conditional independency of US stocks given US dollar index

Mohammad Nabeel Khan

2023-03-02

```r
# functions that we need
# install.packages('bnlearn')
library(bnlearn)
# define a function that takes a dataframe of returns, a type of test and a
# fixed variable it returns a dataframe with p-values and correlations for every pair of stocks
given the fixed variable
ci_test_pairs <- function(df, test, fixed) {
  # get the names of the columns that are not the fixed variable
  stocks <- names(df)[names(df) != fixed]
  # create an empty dataframe to store the results
  results <- data.frame()
  # loop over all pairs of stocks
  for (i in 1:(length(stocks) - 1)) {
    for (j in (i + 1):length(stocks)) {
      # get the names of the pair
      x <- stocks[i]
      y <- stocks[j]
      # perform the conditional independence test given the fixed variable
      test_result <- ci.test(x, y, fixed, data = df, test = test)
      # extract the p-value and the correlation from the test result
      p_value <- test_result$p.value
      correlation <- cor(df[[x]], df[[y]])
      # append a row to the results dataframe with the pair names, p-value and correlation
      results <- rbind(results, data.frame(x = x, y = y, p_value = p_value, correlation = correl
ation))
    }
  }
  # return the results dataframe
  return(results)
}

correlation_threshold <- function(cor_matrix, threshold) {

  # Get the lower triangle of the correlation matrix
  cor_lower <- cor_matrix[lower.tri(cor_matrix)]

  # Get the indices of the lower triangle of the correlation matrix
  cor_indices <- which(lower.tri(cor_matrix), arr.ind = TRUE)

  # Create a data frame with the pairs of variables and their correlations
  df <- data.frame(x = rownames(cor_matrix)[cor_indices[, 1]],
                   y = colnames(cor_matrix)[cor_indices[, 2]],
                   edge = ifelse(abs(cor_lower) >= threshold, 1, 0))

  return(df)
}

p_value_eval<- function(x,thresh){
  if(x<thresh){
    ret=1
  }
  else{
    ret=0

  }
  return(ret)
}


grapher<-function(example1,compl){
library(dplyr)
library(igraph)
#filtered only the edges that contain 1
edges1 <- filter(example1, edge == "1")
edges1
#plotted the graph

g <- graph_from_data_frame(edges1, directed = FALSE)

if(compl==TRUE){
  g<- complementer(g)
}

plot(g, vertex.label = V(g)$names, edge.label = E(g)$weight)
}
```

```r
# reading the data and getting rid of HSI

df<- read.csv("https://raw.githubusercontent.com/knabeel77/conditional-independency-of-US-stocks
-given-US-dollar-index/main/returns_data.csv")

df<- df[,-12]


#different slices corresponding to federal reserve interest rates hiking/cutting/pausing

df1<- df[ df$Date> "2000-11-01"  &   df$Date < "2001-12-31",-1 ]

df2<- df[ df$Date> "2002-01-01"  &   df$Date < "2002-10-31",-1 ]

df3<- df[ df$Date> "2002-11-01"  &   df$Date < "2004-06-31",-1 ]

df4<- df[ df$Date> "2004-07-01"  &   df$Date < "2006-08-31",-1 ]

df5<- df[ df$Date> "2006-08-31"  &   df$Date < "2007-08-01",-1 ]

df6<- df[ df$Date> "2007-08-01"  &   df$Date < "2008-05-01",-1 ]

df7<- df[ df$Date> "2008-05-01"  &   df$Date < "2008-09-31",-1 ]

df8<- df[ df$Date> "2008-09-31"  &   df$Date < "2009-01-01",-1 ]

df9<- df[ df$Date> "2009-01-01"  &   df$Date < "2015-12-31",-1 ]

df10<- df[ df$Date> "2016-01-01"  &   df$Date < "2019-07-31",-1 ]

df11<- df[ df$Date> "2019-07-31"  &   df$Date < "2020-05-01",-1 ]

df12<- df[ df$Date> "2020-05-01"  &   df$Date < "2022-02-01",-1 ]

dff = list(df1,df2,df3,df4,df5,df6,df7,df8,df9,df10,df11,df12)

results=list()

for (i in 1:12){
  res<-ci_test_pairs(df = dff[[i]],fixed="DXY")
  results[[i]]<- res

}

# the result of the test is available as a list, each element corresponds to a time-slice
```

```r
df<- read.csv("https://raw.githubusercontent.com/knabeel77/conditional-independency-of-US-stocks
-given-US-dollar-index/main/returns_data_classified.csv")

df<- df[,-12]


# different slices corresponding to federal reserve interest rates hiking/cutting/pausing

df1<- df[ df$Date> "2000-11-01"  &   df$Date < "2001-12-31",-1 ]

df2<- df[ df$Date> "2002-01-01"  &   df$Date < "2002-10-31",-1 ]

df3<- df[ df$Date> "2002-11-01"  &   df$Date < "2004-06-31",-1 ]

df4<- df[ df$Date> "2004-07-01"  &   df$Date < "2006-08-31",-1 ]

df5<- df[ df$Date> "2006-08-31"  &   df$Date < "2007-08-01",-1 ]

df6<- df[ df$Date> "2007-08-01"  &   df$Date < "2008-05-01",-1 ]

df7<- df[ df$Date> "2008-05-01"  &   df$Date < "2008-09-31",-1 ]

df8<- df[ df$Date> "2008-09-31"  &   df$Date < "2009-01-01",-1 ]

df9<- df[ df$Date> "2009-01-01"  &   df$Date < "2015-12-31",-1 ]

df10<- df[ df$Date> "2016-01-01"  &   df$Date < "2019-07-31",-1 ]

df11<- df[ df$Date> "2019-07-31"  &   df$Date < "2020-05-01",-1 ]

df12<- df[ df$Date> "2020-05-01"  &   df$Date < "2022-02-01",-1 ]

dff_count = list(df1,df2,df3,df4,df5,df6,df7,df8,df9,df10,df11,df12)

results_count=list()

for (i in 1:12){
  res_count<-ci_test_pairs(df = dff_count[[i]],fixed="DXY")
  results_count[[i]]<- res_count

}
```

```
# plotting the graphs for inverse of correlation matrix
inver<- list()
for(i in 1:12){
  inver[[i]]<- solve(cor(dff[[i]]))
}

for(i in 1:12){
   set.seed(123)

   grapher(correlation_threshold(inver[[i]],threshold = 0.01),compl = TRUE)
   title(paste("graphical model for the inverse correlation matrix ", i, "Typographic Error.pdfh
slice"))

}
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
##
## Attaching package: 'igraph'
```

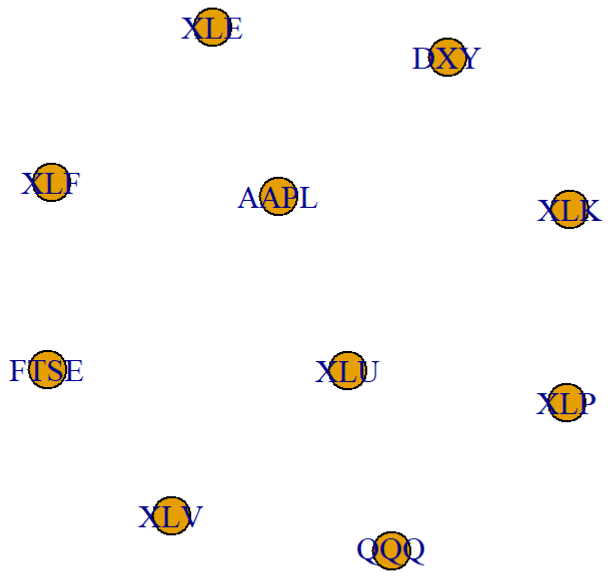
```
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
```

```
## The following objects are masked from 'package:bnlearn':
##
##     as.igraph, compare, degree, subgraph
```

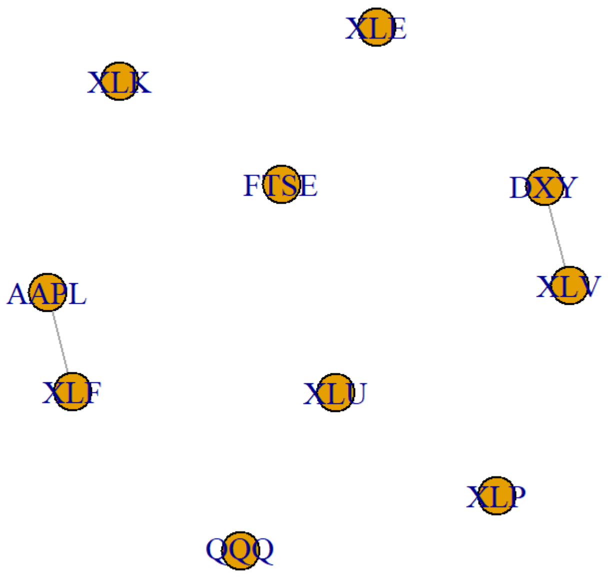
```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

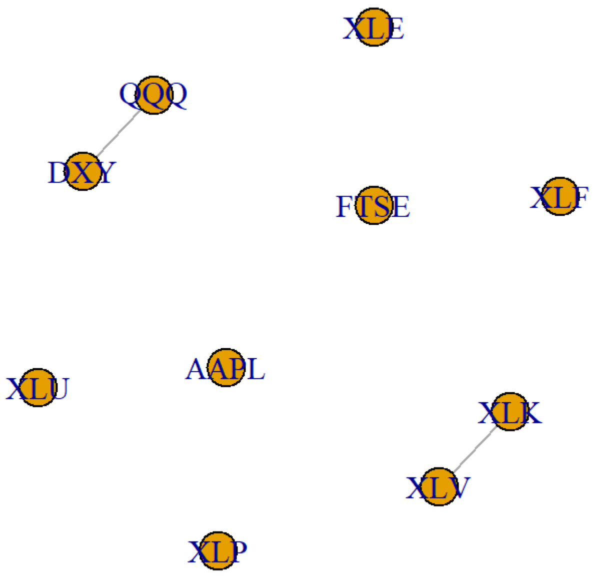
raphical model for the inverse correlation matrix 1 Typographic Error.pdfh



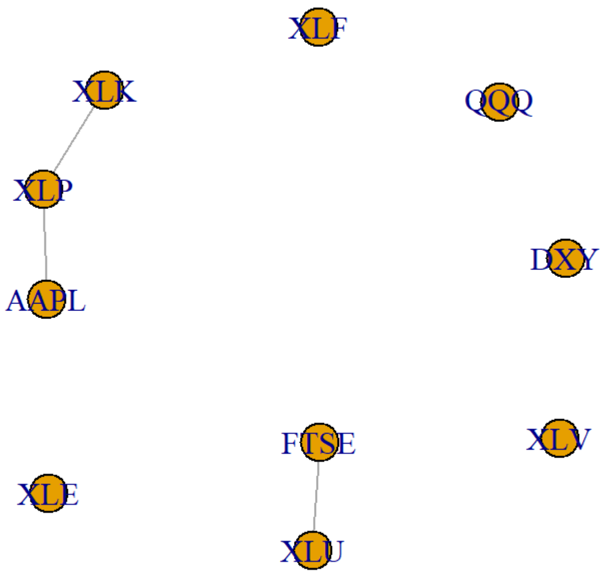
raphical model for the inverse correlation matrix 2 Typographic Error.pdfh



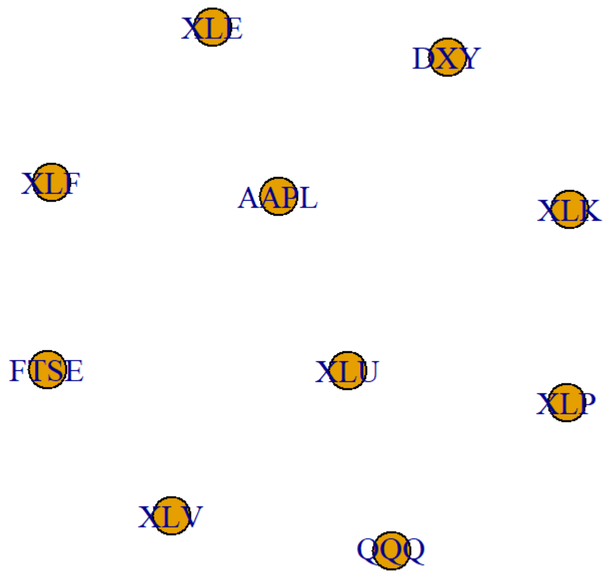
raphical model for the inverse correlation matrix 3 Typographic Error.pdfh



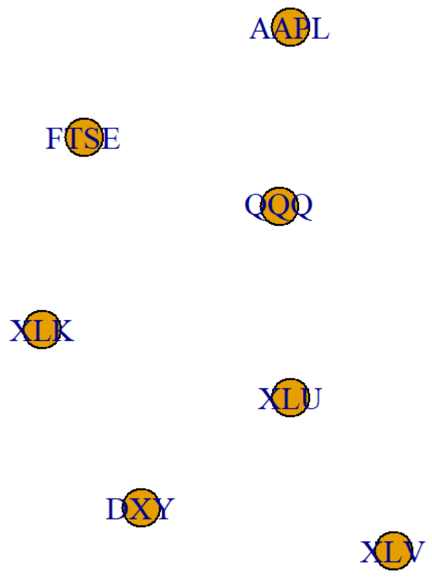
raphical model for the inverse correlation matrix 4 Typographic Error.pdfh



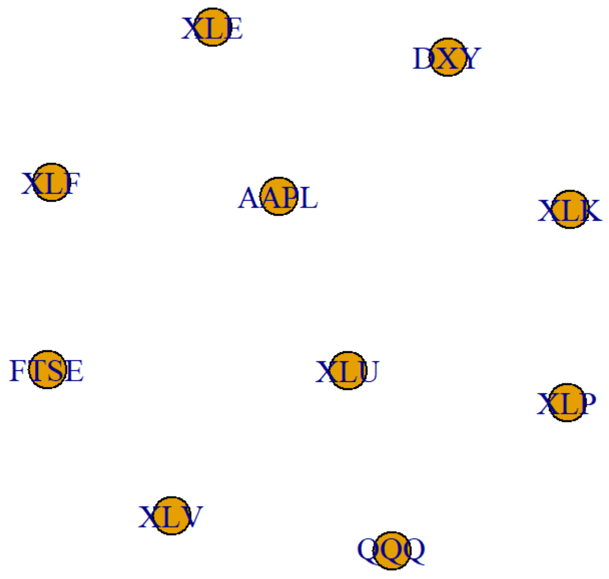
raphical model for the inverse correlation matrix 5 Typographic Error.pdfh



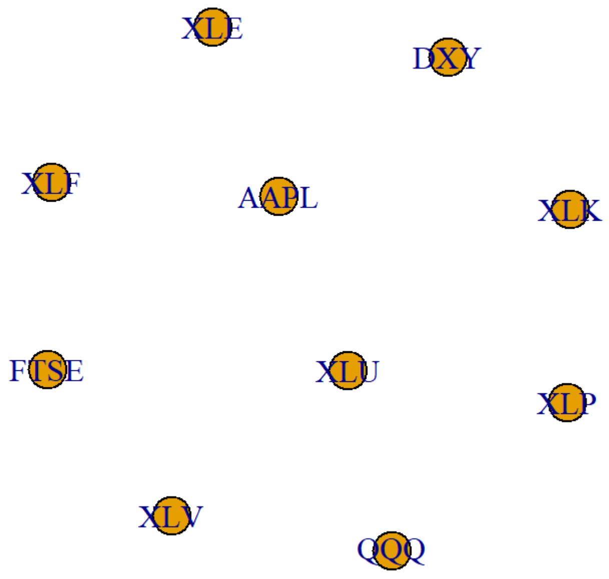
raphical model for the inverse correlation matrix 6 Typographic Error.pdfh



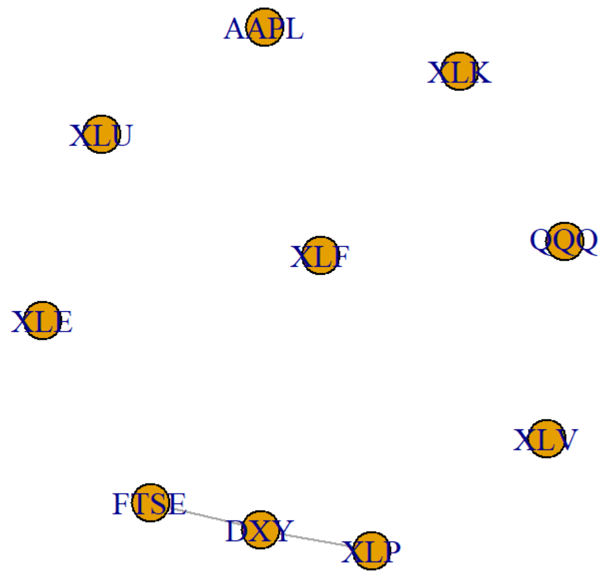
raphical model for the inverse correlation matrix 7 Typographic Error.pdfh



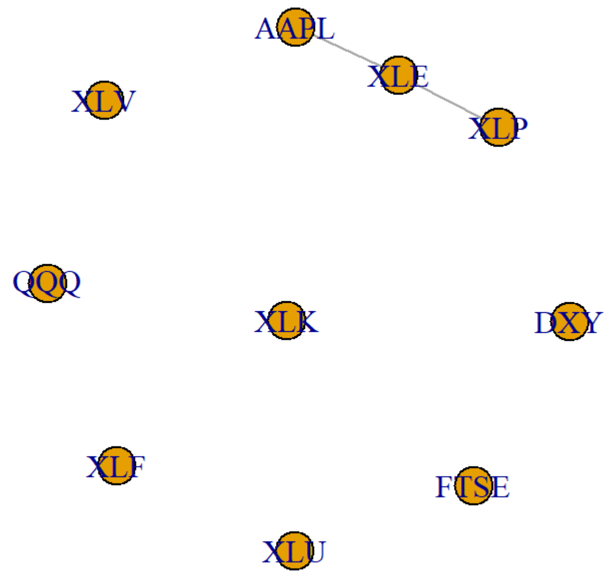
raphical model for the inverse correlation matrix 8 Typographic Error.pdfh



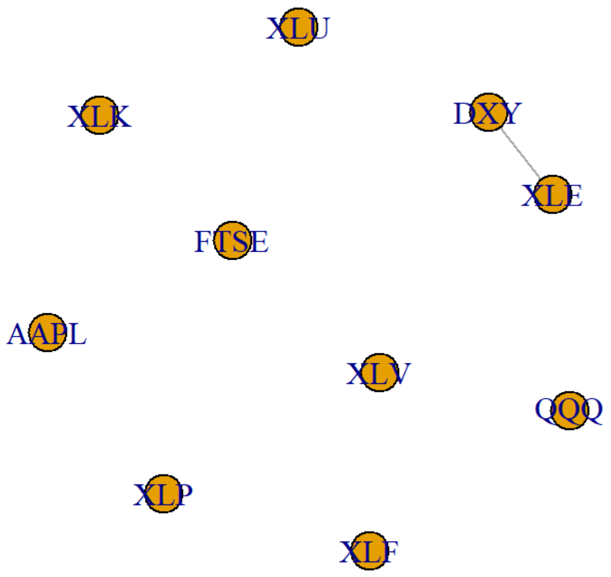
raphical model for the inverse correlation matrix 9 Typographic Error.pdfh



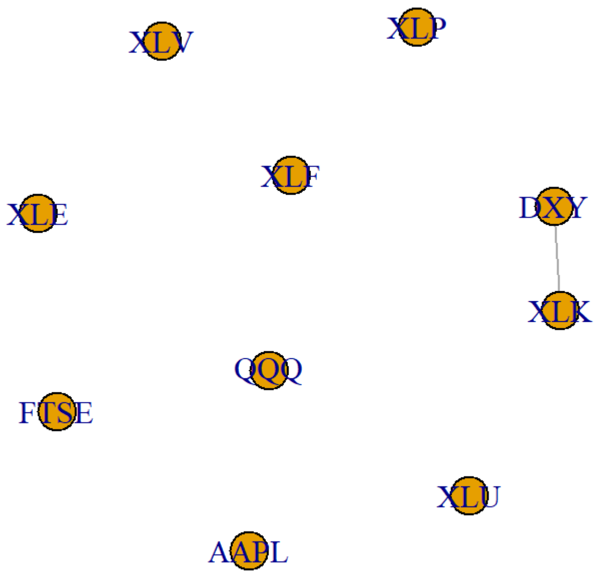
aphical model for the inverse correlation matrix 10 Typographic Error.pdfl



aphical model for the inverse correlation matrix 11 Typographic Error.pdf

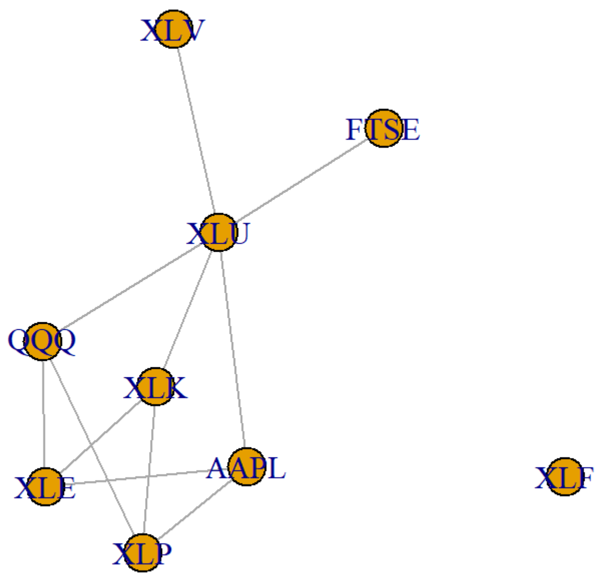


aphical model for the inverse correlation matrix 12 Typographic Error.pdf

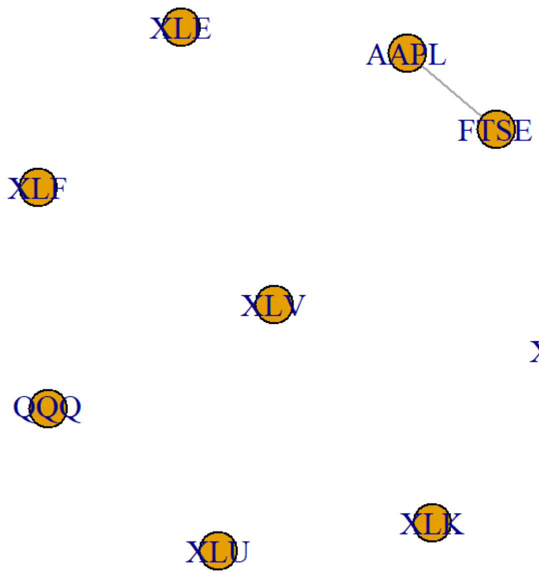


```r
# plotting the graph for conditional independence test on the continuous returns

graphss <- list()
for(i in 1:12){
  set.seed(123)
  l<- lapply(results[[i]]$p_value, p_value_eval, thresh=0.01)
  results[[i]]$edge<- as.integer(l)
  graphss[[i]] <- results[[i]][c("x","y","edge")]
  grapher(graphss[[i]],compl = TRUE)
  title(paste("Conditional independence test of continous returns for the " ,i, "th slice"))
}
```

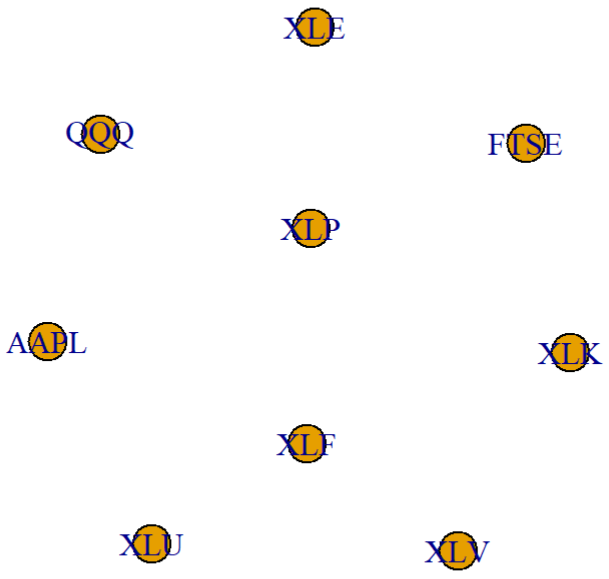
Conditional independence test of continous returns for the 1 th slice



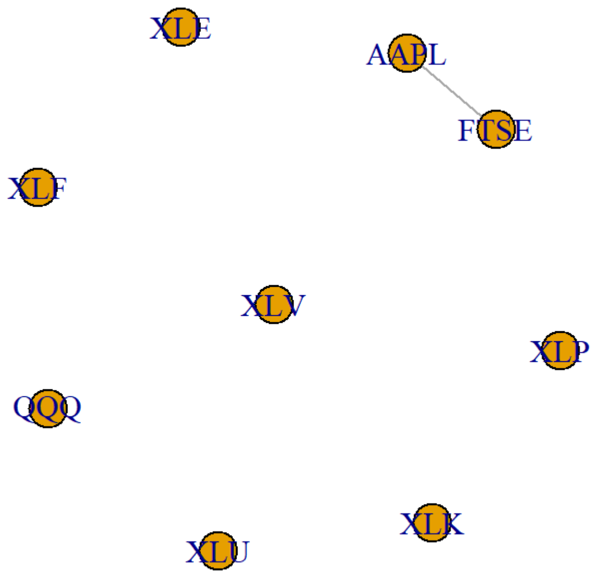
Conditional independence test of continous returns for the 2 th slice



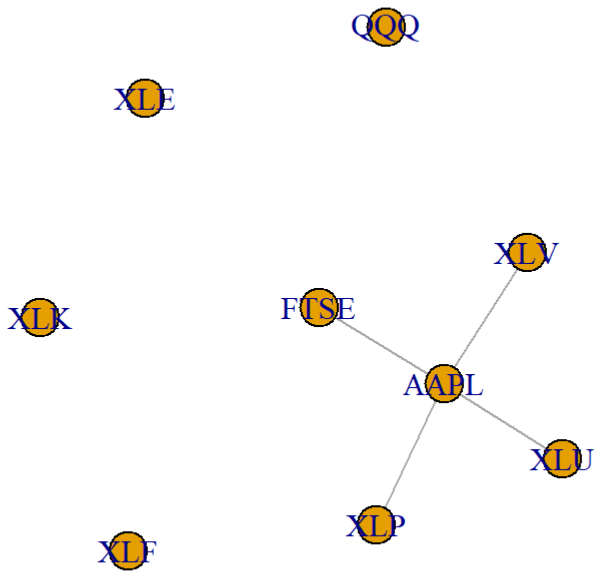
Conditional independence test of continous returns for the 3 th slice



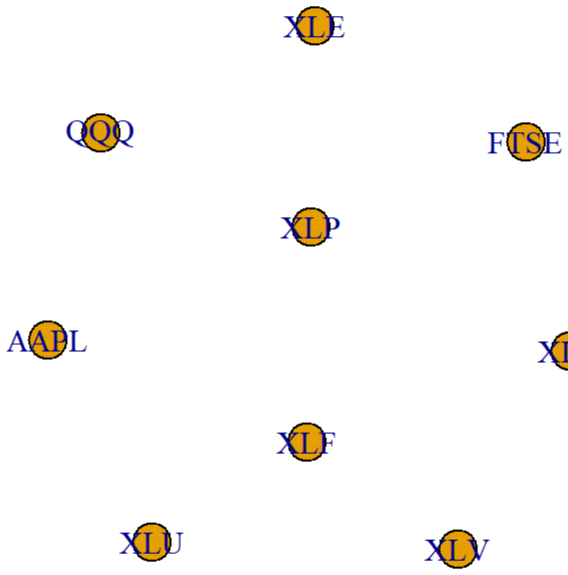
Conditional independence test of continous returns for the 4 th slice



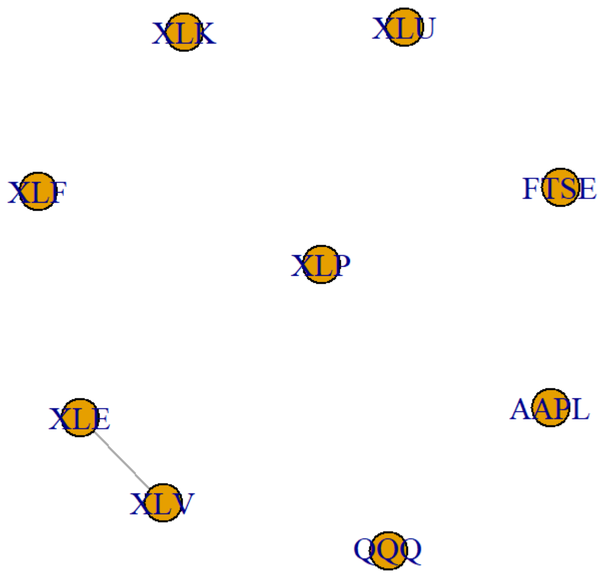
Conditional independence test of continous returns for the 5 th slice



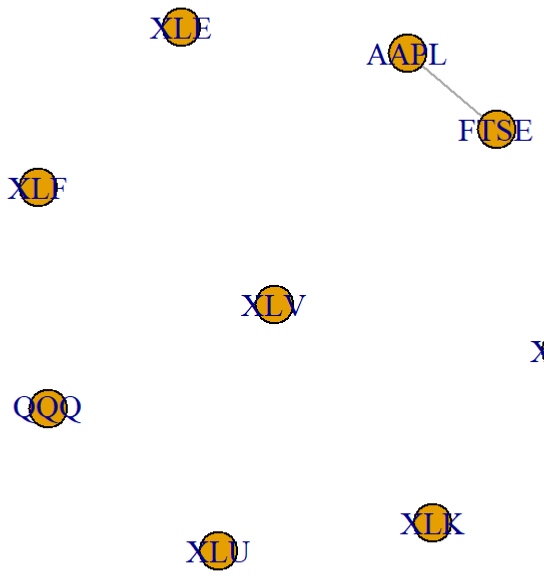
Conditional independence test of continous returns for the 6 th slice



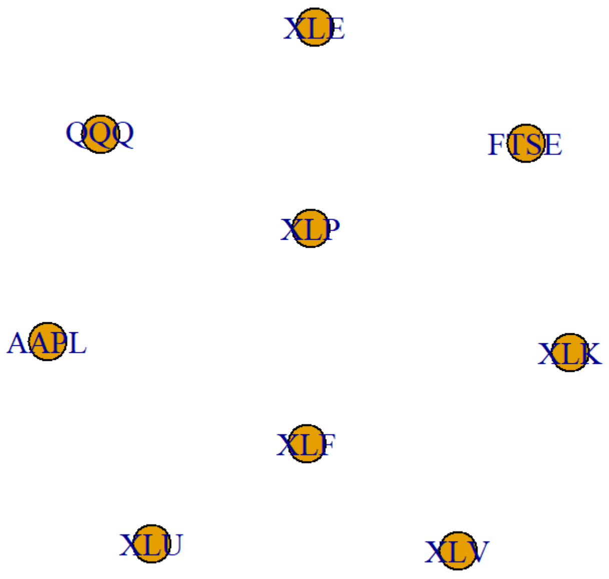
Conditional independence test of continous returns for the 7 th slice



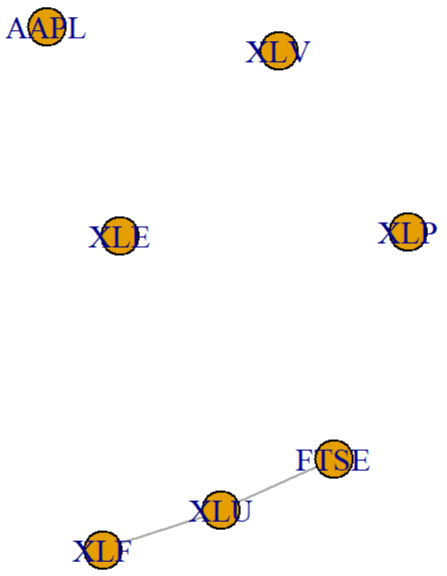
Conditional independence test of continous returns for the 8 th slice



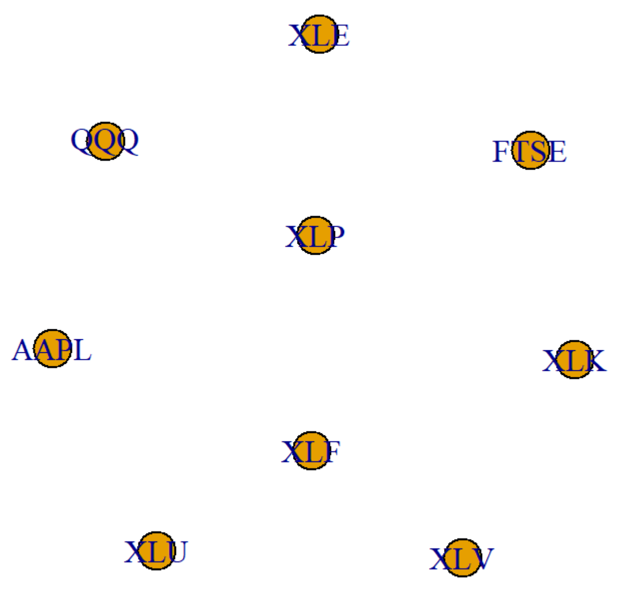
Conditional independence test of continous returns for the 9 th slice



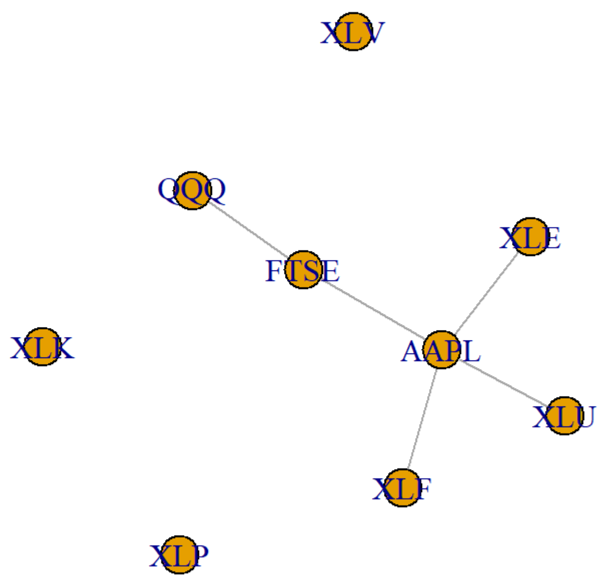
Conditional independence test of continous returns for the 10 th slice



Conditional independence test of continous returns for the 11 th slice

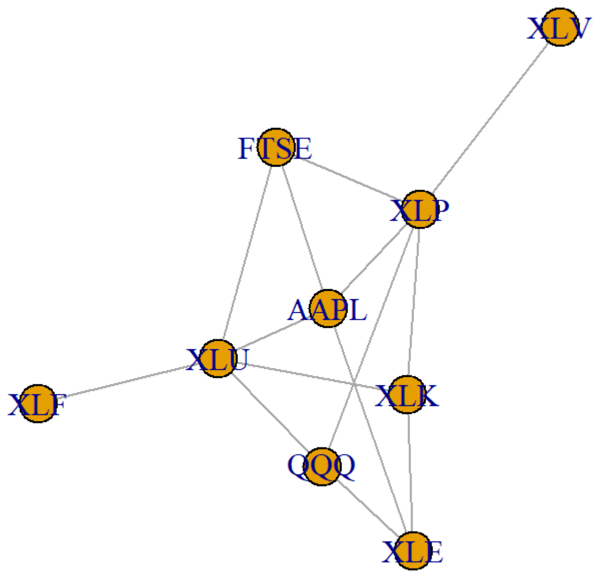


Conditional independence test of continous returns for the 12 th slice

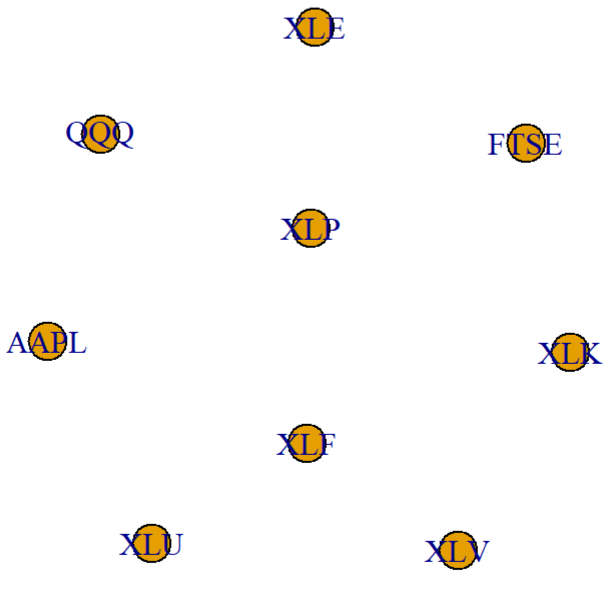


```
#plotting the graph for the conditional independence test on the count(1,0,-1) data

for(i in 1:12){
  set.seed(123)
  l<- lapply(results_count[[i]]$p_value, p_value_eval, thresh=0.01)
  results_count[[i]]$edge<- as.integer(l)
  graphss[[i]] <- results_count[[i]][c("x","y","edge")]
  grapher(graphss[[i]],compl = TRUE)
  title(paste("Conditional independence test of counts of returns for the " ,i, "th slice"))

  }
```

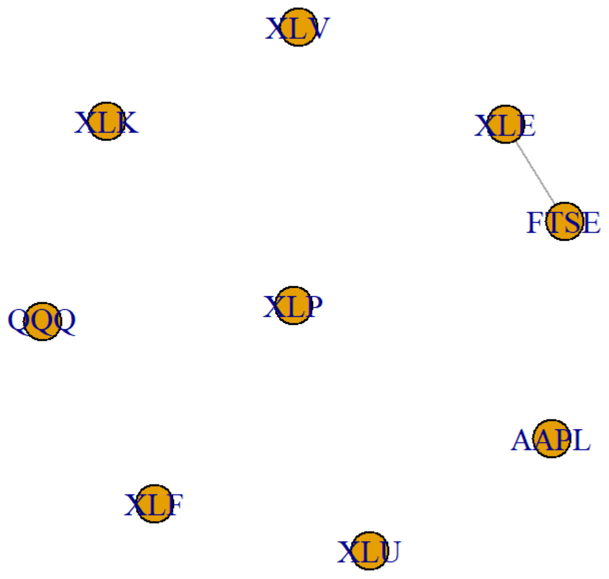
Conditional independence test of counts of returns for the 1 th slice



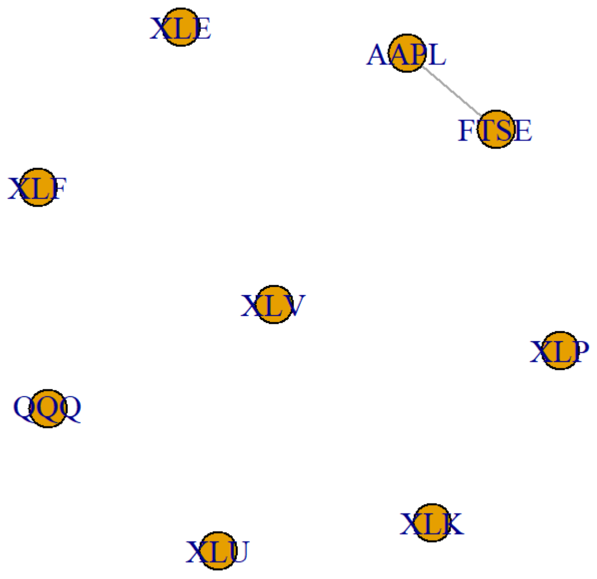
Conditional independence test of counts of returns for the 2 th slice



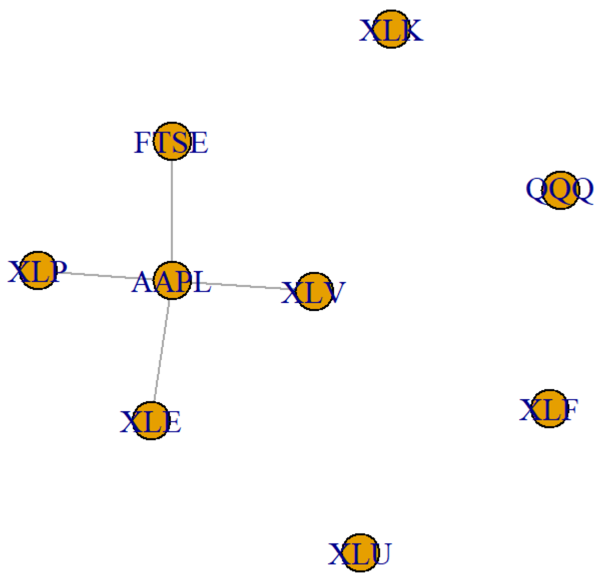
Conditional independence test of counts of returns for the 3 th slice



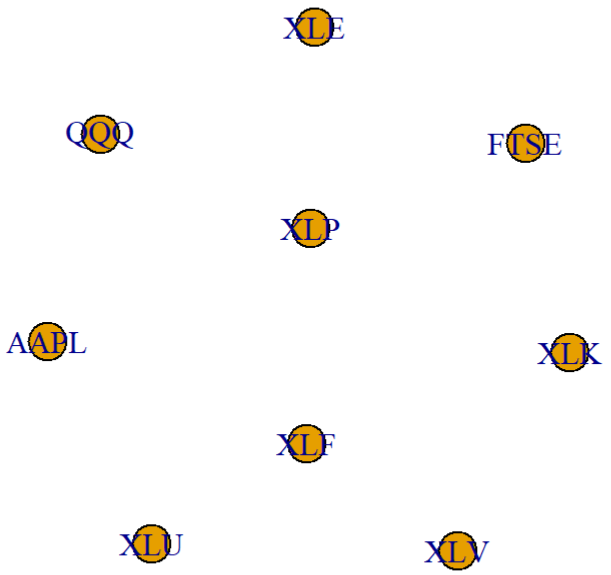
Conditional independence test of counts of returns for the 4 th slice



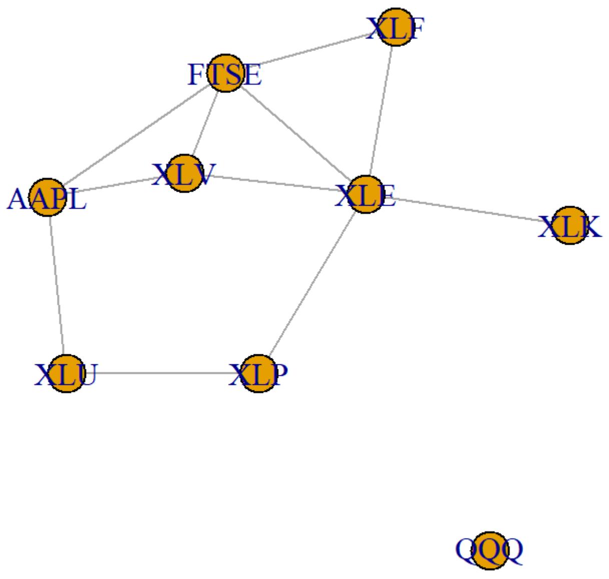
Conditional independence test of counts of returns for the 5 th slice



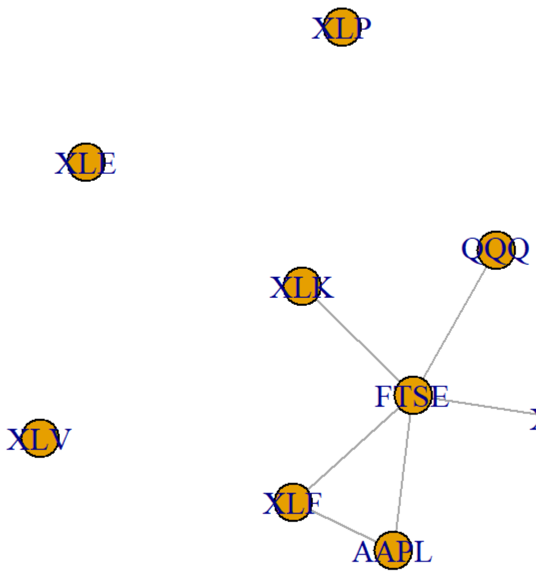
Conditional independence test of counts of returns for the 6 th slice



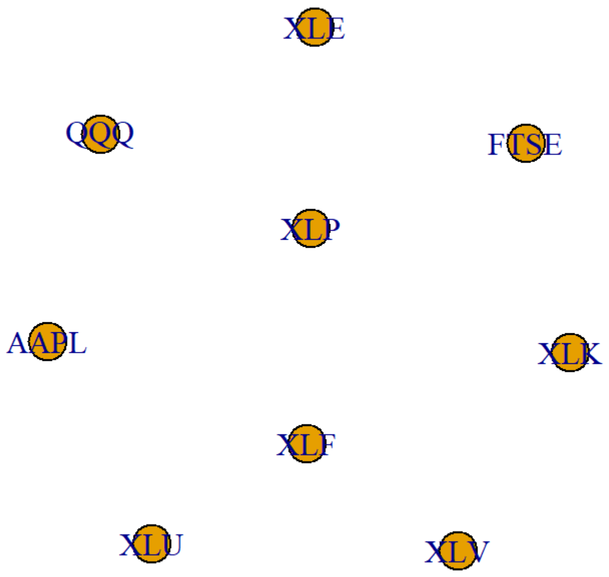
Conditional independence test of counts of returns for the 7 th slice



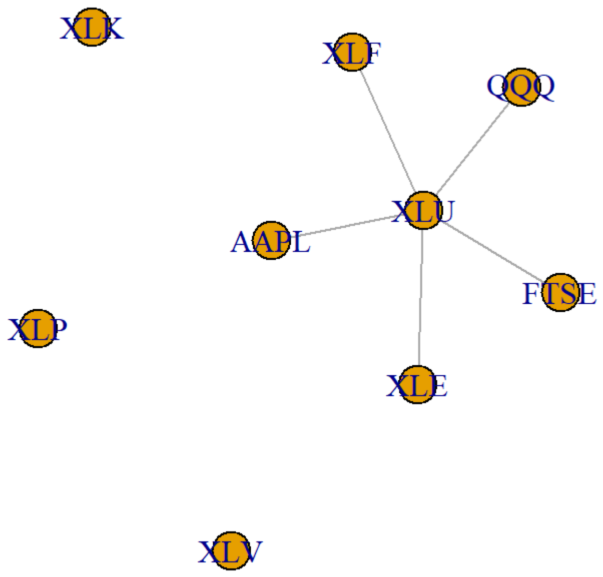
Conditional independence test of counts of returns for the 8 th slice



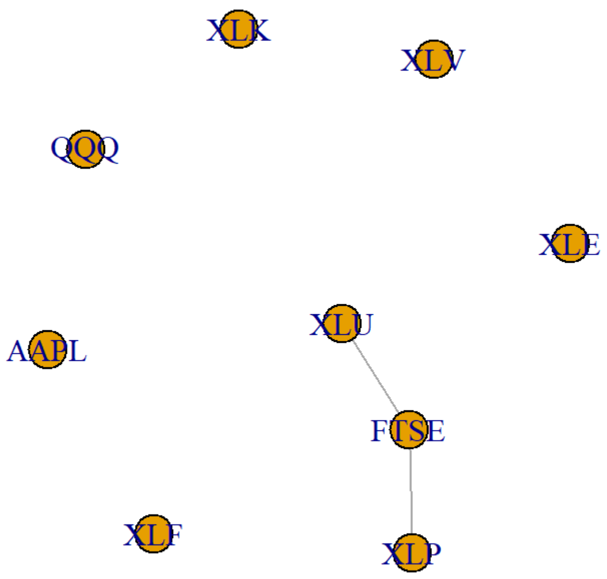
Conditional independence test of counts of returns for the 9 th slice



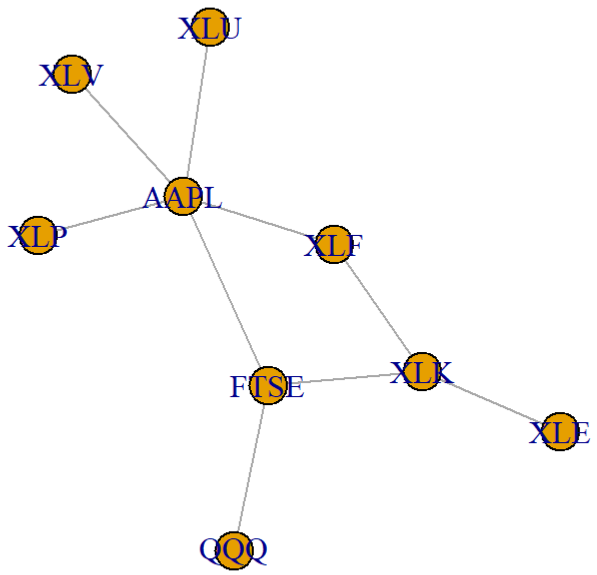
Conditional independence test of counts of returns for the 10 th slice



Conditional independence test of counts of returns for the 11 th slice



Conditional independence test of counts of returns for the 12 th slice


XLU
XLV
AAPL
XLP
XLF
XLK
FTSE
XLE
QQQ