Vector Autoregressive Models

Tyler J. Brough

November 21, 2016

Vector Autoregressive Models

Consider the following two equations:

$$y_{1,t} = ay_{1,t-1} + by_{2,t-1} + v_{1,t}$$
$$y_{2,t} = cy_{1,t-1} + dy_{2,t-1} + v_{2,t}$$

These two equations comprise a vector autoregression (VAR). A VAR is the extension of the autoregressive (AR) model to the case in which there is more than one variable under study. The VAR above is a bivariate VAR(1) (lag-one). Generally, a VAR can consist of K variables and have p lags.

In a VAR model each variable is treated as endogenous. There is one equation for each variable in the system, and each equation consists of lags of it's own variable plus each of the others. Technically speaking, this is called a *reduced form* VAR, which we will distinguish from a *structural VAR* in just a bit.

We can write the VAR(p) model more generally as:

$$\begin{aligned}
y_{1,t} &= \alpha_1 + \sum_{i=1}^{p} a_{1i}y_{1,t-i} + \sum_{i=1}^{p} b_{1i}y_{2,t-i} + \cdots \sum_{i=1}^{p} c_{1i}y_{k,t-i} + v_{1,t} \\
y_{2,t} &= \alpha_2 + \sum_{i=1}^{p} a_{2i}y_{1,t-i} + \sum_{i=1}^{p} b_{2i}y_{2,t-i} + \cdots \sum_{i=1}^{p} c_{2i}y_{k,t-i} + v_{2,t} \\
&\vdots \\
y_{k,t} &= \alpha_k + \sum_{i=1}^{p} a_{ki}y_{1,t-i} + \sum_{i=1}^{p} b_{ki}y_{2,t-i} + \cdots \sum_{i=1}^{p} c_{ki}y_{k,t-i} + v_{k,t}
\end{aligned}$$

We can also write the VAR model in matrix form:

$$Y_t = \Gamma + A_1Y_{t-1} + A_2Y_{t-2} + \cdots + A_pY_{t-p} + V_t$$

Where:

$$Y_t = \begin{bmatrix} y_{1,t} \\ y_{2,t} \\ \vdots \\ y_{k,t} \end{bmatrix}$$

$$A_i = \begin{bmatrix} a_{i,11} & a_{i,12} & \cdots & a_{i,1k} \\ a_{i,21} & a_{i,22} & \cdots & a_{i,2k} \\ \vdots & & & \\ a_{i,k1} & a_{i,k2} & \cdots & a_{i,kk} \end{bmatrix}$$

$$V_t = \begin{bmatrix} v_{1,t} \\ v_{2,t} \\ \vdots \\ v_{k,t} \end{bmatrix}$$

Typical Uses of VAR Models

VARs are the one of the most important econometric tools in empirical macroeconomics and finance. They have three main uses:

1. Forecasting
2. Impulse Response Functions
3. Forecast Error Variance Decomposition

We will focus on using the VAR for calculating Impulse Response Functions (IRFs).

Please see Stock and Watson for a very nice overview of VAR modeling.

VAR Estimation

It turns out that the VAR model, at least as we have presented it here, is a special case of a more general model called a Seemingly Unrelated Regressions (SUR) Model. The VAR is a SUR model for which the right-hand side variables are all the same in each equation. When that is the case there is no added benefit from systems estimation, so we can estimate the VAR model by applying OLS regression equation-by-equation.

An example of estimating a VAR model in R will be given below.

Impulse Response Functions

In applied work, it is often of interest to know the *response* of one variable to *impulse* in another variable in a system containing a number of further variables. If there is a reaction in the one variable due to an impulse in the other variable one might call the latter causal for the former. This kind of analysis is sometimes called *dynamic multiplier analysis*.

Imagine we are given the following three-variable lag-three VAR model:

$$\begin{bmatrix} y_{1,t} \\ y_{2,t} \\ y_{3,t} \end{bmatrix} = \begin{bmatrix} .5 & 0 & 0 \\ .1 & .1 & .3 \\ 0 & .2 & .3 \end{bmatrix} \begin{bmatrix} y_{1,t-1} \\ y_{2,t-1} \\ y_{3,t-1} \end{bmatrix} + \begin{bmatrix} u_{1,t} \\ u_{2,t} \\ u_{3,t} \end{bmatrix}$$

Note: Here we are not worrying about estimation, and we are assuming we know the "true" form of the A_1 matrix.

Tracing out a unit shock in the first variable in period $t = 0$ in this system we get

$$y_0 = \begin{bmatrix} y_{1,0} \\ y_{2,0} \\ y_{3,0} \end{bmatrix} = \begin{bmatrix} u_{1,0} \\ u_{2,0} \\ u_{3,0} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$y_1 = \begin{bmatrix} y_{1,1} \\ y_{2,1} \\ y_{3,1} \end{bmatrix} = A_1 y_0 = \begin{bmatrix} .5 \\ .1 \\ 0 \end{bmatrix}$$

$$y_2 = \begin{bmatrix} y_{1,2} \\ y_{2,2} \\ y_{3,2} \end{bmatrix} = A_1 y_1 = A_1^2 y_0 = \begin{bmatrix} .25 \\ .06 \\ .02 \end{bmatrix}$$

Continuing in this manner, it turns out that $y_i = (y_{1,i}, y_{2,i}, y_{3,i})'$ is just the first column of A_1^i. It turns out that a unit shock in $y_{2,t}$ ($y_{3,t}$) at $t = 0$ after i periods, results in a vector y_i which is just the second (third) column of A_1^i. Thus, the elements of A_1^i represent the effects of the unit shocks in the variables of the system after i periods. Therefore, they are called impulse responses or dynamic multipliers.

This is the simplest way to see how impulse responses work, but usually they are calculated from the Vector Moving Average form of the VAR model. Just as with a univariate $AR(p)$ model, a $VAR(p)$ model can be inverted to an infinite-order $VMA(\infty)$ model. The coeficients of the VMA model are the dynamic multipliers.

VAR Modeling in R

```
suppressMessages(require(vars))
```

We can look at the use of VAR models in R with the `vars` package and its builtin data set on the Canadian macroeconomy.

```
data(Canada)
```

First let's use the AIC to specify a proper number of lags, choosing 10 as the maximum possible lag:

```
VARselect(Canada, lag.max = 10, type = "const")
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      3      2      1      3
##
## $criteria
##                  1            2            3            4           5
## AIC(n) -6.191599834 -6.621627919 -6.709002047 -6.512701777 -6.30174681
## HQ(n)  -5.943189052 -6.174488511 -6.063134014 -5.668105118 -5.25842152
## SC(n)  -5.568879538 -5.500731387 -5.089929279 -4.395452772 -3.68632157
## FPE(n)  0.002048239  0.001337721  0.001237985  0.001534875  0.00195439
##                  6            7            8            9           10
## AIC(n) -6.194596715 -6.011720944 -6.054479536 -5.912126222 -5.867271844
## HQ(n)  -4.952542805 -4.570938409 -4.414968375 -4.073886435 -3.830303432
## SC(n)  -3.080995238 -2.399943231 -1.944525586 -1.303996035 -0.760965421
## FPE(n)  0.002278812  0.002924622  0.003073249  0.004015164  0.004961704
```

The AIC selects a $p = 3$ order model, which we can now fit as follows:

```
fit <- VAR(Canada, p = 3, type = "const")
summary(fit)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: e, prod, rw, U
## Deterministic variables: const
## Sample size: 81
## Log Likelihood: -150.609
## Roots of the characteristic polynomial:
## 1.004 0.9283 0.9283 0.7437 0.7437 0.6043 0.6043 0.5355 0.5355 0.2258 0.2258 0.1607
```

```
## Call:
## VAR(y = Canada, p = 3, type = "const")
##
##
## Estimation results for equation e:
## ==================================
## e = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + e.l3 + prod.l3 + rw.l3 + U.l3 + c
##
##           Estimate Std. Error t value Pr(>|t|)
## e.l1       1.75274    0.15082  11.622  < 2e-16 ***
## prod.l1    0.16962    0.06228   2.723 0.008204 **
## rw.l1     -0.08260    0.05277  -1.565 0.122180
## U.l1       0.09952    0.19747   0.504 0.615915
## e.l2      -1.18385    0.23517  -5.034 3.75e-06 ***
## prod.l2   -0.10574    0.09425  -1.122 0.265858
## rw.l2     -0.02439    0.06957  -0.351 0.727032
## U.l2      -0.05077    0.24534  -0.207 0.836667
## e.l3       0.58725    0.16431   3.574 0.000652 ***
## prod.l3    0.01054    0.06384   0.165 0.869371
## rw.l3      0.03824    0.05365   0.713 0.478450
## U.l3       0.34139    0.20530   1.663 0.100938
## const   -150.68737   61.00889  -2.470 0.016029 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.3399 on 68 degrees of freedom
## Multiple R-Squared: 0.9988,  Adjusted R-squared: 0.9985
## F-statistic:  4554 on 12 and 68 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation prod:
## =====================================
## prod = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + e.l3 + prod.l3 + rw.l3 + U.l3
##
##           Estimate Std. Error t value Pr(>|t|)
## e.l1      -0.14880    0.28913  -0.515   0.6085
## prod.l1    1.14799    0.11940   9.615 2.65e-14 ***
## rw.l1      0.02359    0.10117   0.233   0.8163
## U.l1      -0.65814    0.37857  -1.739   0.0866 .
## e.l2      -0.18165    0.45083  -0.403   0.6883
## prod.l2   -0.19627    0.18069  -1.086   0.2812
## rw.l2     -0.20337    0.13337  -1.525   0.1319
## U.l2       0.82237    0.47034   1.748   0.0849 .
## e.l3       0.57495    0.31499   1.825   0.0723 .
## prod.l3    0.04415    0.12239   0.361   0.7194
## rw.l3      0.09337    0.10285   0.908   0.3672
## U.l3       0.40078    0.39357   1.018   0.3121
## const   -195.86985  116.95813  -1.675   0.0986 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.6515 on 68 degrees of freedom
```

```
## Multiple R-Squared:  0.98,   Adjusted R-squared: 0.9765
## F-statistic: 277.5 on 12 and 68 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation rw:
## ===================================
## rw = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + e.l3 + prod.l3 + rw.l3 + U.l3 +
##
##           Estimate Std. Error t value Pr(>|t|)
## e.l1    -4.716e-01  3.373e-01  -1.398    0.167
## prod.l1 -6.500e-02  1.393e-01  -0.467    0.642
## rw.l1    9.091e-01  1.180e-01   7.702 7.63e-11 ***
## U.l1    -7.941e-04  4.417e-01  -0.002    0.999
## e.l2     6.667e-01  5.260e-01   1.268    0.209
## prod.l2 -2.164e-01  2.108e-01  -1.027    0.308
## rw.l2   -1.457e-01  1.556e-01  -0.936    0.353
## U.l2    -3.014e-01  5.487e-01  -0.549    0.585
## e.l3    -1.289e-01  3.675e-01  -0.351    0.727
## prod.l3  2.140e-01  1.428e-01   1.498    0.139
## rw.l3    1.902e-01  1.200e-01   1.585    0.118
## U.l3     1.506e-01  4.592e-01   0.328    0.744
## const   -1.167e+01  1.365e+02  -0.086    0.932
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.7601 on 68 degrees of freedom
## Multiple R-Squared: 0.9989,  Adjusted R-squared: 0.9987
## F-statistic:  5239 on 12 and 68 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation U:
## ==================================
## U = e.l1 + prod.l1 + rw.l1 + U.l1 + e.l2 + prod.l2 + rw.l2 + U.l2 + e.l3 + prod.l3 + rw.l3 + U.l3 + c
##
##          Estimate Std. Error t value Pr(>|t|)
## e.l1     -0.61773    0.12508  -4.939 5.39e-06 ***
## prod.l1  -0.09778    0.05165  -1.893 0.062614 .
## rw.l1     0.01455    0.04377   0.332 0.740601
## U.l1      0.65976    0.16378   4.028 0.000144 ***
## e.l2      0.51811    0.19504   2.656 0.009830 **
## prod.l2   0.08799    0.07817   1.126 0.264279
## rw.l2     0.06993    0.05770   1.212 0.229700
## U.l2     -0.08099    0.20348  -0.398 0.691865
## e.l3     -0.03006    0.13627  -0.221 0.826069
## prod.l3  -0.01092    0.05295  -0.206 0.837180
## rw.l3    -0.03909    0.04450  -0.879 0.382733
## U.l3      0.06684    0.17027   0.393 0.695858
## const   114.36732   50.59802   2.260 0.027008 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.2819 on 68 degrees of freedom
```

```
## Multiple R-Squared: 0.9736,  Adjusted R-squared: 0.969
## F-statistic: 209.2 on 12 and 68 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##             e     prod       rw        U
## e     0.11550 -0.03161 -0.03681 -0.07034
## prod -0.03161  0.42449  0.05589  0.01494
## rw   -0.03681  0.05589  0.57780  0.03660
## U    -0.07034  0.01494  0.03660  0.07945
##
## Correlation matrix of residuals:
##            e     prod      rw        U
## e     1.0000 -0.14276 -0.1425 -0.73426
## prod -0.1428  1.00000  0.1129  0.08136
## rw   -0.1425  0.11286  1.0000  0.17084
## U    -0.7343  0.08136  0.1708  1.00000
```

As you can see the `summary` function gives a lot of output information. We can see the details of the dataset as follows:

```
help(Canada)
```

```
## starting httpd help server ...
```

```
##  done
```

Let's look at the effects of a unit shock in the `prod` variable to the `e` variable by calculating the impulse response function out just a few steps for presentation purposes:

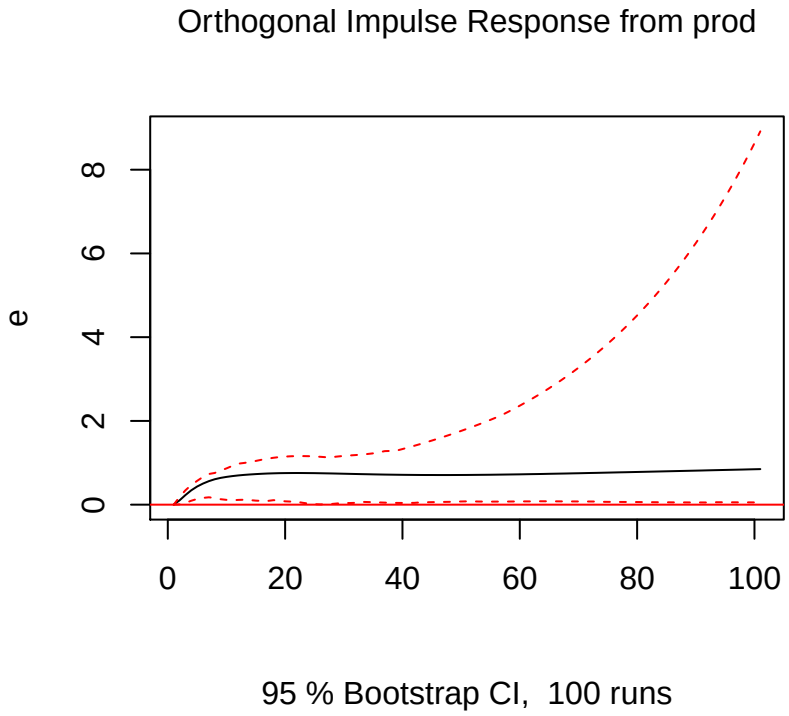
```
irf(fit, impulse = "prod", response = "e", n.ahead = 10)
```

```
##
## Impulse response coefficients
## $prod
##                e
##  [1,] 0.0000000
##  [2,] 0.1028456
##  [3,] 0.2287931
##  [4,] 0.3425870
##  [5,] 0.4329710
##  [6,] 0.5037443
##  [7,] 0.5618587
##  [8,] 0.6068535
##  [9,] 0.6391080
## [10,] 0.6631028
## [11,] 0.6823949
##
##
## Lower Band, CI= 0.95
## $prod
##                e
##  [1,] 0.00000000
##  [2,] 0.03818644
##  [3,] 0.07684938
##  [4,] 0.10091314
```

```
##  [5,] 0.13408008
##  [6,] 0.18176122
##  [7,] 0.21957535
##  [8,] 0.23585279
##  [9,] 0.22397143
## [10,] 0.18730866
## [11,] 0.15930541
##
##
## Upper Band, CI= 0.95
## $prod
##               e
##  [1,] 0.0000000
##  [2,] 0.1617215
##  [3,] 0.3397303
##  [4,] 0.4732099
##  [5,] 0.5400729
##  [6,] 0.6044519
##  [7,] 0.6684665
##  [8,] 0.7127827
##  [9,] 0.7381446
## [10,] 0.7565041
## [11,] 0.7736803
```

We can also plot the IRF:

```r
irf.e <- irf(fit, impulse = "prod", response = "e", n.ahead = 100)
plot(irf.e)
```



As you can see, R makes VAR modeling simple. The key trick is understanding what you are doing!