The sparsevar package

Monica Billio
Università di Venezia

Lorenzo Frattarolo
Università di Venezia

Simone Vazzoler
Università di Venezia

Abstract

The **sparsevar** package it is useful to estimate VAR/VECM models under the sparsity assumption.

Keywords: VAR, VECM, sparse, high-dimensional.

1. Introduction

sparsevar is an R (R Core Team 2016) package that estimates sparse VAR and VECM model using penalized least squares methods (PLS): it is possible to use ENET (Friedman, Hastie, and Tibshirani 2010), SCAD (Breheny and Huang 2011) or MC+ penalties. The sparsity parameter can be estimated using cross-validation, repeated cross-validation or time slicing (Hyndman and Athanasopoulos 2013). When using ENET it is possible to estimate VAR(1) of dimension up to 200, while when using one of the other two is better not to go beyond 50. When estimating a VAR(p) model then the limits are roughly $200/p$ and $50/p$, respectively.

The authors of **sparsevar** are Monica Billio, Lorenzo Frattarolo and Simone Vazzoler and the R package is mantained by Simone Vazzoler. This article describes the usage of **sparsevar** in R. (Lütkepohl 2007), (Basu and Michailidis 2015).

2. VAR/VECM models

Definition 2.1. *A VAR(p) process is a model of the following type:*

$$y_t = \nu + A_1 y_{t-1} + A_2 y_{t-2} + \ldots + A_p y_{t-p} + \varepsilon_t \tag{2.1}$$

for $t = 0, \pm 1, \pm 2, \ldots$, where $y_t = (y_{1t}, \ldots, y_{Kt})^T \in \mathbb{R}^K$ is a $K \times 1$ random vector, the A_i are fixed (i.e. not varying with time) $K \times K$ real matrices, $\nu = (\nu_1, \ldots, \nu_K)^T$ is a fixed vector for the mean $\mathbb{E}(y_t)$ and $\varepsilon_t = (\varepsilon_{1t}, \ldots, \varepsilon_{Kt})^T$ is a K-dimensional white noise (or innovation process) with $\mathbb{E}(\varepsilon_t) = 0$, $\mathbb{E}(\varepsilon_t \varepsilon_t^T) = \Sigma_\varepsilon$ and $\mathbb{E}(\varepsilon_t \varepsilon_s^T) = 0$ for $s \neq t$.

We recall that a VAR(1) process is called **stable** if $\rho(A_1) < 1$ where

$$\rho(A) = \max\{|\lambda_1|, \ldots, |\lambda_K|\} \tag{2.2}$$

with $\lambda_j \in \mathrm{spec}(A)$. In other words a VAR(1) process is called stable if the matrix A_1 has all the eigenvalues inside the unit circle. It is a well known fact that this is equivalent to saying that a VAR(1) process is stable if and only if $\det(\mathbb{I}_K - A_1 z) \neq 0$ for $|z| \leq 1$. We can generalize this condition to a VAR(p) process: if $y_t \sim$ VAR(p) as in Equation (2.1) then we can rewrite it as

$$Y_t = \boldsymbol{\nu} + \boldsymbol{A} Y_{t-1} + U_t \tag{2.3}$$

where

$$Y_t = \begin{pmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-p+1} \end{pmatrix} \in \mathbb{R}^{Kp}; \quad \boldsymbol{\nu} = \begin{pmatrix} \nu \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{Kp}$$

$$\boldsymbol{A} = \begin{pmatrix} A_1 & A_2 & \dots & A_{p-1} & A_p \\ \mathbb{I}_K & \mathbb{O} & \dots & \mathbb{O} & \mathbb{O} \\ \mathbb{O} & \mathbb{I}_K & \dots & \mathbb{O} & \mathbb{O} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbb{O} & \mathbb{O} & \dots & \mathbb{I}_K & \mathbb{O} \end{pmatrix} \in \mathrm{M}_{Kp \times Kp}(\mathbb{R}); \quad U_t = \begin{pmatrix} u_t \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{Kp}$$

So Y_t is a VAR(1) process that is stable if the usual condition on VAR(1) processes holds true:

$$\det\left(\mathbb{I}_{Kp} - \boldsymbol{A} z\right) \neq 0 \quad \text{for } |z| \leq 1$$

Proposition 2.2.

$$\det\left(\mathbb{I}_{Kp} - \boldsymbol{A} z\right) \neq 0 \quad \textit{for } |z| \leq 1 \tag{2.4}$$

if and only if

$$\det\left(\mathbb{I}_K - A_1 z - A_2 z^2 - \dots - A_p z^p\right) \neq 0 \quad \textit{for } |z| \leq 1 \tag{2.5}$$

Proof. [...]

□

2.1. Granger Causality

Granger Causality (GC) is based on the hypothesis that if a variable x influences a variable z, then it should be useful in predicting z.

Definition 2.3. *Let $z_t(h|\Omega_t)$ be the minimum MSE h-step predictor for the process z_t at origin t, based on the information contained in Ω_t. Denote with $\Sigma_z(h|\Omega_t)$ the forecast. The process x_t is said to GC z_t if*

$$\Sigma_z(h|\Omega_t) < \Sigma_z(h|\Omega_t \setminus \{x_s | s \leq t\}) \quad \textit{for at least one } h = 1, 2, \dots \tag{2.6}$$

In other words, if z_t can be predicted more efficiently if the information in the x_t process is taken into account in addition to all other information in the universe, then x_t is Granger Causal (GC) for z_t and we will write $x_t \overset{GC}{\Longrightarrow} z_t$. We now extend the previous definition to the multi-dimensional case.

Definition 2.4. *Let z_t and x_t be M and N dimensional processes respectively. x_t is said to GC z_t if*

$$\Sigma_z(h|\Omega_t) \neq \Sigma_z(h|\Omega_t \setminus \{x_s | s \leq t\}) \tag{2.7}$$

and

$$\Sigma_z(h|\Omega_t) \leq \Sigma_z(h|\Omega_t \setminus \{x_s|s \leq t\}) \tag{2.8}$$

that is, the difference must be positive semidefinite.

Definition 2.5. *We say that there is Instantaneous Granger Causality (IGC) between z_t and x_t if*

$$\Sigma_z(1|\Omega_t \cup \{x_{t+1}\}) \leq \Sigma_z(1|\Omega_t) \tag{2.9}$$

2.2. Cointegration

For an univariate AR(1) process given by $y_t = \alpha y_{t-1} + u_t$ the stability condition is $1 - \alpha z \neq 0$ for $|z| \leq 1$ or, equivalently, $|\alpha| < 1$. For $\alpha = 1$ the process is said to be integrated of order one (written as I(1))

$$y_t = y_{t-1} + u_t = y_0 + \sum_{i=1}^{t} u_i \tag{2.10}$$

It is clear from the previous formula that an integrated process is the sum of all past innovations; easily one can compute $\mathbb{E}(y_t) = y_0$ and $\mathrm{Var}(y_t) = t\mathrm{Var}(u_t) = t\sigma_u^2$.
A general AR(p) model is of the form

$$\Phi(L)y_t = \varepsilon_t \tag{2.11}$$

with $\Phi(L) = 1 - \sum_{i=1}^{p} \alpha_i L^i$ (L is the lag operator). If the polinomial $\Phi(z)$ has one root on the unit circle $|z| = 1$, then the behaviour of the time series y_t is similar to a random walk.

Definition 2.6. *An univariate process with d unit roots is called integrated process of order d and is written $I(d)$.*

If there is only one unit root, i.e. the process is I(1), by taking first differences

$$\Delta y_t = y_t - y_{t-1} = (1 - L)y_t \tag{2.12}$$

it is possible to obtain a stationary process. In the same way, if the process y_t is I(d) then considering $\Delta^d = (1 - L)^d y_t$ we get a stationary process.

3. Estimation of the model

In general the estimation of a VAR model is not an easy task (as can be seen for example in (Tsay 2013)). This fact is more true in the high dimensional setting, where $K > n$ (in our notation K is the dimension of the time series and n is the number of observations available). Thus, to proceed further, we make an additional hypothesis on the matrix A of the VAR(1) process:

(**H**) the matrix A is sparse.

This assumption allows us to use and adapt to our problem, the penalized methods in OLS estimation.

Following (Basu and Michailidis 2015), based on the data $\{y^0, \dots, y^T\}$, we construct the following regression problem

$$\begin{pmatrix} (y^T)' \\ \vdots \\ (y^0)' \end{pmatrix} = \begin{pmatrix} (y^{T-1})' & \dots & (y^{T-d})' \\ \vdots & \ddots & \vdots \\ (y^{d-1})' & \dots & (y^0)' \end{pmatrix} \begin{pmatrix} A_1' \\ \vdots \\ A_d' \end{pmatrix} + \begin{pmatrix} (\varepsilon^T)' \\ \vdots \\ (\varepsilon^d)' \end{pmatrix} \tag{3.1}$$

which can be rewritten as

$$\mathrm{vec}(\mathcal{Y}) = \mathrm{vec}(\mathcal{X}\mathcal{B}^*) + \mathrm{vec}(E) \tag{3.2}$$
$$= (\mathbb{I} \otimes \mathcal{X})\mathrm{vec}(\mathcal{B}^*) + \mathrm{vec}(E) \tag{3.3}$$
$$Y = X\beta^* + \mathrm{vec}(E) \tag{3.4}$$

3.1. Threshold

$$\hat{T} = \frac{1}{\sqrt{pN \log(T)}} \tag{3.5}$$

4. Using sparsevar

The two main functions of the package are `estimateVAR` and `estimateVECM`, which, as the names suggest, allow the user to estimate VAR and a VECM models respectively. There are other function included in the package which are useful for creating a sparse VAR process, such as `simulateVAR` and `createSparseMatrix`.

4.1. Using `estimateVAR`

The function is used to estimate a VAR model given the time series `data` and must be called in the following way

```
estimateVAR(data, p = 1, penalty = "ENET", options = list(...))
```

The variable `data` must be a matrix containing `N` different time series in columns with `nobs` observations displaced on different rows; `p` is the order of the VAR model (by default it has value `p = 1`) and `penalty` is a string which can the take the values `"ENET"` (default), `"SCAD"` or `"MCP"`.
The parameter `options` is a list containing global ([...]) and specific options (related to the penalty used in the estimation).
Global options:

- `parallel`: logical; use parallel computing (default `TRUE`);
- `ncores`: integer; number of cores to use if `parallel = TRUE`;
- `threshold`: logical; default is `FALSE`. If `TRUE` [...]

- `scale`: logical; scale the input `data` normalizing every column (default `TRUE`);
- `nfolds`: integer; the number of folds used in the cross validation (default `nfolds = 10`).

Options for `"ENET"`:

- `alpha`: a value in $[0, 1]$; default is `alpha = 1` which uses LASSO penalty, while setting `alpha = 0` uses Ridge regression; using a value in $(0, 1)$ uses a weighted linear combination of the of the two;
- `type.measure`: `"mse"` or `"mae"`; the error used in the cross validation: `"mse"` mean square error (default) or `"mae"` mean absolute error;
- `nlambda`: integer; the number of lambdas to use in the cross validation;
- `repeatedCV`: logical; if `TRUE` the function performs repeated cross validation for model selection;
- `nRepeats`: integer; the number of repeats in the repeated cross validation (if `repeatedCV = TRUE`);
- `foldIDs`: logical; if `TRUE` (default) the folds in the cross validation will be fixed; their ids will be assigned automatically in increasing order;
- `timeSlice`: logical; if `TRUE` [...]
- `leaveOutLast`: integer; the number of elements to leave out of sample when using the time slice option;
- `horizon`: integer; the number of elements to use to compute the out of sample MSE when using the time slice option;
- `fixedWindow`: logical;

Options for `"SCAD"` and `"MCP"`:

- `eps`: positive real number;

4.2. Using `estimateVECM`

Use to estimate a VECM model.

```
estimateVECM(data, p = 2, penalty = "ENET", logScale = TRUE, options = list(...))
```

The options are the same as in `estimateVAR`, except for the global option:

- `logScale`: logical; take the logarithm of the input `data`.

4.3. Using `simulateVAR`

[...]

```
simulateVAR(N = 100, p = 1, sparsity = 0.05, rho = 0.5, ...
            covariance = "toeplitz")
```

5. Simulation examples

```
est <- estimateVAR(data, p = 1, penalty = "ENET")
```

In Tables 1, 2, 3 and 4 are reported the relative errors computed using the Froebenius norm of the simulations performed. More specifically we recall that $\|A\|_F = \sqrt{\mathrm{tr}(A^T A)}$ and the relative error is defined as $\frac{\|\hat{A}-A\|_F}{\|A\|_F}$. We examined 4 cases: $N = 10$, $T = 30$ with `sparsity = 0.05` and $N = 30$, $T = 120$ with `sparsity = 0.05` both with the option `scale = TRUE` and `scale = FALSE`. All the simulations are reproducible and the code used can be found in the folder `simulations/`.

$\|\hat{A} - A\|_F / \|A\|_F$	**Block 1**			**Block 2**			**Toeplitz**		
	0.5	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**
LASSO	0.34 (0.17)	0.41 (0.23)	0.62 (0.61)	0.33 (0.17)	0.37 (0.20)	0.47 (0.30)	0.33 (0.17)	0.36 (0.18)	0.49 (0.35)
LASSO + Threshold	0.34 (0.18)	0.40 (0.23)	0.62 (0.61)	0.32 (0.17)	0.36 (0.20)	0.47 (0.30)	0.32 (0.17)	0.35 (0.18)	0.48 (0.35)
SCAD	0.29 (0.19)	0.35 (0.25)	0.57 (0.72)	0.27 (0.18)	0.30 (0.23)	0.41 (0.36)	0.27 (0.17)		
SCAD + Threshold									
Basu	0.75	0.75	0.76	0.77	0.8	0.87	0.77	0.8	0.88

Accuracy	**Block 1**			**Block 2**			**Toeplitz**		
	0.5	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**
LASSO	0.88 (0.08)	0.88 (0.10)	0.88 (0.12)						
LASSO + Threshold	0.97 (0.03)	0.96 (0.04)	0.94 (0.07)	0.97 (0.03)	0.96 (0.04)	0.95 (0.06)	0.97 (0.02)	0.96 (0.04)	0.94 (0.08)
SCAD									
SCAD + Threshold									

Table 1: Results of 1000 simulations with $N = 10$, $T = 30$, `sparsity = 0.05` and `scale = FALSE`

References

Basu S, Michailidis G (2015). "Regularized estimation in sparse high-dimensional time series models." *Ann. Statist.*, **43**(4), 1535–1567. doi:10.1214/15-AOS1315. URL http://dx.doi.org/10.1214/15-AOS1315.

$\|\hat{A} - A\|_F / \|A\|_F$	Block 1			Block 2			Toeplitz		
	0.5	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**
LASSO	0.71 (0.09)	0.72 (0.11)	0.76 (0.25)	0.71 (0.09)	0.72 (0.10)	0.77 (0.23)	0.71 (0.09)	0.71 (0.09)	0.78 (0.27)
LASSO + Threshold	0.71 (0.09)	0.72 (0.11)	0.76 (0.25)	0.70 (0.09)	0.72 (0.10)	0.78 (0.22)	0.70 (0.09)	0.71 (0.10)	0.78 (0.27)
SCAD	0.64 (0.12)								
SCAD + Threshold	0.64 (0.12)	0.65 (0.14)	0.70 (0.17)						
Basu	0.75	0.75	0.76	0.77	0.8	0.87	0.77	0.8	0.88
Accuracy	**Block 1**			**Block 2**			**Toeplitz**		
	0.5	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**
LASSO	0.87 (0.08)	0.87 (0.09)	0.87 (0.10)	0.87 (0.09)	0.86 (0.11)	0.86 (0.12)	0.86 (0.09)	0.86 (0.10)	0.84 (0.14)
LASSO + Threshold	0.96 (0.03)	0.96 (0.03)	0.95 (0.05)	0.96 (0.03)	0.96 (0.05)	0.94 (0.07)	0.96 (0.03)	0.96 (0.04)	0.93 (0.09)
SCAD	0.89 (0.05)								
SCAD + Threshold	0.97 (0.02)	0.97 (0.02)	0.97 (0.02)						

Table 2: Results of 1000 simulations with $N = 10$, $T = 30$, `sparsity = 0.05` and `scale = TRUE`

	Block 1			Block 2			Toeplitz		
	0.5	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**
LASSO									
LASSO + Threshold									
SCAD									
MC+									
Basu	0.68	0.73	0.80	0.83	0.90	0.98	0.68	0.72	0.85

Table 3: Results of 1000 simulations with $N = 30$, $T = 120$ and `scale = FALSE`

	Block 1			**Block 2**			**Toeplitz**		
	0.5	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**	**0.5**	**0.7**	**0.9**
LASSO									
LASSO + Threshold									
SCAD									
MC+									
Basu	0.68	0.73	0.80	0.83	0.90	0.98	0.68	0.72	0.85

Table 4: Results of 1000 simulations with $N = 30$, $T = 120$ and `scale = TRUE`

Breheny P, Huang J (2011). "Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection." *Annals of Applied Statistics*, **5**(1), 232–253.

Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, **33**(1), 1–22. ISSN 1548-7660. doi:10.18637/jss.v033.i01. URL https://www.jstatsoft.org/index.php/jss/article/view/v033i01.

Hyndman R, Athanasopoulos G (2013). *Forecasting: principles and practice*. OTexts, Melbourne, Australia. URL http://otexts.org/fpp/.

Lütkepohl H (2007). *New Introduction To Multiple Time Series Analysis*. Springer.

R Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Tsay R (2013). *Multivariate Time Series Analysis: With R and Financial Applications*. Wiley Series in Probability and Statistics. Wiley. ISBN 9781118617755. URL https://books.google.it/books?id=A4QVAgAAQBAJ.

Affiliation:

Monica Billio
Department of Economics
Università Ca' Foscari di Venezia
San Giobbe 873, Venezia
E-mail: billio@unive.it
URL: http://venus.unive.it/billio/

Lorenzo Frattarolo
Department of Economics
Università Ca' Foscari di Venezia

San Giobbe 873, Venezia
E-mail: lorenzo.frattarolo@unive.it

Simone Vazzoler
Department of Economics
Università Ca' Foscari di Venezia
San Giobbe 873, Venezia
E-mail: svazzole@gmail.com