R Notebook

This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.3

## -- Attaching packages ------------------------------------------------------------------------------

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ---------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.0.3
```

```r
library(FinTS)
```

```
## Warning: package 'FinTS' was built under R version 4.0.3

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 4.0.3

##
## Attaching package: 'zoo'

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

```r
library(tidyquant)
```

```
## Warning: package 'tidyquant' was built under R version 4.0.3

## Loading required package: lubridate

## Warning: package 'lubridate' was built under R version 4.0.3

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

## Loading required package: PerformanceAnalytics
## Warning: package 'PerformanceAnalytics' was built under R version 4.0.3

## Loading required package: xts

## Warning: package 'xts' was built under R version 4.0.3

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend

## Loading required package: quantmod

## Loading required package: TTR

## Warning: package 'TTR' was built under R version 4.0.3

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

## == Need to Learn tidyquant? ========================================================================
## Business Science offers a 1-hour course - Learning Lab #9: Performance Analysis & Portfolio Optimiza
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```r
library(ggplot2)
library(dplyr)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.0.3

##
## Attaching package: 'forecast'

## The following object is masked from 'package:FinTS':
##
##     Acf
```

```r
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.0.3

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.0-2
```

```r
library(ESGtoolkit)
```

```
## Loading required package: CDVine
## The CDVine package is no longer developed actively.
## Please consider using the more general VineCopula package
## (see https://CRAN.R-project.org/package=VineCopula),
## which extends and improves the functionality of CDVine.
```

```
## Loading required package: gridExtra
```

```
## Warning: package 'gridExtra' was built under R version 4.0.3
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## Loading required package: reshape2
```

```
## Warning: package 'reshape2' was built under R version 4.0.3
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
## Loading required package: ycinterextra
```

```
## Warning: package 'ycinterextra' was built under R version 4.0.3
```

```
## Loading required package: compiler
```

```
##
## Attaching package: 'ycinterextra'
```

```
## The following objects are masked from 'package:stats':
##
##     deviance, fitted, residuals
```

```
## The following object is masked from 'package:base':
##
##     as.list
```

```r
library(lubridate)
library(plyr)
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
## The following object is masked from 'package:purrr':
##
##     compact
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.0.3
```

```
## corrplot 0.84 loaded
```

```
library(reshape2)
library(fredr)
```

```
## Warning: package 'fredr' was built under R version 4.0.3
```

```
library(RQuantLib)
```

```
## Warning: package 'RQuantLib' was built under R version 4.0.3
```

```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.0.3
```

```
library(reshape2)
library(scales)
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.0.3
```

```
## Registered S3 methods overwritten by 'ggfortify':
##   method                 from
##   autoplot.Arima         forecast
##   autoplot.acf           forecast
##   autoplot.ar            forecast
##   autoplot.bats          forecast
##   autoplot.decomposed.ts forecast
##   autoplot.ets           forecast
##   autoplot.forecast      forecast
##   autoplot.stl           forecast
##   autoplot.ts            forecast
##   fitted.ar              forecast
##   fortify.ts             forecast
##   residuals.ar           forecast
```

```
setwd('D:/OneDrive/College Notebook/Boston University/Fall Senior Year/CS 506/Project/CS506-Fall2020-Pr
directories=lapply(c("/card/", "/auto/", "/consumer/"), function(x) paste(getwd(), x, sep=""))

# card_files=lapply(list.files(directories[[1]]), function(x) read_excel(x))
```

```r
setwd(directories[[2]])
auto_files=lapply(list.files(getwd()), function(x) data.frame(read_excel(x)))
auto_names=lapply(list.files(getwd()), function(x) sub('\\.xlsx$', '', x))

setwd(directories[[3]])
consumer_files=lapply(list.files(getwd()), function(x) data.frame(read_excel(x)))
consumer_names=lapply(list.files(getwd()), function(x) sub('\\.xlsx$', '', x))


refSecurities=read_excel('D:/OneDrive/College Notebook/Boston University/Fall Senior Year/CS 506/Projec
```

```
## New names:
## * `` -> ...1
```

```r
yieldCodes=c("DGS1", "DGS2", "DGS3", "DGS5", "DGS7", "DGS10", "DGS20", "DGS30")
forwardCodes=c('THREEFF1', 'THREEFF2', 'THREEFF3', 'THREEFF4', 'THREEFF5', 'THREEFF6', 'THREEFF7', 'THR
fredr_set_key('488442838674a6777f462b62cca2b708')

yieldConstruction=function(id) {
        df=fredr_series_observations(id, observation_start = as.Date('2000-01-01'), observation_end=as.
        names(df)[3]=id
        return(df[, !names(df)=='series_id'])
}

yieldCurve=yieldConstruction("DGS1")

for(i in yieldCodes[c(2:length(yieldCodes))]) {
        df=yieldConstruction(i)

        yieldCurve=left_join(yieldCurve, df, by=c("date"))
}
```

```
## Request failed [504]. Retrying in 1 seconds...
```

```r
forwardCurve=yieldConstruction("THREEFF1")

for(i in forwardCodes[c(2:length(forwardCodes))]) {
        df=yieldConstruction(i)

        forwardCurve=left_join(forwardCurve, df, by=c("date"))
}
```

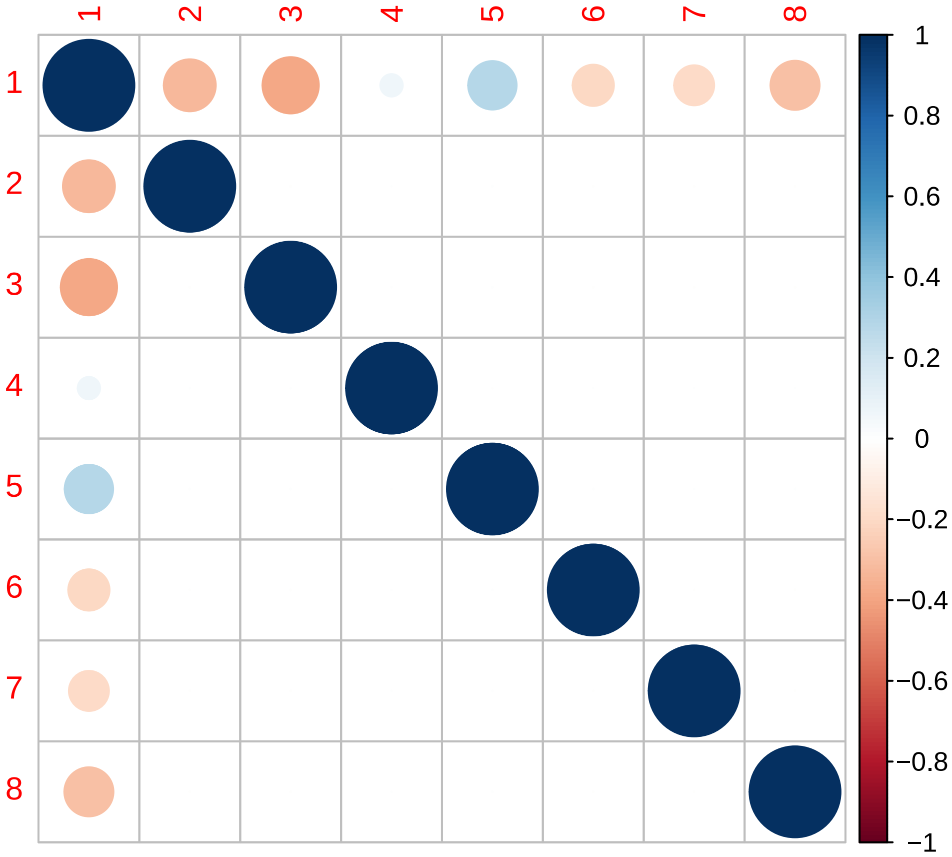
```r
preProcessDF=function(df, security_name) {

        returns = log(df$Price[-1]/df$Price[-length(df$Price)])
        df=df[1:length(df$Price)-1,]
        df$returns = returns


        ref=refSecurities[which(refSecurities$`Security Name`==security_name ),]
        maturityDate=as.Date(ref$Maturity)
        period=2
        rates=c(ref$Cpn/100)
        issueDate=as.Date(ref$`Issue Date`)
        df$YTM=apply(df, 1, function(x) FixedRateBondYield(settlementDays=1, as.double(x['Price']),
                                                           faceAmount=100, effectiveDate=as.Date(x['Date
                                                           maturityDate,
                                                         period=1, calendar='UnitedStates/GovernmentBond
                                                         dayCounter=2, businessDayConvention=0, compound
                                                         redemption=100, issueDate))


        # names(df)[names(df)=='Date'] = 'date'
        df=df %>% map_df(rev)
        start_date = as.Date(df[1, ]$Date, format="%Y-%m-%d")
        start_year = year(start_date)
        start_month = month(start_date)

        return(ts(subset(df, select=-c(Date)), start=c(start_year, start_month), frequency=12))


        # return(df %>% map_df(rev))
       #  return(df)



}

joinCurves=function(df) {

        df=left_join(df, yieldCurve %>%
            group_by(date = format(as.Date(date), '%Y-%m')) %>%
            summarise_each(funs( if(length(na.omit(.))>=15)
                               mean(., na.rm=TRUE) else NA_real_), DGS1:DGS30), by=c('date'))

        df=left_join(df, forwardCurve %>%
            group_by(date = format(as.Date(date), '%Y-%m')) %>%
            summarise_each(funs( if(length(na.omit(.))>=15)
                               mean(., na.rm=TRUE) else NA_real_), THREEFF1:THREEFF10), by=c('date'))

        return(df)
}

consumer_dfs = lapply(c(1:length(consumer_files)), function(x) preProcessDF(consumer_files[[x]], consume
auto_dfs = lapply(c(1:length(auto_files)), function(x) preProcessDF(auto_files[[x]], auto_names[x]))
```

```
doTimeSeries = function(df, name, max_k) {
  df = df
  X = as.matrix(df[, !colnames(df) %in% c("YTM", 'date', 'returns', 'Price')])
  y = df[, colnames(df) %in% c('returns', 'date')]

  X_train = head(X, round(nrow(X) * 0.6))
  s = nrow(X) - nrow(X_train)

  y_train = head(y, round(length(y) * 0.6))

  X_test = tail(X, s)
  y_test = tail(y, s)
  loadings = svd(X_train)$v
  rownames(loadings) = colnames(X_train)
  Z = X_train%*%loadings
  # colnames(Z) = paste0("PC", 1:ncol(X_train))
  corrplot(cor(Z))

  sing.val = svd(X_train/sqrt(ncol(X_train)))$d
  plot(sing.val, xlab="axis", ylab="singular value")
  abline(h=1, col="red")

  meltR = melt(loadings)
  p=ggplot(meltR, aes(x=Var1, y = value)) +
    geom_bar(stat="identity") +
    coord_flip() +
    facet_wrap(. ~ Var2) +
    ggtitle("Loadings")


  loadings = svd(X_test)$v
  rownames(loadings) = colnames(X_test)
  Z_test = X_test%*%loadings
  # colnames(Z_test) = paste0("PC", 1:ncol(X_test))

  k = min(nrow(X_test), max_k)


  if(length(which(sing.val <= 0.05)) > 1) {
    k = min(which(sing.val <= 0.05)[1], k)
  }

  model = auto.arima(y_train,  xreg=as.matrix(Z[, 1:k]), method='ML', seasonal=FALSE)
  y_pred = forecast(model, h=length(y_test), xreg=as.matrix(Z_test[, 1:k]))

  y_df=data.frame(y)
  y_df$date=as.Date(time(y))

  y_pred_df=data.frame(y_pred$mean)
  y_pred_df$date=as.Date(time(y_test))


  p=autoplot(y_pred, fcol="Forecast") + geom_line(data=y_df, aes(x=date, y=y, color='Actual'), group=2)
    scale_color_manual(name = element_blank(), labels = c("Actual"),
      values = c("darkred", "steelblue")) +
    ggtitle(paste(paste("Prediction Results for", name))) + xlab("Date") + ylab("Return") +
    theme(plot.title = element_text(hjust = 0.5))
  print(p)

  return(model)
}
```

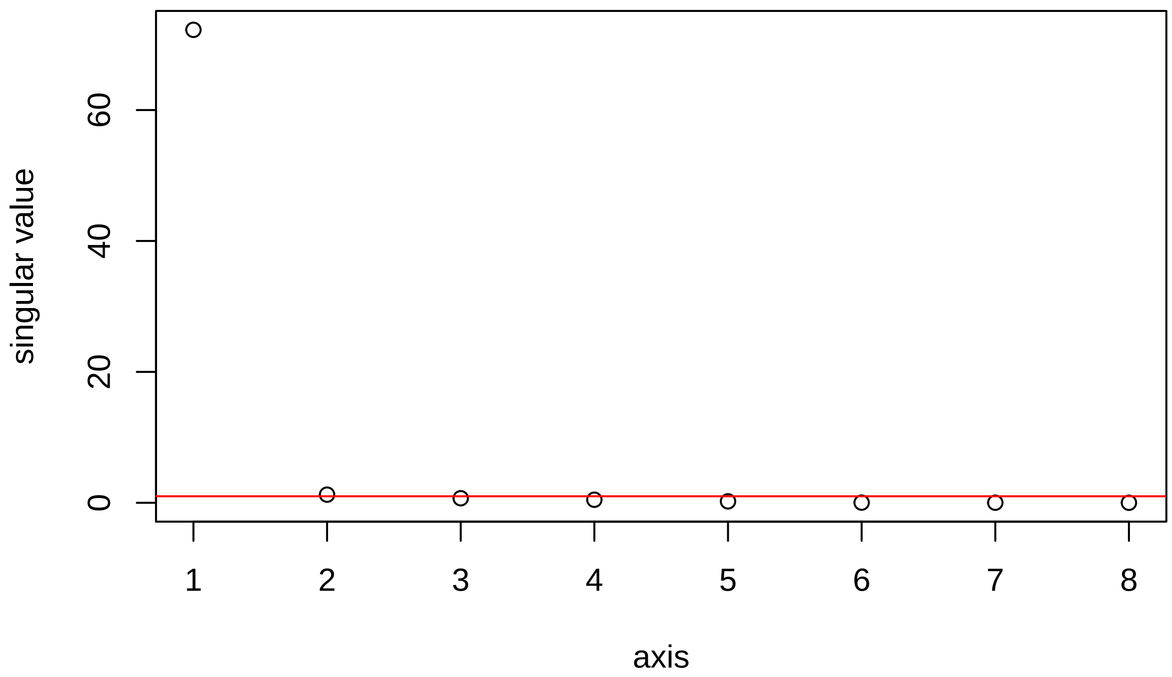
```
consumer_models=list()
auto_models=list()
```

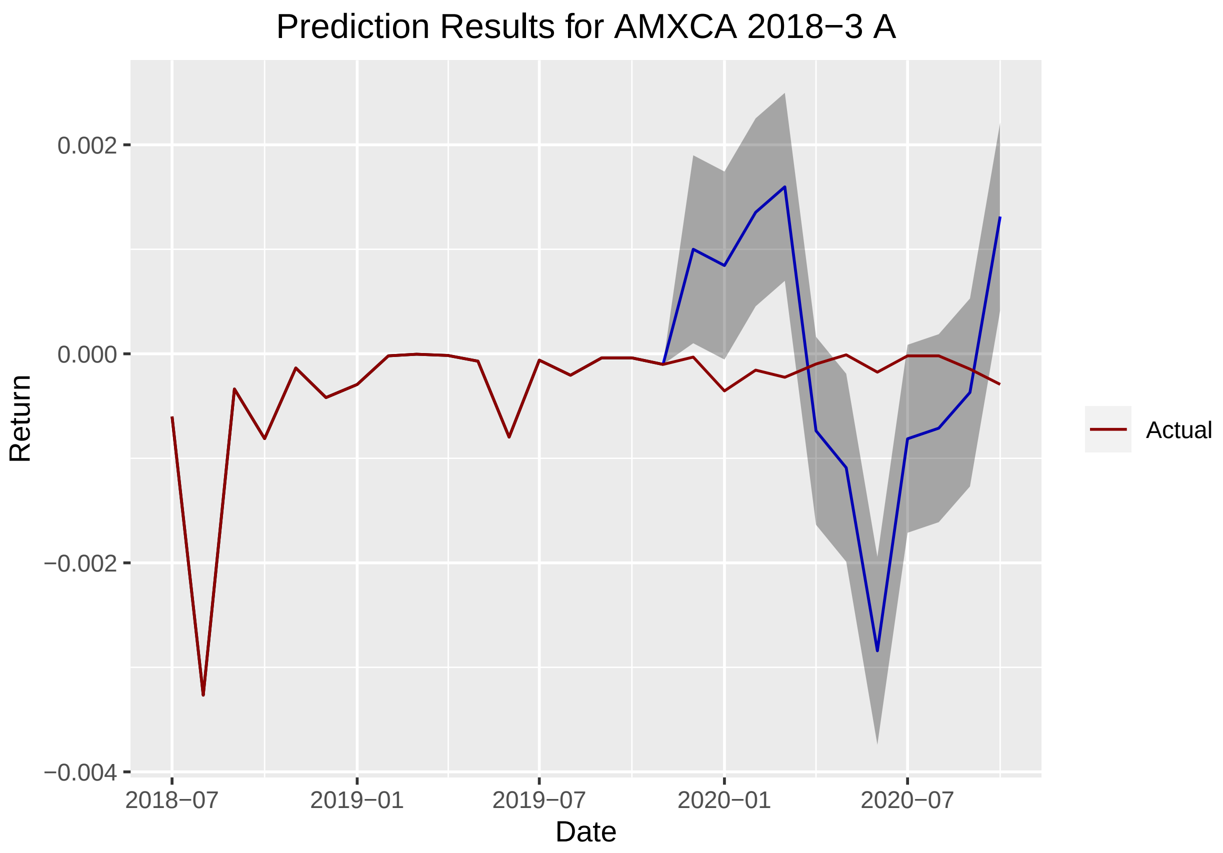
```r
for(i in c(1:length(consumer_dfs))) {
  print(i)
  consumer_models[[i]]=doTimeSeries(consumer_dfs[[i]], consumer_names[[i]], 8)
}
```

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

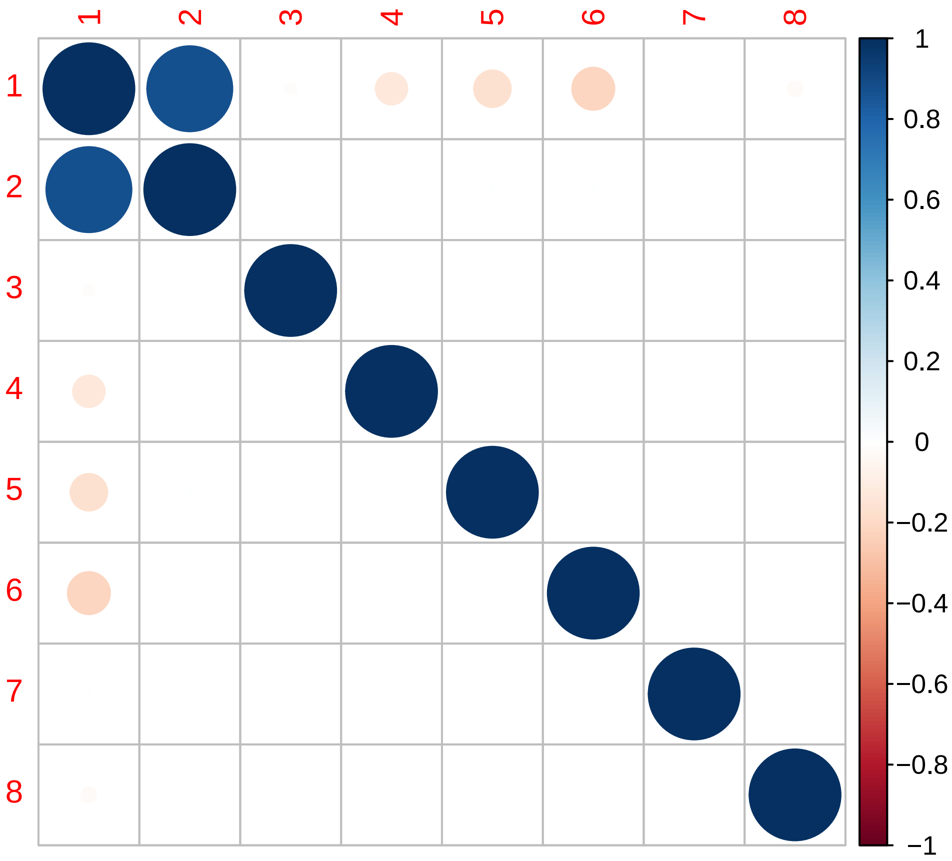


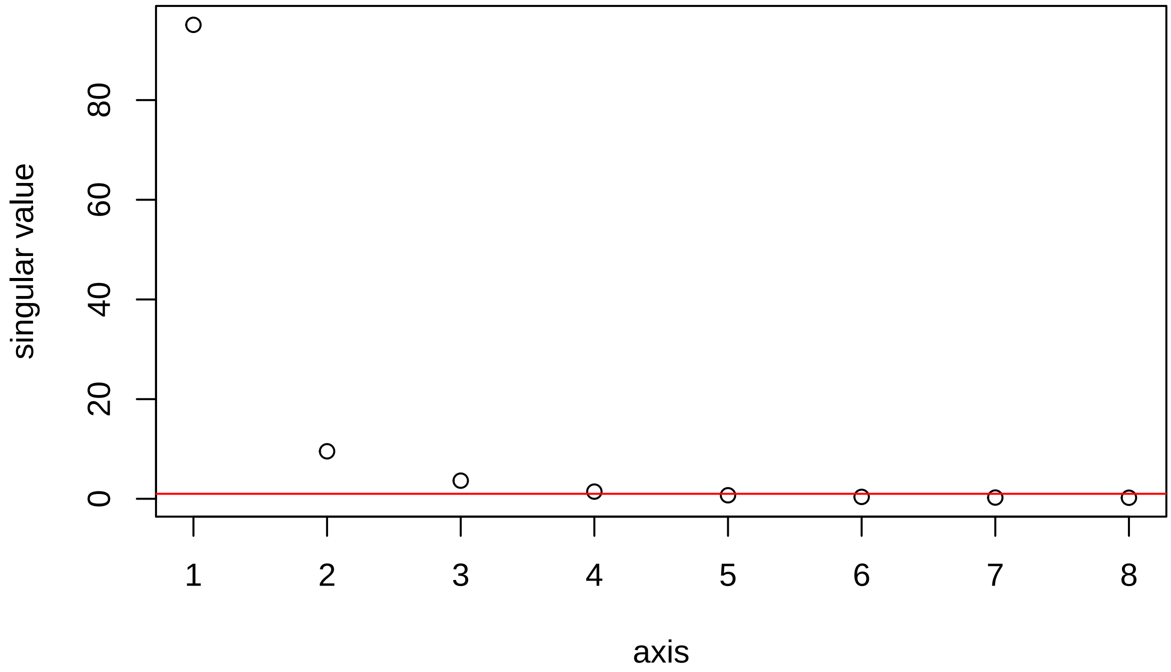
```
## Warning: `filter_()` is deprecated as of dplyr 0.7.0.
## Please use `filter()` instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

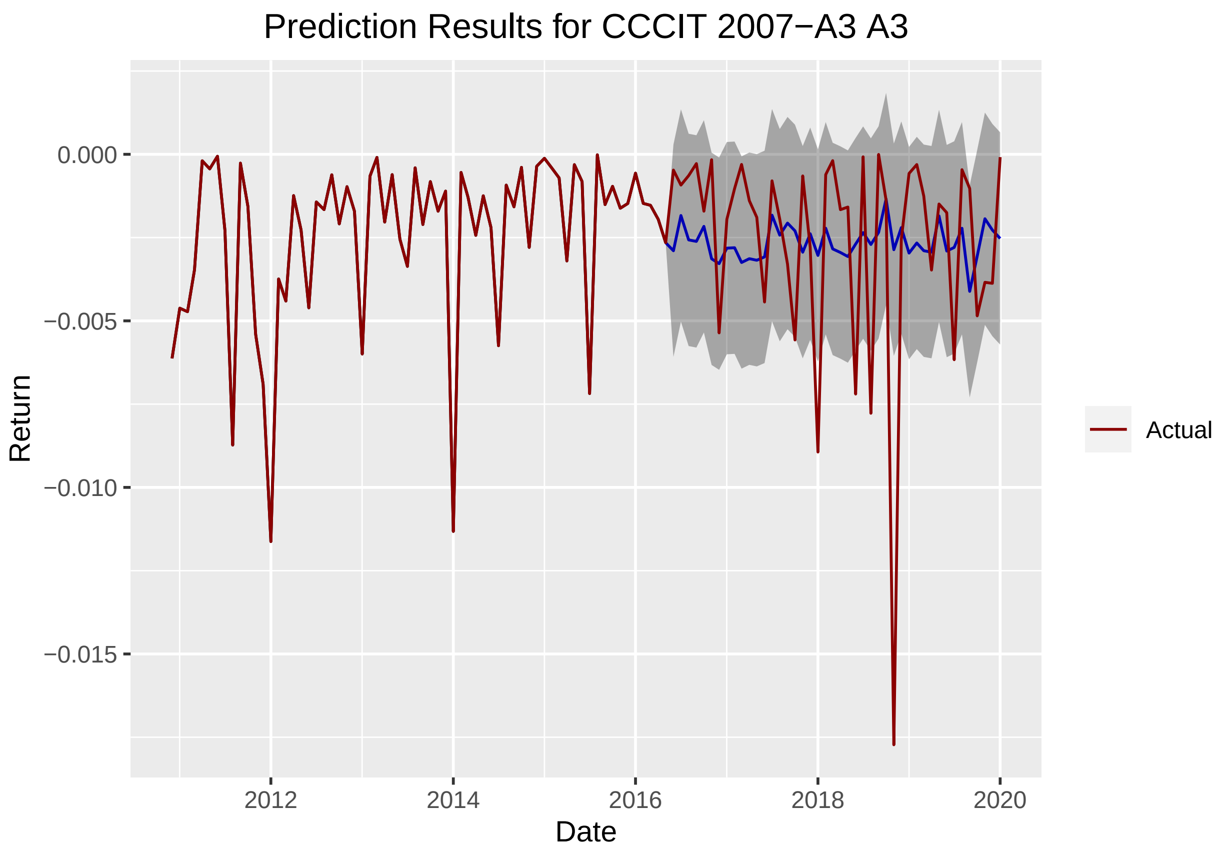

singular value
0
20
40
60
1
2
3
4
5
6
7
8
axis


Prediction Results for AMXCA 2018−3 A
Return
0.002
0.000
−0.002
−0.004
2018−07
2019−01
2019−07
2020−01
2020−07
Date
Actual

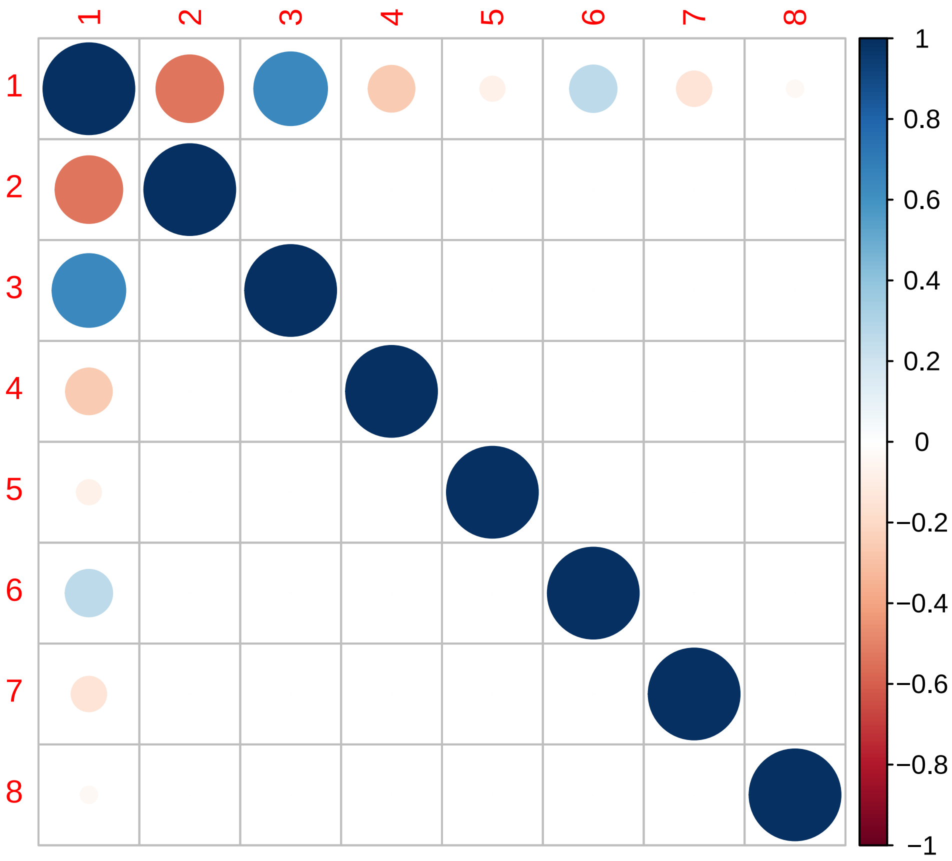
```
## [1] 2
```

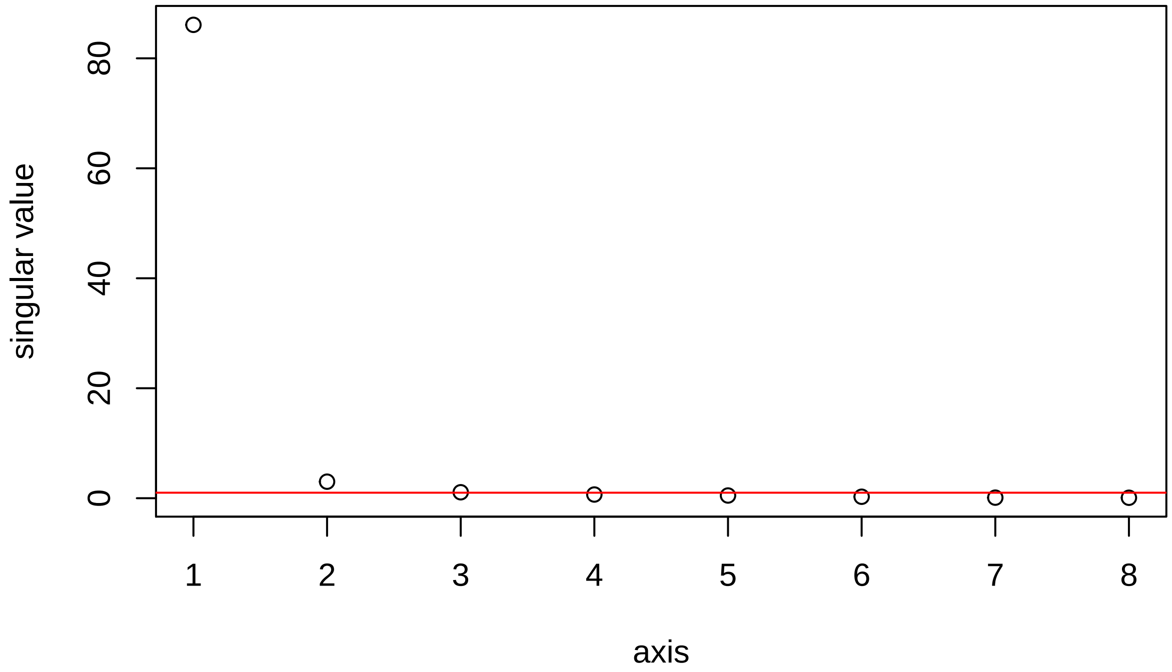

1
2
3
4
5
6
7
8
1
2
3
4
5
6
7
8
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

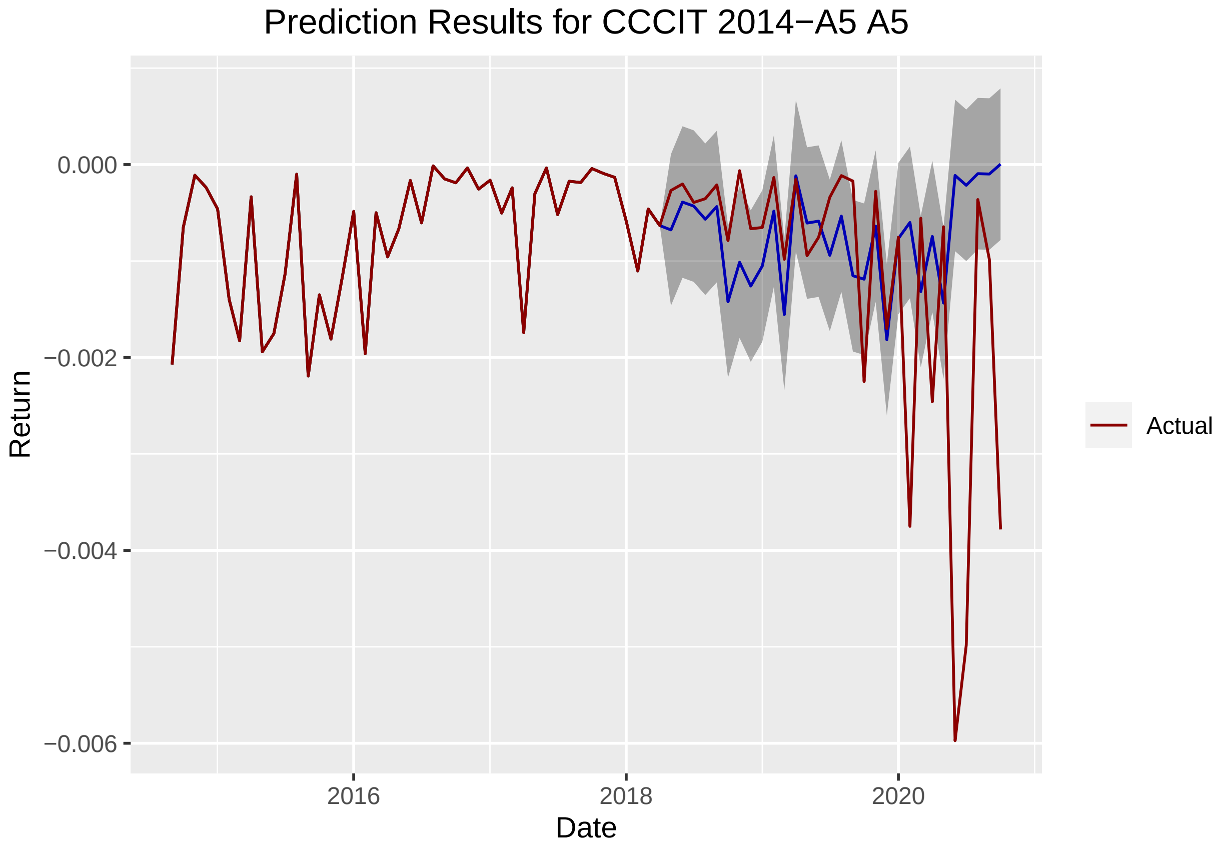

singular value
0
20
40
60
80
1
2
3
4
5
6
7
8
axis


Prediction Results for CCCIT 2007−A3 A3
0.000
−0.005
−0.010
−0.015
Return
2012
2014
2016
2018
2020
Date
Actual

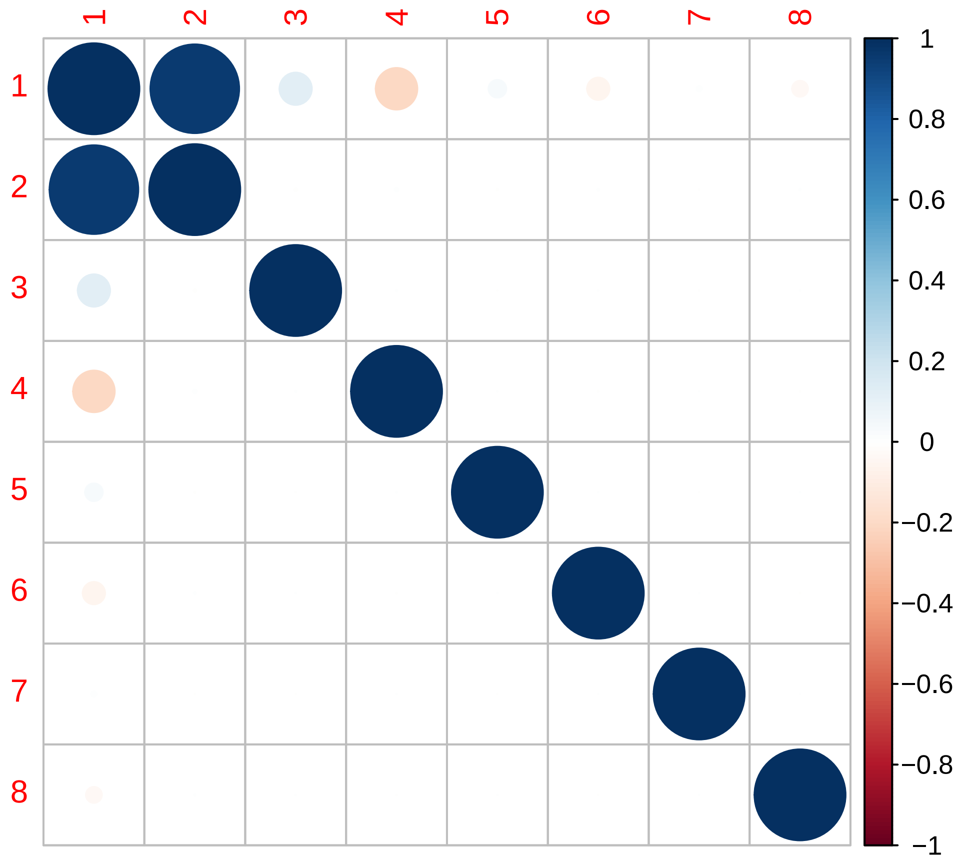
```
## [1] 3
```

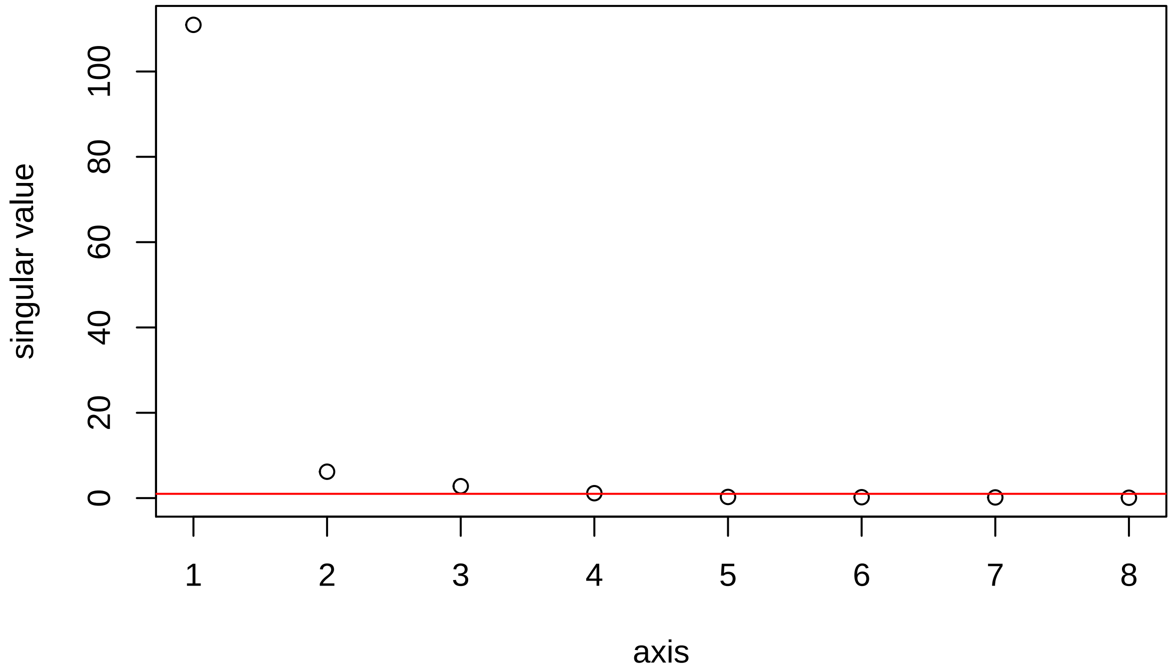

1
2
3
4
5
6
7
8
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

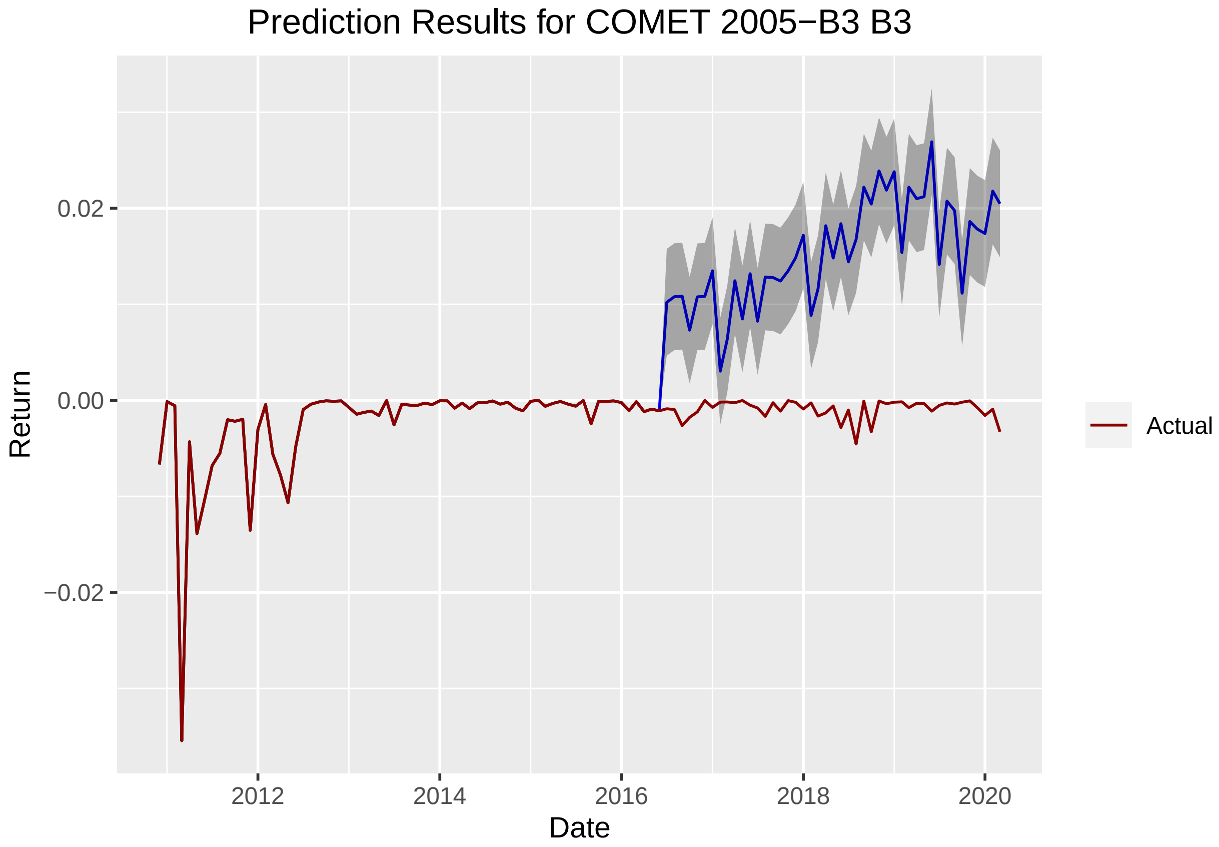

singular value
0
20
40
60
80
1
2
3
4
5
6
7
8
axis


Prediction Results for CCCIT 2014−A5 A5
Return
0.000
−0.002
−0.004
−0.006
2016
2018
2020
Date
Actual

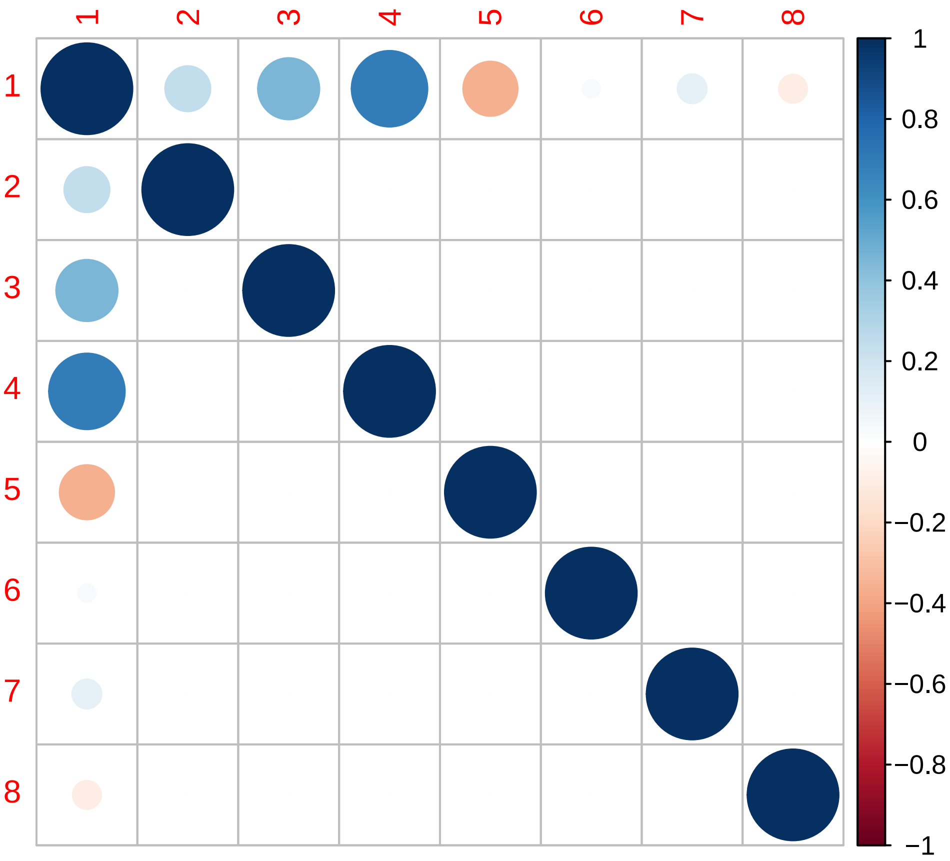
```
## [1] 4
```

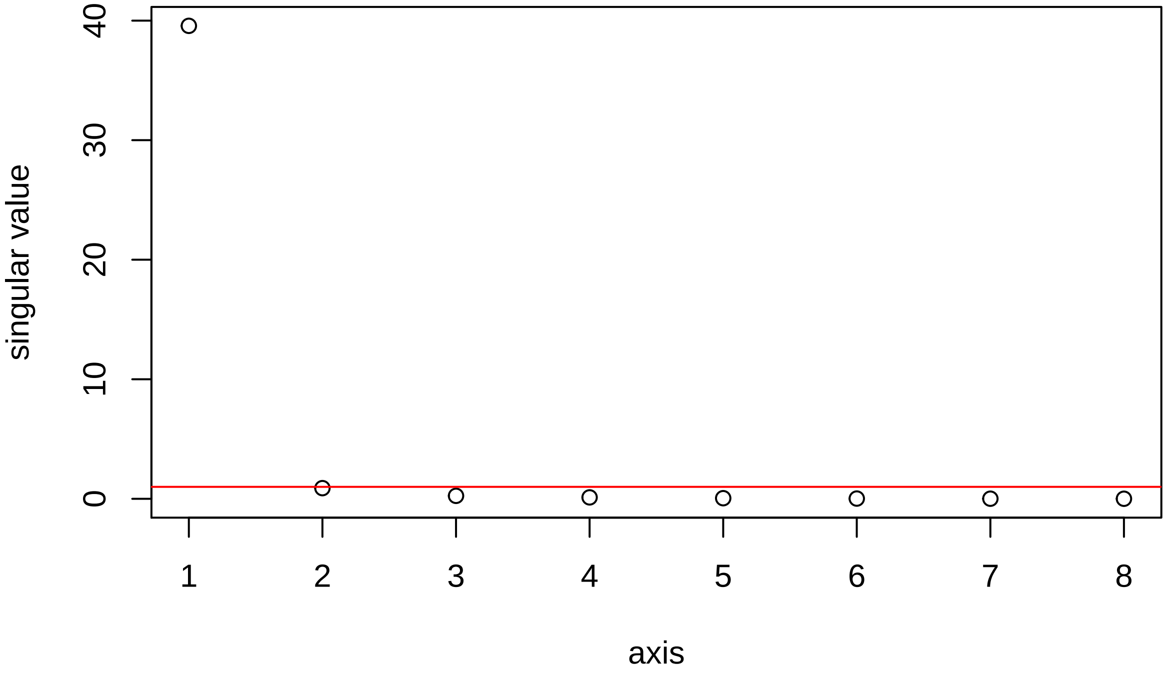

1
2
3
4
5
6
7
8
1
2
3
4
5
6
7
8
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

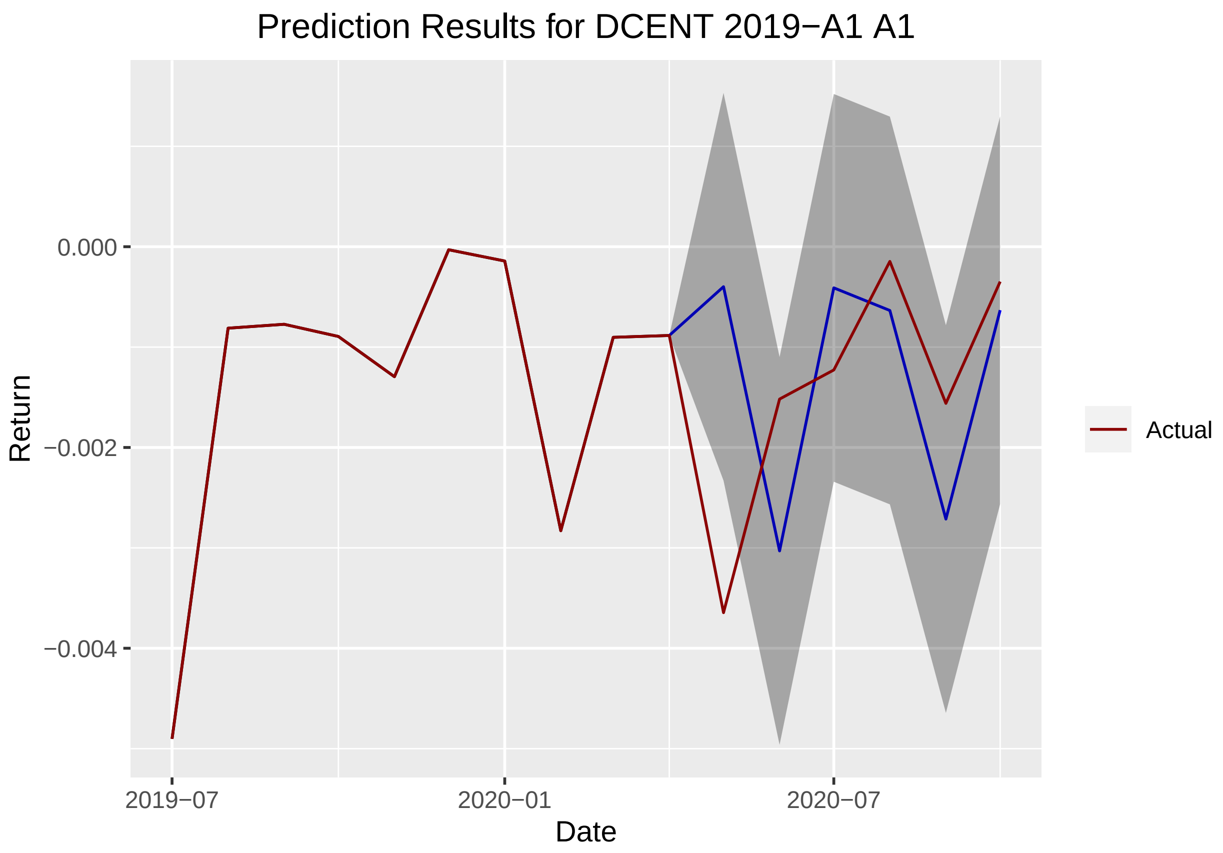

singular value
0
20
40
60
80
100
1
2
3
4
5
6
7
8
axis


Prediction Results for COMET 2005−B3 B3
Return
0.02
0.00
−0.02
2012
2014
2016
2018
2020
Date
Actual

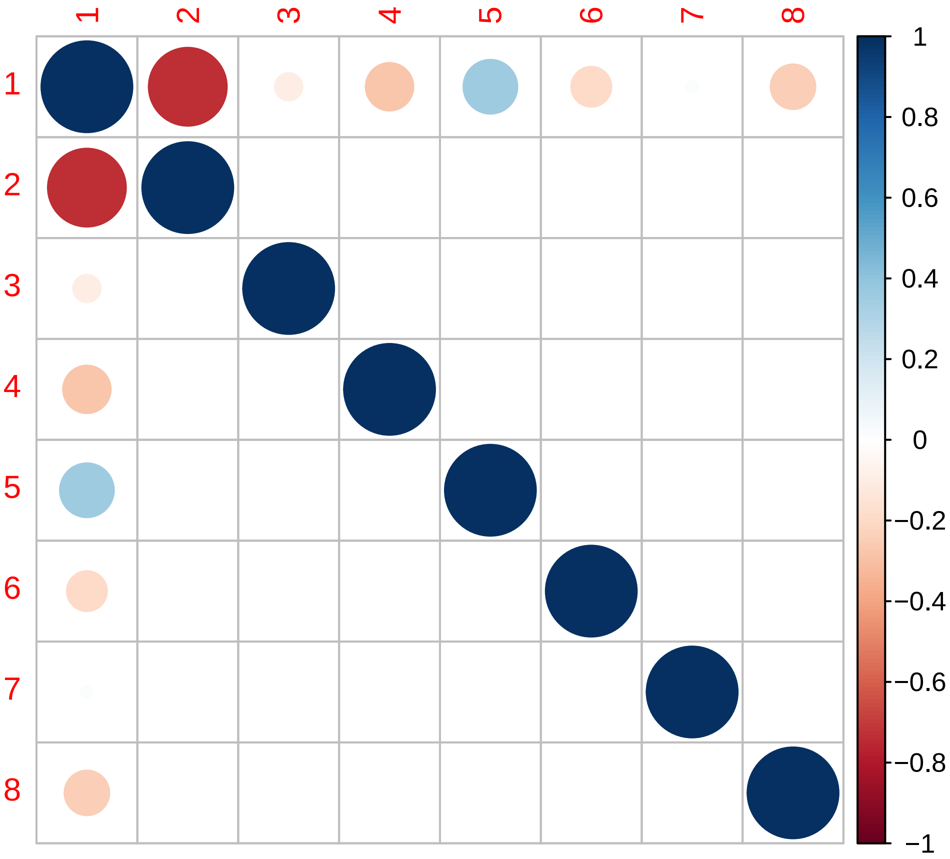
```
## [1] 5
```

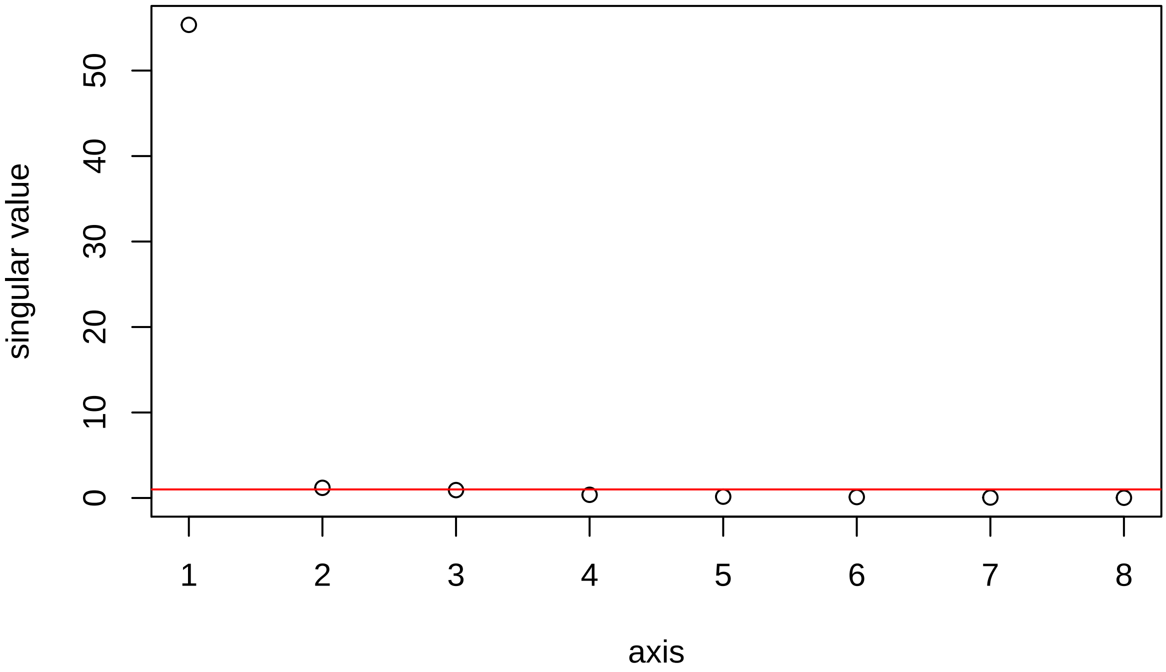

1
2
3
4
5
6
7
8
1
2
3
4
5
6
7
8
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

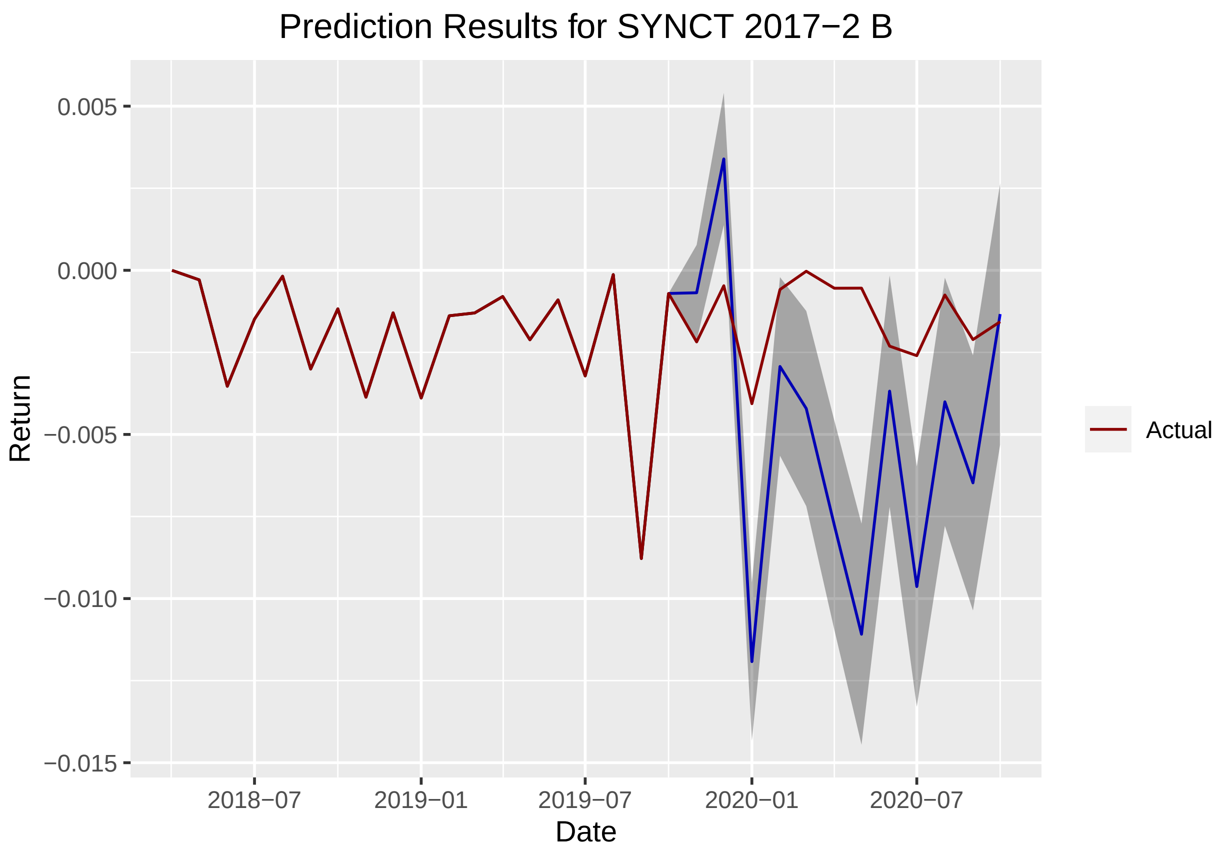

singular value
0
10
20
30
40
1
2
3
4
5
6
7
8
axis


Prediction Results for DCENT 2019−A1 A1
Return
0.000
−0.002
−0.004
2019−07
2020−01
2020−07
Date
Actual

```
## [1] 6
```

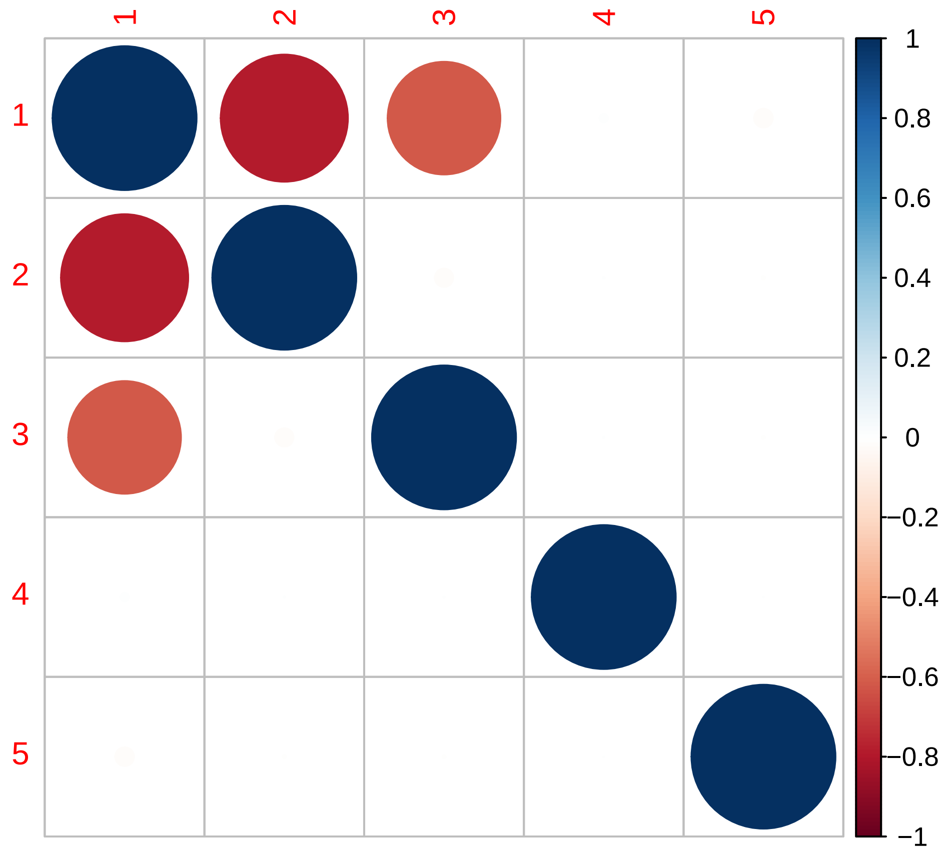

1
2
3
4
5
6
7
8
1
2
3
4
5
6
7
8
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

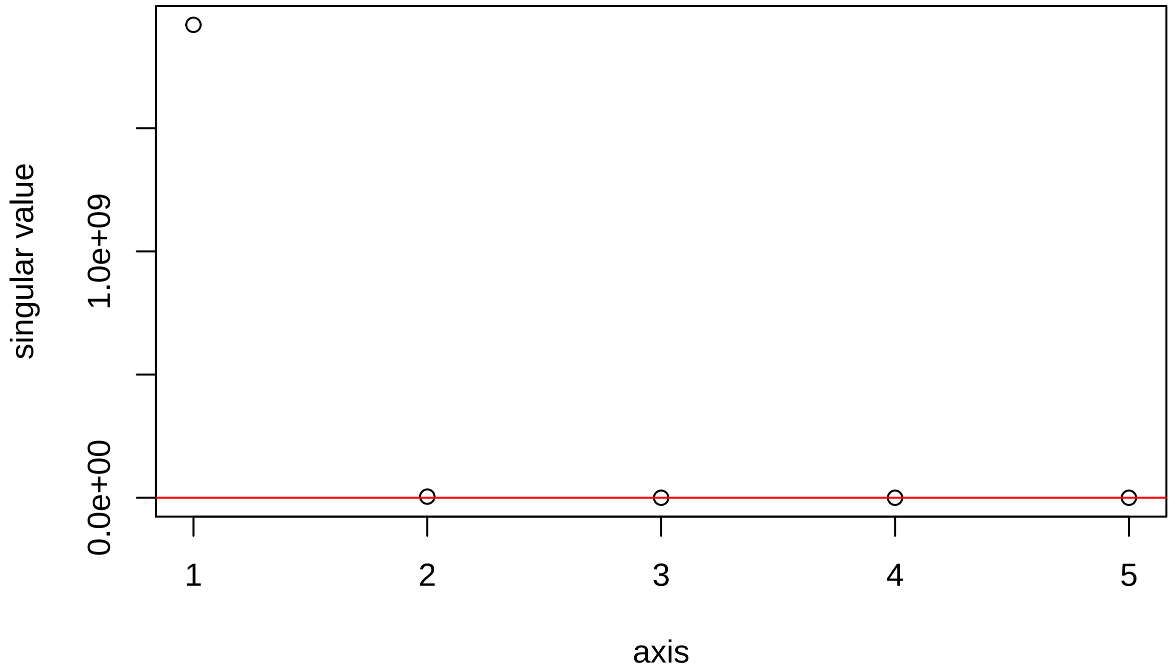

singular value
0
10
20
30
40
50
1
2
3
4
5
6
7
8
axis

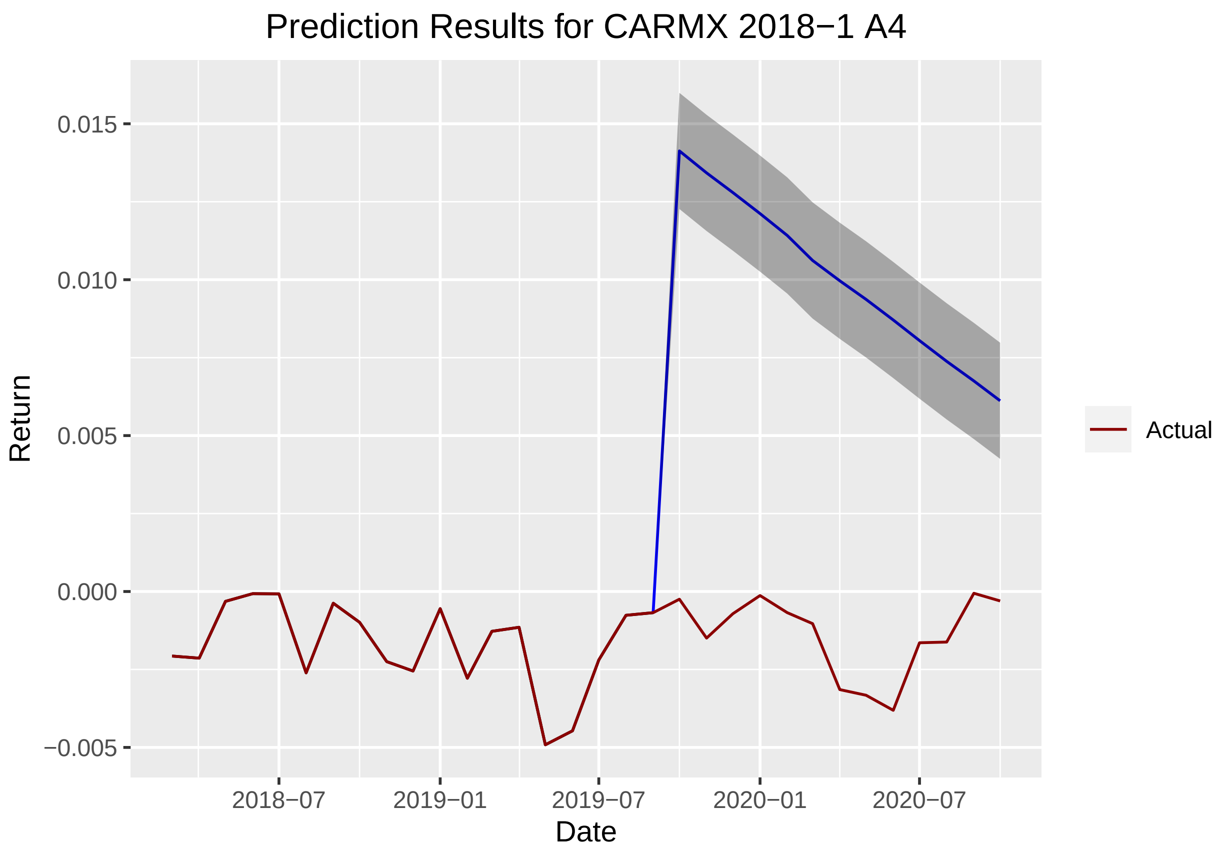


```
for(i in c(1:length(auto_dfs))) {
  print(i)
  print(auto_names[[i]])
  auto_models[[i]]=doTimeSeries(auto_dfs[[i]], auto_names[[i]], 3)
}
```

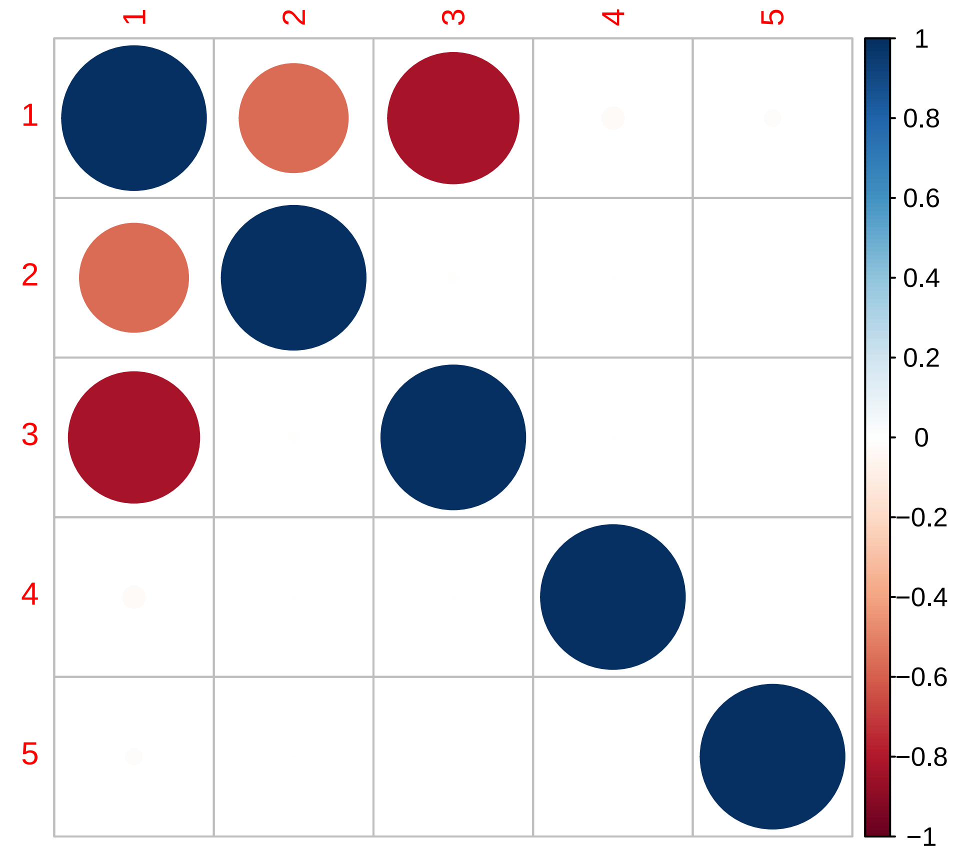
```
## [1] 1
## [1] "CARMX 2018-1 A4"
```

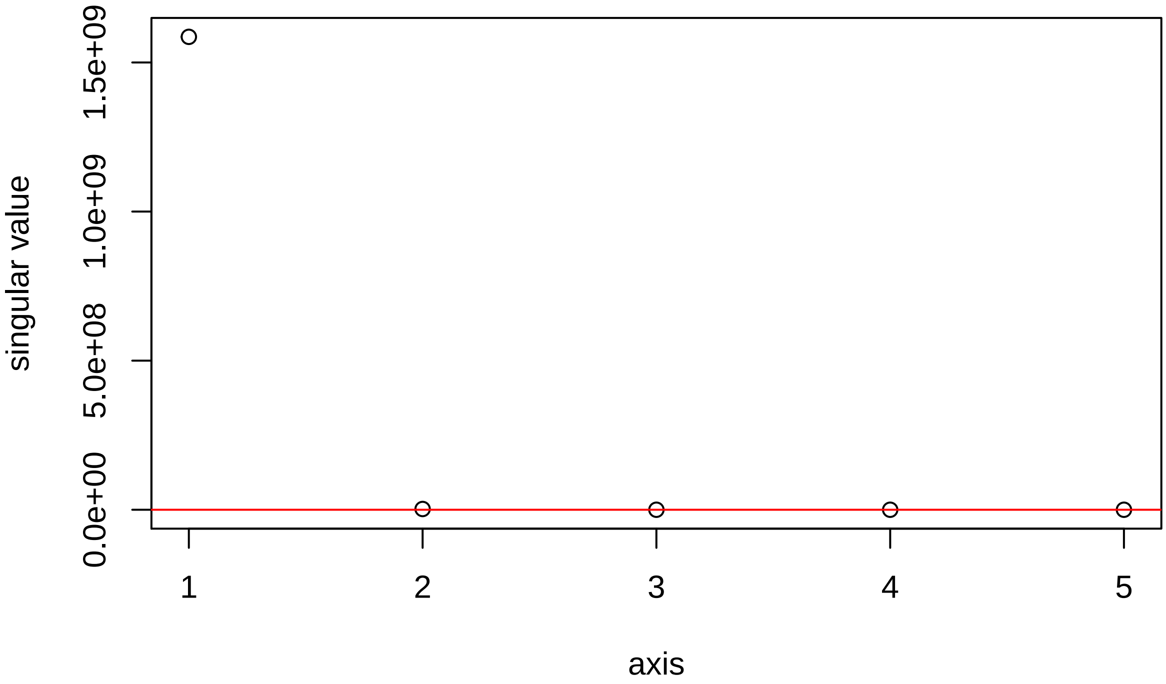

1
2
3
4
5
1
2
3
4
5
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

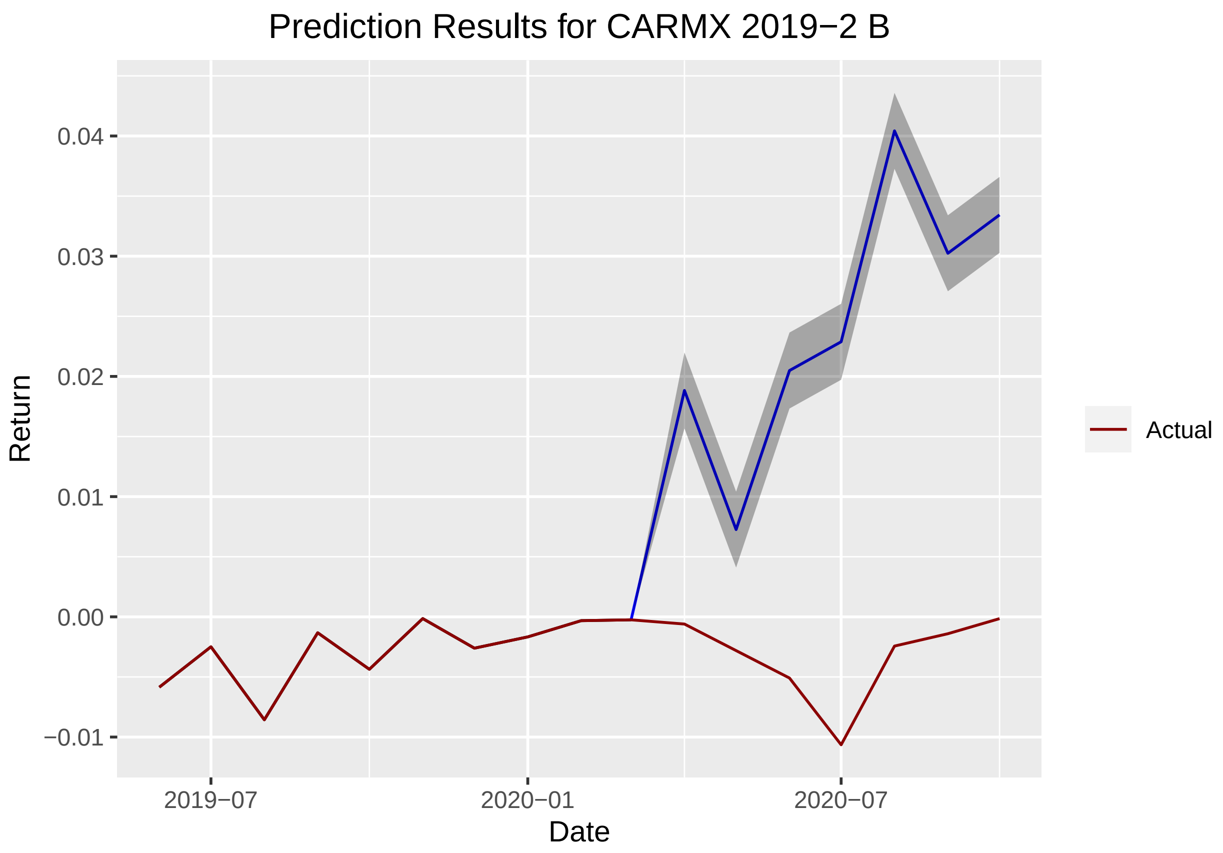

singular value
1.0e+09
0.0e+00
1
2
3
4
5
axis



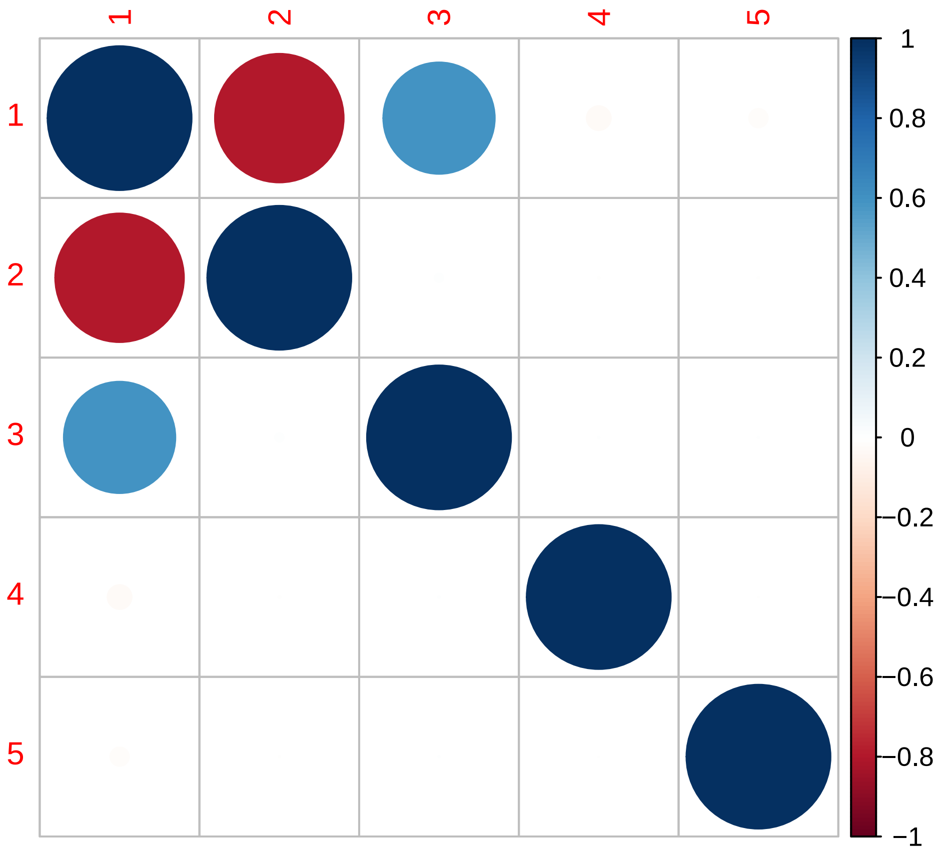
```
## [1] 2
## [1] "CARMX 2019-2 B"
```

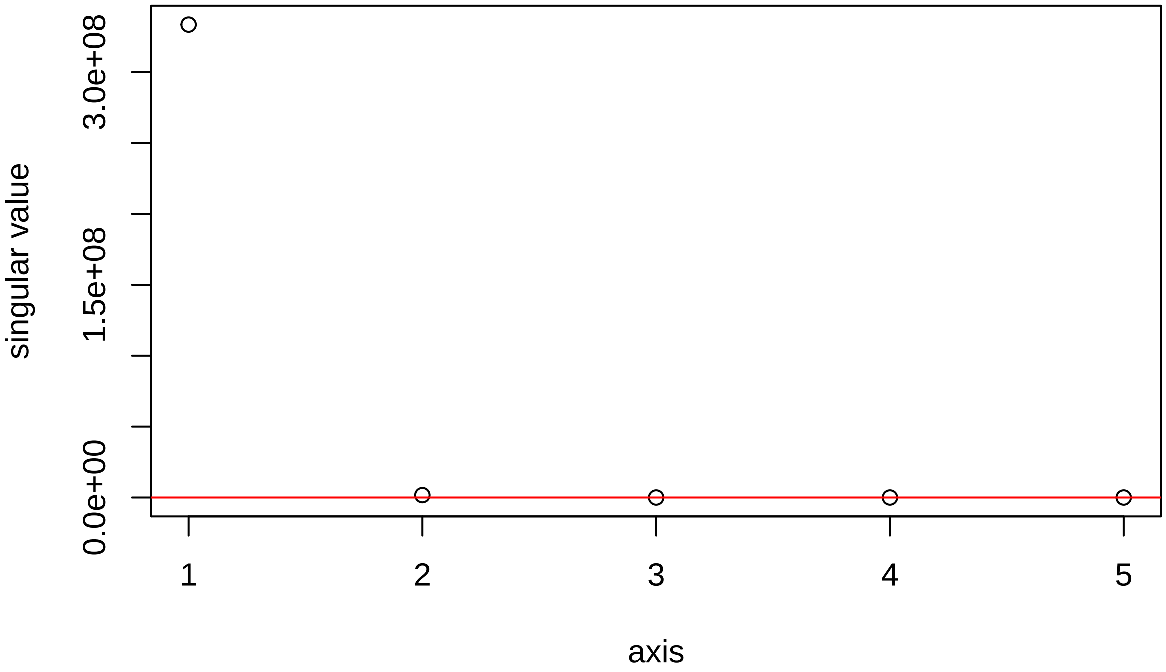

1
2
3
4
5
1
2
3
4
5
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

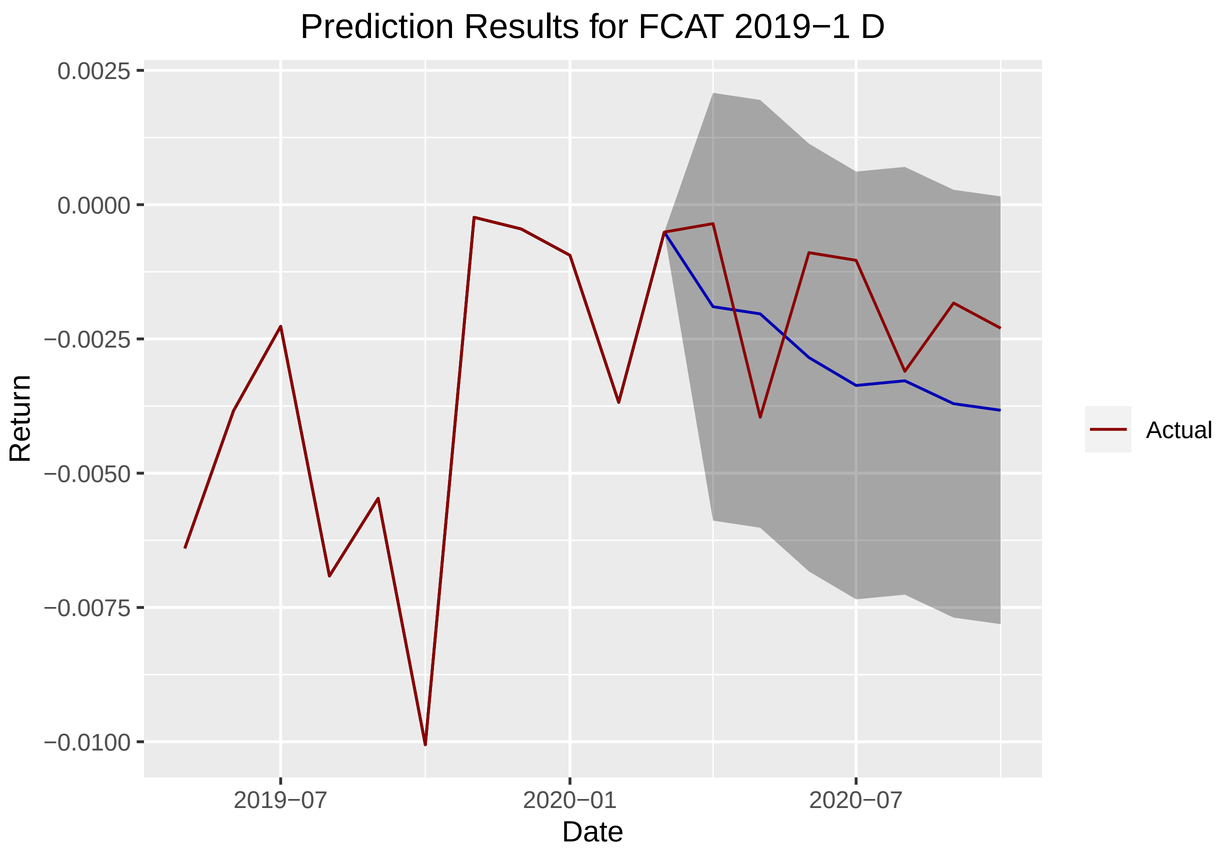

singular value
0.0e+00
5.0e+08
1.0e+09
1.5e+09
1
2
3
4
5
axis



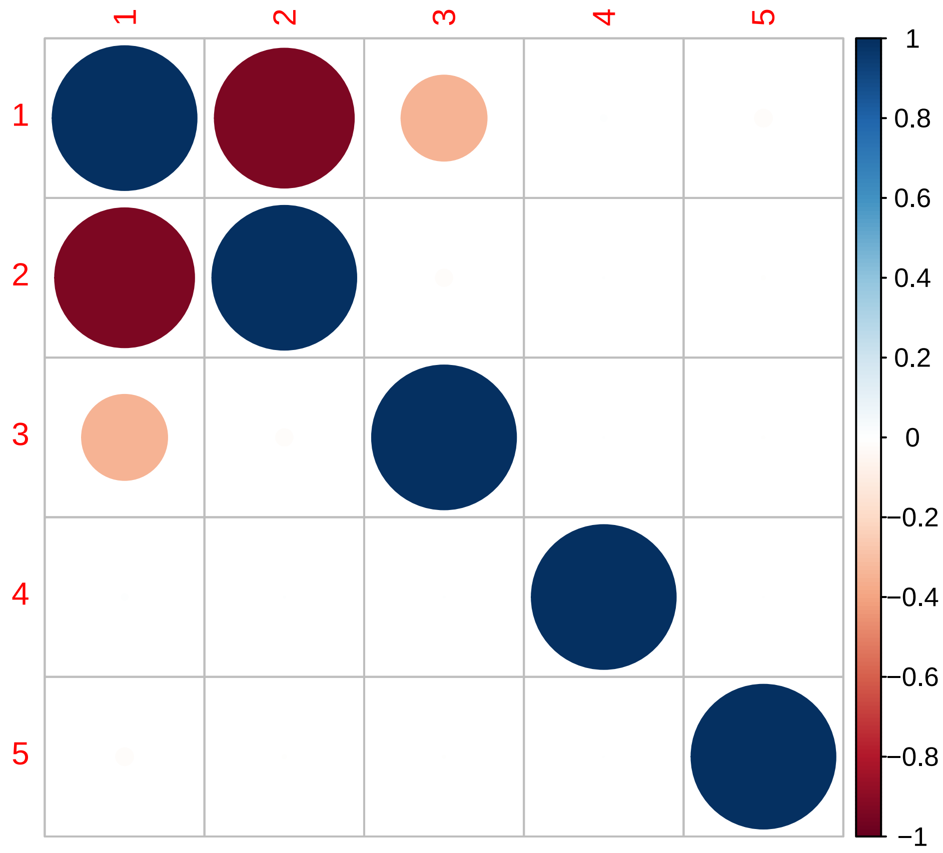
```
## [1] 3
## [1] "FCAT 2019-1 D"
```

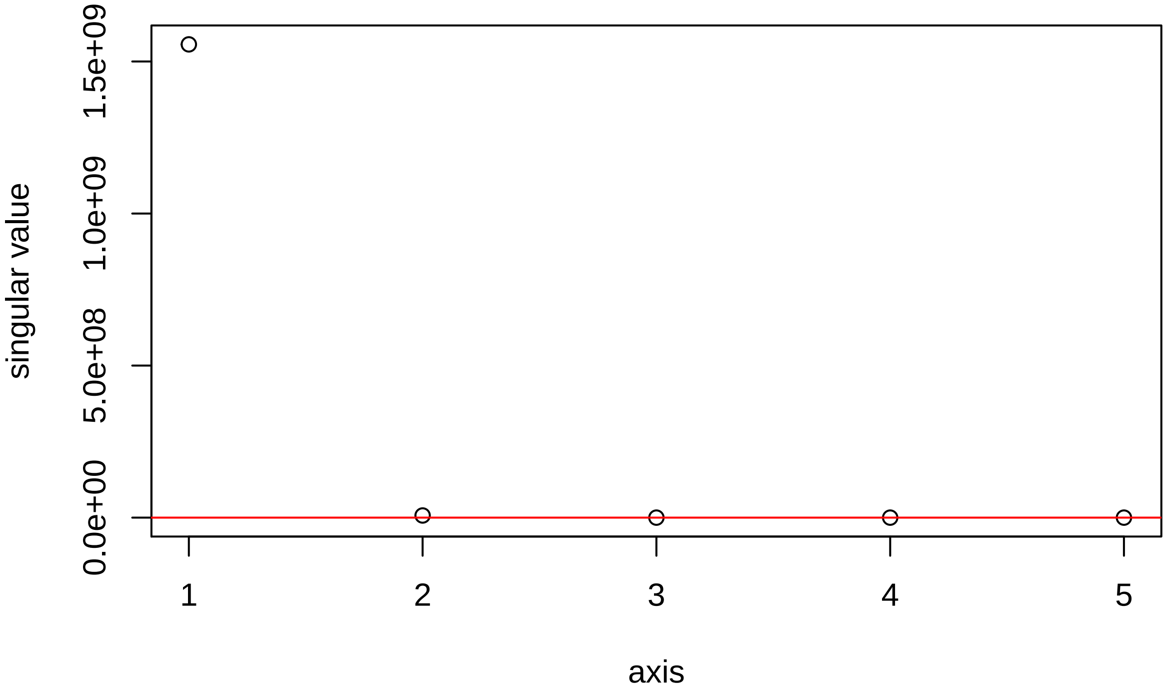

1
2
3
4
5
1
2
3
4
5
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

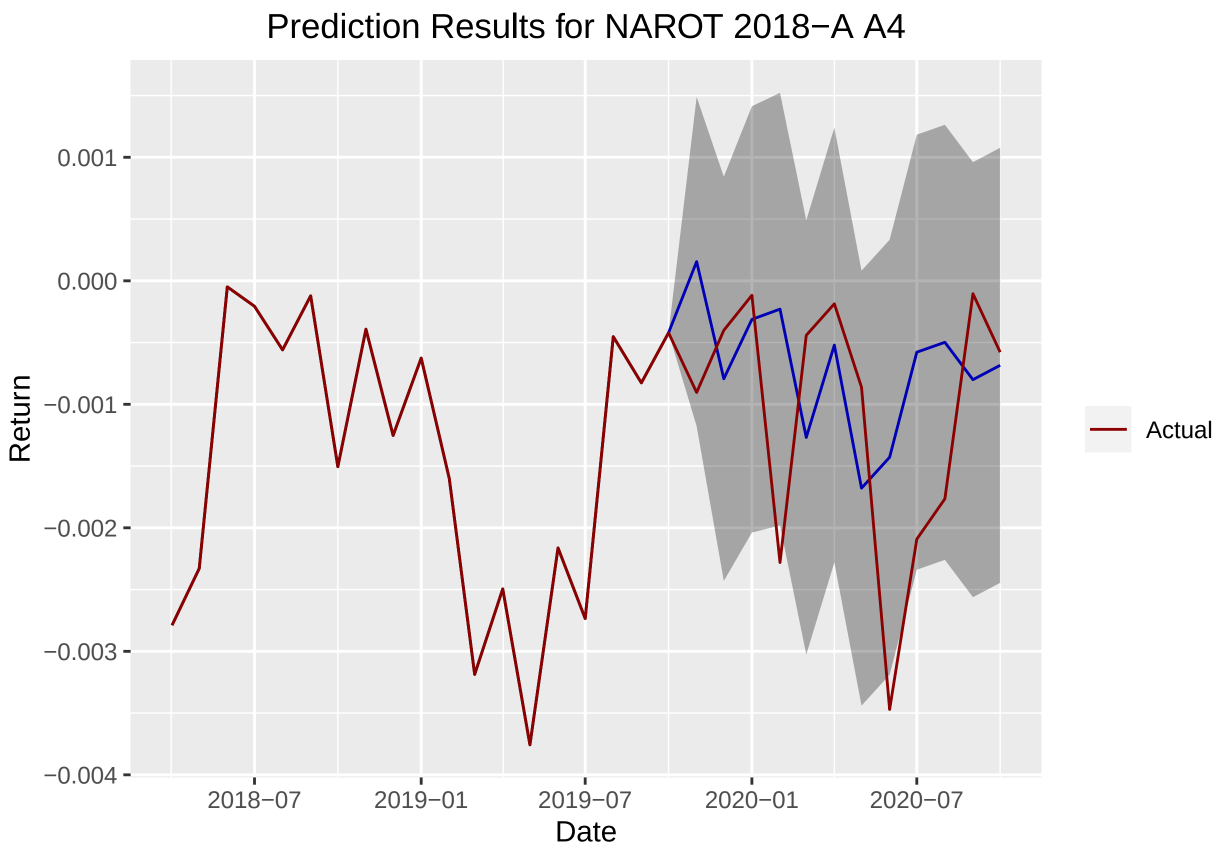

singular value
3.0e+08
1.5e+08
0.0e+00
1
2
3
4
5
axis



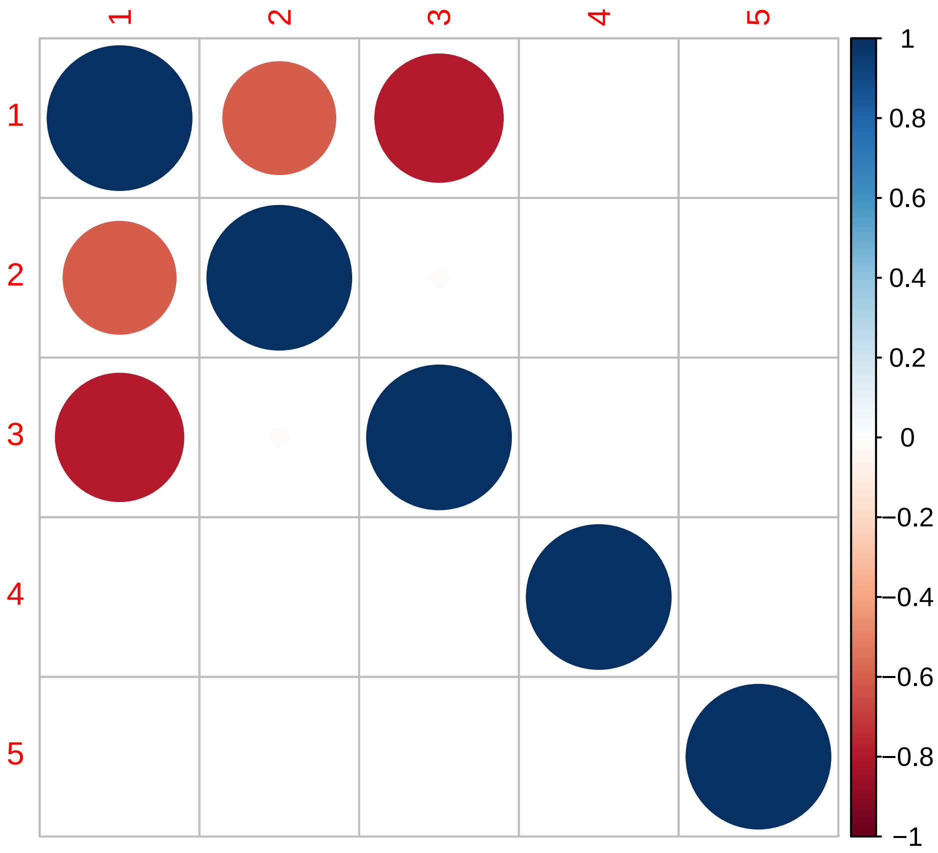
```
## [1] 4
## [1] "NAROT 2018-A A4"
```

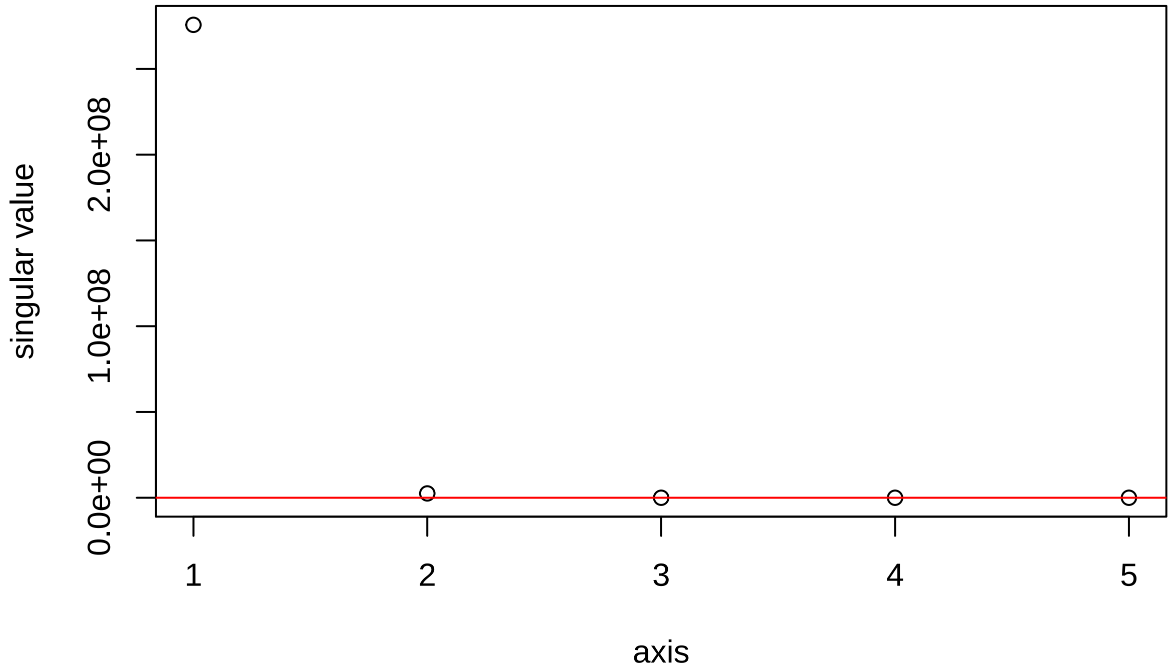

1
2
3
4
5
1
2
3
4
5
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

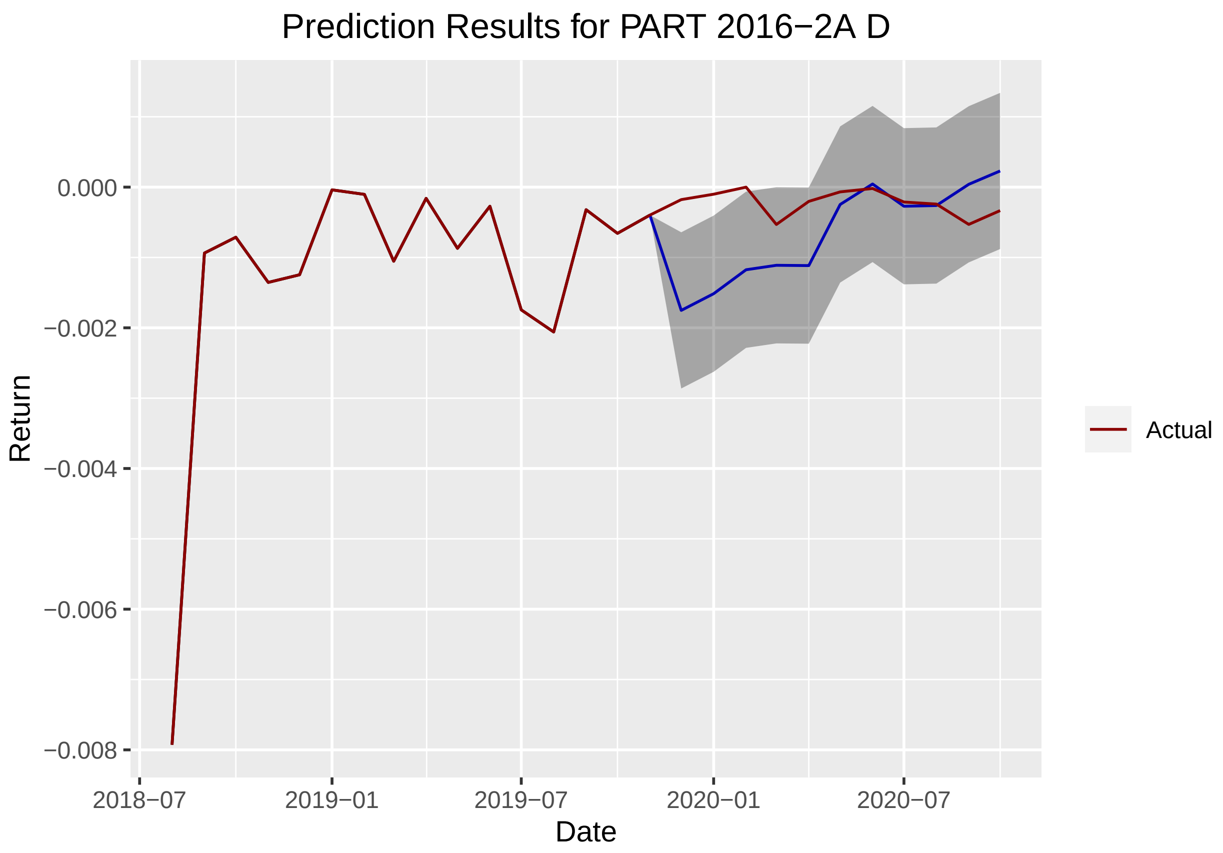

singular value
0.0e+00
5.0e+08
1.0e+09
1.5e+09
1
2
3
4
5
axis



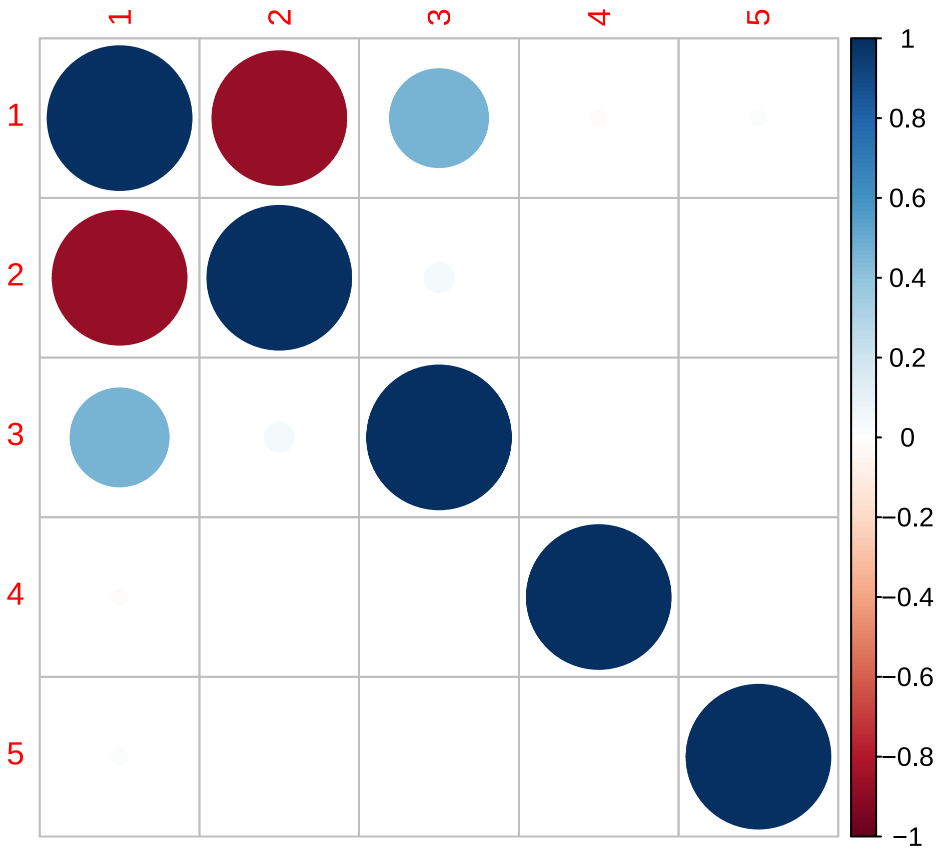
```
## [1] 5
## [1] "PART 2016-2A D"
```

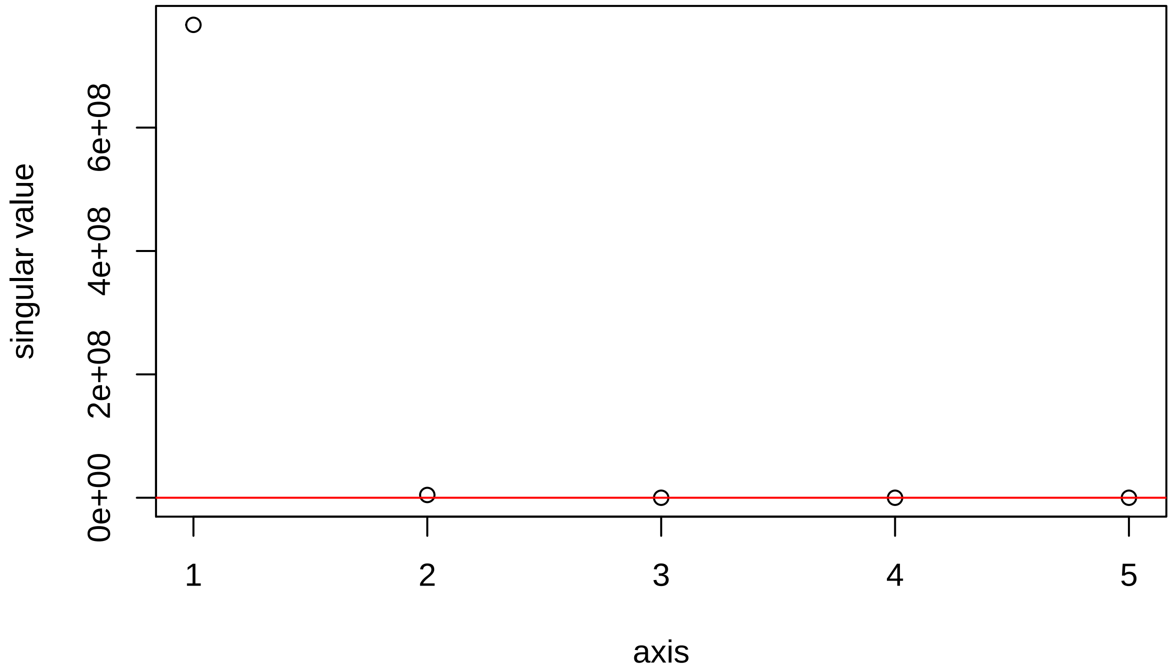

1
2
3
4
5
1
2
3
4
5
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

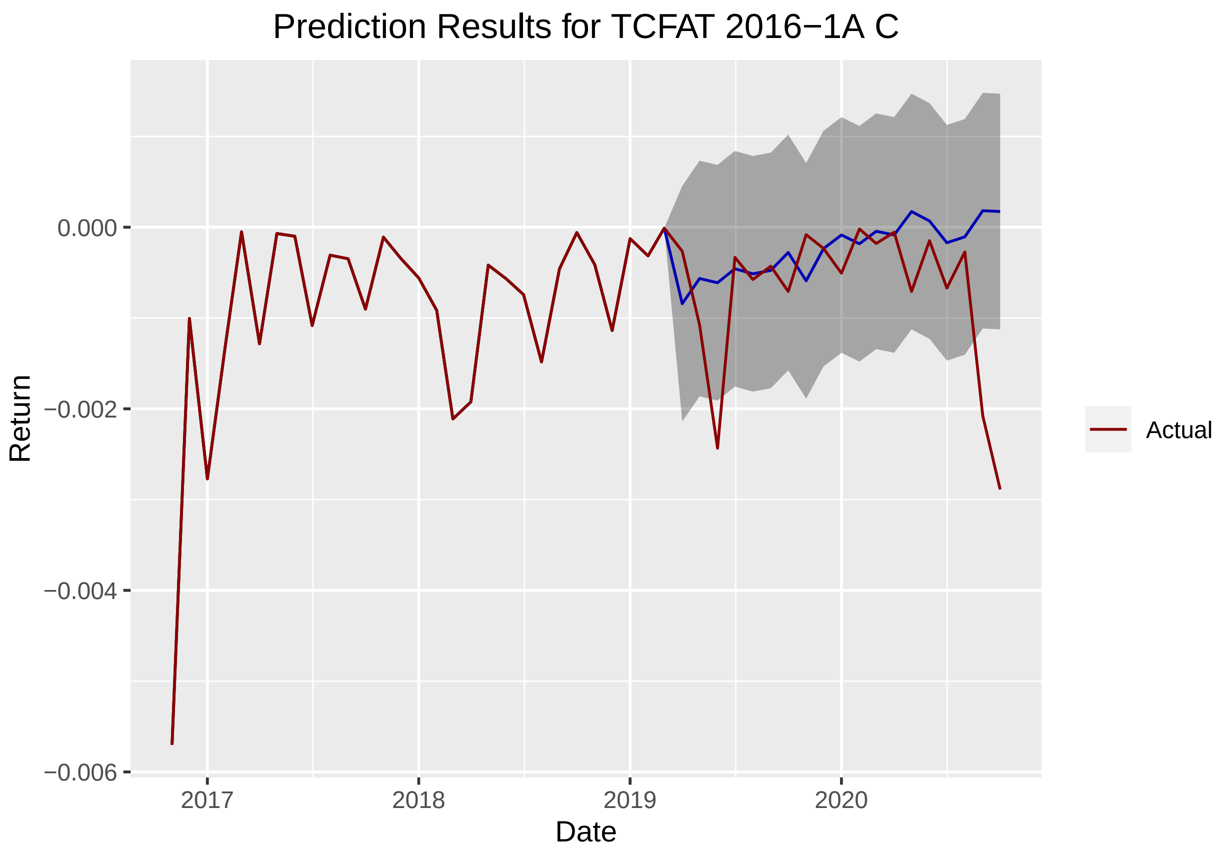

singular value
0.0e+00
1.0e+08
2.0e+08
1
2
3
4
5
axis



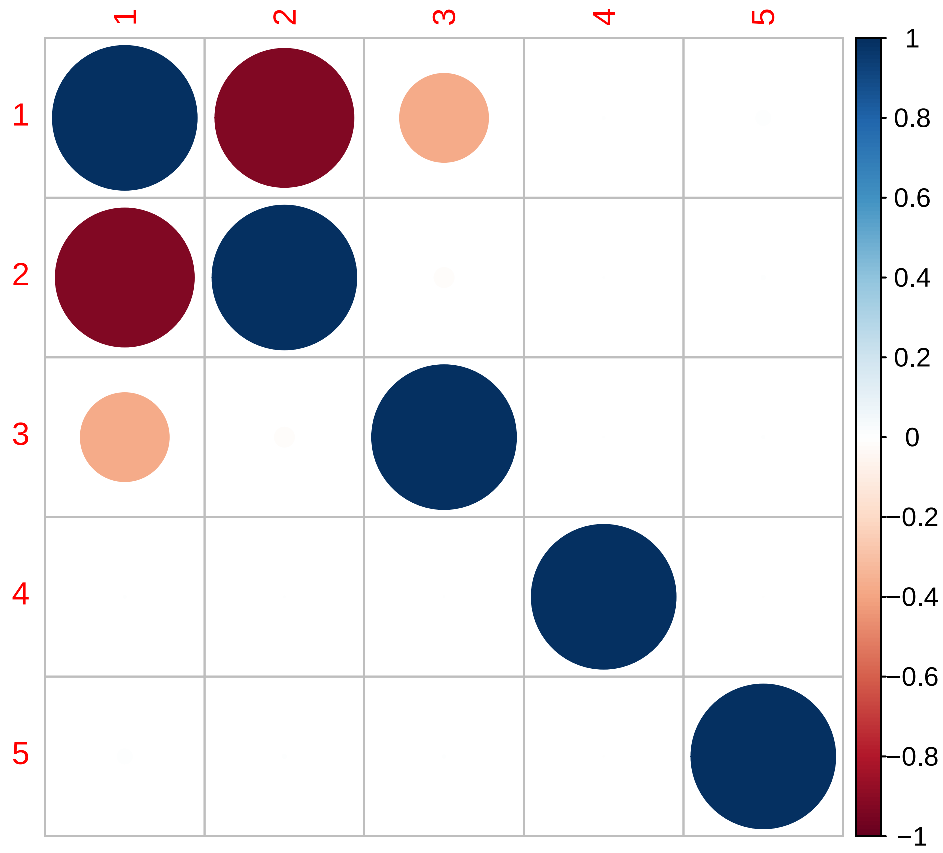
```
## [1] 6
## [1] "TCFAT 2016-1A C"
```

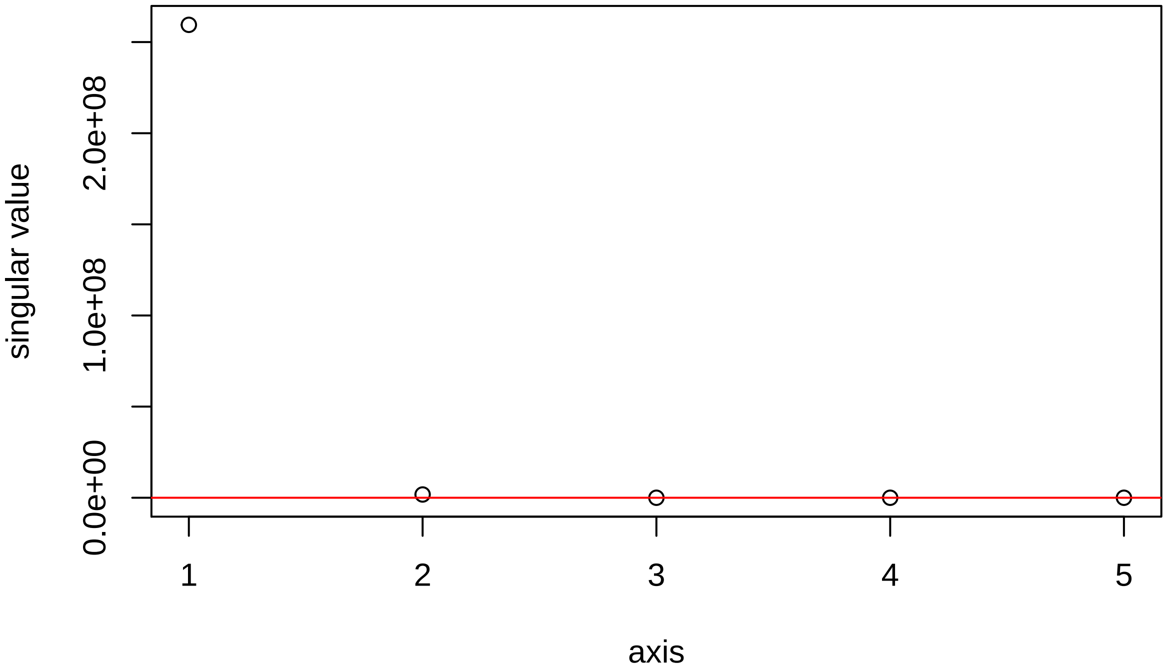

1
2
3
4
5
1
2
3
4
5
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

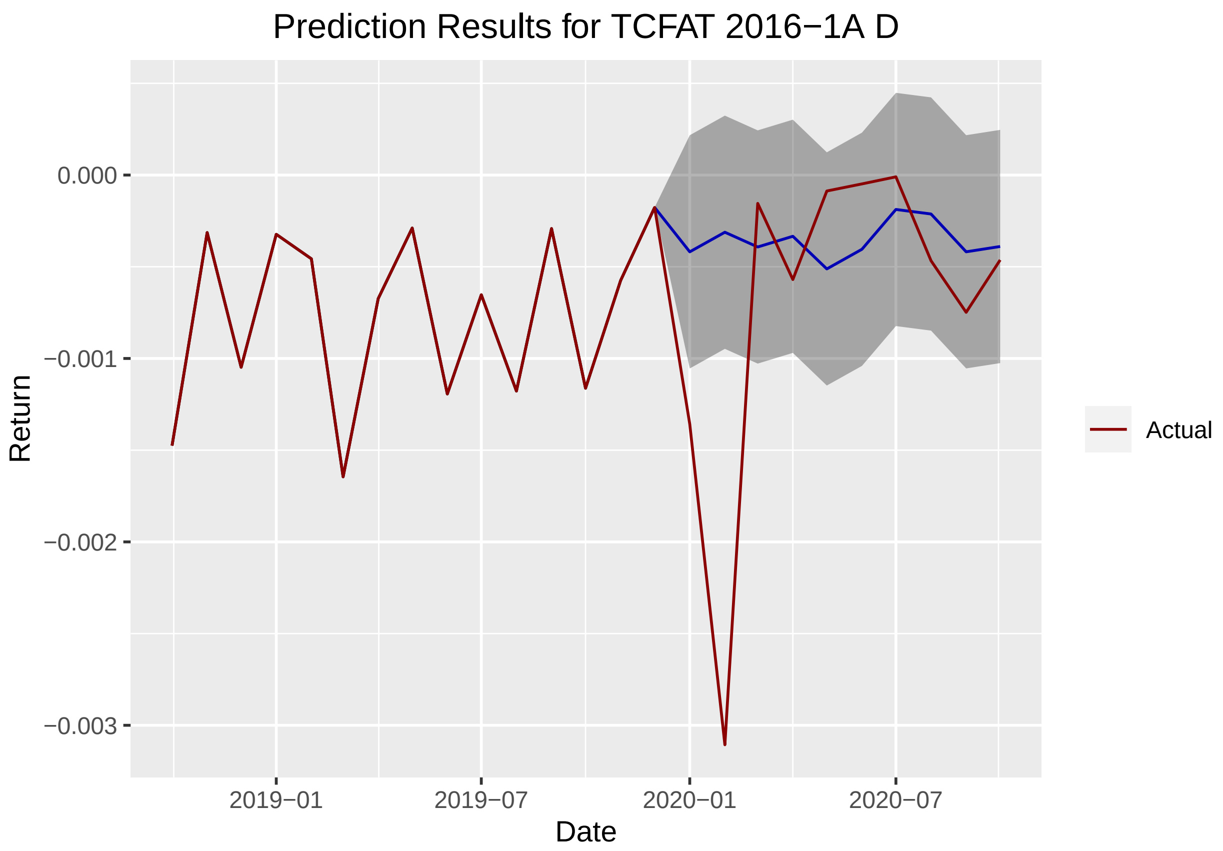

singular value
0e+00
2e+08
4e+08
6e+08
1
2
3
4
5
axis



```
## [1] 7
## [1] "TCFAT 2016-1A D"
```


1
2
3
4
5
1
2
3
4
5
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1


singular value
0.0e+00
1.0e+08
2.0e+08
1
2
3
4
5
axis



```r
for(i in consumer_models) {
  print(summary(i))
}
```

```
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       intercept   xreg1   xreg2   xreg3  xreg4   xreg5    xreg6
##         -0.0200  -4e-04  -4e-04  -3e-04  0e+00  0.0018  -0.0018
## s.e.     0.0051   3e-04   3e-04   4e-04  5e-04  0.0010   0.0071
##
## sigma^2 estimated as 4.925e-07:  log likelihood=103.84
## AIC=-191.68   AICc=-173.68   BIC=-185.01
##
## Training set error measures:
##                        ME         RMSE          MAE      MPE     MAPE      MASE
## Training set 9.167497e-21 0.0005382541 0.0003701738 298.0501 981.3597 0.3933817
##                    ACF1
## Training set -0.3213731
##                        ME         RMSE          MAE      MPE     MAPE      MASE
## Training set 9.167497e-21 0.0005382541 0.0003701738 298.0501 981.3597 0.3933817
##                    ACF1
## Training set -0.3213731
## Series: y_train
## Regression with ARIMA(0,0,0) errors
```

```
##
## Coefficients:
##       xreg1   xreg2   xreg3   xreg4   xreg5   xreg6   xreg7   xreg8
##       1e-04  -3e-04  -2e-04  -8e-04  0.0002  0.0001  0.0003  0.0049
## s.e.  1e-04   2e-04   3e-04   6e-04  0.0012  0.0022  0.0033  0.0039
##
## sigma^2 estimated as 6.184e-06:  log likelihood=306.4
## AIC=-594.8   AICc=-591.59   BIC=-575.09
##
## Training set error measures:
##                         ME        RMSE         MAE       MPE    MAPE     MASE
## Training set -1.466781e-05 0.002331228 0.001628778 -396.8533 429.672 0.876999
##                    ACF1
## Training set 0.06176553
##                         ME        RMSE         MAE       MPE    MAPE     MASE
## Training set -1.466781e-05 0.002331228 0.001628778 -396.8533 429.672 0.876999
##                    ACF1
## Training set 0.06176553
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       xreg1  xreg2   xreg3  xreg4   xreg5   xreg6   xreg7   xreg8
##       0e+00  1e-04  -6e-04  4e-04  -3e-04  0.0017  0.0007  0.0005
## s.e.  2e-04  2e-04   2e-04  3e-04   4e-04  0.0007  0.0019  0.0023
##
## sigma^2 estimated as 3.758e-07:  log likelihood=267.46
## AIC=-516.91   AICc=-511.62   BIC=-500.85
##
## Training set error measures:
##                        ME         RMSE          MAE       MPE     MAPE
## Training set 1.543034e-06 0.0005544851 0.0004564073 -256.0594 290.2479
##                   MASE      ACF1
## Training set 0.6743832 0.1051752
##                        ME         RMSE          MAE       MPE     MAPE
## Training set 1.543034e-06 0.0005544851 0.0004564073 -256.0594 290.2479
##                   MASE      ACF1
## Training set 0.6743832 0.1051752
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       intercept    xreg1   xreg2  xreg3    xreg4   xreg5    xreg6    xreg7
##         -0.0993  -0.0025  0.0026  1e-03  -0.0044  0.0087  -0.0082  -0.0088
## s.e.     0.0325   0.0009  0.0013  6e-04   0.0020  0.0054   0.0072   0.0089
##         xreg8
##       -0.0234
## s.e.   0.0169
##
## sigma^2 estimated as 1.883e-05:  log likelihood=274.25
## AIC=-528.49   AICc=-524.56   BIC=-506.44
##
## Training set error measures:
##                        ME        RMSE         MAE MPE MAPE      MASE       ACF1
## Training set 1.334107e-17 0.004037402 0.002353218 Inf  Inf 0.9834184 -0.1154649
##                         ME        RMSE         MAE MPE MAPE      MASE       ACF1
## Training set 1.334107e-17 0.004037402 0.002353218 Inf  Inf 0.9834184 -0.1154649
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       xreg1  xreg2    xreg3    xreg4    xreg5    xreg6
##       0e+00  6e-04  -0.0005  -0.0065  -0.0115  -0.0007
## s.e.  3e-04  5e-04   0.0014   0.0028   0.0064   0.0174
##
## sigma^2 estimated as 2.271e-06:  log likelihood=55.37
## AIC=-96.74   AICc=-40.74   BIC=-94.62
##
## Training set error measures:
##                          ME         RMSE          MAE       MPE     MAPE MASE
## Training set -2.153586e-06 0.0009530809 0.0007169266 -25.13111 118.4964  NaN
##                     ACF1
## Training set -0.01077045
##                          ME         RMSE          MAE       MPE     MAPE MASE
## Training set -2.153586e-06 0.0009530809 0.0007169266 -25.13111 118.4964  NaN
##                     ACF1
## Training set -0.01077045
## Series: y_train
## Regression with ARIMA(1,0,0) errors
##
## Coefficients:
##           ar1  intercept   xreg1  xreg2  xreg3   xreg4    xreg5   xreg6
##       -0.9529     0.0653  0.0019  3e-04  4e-04  0.0052  -0.0040  0.0088
## s.e.   0.0603     0.0256  0.0007  5e-04  4e-04  0.0009   0.0018  0.0024
##         xreg7   xreg8
##       -0.0007  0.0066
## s.e.   0.0042  0.0128
##
## sigma^2 estimated as 1.294e-06:  log likelihood=107.75
## AIC=-193.5   AICc=-155.78   BIC=-183.11
##
## Training set error measures:
##                          ME         RMSE          MAE      MPE     MAPE
## Training set -2.292951e-05 0.0007827791 0.0005789557 1270.405 1464.137
##                   MASE       ACF1
## Training set 0.3052937 -0.3519046
##                          ME         RMSE          MAE      MPE     MAPE
## Training set -2.292951e-05 0.0007827791 0.0005789557 1270.405 1464.137
##                   MASE       ACF1
## Training set 0.3052937 -0.3519046
```

```r
for(i in auto_models) {
  print(summary(i))
}
```

```
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
```

```
##       intercept  xreg1  xreg2   xreg3
##          0.0249  0e+00  0e+00  0.0005
## s.e.     0.0954  2e-04  2e-04  0.0018
##
## sigma^2 estimated as 2.114e-06:  log likelihood=99.42
## AIC=-188.85   AICc=-184.23   BIC=-184.12
##
## Training set error measures:
##                         ME       RMSE         MAE       MPE     MAPE      MASE
## Training set -2.567736e-19 0.00129179 0.001144413 -241.2418 278.1757 0.5322414
##                   ACF1
## Training set 0.1590313
##                         ME       RMSE         MAE       MPE     MAPE      MASE
## Training set -2.567736e-19 0.00129179 0.001144413 -241.2418 278.1757 0.5322414
##                   ACF1
## Training set 0.1590313
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       intercept  xreg1  xreg2   xreg3
##          0.1028  0e+00  0e+00  0.0016
## s.e.     0.2483  3e-04  3e-04  0.0047
##
## sigma^2 estimated as 6.082e-06:  log likelihood=48.42
## AIC=-86.83   AICc=-71.83   BIC=-85.32
##
## Training set error measures:
##                        ME        RMSE         MAE       MPE     MAPE MASE
## Training set 3.446424e-15 0.001910286 0.001462028 -196.9166 263.6862  NaN
##                    ACF1
## Training set -0.7077327
##                        ME        RMSE         MAE       MPE     MAPE MASE
## Training set 3.446424e-15 0.001910286 0.001462028 -196.9166 263.6862  NaN
##                    ACF1
## Training set -0.7077327
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       xreg1  xreg2  xreg3
##       0e+00  0e+00  3e-04
## s.e.  3e-04  3e-04  4e-04
##
## sigma^2 estimated as 9.655e-06:  log likelihood=49.66
## AIC=-91.31   AICc=-84.65   BIC=-89.72
##
## Training set error measures:
##                         ME        RMSE         MAE       MPE     MAPE MASE
## Training set -1.415801e-06 0.002649901 0.002169833 -191.7948 223.6935  NaN
##                    ACF1
## Training set -0.0776497
##                         ME        RMSE         MAE       MPE     MAPE MASE
## Training set -1.415801e-06 0.002649901 0.002169833 -191.7948 223.6935  NaN
##                     ACF1
## Training set -0.0776497
## Series: y_train
## Regression with ARIMA(2,0,0) errors
##
## Coefficients:
##          ar1      ar2  xreg1  xreg2  xreg3
##       0.7126  -0.0974      0      0  1e-04
## s.e.  0.4940   0.4450      0      0  1e-03
##
## sigma^2 estimated as 1.082e-06:  log likelihood=106.16
## AIC=-200.31   AICc=-193.31   BIC=-194.64
##
## Training set error measures:
##                        ME         RMSE          MAE      MPE    MAPE      MASE
## Training set 5.992795e-05 0.0008929356 0.0006902449 -268.244 290.982 0.5848482
##                      ACF1
## Training set -0.06289264
##                        ME         RMSE          MAE      MPE    MAPE      MASE
## Training set 5.992795e-05 0.0008929356 0.0006902449 -268.244 290.982 0.5848482
##                      ACF1
## Training set -0.06289264
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       xreg1  xreg2  xreg3
##       0e+00  0e+00  1e-04
## s.e.  3e-04  2e-04  3e-04
##
## sigma^2 estimated as 7.507e-07:  log likelihood=91.78
## AIC=-175.55   AICc=-171.92   BIC=-172.46
##
## Training set error measures:
##                        ME         RMSE         MAE       MPE     MAPE      MASE
## Training set 3.514821e-07 0.0007809674 0.000637104 -237.8013 302.4408 0.3398928
##                      ACF1
## Training set -0.07924416
##                        ME         RMSE         MAE       MPE     MAPE      MASE
## Training set 3.514821e-07 0.0007809674 0.000637104 -237.8013 302.4408 0.3398928
##                      ACF1
## Training set -0.07924416
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       xreg1  xreg2  xreg3
##       0e+00  0e+00  0e+00
## s.e.  2e-04  2e-04  2e-04
##
## sigma^2 estimated as 1.026e-06:  log likelihood=160.39
## AIC=-312.78   AICc=-311.11   BIC=-307.31
##
## Training set error measures:
```

```
##                         ME        RMSE          MAE       MPE     MAPE
## Training set -3.503354e-07 0.000958992 0.0006737066 -511.2406 545.1092
##                   MASE        ACF1
## Training set 0.6002386 -0.05605915
```

```
##                         ME        RMSE          MAE       MPE     MAPE
## Training set -3.503354e-07 0.000958992 0.0006737066 -511.2406 545.1092
##                   MASE        ACF1
## Training set 0.6002386 -0.05605915
```

```
## Series: y_train
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       xreg1  xreg2  xreg3
##       0e+00  0e+00  0e+00
## s.e.  3e-04  3e-04  3e-04
##
## sigma^2 estimated as 2.462e-07:  log likelihood=94.52
## AIC=-181.04   AICc=-177.04   BIC=-178.21
##
## Training set error measures:
##                         ME        RMSE         MAE       MPE     MAPE      MASE
## Training set -1.835183e-07 0.000443778 0.000387465 -53.40996 83.77371 0.8040592
##                   ACF1
## Training set -0.329438
```

```
##                         ME        RMSE         MAE       MPE     MAPE      MASE
## Training set -1.835183e-07 0.000443778 0.000387465 -53.40996 83.77371 0.8040592
##                   ACF1
## Training set -0.329438
```