Spatial models in *gretl*: the SPM package

Chiara Casoli*, Luca Pedini*, and Francesco Valentini*

*Università Politecnica delle Marche

version 0.1

Abstract

This package allows practitioners to estimate cross-sectional spatial models in *gretl*. The package, presented in Casoli et al. (2019) can handle three types of models: Spatial Autoregressive Models (SAR), Spatial Durbin Models (SDM) and Spatial Error Models (SEM). Computation of the Hessian matrix is performed in both analytical and mixed ways. Some speed-up procedures for the computation of the log-determinant term are available.

Contents

1 The Spatial Models
   1.1 The Spatial Autoregressive model and the Spatial Durbin Model
   1.2 The Spatial Error Model

2 The functions
   2.1 The function *sr()*
   2.2 The function *sem()*
   2.3 The function *printres()*
   2.4 The function *spatial_GUI()*

3 Example: *spm_example.inp*
1 The Spatial Models

SPM package allows estimation of three types of spatial models: the SAR, the SDM and the SEM. For further details see LeSage and Pace (2009).

1.1 The Spatial Autoregressive model and the Spatial Durbin Model

The SAR model includes spatial lags of the dependent variable only, whereas the SDM adds also spatial lags of the covariates. Generalising, and defining 

\[ Z = [\iota_n X W X] \]

and

\[ \delta = [\alpha \beta \theta]' \]

it is possible to write:

\[ y = \rho W y + Z\delta + \varepsilon \] (1)

\[ y = (I_n - \rho W)^{-1}Z\delta + (I_n - \rho W)^{-1}\varepsilon \] (2)

\[ \varepsilon \sim N(0, \sigma^2 I_n), \]

where equation (1) denotes the SAR if \( Z = [\iota_n X] \) or the SDM if \( Z = [\iota_n X W X] \), and equation (2) the related DGP.

We denote \( y \) as an \( n \times 1 \) vector of the dependent variable, \( W \) as the \( n \times n \) spatial weight matrix, \( \iota_n \) as the constant term, \( X \) as the \( n \times k \) matrix of explanatory variables, and \( \varepsilon \) as the error component. Spatial dependence is captured by the parameter \( \rho \).

Estimation of parameters \( \rho, \delta \) and \( \sigma^2 \) is implemented via Maximum Likelihood. In particular, assuming \( \rho \) as known, defined as \( \rho^* \), the model becomes \( y - \rho^* W y = Z\delta + \varepsilon \), suggesting that parameters \( \delta \) and \( \sigma^2 \) can be easily estimated as follows:

\[ \hat{\delta} = (Z'Z)^{-1}Z'(I_n - \rho^* W)y \]

and

\[ \sigma^2 = n^{-1}(y - \rho^* Wy - Z\delta)'(y - \rho^* Wy - Z\delta). \]

The log-likelihood function is given by:

\[ \ln L = -(n/2)\ln(\pi \sigma^2) + \ln|I_n - \rho W| - \frac{e'e}{2\sigma^2} \] (3)

in which \( \omega \) contains the eigenvalues of the spatial weights matrix. If \( W \) has been scaled such to have the maximum eigenvalue equal to 1, it is possible to restrict the interval such that \( \rho \in (\min(\omega)^{-1}, \max(\omega)^{-1}) \). The choice of matrix \( W \) is up to users; SPM automatically provides a row-standardisation.

The optimisation problem can be easily handled using the concentrated log-likelihood (equation (5)) as a function of the only parameter \( \rho \). \( \delta \) and \( \sigma^2 \) can be consequently derived as a function of the estimated \( \rho \). This can be
summarised in:

\[
\ln L(\rho) = c + \ln |I_n - \rho W| - (n/2)\ln[(e_0 - \rho e_d)'(e_0 - \rho e_d)]
\]

(5)

\[
e_0 = y - Z\delta_0
\]

(6)

\[
e_d = Wy - Z\delta_d
\]

(7)

\[
\delta_0 = (Z'Z)^{-1}Z'y
\]

(8)

\[
\delta_d = (Z'Z)^{-1}Z'Wy,
\]

(9)

in which \(c\) is a constant term, \(\delta_0, e_0, \delta_d\) and \(e_d\) are computed \textit{ex ante} from two auxiliary regressions of \(y\) and \(Wy\) on \(Z\) respectively. The Maximum Likelihood estimates of parameters \(\hat{\delta}, \hat{\sigma}^2\) and the associated disturbances variance-covariance matrix \(\hat{\Omega}\) are given by: \(\hat{\delta} = \delta_0 - \hat{\rho}\delta_d, \hat{\sigma}^2 = n^{-1}(e_0 - \hat{\rho}e_d)'(e_0 - \hat{\rho}e_d)\) and \(\hat{\Omega} = \hat{\sigma}^2[(I_n - \hat{\rho}W)'(I_n - \hat{\rho}W)]^{-1}\). Finally, to calculate standard errors and the related \(t\) statistics, the variance-covariance matrix of the parameters is computed in two different ways: pure analytical and mixed (analytical/numerical), suggested in LeSage and Pace (2009). The choice of the technique is left to the users.

### 1.2 The Spatial Error Model

The SEM contains spatial dependences in the disturbances, as shown in equation (10), with (11) being the DGP.

\[
y = X\beta + u
\]

(10)

\[
u = \lambda Wu + \varepsilon
\]

\[
y = X\beta + (I_n - \lambda W)^{-1}\varepsilon
\]

(11)

\[
\varepsilon \sim N(0, \sigma^2 I_n).
\]

Here the spatial dependence is expressed by the parameter \(\lambda\); the other variables follow the notation described above.

The full log-likelihood is given by:

\[
\ln L = -(n/2)\ln(\pi \sigma^2) + \ln |I_n - \lambda W| - \frac{e'e}{2\sigma^2}
\]

(12)

\[
e = (I_n - \lambda W)(y - X\beta).
\]

Again, it is possible to concentrate the log-likelihood, as a function of the only parameter \(\lambda\), and then recovering \(\beta\) and \(\sigma^2\); unlike the previous case, however, \(e(\lambda)'e(\lambda)\) is not a simple quadratic form of the parameters, but is derived from moment matrices as in (14)

\[
\ln L(\lambda) = c + \ln |I_n - \lambda W| - (n/2)\ln(e(\lambda)'e(\lambda))
\]

(13)
\[ A_{XX}(\lambda) = X'X - \lambda X'WX - \lambda X'W'X + \lambda^2 X'W'WX \quad (14) \]

\[ A_{XY}(\lambda) = X'y - \lambda X'Wy - \lambda X'W'y + \lambda^2 X'W'Wy \]

\[ A_{YY}(\lambda) = y'y - \lambda y'Wy - \lambda y'W'y + \lambda^2 y'W'Wy \]

\[ \beta(\lambda) = A_{XX}(\lambda)^{-1}A_{XY}(\lambda) \]

\[ e(\lambda)'e(\lambda) = A_{YY}(\lambda) - \beta(\lambda)'A_{XX}(\lambda)\beta(\lambda) \]

The values for \( \hat{\beta} \) and \( \hat{\sigma}^2 \) can be recovered, again, straightforwardly (LeSage and Pace, 2009). The variance-covariance matrix is computed in analytical way.

## 2 The functions

The package provides 4 public functions. Via scripting, the functions are the following:

- **sr()**: this function provides estimation of a SAR model or a SDM.
- **sem()**: allows estimation of a SEM
- **printres()**: prints the results of sr() and sem()
- **spatial_GUI()**: recollects both sr() and sem() in a single function.

### 2.1 The function sr()

sr(series y, list X, matrix W, bool sdm, bool hess_form, int lik_type, scalar moments, scalar n_rep, scalar poly_ord)

**Return type**: bundle

- **y**: the dependent variable series
- **X**: the list of regressors (without constant);
- **W**: the weight matrix;
- **sdm**: a boolean (0 for the SAR, 1 for the SDM);
- **hess_form**: a boolean (0 for analytical Hessian, 1 for mixed);
- **int_lik**: an integer determining the log-determinant computation;
- **moments**: see below;
The integer lik_type assumes value 0 for analytical computation, 1 for the Ord decomposition of the log-determinant, 2 for Monte Carlo numerical approximation and 3 for approximation with Chebychev polynomials (LeSage and Pace, 2009). The default is 0. If lik_type = 2, the moments and n_rep scalars define the number of moments and the number of replications, respectively, necessary for the Monte Carlo approximation. If omitted, the defaults are 50 moments and 100 replications. Instead, if lik_type = 3, the scalar poly_ord specifies the Chebychev polynomial order. The default is 10.

The output of the function is a bundle containing:

- model: the estimated model
- dependent: name of the dependent variable
- variables: name of regressors
- weight: selected weight matrix
- beta: estimated coefficients
- betastderr: standard errors of estimated coefficients
- beta_t: the \( t \)-statistic for \( \hat{\beta} \)
- rho: estimated \( \rho \)
- rhostderr: standard errors of estimated \( \rho \)
- rho_t: the \( t \)-statistic for \( \hat{\rho} \)
- s2: estimated variance of the error
- s2stderr: standard errors of \( \hat{\sigma}^2 \)
- s2_t: the \( t \)-statistic for \( \hat{\sigma}^2 \)
- hess: Hessian computation type
- lkt: log-determinant computation type
- lk: log-likelihood
- Sigma: covariance matrix of errors
- CPUtime: elapsed time in seconds
2.2 The function \texttt{sem()} \\

\texttt{sem(series y, list X, matrix W, int lik_type, scalar moments,} \\
\texttt{scalar n_rep, scalar poly_ord)} \\

Return type : bundle \\

\texttt{y} : the dependent variable series \\
\texttt{X} : the list of regressors (without constant); \\
\texttt{W} : the weight matrix; \\
\texttt{int\_lik} : an integer determining the log-determinant computation; \\
\texttt{moments} : see below; \\
\texttt{n\_rep} : see below; \\
\texttt{poly\_ord} : see below. \\

The integer \texttt{lik\_type} assumes value 0 for analytical computation, 1 for the Ord decomposition of the log-determinant, 2 for Monte Carlo numerical approximation and 3 for approximation with Chebychev polynomials (LeSage and Pace 2009). The default is 0. If \texttt{lik\_type} = 2, the \texttt{moments} and \texttt{n\_rep} scalars define the number of moments and the number of replications, respectively, necessary for the Monte Carlo approximation. If omitted, the defaults are 50 moments and 100 replications. Instead, if \texttt{lik\_type} = 3, the scalar \texttt{poly\_ord} specifies the Chebychev polynomial order. The default is 10. \\
The output of the function is a bundle containing: \\

- \texttt{model}: the estimated model \\
- \texttt{dependent}: name of the dependent variable \\
- \texttt{variables}: name of regressors \\
- \texttt{weight}: selected weight matrix \\
- \texttt{beta}: estimated coefficients \\
- \texttt{betastderr}: standard errors of estimated coefficients \\
- \texttt{beta\_t}: the \texttt{t}-statistic for \( \hat{\beta} \) \\
- \texttt{lambda}: estimated \( \lambda \) \\
- \texttt{lambdastderr}: standard errors of estimated \( \lambda \)
• \( \lambda_t \): the \( t \)-statistic for \( \hat{\lambda} \)
• \( s^2 \): estimated variance of the error
• \( s^2_{\text{stderr}} \): standard errors of \( \hat{\sigma}^2 \)
• \( s^2_t \): the \( t \)-statistic for \( \hat{\sigma}^2 \)
• \( 1kt \): log-determinant computation type
• \( 1k \): log-likelihood
• \( \Sigma \): covariance matrix of errors
• \( \text{CPUtime} \): elapsed time in seconds

2.3 The function \texttt{printres()}

\texttt{printres(bundle b)}

\textbf{Return type}: \texttt{void}

\( b \): the bundle containing previous model estimations

The function prints estimation related results. A visual example of the output is presented in the empirical illustration.

2.4 The function \texttt{spatial\_GUI()}

\texttt{spatial\_GUI(series y, list X, matrix W, int modtype, int hess\_form, int lik\_type, scalar moments, scalar n\_rep, scalar poly\_ord)}

\textbf{Return type}: \texttt{bundle}

\( y \): the dependent variable series

\( X \): the list of regressors (without constant);

\( W \): the weight matrix;

\( \text{mod\_type} \): an integer which is 1 for the SAR, 2 for the SDM, 3 for SEM;

\( \text{hess\_form} \): a boolean (0 for analytical Hessian, 1 for mixed);

\( \text{int\_lik} \): an integer determining the log-determinant computation;
moments;
n_rep;
poly_ord.

\texttt{spatial\_GUI(\() \text{ allows for a unified function for the previous presented spatial models; in particular mod\_type identifies the specification. All other inputs and the output derive exactly from the previous functions, so additional explanations are not required. The graphical interface can be found in Model/Other Linear Models and is reported in Figure 1.}

![Figure 1: The spm GUI](image)
3 Example: spm_example.inp

As an example, we exploit data in *columbusdata.gdt* reporting observations for 49 contiguous Planning Neighborhoods in the city of Columbus, Ohio, for the year 1980 (Anselin [1988]). The three variables of interest are *crime*, *income* and *hoval*, which denote the aggregation of burglaries and vehicle thefts (per 1000 households), income (in thousand USD) and housing values (in thousand USD), respectively. The aim of this example is to include some sort of spatial dependence in a liner model where the dependent variable is *crime*. The columns of the weight matrix used in this example can be found as series in the dataset *columbuswm.gdt* and it is stored in the *gretl* session as the matrix *W* by

```gretl
open columbuswm.gdt --frompkg=spm
matrix W = { dataset } # Weight matrix
```

The following tables report the output for three different models, where spatial dependence is handled in different ways. Note that the output is the same for the command line interface and for the GUI. Table 1 reports the output for a SAR model, where we can find a recap of the options we have chosen for the estimation, such as the model, the log-determinant and the Hessian computation as well as the standard output of a regression.

**Table 1: Output of a SAR model**

```gretl
out_sar = sr(crime, X, W,,, 3,,,20)
printres(out_sar)
```

<table>
<thead>
<tr>
<th>coefficient</th>
<th>std. error</th>
<th>z</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>const</td>
<td>45.0793</td>
<td>7.17735</td>
<td>6.281</td>
</tr>
<tr>
<td>hoval</td>
<td>-0.265926</td>
<td>0.0884986</td>
<td>-3.005</td>
</tr>
<tr>
<td>income</td>
<td>-1.03162</td>
<td>0.305143</td>
<td>-3.381</td>
</tr>
<tr>
<td>rho</td>
<td>0.431023</td>
<td>0.117681</td>
<td>3.663</td>
</tr>
<tr>
<td>sigma2</td>
<td>95.4945</td>
<td>19.4878</td>
<td>4.900</td>
</tr>
</tbody>
</table>

CPU time: 0.03227(sec)
Log-likelihood: -165.40832
The output for a SDM in Table 2 is similar to the previous one, but in this case the spatial lag of the regressors are reported as $W_*$ followed by the name of the original series. Finally, Table 3 report the results for a SEM, whose interpretation should be now straightforward.

Table 2: Output of example script: SDM

```python
out_sdm = sr(crime, X, W, 1)
printres(out_sdm)
```

<table>
<thead>
<tr>
<th>coefficient</th>
<th>std. error</th>
<th>z</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>const</td>
<td>42.8224</td>
<td>12.6672</td>
<td>3.381</td>
</tr>
<tr>
<td>hoval</td>
<td>-0.293738</td>
<td>0.0892119</td>
<td>-3.293</td>
</tr>
<tr>
<td>income</td>
<td>-0.914223</td>
<td>0.331094</td>
<td>-2.761</td>
</tr>
<tr>
<td>$W_{hoval}$</td>
<td>0.245640</td>
<td>0.178917</td>
<td>1.373</td>
</tr>
<tr>
<td>$W_{income}$</td>
<td>-0.520284</td>
<td>0.565129</td>
<td>-0.9206</td>
</tr>
<tr>
<td>rho</td>
<td>0.426335</td>
<td>0.156234</td>
<td>2.729</td>
</tr>
<tr>
<td>sigma2</td>
<td>91.7912</td>
<td>18.8640</td>
<td>4.866</td>
</tr>
</tbody>
</table>

CPU time: 0.00496(sec)
Log-likelihood: -164.41141
Table 3: Output of example script: SEM

```r
out_sem = sem(crime, X, W, 2, 70, 200)
printres(out_sem)
```

<table>
<thead>
<tr>
<th></th>
<th>coefficient</th>
<th>std. error</th>
<th>z</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>const</td>
<td>59.8571</td>
<td>5.37399</td>
<td>11.14</td>
<td>8.17e-29 ***</td>
</tr>
<tr>
<td>hoval</td>
<td>-0.302306</td>
<td>0.0904318</td>
<td>-3.343</td>
<td>0.0008 ***</td>
</tr>
<tr>
<td>income</td>
<td>-0.938814</td>
<td>0.330523</td>
<td>-2.840</td>
<td>0.0045 ***</td>
</tr>
<tr>
<td>lambda</td>
<td>0.564076</td>
<td>0.135728</td>
<td>4.156</td>
<td>3.24e-05 ***</td>
</tr>
<tr>
<td>sigma2</td>
<td>95.4944</td>
<td>20.2054</td>
<td>4.726</td>
<td>2.29e-06 ***</td>
</tr>
</tbody>
</table>

CPU time: 0.26869(sec)
Log-likelihood: -166.39848
References

