R Code for the Paper: "The Joint Spillover Index"

Thomas F. P. Wiesen
University of Maine School of Economics
5782 Winslow Hall
Orono, ME 04469, USA
thomas.wiesen@maine.edu

2021

Introduction and Directions

This supplemental document provides R code to compute the joint spillover metrics proposed in Lastrapes and Wiesen (2021, "The Joint Spillover Index," *Economic Modelling*. 94: 681-691). Please cite the aforementioned paper if utilizing this code. For comparison purposes, this code also calculates the generalized spillover metrics of Diebold and Yilmaz (2012, 2014). We recommend that users of this supplemental document copy and paste the provided code into their own code to calculate the desired spillover measures.

First load your data into R and estimate the K-variable VAR(P) model using your estimation package of choice (e.g., the vars package in R). Then obtain the VMA representation of the VAR:

$$\boldsymbol{Y_t} = \boldsymbol{c} + \sum_{h=0}^{\infty} \boldsymbol{A}_h \boldsymbol{\epsilon}_{t-h},$$

where $\boldsymbol{A}_0 = \boldsymbol{I}_K$. Save the $K \times K$ shock covariance matrix $\boldsymbol{\Sigma}_\epsilon = E\left(\boldsymbol{\epsilon}_t \boldsymbol{\epsilon}_t'\right)$ and denote it in your R code as `sigma_eps`. Also save the VMA coefficient matrices up to the chosen time horizon as a $K \times K \times H$ array denoted as `A` in your R code. If you only wish to measure spillovers using a single period horizon ($H = 1$), then only save $\boldsymbol{A}_0$. If you wish to measure spillovers using a two-period horizon ($H = 2$), then save $\boldsymbol{A}_0$ and $\boldsymbol{A}_1$. If you wish to measure spillovers using a three-period horizon ($H = 3$), then save $\boldsymbol{A}_0$, $\boldsymbol{A}_1$, and $\boldsymbol{A}_2$. And so on. The required inputs of the code below are the $K \times K$ shock covariance matrix ($\boldsymbol{\Sigma}_\epsilon$) and the VMA coefficient matrices ($\boldsymbol{A}_0, \boldsymbol{A}_1, ..., \boldsymbol{A}_{H-1}$) written as a $K \times K \times H$ array.

These first steps are not included in the R code. However, we provide a simple numeric example in the code that sets

$$\boldsymbol{\Sigma}_\epsilon = \begin{bmatrix} 2 & 1.3 & .7 \\ 1.3 & 1.5 & .6 \\ .7 & .6 & 1 \end{bmatrix}$$

and

$$\boldsymbol{A}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{A}_1 = \begin{bmatrix} .5 & .2 & .1 \\ -.2 & .4 & .3 \\ -.1 & 0 & .6 \end{bmatrix} \quad \boldsymbol{A}_2 = \begin{bmatrix} .2 & .18 & .17 \\ -.21 & .12 & .28 \\ -.11 & -.02 & .35 \end{bmatrix}.$$

Thus, in this example, $K = 3$ and $H = 3$. Delete or comment out this example from the R code when estimating the VAR parameters.

Once the VMA coefficient matrices and the shock covariance matrix are saved, the provided code will calculate the joint total spillovers from all others to a particular variable, the joint total spillovers to all others from a particular variable, the joint net total spillovers for a particular variable, the scaling factor, and the joint spillover index. Furthermore, the provided code will calculate the generalized total spillovers from all others to a particular variable, the generalized total spillovers to all others from a particular variable, the generalized

net total spillovers for a particular variable, the generalized pairwise spillovers (i.e., the generalized spillover table), and the generalized spillover index. See the main text of Lastrapes and Wiesen (2021) for details.

Notation of Inputs

Mathematical Notation	R Code Notation	Dimension	Description
$\boldsymbol{\Sigma}_\epsilon$	`sigma_eps`	$K \times K$	shock covariance matrix
$\boldsymbol{A}_0, ..., \boldsymbol{A}_{H-1}$	`A`	$K \times K \times H$	VMA coefficient matrices
H	`H`	1×1	forecast horizon
K	`K`	1×1	number of variables in VAR

Notation of Outputs

Mathematical Notation	R Code Notation	Equation Reference	Dimension	Description
$\boldsymbol{gSOT}$	`gSOT`	(12)	$K \times K$	generalized spillover table
$\mathcal{S}^{gen}_{\bullet \rightarrow i}, \forall i$	`S_gen_from_others`	(13)	$K \times 1$	generalized total spillover from others
$\mathcal{S}^{gen}_{\bullet \leftarrow i}, \forall i$	`S_gen_to_others`	(16)	$K \times 1$	generalized total spillover to others
$\mathcal{S}^{gen}_{net,i}, \forall i$	`S_gen_net`	(17)	$K \times 1$	generalized net total spillover
$gSOI$	`gSOI`	(18)	1×1	generalized spillover index
$\mathcal{S}^{jnt}_{\bullet \rightarrow i}, \forall i$	`S_jnt_from_others`	(14)	$K \times 1$	joint total spillover from others
$\mathcal{S}^{jnt}_{\bullet \leftarrow i}, \forall i$	`S_jnt_to_others`	(21)	$K \times 1$	joint total spillover to others
$\mathcal{S}^{jnt}_{net,i}, \forall i$	`S_jnt_net`	(22)	$K \times 1$	joint net total spillover
λ	`lambda`	(20)	1×1	scaling factor
$jSOI$	`jSOI`	(23)	1×1	joint spillover index

R code

```
#R code to calculate the joint and generalized spillover metrics.
#Code Author: Thomas F. P. Wiesen.
#The user must first estimate the VAR and obtain the VMA coefficient matrices (A)...
#and the shock covariance matrix (sigma_eps).  This step is not shown.  Note that...
#the "vars" package calls the VMA coefficient matrices Phi, but this code calls...
#them A.


#Set example values for A and sigma_eps.
sigma_eps=array(c(2,1.3,.7,1.3,1.5,.6,.7,.6,1), dim=c(3,3))
A=array(c(1,0,0,0,1,0,0,0,1,.5,-.2,-.1,.2,.4,0,.1,.3,.6,.2,-.21,-.11,.18,.12,-.02,
          .17,.28,.35), dim=c(3,3,3))



#############################################################################
#Set the forecast horizon (H).
H=dim(A)[3]
#When H=1, the above line returns H=na because R thinks there is no third...
#dimension.  Thus, in that case, we must force H=1.
if (is.na(H)==TRUE) {
```

```
  H=1
}
#Set the number of variables in the VAR (K).
K=dim(sigma_eps)[1]


#Calculate the forecast error covariance matrix (Xi).
if (H>1){
  Xi_h=array(0,dim=c(K,K,H))
  for (h in 1:H){
      Xi_h[,,h]=A[,,h]%*%sigma_eps%*%t(A[,,h])  #Calculated Xi at each h.
  }
  #Sum them along THIRD dimension to form Xi.
  Xi=rowSums(Xi_h, dims=2)
  #Note that because the above function is a row sum, dims=2 actually sums along...
  #the third dimension.
}
if (H==1){
  Xi=sigma_eps   #When H=1, Xi is equal to sigma_eps because A_0=I_K.
}


#Identity matrix.
I_K=diag(1,nrow=K,ncol=K)


#Calculate the gFEVD.
if (H>1){
  A_times_sigma_squar_h=array(0,dim=c(K,K,H))
  for (h in 1:H){
    #Calculate A times Sigma squared (element by element squared) at each h.
    A_times_sigma_squar_h[,,h]=(A[,,h]%*%sigma_eps)*(A[,,h]%*%sigma_eps)
  }
  #Sum them along THIRD dimension
  A_times_sigma_squar_sum=rowSums(A_times_sigma_squar_h, dims=2)
  #Note that because the above function is a row sum, dims=2 actually sums along...
  #the third dimension.
}
if (H==1){
  #When H=1, the numerator of the gFEVD is just the element by element squares...
  #of sigma_eps.
  A_times_sigma_squar_sum=sigma_eps*sigma_eps
}
gFEVD=array(0,dim=c(K,K))
for (i in 1:K){
  for (j in 1:K){
    gFEVD[i,j]=A_times_sigma_squar_sum[i,j]/(sigma_eps[j,j]*Xi[i,i])
  }
}


#Calculate the gSOT.
gFEVD_row_sum=rowSums(gFEVD)  #Calculate the gFEVD row sums.
gSOT=array(0,dim=c(K,K))
for (i in 1:K){
  for (j in 1:K){
    gSOT[i,j]=(gFEVD[i,j])/gFEVD_row_sum[i]    #Calculate the gSOT.
    }
}
```

```
#Calculate the generalized total spillover from all others to variable i...
#(S_gen_from_others) and the generalized total spillover to others from variable...
#i (S_gen_to_others).
S_gen_from_others=array(0,dim=c(K))
S_gen_to_others=array(0,dim=c(K))
for (i in 1:K){
  #Generalized total spillover from others to variable i.
  S_gen_from_others[i]=sum(gSOT[i,])-gSOT[i,i]
  #Generalized total spillover to others from variable i.
  S_gen_to_others[i]=sum(gSOT[,i])-gSOT[i,i]
}


#Calculate the generalized net total spillover (S_gen_net).
S_gen_net=S_gen_to_others-S_gen_from_others


#Calculate the generalized spillover index (gSOI).
gSOI=mean(S_gen_from_others)


#Calculate the elimination matrices.  These are usually denoted as a KxK-1 matrix...
#M_i.  Here, they are an array where M[,,1]=M_1, and in general M[,,i]=M_i.
M=array(0,dim=c(K,K-1,K))
for (i in 1:K){
  M[,,i]=I_K[,-i]  #Calculate the elimination matrices.
}


#Calculate the joint total spillover from all others to variable i...
#(S_jnt_from_others).
#Calculate the numerator of S_jnt_from_others.
if (H>1){
  S_jnt_from_numerator_h=array(0,dim=c(K,H))
  for (i in 1:K){
    for (h in 1:H){
      #Calculate the numerator of S_jnt_from_others at each h.
      S_jnt_from_numerator_h[i,h]=I_K[i,]%*%A[,,h]%*%sigma_eps%*%M[,,i]%*%
        (solve(t(M[,,i])%*%sigma_eps%*%M[,,i]))%*%t(M[,,i])%*%sigma_eps%*%
        t(A[,,h])%*%I_K[,i]
    }
  }
  S_jnt_from_numerator=array(0,dim=c(K))
  for (i in 1:K){
    #Calculate the numerator of S_jnt_from_others (sum over h).
    S_jnt_from_numerator[i]=sum(S_jnt_from_numerator_h[i,])
  }
}
if (H==1){
  S_jnt_from_numerator=array(0,dim=c(K))
  for (i in 1:K){
    S_jnt_from_numerator[i]=I_K[i,]%*%sigma_eps%*%M[,,i]%*%(solve(t(M[,,i])%*%
         sigma_eps%*%M[,,i]))%*%t(M[,,i])%*%sigma_eps%*%I_K[,i]
    #When H=1, A is just the identity matrix, and thus it cancels out.
  }
}
S_jnt_from_others=array(0,dim=c(K))
```

```
for (i in 1:K){
  S_jnt_from_others[i]=S_jnt_from_numerator[i]/Xi[i,i]
}


#Calculate the joint spillover index (jSOI).
jSOI=mean(S_jnt_from_others)


#Calculate the scaling factor lambda.
lambda=jSOI/gSOI


#Calculate gSOT tilde (gSOT_tilde).
gSOT_tilde=lambda*gSOT


#Calculate the joint total spillover to all others from variable i...
#(S_jnt_to_others).
S_jnt_to_others=array(0,dim=c(K))
for (i in 1:K){
  #Joint total spillover to others from variable i.
  S_jnt_to_others[i]=sum(gSOT_tilde[,i])-gSOT_tilde[i,i]
}


#Calculate the joint net total spillover (S_jnt_net).
S_jnt_net=S_jnt_to_others-S_jnt_from_others


################################################################################
#OUTPUT.
#Print the spillover measures.  All spillover measures (excluding lambda) have...
#been multiplied by 100 and thus are reported as a percent.
#Print the generalized spillover table.
print(100*gSOT)
#Print the generalized total spillover from all others to the individual variables.
print(100*S_gen_from_others)
#Print the generalized total spillover to all others from the individual variables.
print(100*S_gen_to_others)
#Print the generalized net total spillover.
print(100*S_gen_net)
#Print the generalized spillover index.
print(100*gSOI)
#Print the joint total spillover from all others to the individual variables.
print(100*S_jnt_from_others)
#Print the joint total spillover to all others from the individual variables.
print(100*S_jnt_to_others)
#Print the joint net total spillover.
print(100*S_jnt_net)
#Print the scaling factor.
print(lambda)
#Print the joint spillover index.
print(100*jSOI)
```