Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

spsur: an R package for spatial seemingly unrelated regressions

Abstract

In recent decades, several methodological improvements have been suggested to specify, estimate and validate Seemingly Unrelated Regression (SUR) models in a spatial framework. These new procedures allow us testing for the presence of spatial dependence, and estimate spatial models using different algorithms. Furthermore, new misspecification tests, singularly in a maximum likelihood framework, have also been developed. Unfortunately, no standard, friendly software is easily avaliable. This is the deficit that the **spsur** R package pretends to fill. The new package allows for the estimation of the most popular spatial econometric models by maximum likelihood or instrumental variable procedures. Moreover, **spsur** implements a collection of Lagrange Multipliers and Likelihood Ratios to test for misspecifications in the SUR. Additional functions allow for the estimation of the so-called *spatial impacts* (direct, indirect and total impacts) and also obtains random data sets, of a SUR nature, with the features decided by the user. An important aspect of **spsur** is that it operates both in a pure cross-sectional setting or in panel data sets. We include well-known examples in the applied literature on spatial data to illustrate the main functionalities of the new R package.

Keywords: spatial seemingly unrelated regression models, lagrange multipliers test, maximum likelihood, instrumental variables, panel data.

1. Introduction

Seemingly unrelated regression models (SUR from now on) are a type of multivariate econometric formulation very popular since the seminal papers of **?**, **?** or **?**. SUR have been used in many research areas in economics and others fields (for instance, see **?**), especially in the topics of consumption, production and environmental economics, but also on economic growth, health economics, real estate or competition between local agents. Distinct mechanisms may result in the type of dependence that lies behind SUR models such as omitted variables, unobserved effects or common factors, with a dfferent impact for each equation but

uniform among the individuals. It is clear that if the errors are connected, a SUR framework assures efficiency gains by estimating the system jointly rather than processing each equation separately. These gains explain the relevant position that SUR models occupy in the applied research agenda. Spatial econometrics is, surprisingly, an exception: the applied literature is very short in spite of a quite extensive methodological research.

? introduced the term *spatial SUR* in reference to a model made of *'an equation for each time period, which is estimated for a cross section of spatial units'* (p. 141). The approach of Anselin focuses on the problem of serial dependence in the errors of an equation, which is perfectly natural if the same agents intervene in every period. According to usual practice at the end of the eighties, the proposal of Anselin allows for a very limited heterogeneity. In fact, the regression coefficients are assumed to be the same across individuals, and unobserved effects are excluded. These constraints have been overcomed in the more recent works of **?** and **?**, where spatial and nonspatial parameters can take on different values in different cross-section.

Our case assumes that the model may contain G equations, with spatial structure. Additionally, it is assumed that the cross-sectional dimension (N, number of individuals) is large, and that the time dimension (Tm, number of periods) may increase. **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?** or **?** are well-known examples in this line. If the number of individuals is small and they are observed for a long time period (finite N and large Tm), the inference should be based on the time dimension, for example, specifying an equation for each spatial unit. **?**, **?** and **?** follow the last approach.

A significant contribution is **?**, who allow for greater heterogeneity among the individuals by introducing unobserved random effects. They extend the equation of the errors by including a mechanism of error components 'a la KKP', from **?**, where the unobserved effects are intermingled with the random term of the spatial errors. This scheme is different from the proposal of **?**, BSK, where the error is the sum of a spatial error mechanism plus an unobserved individual effect. Another point of interest in the work of Wang and Kockelman is that they built a standard SUR model, with G equations, Tm cross-sections and N individuals. **?** extend the results of Wang and Kockelman by developing a collection of misspecification tests, under a maximum-likelihood, ML, framework, whereas **?** introduce generalized method of moments, GMM, in a spatial SUR with spatial errors, SEM, also 'a la KKP'. Finally **?** address the problem of selecting the best specification for spatial SUR models, using a battery of Lagrange Multipliers obtained in a ML framework. **?**, **?**, **?**, **?** and **?** present nice examples in this direction.

From the discussion above, it is clear that there is a growing interest in the case of seemingly unrelated regressions in a spatial context, although the lack of specific software has slowed its adoption in applied research. Our purpose is to introduce a new R package, called **spsur**, to fill this gap. The preferred framework is a large N and an increasing Tm. The core of the package lies in ML estimation because, at this time, we value exactitude and reliability over quickness. There is no need to recall that the hypothesis of normality plays a key role in this approach, which may be difficult to admit in some cases. For that reason, and also to allow for speed of calculus in large samples, the ML approach is completed with a module devoted to IV procedures.

Among the limitations, let us note that we rely explicitly on the assumptions of linearity and dynamic relations are not allowed. This version of the package can remove unobserved

individual effects from the equations, which are not estimated. Finally, we require the user to specify the weighting matrix, which is assumed exogenous, known, time invariant and constant across equations.

Second Section introduces notation and reviews some basic results in the literature on spatial SUR models, using the approach of Anselin. Section 3 presents the structure and some of the main functions included in **spsur**. Section 4 focusses on misspecification tests included in **spsur**. Section 5 is devoted to the issue of spatial multipliers whereas Section 6 presents the general case of a SUR model with G equations, Tm time periods and N individuals. Section 7 introduces an additional functionality that allows for simulation experiments. Finally, Section 8 concludes with a brief summary and prospects for the future.

2. SUR and Spatial SUR models

The SUR model consist of several equations, possibly with different regressors, where the error terms are correlated among equations. We focus on the Anselin's case with a single equation, G=1, several time periods, Tm, and N individuals in the cross-section; we assume that $N>Tm$ (note that the discussion is almost similar if Tm=1 and G>1). Our baseline model, without spatial effects, is the following:

$$\mathbf{y}_t = \mathbf{X}_t\boldsymbol{\beta}_t + \boldsymbol{\varepsilon}_t \; ; \qquad E[\boldsymbol{\varepsilon}_t] = 0 \; ; \qquad E[\boldsymbol{\varepsilon}_t\boldsymbol{\varepsilon}'_s] = \sigma_{ts}\mathbf{I}_N \qquad t,s = 1,...,Tm \tag{1}$$

where $\mathbf{y_t}$, $\mathbf{u}_t$ and $\boldsymbol{\varepsilon}_t$ are $N\times 1$ vectors, $\mathbf{X}_t$ is a $N\times p_t$ matrix, with p_t the number of regressors that appear in the t-th equation and $\boldsymbol{\beta}_t$ the $(p_t \times 1)$ vector of coefficients. We call this model as SUR-SIM, from spatially independent model following the terminology in **?**.

The singular aspect of (1) is that the SUR structure appears because there is intra-individual serial dependence; inter-individual serial dependence is excluded. The serial dependence in the errors is not parameterized, but estimated in the $(Tm\times Tm)$ sampling covariance matrix, $\boldsymbol{\Sigma}$=(σ_{ts}).

The equation of (1) may be enough in some circumstances but not in a spatial setting where a potential problem, often present in applied research, is the existence of spatial dependence. The spatial mechanisms may have different forms (**?**). A general SUR model that include most of the possible spatial effects of interest is:

$$\mathbf{y}_t = \lambda_t\mathbf{W}^y_t\mathbf{y}_t + \mathbf{X}_t\boldsymbol{\beta}_t + \mathbf{W}^x_t\mathbf{X}^*_t\boldsymbol{\gamma}_t + \mathbf{u}_t; \; \mathbf{u}_t = \rho_t\mathbf{W}^u_t\mathbf{u}_t + \boldsymbol{\varepsilon}_t \; ; \;\; E[\boldsymbol{\varepsilon}_t] = 0 \; ; \;\; E[\boldsymbol{\varepsilon}_t\boldsymbol{\varepsilon}'_s] = \sigma_{ts}\mathbf{I}_N \tag{2}$$

where $\mathbf{X}^*_t$ is the matrix of regressors excluded the intercept term; $\mathbf{W}^y_t$, $\mathbf{W}^x_t$, $\mathbf{W}^u_t$ are $(N\times N)$ weighting matrices. The model in (2) is often referred to as a General Nesting Model, SUR-GNM. Several constrained specifications emerge from this equation, such as:

1. The SUR-SIM, spatially independent model, $\lambda_t = \rho_t = \gamma_t = 0, (\forall t)$.
2. The SUR-SLX, spatial lags in Xs model, where $\lambda_t = \rho_t = 0, (\forall t)$.
3. The SUR-SLM, spatial lag model, where $\rho_t = \gamma_t = 0, (\forall t)$.
4. The SUR-SEM, spatial error model, where $\lambda_t = \gamma_t = 0, (\forall t)$.
5. The SUR-SDM, spatial Durbin model, where $\rho_t = 0, (\forall t)$.
6. The SUR-SDEM, spatial Durbin error model, where $\lambda_t = 0, (\forall t)$.

7. The SUR-SARAR, spatial lag model with autoregresive errors, where $\gamma_t = 0, (\forall t)$.

In the applied literature, it is usual to assume that the weighting matrices are the same $\mathbf{W}_t^x = \mathbf{W}_t^u = \mathbf{W}_t^\varepsilon = \mathbf{W}$. The user should supply this matrix, based on apriori knowledge, which, in any case, must conform to he requisites described in **?**: the terms on the main diagonal are zero and the row and column sums of $\mathbf{W}$ are uniformly bounded in absolute value, the same as $\mathbf{A}_t^{-1} = (\mathbf{I}_N - \lambda_t \mathbf{W})^{-1}$ and $\mathbf{B}_t^{-1} = (\mathbf{I}_N - \rho_t \mathbf{W})^{-1}$ which must exist.

The model of (2), for the case of G=1, can be expressed using obvious matrix notation:

$$\mathbf{A}\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + (\mathbf{I}_{Tm} \otimes \mathbf{W})\mathbf{X}^*\boldsymbol{\gamma} + \mathbf{u}; \quad \mathbf{B}\mathbf{u} = \boldsymbol{\varepsilon}; \quad \boldsymbol{\varepsilon} \sim N(0, \boldsymbol{\Omega})$$

$$\mathbf{y} = \underset{TmN\times 1}{\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \cdots \\ \mathbf{y}_{Tm} \end{bmatrix}}; \mathbf{X} = \underset{TmN\times p}{\begin{bmatrix} \mathbf{X}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{X}_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \mathbf{X}_{Tm} \end{bmatrix}}; \boldsymbol{\beta} = \underset{p\times 1}{\begin{bmatrix} \boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2 \\ \cdots \\ \boldsymbol{\beta}_{Tm} \end{bmatrix}}; \mathbf{u} = \underset{TmN\times 1}{\begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \cdots \\ \mathbf{u}_{Tm} \end{bmatrix}}; \boldsymbol{\varepsilon} = \underset{TmN\times 1}{\begin{bmatrix} \boldsymbol{\varepsilon}_1 \\ \boldsymbol{\varepsilon}_2 \\ \cdots \\ \boldsymbol{\varepsilon}_{Tm} \end{bmatrix}} \tag{3}$$

where $p = \sum p_t$, $\mathbf{A} = \mathbf{I}_{TmN} - \boldsymbol{\Lambda} \otimes \mathbf{W}$ with $\boldsymbol{\Lambda} = diag(\lambda_1, \ldots, \lambda_{Tm})$; $\mathbf{B} = \mathbf{I}_{TmN} - \boldsymbol{\Gamma} \otimes \mathbf{W}$ with $\boldsymbol{\Gamma} = diag(\rho_1, \ldots, \rho_{Tm})$ and $\boldsymbol{\Omega} = \boldsymbol{\Sigma} \otimes \mathbf{I}_{TmN}$, being $\otimes$ the Kronecker product. Moreover, $\mathbf{X}^*$ is a *NTm* $\times$ *(p-Tm)* that omits the columns of the intercepts because these terms cannot be spatially lagged. Assuming that the errors are normally distributed, the log-likelihood function of (2) can be written as:

$$l(\mathbf{y}; \boldsymbol{\eta}) = -\frac{NTm}{2}\ln(2\pi) - \frac{N}{2}\ln|\boldsymbol{\Sigma}| + Tm\sum_{t=1}^{Tm}\ln|\mathbf{B}_t| + Tm\sum_{t=1}^{Tm}\ln|\mathbf{A}_t| - \frac{\left(\mathbf{A}\mathbf{y} - \bar{\mathbf{X}}[\boldsymbol{\beta}\ \boldsymbol{\gamma}]'\right)' \mathbf{B}'\boldsymbol{\Omega}^{-1}\mathbf{B}\left(\mathbf{A}\mathbf{y} - \bar{\mathbf{X}}[\boldsymbol{\beta}\ \boldsymbol{\gamma}]'\right)}{2} \tag{4}$$

where $\bar{\mathbf{X}} = [\mathbf{X}, \mathbf{W}\mathbf{X}^*]$ is a $NTm \times (2p\text{-}Tm)$.

The vector of parameters is $\boldsymbol{\eta}' = [\boldsymbol{\beta}'; \boldsymbol{\gamma}'; \lambda_1; \ldots; \lambda_{Tm}; \rho_1; \ldots; \rho_{Tm}; \sigma_{ij}]$, where there are *P= (2p-Tm)+Tm+Tm(Tm+1)/2* unknown coefficients **?** and **?** use numerical optimization techniques to solve the maximum likelihood estimates. Note that the presence of the log of the Jacobian terms, $\ln|\mathbf{B}_t|$ and $\ln|\mathbf{A}_t|$, forces the spatial parameters, λ_t and ρ_t, to lie inside the so-called *stability interval* (if the weighting matrix is row-standardized, a rough approximation is -1,+1). Under the assumption that the spatial SUR model is correctly specified (including normality), the ML estimators are consistent, efficient and asymptotically normally distributed (**?**).

2.1. Testing for spatial effects

The estimation of spatial SUR models can be complex for strong spatial dependence mechanisms, so it is advisable to avoid needless efforts. This is the purpose of the Lagrange Multipliers presented below. As usual in the spatial econometrics literature, our starting point is the SUR-SARAR model.

The first Multiplier is a global test for the presence of spatial effects in the SUR:

$$H_0 : \lambda_t = \rho_t = 0 \quad (\forall t) \qquad vs \qquad H_A : No \;\; H_0 \tag{5}$$

obtained as:

$$\mathrm{LM}^{\mathrm{SUR}}_{\mathrm{SARAR}} = \mathbf{g}'_{(\boldsymbol{\eta})_{|H_0}} \left[\mathbf{I}_{(\boldsymbol{\eta})_{|H_0}}\right]^{-1} \mathbf{g}_{(\boldsymbol{\eta})_{|H_0}} \underset{as}{\sim} \chi^2(2Tm) \tag{6}$$

where $\mathbf{g}_{(\boldsymbol{\eta})_{|H_0}}$ is the score vector of order $(Px1)$, evaluated under the null hypothesis. $\mathbf{I}_{(\boldsymbol{\eta})_{|H_0}}$ is the $(P\mathrm{x}P)$ information matrix of the SUR-SARAR model also under the null of (5); more details in **?**.

If we change the model of the alternative, from the SUR-SARAR to the SUR-SLM, we can test for the significance of the spatial lags:

$$H_0 : \lambda_t = 0 \quad (\forall t) \qquad vs \qquad H_A : No \;\; H_0 \tag{7}$$

The model of the null hypothesis continues to be the SUR-SIM, and it is implicitly assumed that the ρ parameters are zero. The discussion is entirely similar to the test of (6) in the SARAR case, just changing this model by the SLM. The corresponding Lagrange Multiplier is called $\mathrm{LM}^{\mathrm{SUR}}_{\mathrm{SLM}}$.

Similarly, a SEM model can be placed in the alternative, assuming that the λ parameters are zero. We can test for the presence of spatial structure in the errors:

$$H_0 : \rho_t = 0 \quad (\forall t) \qquad vs \qquad H_A : No \;\; H_0 \tag{8}$$

The Lagrange Multiplier associated to the hypothesis of (8) is called $\mathrm{LM}^{\mathrm{SUR}}_{\mathrm{SEM}}$.

? show that the last two *raw* Multipliers, $\mathrm{LM}^{\mathrm{SUR}}_{\mathrm{SLM}}$ and $\mathrm{LM}^{\mathrm{SUR}}_{\mathrm{SEM}}$, are oversized in case of misspecification of the alternative hypothesis. This lack of robustness complicates the identification of the correct specification. Let us note that this problem is not specific of a spatial setting but extends to other cases where there are groups of parameters whose scores are not orthogonal. The solution, as suggested in **?**, is to *robustify* the *raw* Multipliers to obtain the so-called robust Lagrange Multipliers, denoted as $\mathrm{LM}^{*\mathrm{SUR}}_{\mathrm{SLM}}$ and $\mathrm{LM}^{*\mathrm{SUR}}_{\mathrm{SEM}}$ respectively. The size of the robust Multipliers is more balanced at the cost of a small decrease in power; in any case, they are very popular among practitioners in spatial data analysis.

2.2. The IV estimation for spatial SUR models

The ML approach offers an adequate framework for solving the inference of spatial SUR models. However, as the size of the cross-section, N, increases they become highly demanding in terms of computational effort due to the presence of the Jacobian matrices in the likelihoods. Under some circumstances, it may be reasonable to use other algorithms, such as Instrumental Variables, IV.

? and **?** introduced the IV approach as a method to deal with problems of endogeneity. This is the typical situation in a spatial model that includes lags of the explained variable among the regressors. For example, in the SUR-SLM case is inmediate to conclude that $E\left[\bar{\mathbf{y}}'\mathbf{u}\right] = tr\left[\boldsymbol{\Delta}'^{-1}\left(\mathbf{I}_{TmN} \otimes \mathbf{W}'\right)\boldsymbol{\Omega}\right] \neq 0$, being $\boldsymbol{\Delta} = \mathbf{I}_{Tm} \otimes \boldsymbol{\Lambda} \otimes \mathbf{W}$ and $\bar{\mathbf{y}}$ the spatial lag

of the explained variable, $\bar{\mathbf{y}} = [\mathbf{I}_{Tm} \otimes \mathbf{W}]\,\mathbf{y}$. **?** and **?** adapt the IV procedure to a spatial setting, advocating for the use of the so-called *spatial instrumental variables*, which is the approach implemented in **spsur**.

As known, a good instrument, say $\mathbf{Z}$, should combine two properties:

- the instrument must be orthogonal with the error term, $E\left[\mathbf{Z}'\mathbf{u}\right] = 0$
- the instrument should be highly correlated with the endogenous regressor $E\left[\mathbf{Z}'\bar{\mathbf{y}}\right] \neq 0$

Kelejian and Prucha show that if the $\mathbf{X}$ variables are really exogenous, the spatial lags of these variables are good candidates to become IV for the spatial lag of the explained variable. However, our problem is not standard because we are working in a multivariate SUR framework, which lead us to a Three-Stage Least-Squares approach using spatial instrumental variables. The procedure is as follows:

1. List the instruments in matrix $\mathbf{H}$. By default, **spsur** sets $\mathbf{H} = [\mathbf{X}, (\mathbf{I}_{Tm} \otimes \mathbf{W})\,\mathbf{X}^*], \left(\mathbf{I}_{Tm} \otimes \mathbf{W}^2\right)\mathbf{X}^*]$ for the SUR-SLM and $\mathbf{H} = \left[\mathbf{X}, (\mathbf{I}_{Tm} \otimes \mathbf{W})\,\mathbf{X}^*, \left(\mathbf{I}_{Tm} \otimes \mathbf{W}^2\right)\mathbf{X}^*, \left(\mathbf{I}_{Tm} \otimes \mathbf{W}^3\right)\mathbf{X}^*\right]$ for the SUR-SDM. This means that, in the SLM, $\mathbf{X}$ variables instrument themselves, $\mathbf{WX}^*$ and $\mathbf{W^2X}^*$ serves as instrument for $\mathbf{Wy}$; in the SDM case, $\mathbf{X}$ and $\mathbf{WX}^*$ instrument themselves and $\mathbf{W^2X}^*$, $\mathbf{W^3X}^*$ instrument $\mathbf{Wy}$.

2. Solve a Least-Squares, LS, regresion of the explained variable on the list of instruments to obtain an approximation to the *optimal instruments*, $\hat{\mathbf{y}} = \mathbf{H}\left(\mathbf{H}'\mathbf{H}\right)^{-1}\mathbf{H}'\mathbf{y}$

3. Substitute the spatial lag of the explained variable, $\mathbf{Wy}$, for its estimates $\mathbf{W}\hat{\mathbf{y}}$ in the corresponding SUR equation. In this moment, the endogeneity problem has been corrected.

4. The errors of the SUR model are not standard because they are correlated among the Tm equations. However, LS produces unbiased and consistent estimates of the β, θ and λ parameters. Accordingly, the second stage is to solve a LS regression with this group of variables.

5. Using the LS residuals of the previous step, we obtain a consistent estimate of the covariance matrix of the SUR equations, $\hat{\mathbf{\Sigma}}_{g,h} = \frac{\sum_{t=1}^{Tm} \hat{\mathbf{u}}'_{tg}\hat{\mathbf{u}}_{th}}{Tm}$, being $\hat{\mathbf{u}}_{tg}$ the $(N \times 1)$ vector of LS residuals of the g-th equation in time period t (from now on we will use two subscrits for time and equation, respectively). This estimate is consistent with Tm.

6. The last step is a Feasible Generalized Least-Squares estimate of the β, θ and λ parameters with the aim of improving their efficiency; that is:

$$\hat{\boldsymbol{\eta}}^*_{3sls} = \left[\hat{\mathbf{X}}'\left(\hat{\mathbf{\Sigma}}^{-1} \otimes\ \mathbf{I}_{TmN}\right)\hat{\mathbf{X}}\right]^{-1}\left[\hat{\mathbf{X}}'\left(\hat{\mathbf{\Sigma}}^{-1} \otimes\ \mathbf{I}_{TmN}\right)\mathbf{y}\right] \tag{9}$$

being $\boldsymbol{\eta}^* = [\boldsymbol{\beta}; \boldsymbol{\theta}; \boldsymbol{\lambda}]'$ and $\hat{\mathbf{X}}$ the corresponding matrix of regressors of the third step described above.

In sum, the `spsur3sls()` function consists in a Three-Stages Least-Squares, 3SLS, algorithm based on spatial instrumental variables. This estimation procedure is almost linear, which expedites the computational effort. On the negative side, let us note that the estimation of the λ parameters is not restricted by the Jacobian term, which allows for estimates of λ outside the *stability interval*: $\frac{1}{\lambda_t^-} < \lambda_t < \frac{1}{\lambda_t^+}$, being λ_t^- and λ_t^+ the greatest negative and positive eigenvalues, respectively, of the weighting matrix. Moreover, the parameters associated with the spatial errors, ρs, cannot be estimated using this algorithm.

2.3. Software review

The aim of **spsur** is to stimulate the use of spatial SUR models in applied research, by offering a friendly software easily accessible. As said, at present the user has very limited options, which has slowed the spreading of spatial SUR models.

In the field of quantitative analysis for spatial data, we can cite several R specialized packages. Probably, the most popular are **spdep** (**?**) and **spml** (**?**). The **spdep** is an R package with high functionalities for econometric analysis of cross-sectional data whereas **spml** is more focused in the case of spatial panels. On the other hand, the R package **systemfit** (**?**) is devoted to SUR models, but do not incorporate spatial effects. That is, none of the three packages have specific utilities to solve inference of spatial SUR models.

Indeed, to our knowledge, the closest antecedent of **spsur** is the beta version of **Spacestat** (**?**), called **SpaceStat v.May91**, which had limited functionalities and required Windows 98 (see video YouTube). Of course, nowadays it would be very difficult doing applied research using **SpaceStat v.May91**.

There is a second alternative in Python; it is the library **pysal** (**?**). The module `spreg.sur` provides SUR estimation of models with no spatial effects; the output includes several diagnostic measures among which appear the *raw* Lagrange Multpliers. Moreover, the codes `spreg.sur_error` and `spreg.sur_lag` provides the estimation of spatial SUR models with spatial errors and with spatial lags of the explained variable, respectively. The SUR-SEM model is estimated by ML whereas the SUR-SLM is estimated using spatial instrumental variables throug 3SLS. Finally, there is some code in MATLAB, related to the work of (**?**), but the source is not free software and the codes are not fully integrated.

3. Introducing the spsur package

spsur is an open-source software for the R computing platform (**?**). **spsur** depends on the packages **Formula**, **MASS**, **Matrix**, **methods**, **numDeriv**, **minqa**, **sparseMVN**, **spdep** and **stats**. The package can be freely dowloaded from CRAN and a development version is available in the link Github-spsur. The users will find a complete helping documentation and a vignette to guide them through the functionalities of the package.

spsur is made of 10 different functions which allow for a thoroughful analysis of a spatial SUR specification. Figure 1 shows the main functionalities of the package. **spsur** includes one function to evaluate the Lagrange Multipliers for the presence of spatial correlation in a SUR-SIM model (`lmtestspsur()`); two functions to estimate spatial SUR models, by maximum likelihood (`spsurml()`) or using instrumental variables (`spsur3sls()`) for the general case of G>1 (including both Tm=1 and Tm>1), and a third one designed to deal with pure spatial panels

(`spsurtime()`) for G=1 and Tm>1; four additional functions allow to improve the specification (`lrtestspsur()`; `wald_betas()`, `wald_deltas()` and `lr_betas_spsur()`); another function obtains the spatial impacts of the regressors (`impacts()`). Finally `dgp_spsur()` allows the user to solve Monte Carlo experiments using spatial SUR models.

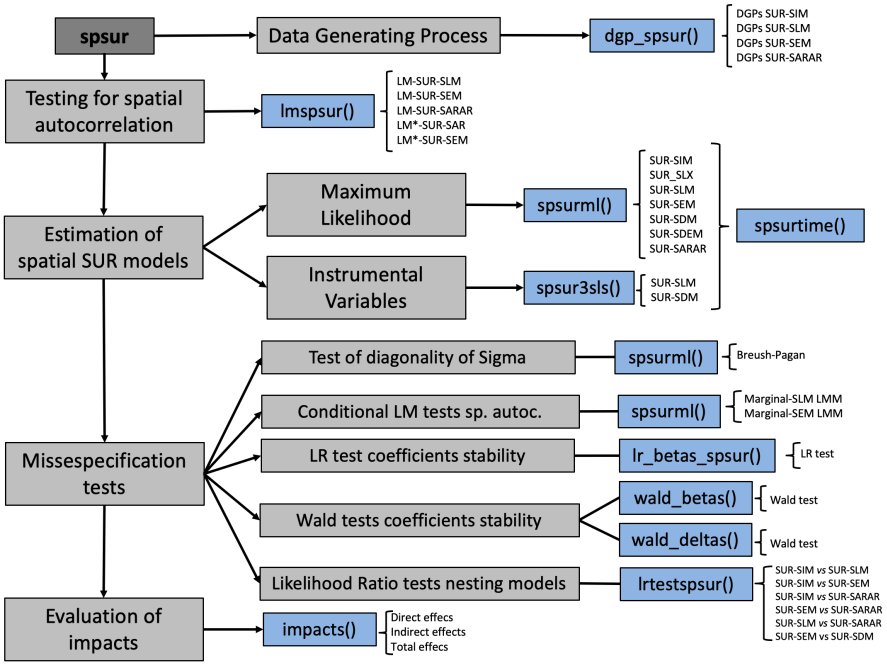

Figure 1: Main functionatities of spsur package

3.1. Datasets and baseline models

Two main data sets have been included in the package with the aim of showing the functionalities of **spsur**, namely, `spc` and `NCOVR`. Each data set consists of the corresponding data frame plus a weighting matrix that must be row-standardized; if the matrix is not row-standardized, then **spsur** does it.

Example 1: The first dataset, `spc`, constitutes a classical example taken from **?** p. 203-211, for 25 counties in South-West Ohio and two time periods, 1981 and 1983; the associated weighting matrix is `Wspc`. This is a Phillips-Curve problem where the explained variable measures the changes in wage rates (WAGE), whereas the regressors are unemployment rate (UN), net-migration rate (NMR) and a dummy variable (SMSA) with a value of 1 for metropolitan counties.

The SUR model estimated by Anselin is:

$$\begin{aligned} WAGE_{83} &= \beta_{10} + \beta_{11}\, UN_{83} + \beta_{12}\, NMR_{83} + \beta_{13}\, SMSA + \varepsilon_{83} \\ WAGE_{81} &= \beta_{20} + \beta_{21}\, UN_{80} + \beta_{22}\, NMR_{80} + \beta_{23}\, SMSA + \varepsilon_{81} \end{aligned} \tag{10}$$

The package is installed from the CRAN repository and the data set loaded as usual,

```
R> # install.packages("spsur")
```

```
R> library(spsur)
R> data("spc", package = "spsur")
```

We use the **Formula** package [**?** to especify the multiequational SUR model of (10):

```
R> spcformula <- WAGE83 | WAGE81 ~ UN83 + NMR83 + SMSA | UN80 + NMR80 + SMSA
```

Note that in the left side of the formula, two dependent variables have been included separated by the symbol |. In the right side appear the independent variables for each equation, separated again by a vertical bar |, and keeping the same order that in the left side.

Example 2: The second data set (`NCOVR`) comes from GeoDa Data and Lab collection and refers to homicide rates for 3,085 continental U.S. counties for four time periods (1960, 1970, 1980 and 1990), whose full description can be found in GeoDa. This data set includes a high number of socio-economic characteristics of the counties. The matrix **W** is included in **spsur**. This data set has been used by **?** and collected via a non-gridded sampling design.

To illustrate the functionalities of **spsur**, we build a SUR model with three equations and different number of regressors in each equation:

$$\begin{aligned} HR_{80} &= \beta_{10} + \beta_{11}\ PS_{80} + \beta_{12}\ UE_{80} + \varepsilon_{HR} \\ DV_{80} &= \beta_{20} + \beta_{21}\ PS_{80} + \beta_{22}\ UE_{80} + \beta_{23}\ SOUTH + \varepsilon_{DV} \\ FP_{79} &= \beta_{30} + \beta_{31}\ PS_{80} + \varepsilon_{FP} \end{aligned} \tag{11}$$

HR_{80} is homicide rates per 100,000 in 1980, PS_{80} measures the population structure in 1980, UE_{80} the unemployment rate, DV_{80} is the divorce rate, FP_{79} is the percentage of families below poverty in 1980 whereas SOUTH is a dummy variable for Southern counties. Note that the model of (11) has three equations but only one cross-section. To input this model into R we should write:

```
R> data(NCOVR, package = "spsur")
R> NCOVRformula <- HR80 | DV80 | FP79 ~ PS80 + UE80 | PS80 + UE80 + SOUTH | PS80
```

The LM tests in spsur package

The function `lmtestspur()` computes 5 Lagrange Multipliers, LM, for omitted spatial dependence in a SUR-SIM model. The syntax of this function is:

```
lmtestspsur(Form = NULL, data = NULL, W = NULL, X = NULL,Y = NULL, time = NULL,
G = NULL, N = NULL, Tm = NULL, print_table = TRUE)
```

Note that, as the examples below show, most of the argument with `NULL` default can be omitted from the syntaxis, if they are not needed, so that the final expressions simplify to a great deal. The simplest way to obtain the LMs is through a formula, `Form`, previously defined by the user, a data frame, `data`, and a weighting matrix, `W`, such as:

```
R> LMs.spc <- lmtestspsur(Form = spcformula, data = spc, W = Wspc)
```

```
#>              LM-Stat. DF p-value
#> LM-SUR-SLM     5.2472  2  0.0725 .
```

```
#> LM-SUR-SEM     3.3050  2  0.1916
#> LM*-SUR-SLM    2.1050  2  0.3491
#> LM*-SUR-SEM    0.1628  2  0.9218
#> LM-SUR-SARAR   5.7703  4  0.2170
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

spc contains information for only 25 spatial units and, therefore, the power of the LM tests is very low. Similarly, for the homicide rates case of (11), we obtain:

```
R> LMs.NCOVR <- lmtestspsur(Form = NCOVRformula, data = NCOVR, W = W)
```

```
#>               LM-Stat. DF  p-value
#> LM-SUR-SLM    5494.02  3  < 2e-16 ***
#> LM-SUR-SEM    5620.26  3  < 2e-16 ***
#> LM*-SUR-SLM     53.16  3 1.69e-11 ***
#> LM*-SUR-SEM    179.41  3  < 2e-16 ***
#> LM-SUR-SARAR  5741.17  6  < 2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There is a second way to compute the LMs, based on matrices instead of formulas. In first place, it is necessary to create the requiered matrices; for example, according to model (11) we should write:

```
R> Y <- as.matrix(c(NCOVR$HR80,NCOVR$DV80,NCOVR$FP79))
R> Intercep <- matrix(1,ncol = 1,nrow = 3085)
R> X1 <- cbind(Intercep,NCOVR$PS80,NCOVR$UE80)
R> X2 <- cbind(Intercep,NCOVR$PS80,NCOVR$UE80,NCOVR$SOUTH)
R> X3 <- cbind(Intercep,NCOVR$PS80)
R> X <- as.matrix(Matrix::bdiag(X1,X2,X3))
```

The last matrix, $\mathbf{X}$ it is block-diagonal. Then, the user introduces these matrices as arguments in `lmtestspsur()`. The number of individuals (N), equations (G) and time periods (Tm) must also be specified,

```
R> LMs.NCOVR.matrix <- lmtestspsur(Y = Y, X = X, G = 3, Tm = 1, N = 3085, W = W)
```

In the example above, all the LMs reject their respective null hypotheses, which clearly discards the SUR-SIM model.

Maximum Likelihood estimation of spatial SUR models

If the LM tests reject their associated null hypothesis, a spatial SUR model must be estimate. The function `spsurml()` obtains the ML estimation of the different models listed in Section 2. The syntax of this fuction is:

```
spsurml(Form = NULL, data = NULL, R = NULL, r = NULL, W = NULL, X = NULL, Y = NULL,
G = NULL, N = NULL, Tm = NULL, p = NULL, demean = FALSE, type = "sim", cov = TRUE,
control = list(tol = 0.001, maxit = 200, trace = TRUE))
```

Similarly, most of the argument with `NULL` default can be omitted from the syntaxis, if they are not needed. Once again, the simplest way to estimate a spatial SUR is by using a formula, a data frame and a weighting matrix, which are the arguments that appear in `Form`, `data` and `W`. The second way implies the use of matrices, similar as before.

To estimate a SUR-SLM model with the data set of Example 1, the syntax is:

```
R> spcsur.slm <- spsurml(Form = spcformula, data = spc, type = 'slm', W = Wspc)

#> Initial point:   log_lik:  113.197  lambdas:  -0.472 -0.446
#> Iteration:  1   log_lik:  114.085  lambdas:  -0.506 -0.482
#> Iteration:  2   log_lik:  114.096  lambdas:  -0.506 -0.482
#> Time to fit the model:  2.08  seconds
#> Computing marginal test...
#> Time to compute covariances:  0.33  seconds
```

The output of `spsurml()` includes details about the iteration process and time of computation. Let us also note that the user can configure the optimization process using the `control` list, as an additional argument to the function. In this way, the tolerance of the convergence process can be set to `tol`; the same as the number of maximum iterations, `maxit`, whereas `trace` allows the user hide intermediate results.

Consistent with the conventions in the R environment, the `summary()` method prints the output equation by equation.

```
R> summary(spcsur.slm)

#> Call:
#> spsurml(Form = spcformula, data = spc, W = Wspc, type = "slm")
#>
#>
#> Spatial SUR model type:  slm
#>
#> Equation  1
#>                  Estimate Std. Error t value  Pr(>|t|)
#> (Intercept)_1  1.4955217  0.2467240  6.0615 5.183e-07 ***
#> UN83_1         0.8070029  0.2557439  3.1555  0.003179 **
#> NMR83_1       -0.5194114  0.2590550 -2.0050  0.052318 .
#> SMSA_1        -0.0073247  0.0118519 -0.6180  0.540347
#> lambda_1      -0.5057334  0.2405734 -2.1022  0.042401 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.6224
#>   Equation  2
#>                  Estimate Std. Error t value  Pr(>|t|)
```

```
#> (Intercept)_2  1.7094414  0.2925620  5.8430 1.024e-06 ***
#> UN80_2        -0.6745562  0.3870737 -1.7427   0.08969 .
#> NMR80_2        0.7502934  0.3842670  1.9525   0.05847 .
#> SMSA_2         0.0014181  0.0241859  0.0586   0.95356
#> lambda_2      -0.4821428  0.2557758 -1.8850   0.06730 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.4743
#>   Variance-Covariance Matrix of inter-equation residuals:
#>   0.0003085954 -0.0003561928
#>  -0.0003561928  0.0015864976
#> Correlation Matrix of inter-equation residuals:
#>   1.000000 -0.509062
#>  -0.509062  1.000000
#>
#>  R-sq. pooled: 0.6603
#>  Log-Likelihood:  114.096
#>  Breusch-Pagan: 6.516  p-value: (0.0107)
#>  LMM: 0.50489  p-value: (0.477)
```

The summary includes basic information for each equation such as estimated coefficients, standard deviations, etc. Moreover, in a SUR context, the matrices of covariances and of correlations of the residuals among the Tm equations are of special interest for the user. The summary also reports the value of maximum log likelihood of the model, some measures of goodness of fit, the Breusch-Pagan test of diagonality, (**?**) and the corresponding marginal LM tests, for omitted spatial errors in SLM or SDM models or omitted lags of the explained variable in SEM or SDEM models (more details in Section 4.2 below). The marginal multipliers are printed in the output simply as LMM.

The syntax for other spatial specifications is very similar, just changing the argument `type`. For example:

```
R> spcSUR.sim <- spsurml(Form = spcformula, data = spc, type = 'sim', W = Wspc)
R> spcSUR.slx <- spsurml(Form = spcformula, data = spc, type = 'slx', W = Wspc)
R> spcSUR.sem <- spsurml(Form = spcformula, data = spc, type = 'sem', W = Wspc)
R> spcSUR.sarar <- spsurml(Form = spcformula, data = spc, type = 'sarar', W = Wspc)
R> spcSUR.sdm <- spsurml(Form = spcformula, data = spc, type = 'sdm', W = Wspc)
R> spcSUR.sdem <- spsurml(Form = spcformula, data = spc, type = 'sdem', W = Wspc)
```

For reasons of space, we skip the output for these other models, and continue using the SUR-SLM case.

As said before, the control options can be used to customize the iteration process. In the case below, the convergence process will stop if the difference in the estimated likelihood between two consecutive iterations is smaller than `tol=0.1` or the number of iterations reaches the value `maxit=20`. The controls also advises to hidden the intermediate results, with the argument `trace`.

```
R> mlcontrol <- list(tol = 0.1, maxit = 20, trace = FALSE)
R> NCOVRSUR.slm <- spsurml(Form = NCOVRformula, data = NCOVR,
R+                         type = 'slm', W = W, control = mlcontrol, cov = FALSE)
```

The second way to estimate a SUR-SLM model uses arguments based on matrices such as, for example:

```
R> mlcontrol <- list(tol = 0.1, maxit = 20, trace = FALSE)
R> NCOVRSUR.slm.matrix <- spsurml(Y = Y, X = X, G = 3, Tm = 1, N = 3085,
R+                                W = W, type = 'slm', control = mlcontrol,
R+                                cov = FALSE)
```

The output of the funtion `spsurml()` is an object of an `spsur` class.

3SLS estimation

The function `spsur3sls()` estimates spatial SUR models by spatial IVs in a 3SLS framework. The syntax of this fuction is very similar to `spsurml()`.

```
spsur3sls(Form = NULL, data = NULL, R = NULL, b = NULL, W = NULL,X = NULL, Y =
NULL, G = NULL, N = NULL, Tm = NULL, p = NULL, demean = FALSE, type = "slm", maxlagW
= 2)
```

The main differences between the two are: (i) the argument `type` now only supports two spatial SUR models: `slm` and `sdm` and (ii) a new argument is needed, `maxlagW`, to set the maximum order of spatial lags of the regressors to obtain the IVs. The default value is `maxlagW=2` for SUR-SLM models and `maxlagW=3` for the SDM variant.

The 3SLS algorithm is especially adequate in cases of large datasets or when `spsur3sls()` has to deal with very dense Jacobians. Our *Example 2* includes 3,085 US counties and it is a good candidate for `spsur3sls()`. The code to estimate a SUR-SLM model for the SUR of (11) is,

```
R> NCOVRSUR.slm.3sls <-spsur3sls(Form = NCOVRformula, data = NCOVR, type = "slm", W = W)

#> Time to fit the model:  0.2  seconds

R> summary(NCOVRSUR.slm.3sls)

#> Call:
#> spsur3sls(Form = NCOVRformula, data = NCOVR, W = W, type = "slm")
#>
#>
#> Spatial SUR model type:  slm
#>
#> Equation  1
#>                 Estimate Std. Error t value  Pr(>|t|)
#> (Intercept)_1  8.877617   1.307821  6.7881 1.206e-11 ***
#> PS80_1         0.892351   0.138898  6.4245 1.388e-10 ***
```

```
#> UE80_1        -0.060245   0.033200 -1.8146   0.06962 .
#> lambda_1      -0.221472   0.191153 -1.1586   0.24664
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.1405
#>   Equation  2
#>                Estimate Std. Error t value  Pr(>|t|)
#> (Intercept)_2 2.9935206  0.3798303  7.8812  3.61e-15 ***
#> PS80_2        0.2533626  0.0262643  9.6467 < 2.2e-16 ***
#> UE80_2        0.0914432  0.0095959  9.5294 < 2.2e-16 ***
#> SOUTH_2       0.0346575  0.0471028  0.7358    0.4619
#> lambda_2      0.2124110  0.0900048  2.3600    0.0183 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.291
#>   Equation  3
#>                Estimate Std. Error t value  Pr(>|t|)
#> (Intercept)_3  7.670117   1.095699  7.0002 2.734e-12 ***
#> PS80_3        -1.585681   0.160403 -9.8856 < 2.2e-16 ***
#> lambda_3       0.386346   0.087474  4.4167 1.014e-05 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.5611
#>   Variance-Covariance Matrix of inter-equation residuals:
#>  45.377766  1.867308 18.961586
#>   1.867308  1.938112 -1.427426
#>  18.961586 -1.427426 34.820491
#> Correlation Matrix of inter-equation residuals:
#>  1.0000000  0.1991154  0.4770190
#>  0.1991154  1.0000000 -0.1737588
#>  0.4770190 -0.1737588  1.0000000
#>
#>  R-sq. pooled: 0.3595
```

A great advantage of `spsur3sls()` is the serious reduction of computing time, compared with ML methods. In the case of `NCOVR`, with 3,085 observations and 3 equations, the estimation took less than a second. The ML estimation, using the function `spsurml()` for the same problem, required of almost 10 minutes (using a Intel Core i7 2.93 GHz).

Like in the case of `spsurml()`, the output of the funtion `spsur3sls()` is an object of the `spsur` class.

3.2. Spatial panel SUR models

It is not unusual, in applied work, to have data for only one equation, G=1, such as in **?**. We

refer to this case as an *Spatial panel SUR* with, for example, a SLM structure:

$$\begin{aligned} &\mathbf{y}_t = \lambda_t \mathbf{W}\mathbf{y}_t + \mathbf{x}_t\boldsymbol{\beta}_t + \boldsymbol{\varepsilon}_t; t = 1, 2, \ldots, Tm \\ &E[\boldsymbol{\varepsilon}_t] = 0 \quad E[\boldsymbol{\varepsilon}_t \boldsymbol{\varepsilon}'_s] = \sigma_{t,s}\mathbf{I}_N \end{aligned} \tag{12}$$

Now $\mathbf{y}_t$ and $\boldsymbol{\varepsilon}_t$ are $(N \times 1)$ vectors and $\mathbf{x}_t$ in a $(N \times p_t)$ matrix where p_t is the number of regressors in the t-th cross-section. The spatial parameters, λs, as well as the βs are allowed to change from cross-section to cross-section. The SUR structure appears because there is serial dependence in the error terms of each individual in the sample; serial dependence among differente individuals is not allowed. The serial dependence is estimated in the $(Tm \times Tm)$ $\boldsymbol{\Sigma}$ covariance matrix by the function `spsurtime()`. An important constraint to mention is that the serial dependence is assumed to be uniform for all individuals in the sample.

The syntax of `spsurtime()` combines arguments of `spsurml()` and `spsur3sls()` and it is quite transparent:

```
spsurtime(Form, data, time, type = "sim", method = "ml", maxlagW = 2, W = NULL,
cov = TRUE, demean = FALSE, trace = TRUE, R = NULL, b = NULL)
```

The spatial models supported by `spsurtime()` are the seven specifications listed in Section 2, SUR-SIM, SUR-SLX, SUR-SLM, SUR-SEM, SUR-SDM, SUR-SDEM and SUR-SARAR. Moreover, the user can decide if the estimation algorithm is ML, which is the default, or 3SLS following the same treatment described in Section 3.

Finally, this function also allows to demean the data with the purpose of removing potential unobserved effects in the sample. We use the most popular transformation, which simply subtracts the time sampling average of each individual from the corresponding observation, using the *demeaning* $\mathbf{Q}$ matrix: $\mathbf{Q} = \mathbf{Q}^+ \otimes \mathbf{I}_N$, where $\mathbf{Q}^+ = \left(\mathbf{I}_{Tm} - \frac{1}{Tm}\mathbf{J}_{Tm}\right)$, with $\mathbf{J}_{Tm} = \boldsymbol{\tau}_{Tm}\boldsymbol{\tau}'_{Tm}$ and $\boldsymbol{\tau}_{Tm}$ a $(Tm \times 1)$) vector of *1s*. Note that the $\mathbf{Q}$ matrix needs the data to be ordered, first, by time and second by individuals. If this is not the case, the data are routinely sorted by `spsurtime()`, according to the argument `time`.

Demeaning the data can produce strange results, in spatial SUR panels, if the time dimension, Tm, is very large and N, number of spatial units, is finite in which case we have a problem of high-dimensionality: we need to estimate $\frac{Tm \times (Tm-1)}{2}$ covariance terms with $N \times Tm$ observations. Moreover, in the general case where G>1, if Tm is very small and N is finite *demeaning* can produce problems of singularity in the covariance matrix because it inflates the correlations among the error terms by a proportionality factor of order $O(\frac{1}{Tm})$, (**?**). Thus, our advice is using *demeaning* for large N and moderate values of Tm in the spatial panel case and avoid *demeaning* if Tm is very small and N is moderate, in both cases.

A case of example

In this example we employ, once more, the data set `NCOVR`. We should remind that this data frame has not a panel data structure. So, the first question is to reshape `NCOVR`. This is the purpose of the indices *index_time* and *index_indiv*. The panel equation explains the homicides rates, HR, in each county in function of the population structure, PS, and the unemployment rate, UE; the equation has a SLM structure, which is estimated by 3sls.

```
R> data(NCOVR,package="spsur")
R> N <- nrow(NCOVR)
R> Tm <- 4
R> index_time <- rep(1:Tm, each = N)
R> index_indiv <- rep(1:N, Tm)
R> pHR <- c(NCOVR$HR60, NCOVR$HR70, NCOVR$HR80, NCOVR$HR90)
R> pPS <- c(NCOVR$PS60, NCOVR$PS70, NCOVR$PS80, NCOVR$PS90)
R> pUE <- c(NCOVR$UE60, NCOVR$UE70, NCOVR$UE80, NCOVR$UE90)
R> pNCOVR <- data.frame(indiv = index_indiv, time = index_time,
R+                        HR = pHR, PS = pPS, UE = pUE)
R> rm(NCOVR,pHR,pPS,pUE,index_time,index_indiv)
R> form_pHR <- HR ~ PS + UE
R> pHR_slm <- spsurtime(Form = form_pHR, data = pNCOVR, W = W,
R+                      time = pNCOVR$time, type = "slm", method = "3sls")

#> Time to fit the model:  0.29  seconds

R> summary(pHR_slm)

#> Call:
#> spsur3sls(W = W, X = X, Y = Y, G = G, N = N, Tm = Tm, p = p,
#>     type = type, maxlagW = maxlagW)
#>
#>
#> Spatial SUR model type:  slm
#>
#> Equation  1
#>                Estimate Std. Error t value Pr(>|t|)
#> (Intercept)_1  2.501017   1.645140  1.5202   0.1285
#> PS_1          -0.029887   0.105860 -0.2823   0.7777
#> UE_1           0.029898   0.046422  0.6441   0.5196
#> lambda_1       0.405674   0.392876  1.0326   0.3018
#> R-squared: 0.2843
#>   Equation  2
#>                Estimate Std. Error t value Pr(>|t|)
#> (Intercept)_2 1.8871930  1.9497433  0.9679  0.33310
#> PS_2          0.1705918  0.1651996  1.0326  0.30179
#> UE_2          0.0099723  0.0497241  0.2006  0.84105
#> lambda_2      0.6971788  0.2942217  2.3696  0.01782 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.3604
#>   Equation  3
#>                Estimate Std. Error t value  Pr(>|t|)
#> (Intercept)_3  6.881003   1.220794  5.6365 1.774e-08 ***
#> PS_3           0.658929   0.135891  4.8489 1.256e-06 ***
#> UE_3           0.183858   0.032266  5.6982 1.239e-08 ***
```

```
#> lambda_3      -0.172457   0.178288 -0.9673    0.3334
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.06152
#>   Equation  4
#>               Estimate Std. Error t value  Pr(>|t|)
#> (Intercept)_4 3.917895   0.369609 10.6001 < 2.2e-16 ***
#> PS_4          0.909703   0.119384  7.6200 2.723e-14 ***
#> UE_4          0.319217   0.042374  7.5332 5.295e-14 ***
#> lambda_4      0.023133   0.073747  0.3137    0.7538
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.134
#>   Variance-Covariance Matrix of inter-equation residuals:
#>  31.72416 20.95984 16.31181 13.23690
#>  20.95984 53.75722 26.29607 19.95032
#>  16.31181 26.29607 45.37777 19.69094
#>  13.23690 19.95032 19.69094 38.89755
#> Correlation Matrix of inter-equation residuals:
#>  1.0000000 0.5075448 0.4299180 0.3768166
#>  0.5075448 1.0000000 0.5324158 0.4362851
#>  0.4299180 0.5324158 1.0000000 0.4686880
#>  0.3768166 0.4362851 0.4686880 1.0000000
#>
#>  R-sq. pooled: 0.1608
```

The output reported by `spsurtime()` is quite detailed. Note that the ML estimation of the SUR-SLM panel equation takes almost 4 minutes while the 3sls requires of only 0.28 seconds.

4. Misspecification tests and other utilities

The estimation of spatial SUR models needs of a series of assumptions that are critical to guarantee the good properties of the (ML, 3SLS) estimates. Some of the tests required to build the spatial structure of the model have been presented in Section 2.1. Below we add a series of misspecification tests, implemented in **spsur**, that may help to improve the specification.

4.1. Testing for the diagonality of the covariance matrix

The problem is well known in the econometric literature: if the $\mathbf{\Sigma}$ matrix is diagonal, the SUR structure collapses to a set of non-related equations that can be estimated separately. The null and alternative hypotheses are:

$$\begin{array}{l} H_0 : \mathbf{\Sigma} = diag(\sigma_{11}, ..., \sigma_{TmTm}) \\ H_A : no \;\; H_0 \end{array} \tag{13}$$

? obtained a simple Lagrange Multiplier test, that is become very popular: $LM_{\mathbf{\Sigma}} = N \sum_{t=1}^{Tm} \sum_{j=1}^{t-1} r_{ij}^2 = \chi^2_{Tm(Tm-1)/2}$. The correlation coefficients, r_{ij}, are obtained from the ML residuals correspond-

ing to the Tm time periods, as said, estimated separately. Note that in the general case of $G>1$, $Tm>1$ and $N>1$, the covariance matrix is of order $(G\times G)$. The LM test of diagonality is implement in the function `spsurml()` and printed routinely.

4.2. Marginal Lagrange Multipliers

The purpose of the Marginal Lagrange Multipliers is to check if there remains correlation in the error terms, *once we have estimated* a SUR-SLM model or if a spatial lag of the explained variable has been omitted, *once we have estimated* a SUR-SEM model (**?**). The null hypothesis for the first case is the same as that in (7), but now the hypothesis is tested in a SUR-SLM, not in a SUR-SIM model. The model of the alternative is a SUR-SARAR equation. Similarly, the null and alternative hypotheses of the second case are the same as those that appear in (8), which are tested in a SUR-SEM ML estimation. The model of the alternative is, once again, a SUR-SARAR.

Both tests are printed routinely in the output of the `spsurml()` function, denoted as LMM. There is no risk of confusion: if we have estimated a SUR-SLM model (or a SUR-SDM) with `spsurml()`, LMM tests for omitted spatial errors and for omitted spatial lags in case of having estimated a SUR-SEM model (or a SUR-SDEM).

4.3. Likelihood Ratio tests

The models listed in Section 2 are related in a nesting sequence of restrictions. Given that `spsurml()` operates in a ML framework, we can easily obtain the corresponding Likelihood Ratios, LR, to compare all the competing models. This is the purpose of the function `lrtestspsur()`, whose use is very simple. In case of Example 1, the syntax to obtain the LR test is:

```
R> LRtests <- lrtestspsur(Form = spcformula, data = spc, W = Wspc)

#> LogLik SUR-SIM:   111.378
#> LogLik SUR-SLM:   114.096
#> LogLik SUR-SEM:   113.719
#> LogLik SUR-SARAR:  115.26
#> LogLik SUR-SDM:   116.646
#>
#>  LR test SUR-SIM versus SUR-SLM:
#> statistic:  5.437  p-value: ( 0.066 )
#>  LR test SUR-SIM versus SUR-SEM:
#> statistic:  4.684  p-value: ( 0.096 )
#>  LR test SUR-SIM versus SUR-SARAR:
#> statistic:  7.766  p-value: ( 0.101 )
#>  LR test SUR-SLM versus SUR-SARAR:
#> statistic:  2.328  p-value: ( 0.312 )
#>  LR test SUR-SEM versus SUR-SARAR:
#> statistic:  3.082  p-value: ( 0.214 )
#>
#>  LR testSUR-SLM versus SUR-SDM:
```

```
#> statistic:  5.1  p-value: ( 0.078 )
#>  COMFAC LR test SUR-SEM versus SUR-SDM:
#> statistic:  5.854  p-value: ( 0.054 )
```

The output of the function only shows the value of the corresponding LRs and their associated pvalues, obtained from a χ^2 distribution with degrees of freedom equal to the number of restrictions in the respective null hypothesis. `lrtestspsur()` does not show the estimated models, although they are computed by the function.

The last LR in the example above, COMFAC, is known as the *Common Factor* test and occupies a relevant position into a *Gets*, General to specific, selection strategy of spatial models because the two most popular specifications, SLM and SEM, are not nested and cannot be directly related. The COMFAC offers the possibility of, indirectly, compare the two specifications through a spatial Durbin equation (**?**).

4.4. Testing for linear restrictions on the parameters

As described before, the functions `spsurml()` and `spsur3sls()` solve ML and 3SLS estimation of the spatial SUR models. Now we are interested in a strategy to test linear restrictions on the parameters of the model, which shoul rely on the asymptotic properties of both algorithms.

? showed that, in a pure cross-sectional setting, the ML estimates of the SLM model are $\sqrt{N}$-*consistent*. Under mild regularity conditions, they are also asymptotically normally distributed. These conclusions can be extended to other spatial specifications, such as those computed in **spsur**. As indicated in **?**, the ML estimates maintain their good properties in a panel data framework, which now can be consistent with N, Tm or with both simultaneously (the exception is the recovery of the unobserved effects, that requires of a fixed N). In a SUR context, **?** showed that the ML algorithm is also consistent, efficient and asymptotically normally distributed, with Tm. Similarly, **?** demonstrates the asymptotic good properties of the 3SLS estimates in multivariate models with endogenous regressors, assuming that all instruments are orthogonal with all error terms.

That is, in both cases it is possible to sustain the asymptotic normality, with Tm and/or N, of the corresponding estimates. In the case of ML, under the standard conditions described in **?** and **?**, we can write that:

$$\hat{\boldsymbol{\eta}} \underset{as}{\sim} N\left(\boldsymbol{\eta}; \mathbf{V}(\boldsymbol{\eta})\right) \tag{14}$$

where $\boldsymbol{\eta}$ is the vector of parameters included in the spatial SUR model, as defined in (4) and $\mathbf{V}(\boldsymbol{\eta})$ the corresponding covariance matrix of the ML estimates, obtained from the *Information Matrix* as described in **?**. A similar result can be stated for the case of the 3SLS estimates, under the conditions of orthogonality stated previouly (the details on the covariance matrix can be found in Chapter 7 of **?**). Let us note that the ρ and σ parameters are excluded from the inference in the 3SLS approach.

Using the asymptotic distribution of (14), it is immediate to test for any kind of linear restriction on the vector of parameters of the model, such as exclusion restrictions or across-equation linear restrictions. The procedure is as usual. First we need to write down the $(r \times p)$

matrix $\mathbf{R}$, with rank r, that captures the linear restrictions:

$$H_0: \quad \mathbf{R}\boldsymbol{\eta} = \mathbf{b} \quad vs \quad H_A: \quad \mathbf{R}\boldsymbol{\eta} \neq \mathbf{b} \tag{15}$$

The Wald statistic for testing (15) is

$$W = (\mathbf{R}\hat{\boldsymbol{\eta}} - \mathbf{b})' \left(\mathbf{R}\hat{\mathbf{V}}\mathbf{R}'\right)^{-1} (\mathbf{R}\hat{\boldsymbol{\eta}} - \mathbf{b}) \underset{as}{\sim} \chi^2_r \tag{16}$$

With slight variations, we obtain a similar result for the 3SLS algorithm.

As a complement to the Wald tests, **spsur** offers the possibility to test linear restrictions on the β parameters through a Likelihood Ratio. Of course, this option is restricted to ML estimates and it is a bit more costly than the Wald test because we need to estimate two models, the model of the null hypothesis and the model of the alternative hypothesis.

Illustration

spsur includes two functions to test linear restrictions on the parameters of the model. The first, is the function `wald_betas()`, directed at testing linear restrictions on the β parameters whereas the second, `wald_deltas()`, evaluates linear restrictions on the spatial coefficients (λ or ρ parameters). Both functions operate with the Wald statistic of (16). As said before, the linear restrictions on the β parameters can also be evaluated through a LR test using the function `lr_betas_spsur()`.

`wald_betas()` operates with the β parameters. The function needs three arguments. The first is an object of `spsur` class obtained with the functions `spsurml()` or `spsur3sls()`, the second is a matrix that must reflect the restrictions on the β parameters and the third is a vector with the values of the restrictions, under the null hypothesis. Let us assume that we need to test whether the two intercepts, in the two equations of the SUR-SLM model of Example 1, are equal:

$$H_0: \beta_{10} = \beta_{20} \quad vs \quad H_A: \beta_{10} \neq \beta_{20} \tag{17}$$

The code for this problem is:

```
R> R1 <- matrix(c(1,0,0,0,-1,0,0,0), nrow = 1)
R> b1 <- matrix(0, ncol = 1)
R> Wald_beta <- wald_betas(results = spcsur.slm, R = R1, b = b1)

#> Wald stat.: 0.396 p-value: (0.529)
```

If, as in the example above, the `wald_betas()` leads to accept the restrictions, the user, probably, would want to estimate the constrained version of the corresponding model, such as:

$$\begin{aligned} WAGE_{83} &= \beta_0 + \beta_{11}\, UN_{83} + \beta_{12}\, NMR_{83} + \beta_{31}\, SMSA + \lambda_1 WAGE_{83} + \varepsilon_{83} \\ WAGE_{81} &= \beta_0 + \beta_{21}\, UN_{80} + \beta_{22}\, NMR_{80} + \beta_{32}\, SMSA + \lambda_2 WAGE_{81} + \varepsilon_{81} \end{aligned} \tag{18}$$

`spsurml()` and `spsur3sls()` provide restricted ML and 3SLS estimation for the spatial SUR models through the optional arguments `R=R1` and `b=b1` in the description of the functions. The code, for the example that we are considering is,

```
R> R1 <- matrix(c(1,0,0,0,-1,0,0,0),nrow=1)
R> b1 <- matrix(0, ncol = 1)
R> spcSUR.slm.restricted <- spsurml(Form = spcformula, data = spc, type = "slm",
R+                                  W = Wspc, R = R1, b = b1,
R+                                  control=list(tol=0.01,maxit=200,trace=FALSE))
```

Note that in the output only the intercept is in the first equation but is the same for all the equations.

```
R> summary(spcSUR.slm.restricted)
```

```
#> Call:
#> spsurml(Form = spcformula, data = spc, R = R1, b = b1, W = Wspc,
#>     type = "slm", control = list(tol = 0.01, maxit = 200, trace = FALSE))
#>
#>
#> Spatial SUR model type:  slm
#>
#> Equation  1
#>                 Estimate Std. Error t value  Pr(>|t|)
#> (Intercept)_1  1.5753057  0.2075381  7.5904 3.965e-09 ***
#> UN83_1         0.7879259  0.2527386  3.1176  0.003468 **
#> NMR83_1       -0.4773175  0.2550271 -1.8716  0.068970 .
#> SMSA_1        -0.0052895  0.0115539 -0.4578  0.649697
#> lambda_1      -0.5819558  0.2030220 -2.8665  0.006732 **
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.627
#>   Equation  2
#>            Estimate Std. Error t value Pr(>|t|)
#> UN80_2   -0.6155921  0.3890947 -1.5821  0.12191
#> NMR80_2   0.6980038  0.3912165  1.7842  0.08238 .
#> SMSA_2    0.0026959  0.0245851  0.1097  0.91326
#> lambda_2 -0.3677309  0.1848838 -1.9890  0.05394 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.4578
#>   Variance-Covariance Matrix of inter-equation residuals:
#>   0.0003056010 -0.0003758212
#>  -0.0003758212  0.0016804187
#> Correlation Matrix of inter-equation residuals:
#>   1.0000000 -0.5244392
#>  -0.5244392  1.0000000
#>
#>  R-sq. pooled: 0.6459
#>  Log-Likelihood:  113.806
```

```
#>  Breusch-Pagan: 6.793  p-value: (0.00915)
#>  LMM: 0.4578  p-value: (0.499)
```

The purpose of the function, `wald_deltas()`, is to obtain a Wald test to evaluate linear restrictions on the spatial parameters of the model, λ or ρ parameters. The use of the function is totally analogous to `wald_betas()`, and extends to ML and 3SLS algorithms. For example, in the SUR-SLM of (18) the user may be interested in testing if the λ parameters of the two equations are equal, that is:

$$H_0 : \lambda_1 = \lambda_2 \quad vs \quad H_A : \lambda_1 \neq \lambda_2 \tag{19}$$

```
R> R2 <- matrix(c(1,-1), nrow = 1)
R> b2 <- matrix(0, ncol = 1)
R> Wald_lambda <- wald_deltas(results=spcSUR.slm.restricted, R = R2, b = b2)

#>
#>  Wald stat.: 16.739 (0)
```

`lr_betas_spsur()` operates only with objects produced by `spsurml()`, which correspond to ML estimates of spatial SUR models. For the previous example, the code is simply:

```
R> R1 <- matrix(c(1,0,0,0,-1,0,0,0), nrow = 1)
R> b1 <- matrix(0, ncol = 1)
R> LR_SMSA <- lr_betas_spsur(Form = spcformula, data = spc, W = Wspc,
R+                           type = "slm", R = R1, b = b1, trace = FALSE,
R+                           printmodels = FALSE)

#>
#>  Fitting unrestricted model ...
#>
#>  Time to fit unrestricted model:  2.39  seconds
#>
#>  Fitting restricted model ...
#> Time to fit restricted model:  12.87  seconds
#>
#>  LR-Test
#>
#>  Log-likelihood unrestricted model:  114.096
#>  Log-likelihood restricted model:  113.809
#>  LR statistic:  0.575  degrees of freedom:  1  p-value: ( 0.4482648 )
```

5. Direct, Indirect and Total impacts

According to **?**, a major difference between times series models and spatial models consists in the interpretation of the β parameters. In a time series model, the β parameters have

a unequivocal meaning: they measure the expected increase in the explained variable as a consequence of a unitary change in a given regressor. This impact is constant across time, $\frac{\partial y_t}{\partial x_t} = \beta_k$, and its significance can be checked by a simple tratio.

Things are not so clear in a spatial setting because of the feed-back effects. In fact, a change in a regressor, in a given location, will induce a chain of reactions that will spill over into the space, depending on the weights matrix, $\mathbf{W}$. Consider, for example, the reduced form of a SUR-SARAR model:

$$\mathbf{y} = \mathbf{A}^{-1}\mathbf{X}\boldsymbol{\beta} + \mathbf{A}^{-1}\mathbf{B}^{-1}\boldsymbol{\varepsilon} = \mathbf{I}_{Tm} \otimes \left[\mathbf{I}_N + \boldsymbol{\Lambda} \otimes \mathbf{W} + \boldsymbol{\Lambda}^2 \otimes \mathbf{W}^2 + \ldots\right] \mathbf{X}\boldsymbol{\beta} + \mathbf{A}^{-1}\mathbf{B}^{-1}\boldsymbol{\varepsilon} \quad (20)$$

The expression above shows that a change in the value of a regressor, in certain point in space, will impact the explained variable in the same location and also in other points of space. Note that $\mathbf{A}$ is a block-diagonal matrix whose non-zero matrices are $\mathbf{A}_t = \mathbf{I}_N - \lambda_t \mathbf{W}; t = 1, 2, \ldots Tm$. The inverse of $\mathbf{A}_t$ admits a power expansion on the weights matrix, $\mathbf{W}$, and the spatial parameters λ_t, which lead us to the result in (20).

Turning to the general case of $G>1$ and $Tm>1$ and using the notation in **?**, we may write the g-th equation of (20), in period t, as:

$$\mathbf{y}_{tg} = \begin{bmatrix} y_{tg1} \\ y_{tg2} \\ \cdots \\ y_{tgN} \end{bmatrix} = \sum_{h=1}^{p_q} \begin{bmatrix} S_{tg}^h(\mathbf{W})_{11} & S_{tg}^h(\mathbf{W})_{12} & \cdots & S_{tg}^h(\mathbf{W})_{1N} \\ S_{tg}^h(\mathbf{W})_{21} & S_{tg}^h(\mathbf{W})_{22} & \cdots & S_{tg}^h(\mathbf{W})_{2N} \\ \cdots & \cdots & \cdots & \cdots \\ S_{tg}^h(\mathbf{W})_{N1} & S_{tg}^h(\mathbf{W})_{N2} & \cdots & S_{tg}^h(\mathbf{W})_{NN} \end{bmatrix} \begin{bmatrix} x_{tg1} \\ x_{tg2} \\ \cdots \\ x_{tgN} \end{bmatrix} + \mathbf{A}_g^{-1}\mathbf{B}_g^{-1} \begin{bmatrix} \varepsilon_{tg1} \\ \varepsilon_{tg2} \\ \cdots \\ \varepsilon_{tgN} \end{bmatrix} \quad (21)$$

being $\mathbf{S}_{tg}^h(\mathbf{W}) = \beta_h \mathbf{A}_g^{-1}$ and $S_{tg}^h(\mathbf{W})_{ij}$ its $i, j - th$ element. From above, it is clear that the value of the partial derivative of y with respect to x_h depends on the point in space that we are impacting; that is: $\frac{\partial y_{itg}}{\partial x_{h_{jtg}}} = S_{tg}^h(\mathbf{W})_{ij}$. Moreover, it is evident, form (21), that a shock in a point will also spread out through the space, but the discussion about spatial multipliers excludes the case of impacting the error terms.

Specifically, *for each equation* and every regressor, we can evaluate three effects, such as:

$$\begin{aligned} \mathbf{M}_{Total_g^h} &= \frac{1}{N}\tau_N^{'}\mathbf{S}_{.g}^h\tau_N \\ \mathbf{M}_{Direct_g^h} &= \frac{1}{N}tr\mathbf{S}_{.g}^h \\ \mathbf{M}_{Indirect_g^h} &= \mathbf{M}_{Total_g^h} - \mathbf{M}_{Direct_g^h} \end{aligned} \quad (22)$$

$$h = 1, \ldots p_g; g = 1 \ldots G$$

being τ_N a $N \times 1$ unitary vector and $\mathbf{S}_{.g}^h = \sum_{t=1}^{Tm} \mathbf{S}_{tg}^h(\mathbf{W})$.

Under the assumption that the spatial parameters and the weighting matrices are constant across time, the partial derivatives in (21) are also constant. Futhermore, it is important to recall that the models in **spsur** have no time dynamics, so the impact of a modification in a regressors must be absorved in the same time period.

Finally, we can aggregate the multipliers in (22) to obtain overall measures for each variable:

$$\begin{aligned} \mathbf{M}_{Total_{\bullet}^{h}} &= \sum_{g=1}^{G} \mathbf{M}_{Total_{g}^{h}} \\ \mathbf{M}_{Direct_{\bullet}^{h}} &= \sum_{g=1}^{G} \mathbf{M}_{Direct_{g}^{h}} \\ \mathbf{M}_{Indirect_{\bullet}^{h}} &= \sum_{g=1}^{G} \mathbf{M}_{Indirect_{g}^{h}} \end{aligned} \tag{23}$$

$$h = 1, \dots p_g$$

The multipliers of (23) should be accompanied by measures of statistical significance to be useful for the user. **?** suggest two possibilities. First is the *Bayesian Markov Chain Monte Carlo* method, MCMC, which draws samples of parameters from their posterior distribution and computes random sequences of multipliers, to calibrate the estimates. However, to our knowledge, there are no results available for spatial SUR models using MCMC. The second alternative is bootstrapping. In the case of **spsur**, we prefer a parametric bootstrap, which means that:

- We estimate the spatial SUR model by ML.
- We draw B samples of random vectors of the parameters in our spatial SUR, $\boldsymbol{\eta}$, using a multivariate normal distribution, $N\left(\hat{\boldsymbol{\eta}}; \hat{\mathbf{V}}(\hat{\boldsymbol{\eta}})\right)$, where $\hat{\boldsymbol{\eta}}$ is the ML estimation of the parameters and $\hat{\mathbf{V}}(\hat{\boldsymbol{\eta}})$ their corresponding estimated covariance matrix. Let denote by $\boldsymbol{\eta}_b$ the b-th draw.
- We re-evaluate the effects using the boostrapped vectors $\boldsymbol{\eta}_b$, maintaining the data of the regressors. In this way, a sequence of values: $M_{Total_{g}^{h(b)}}$, $M_{Direct_{g}^{h(b)}}$ and $M_{Indirect_{g}^{h(b)}}$ for $b = 1, \dots B$ is obtained.
- Finally, we calculate the corresponding measure of dispersion as the standard deviation of the series of estimated multipliers:

$$\sigma_{M_{Effect_{g}^{h}}} = \sqrt{\frac{1}{B}\sum_{b=1}^{B}\left(M_{Effect_{g}^{h(b)}} - \overline{M_{Effect_{g}^{b}}}\right)^{2}} \tag{24}$$

with *Effect=Total, Direct, Indirect.*

A case of example

The discussion about the multipliers is solved quite easily in **spsur** through the function `impacts()`, which has only two arguments: an object of class `spsur` and the number of simulations to estimate the significance measures, `nsim`, whose default value is 1000.

Below we continue with the case of *Example 1* where a SUR-SLM model was specified for the data frame `spc`. Let us recall that this is a spatial Phillips curve explaining unemployment, `UN`, through net migration, `NMR`, and a dummy variable, `SMSA`, for metropolitan counties. The code is very simple

```
R> spcsur.slm.impacts <- impacts(spcsur.slm)
```

```
#>
#> Spatial SUR model type:  slm
#>
#>  Direct effects
#>
#>                mean          sd  t-stat    p-val
#> UN83_1   0.88305059  0.28621010  3.0853 0.002033 **
#> NMR83_1 -0.55212471  0.27952172 -1.9752 0.048240 *
#> SMSA_1  -0.00805628  0.01289044 -0.6250 0.531984
#> UN80_2  -0.72586584  0.40925036 -1.7736 0.076121 .
#> NMR80_2  0.78117832  0.42259476  1.8485 0.064526 .
#> SMSA_2   0.00090907  0.02559427  0.0355 0.971666
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#>  Indirect effects
#>
#>                mean          sd  t-stat  p-val
#> UN83_1  -0.32440343  0.19205753 -1.6891 0.0912 .
#> NMR83_1  0.19721952  0.14175779  1.3912 0.1642
#> SMSA_1   0.00237863  0.00512170  0.4644 0.6423
#> UN80_2   0.25656173  0.21241940  1.2078 0.2271
#> NMR80_2 -0.26695846  0.20976695 -1.2726 0.2031
#> SMSA_2   0.00091969  0.00986562  0.0932 0.9257
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#>  Total effects
#>
#>               mean         sd  t-stat    p-val
#> UN83_1   0.5586472  0.1956443  2.8554 0.004298 **
#> NMR83_1 -0.3549052  0.1936613 -1.8326 0.066861 .
#> SMSA_1  -0.0056777  0.0087628 -0.6479 0.517030
#> UN80_2  -0.4693041  0.2809791 -1.6702 0.094871 .
#> NMR80_2  0.5142199  0.3067234  1.6765 0.093641 .
#> SMSA_2   0.0018288  0.0176120  0.1038 0.917299
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The output of the function are three tables of impacts for each regressor, with their respective measures of statistical significance. As indicated, the user can specify the number of draws for the measures of significance. See the help of the function for more examples using the database `NCOVR`.

6. A general case: G, Tm and N bigger than 1

The framework of the previous Sections can be generalized quite easily to a case where the user has data for more than one single equation; that is, G>1. The equivalent to the GNM model of expression (2) is:

$$\left.\begin{array}{l} \mathbf{y}_{tg} = \lambda_g \mathbf{W}\mathbf{y}_{tg} + \mathbf{X}_{tg}\boldsymbol{\beta}_g + \mathbf{W}\mathbf{X}^*_{tg}\boldsymbol{\gamma}_g + \mathbf{u}_{tg} \\ \mathbf{u}_{tg} = \rho_g \mathbf{W}\mathbf{u}_{tg} + \boldsymbol{\varepsilon}_{tg} \\ E\left[\boldsymbol{\varepsilon}_{tg}\right] = 0 \quad E\left[\boldsymbol{\varepsilon}_{tg}\,\boldsymbol{\varepsilon}'_{sh}\right] = \left\{ \begin{array}{ll} \sigma_{gh}\mathbf{I}_N & t = s \\ \mathbf{0}_N & t \neq s \end{array} \right. \end{array}\right\} \tag{25}$$

for g=1,...G; t=1,...Tm. The SUR nature of the model appears in the form of *contemporaneous* correlation among the errors of the different equations, corresponding to the same individual. This correlation is zero for errors belonging to equations of different individuals and/or different time periods. Serial correlation is only allowed in Anselin's approach (treated with the function `spsurtime()`, as shown before). If we stack the G vectors of errors corresponding to period t in a $(NG\times 1)$ vector $\boldsymbol{\varepsilon}_t$, its covariance matrix can be written as $E\left[\boldsymbol{\varepsilon}_t\,\boldsymbol{\varepsilon}'_t\right] = \boldsymbol{\Sigma}\otimes\mathbf{I}_N$; this is a $(NG\times NG)$ matrix. In continuation, we can stack these Tm vectors into a general $(TmNG\times 1)$ vector by ordering the observations, first, by time period, then by equation and finally by individuals. Using obvious matrix notation, we obtain:

$$\left.\begin{array}{l} \mathbf{A}\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + (\mathbf{I}_{Tm}\otimes\mathbf{I}_G\otimes W)\mathbf{X}^*\boldsymbol{\gamma} + \mathbf{u} \\ \mathbf{B}\mathbf{u} = \boldsymbol{\varepsilon};\ \boldsymbol{\varepsilon}\sim(0,\boldsymbol{\Omega}) \end{array}\right\}$$

$$\mathbf{y} = \underset{TmGN\times 1}{\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \cdots \\ \mathbf{y}_{Tm} \end{bmatrix}};\ \mathbf{X} = \underset{TmGN\times p}{\begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \\ \cdots \\ \mathbf{X}_{Tm} \end{bmatrix}};\ \mathbf{X}^* = \underset{TmGN\times(p-G)}{\begin{bmatrix} \mathbf{X}^*_1 \\ \mathbf{X}^*_2 \\ \cdots \\ \mathbf{X}^*_{Tm} \end{bmatrix}};\ \mathbf{u} = \underset{TmGN\times 1}{\begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \cdots \\ \mathbf{u}_{Tm} \end{bmatrix}};\ \boldsymbol{\varepsilon} = \underset{TmGN\times 1}{\begin{bmatrix} \boldsymbol{\varepsilon}_1 \\ \boldsymbol{\varepsilon}_2 \\ \cdots \\ \boldsymbol{\varepsilon}_{Tm} \end{bmatrix}};$$

$$\mathbf{X}_t = \underset{GN\times p}{\begin{bmatrix} \mathbf{X}_{1t} & 0 & \cdots & 0 \\ 0 & \mathbf{X}_{2t} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \mathbf{X}_{Gt} \end{bmatrix}};\mathbf{X}^*_t = \underset{GN\times(p-G)}{\begin{bmatrix} \mathbf{X}^*_{1t} & 0 & \cdots & 0 \\ 0 & \mathbf{X}^*_{2t} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \mathbf{X}^*_{Gt} \end{bmatrix}};\boldsymbol{\beta} = \underset{p\times 1}{\begin{bmatrix} \boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2 \\ \cdots \\ \boldsymbol{\beta}_G \end{bmatrix}};\boldsymbol{\gamma} = \underset{(p-G)\times 1}{\begin{bmatrix} \boldsymbol{\gamma}_1 \\ \boldsymbol{\gamma}_2 \\ \cdots \\ \boldsymbol{\gamma}_G \end{bmatrix}}; \tag{26}$$

where $\mathbf{A} = \mathbf{I}_{Tm}\otimes[\mathbf{I}_{GN} - \boldsymbol{\Lambda}\otimes\mathbf{W}] = \mathbf{I}_{Tm}\otimes diag\left\{\mathbf{A}_g; g = 1,2,..,G\right\}$, and $\mathbf{B} = \mathbf{I}_{Tm}\otimes[\mathbf{I}_{GN} - \boldsymbol{\Upsilon}\otimes\mathbf{W}] = diag\left\{\mathbf{B}_g; g = 1,2,..,G\right\}$, $\boldsymbol{\Lambda}$ and $\boldsymbol{\Upsilon}$ are two $(G\times G)$ diagonal matrices containing the spatial parameters λ_g and ρ_g, respectively. Moreover $\boldsymbol{\Omega} = \mathbf{I}_{Tm}\otimes\boldsymbol{\Sigma}\otimes\mathbf{I}_N$. The same as before, we allow for a different number of regressors in the right hand side of each equation, p_g, where the total number of β parameters is: $p = \Sigma_{g=1}^{G} p_g$ and p-G is the total number of γ parameters.

Assuming that the errors are normally distributed, the log-likelihood function of the spatial GNM-SUR model of (26) can be written as:

$$l(\mathbf{y};\boldsymbol{\eta}) = -\frac{NTmG}{2}\ln(2\pi) - \frac{NTm}{2}\ln|\boldsymbol{\Sigma}| + Tm\left\{\sum_{g=1}^{G}\ln|\mathbf{B}_g| + \sum_{g=1}^{G}\ln|\mathbf{A}_g|\right\} \quad (27)$$

$$- \frac{\{\mathbf{Ay} - [\mathbf{X};\mathbf{X}^*]\boldsymbol{\Pi}\}'\,\mathbf{B}'\,\boldsymbol{\Omega}^{-1}\,\mathbf{B}\,\{\mathbf{Ay} - [\mathbf{X};\mathbf{X}^*]\boldsymbol{\Pi}\}}{2} \quad (28)$$

where $\boldsymbol{\eta}' = [\boldsymbol{\Pi}';\lambda_1;\cdots;\lambda_G;\rho_1;\cdots;\rho_G;\sigma_{ij}]$ is the vector parameters of the model, being $\boldsymbol{\Pi} = [\boldsymbol{\beta};\boldsymbol{\gamma}]'$. The discussion that continues from this point is totally analogous to the spatial SUR panel analyzed so far, thus we leave it now to avoid repetitions.

The example below shows the case of a SUR-SLM model, for `NCOVR`, where G=3, Tm=4 and N=3,085, including the reshaping of a database into a classical panel:

```
R> data(NCOVR,package="spsur")
R> N <- nrow(NCOVR)
R> Tm <- 4
R> index_time <- rep(1:Tm, each = N)
R> index_indiv <- rep(1:N, Tm)
R> pHR <- c(NCOVR$HR60, NCOVR$HR70, NCOVR$HR80, NCOVR$HR90)
R> pPS <- c(NCOVR$PS60, NCOVR$PS70, NCOVR$PS80, NCOVR$PS90)
R> pUE <- c(NCOVR$UE60, NCOVR$UE70, NCOVR$UE80, NCOVR$UE90)
R> pDV <- c(NCOVR $DV60, NCOVR$DV70, NCOVR$DV80, NCOVR$DV90)
R> pFP <- c(NCOVR$FP59, NCOVR$FP70, NCOVR$FP80, NCOVR$FP90)
R> pSOUTH <- rep(NCOVR$SOUTH, Tm)
R> pNCOVR <- data.frame(indiv = index_indiv, time = index_time,
R+                      HR = pHR, PS = pPS, UE = pUE, DV = pDV,
R+                      FP = pFP, SOUTH = pSOUTH)
R> rm(NCOVR,pHR,pPS,pUE,pDV,pFP,pSOUTH,index_time,index_indiv)
R> pform <- HR | DV | FP ~ PS + UE | PS + UE + SOUTH | PS
```

The estimation of a SLM model with G=3, Tm=4 and N=3,085 can be done as usual using the `spsurml()` function (to prevent overflows of memory we set `cov` argument equals to `FALSE`). The last sentence of the code just print the vector of estimates of the λ and β parameters and the correlation matrix among the residuals of the three equations:

```
R>  psur_slm <- spsurml(Form = pform, data = pNCOVR, W = W,
R+                      type = "slm", cov = FALSE,
R+                    control = list(tol = 0.1, maxit = 200, trace = FALSE))
R> psur_slm$deltas; psur_slm$betas; psur_slm$Sigma_corr

#>  lambda_1  lambda_2  lambda_3
#> 0.5761895 0.8894981 0.8296673

#> (Intercept)_1          PS_1          UE_1 (Intercept)_2          PS_2
#>   1.427602305   0.385014192   0.190392695   0.108058188   0.099139394
#>          UE_2       SOUTH_2 (Intercept)_3          PS_3
#>   0.059332730  -0.003307201   6.124085927  -3.575372137
```

```
#>            [,1]       [,2]       [,3]
#> [1,] 1.0000000  0.1482262  0.1349689
#> [2,] 0.1482262  1.0000000 -0.1724809
#> [3,] 0.1349689 -0.1724809  1.0000000
```

7. How to generate random spatial SUR datasets

This section describes an additional functionality included in **spsur**, which offers the possibility of generate random data sets with the dimensions and spatial structure decided by the user. This is the purpose of `dgp_spsur()`. The function has great flexibility to accomodate the necessities of the user and may be of interest in two different situations: (i) as part of a larger simulation experiment where the user needs random data sets with specific features (SUR nature, spatial structure, etc.) or (ii) with the aim of showing specific properties of spatial SUR models and inferential procedures related to them.

`dgp_spsur()` is a Data Generating Process where the user decides the dimensions of the required data set, specificies the values of the parameters that intervene in the corresponding SUR model and selects the distribution function from which the regressors and the random terms are to be drawn. The syntax of the function is the following:

```
dgp_spsur <- function(Sigma, Tm = 1, G, N, Betas, Thetas = NULL, durbin = FALSE,
rho = NULL, lambda = NULL, p = NULL, W = NULL, X = NULL, pdfU = "nvrnorm", pdfX
= "nvrnorm")
```

The dimensions of the data set are defined by the arguments `Tm`, `N` and `G`. Then the user specifies the SUR structure, begining by the covariance matrix among the residuals of the G equations, which is the core of the SUR model, with the argument `Sigma`. Of course, a $(N \times N)$ spatial weighting matrix should be uploaded in the argument `W`.

In continuation, the user should think in the parameters that intervene in the equations of the SUR. This is the purpose of the arguments `Betas`, `Thetas`, `rho` and `lambda` which are defined through row vectors of the adequate dimensions. A fundamental piece of information is the argument `p` that defines the number of regressors (that is, x variables) that appear in each equation; if `p` is a scalar, every equation has the same number of regressors whereas if we need a model with different number of regressor in each equation, `p` must be a $(1 \times G)$ vector. In both cases, `Betas` is a row vector of order $(1 \times G^*)$, where G^* is pG in the first case and $\Sigma_1^G p_g$ in the second case.

Note that, depending on the values given to these arguments, the type of spatial model is unequivocally determined. For example, if `Thetas=NULL`, `rho=NULL` but `lambda` is not `NULL`, the user is specifying a SUR-SLM model or a SUR-SDEM model in the case of `lambda=NULL` but `Thetas` and `rho` different from `NULL`.

There are two possibilities to build the matrix $\mathbf{X}$ of regressors. In Monte Carlo experiments is very usual to maintain fixed the values of the regressors and repeatedly draw random matrices of error terms, to simulate the explained variable. If this is the case, the user should upload the required $\mathbf{X}$ matrix in the argument `X`, which must be consistent with the dimensions of the SUR model. If `X=NULL` the user prefers to randomly draw this matrix using the function `dgp_spsur()`. This is the purpose of the argument `pdfX`. By default, `dgp_spsur()` uses a multivariate standard Normal distribution to draw the observations of the regressors; the

alternative is a Uniform distribution, in the interval $[0, 1]$, for each regressor. In both cases, the regressors are generated independently.

Finally, the argument `dfU` informs which multivariate probability distribution function should be used to draw the values of the error terms. `dgp_spsur()` offers two possibilities, the multivariate Normal, which is the default, and the log-Normal distribution which means to exponentiate the sample drawn from the multivariate Normal distribution. In both cases, the covariance matrix matches the $\mathbf{\Sigma}$ matrix uploaded previously. The log-Normal should be taken as a clear departure from the assumption of normality.

The output of `dgp_spsur()` is a vector, called $\mathbf{Y}$, of order $(TmNG \times 1)$, with the values generated for the explained variable in the G equations of the spatial SUR model. If the argument `X` is set to `NULL` the user will receive another matrix, called $\mathbf{XX}$, of order $(TmNG \times \Sigma_{g=1}^{G} p_g)$, with the values generated for the regressors of the SUR.

The example below shows the random generation of 3 variables, with a SUR structure, for 500 individuals and 1 cross-section, using a SUR-SDM model. Each equation contains 3 regressors, as specified in `p`. The weighting matrix, `W`, is obtained randomly applying the *5 nearest neighbors* criteria, after having drawn the lattitude and longitude of each point in space

```
R> Tm <- 2; G <- 3; N <- 500
R> Sigma <- matrix(0.3, ncol = G, nrow = G)
R> diag(Sigma) <- 1
R> Betas <- c(1,2,3,1,-1,0.5,1,-0.5,2)
R> Thetas <- c(1,-1,0.5,-0.5,1,0)
R> lambda <- 0.5
R> co <- cbind(runif(N,0,1),runif(N,0,1))
R> W <- spdep::nb2mat(spdep::knn2nb(spdep::knearneigh(co, k = 5,longlat = FALSE)))
R> DGP <- dgp_spsur(Sigma = Sigma, Betas = Betas, Thetas = Thetas,
R+                  lambda = lambda, Tm = Tm, G = G, N = N, p = 3, W = W)
```

As said, the output of this function is a list with the data (explained variable and regressors) in matrix form. Next code estimates a SDM model for the simulated data:

```
R> sdm_sim <- spsurml(Y = DGP$Y, X = DGP$X, type = "sdm", G = G,
R+                    Tm = Tm, N = N, p = 3, W = W,
R+                    control = list(tol = 0.01, maxit = 200,
R+                                   trace = FALSE))
R> summary(sdm_sim)

#> Call:
#> spsurml(W = W, X = DGP$X, Y = DGP$Y, G = G, N = N, Tm = Tm, p = 3,
#>     type = "sdm", control = list(tol = 0.01, maxit = 200, trace = FALSE))
#>
#>
#> Spatial SUR model type:  sdm
#>
#> Equation  1
```

```
#>             Estimate Std. Error  t value  Pr(>|t|)
#> Intercep_1  1.020361   0.060097  16.9787 < 2.2e-16 ***
#> X1_1        2.008810   0.033172  60.5572 < 2.2e-16 ***
#> X1_2        2.942658   0.029366 100.2061 < 2.2e-16 ***
#> W_X1_1      1.066664   0.122178   8.7304 < 2.2e-16 ***
#> W_X1_2     -0.868682   0.114411  -7.5926 4.169e-14 ***
#> lambda_1    0.467882   0.027579  16.9649 < 2.2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.937
#>   Equation  2
#>             Estimate Std. Error  t value  Pr(>|t|)
#> Intercep_2  1.039629   0.075174  13.8297 < 2.2e-16 ***
#> X2_1       -0.992082   0.029661 -33.4472 < 2.2e-16 ***
#> X2_2        0.516674   0.028123  18.3719 < 2.2e-16 ***
#> W_X2_1      0.426443   0.078068   5.4624 5.084e-08 ***
#> W_X2_2     -0.453934   0.064198  -7.0708 1.911e-12 ***
#> lambda_2    0.487447   0.033832  14.4078 < 2.2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.5906
#>   Equation  3
#>             Estimate Std. Error  t value Pr(>|t|)
#> Intercep_3  0.896685   0.062645  14.3137   <2e-16 ***
#> X3_1       -0.513205   0.030333 -16.9190   <2e-16 ***
#> X3_2        1.995177   0.030440  65.5454   <2e-16 ***
#> W_X3_1      0.888765   0.066465  13.3719   <2e-16 ***
#> W_X3_2     -0.117934   0.107134  -1.1008   0.2711
#> lambda_3    0.531207   0.029068  18.2745   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> R-squared: 0.8419
#>   Variance-Covariance Matrix of inter-equation residuals:
#>  1.0242470 0.3080020 0.3244861
#>  0.3080020 0.9148565 0.2851121
#>  0.3244861 0.2851121 1.0415104
#> Correlation Matrix of inter-equation residuals:
#>  1.0000000 0.3181813 0.3141680
#>  0.3181813 1.0000000 0.2920837
#>  0.3141680 0.2920837 1.0000000
#>
#>  R-sq. pooled: 0.8812
#>  Log-Likelihood:  -4190.32
#>  Breusch-Pagan: 285.3  p-value: (1.55e-61)
#>  LMM: 0.87389  p-value: (0.832)
```

8. Conclusions and Directions for Future Research

This article introduces a new R package, **spsur**, directed at estimating and solving inference on spatial SUR models. Our intention was to fill a gap in the applied spatial econometrics literature, in the sense that there are few papers which use spatial SUR models as a useful research technique. Our impression is that the lack of an efficient, friendly software has slowed its adoption in this field

The package consist of 10 inter-related functions. `lmtestspsur()` test for the existence of spatial effects in a model specified without spatial mechanisms, or SUR-SIM model. Then, assuming that some of these tests are positive, three functions `spsurml()`, `spsur3sls()` and `spsurtime()` allow to estimate, by ML or 3SLS, the prefered spatial specification. Four additional functions, `wald_betas()`, `wald_deltas()`, `lr_betas_spsur()` and `lrtestspsur()` can be used to improve the specification. The function `impacts()` is useful to obtain the spatial multipliers of the variables in the model selected, that come with a collection of measures of statistical significance. Finally, the function `dgp_sur()` is intented to be used in Monte Carlo experiments, involving spatial SUR models, or to illustrate specific features of this type of data.

Of course, this is only the first, basic version of the package that should be improved in the future. A list of actions already in the desk of the coauthors are (i) the extension of 3SLS to the treatment of endogenous regressors, others than the spatial lag of the explained variable, and the use of external instruments, (ii) a more efficient treatment of the unobserved effects, which allowed for their estimation and, at the same time, enlarge the options to demean the data, (iii) the introduction of time dynamics in a strictly stationary framework.

References

Affiliation:

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd