CH5350 Applied Time Series Analysis

Final Project

Avinash Kori (ED15B006)

25 November 2017

Libraries used:

- MTS
- vars
- complexplus

Introduction

Causality

Causality is an idea of cause-and-effect, although it isn't exactly the same. Causality can be defined as an intervention of one component of a multivariate data on another component at some other time point. A variable X is causal to variable Y if X is the cause of Y or Y is the cause of X.

Project PS

In this project, we try to understand widely used method of empirical measure of **Granger causality**. which stems from econometrics, but finds wide-applicability in several scientific areas. Which involves the implementation of VAR *vector auto-regressive* models, for checking the relation between two variables.

Mathematics

Time Domain

$$\boldsymbol{z}[k] = \Sigma \boldsymbol{A}_r \boldsymbol{z}[k-r] + \boldsymbol{e}[k]$$

where $\boldsymbol{z} = [z_1[k], z_2[k], z_3[k], z_4[k], ..., z_M[k]]^T$ and $\boldsymbol{A}_r, r = 1, 2, 3..., P$ are MxM coefficient matrices. \

Granger-causal between $z_i \rightarrow z_j$ doesn't exist iff

$$a_{ij}(r) = 0 \quad \forall r$$

In practice, the estimated VAR model coefficient matrices have to be used; in which case, the existence of Granger-causal relationship from $z_j \rightarrow z_i$ can be tested by conducting a hypothesis test using the statistic.

$$S_i j = N \hat{a}_{ij}^T \widehat{V}_{ij}^{-1} \hat{a}_{ij}$$

where N is number of observations, $\widehat{\boldsymbol{a}}_i j = (a_{ij}(1), a_{ij}(2), a_{ij}(3), ..., a_{ij}(P))^T \quad \boldsymbol{V}_{ij}$ is PxP matrix, obtained by covariance matrix of VAR model.

Frequency Domain

The frequency-domain implementations of Granger causality, of which the direct pathway function (DPF) is an effective measure. The DRF, denoted by $\psi(\omega)$, connects directionally with innovations e_i to variable z_i at each frequency.

$$\psi_{ij}(\omega) = \frac{h_{D,ij}(\omega)}{\sqrt{\Sigma |h_{D,ij}(\omega)|^2}}$$

where $h_{D,ij}(\omega)$ is the frequency response function of the direct pathway from e_j to z_i.

A variable z_j does not Granger-cause z_i iff

$$|\hat{\psi}_{ij}(\omega)|^2 = 0 \quad \forall\omega$$

R Functions for all equations

a) These functons are for finding S_{ij} time-domain Granger-causality check.

if the returned histogram of Sij_realization function is gaussian that implies that i is Granger-cause of j

```
Vhat <- function(Data, Order)
{
  # Read input arguments
  Z = Data
  P = Order

  # Size of the data
  Size = dim(Z)
  N = Size[1]
  M = Size[2]

  # The regressor matrix
  BigZ = {}
  BigZ1 = {}

  for (k in 1:P){
    BigZ = cbind(BigZ, Z[k:(N-P+k-1),])
  }

  # Re-arrange the regressor matrix
  for (j in 1:M){
    BigZ1 = cbind(BigZ1,BigZ[,(seq(j,M*P,M))])
  }
  # The Y matrix
  Y = Z[(P+1):N,]

  # Regressors covariance matrix and its inverse
  SigmaR = (t(BigZ1)%*%BigZ1)/N
  invSigmaR = qr.solve(SigmaR)
```

```r
  # Estimate VAR model coefficient matrices
  Ahat = (qr.solve(t(BigZ1)%*%BigZ1)%*%t(BigZ1))%*%Y

  # Innovations covariance matrix
  SigmaE = (1/(N-P))*t((Y-BigZ1%*%Ahat))%*%(Y-BigZ1%*%Ahat)

  # Covariance matrix of VAR model coefficients
  SigmaA = kronecker(invSigmaR,SigmaE)


  # Estimation of V matrix
  Vmatrix = array(0,dim = c(P,P,M*M))
  for (i in seq(1,M*M)){
    Vmatrix[,,i] = SigmaA[((i-1)*P+1):(i*P),((i-1)*P+1):(i*P)]
  }
  return(Vmatrix)
}

##########################################################################################

aij <- function(Avec, i1, j1){
  # A is vector of matrices each of PXP dimensions
  # which form a coefficients VAR model

  aij_vector <- c()

  for(i in 1:dim(Avec)[3]){
    # paste(c(i,i1, j1))
    aij_vector <- c(aij_vector, Avec[i1,j1,i])
  }

  return(aij_vector)
}

##########################################################################################
Sij <- function(data,order){
  # aij is vector of A's at given i,j
  # N is total number of observations
  # Vij is output of vhat function
  Vhat_vector <- Vhat(data, order) # 3X3XM2 dimensions
  N <- dim(data)[1]
  M <- dim(data)[2]
  Avec <- model_coefficients(data, order)
  # print(Avec)

  Sij_vector <- c()
  for(i in 1:M){
      for(j in 1:M){
        aij_vec <- aij(Avec, i, j)
        # print(aij_vec)
        Sij_vector = c(Sij_vector, N*t(aij_vec)%*%qr.solve(Vhat_vector[, ,i])%*%aij_vec)
      }
  }
```

```r
  return(array(Sij_vector, dim = c(M,M)))
}

#####################################################################################

Sij_realizations <- function(data, order, R=1000){
  M <- dim(data)[2]
  Sij_matrix <- array(0, dim = c(M*M, R))

  for(r in 1:R){
    ek <- matrix(rnorm(dim(data)[1]*dim(data)[2]), nrow = dim(data)[1])
    Sij_vector <- c(Sij(data+ek, order))


    # print(Sij_vector)
    Sij_matrix[,r] <- Sij_vector
  }

  par(mfrow=c(M,M))
  for(i in 1:(M*M))
    hist(Sij_matrix[i,], main = "Sij Distribution")
}

#####################################################################################

model_coefficients <- function(data, order){
  model <- MTS::VAR(data, order, output = F);
  M <- dim(data)[2]
  Avec <- array(model$Phi, dim = c(M,M, order))
  return(Avec)
}

# Sij(data,3)
```

b) These functons is for finding ψ_{ij} fequency domain Granger-causality check.

```r
psi <-function(model, freq_band, plot=F){
  col_size <- dim(model$Phi)[2] # as all the coefficient matrix are stacked together
  M <- dim(model$Phi)[1] # row size of matrix A

  model_order <- col_size/M
  Avec <- array(model$Phi, dim = c(M, M, model_order))
  # print("AVEC TENSOR")
  # print(Avec)

  # Avec got updated
  # Avec is three dimensional tensor with coefficient matrix in order
  # Abar_matrix is matrix obtained from Avec as shown in given in above equation xx

  psi_matrix <- c()

  for(w in 1:length(freq_band)){
    Abar_matrix <- Abar(Avec, w)
    # print("ABAR MATRIX")
```

```
    # print(Abar_matrix)

    psi_vector <- c()
    for(psi_i in 1:M){
      for(psi_j in 1:M){
        # M is minor matrix of Abar obtained by eliminating ith row & jth col
        # Mbar is minor matrix of Abar obtained by eliminating ith & jth row & col
        h_denominator2 <- 0
        # for denominator calculation
        for(i in 1:M){
            Mij <- Abar_matrix[-c(i), -c(psi_j)]
            Mbarij <- Abar_matrix[-c(i,psi_j), -c(i,psi_j)]
            h_denominator2 <- h_denominator2 +
                      abs(hij(Abar_matrix, Mij, Mbarij, i, psi_j))^2
          }
        h_denominator <- sqrt(h_denominator2)
        # print("h_denominator")
        # print(h_denominator)

        # for numerator calculation
        Mij <- Abar_matrix[-c(psi_i), -c(psi_j)]
        Mbarij <- Abar_matrix[-c(psi_i,psi_j), -c(psi_i,psi_j)]
        h_numerator <- hij(Abar_matrix, Mij, Mbarij, psi_i, psi_j)

        # psi_value calculation...
        psi_value <- h_numerator/h_denominator
        psi_vector <- c(psi_vector, psi_value)
      }
    }
    # print("PSI VECTOR")
    # print(psi_vector)
    psi_matrix <- rbind(psi_matrix, psi_vector)
  }

  psi_matrix <- array(psi_matrix, dim = c(length(freq_band),M*M))

  # plot magnitude
  if(plot == T){
    plot.ts(abs(psi_matrix)^2)
    #plot.ts(Arg(psi_matrix))
  }
  return(t(abs(psi_matrix)^2))
}

########################################################################################

Abar <- function(Avec, w){
  # Abar is matrix obtained by sum of Avec tensor(3D)
  # abar zero initialization

  abar <- 0
  M <- dim(Avec)[1][1] # Dimension of coefficient matrix of model
  I <- diag(M) # Identity matrix of order M
```

```
  # abar updates...
  for(r in 1:dim(Avec)[3]){
    abar <- abar + Avec[, ,r]*exp(-1i*r*w)
  }

  abar <- I - abar

  # abar matrix return..
  return(abar)
}

#########################################################################################

hij <- function(Abar_vector, Mij, Mbarij, i, j){
  # equation implementation of h(w)
  dr <- Det(Abar_vector) # denominator of hij(w)

  if(i == j){
    if(length(Mij) == 1){
      nr <- Mij
    }
    else{
      nr <- Det(Mij) # numerator in case of i==j
    }
  }
  else{
    if(length(Mbarij) == 1){
      nr <- -1*Abar_vector[i,j]*Mbarij
    }
    else{
      nr <- -1*Abar_vector[i,j]*Det(Mbarij)# numerator in other cases
    }
  }
  # print("hijD")
  # print(nr/dr)
  return(nr/dr)
}
```

c & d) closed-form approximation for distribution of $|\hat{\psi}_{ij}(\omega)|^2$ functions innolved are..

Quantile calculation for large number of realizations standard gaussian distribution is assumed

```
cform_distribution <- function(zk, R, order, freq_band){
  # R is simulation steps

  dpf_datasets <- gen_realization(zk, R)

  inference_matrix_realizations <- array(1,
                                  dim = c(1, dim(zk)[2]*dim(zk)[2]*length(freq_band)))
  inference_matrix <- array(0, dim = c(dim(zk)[2], dim(zk)[2]))

  psi_tensor <- array(0, dim = c(dim(zk)[2]^2, length(freq_band),  R))

  for(i in 1:R){
```

```r
    # print(zk_datasets[, , i])
    model <- MTS::VAR(dpf_datasets[, , i], order, output = FALSE)
    psi_tensor[, , i] <- psi(model, freq_band)
  }
  zetaij_matrix <- zetaij(psi_tensor)

  # print("ZETA MATRIX")
  # print(zetaij_matrix)

  # check on data...
  psi_matrix <- psi(MTS::VAR(zk, order, output = F), freq_band)

  # print("PSI MATRIX")
  # print(psi_matrix)

  # compare psi_matrix with zetaij_matrix
  # 1 if ej is granger-cause of zi else 0
  print("INFERENCE MATRIX")
  inference_matrix_realizations[which(psi_matrix < zetaij_matrix)] = 0
  inference_matrix_realizations <- array(inference_matrix_realizations,
                                  dim = c(dim(zk)[2],dim(zk)[2],length(freq_band)))


  for(i in 1:dim(zk)[2]){
    for(j in 1:dim(zk)[2]){
      if(sum(inference_matrix_realizations[i,j,]) > 0){
        inference_matrix[i,j] = 1
      }
    }
  }

  # print(inference_matrix_realizations)
  # print(inference_matrix)

  return(t(inference_matrix))
}

##########################################################################################

gen_realization <-function(zk, R){
  # null initialized datasets
  zk_datasets <- array(0, dim = c(dim(zk)[1], dim(zk)[2], R))
  for(i in 1:R){
    # step 1 DFT of data
    zk_dft <- fft(zk)

    # step 2 Randomize the phase by replacing the phase with a randomly generated
    # sequence over the interval [0, 2pi]
    # set.seed(0)
    phase_random <- runif(dim(zk)[1], 0, 2*pi)

    # zk_dft updated with random phase
    zk_dft_phase <- zk_dft * exp(1i*phase_random)
```

```
    # check for ifft calculation only real part...
    zk_datasets[,,i] <- Re(fft(zk_dft_phase, inverse=T))/length(zk)
  }

  return(zk_datasets)
}

##########################################################################################

zetaij <- function(psi_tensor, alpha=0.95){
  M2 <- dim(psi_tensor)[1]
  w <- dim(psi_tensor)[2]
  zeta_mat <- array(0, dim = c(M2, w))
  for(i in 1:M2){
    for(j in 1:w){
      ts <- psi_tensor[i, j, ]

      # hist_val <- hist(ts, plot=F)
      # alpha% qunatile distribution calculation
      zeta_mat[i,j] <- qnorm(alpha)*sd(ts)/sqrt(length(ts))+mean(ts)
    }
  }
  return(zeta_mat)
}
```

e) Simulation using VARMAsim based on SIC

```
N <- 2000
A1 <- matrix(c(0.3, 0, 0, 0.6, 0.4, 0, 0, 0.4, 0.5), nrow=3, byrow = T)
A2 <- matrix(c(0.2, 0, 0, 0, 0.5, 0, 0, 0.3, 0.4), nrow=3, byrow = T)

sig <- diag(3)
# set.seed(20)
zksim <- MTS::VARMAsim(N, arlags = c(1, 2), phi = cbind(A1,A2), sigma = sig)

sim_model <- vars::VARselect(zksim$series, lag.max = 10)

# best model is...
best_order <- which.min(sim_model$criteria["SC(n)",])
cat("Best VAR model based on SIC is: ", best_order)
```

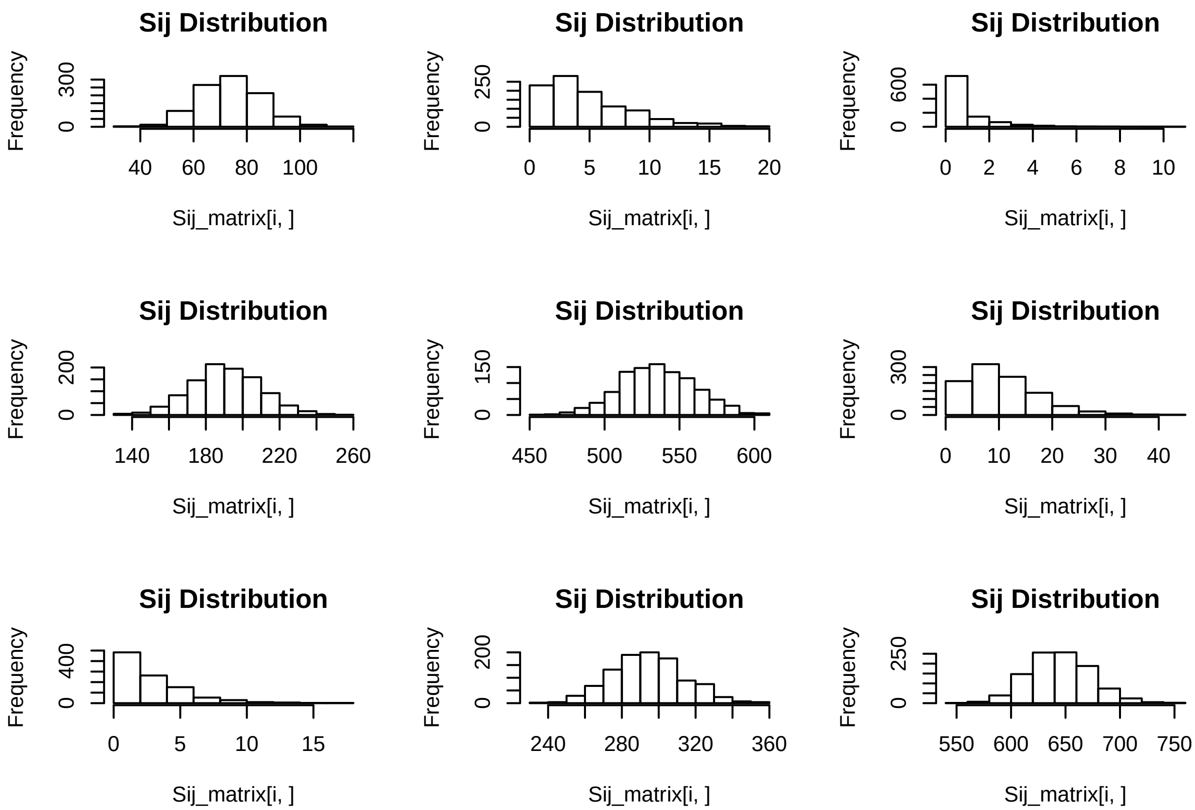
```
## Best VAR model based on SIC is:  2
```

```
# model is

best_model <- MTS::VAR(zksim$series, best_order)
```

```
## Constant term:
## Estimates:  0.01586428 0.04402045 0.000985901
## Std.Error:  0.02377998 0.02295323 0.02382566
## AR coefficient matrix
## AR( 1 )-matrix
##         [,1]  [,2]    [,3]
## [1,] 0.29096 0.031 -0.0025
## [2,] 0.62542 0.381 -0.0184
```

```
## [3,] 0.00442 0.423  0.5380
## standard error
##        [,1]   [,2]   [,3]
## [1,] 0.0221 0.0194 0.0205
## [2,] 0.0213 0.0187 0.0197
## [3,] 0.0221 0.0195 0.0205
## AR( 2 )-matrix
##         [,1]    [,2]    [,3]
## [1,]  0.1908 -0.0321 0.00348
## [2,]  0.0123  0.5158 0.01554
## [3,] -0.0247  0.2576 0.36570
## standard error
##        [,1]   [,2]   [,3]
## [1,] 0.0256 0.0206 0.0192
## [2,] 0.0247 0.0199 0.0185
## [3,] 0.0257 0.0207 0.0192
##
## Residuals cov-mtx:
##             [,1]        [,2]        [,3]
## [1,]  1.02942579 -0.01126937  0.03728825
## [2,] -0.01126937  0.95909089 -0.05372049
## [3,]  0.03728825 -0.05372049  1.03338467
##
## det(SSE) =  1.015884
## AIC =  0.03375876
## BIC =  0.08416688
## HQ  =  0.05226757
```

```r
# time domain causality test : inferance based on Gaussian
# distributed histogram
Sij_realizations(zksim$series, best_order)
```




```r
# frequency domain causality test
cform_distribution(zksim$series, 50, best_order, w)
```

```
## [1] "INFERENCE MATRIX"

##      [,1] [,2] [,3]
## [1,]    0    1    0
## [2,]    1    1    1
## [3,]    0    1    1
```

f)

- For large number of realizations both time and frequency domain methods yeild correct results for *Granger-causality*
- Frequency domain *Granger-causality* will give more accurate results as it uses more number of observations
- Answer remains invariant with given realization, but is highly dependent on number of Realizations which can be observed:

```r
cform_distribution(Zk, R=2, 2, w)
```

```
## [1] "INFERENCE MATRIX"

##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    1    0    1
```

```r
cform_distribution(Zk, R=20, 2, w)
```

```
## [1] "INFERENCE MATRIX"

##      [,1] [,2] [,3]
```

```
## [1,]    0    0    0
## [2,]    1    1    0
## [3,]    0    0    1
```

```r
cform_distribution(Zk, R=50, 2, w)
```

```
## [1] "INFERENCE MATRIX"
```

```
##      [,1] [,2] [,3]
## [1,]    0    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

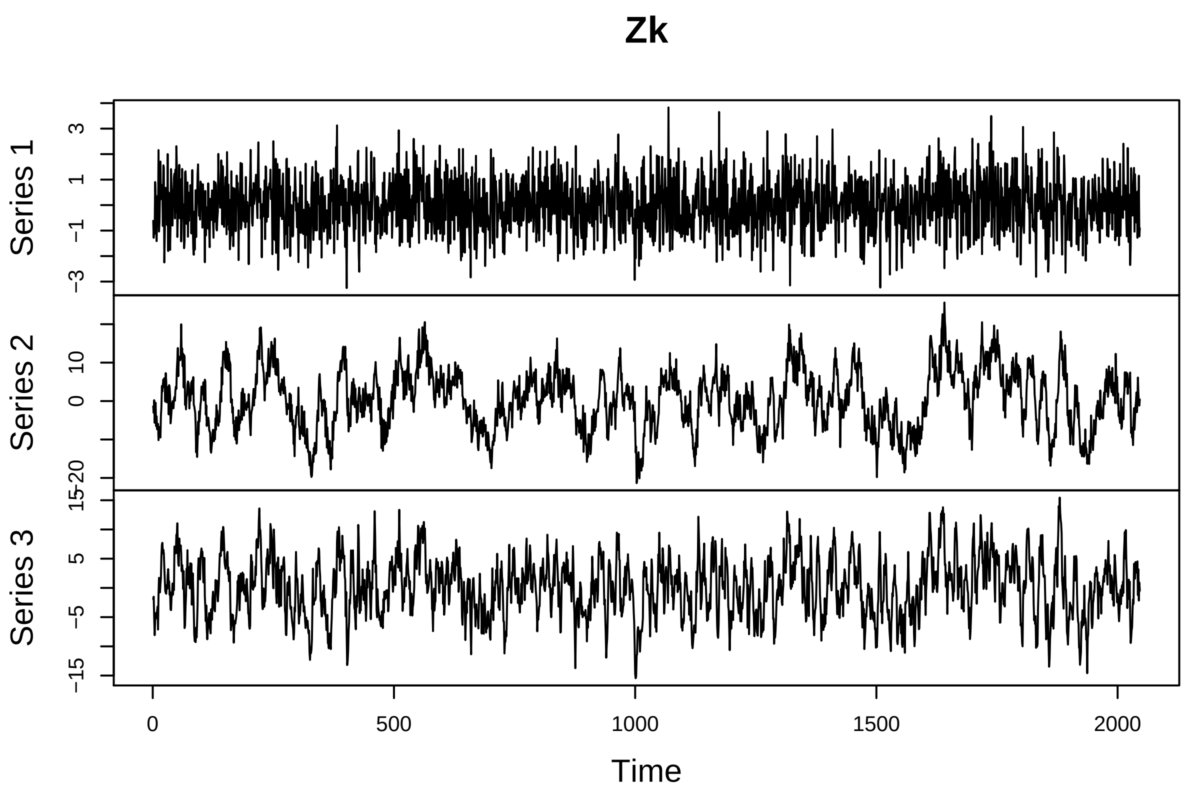
---

2)

Data Pre-Processing & Model Building

Plots and nature of data

```r
plot.ts(Zk)
```



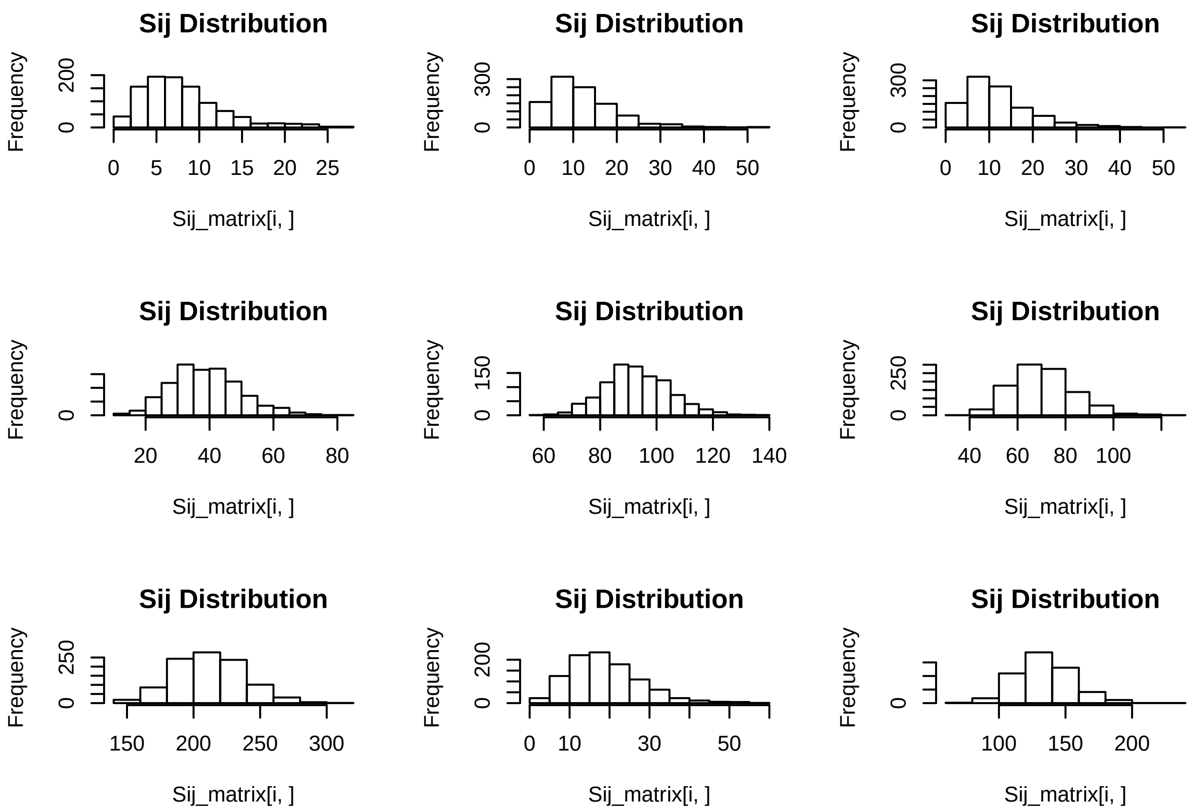
Normalization

```r
# normaliaztion to bring all series in same range

Zk[,1] <- (Zk[,1]-mean(Zk[,1]))/sd(Zk[,1])
```

```r
Zk[,2] <- (Zk[,2]-mean(Zk[,2]))/sd(Zk[,2])
Zk[,3] <- (Zk[,3]-mean(Zk[,3]))/sd(Zk[,3])

## 1a & 1b implementation
data_model <- vars::VARselect(Zk, lag.max = 10)

# best model is...
data_order <- which.min(data_model$criteria["SC(n)",])

Sij(Zk, data_order)
```

```
##            [,1]      [,2]       [,3]
## [1,]  32.54673 325.30671 1891.19592
## [2,]  37.02977 157.12525   14.48791
## [3,] 348.20709  38.20473   67.51265
```

```r
# evaluation for realizations...
question1_a <- Sij_realizations(Zk, data_order, 1000)
```




```r
# evaluation in freq domain
cform_distribution(Zk, R=50, data_order, w)
```

```
## [1] "INFERENCE MATRIX"

##      [,1] [,2] [,3]
## [1,]    0    0    0
## [2,]    1    1    0
## [3,]    1    0    1
```

```
model <- MTS::VAR(Zk, data_order, output = F)

question1_b <- psi(model, w, T)
```