R PACKAGE BHSBVAR

Paul Richardson
p.richardson.54391@gmail.com
June, 2021

ACKNOWLEDGMENT

The `BHSBVAR` package is based on the MATLAB programs created by Baumeister and Hamilton (2015/2017/2018). I thank them for sharing their MATLAB programs online.

INTRODUCTION

Identifying structural innovations from Structural Vector Autoregression (SVAR) models requires the researcher to make assumptions about the structural parameters in the model. Recursively identifying structural innovations with the Cholesky decomposition of the residual covariance matrix requires the researcher to assume a recursive order of the endogenous variables in the model and exclusion or information lag restrictions for structural parameters. Identifying structural innovations with sign restrictions prevents researchers from having to make zero restrictions for structural parameters but it does implicitly require researchers to assume a prior distribution for structural parameters they may not agree with. The method developed by Baumeister and Hamilton (2015/2017/2018) for estimating the parameters of a Structural Bayesian Vector Autoregression (SBVAR) model is an alternative method that allows the researcher to explicitly include prior information about the parameters of the model. Their method does not require the researcher to assume a recursive order of the endogenous variables in the model or a prior distribution about structural parameters the researcher does not agree with. For detailed information about their method see Baumeister and Hamilton (2015/2017/2018).

MODEL

Let Y be an $(n \times T)$ matrix of endogenous variables. X is a $(k \times T)$ matrix containing L lags of the endogenous variables and a constant. A is an $(n \times n)$ matrix containing the short-run elasticities or the structural relationships between the endogenous variables in Y from an SVAR model. B is an $(n \times k)$ matrix containing lagged structural coefficients. U is an $(n \times T)$ matrix of structural innovations. D is an $(n \times n)$ diagonal covariance matrix of the innovations from the structural model. n is the number of endogenous variables or equations. T is the number of observations and $k = nL + 1$.

Structural Vector Autoregression Model:

$$AY = BX + U \qquad U \sim N(0, D) \tag{1}$$

$$B = A\Phi \tag{2}$$

$$U = A\epsilon \tag{3}$$

$$D = \frac{UU^\top}{T} = A\Omega A^\top \tag{4}$$

Φ is an $(n \times k)$ matrix containing the lagged coefficients from the reduced form Vector Autoregression (VAR) model. ϵ is an $(n \times T)$ matrix of the VAR model residuals. Ω is an $(n \times n)$ symmetric covariance matrix of the residuals from the VAR model.

Reduced Form Vector Autoregression Model:

$$Y = \Phi X + \epsilon \qquad \epsilon \sim N(0, \Omega) \tag{5}$$

$$\Phi = (YX^\top)(XX^\top)^{-1} \tag{6}$$

$$\epsilon = Y - \Phi X \tag{7}$$

$$\Omega = \frac{\epsilon\epsilon^\top}{T} \tag{8}$$

Let y_i be a $(1 \times T)$ matrix containing a single endogenous variable from Y for $i = 1, 2, ..., n$. Let x_i be an $((L + 1) \times T)$ matrix of L lags of y_i and a constant for $i = 1, 2, ..., n$. ϕ_i is a $(1 \times (L + 1))$ matrix containing the lagged coefficients from the reduced form univariate Autoregression (AR) model for $i = 1, 2, ..., n$. e_i is a $(1 \times T)$ matrix of the residuals from the univariate AR model for $i = 1, 2, ..., n$. e is an $(n \times T)$ matrix of residuals from the univariate AR models. Σ is an $(n \times n)$ symmetric covariance matrix of the residuals from the univariate AR models. Σ_i is the (i, i) element of Σ for $i = 1, 2, ..., n$.

Reduced Form Univariate Autoregression Model:

$$y_i = \phi_i x_i + e_i \qquad e_i \sim N(0, \Sigma_i) \tag{9}$$

$$\phi_i = (y_i x_i^\top)(x_i x_i^\top)^{-1} \tag{10}$$

$$e_i = y_i - \phi_i x_i \tag{11}$$

$$\Sigma = \frac{ee^\top}{T} \tag{12}$$

Let P be an $(n \times k)$ matrix containing the prior position values for the reduced form lagged coefficient matrix, Φ. M^{-1} is a $(k \times k)$ symmetric matrix indicating confidence in P. R is an $(n \times k)$ matrix containing the prior position values for long-run restrictions on the lagged structural coefficient matrix B. $R_{i,*}$ is row i of R. V_i^{-1} is a $(k \times k)$ symmetric matrix indicating confidence in R, one matrix for each equation $i = 1, 2, ..., n$. β_i is a $(1 \times k)$ lagged structural coefficient matrix, one matrix for each equation $i = 1, 2, ..., n$. $B_{i,*}$ is row i of the lagged structural coefficients matrix B for $i = 1, 2, ..., n$. Z is an $(n \times n)$ diagonal matrix. Z_i is the (i, i) element of the diagonal matrix Z for $i = 1, 2, ..., n$. $diag(A\Sigma A^\top)$ is an $(n \times n)$ diagonal matrix whose main diagonal elements are the main diagonal elements from the matrix $A\Sigma A^\top$. κ is an $(n \times n)$ diagonal matrix whose elements along the main diagonal represent confidence in the priors for the structural variances in D. κ_i and τ_i refers to element (i, i) of κ and τ, respectively for $i = 1, 2, ..., n$. The prior for $D_i \sim \frac{1}{\Gamma(\kappa_i, \tau_i)}$ where D_i refers to element (i, i) of D for $i = 1, 2, ..., n$.

Structural Bayesian Vector Autoregression Model:

$$AY = BX + U \qquad U \sim N(0, D) \tag{13}$$

$$\beta_i = (A_{i,*}(YX^\top + PM^{-1}) + R_{i,*}V_i^{-1})(XX^\top + M^{-1} + V_i^{-1})^{-1} \tag{14}$$

$$B_{i,*} = \beta_i \tag{15}$$

$$Z_i = (A_{i,*}(YY^\top + PM^{-1}P^\top)A_{i,*}^\top + R_{i,*}V_i^{-1}R_{i,*}^\top) - [(A_{i,*}(YX^\top + PM^{-1}) + R_{i,*}V_i^{-1}) (XX^\top + M^{-1} + V_i^{-1})^{-1}(A_{i,*}(YX^\top + PM^{-1}) + R_{i,*}V_i^{-1})^\top] \tag{16}$$

$$\tau = \kappa diag(A\Sigma A^\top) \tag{17}$$

$$\tau^* = \tau + \tfrac{1}{2}Z \tag{18}$$

Baumeister and Hamilton (2015/2017/2018) developed an algorithm that estimates the parameters of an SVAR model using Bayesian methods (SBVAR). Their algorithm applies a random-walk Metropolis-Hastings algorithm to seek elasticity values for A, considering prior information, that diagonalizes the covariance matrix of the reduced form errors. For detailed information about their algorithm see Baumeister and Hamilton (2015/2017/2018).

Posterior Distribution:

$$p(A|Y) \propto p(A)[det(A\Omega A^\top)]^{\frac{T}{2}} \prod_{i=1}^{n} (\tau_i)^{\kappa_i} (\tfrac{2}{T}\tau_i^*)^{-(\kappa_i + \frac{T}{2})} \tag{19}$$

$p(A|Y)$ is the posterior density of $A|Y$. The products of $p(det(A))$ and/or $p(H)$ are multiplied by $p(A|Y)$ when priors are chosen for $det(A)$ and/or H (A^{-1}), respectively. $p(A)$, $p(det(A))$, and $p(H)$ are products of the prior densities for A, $det(A)$, and H (A^{-1}), respectively. $det(A)$ and $det(A\Omega A^\top)$ are the determinants of the matrices A and $A\Omega A^\top$, respectively.

Algorithm:

Step 1)
Generate draws for $A|Y$ ($\tilde{A}^{(c+1)}$). $\tilde{A}^{(c)}$ are the starting values for A when $c = 1$. Compute $p(\tilde{A}^{(c)}|Y)$ and $p(\tilde{A}^{(c+1)}|Y)$. If $p(\tilde{A}^{(c+1)}|Y) < p(\tilde{A}^{(c)}|Y)$ set $\tilde{A}^{(c+1)} = \tilde{A}^{(c)}$ with probability $1 - \frac{p(\tilde{A}^{(c+1)}|Y)}{p(\tilde{A}^{(c)}|Y)}$.

Step 2)
Generate draws for $D|A, Y$ ($\tilde{D}^{(c+1)}$). $\tilde{D}_i^{(c+1)} \sim \frac{1}{\Gamma(\kappa_i + \frac{T}{2}, \tilde{\tau}_i^{*(c+1)})}$. $\tilde{D}_i^{(c+1)}$, κ_i, and $\tilde{\tau}_i^{*(c+1)}$ refers to element (i, i) of $\tilde{D}^{(c+1)}$, κ, and $\tilde{\tau}^{*(c+1)}$, respectively for $i = 1, 2, ..., n$. $\tilde{\tau}^{*(c+1)}$ are estimates of τ^*, replacing A with $\tilde{A}^{(c+1)}$.

Step 3)
Generate draws for $B|D, A, Y$ ($\tilde{B}^{(c+1)}$). $\tilde{B}_{i,*}^{(c+1)} = \hat{\beta}_i^{(c+1)}$ for $i = 1, 2, ..., n$. $\hat{\beta}_i^{(c+1)} \sim N(\tilde{\beta}_i^{(c+1)}, \tilde{\Psi}_i^{(c+1)})$ for $i = 1, 2, ..., n$. $\tilde{\beta}_i^{(c+1)}$ are estimates of β_i, replacing A with $\tilde{A}^{(c+1)}$ for $i = 1, 2, ..., n$. $\tilde{\Psi}_i^{(c+1)} = \tilde{D}^{(c+1)}(XX^\top + M^{-1} + V_i^{-1})^{-1}$ for $i = 1, 2, ..., n$.

Step 4)
Increase c by 1 and repeat Steps 1-4 for $c = 1, 2, ..., C$. C is the total number of iterations.

EXAMPLE

The `BHSBVAR` package provides a function for estimating the parameters of SBVAR models and several functions for plotting results. The `BH_SBVAR()` function estimates the parameters of an SBVAR model with the method developed by Baumeister and Hamilton (2015/2017/2018). The `IRF_Plots()` function creates plots of impulse responses. The `FEVD_Plots()` function creates plots of forecast error variance decompositions. The `HD_Plots()` function creates plots of historical decompositions. The `Dist_Plots()` function creates posterior density plots of the model parameters in A, $det(A)$, and H overlaid with prior densities to illustrate the difference between posterior and prior distributions. The following example illustrates how these functions can be applied to reproduce the results from Baumeister and Hamilton (2015).

Code Chunk 1

```
> rm(list = ls())
> library(BHSBVAR)
> set.seed(123)
```

```
> data(USLMData)
> y0 <- matrix(data = c(USLMData$Wage, USLMData$Employment), ncol = 2)
> y <- y0 - (matrix(data = 1, nrow = nrow(y0), ncol = ncol(y0)) %*%
+              diag(x = colMeans(x = y0, na.rm = FALSE, dims = 1)))
> colnames(y) <- c("Wage", "Employment")
```

The first line from Code Chunk 1 clears the workspace. The second line loads the `BHSBVAR` package namespace. The third line sets the seed for random number generation. The fourth line imports the data used in this example. The fifth through seventh line creates a matrix (`y`) containing quarter over quarter percent change of U.S. real wage and employment data used by Baumeister and Hamilton (2015).

Code Chunk 2

```
> nlags <- 8
> itr <- 200000
> burn <- 0
> thin <- 20
> acc <- TRUE
> h <- 20
> cri <- 0.95
```

`nlags` from Code Chunk 2 sets the lag length used in the SBVAR model. `itr` sets the number of iterations for the algorithm. `burn` is the number of draws to throw out at the beginning of the algorithm. `thin` sets the thinning parameter which will thin the Markov chains. `acc` indicates whether accumulated impulse responses are to be computed and returned. `h` indicates the time horizon for computing impulse responses. `cri` indicates the credibility intervals to be returned. A `cri` value of 0.95 will return 95% credibility intervals.

Code Chunk 3

```
> pA <- array(data = NA, dim = c(ncol(y), ncol(y), 8))
> pA[, , 1] <- c(0, NA, 0, NA)
> pA[, , 2] <- c(1, NA, -1, NA)
> pA[, , 3] <- c(0.6, 1, -0.6, 1)
> pA[, , 4] <- c(0.6, NA, 0.6, NA)
> pA[, , 5] <- c(3, NA, 3, NA)
> pA[, , 6] <- c(NA, NA, NA, NA)
> pA[, , 7] <- c(NA, NA, 1, NA)
> pA[, , 8] <- c(2, NA, 2, NA)
```

The lines from Code Chunk 3 create an array containing all the information needed to set priors for each element in A. Each column contains the prior information for the parameters in each equation. The third dimension of `pA` should always have a length of 8. The first slice of the third dimension of `pA` indicates the prior distribution (NA - no prior, 0 - symmetric

t-distribution, 1 - non-central t-distribution, 2 - inverted beta distribution, 3 - beta distribution). The second slice indicates sign restrictions (NA - no restriction, 1 - positive restriction, -1 - negative restriction). The third slice indicates the position of the prior. The fourth slice indicates the scale or confidence in the prior for t-distributions and shape1 (α) parameter for inverted beta and beta distributions. The fifth slice indicates the degrees of freedom for t-distributions and shape2 (β) parameter for the inverted beta and beta distributions. The sixth slice indicates skew for non-central t-distributions. The seventh slice indicates priors for long-run restrictions (NA - no long-run restriction, 1 - long-run restriction). The eighth slice indicates the random-walk proposal scale parameters which adjust the algorithm's acceptance rate and the ability of the algorithm to adequately cover the model's parameter space. For information about priors for A see Baumeister and Hamilton (2015/2017/2018). The functions used to compute the density of the prior distributions for A, $det(A)$, and $H(A^{-1})$ are listed in the Appendix.

Code Chunk 4

```
> pP <- matrix(data = 0, nrow = ((nlags * ncol(pA)) + 1), ncol = ncol(pA))
> pP[1:nrow(pA), 1:ncol(pA)] <-
+   diag(x = 1, nrow = nrow(pA), ncol = ncol(pA))
> x1 <-
+   matrix(data = NA, nrow = (nrow(y) - nlags),
+          ncol = (ncol(y) * nlags))
> for (k in 1:nlags) {
+   x1[, (ncol(y) * (k - 1) + 1):(ncol(y) * k)] <-
+     y[(nlags - k + 1):(nrow(y) - k),]
+ }
> x1 <- cbind(x1, 1)
> colnames(x1) <-
+   c(paste(rep(colnames(y), nlags),
+           "_L",
+           sort(rep(seq(from = 1, to = nlags, by = 1), times = ncol(y)),
+                decreasing = FALSE),
+           sep = ""),
+     "cons")
> y1 <- y[(nlags + 1):nrow(y),]
> ee <- matrix(data = NA, nrow = nrow(y1), ncol = ncol(y1))
> for (i in 1:ncol(y1)) {
+   xx <- cbind(x1[, seq(from = i, to = (ncol(x1) - 1), by = ncol(y1))], 1)
+   yy <- matrix(data = y1[, i], ncol = 1)
+   phi <- solve(t(xx) %*% xx, t(xx) %*% yy)
+   ee[, i] <- yy - (xx %*% phi)
+ }
> somega <- (t(ee) %*% ee) / nrow(ee)
> lambda0 <- 0.2
> lambda1 <- 1
```

```
> lambda3 <- 100
> v1 <- matrix(data = (1:nlags), nrow = nlags, ncol = 1)
> v1 <- v1^((-2) * lambda1)
> v2 <- matrix(data = diag(solve(diag(diag(somega)))), ncol = 1)
> v3 <- kronecker(v1, v2)
> v3 <- (lambda0^2) * rbind(v3, (lambda3^2))
> v3 <- 1 / v3
> pP_sig <- diag(x = c(v3), nrow = nrow(v3), ncol = nrow(v3))
```

The lines from Code Chunk 4 create matrices containing prior position (`pP`) and scale or confidence (`pP_sig`) information for the reduced form lagged coefficient matrix Φ. `pP` and `pP_sig` correspond to the P and M^{-1} matrices from Equation 14, respectively. Variance estimates from univariate Autoregression models, `lambda0`, `lambda1`, and `lambda3` are used to construct `pP_sig` in this example. `lambda0` controls the overall confidence in the priors, `lambda1` controls the confidence in higher order lags, and `lambda3` controls the confidence in the constant term. For information about priors for Φ and B see Baumeister and Hamilton (2015/2017/2018), Doan, Sims, and Zha (1984), Doan (2018), Litterman (1986), and Sims and Zha (1998).

Code Chunk 5

```
> pR_sig <-
+   array(data = 0,
+         dim = c(((nlags * ncol(y)) + 1),
+                 ((nlags * ncol(y)) + 1),
+                 ncol(y)))
> Ri <-
+   cbind(kronecker(matrix(data = 1, nrow = 1, ncol = nlags),
+                   matrix(data = c(1, 0), nrow = 1)),
+         0)
> pR_sig[, , 2] <- (t(Ri) %*% Ri) / 0.1
> kappa1 <- matrix(data = 2, nrow = 1, ncol = ncol(y))
```

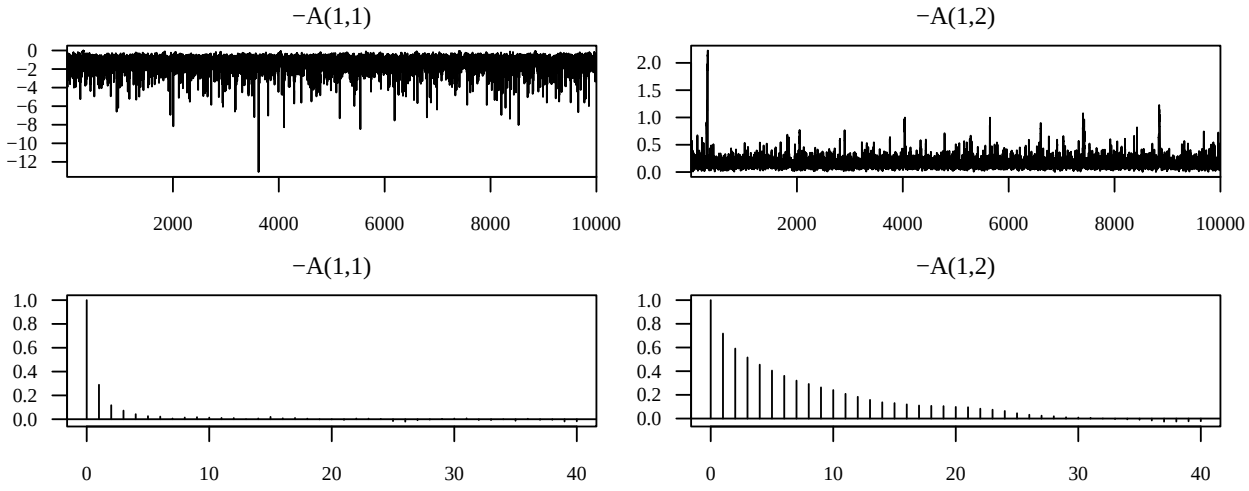
The lines from Code Chunk 5 create an array (`pR_sig`) containing values indicating confidence in the priors for the long-run restrictions. `pR_sig` corresponds to the V_i^{-1} matrix from Equation 14. The matrix R from Equation 14 will be created automatically by the `BH_SBVAR()` function. The length of the third dimension of `pR_sig` is equal to the number of endogenous variables or the number of equations in the model. The first slice of the third dimension contains all zeros since there are no long-run restrictions in the first equation of the SBVAR model for this example. The second slice contains values indicating the confidence in the prior for the long-run restriction assigned to the lagged parameters in the second equation of the SBVAR model for this example. For information about long-run restrictions see Baumeister and Hamilton (2015/2018) and Blanchard and Quah (1989). `kappa1` is a $(1 \times n)$ matrix whose values correspond to the elements along the main diagonal of κ from Equation 17 and indicates the confidence in prior information about the structural variances

in D. Additional information required to set priors for D (τ) will be created automatically by the `BH_SBVAR()` function following Baumeister and Hamilton (2015/2017/2018).

Code Chunk 6

```
> par(cex.axis = 0.8, cex.main = 1, font.main = 1, family = "serif",
+     mfrow = c(2, 2), mar = c(2, 2.2, 2, 1), las = 1)
> results1 <-
+   BH_SBVAR(y = y, nlags = nlags, pA = pA, pP = pP, pP_sig = pP_sig,
+            pR_sig = pR_sig, kappa1 = kappa1, itr = itr,
+            burn = burn, thin = thin, cri = cri)
```

Figure 1
Line and Autocorrelation Diagnostic Plots of the Posterior Estimates



The first line in Code Chunk 6 sets the parameters used to display plots that will be created by the `BH_SBVAR()` function. The `BH_SBVAR()` function allows the user to include prior information for A, $det(A)$, H (A^{-1}), Φ, and D directly when estimating the parameters of an SBVAR model. The `pdetA` and `pH` arguments are arrays containing prior information for $det(A)$ and the elements of H but are not included in this example. The `BH_SBVAR()` function returns a list that includes the acceptance rate (`accept_rate`) of the algorithm, a matrix containing the endogenous variables (`y`), a matrix containing lags of the endogenous variables (`x`), a numeric value indicating the number of lags, and the prior information provided directly or indirectly to the function (`pA`, `pdetA`, `pH`, `pP`, `pP_sig`, `pR_sig`, `tau1`, and `kappa1`). A matrix containing the starting values of the model parameters in A from an optimization routine is returned (`A_start`). Arrays containing estimates of the model parameters are returned (`A`, `detA`, `H`, `B`, `Phi`). The first, second, and third slices of the third dimension of these arrays are lower, median, and upper bounds of the estimates, respectively. Lists containing horizontal and vertical axis coordinates of posterior densities, for the estimates of the parameters in A, $det(A)$, and H with priors, are returned (`A_den`, `detA_den`, and `H_den`).

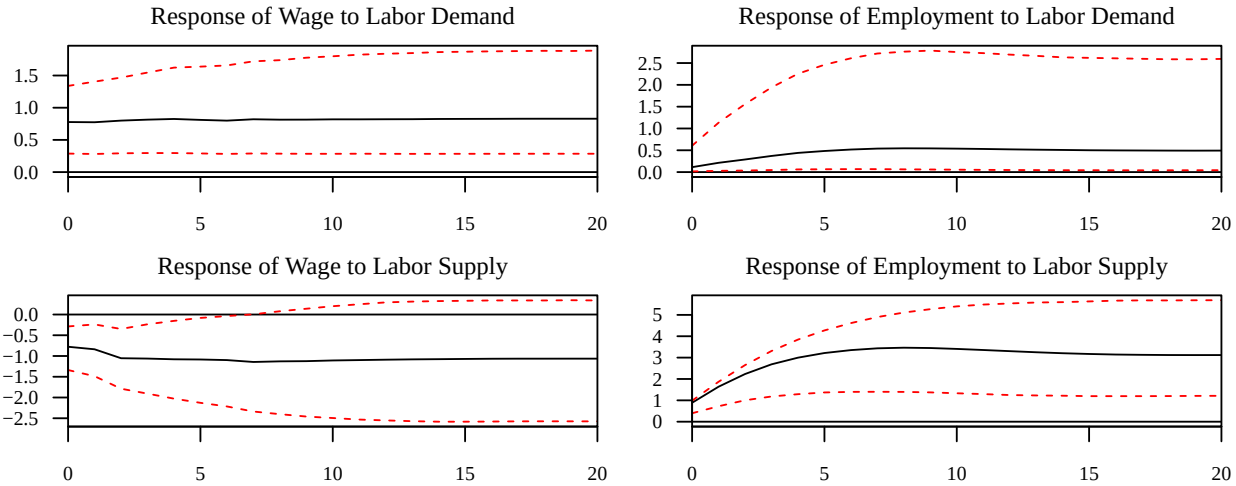
Raw estimates of the elements of A, B, D, $detA$, and H are returned (`A_chain`, `B_chain`, `D_chain`, `detA_chain`, `H_chain`). In addition, line and autocorrelation plots of the Markov chains of A are returned for diagnostic purposes. The line and autocorrelation plots provide an indication of how well the algorithm covers the model's parameter space. The line plots in Figure 1 display the Markov chains of the estimates from the algorithm with the estimate values shown on the vertical axis and the iteration number shown on the horizontal axis. The autocorrelation plots in Figure 1 displays the autocorrelation of the Markov chains of the estimates from the algorithm with the correlation estimates on the vertical axis and the lag length shown on the horizontal axis.
The titles of the plots in Figure 1 indicate the element of the coefficient matrix that is plotted. The plots of the estimated parameters in A are automatically multiplied by -1 to illustrate elasticity values and/or isolate the dependent variable for each equation. These elements correspond to those found in the results from running the `BH_SBVAR()` function and the transpose of those from the mathematical representation from Equation 13. In other words, each column of the coefficient matrix arrays in the resulting list object from running the `BH_SBVAR()` function contain coefficient estimates for each equation. However, each row of the coefficient matrices from the mathematical representation described in Equation 13 represent the parameters of each equation.

Code Chunk 7

```
> irf <- IRF(results = results1, h = h, acc = acc, cri = cri)
> varnames <- colnames(USLMData)[2:3]
> shocknames <- c("Labor Demand","Labor Supply")
> par(cex.axis = 0.8, cex.main = 1, font.main = 1, family = "serif",
+     mfrow = c(2, 2), mar = c(2, 2.2, 2, 1), las = 1)
> irf_results <-
+   IRF_Plots(results = irf, varnames = varnames,
+             shocknames = shocknames)
```

The first line in Code Chunk 7 provides the impulse response estimates. Lines two and three in Code Chunk 7 store the names of endogenous variables and structural shocks. The `IRF_Plots()` function creates plots of impulse responses. This function can be used to display the response of the endogenous variables following a structural shock. The `results` argument is a list object containing the unaltered results from the `BH_SBVAR()` function. The `varnames` and `shocknames` argument are character vectors containing the variable names and shock names, respectively. The `xlab` and `ylab` arguments are not included in this example, but they allow the user to include labels for the horizontal and vertical axes, respectively. Figure 2 displays the cumulative response of U.S. real wage growth and employment growth to U.S. labor demand and supply shocks. The units along the horizontal axis in the plots from Figure 2 represent time periods following an initial shock. The units along the vertical axis in the plots from Figure 2 represent percent change following an initial shock since the endogenous variables included in the model are mean centered quarter over quarter percent change of U.S. real wage and employment. In addition, this function returns a list containing the data used to produce the plots in Figure 2.

Figure 2
Impulse Responses



Note: Horizontal axis indicates time periods following an initial shock. Vertical axis indicates percent change. Black solid lines indicate the posterior median. Red dashed lines indicate credibility intervals.

Code Chunk 8

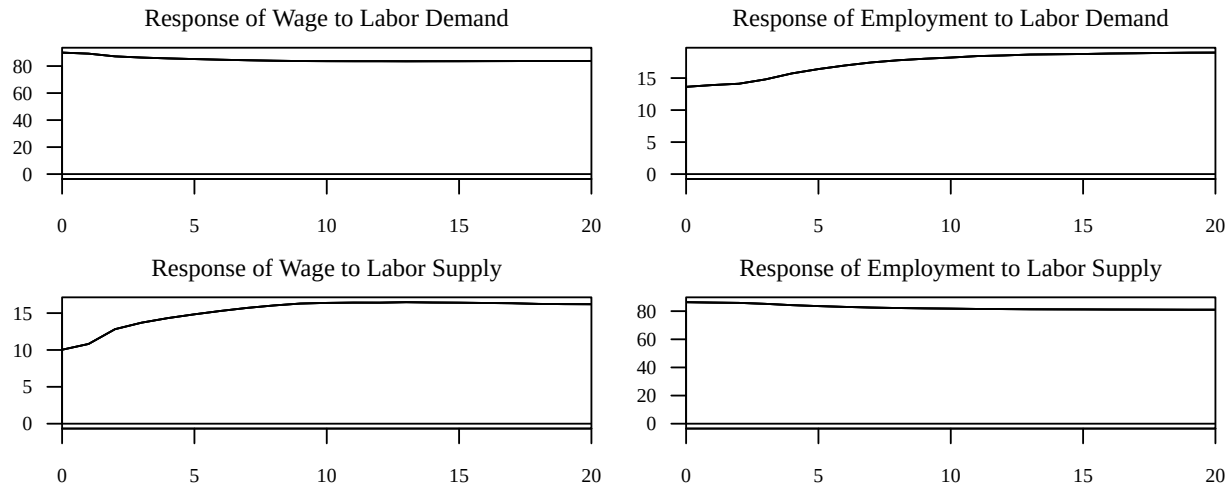
```
> fevd <- FEVD(results = results1, h = h, acc = acc, cri = cri)
> varnames <- colnames(USLMData)[2:3]
> shocknames <- c("Labor Demand","Labor Supply")
> par(cex.axis = 0.8, cex.main = 1, font.main = 1, family = "serif",
+     mfrow = c(2, 2), mar = c(2, 2.2, 2, 1), las = 1)
> fevd_results <-
+   FEVD_Plots(results = fevd, varnames = varnames,
+              shocknames = shocknames)
```

The first line in Code Chunk 8 provides the forecast error variance decomposition estimates. The `FEVD_Plots()` function from Code Chunk 8 creates plots of forecast error variance decompositions. This function can be used to display the forecast error variance that is explained by the structural shocks. The `results`, `varnames`, `shocknames`, `xlab`, `ylab` arguments for the `FEVD_Plots()` function are the same as those from the `IRF_Plots()` function. The `rel` argument is used to display forecast error variance explained by shocks as a percent of total forecast error variance. The units along the horizontal axis in the plots from Figure 3 represent time periods following an initial shock. The units along the vertical axis in the plots from Figure 3 represent forecast error variance explained by the structural shock as a percent of total forecast error variance since `rel = TRUE`. This function returns a list containing the data used to produce the plots in Figure 3.

Code Chunk 9

```
> hd <- HD(results = results1, cri = cri)
> varnames <- colnames(USLMData)[2:3]
> shocknames <- c("Labor Demand","Labor Supply")
```

Figure 3
Forecast Error Variance Decompositions



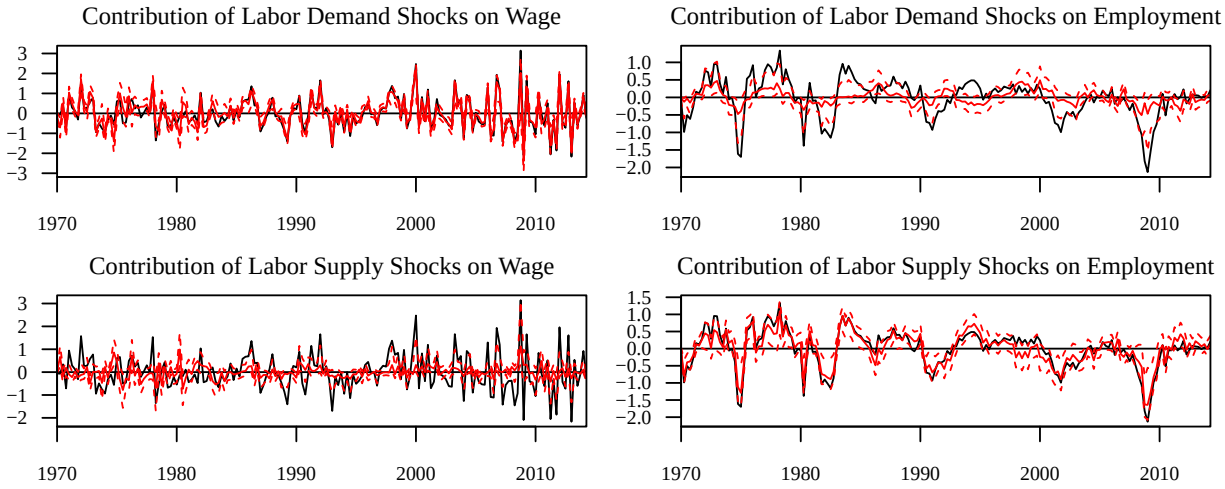
Note: Horizontal axis indicates time periods following an initial shock. Vertical axis indicates forecast error variance explained by the structural shock as a percent of total forecast error variance.

```
> freq <- 4
> start_date <-
+   c(floor(USLMData[(nlags + 1), 1]),
+     (floor(((USLMData[(nlags + 1), 1] %% 1) * freq)) + 1))
> par(cex.axis = 0.8, cex.main = 1, font.main = 1, family = "serif",
+     mfrow = c(2, 2), mar = c(2, 2.2, 2, 1), las = 1)
> hd_results <-
+   HD_Plots(results  = hd, varnames = varnames,
+            shocknames = shocknames,
+            freq = freq, start_date = start_date)
```

The first line in Code Chunk 9 provides the historical decomposition estimates. The `HD_Plots()` function from Code Chunk 9 creates plots of historical decompositions. This function can be used to display the cumulative effect of specific shocks on an endogenous variable at any given time period. The `results`, `varnames`, `shocknames`, `xlab`, `ylab` arguments for the `HD_Plots()` function are the same as those from the `IRF_Plots()` function. Figure 4 displays the historical decompositions. The units along the horizontal axis in the plots from Figure 4, produced by the `HD_Plots` function, represent actual time periods. The units along the horizontal axis of each plot are created with the `freq` and `start_date` arguments. The `freq` argument is set to `4` since the endogenous variables are measured at a quarterly frequency in this example. The `start_date` argument represents the date of the first observation which is Q1 1970 in this example so `start_date <- c(1970, 1)`. The units along the vertical axis in the plots from Figure 2 represent percent change since the endogenous variables included in the model are mean centered quarter over quarter percent change of U.S. real wage and employment. This function also returns a list of the data used to produce the plots in Figure 4.

Code Chunk 10

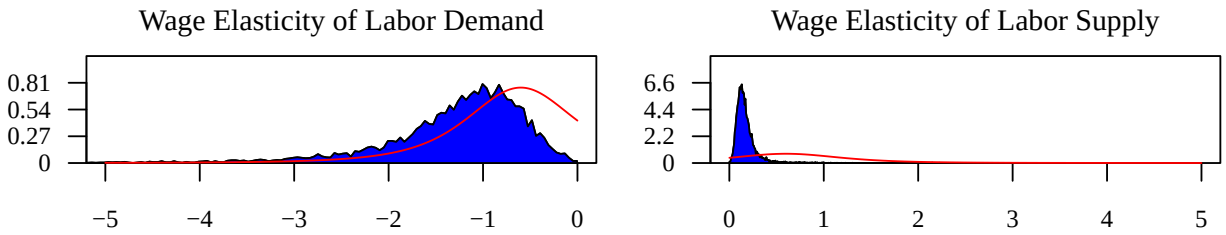
Figure 4
Historical Decompositions



Note: Horizontal axis indicates percent change. Vertical axis indicates actual dates. Black solid lines are mean centered endogenous variables. Red solid lines indicate the posterior median. Red dashed lines indicate credibility intervals.

```
> A_titles <-
+   matrix(data = NA_character_, nrow = dim(pA)[1], ncol = dim(pA)[2])
> A_titles[1, 1] <- "Wage Elasticity of Labor Demand"
> A_titles[1, 2] <- "Wage Elasticity of Labor Supply"
> par(cex.axis = 0.8, cex.main = 1, font.main = 1, family = "serif",
+     mfrow = c(1, 2), mar = c(2, 2.2, 2, 1), las = 1)
> Dist_Plots(results = results1, A_titles = A_titles)
```

Figure 5
Posterior and Prior Distribution Plots



Note: Horizontal axis indicates percent change. Vertical axis indicates density. Blue solid regions indicate posterior density evaluated at values along the horizontal axis. Red solid lines indicate prior density evaluated at values along the horizontal axis.

The `Dist_Plots()` function from Code Chunk 10 creates posterior density plots for the estimates for the parameters in A, H, and $det(A)$. Prior densities are also plotted to illustrate the differences between posterior and prior distributions. The `results`, `xlab`, and `ylab` arguments for the `Dist_Plots()` function are the same as those from the `IRF_Plots()` and `HD_Plots()` functions. `A_titles` and `H_titles` arguments are matrices that contain the titles of the plots. The elements of the `A_titles` and `H_titles` matrices correspond to the elements of the first and second dimensions of the `A` and `H` arrays from the results of

the `BH_SBVAR()` function. The posterior and prior density plots for the estimates of the parameters in A are multiplied by -1 to illustrate elasticity values and/or the value of the coefficient if the dependent variable for each equation were isolated.

APPENDIX

List of functions used to compute the density of the prior distributions at some proposal value:
a1: is the proposal value.
p1: is the prior position parameter.
sigma1: is the prior confidence in the position parameter, c1.
nu: is the degrees of freedom.
lam1: is the non-centrality or skew parameter.
sh1: is the shape1 (α) parameter.
sh2: is the shape2 (β) parameter.

Student t-distribution:

```
> density <-
+   dt(x = ((a1 - p1) / sigma1), df = nu, ncp = 0, log = FALSE) / sigma1
```

Non-central Student t-distribution:

```
> density <-
+   dt(x = ((a1 - p1) / sigma1), df = nu, ncp = lam1, log = FALSE) / sigma1
```

Student t-distribution truncated to be positive:

```
> density <-
+   dt(x = ((a1 - p1) / sigma1), df = nu, ncp = 0, log = FALSE) /
+   (sigma1 *
+      (1 - pt(q = ((-p1) / sigma1), df = nu, ncp = 0, lower.tail = TRUE,
+              log.p = FALSE)))
```

Student t-distribution truncated to be negative:

```
> density <- dt(x = ((a1 - p1) / sigma1), df = nu, ncp = 0, log = FALSE) /
+   (sigma1 * pt(q = ((-p1) / sigma1), df = nu, ncp = 0, lower.tail = TRUE,
+                log.p = FALSE))
```

Inverted Beta-distribution (multiply a1 by sign restriction):

```
> density <- 0
> if (a1 >= 1) {
+   density <- exp(
+     ((sh2 - 1) * log((a1 - 1))) +
+       (((-1) * (sh2 + sh1)) * log((1 + (a1 - 1)))) -
+       log(beta(sh2, sh1))
+     )
+ }
```

Beta-distribution (multiply a1 by sign restriction):

```
> density <- dbeta(x = a1, shape1 = sh1, shape2 = sh2, ncp = 0, log = FALSE)
```

REFERENCES

Baumeister, C., & Hamilton, J. D. (2015). Sign restrictions, structural vector autoregressions, and useful prior information. *Econometrica*, 83(5), 1963-1999.

Baumeister, C., & Hamilton, J. D. (2017). Structural interpretation of vector autoregressions with incomplete identification: Revisiting the role of oil supply and demand shocks (No. w24167). National Bureau of Economic Research.

Baumeister, C., & Hamilton, J. D. (2018). Inference in structural vector autoregressions when the identifying assumptions are not fully believed: Re-evaluating the role of monetary policy in economic fluctuations. *Journal of Monetary Economics*, 100, 48-65.

Blanchard, O. J., & Quah, D. (1989). The Dynamic effects of aggregate demand and supply disturbances. *The American Economic Review*, 79(4), 655-673.

Litterman, R. B. (1986). Forecasting with Bayesian vector autoregressions: Five years of experience. *Journal of Business & Economic Statistics*, 4(1), 25-38.

Sims, C. A., & Zha, T. (1998). Bayesian methods for dynamic multivariate models. *International Economic Review*, 39(4), 949-968.

Doan, T., Litterman, R. B., & Sims, C. (1984). Forecasting and conditional projection using realistic prior distributions. *Econometric Reviews*, 3(1), 1-100.

Doan, T. (2018). *RATS User's Guide, Version 10*. https://www.estima.com.