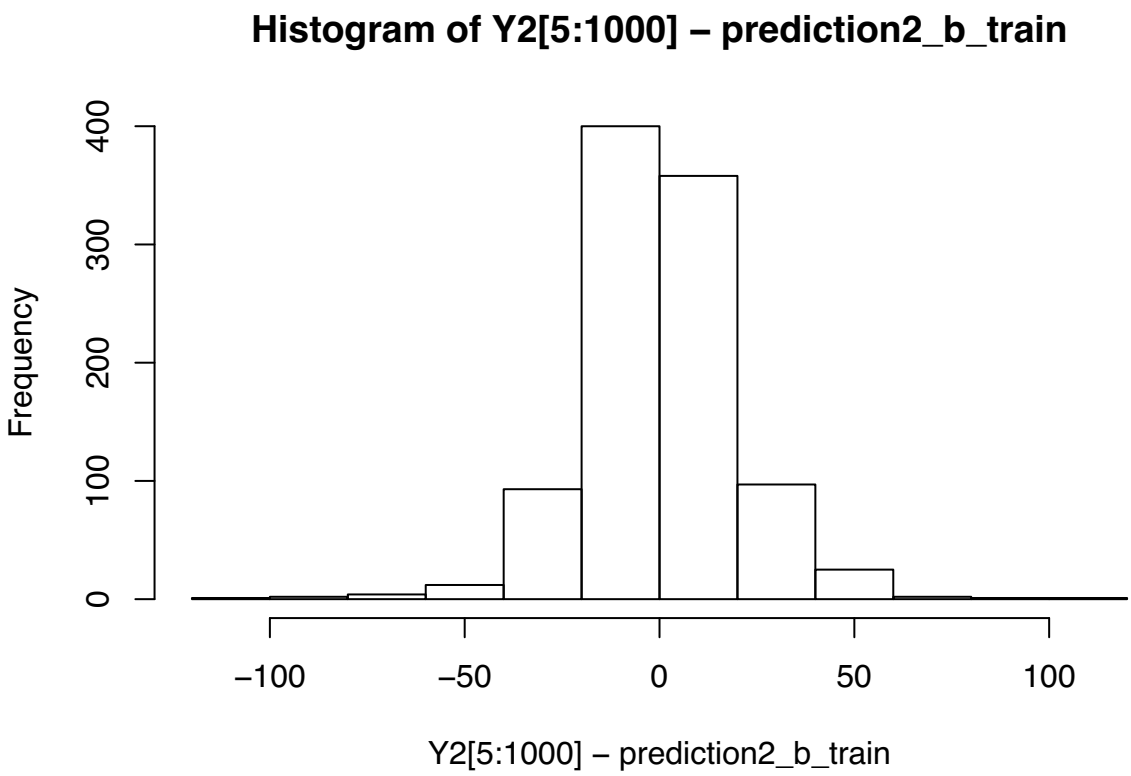
```
plot1_train_b=ggplot(data=NULL,aes(Time, Y1, colour = prediction))+
  geom_line(aes(x=c(5:1000), y=Y1[5:1000]), color="black")+
  geom_line(aes(x=c(5:1000), y=prediction1_b_train, color="red"))
print(plot1_train_b) #this looks so good as we expected
```




```
#prediction for increments in Y2. I will do prediction for Y2inc on the train set.
prediction2_b_inc = predict(model2_b,covariates)
prediction2_b_train=Y2[5:1000]  + prediction2_b_inc  #we added the original one to get prediction on Y2
#prediction2_b
head(prediction2_b_train)
```

```
##        1        2        3        4        5        6
## 174.2458 111.7770 115.6104 147.6885 167.1237 192.8537
```

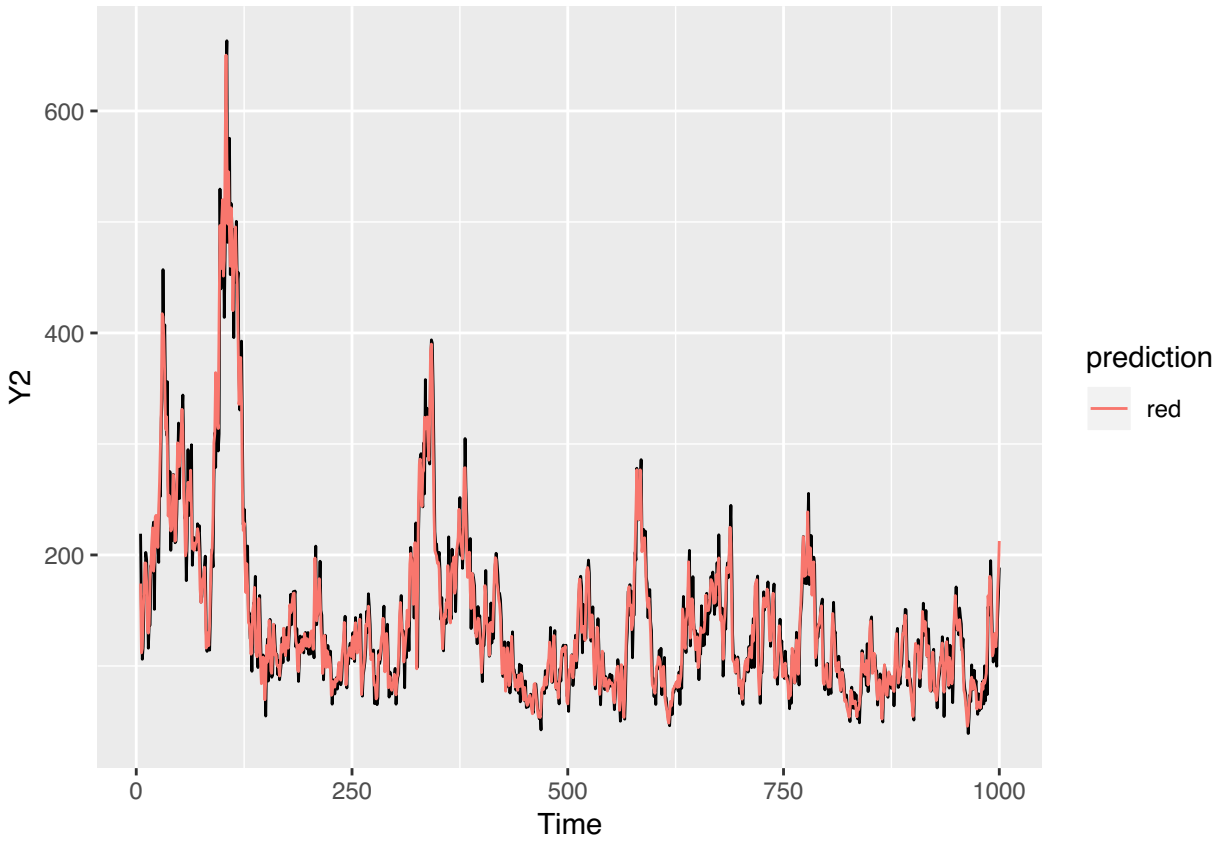
```
#residuals2_b
hist(Y2[5:1000]  - prediction2_b_train) #it looks normal that is good.
```




```
MSE2_b_train = 1/996 * sum(( Y2[5:1000]   - prediction2_b_train)^2)
MSE2_b_train #this is mean square error for increments Y2inc
```

```
## [1] 378.267
```

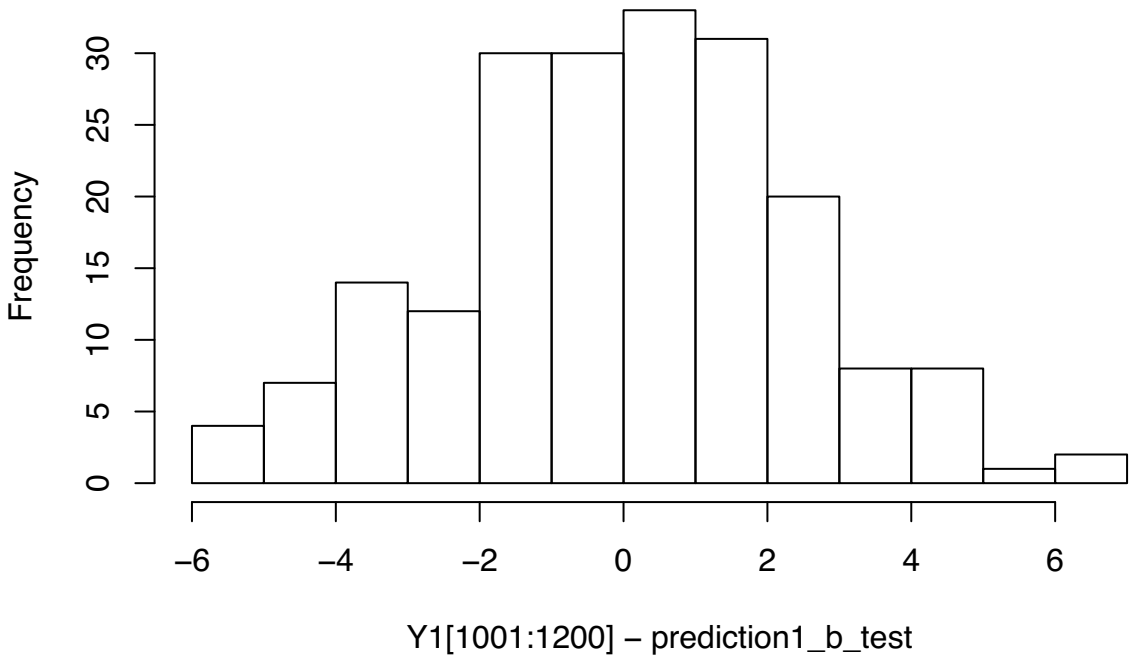
```
plot2_train_b=ggplot(data=NULL,aes(Time, Y2, colour = prediction))+
  geom_line(aes(x=c(5:1000), y=Y2[5:1000]), color="black")+
  geom_line(aes(x=c(5:1000), y=prediction2_b_train, color="red"))
print(plot2_train_b) #this looks so good as we expected
```




```
########################################################## finishing the test set.

##When we look mean square errors on train sets
#we see that MSE1_a_train way better than MSE1_b_train
#Similarly,  MSE2_a_train way better than MSE2_b_train
# So we conclude model in a is better than model in b on train set.
#lets see what is the case on test set.


#############################################################################################
#Now, it is time to check on  a test set
#I will do prediction for Y1inc on the test set
prediction1_b_inc = predict(model1_b, covariates_test)
prediction1_b_test=Y1[1000:1199] + prediction1_b_inc #we added the original one to get prediction on Y1
#prediction1_b
head(prediction1_b_test)
```

```
##        1        2        3        4        5        6
## 46.07365 46.29716 43.73886 37.32133 37.97970 34.60023
```

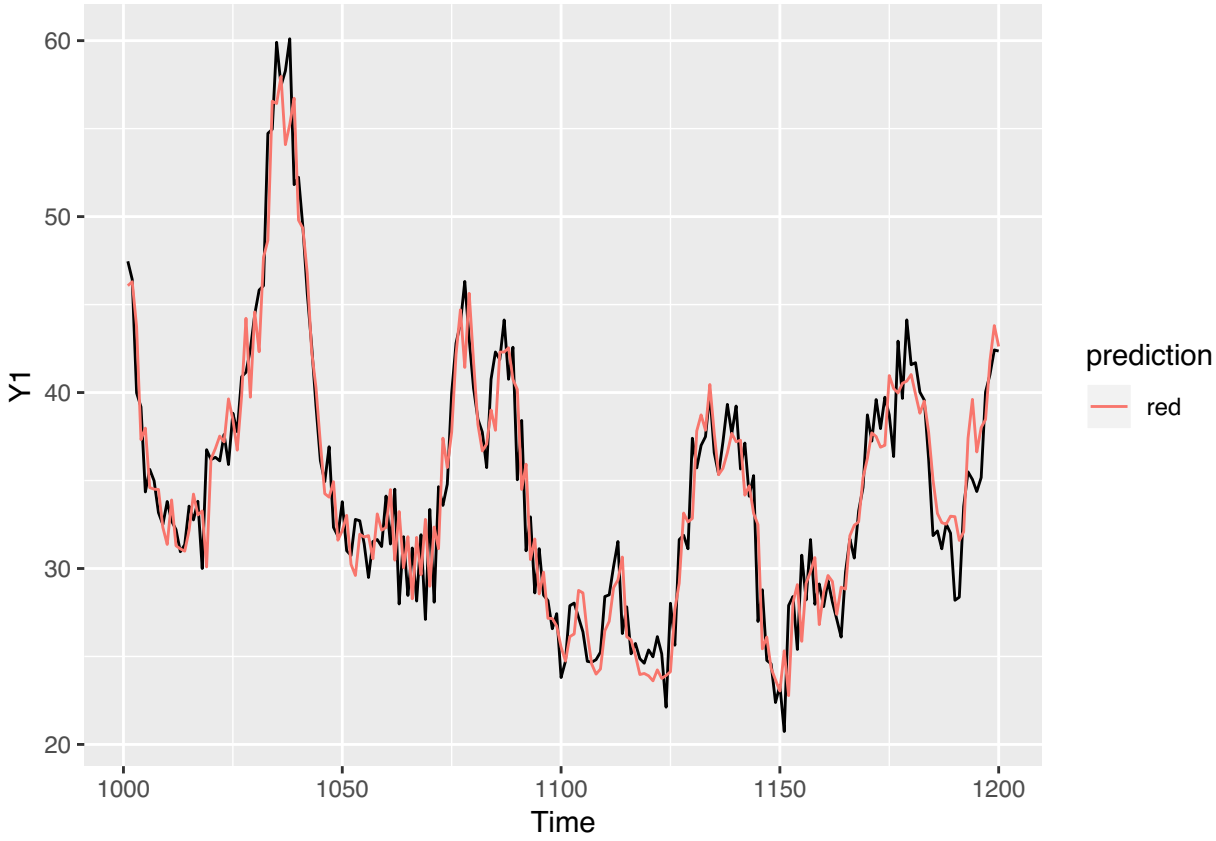
```
hist(Y1[1001:1200]   - prediction1_b_test) #it looks normal that is good.
```




```
MSE1_b_test= 1/200 * sum((Y1[1001:1200]   - prediction1_b_test)^2)
MSE1_b_test #this is mean square error for increments Y1
```

```
## [1] 5.832033
```

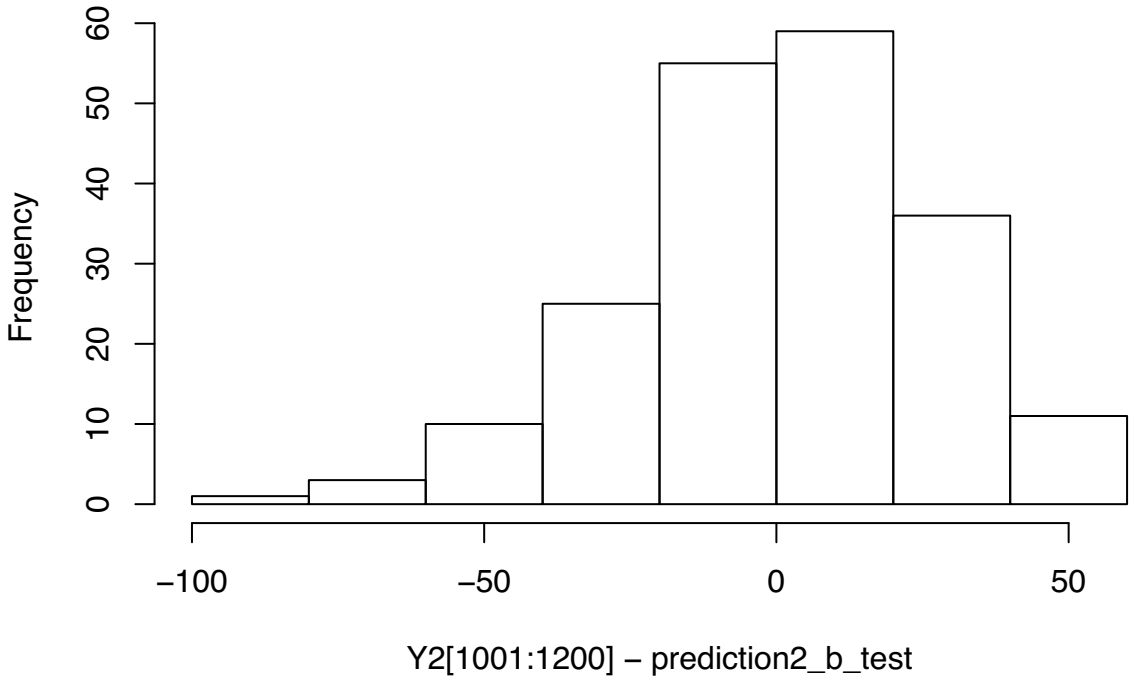
```
plot1_b_test=ggplot(data=NULL,aes(Time, Y1, colour = prediction))+
  geom_line(aes(x=c(1001:1200), y=Y1[1001:1200]), color="black")+
  geom_line(aes(x=c(1001:1200), y=prediction1_b_test, color="red"))
print(plot1_b_test)
```




```
#prediction for increments in Y2
#I will do prediction for Y2inc on the test set
prediction2_b_inc = predict(model2_b, covariates_test)
prediction2_b_test=Y2[1000:1199] + prediction2_b_inc  #we added the original one to get prediction on Y
#prediction2_b
head(prediction2_b_test)
```

```
##        1        2        3        4        5        6
## 193.8985 210.0268 146.4600 188.1269 171.1992 171.0874
```

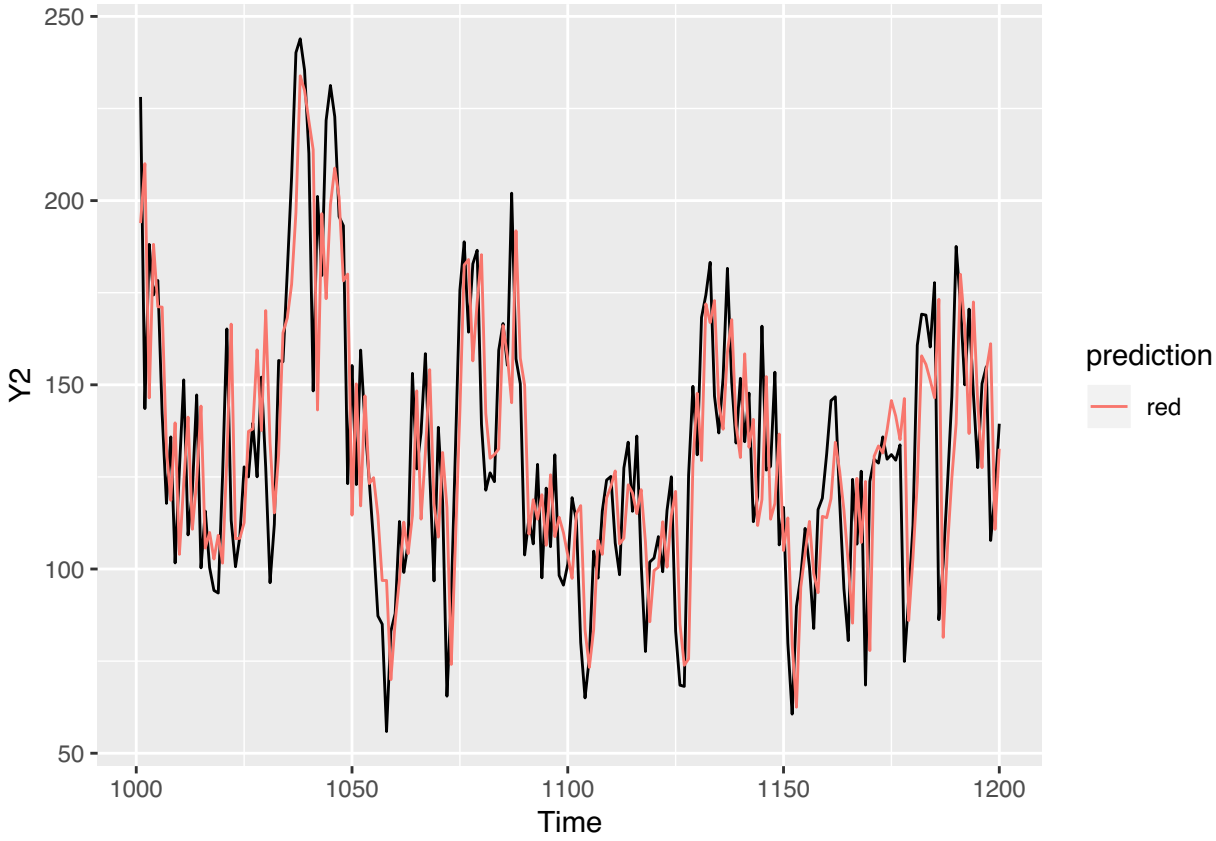
```
hist( Y2[1001:1200]   - prediction2_b_test) #it looks normal that is good.
```




```
MSE2_b_test= 1/200 * sum(( Y2[1001:1200]    - prediction2_b_test)^2)
MSE2_b_test#this is mean square error for increments Y2
```

```
## [1] 682.5334
```

```
plot2_b_test=ggplot(data=NULL,aes(Time, Y2, colour = prediction))+
  geom_line(aes(x=c(1001:1200), y=Y2[1001:1200]), color="black")+
  geom_line(aes(x=c(1001:1200), y=prediction2_b_test, color="red"))
print(plot2_b_test)
```

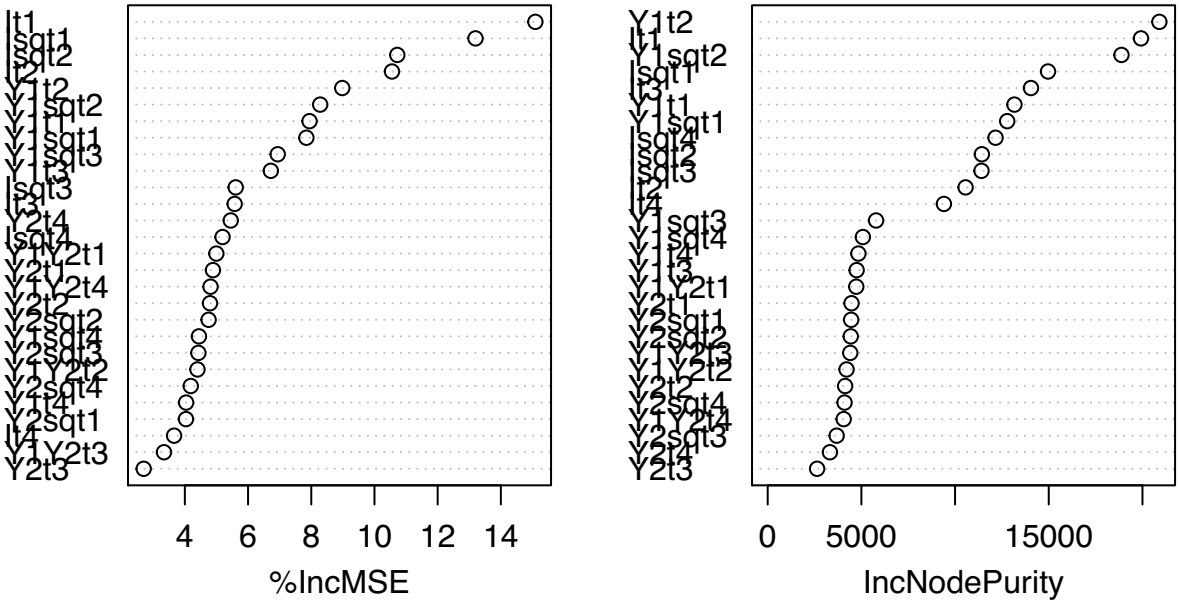



```
########################################################## finishing the test set.
#prediction for Y1inc is  far better for prediction for Y2inc.

# Even though MSE1_b_test for Y1 is better in Model b, MSE2_b_test for Y2 is far worse in model b,
#We compared MSE1_a_test with MSE1_b_test and MSE2_a_test with MSE2_b_test
# So we pick model in a, over all as our best model.
```

Part b finishes here.

```
########  part c

#########################################################################################################
V1=(Y1[5:1000]-predict(model1_a))^2  #variance for Y1
V2=(Y2[5:1000]-predict(model2_a))^2 #variance for Y2
Cov12=sqrt(V1*V2)  ##covariance for Y1  and Y2

#########################################################
# Create a Random Forests for variance and covariance
#create random forest for V1
model1_c=randomForest(V1 ~ ., data = covariates,  importance=TRUE)
varImpPlot(model1_c) #this tells me that I(t-1) is the most important.
```

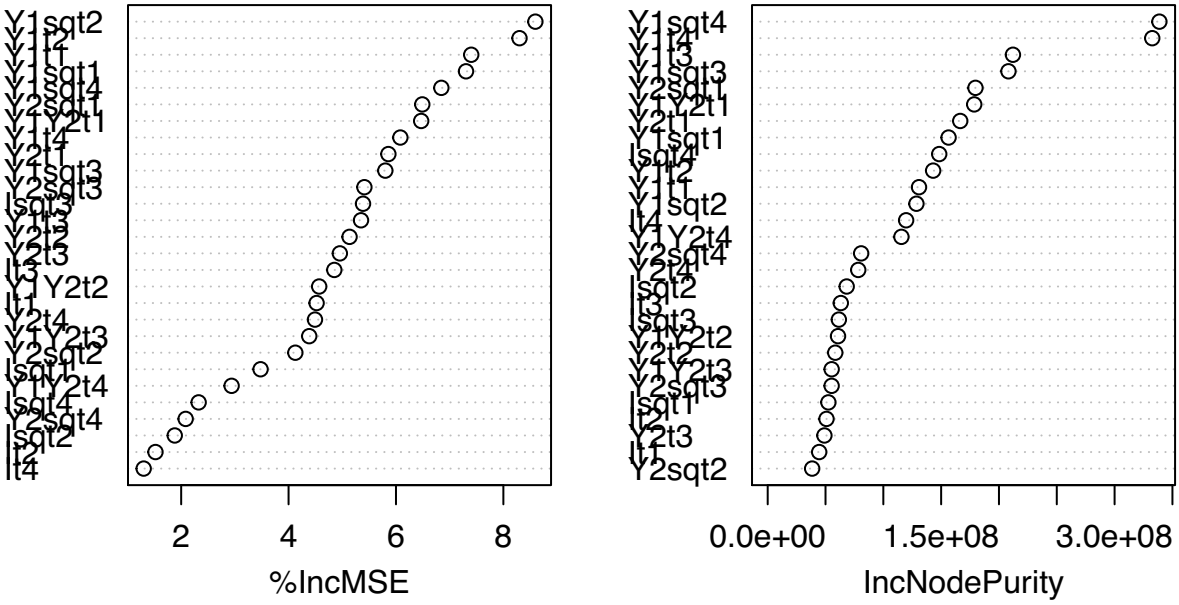
model1_c



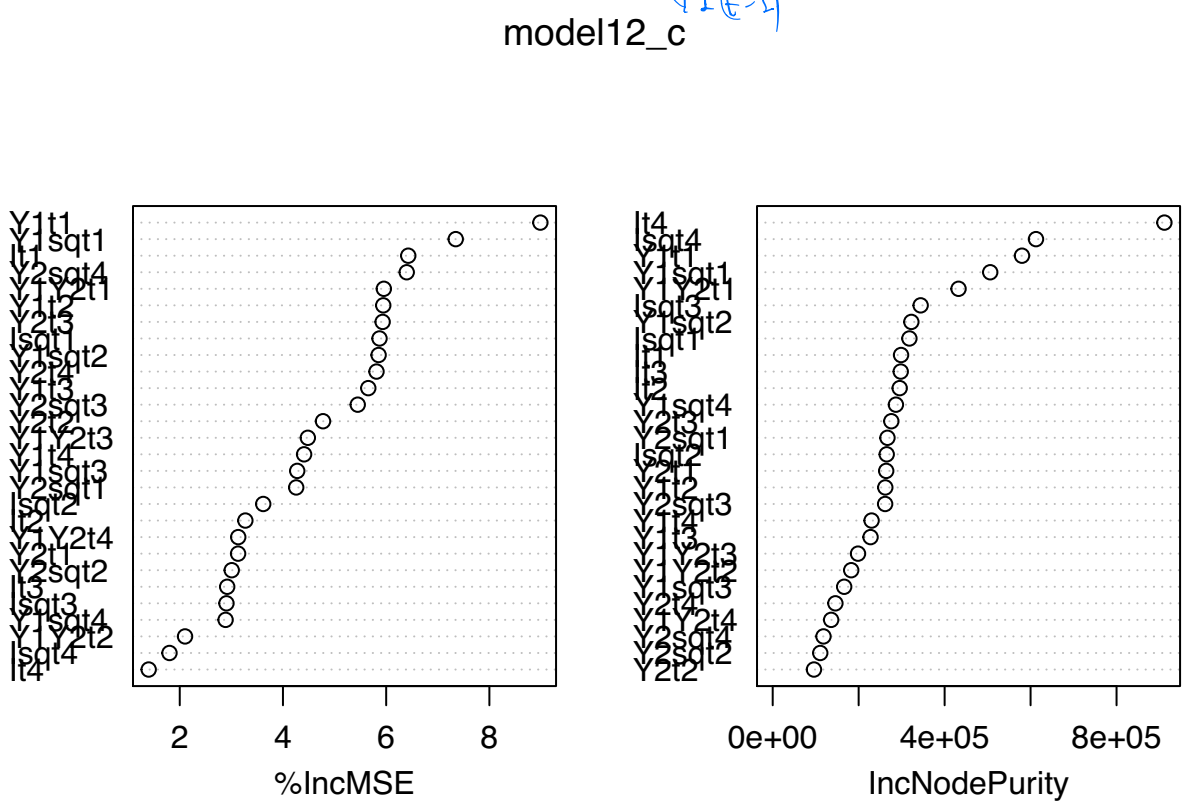

```
#create random forest for V2
model2_c=randomForest(V2 ~ ., data = covariates,importance=TRUE)
varImpPlot(model2_c) #this tells me that ~~Y1(t-2)~~ is the most important.
```

Y1 square (t-2)

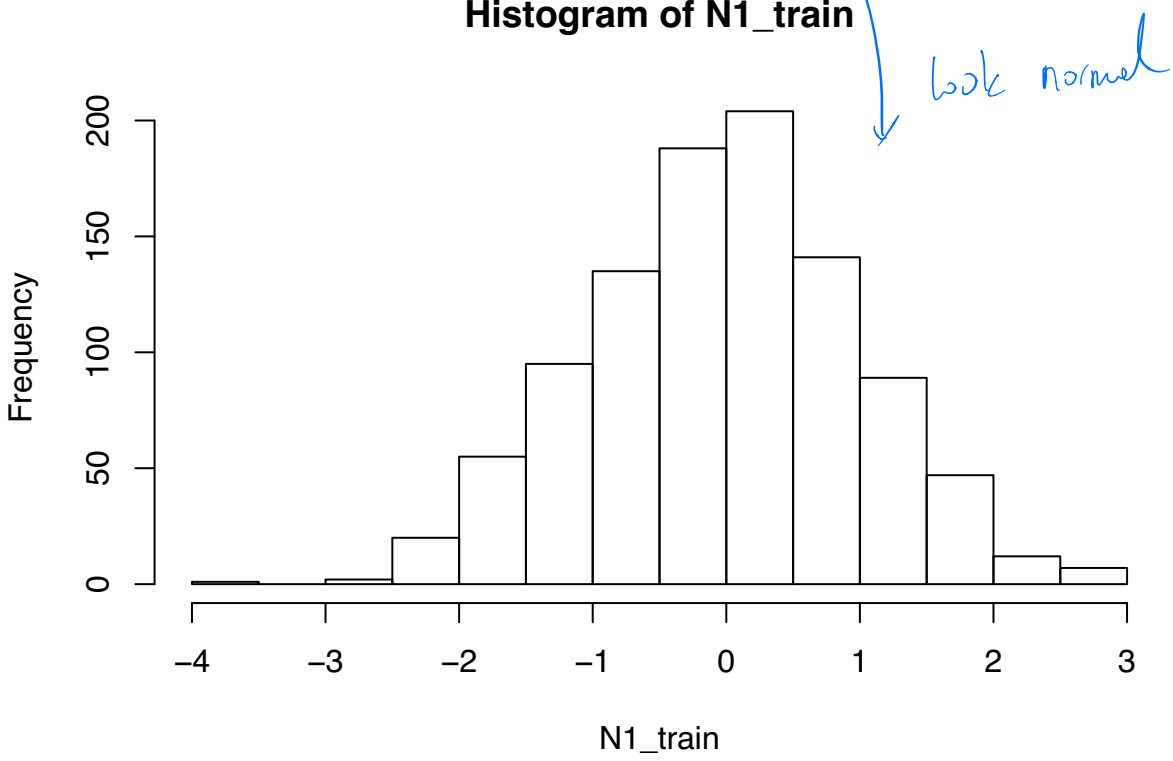
model2_c



```r
#create random forest for V12
model12_c=randomForest(Cov12 ~ ., data = covariates,importance=TRUE)
varImpPlot(model12_c) #this tells me that Y2t(t-1) is the most important.
```

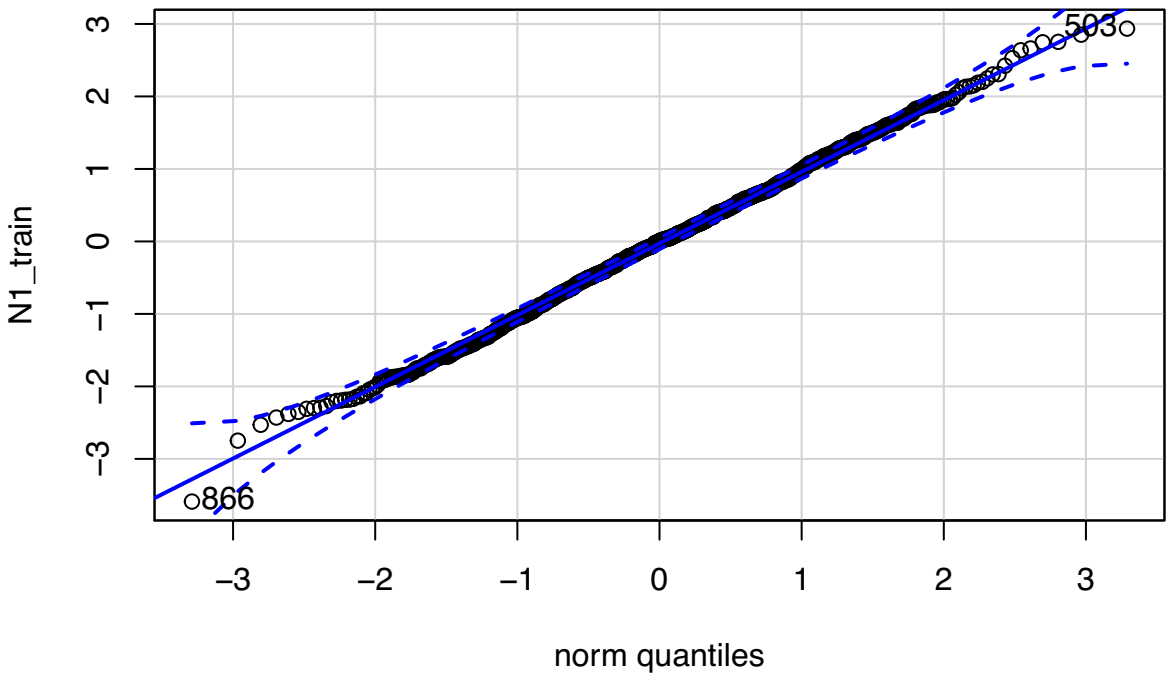



```r
##########################################################

##########################################################################################################################
#First we will check on train set
Y1sd_train = predict(model1_c) #predicting the variance for Y1
Y2sd_train = predict(model2_c)  #predicting the variance for Y2
Y1Y2cov_train = predict(model12_c)   #predicting the covariance for Y1 and Y2

#we expect that to be normal
N1_train=(Y1[5:1000] - predict(model1_a))/(sqrt(Y1sd_train))
hist(N1_train)
```



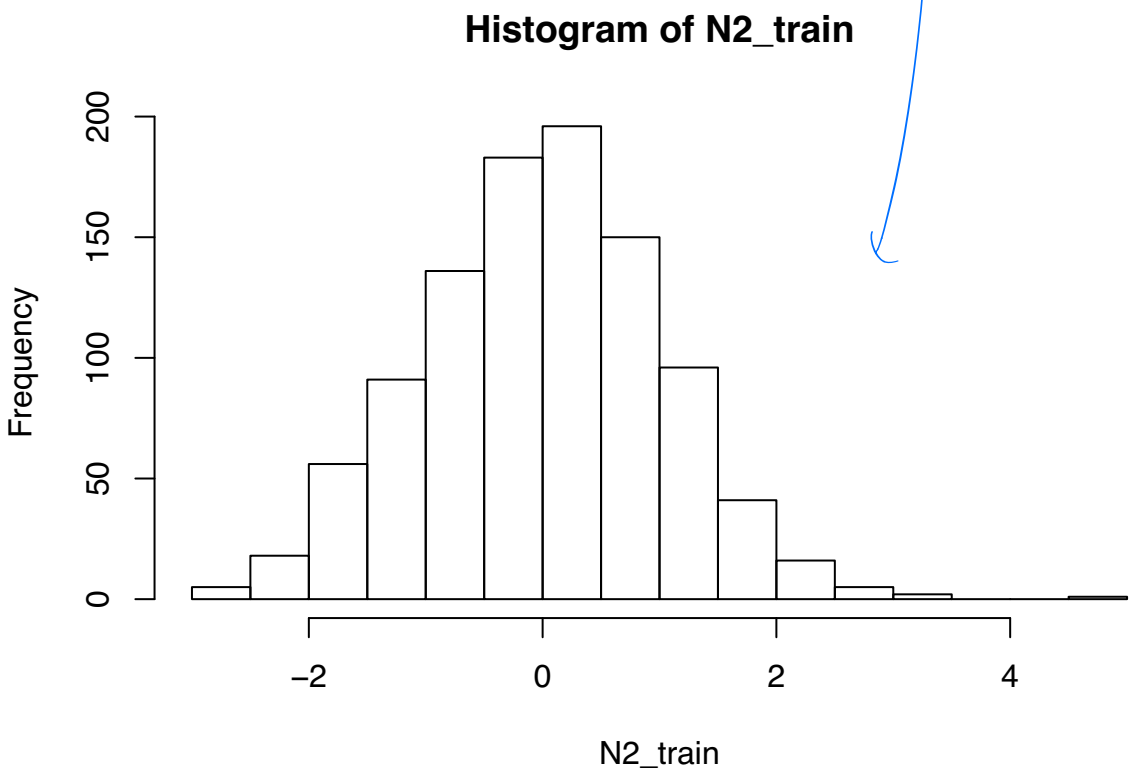

```
qqPlot(N1_train)
```

looks very good



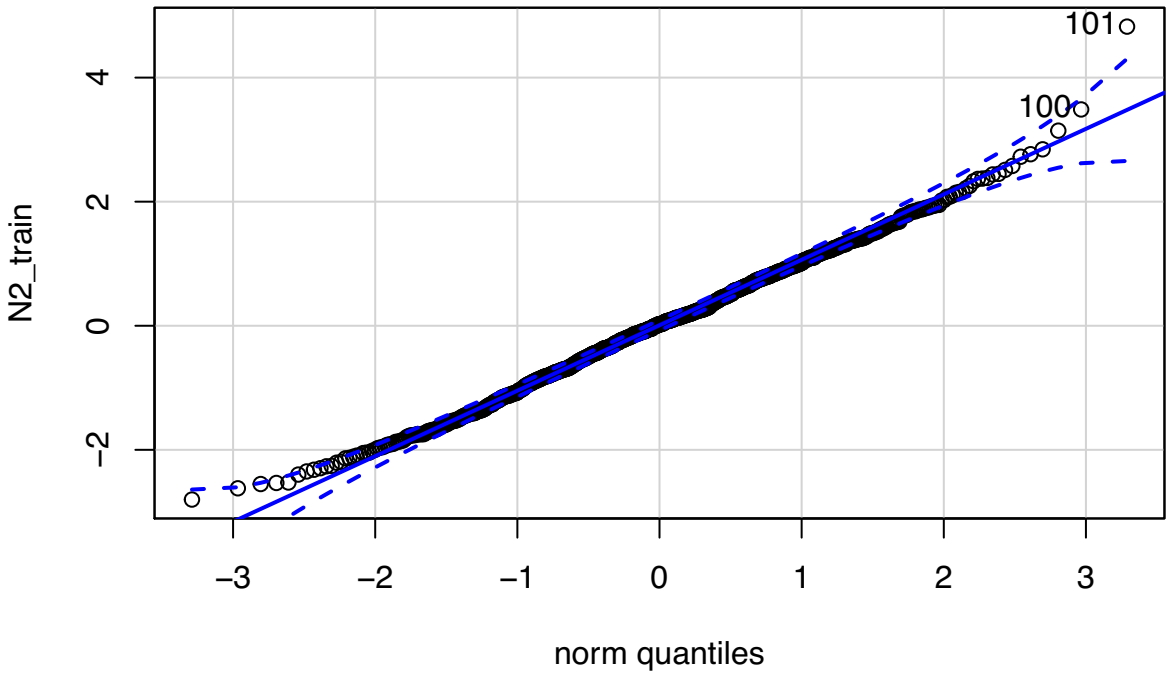

```
## [1] 866 503
```

```
#same for Y2 on train set
N2_train=(Y2[5:1000] - predict(model2_a))/(sqrt(Y2sd_train))
hist(N2_train)
```




```
qqPlot(N2_train)
```

looks very good




```
## [1] 101 100
#they look pretty normal as we wish.


#evaluate 2-d case for the covariance  on train set
M_train=matrix(0,ncol=2,nrow = 996)
```

```r
N1_wocov_train=Y1[5:1000] - predict(model1_a)
N2_wocov_train=Y2[5:1000] - predict(model2_a)
sum(Y1sd_train*Y2sd_train-Y1Y2cov_train^2 > 0) #to check positive definite case
```

```
## [1] 979
```

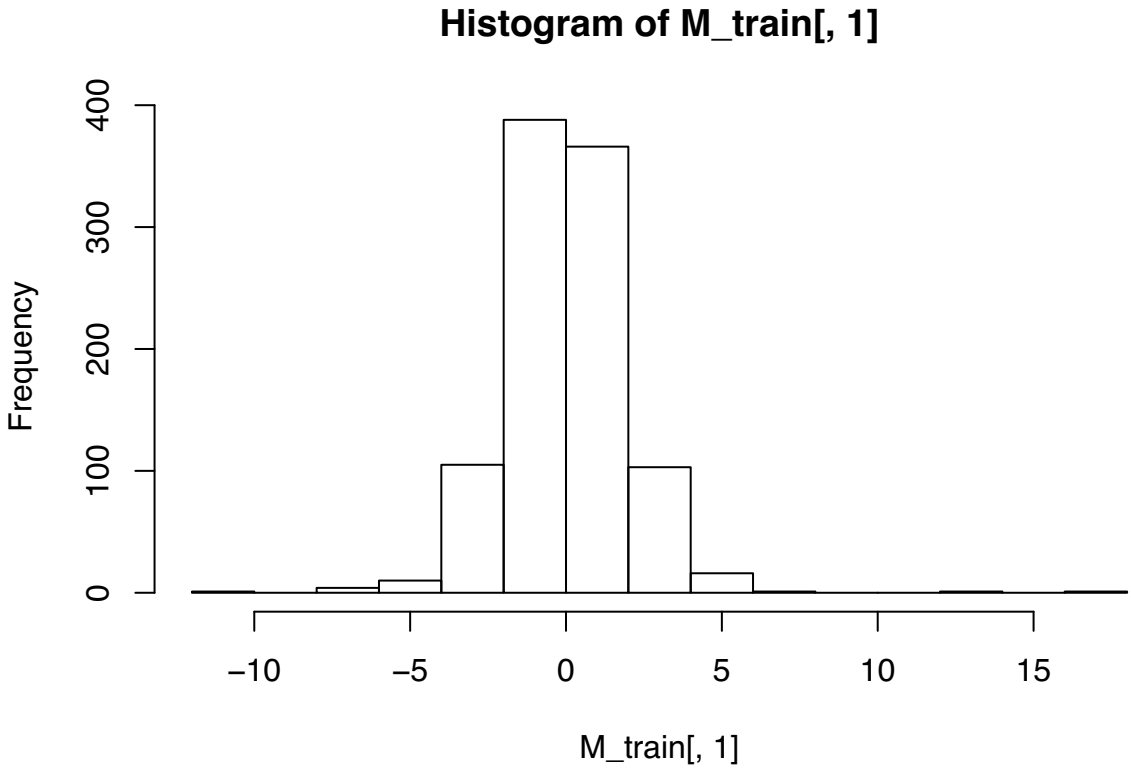
```r
for(i in 1:996){
  if((Y1Y2cov_train[i])/sqrt(Y1sd_train[i]*Y2sd_train[i]) > 1){
    Y1Y2cov_train[i]=0.90*sqrt(Y1sd_train[i]*Y2sd_train[i])
  }
  if((Y1Y2cov_train[i])/sqrt(Y1sd_train[i]*Y2sd_train[i]) < -1){
    Y1Y2cov_train[i]=(-0.90)*sqrt(Y1sd_train[i]*Y2sd_train[i])
  }
  K=matrix(c(Y1sd_train[i], Y1Y2cov_train[i],Y1Y2cov_train[i], Y2sd_train[i]),ncol=2)
  M_train[i,]=solve(sqrtm(K))%*%t(as.matrix(cbind(N1_wocov_train[i],N2_wocov_train[i])))
}
colMeans(M_train) #this looks very good because it is so close to zero.
```

```
## [1] -0.023629716  0.002716292
```

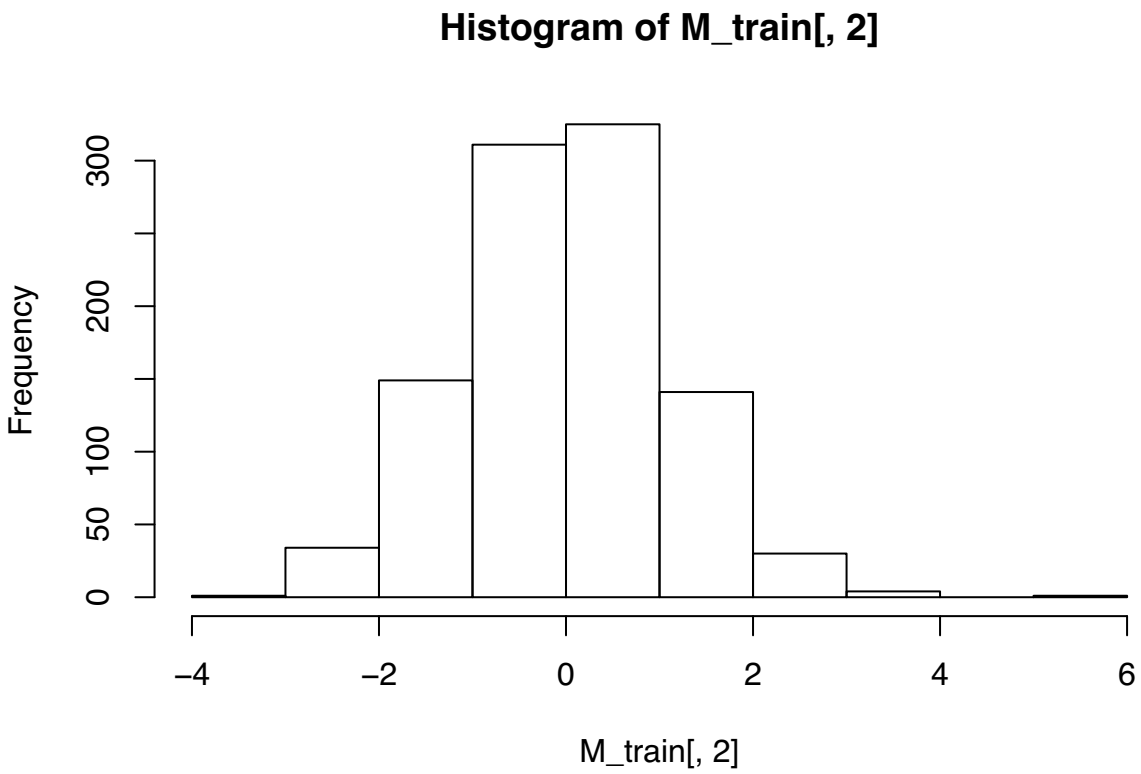
```r
cov(M_train) #with this we finished part c here we see that it doesn't look identity matrix much.
```

```
##           [,1]      [,2]
## [1,]  3.737858 -1.289074
## [2,] -1.289074  1.228550
```

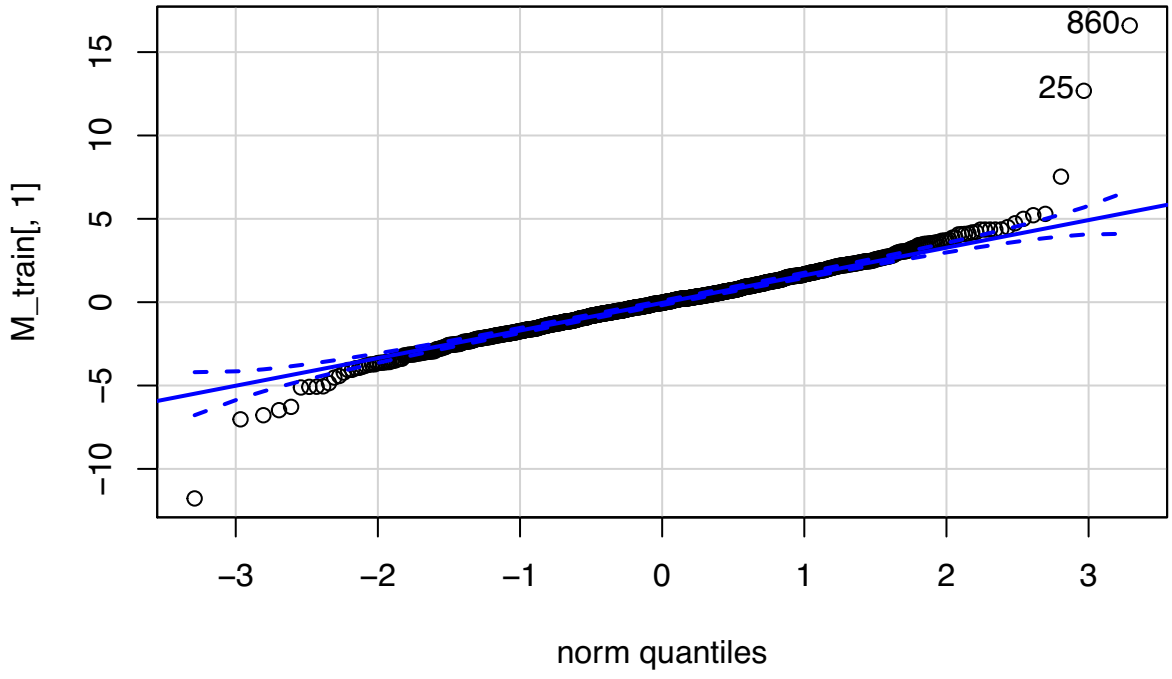
```r
hist(M_train[,1])
```



```
hist(M_train[,2])
```

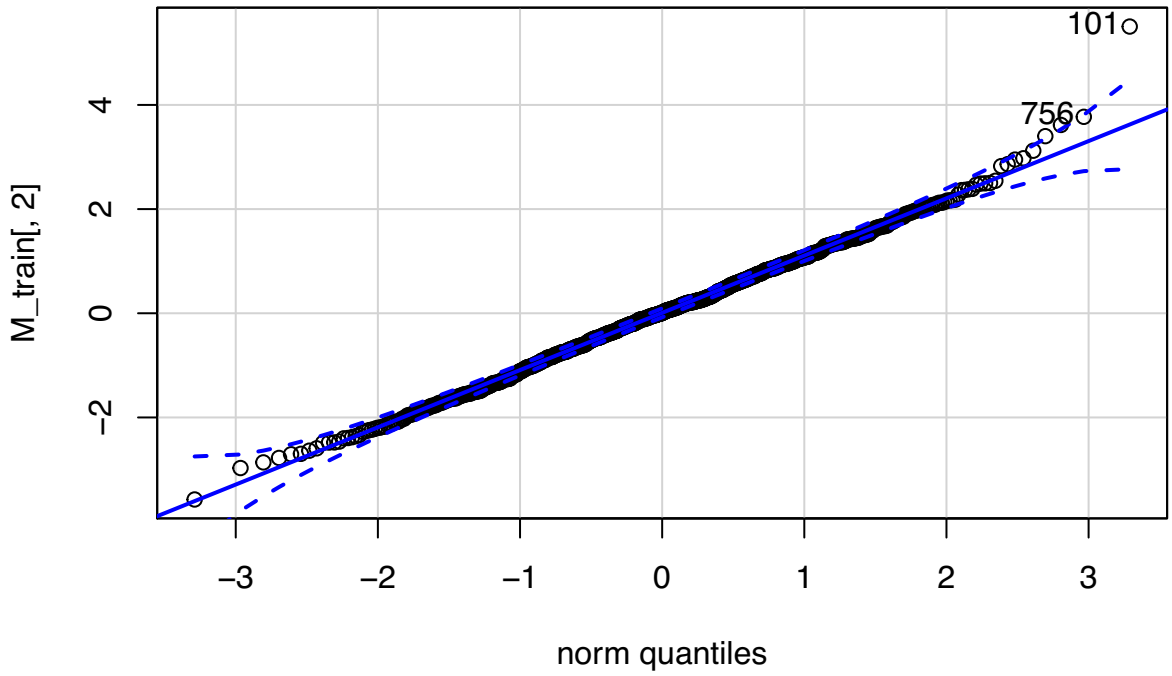



```
qqPlot(M_train[,1]) #some part is slighly off.
```



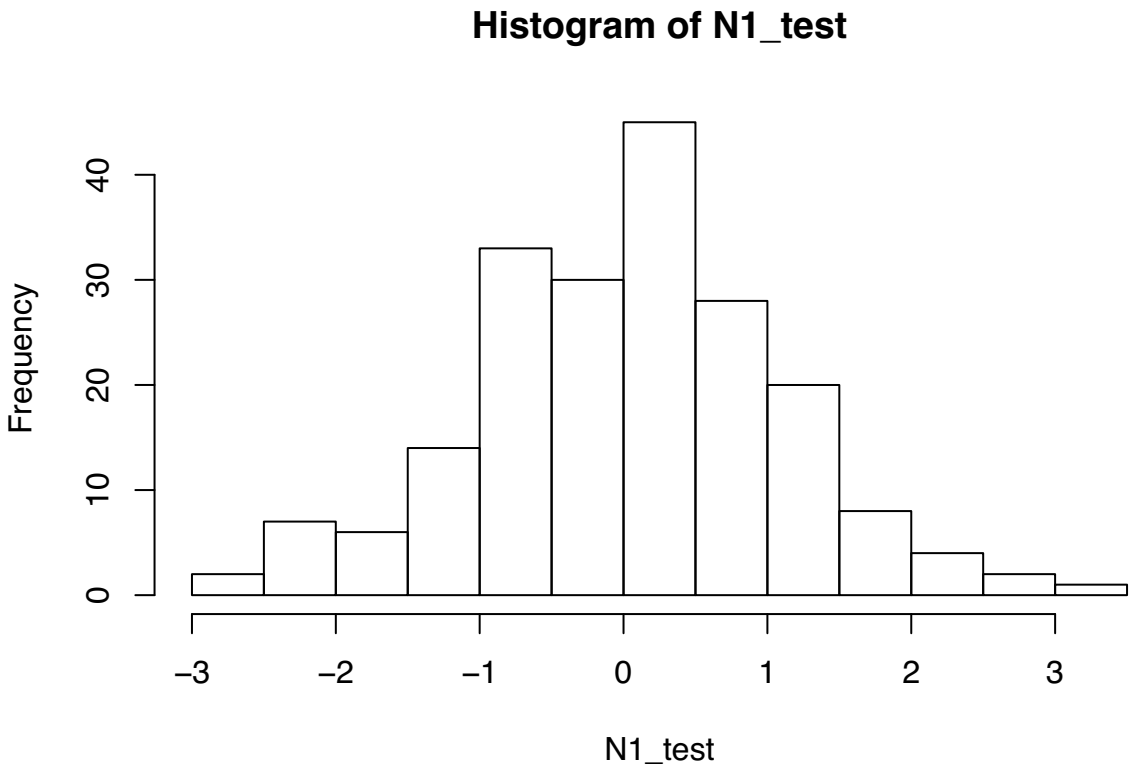

```
## [1] 860  25
```

```
qqPlot(M_train[,2]) #this look pretty good
```

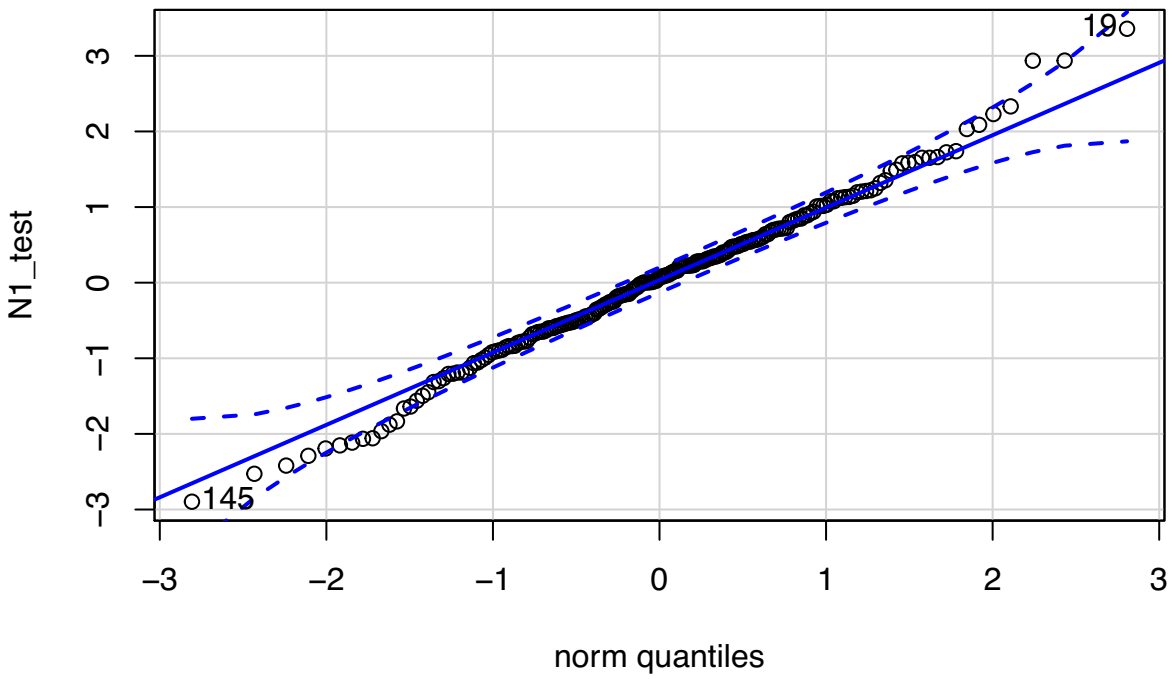



```
## [1] 101 756
```

```
########################################################################## here we finish train set.


########################################################################## Starts  testing on test set.
Y1sd = predict(model1_c,covariates_test)
Y2sd = predict(model2_c,covariates_test)
Y1Y2cov= predict(model12_c,covariates_test)


N1_test=(Y1[1001:1200] - predict(model1_a, covariates_test))/(sqrt(Y1sd))
hist(N1_test) #looks normal that is good.
```

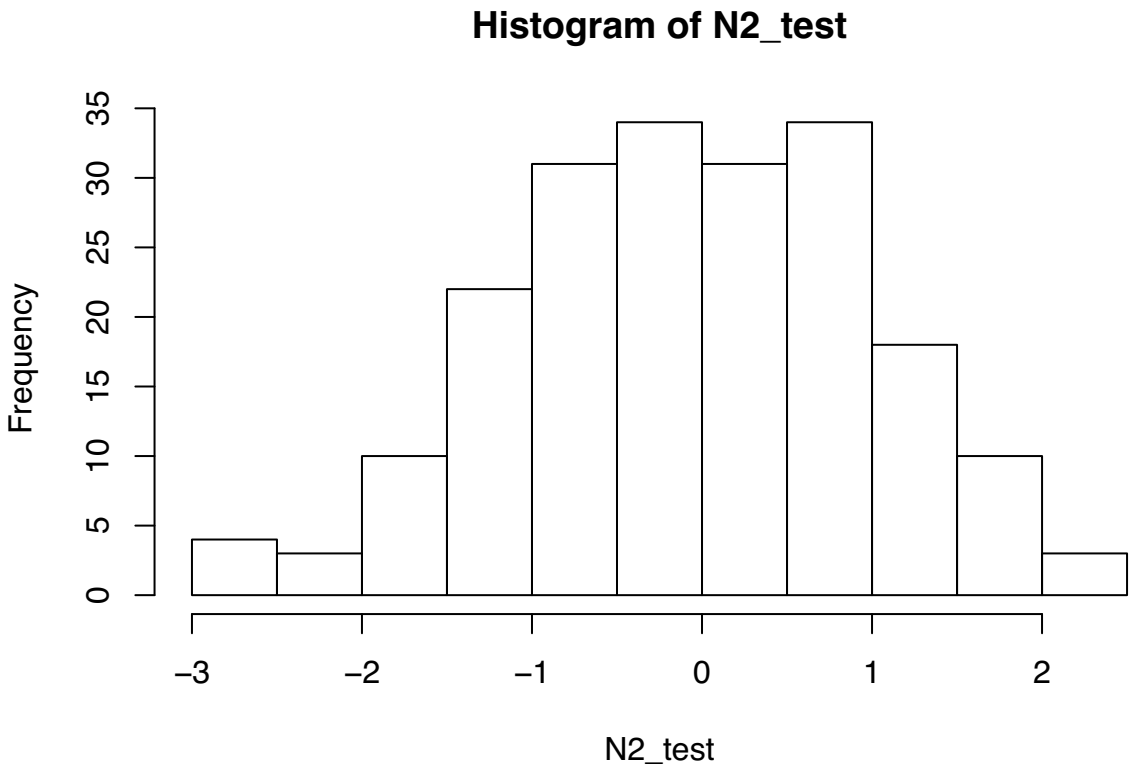



```
qqPlot(N1_test) #this looks  good.
```

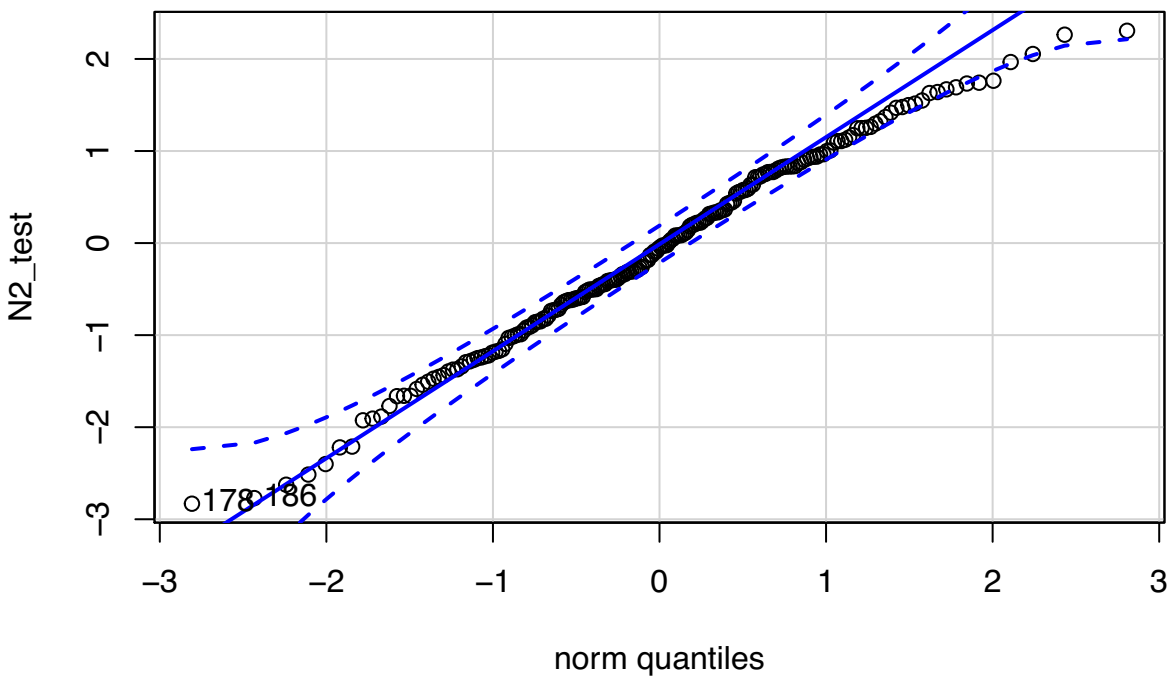



```
## [1]  19 145
```

```
N2_test=(Y2[1001:1200] - predict(model2_a, covariates_test))/(sqrt(Y2sd))
hist(N2_test)
```




```
qqPlot(N2_test) #this looks  good.
```




```
## [1] 178 186
```

```
#evaluate 2-d case for the covariance  on test set
M_test=matrix(0,ncol=2,nrow = 200)
N1_wocov_test=Y1[1001:1200] - predict(model1_a, covariates_test)
N2_wocov_test=Y2[1001:1200] - predict(model2_a, covariates_test)
sum(Y1sd*Y2sd-Y1Y2cov^2 > 0)
```

```
## [1] 198
```

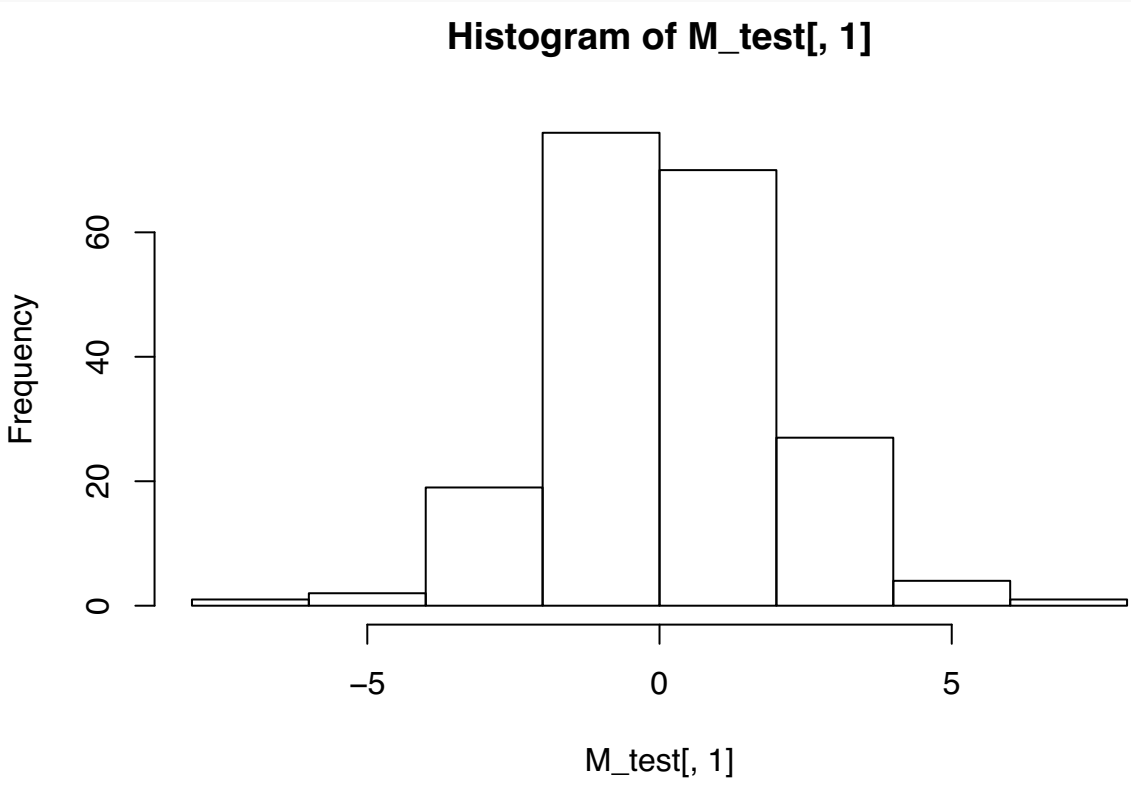
```
library(expm)
for(i in 1:200){
  if((Y1Y2cov[i])/sqrt(Y1sd[i]*Y2sd[i]) > 1){
    Y1Y2cov[i]=0.90*sqrt(Y1sd[i]*Y2sd[i])
  }
  if((Y1Y2cov[i])/sqrt(Y1sd[i]*Y2sd[i]) < -1){
    Y1Y2cov[i]=(-0.90)*sqrt(Y1sd[i]*Y2sd[i])
  }
  K=matrix(c(Y1sd[i], Y1Y2cov[i],Y1Y2cov[i], Y2sd[i]),ncol=2)
  M_test[i,]=solve(sqrtm(K))%*%t(as.matrix(cbind(N1_wocov_test[i],N2_wocov_test[i])))
}
colMeans(M_test) #great, we should expect it to close to zero.
```

```
## [1]  0.19635984 -0.08372921
```

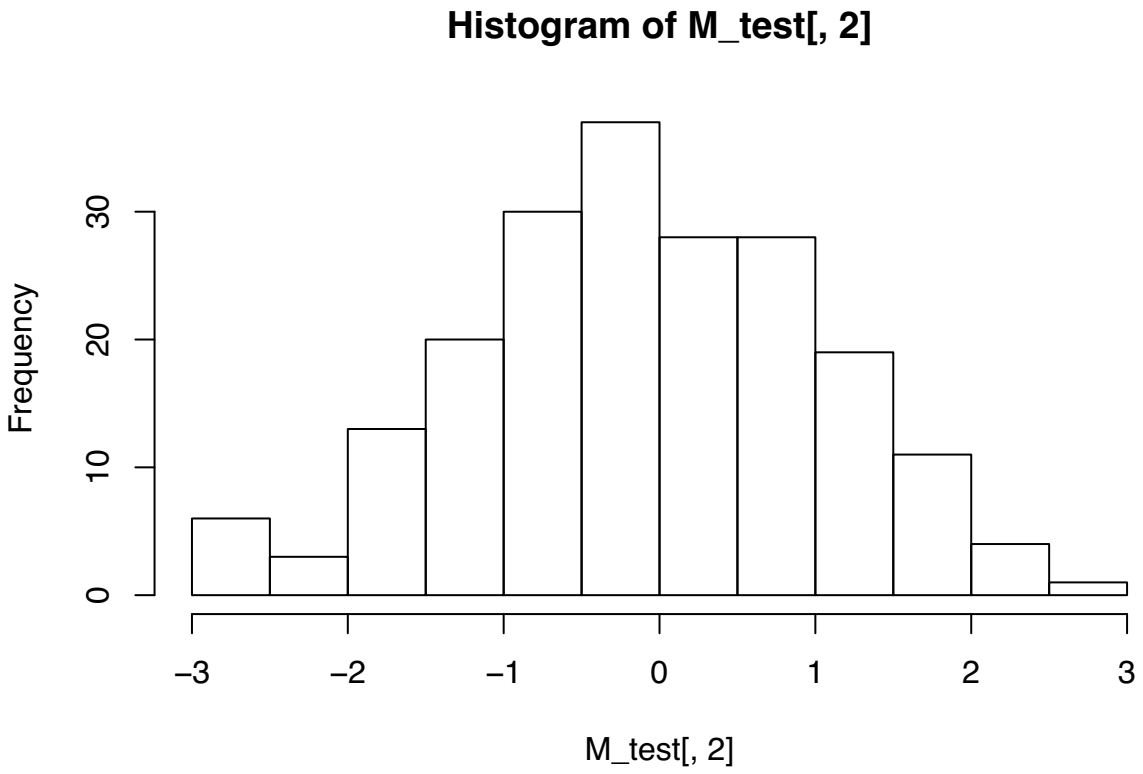
```
cov(M_test) #with this we finished part c here we see that it doesn't look identity matrix much.
```

```
##           [,1]      [,2]
## [1,]  3.474919 -1.233411
## [2,] -1.233411  1.278205
```

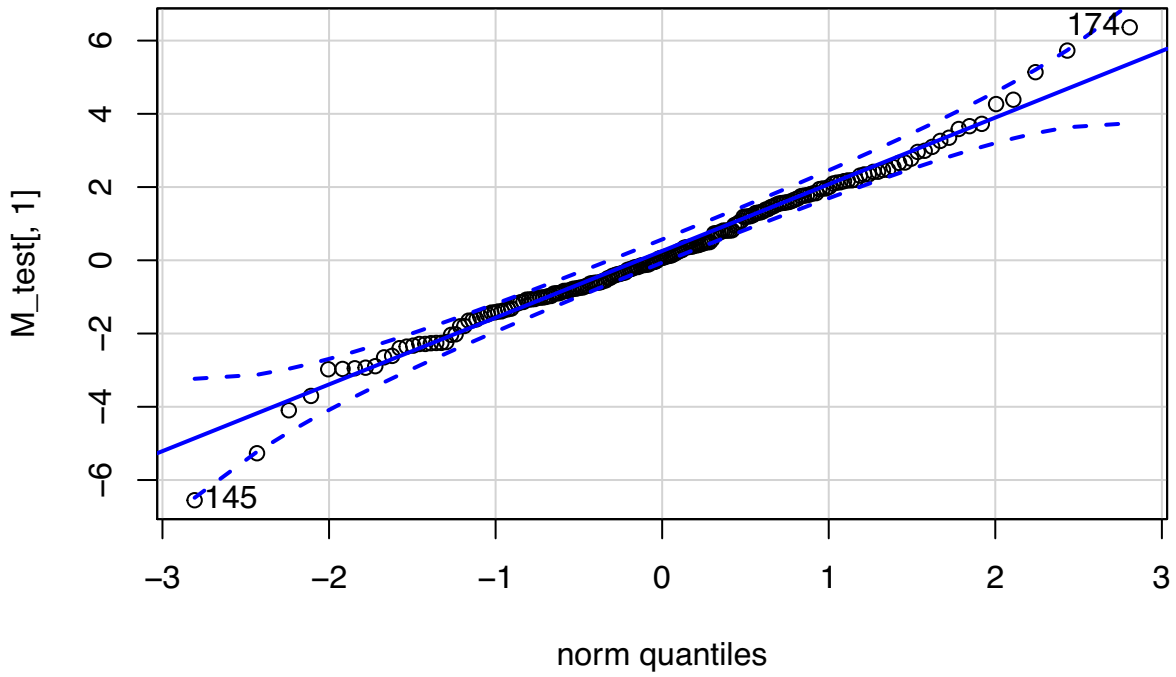
```
hist(M_test[,1]) #kinda normal but mean is not zero
```




```
hist(M_test[,2]) #kinda normal and mean is close to zero.
```

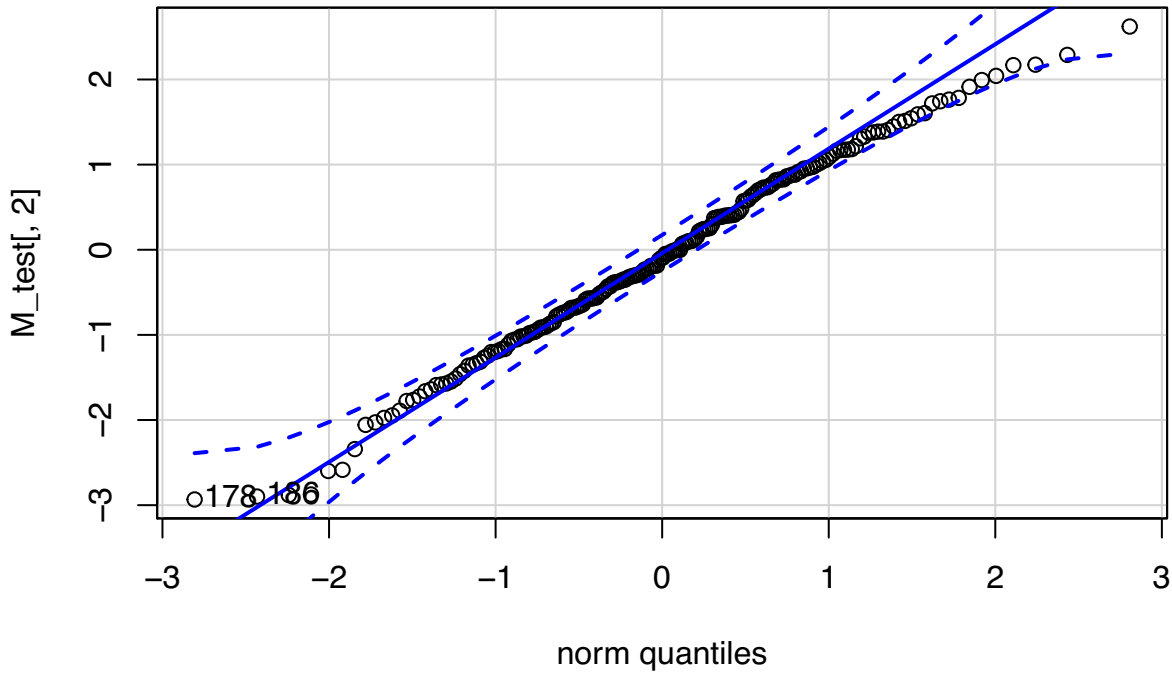



```
qqPlot(M_test[,1])
```




```
## [1] 145 174
```

```
qqPlot(M_test[,2])
```




```
## [1] 178 186
```

```
################## finishint checking for test set
```

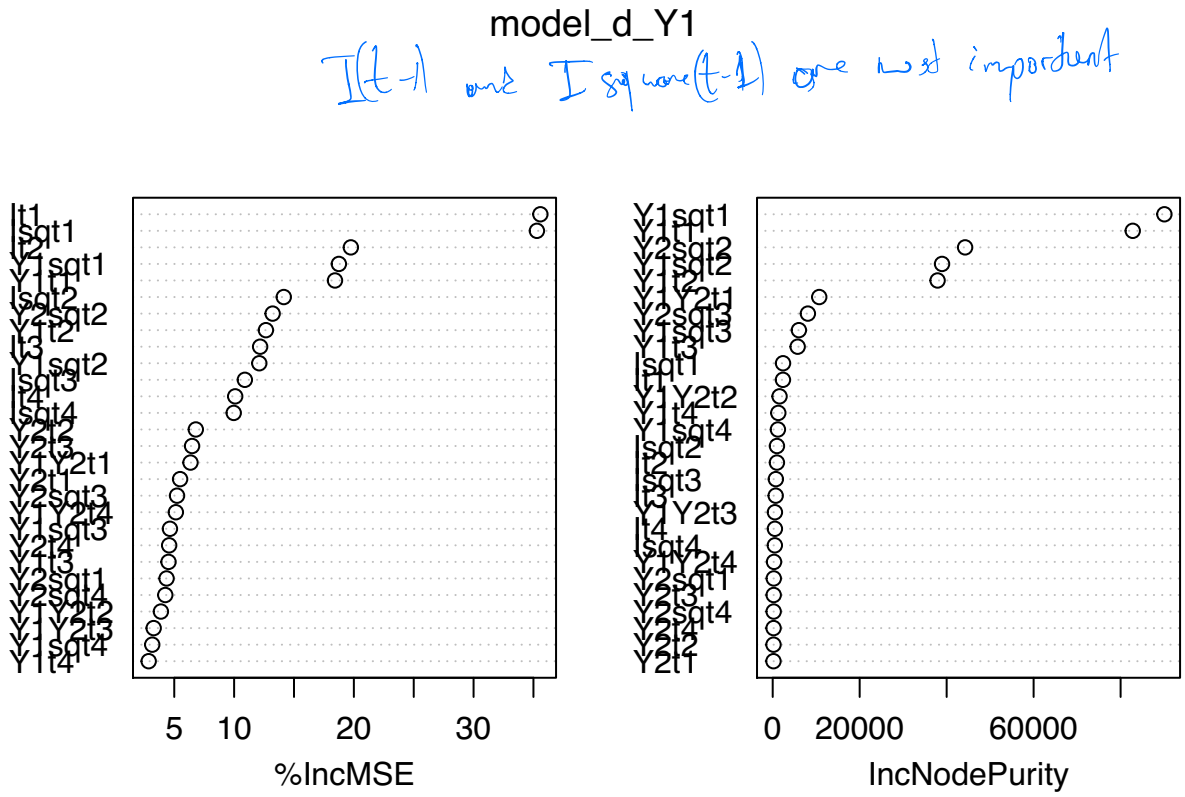
Part c finishes.

```
###### part d
#first we are taking whole set as a training set. So
```

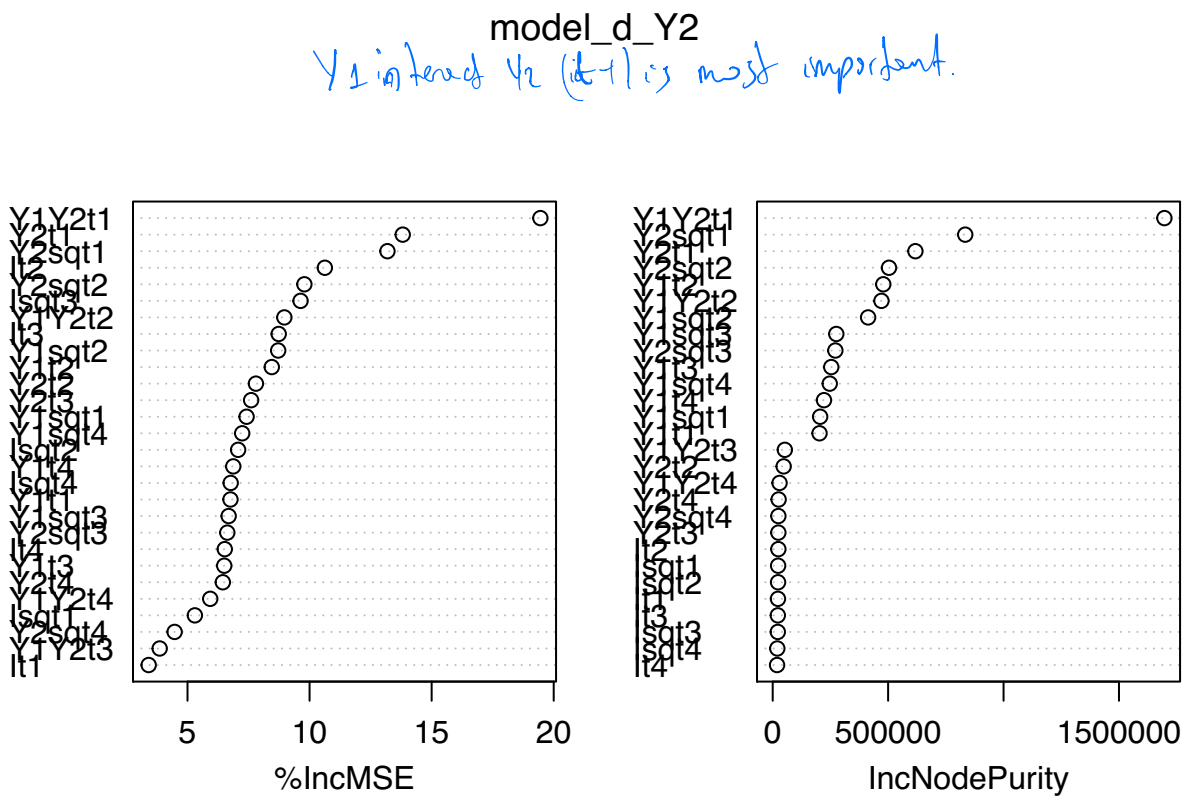
I am defining new covariates.

```
covariates_d=cbind(Y1[1:1196],Y1[2:1197],Y1[3:1198],Y1[4:1199],
                   Y2[1:1196],Y2[2:1197],Y2[3:1198],Y2[4:1199],
                   I[1:1196],I[2:1197],I[3:1198],I[4:1199],
                   Y1square[1:1196],Y1square[2:1197],Y1square[3:1198],Y1square[4:1199],
                   Y2square[1:1196],Y1square[2:1197],Y1square[3:1198],Y2square[4:1199],
                   Isquare[1:1196],Isquare[2:1197],Isquare[3:1198],Isquare[4:1199],
                   Y1interactY2[1:1196],Y1interactY2[2:1197],Y1interactY2[3:1198],Y1interactY2[4:1199])
# Y1interactI[1:1196],Y1interactI[2:1197],Y1interactI[3:1198],Y1interactI[4:1199],
#Y2interactI[1:1196],Y2interactI[2:1197],Y2interactI[3:1198],Y2interactI[4:1199])
#we are giving column names here. Note Y1t4 means Y1(t-4) and Y1t3=Y1(t-3) etc. The rest is
# similar idea. It will be easy to read the important variable when we use varImpPlot function.
colnames(covariates_d)=c("Y1t4","Y1t3","Y1t2","Y1t1",
                         "Y2t4","Y2t3","Y2t2","Y2t1",
                         "It4","It3","It2","It1",
                         "Y1sqt4","Y1sqt3","Y1sqt2","Y1sqt1",
                         "Y2sqt4","Y2sqt3","Y2sqt2","Y2sqt1",
                         "Isqt4","Isqt3","Isqt2","Isqt1",
                         "Y1Y2t4","Y1Y2t3","Y1Y2t2","Y1Y2t1")
# "Y1It4","Y1It3","Y1It2","Y1It1",
#"Y2It4","Y2It3","Y2It2","Y2It1")
# covariates_d_try=cbind(Y1[1:1194],Y1[2:1195],Y1[3:1196],Y1[4:1197],Y1[5:1198],
#                        Y2[1:1194],Y2[2:1195],Y2[3:1196],Y2[4:1197],Y2[5:1196],
#                        I[1:1194],I[2:1195],I[3:1196],I[4:1197],I[5:1198])
#
# colnames(covariates_d_try)=c("Y1t5","Y1t4","Y1t3","Y1t2","Y1t1","Y2t5","Y2t4","Y2t3","Y2t2","Y2t1",
```

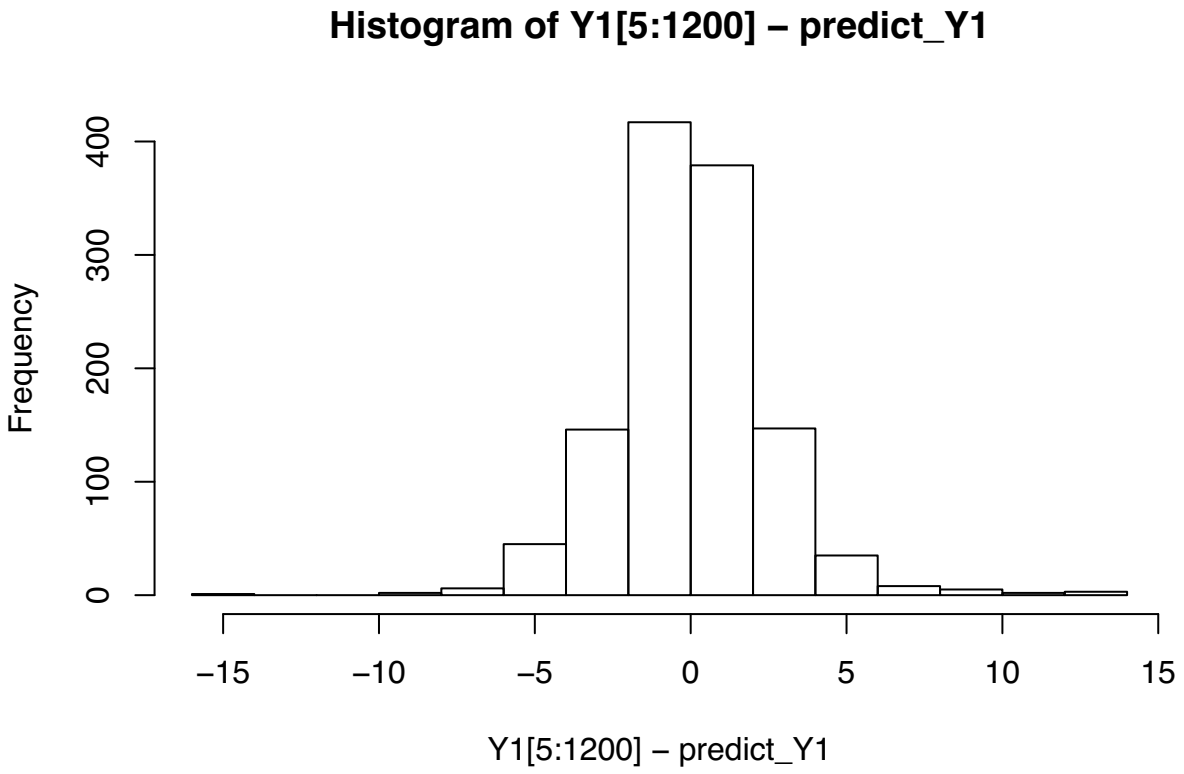
```
#                                  "It5","It4","It3","It2","It1")
# covariates_d_try_test=cbind(Y1[1195],Y1[1196],Y1[1197],Y1[1198],Y1[1199],
#                             Y2[1195],Y2[1196],Y2[1197],Y2[1198],Y2[1199],
#                            I[1195],I[1196],I[1197],I[1198],I[1199])
covariates_d_test=cbind(Y1[1197],Y1[1198],Y1[1199],Y1[1200],
                        Y2[1197],Y2[1198],Y2[1199],Y2[1200],
                        I[1197],I[1198],I[1199],I[1200],
                        Y1square[1197],Y1square[1198],Y1square[1199],Y1square[1200],
                        Y2square[1197],Y2square[1198],Y2square[1199],Y2square[1200],
                        Isquare[1197],Isquare[1198],Isquare[1199],Isquare[1200],
                        Y1interactY2[1197],Y1interactY2[1198],Y1interactY2[1199],Y1interactY2[1200])
#Y1interactI[1197],Y1interactI[1198],Y1interactI[1199],Y1interactI[1200],
#Y2interactI[1197],Y2interactI[1198],Y2interactI[1199],Y2interactI[1200])
colnames(covariates_d_test)=c("Y1t4","Y1t3","Y1t2","Y1t1",
                              "Y2t4","Y2t3","Y2t2","Y2t1",
                              "It4","It3","It2","It1",
                              "Y1sqt4","Y1sqt3","Y1sqt2","Y1sqt1",
                              "Y2sqt4","Y2sqt3","Y2sqt2","Y2sqt1",
                              "Isqt4","Isqt3","Isqt2","Isqt1",
                              "Y1Y2t4","Y1Y2t3","Y1Y2t2","Y1Y2t1")
#"Y1It4","Y1It3","Y1It2","Y1It1",
#"Y2It4","Y2It3","Y2It2","Y2It1")

##########################################we creat the random forests for Y1 and Y2
model_d_Y1=randomForest(Y1[5:1200]~ ., data = covariates_d,  importance=TRUE)
varImpPlot(model_d_Y1)
```



```
model_d_Y2=randomForest(Y2[5:1200]~ ., data = covariates_d,  importance=TRUE)
varImpPlot(model_d_Y2)
```



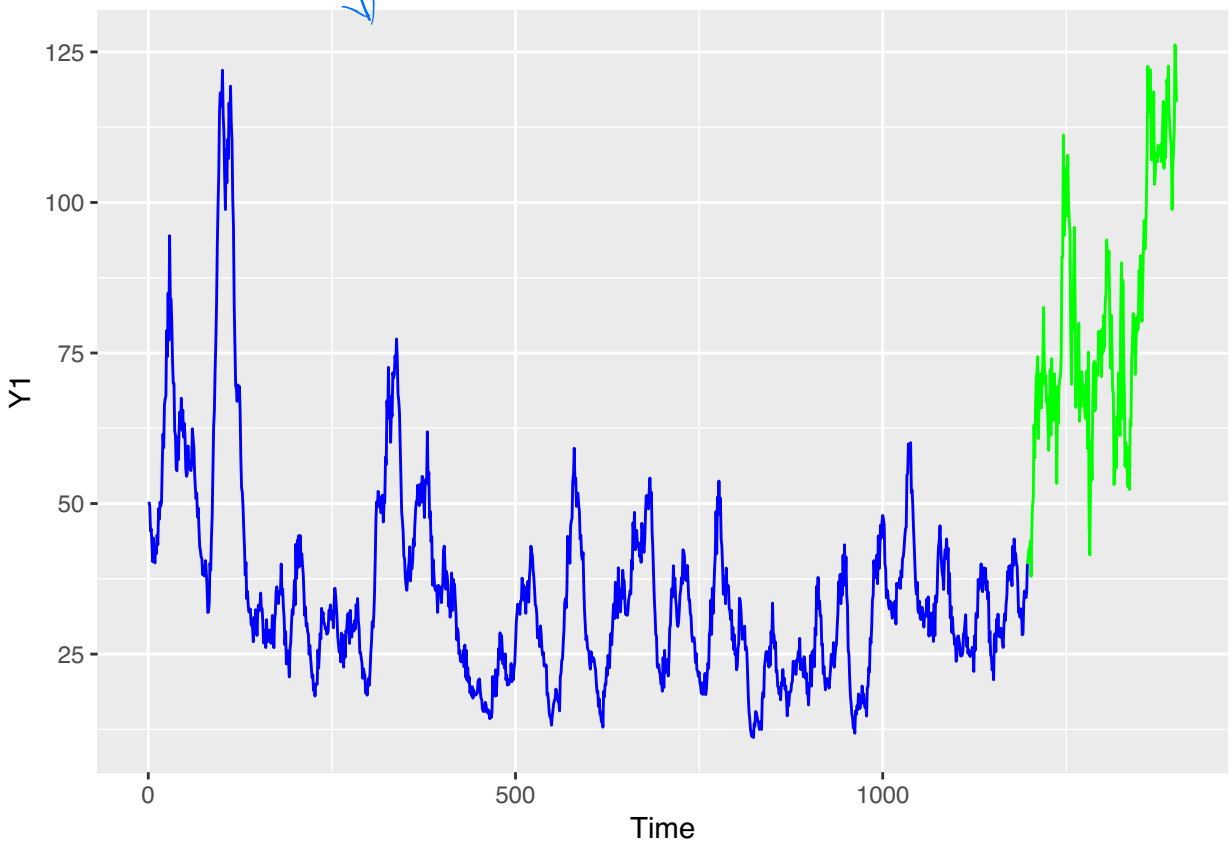

```
## as in part c

V1_d=(Y1[5:1200]-predict(model_d_Y1))^2 #variance for Y1
V2_d=(Y2[5:1200]-predict(model_d_Y2))^2 #variance for Y2
Cov12_d=sqrt(V1_d*V2_d) #covariance for Y1 and Y2

##########################################we creat the random forests for variences on whole set

model_d_V1=randomForest(V1_d~ ., data = covariates_d,  importance=TRUE)
model_d_V2=randomForest(V2_d~ ., data = covariates_d,  importance=TRUE)
model_d_V12=randomForest(Cov12_d~ ., data = covariates_d,  importance=TRUE)


#Predict on train set and test set.
predict_Y1=predict(model_d_Y1)
hist(Y1[5:1200]-predict_Y1)#looks pretty normal with mean zero.
```



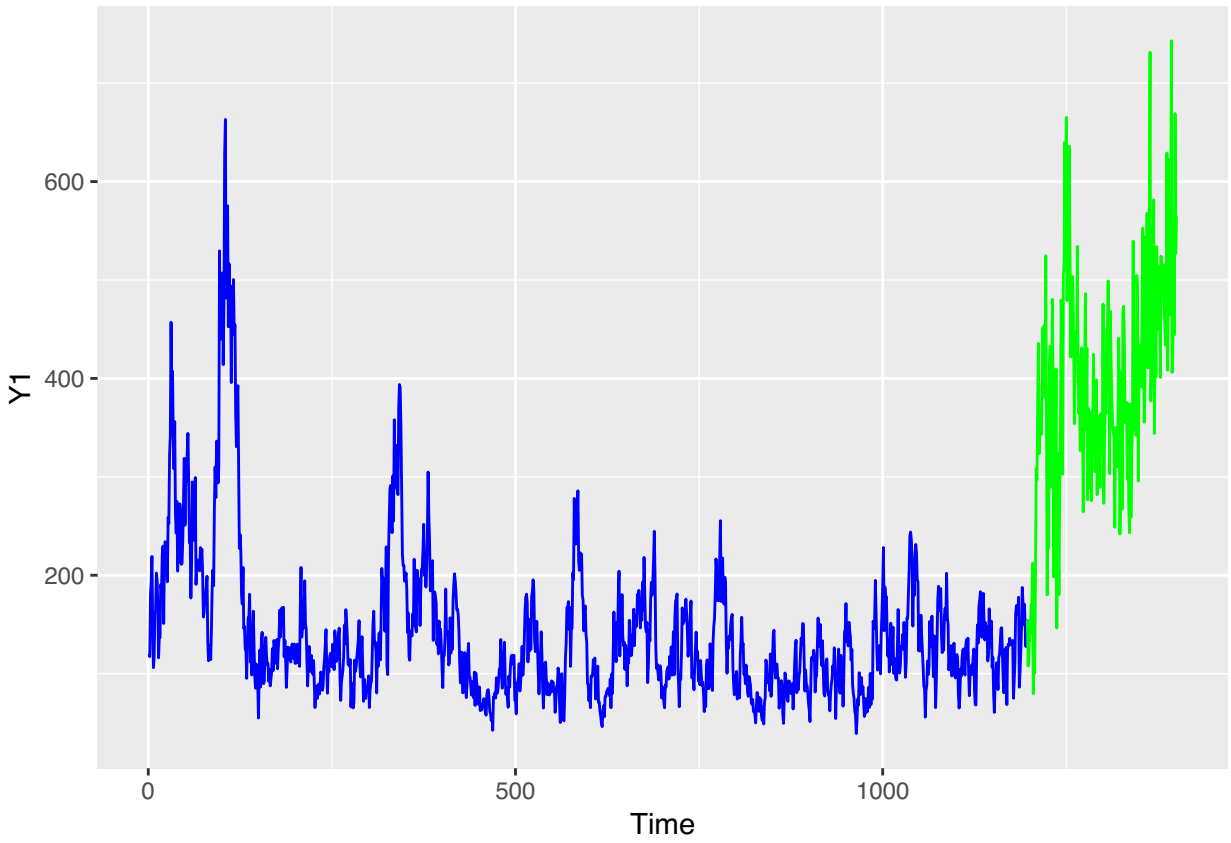

```
predict_Y1_test=predict(model_d_Y1,covariates_d_test)


############################################## Now first we do one future scenario

Ilast4=I[1197:1200]

Inew=c(Ilast4,Ifuture)
Isqrnew=Inew^2

Y1last4=Y1[1197:1200]
Y1sqrlast4=Y1square[1197:1200]

Y2last4=Y2[1197:1200]
Y2sqrlast4=Y2square[1197:1200]

Y1Y2last4=Y1interactY2[1197:1200]


Y1last204=rep(0,204)
Y2last204=rep(0,204)
for(i in 1:200){
  k=4+i
  currentposition=matrix(c(Y1last4[(k-4):(k-1)], Y2last4[(k-4):(k-1)],  Inew[(k-4):(k-1)],
                           Y1sqrlast4[(k-4):(k-1)],  Y2sqrlast4[(k-4):(k-1)],  Isqrnew[(k-4):(k-1)],
                           Y1Y2last4[(k-4):(k-1)]),
                         nrow=1)

  colnames(currentposition)=c("Y1t4","Y1t3","Y1t2","Y1t1",
```

```
                                "Y2t4","Y2t3","Y2t2","Y2t1",
                                "It4","It3","It2","It1",
                                "Y1sqt4","Y1sqt3","Y1sqt2","Y1sqt1",
                                "Y2sqt4","Y2sqt3","Y2sqt2","Y2sqt1",
                                "Isqt4","Isqt3","Isqt2","Isqt1",
                                "Y1Y2t4","Y1Y2t3","Y1Y2t2","Y1Y2t1")
  prediction_Y=cbind(predict(model_d_Y1,currentposition), predict(model_d_Y2,currentposition))

  Sigma11=predict(model_d_V1,currentposition)
  Sigma22=predict(model_d_V2,currentposition)
  Sigma12=predict(model_d_V12,currentposition)

  if((Sigma12)/sqrt(Sigma11*Sigma22) > 1){
    Sigma12=0.9*sqrt(Sigma11*Sigma22)
  }
  if((Sigma12)/sqrt(Sigma11*Sigma22) < -1){
    Sigma12=(-0.9)*sqrt(Sigma11*Sigma22)
  }
  K_d=matrix(c(Sigma11, Sigma12,Sigma12, Sigma22),ncol=2)

  newpoint=t(prediction_Y)+sqrtm(K_d)%*%rnorm(2,mean=0,sd=1)

  Y1last4=c(Y1last4,newpoint[1])
  Y2last4=c(Y2last4,newpoint[2])
  Y1sqrlast4=c(Y1sqrlast4,log(newpoint[1]^2))
  Y2sqrlast4=c(Y2sqrlast4,log(newpoint[2]^2))
  Y1Y2last4=c(Y1Y2last4,log(newpoint[1]*newpoint[2]))
}

plot1_d_one=ggplot(data=NULL,aes(Time, Y1))+
  geom_line(aes(x=c(1:1197), y=Y1[c(1:1197)]), color="blue")+
  geom_line(aes(x=c(1197:1400), y=Y1last4), color="green")
print(plot1_d_one)
```

This seem slightly off. I believe this comes because I added $Y1^2$, $Y2^2$ and $Y1 \cdot Y2$. So I didn't like these simulations results. So I added my old ones. Those simulation are without them (i.e. without interactions) and 5 steps back. It gives me better MSE's and better predictions. I only kept this because I didn't have enough time to go back and to change everything.

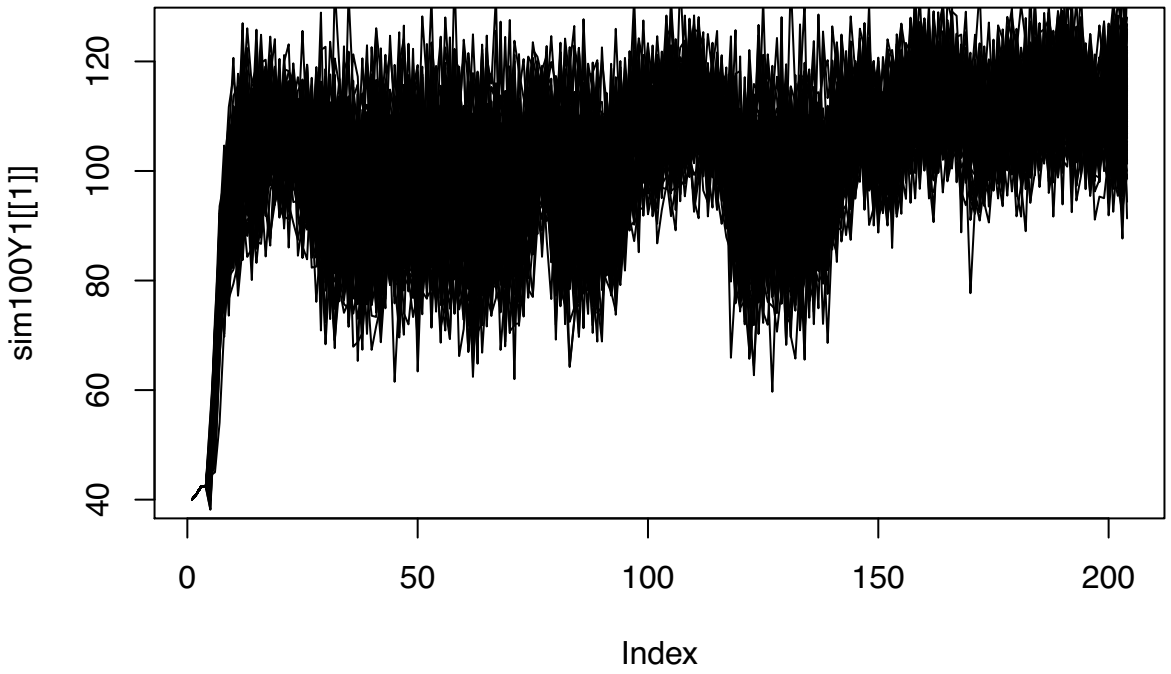



```
plot2_d_one=ggplot(data=NULL,aes(Time, Y1))+
  geom_line(aes(x=c(1:1197), y=Y2[c(1:1197)]), color="blue")+
  geom_line(aes(x=c(1197:1400), y=Y2last4), color="green")
print(plot2_d_one)
```

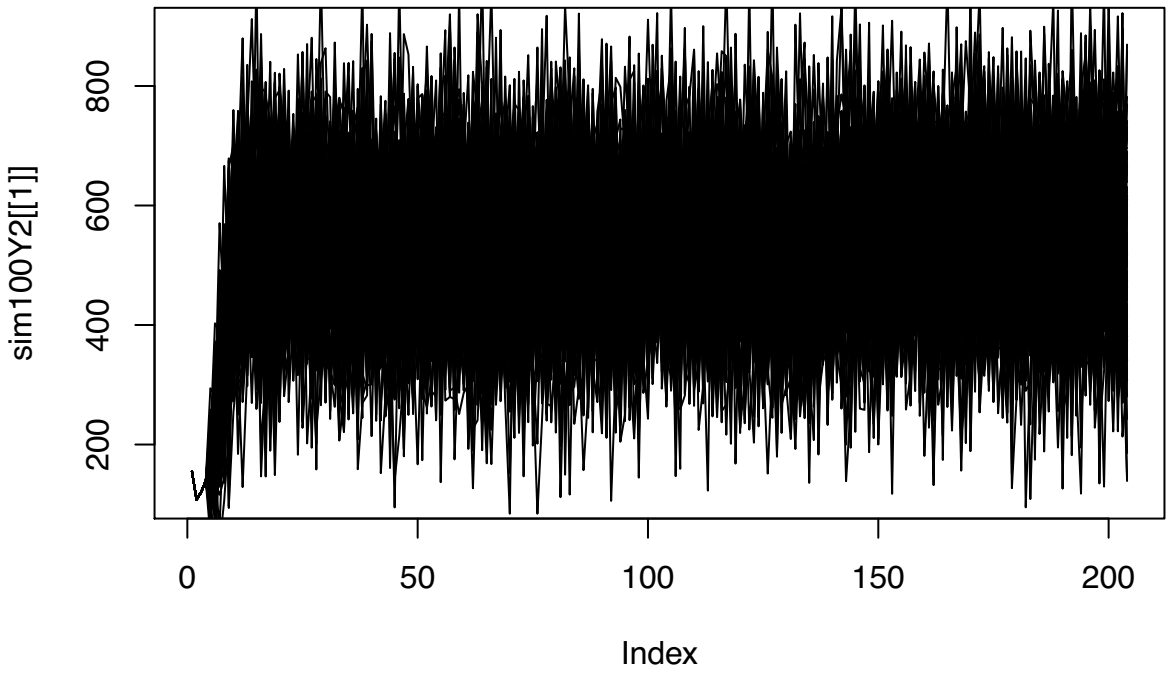



```
############################################### finishing one scenario..

############################################### Now lets do 100 scenario
sim100Y1=list()
sim100Y2=list()
for(m in 1:100){
  Ilast4=I[1197:1200]

  Inew=c(Ilast4,Ifuture)
  Isqrnew=Inew^2

  Y1last4=Y1[1197:1200]
  Y1sqrlast4=Y1square[1197:1200]

  Y2last4=Y2[1197:1200]
  Y2sqrlast4=Y2square[1197:1200]

  Y1Y2last4=Y1interactY2[1197:1200]
  for(i in 1:200){
    k=4+i
    currentposition=matrix(c(Y1last4[(k-4):(k-1)], Y2last4[(k-4):(k-1)],  Inew[(k-4):(k-1)],
                             Y1sqrlast4[(k-4):(k-1)],  Y2sqrlast4[(k-4):(k-1)],  Isqrnew[(k-4):(k-1)],
                             Y1Y2last4[(k-4):(k-1)]),
                           nrow=1)

    colnames(currentposition)=c("Y1t4","Y1t3","Y1t2","Y1t1",
                                "Y2t4","Y2t3","Y2t2","Y2t1",
```

```
                                  "It4","It3","It2","It1",
                                  "Y1sqt4","Y1sqt3","Y1sqt2","Y1sqt1",
                                  "Y2sqt4","Y2sqt3","Y2sqt2","Y2sqt1",
                                  "Isqt4","Isqt3","Isqt2","Isqt1",
                                  "Y1Y2t4","Y1Y2t3","Y1Y2t2","Y1Y2t1")
    prediction_Y=cbind(predict(model_d_Y1,currentposition), predict(model_d_Y2,currentposition))

    Sigma11=predict(model_d_V1,currentposition)
    Sigma22=predict(model_d_V2,currentposition)
    Sigma12=predict(model_d_V12,currentposition)

    if((Sigma12)/sqrt(Sigma11*Sigma22) > 1){
      Sigma12=0.9*sqrt(Sigma11*Sigma22)
    }
    if((Sigma12)/sqrt(Sigma11*Sigma22) < -1){
      Sigma12=(-0.9)*sqrt(Sigma11*Sigma22)
    }
    K_d=matrix(c(Sigma11, Sigma12,Sigma12, Sigma22),ncol=2)

    newpoint=t(prediction_Y)+sqrtm(K_d)%*%rnorm(2,mean=0,sd=1)

    Y1last4=c(Y1last4,newpoint[1])
    Y2last4=c(Y2last4,newpoint[2])
    Y1sqrlast4=c(Y1sqrlast4,newpoint[1])
    Y2sqrlast4=c(Y2sqrlast4,newpoint[2])
    Y1Y2last4=c(Y1Y2last4,log(newpoint[1]*newpoint[2]))
  }

  Y1last204=Y1last4
  Y2last204=Y2last4
  sim100Y1[[m]]=Y1last204
  sim100Y2[[m]]=Y2last204
}
plot(sim100Y1[[1]],type="l")
for(i in 2:100){
  lines(sim100Y1[[i]])
}
```




```
plot(sim100Y2[[1]],type="l")
for(i in 2:100){
  lines(sim100Y2[[i]])
}
```



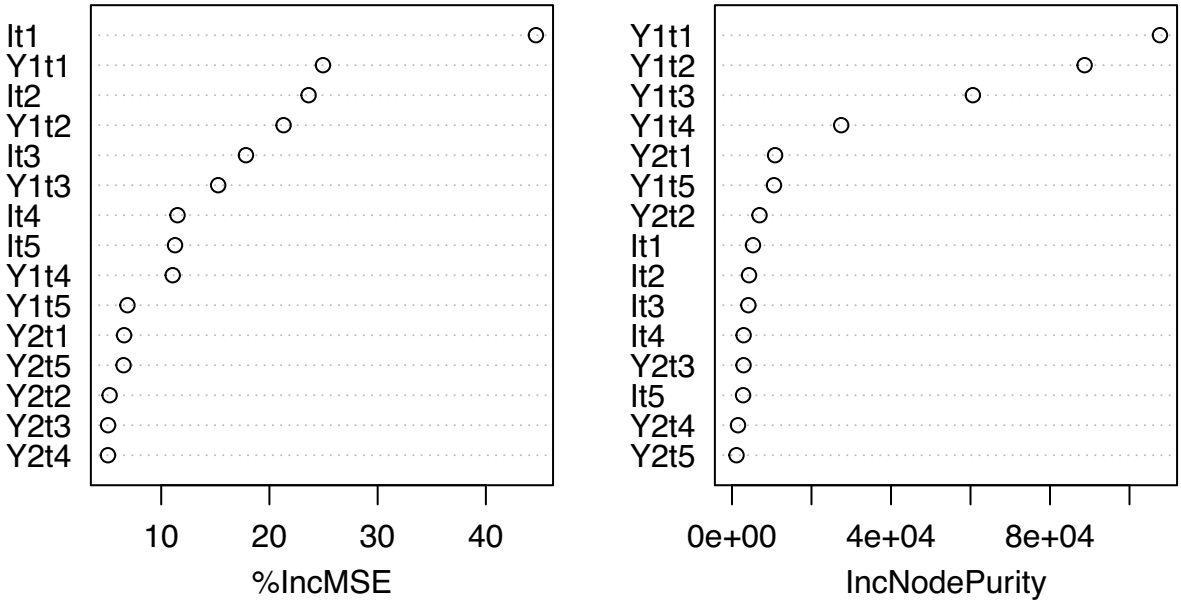
```
################################################################## finishing 100 scenario with simulations

### I get a little bit bad similation results with these covarities.

######################### ######################### ######################### ########################
# But I told your earlier I did 5 steps back with different covarities so I am adding
```

But these give me better results.

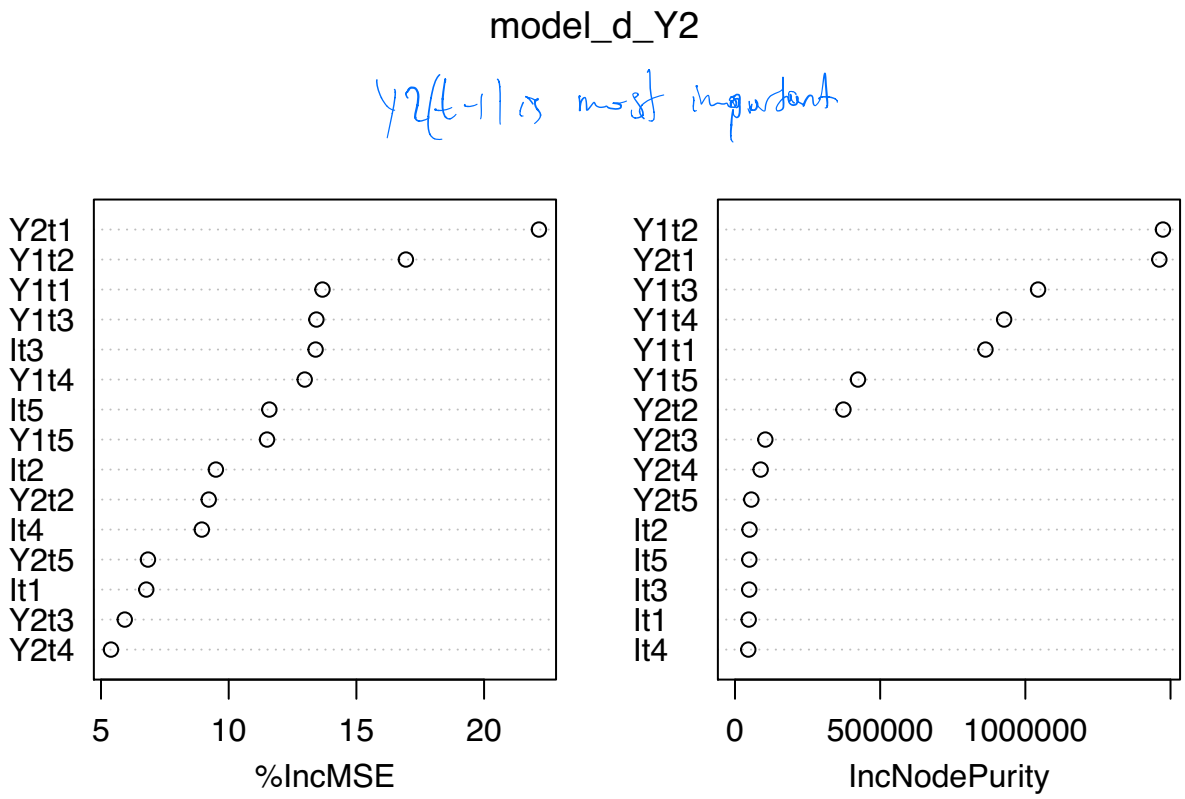
```
#here that one as well. but that is basically same thing but without interactions and 5 steps back.
#########################################################################################################

###### part d alternative simulations
#first we are taking whole set as a training set. So
# covariates_d=cbind(Y1[1:1196],Y1[2:1197],Y1[3:1198],Y1[4:1199],
#                    Y2[1:1196],Y2[2:1197],Y2[3:1198],Y2[4:1199],
#                     I[1:1196],I[2:1197],I[3:1198],I[4:1199],
#                  Y1inc[1:1196],Y1inc[2:1197], Y1inc[3:1198],
#                  Y2inc[1:1196],Y2inc[2:1197], Y2inc[3:1198])
# Here I am trying 5 steps back instead of 4  in order to get better prediction.
covariates_d_try=cbind(Y1[1:1194],Y1[2:1195],Y1[3:1196],Y1[4:1197],Y1[5:1198],
                       Y2[1:1194],Y2[2:1195],Y2[3:1196],Y2[4:1197],Y2[5:1198],
                       I[1:1194],I[2:1195],I[3:1196],I[4:1197],I[5:1198])
# colnames(covariates_d)=c("Y1t4","Y1t3","Y1t2","Y1t1", "Y2t4","Y2t3","Y2t2","Y2t1",
#                          "It4","It3","It2","It1", "Y1inct3","Y1inct2","Y1inct1",
#                          "Y2inct3","Y2inct2","Y2inct1")
colnames(covariates_d_try)=c("Y1t5","Y1t4","Y1t3","Y1t2","Y1t1","Y2t5","Y2t4","Y2t3","Y2t2","Y2t1",
                             "It5","It4","It3","It2","It1")
covariates_d_try_test=cbind(Y1[1195],Y1[1196],Y1[1197],Y1[1198],Y1[1199],
                            Y2[1195],Y2[1196],Y2[1197],Y2[1198],Y2[1199],
                            I[1195],I[1196],I[1197],I[1198],I[1199])
colnames(covariates_d_try_test)=c("Y1t5","Y1t4","Y1t3","Y1t2","Y1t1","Y2t5","Y2t4","Y2t3","Y2t2","Y2t1"
                                  "It5","It4","It3","It2","It1")
model_d_Y1=randomForest(Y1[6:1199]~ ., data = covariates_d_try,  importance=TRUE)
varImpPlot(model_d_Y1)
```

model_d_Y1

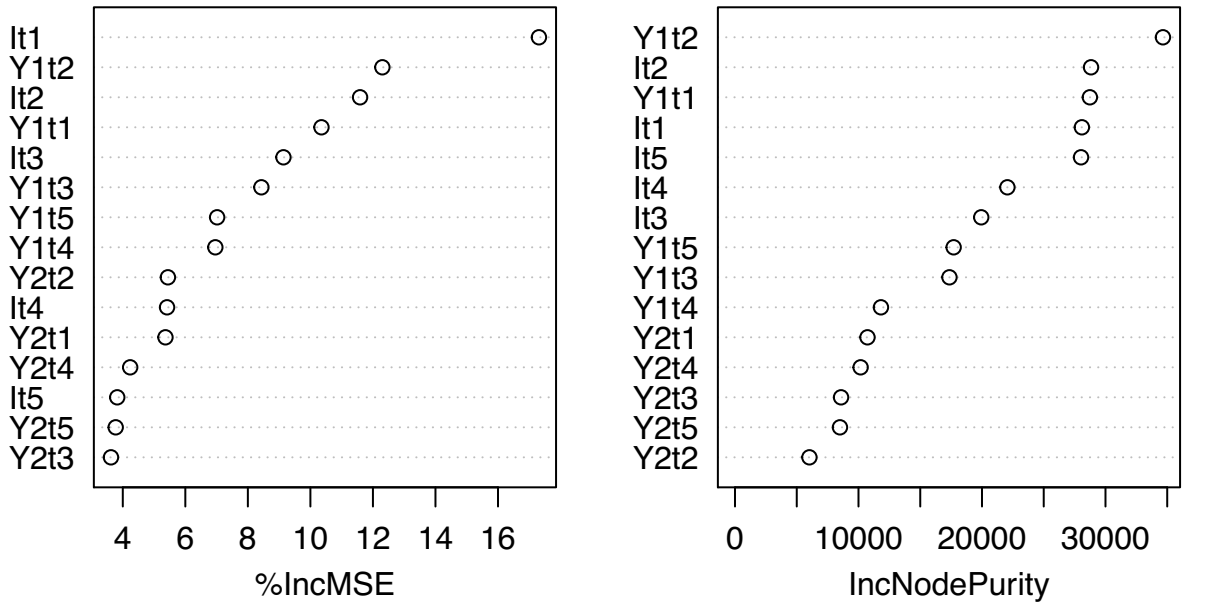
I(t-1) is most important.



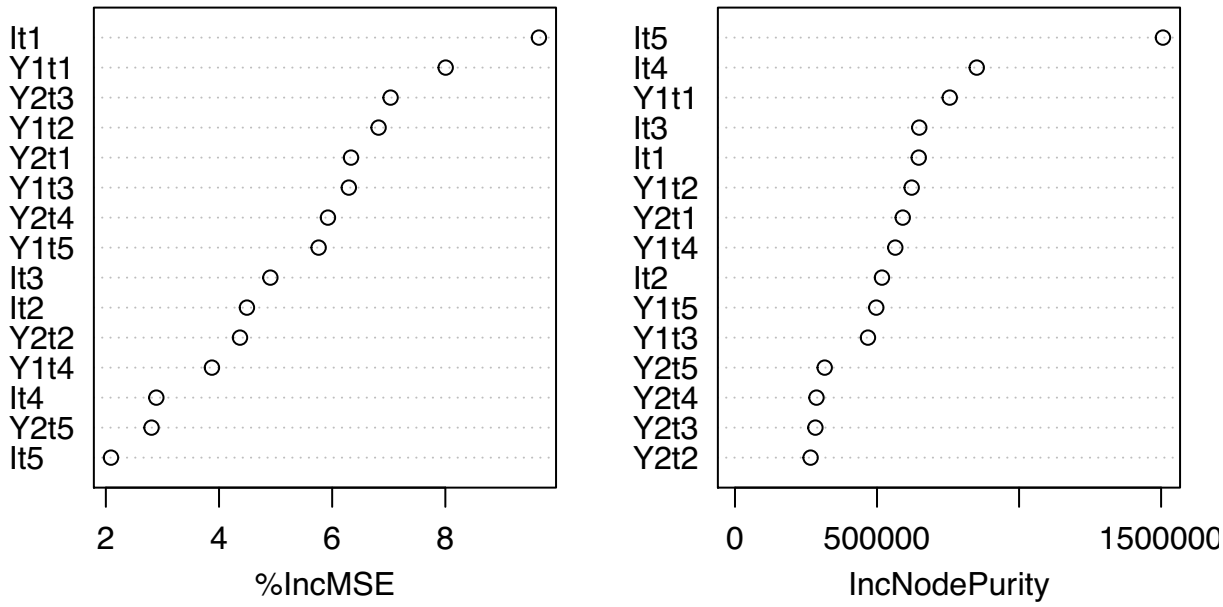
```
model_d_Y2=randomForest(Y2[6:1199]~ ., data = covariates_d_try,  importance=TRUE)
varImpPlot(model_d_Y2)
```




```
V1_d=(Y1[6:1199]-predict(model_d_Y1))^2
V2_d=(Y2[6:1199]-predict(model_d_Y2))^2
Cov12_d=sqrt(V1_d*V2_d)


model_d_V1=randomForest(V1_d~ ., data = covariates_d_try,  importance=TRUE)
varImpPlot(model_d_V1)
model_d_V2=randomForest(V2_d~ ., data = covariates_d_try,  importance=TRUE)
varImpPlot(model_d_V1)
```

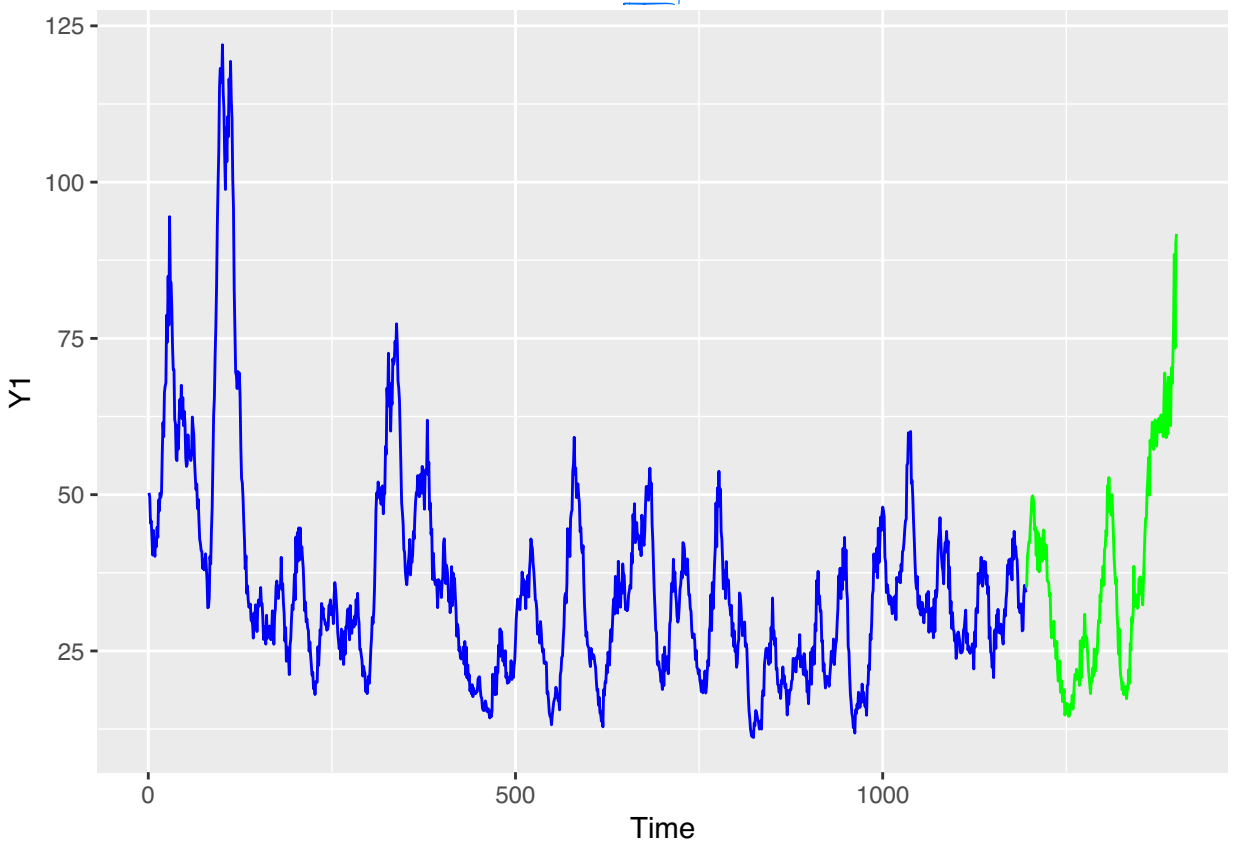



```
model_d_V12=randomForest(Cov12_d~ ., data = covariates_d_try,  importance=TRUE)
varImpPlot(model_d_V12)
```



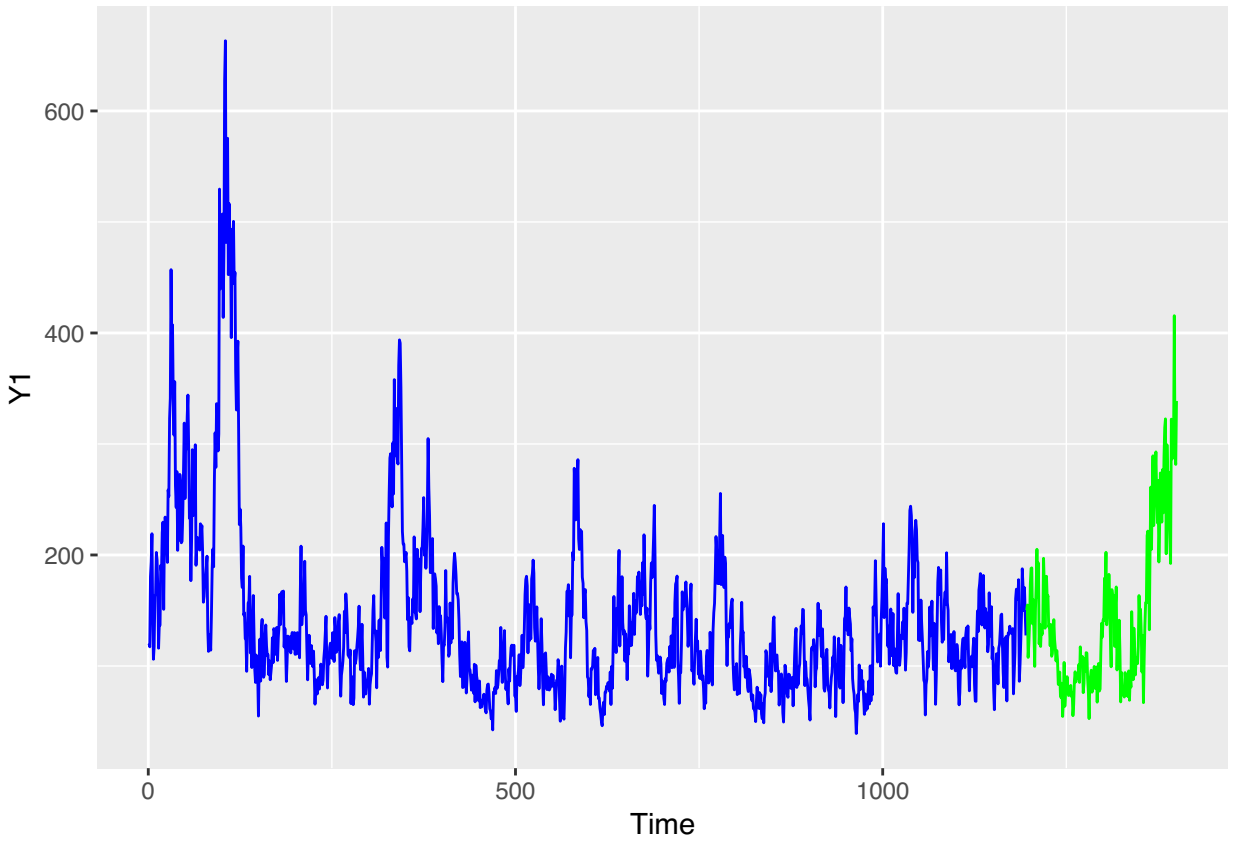
```
predict_Y1=predict(model_d_Y1,covariates_d_try_test)


#first we check for one scenario
Ilast5=I[1196:1200]
Inew=c(Ilast5,Ifuture)
Y1last5=Y1[1196:1200]
Y2last5=Y2[1196:1200]
Y1last205=rep(0,205)
Y2last205=rep(0,205)
for(i in 1:200){
  k=5+i
  currentposition=matrix(c(Y1last5[(k-5):(k-1)],Y2last5[(k-5):(k-1)],Inew[(k-5):(k-1)]),nrow=1)

  colnames(currentposition)=c("Y1t5","Y1t4","Y1t3","Y1t2","Y1t1","Y2t5","Y2t4","Y2t3","Y2t2","Y2t1",
                              "It5","It4","It3","It2","It1")
  prediction_Y=cbind(predict(model_d_Y1,currentposition), predict(model_d_Y2,currentposition))

  Sigma11=predict(model_d_V1,currentposition)
  Sigma22=predict(model_d_V2,currentposition)
  Sigma12=predict(model_d_V12,currentposition)

  if((Sigma12)/sqrt(Sigma11*Sigma22) > 1){
    Sigma12=0.9*sqrt(Sigma11*Sigma22)
  }
  if((Sigma12)/sqrt(Sigma11*Sigma22) < -1){
    Sigma12=(-0.9)*sqrt(Sigma11*Sigma22)
  }
  K_d=matrix(c(Sigma11, Sigma12,Sigma12, Sigma22),ncol=2)

  newpoint=t(prediction_Y)+sqrtm(K_d)%*%rnorm(2,mean=0,sd=1)

  Y1last5=c(Y1last5,newpoint[1])
  Y2last5=c(Y2last5,newpoint[2])
}
plot1_d=ggplot(data=NULL,aes(Time, Y1))+
  geom_line(aes(x=c(1:1195), y=Y1[c(1:1195)]), color="blue")+
  geom_line(aes(x=c(1196:1400), y=Y1last5), color="green")
print(plot1_d)
```

This looks very reasonable prediction.



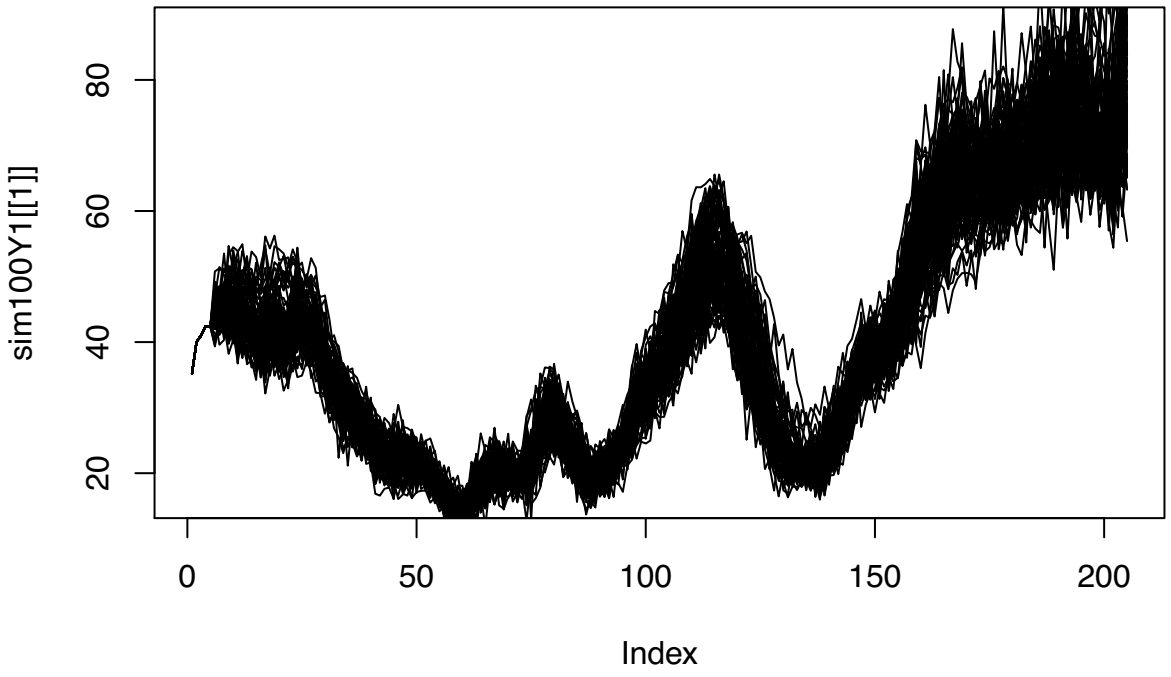

```
plot2_d=ggplot(data=NULL,aes(Time, Y1))+
  geom_line(aes(x=c(1:1195), y=Y2[c(1:1195)]), color="blue")+
  geom_line(aes(x=c(1196:1400), y=Y2last5), color="green")
print(plot2_d)
```

This also looks very reasonable prediction

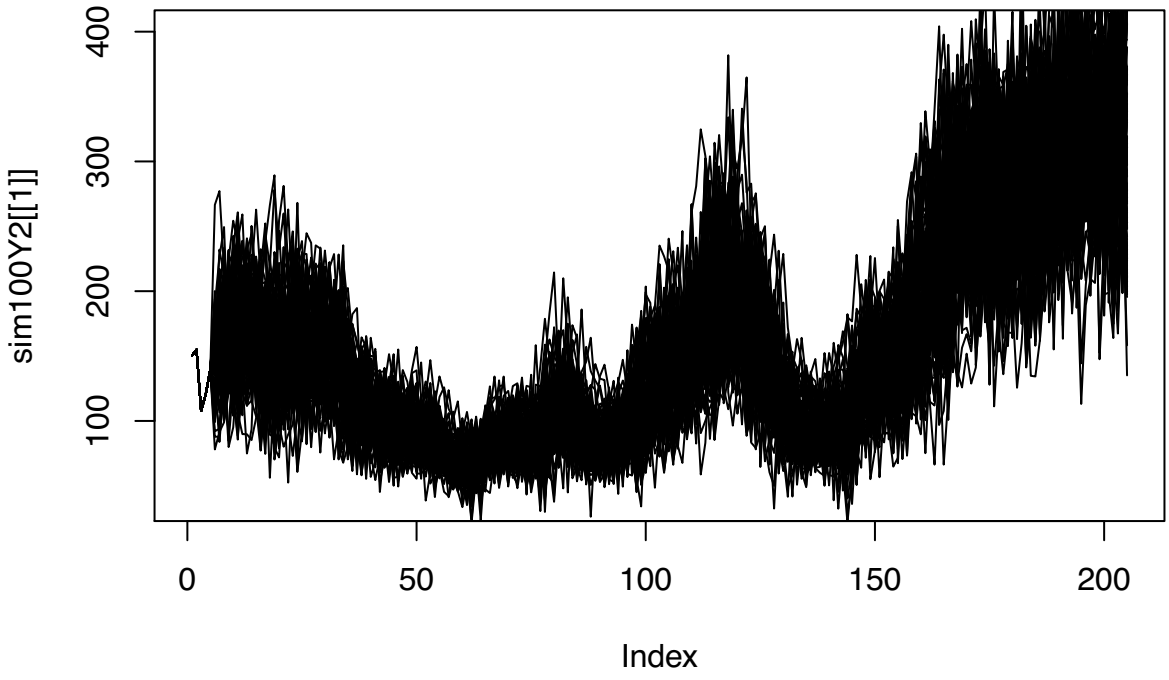



```
#now lets check for 100 scenario
sim100Y1=list()
sim100Y2=list()
for(m in 1:100){
  Ilast5=I[1196:1200]
  Inew=c(Ilast5,Ifuture)
  Y1last5=Y1[1196:1200]
  Y2last5=Y2[1196:1200]
  for(i in 1:200){
    k=5+i
    currentposition=matrix(c(Y1last5[(k-5):(k-1)],Y2last5[(k-5):(k-1)],Inew[(k-5):(k-1)]),nrow=1)

    colnames(currentposition)=c("Y1t5","Y1t4","Y1t3","Y1t2","Y1t1","Y2t5","Y2t4","Y2t3","Y2t2","Y2t1",
                                "It5","It4","It3","It2","It1")
    prediction_Y1=cbind(predict(model_d_Y1,currentposition), predict(model_d_Y2,currentposition))

    Sigma11=predict(model_d_V1,currentposition)
    Sigma22=predict(model_d_V2,currentposition)
    Sigma12=predict(model_d_V12,currentposition)
    #here we check positive definiteness
    if((Sigma12)/sqrt(Sigma11*Sigma22) > 1){
      Sigma12=0.9*sqrt(Sigma11*Sigma22)
    }
    if((Sigma12)/sqrt(Sigma11*Sigma22) < -1){
      Sigma12=(-0.9)*sqrt(Sigma11*Sigma22)
    }
    K_d=matrix(c(Sigma11, Sigma12,Sigma12, Sigma22),ncol=2)
```

```
    newpoint=t(prediction_Y1)+sqrtm(K_d)%*%rnorm(2,mean=0,sd=1)

    Y1last5=c(Y1last5,newpoint[1])
    Y2last5=c(Y2last5,newpoint[2])
  }
  Y1last205=Y1last5
  Y2last205=Y2last5
  sim100Y1[[m]]=Y1last205
  sim100Y2[[m]]=Y2last205
}
plot(sim100Y1[[1]],type="l")
for(i in 2:100){
  lines(sim100Y1[[i]])
}
```




```
plot(sim100Y2[[1]],type="l")
for(i in 2:100){
  lines(sim100Y2[[i]])
}
```



Thank you so much for a wonderful semester.

Kindest regards

Segaty.