Journal of Statistical Software

October 2023, Volume 107, Issue 8. *doi: 10.18637/jss.v107.i08*

GMM Estimators for Binary Spatial Models in R

Gianfranco Piras
The Catholic University of America
University of Chieti-Pescara

Mauricio Sarrias
Universidad de Talca

Abstract

Despite the huge availability of software to estimate cross-sectional spatial models, there are only few functions to estimate models dealing with spatial limited dependent variable. This paper fills this gap introducing the new R package **spldv**. The package is based on generalized methods of moment (GMM) estimators and includes a series of one- and two-step estimators based on different choices of the weighting matrix for the moments conditions in the first step, and different estimators for the variance-covariance matrix of the estimated coefficients. An important feature of **spldv** is that users can estimate the spatial Durbin model and compute the direct, indirect, and total effects in a friendly and flexible way.

Keywords: binary dependent variables, spatial model, GMM, R.

1. Introduction

In recent years, spatial econometrics has gained interest and importance in economics and related fields. The proliferation of applications in various areas has been supported by the growth of available software.[1] Currently, spatial econometrics routines for the estimation of spatial models are accessible in many commercial (and non commercial) software environments, such as MATLAB (LeSage and Pace 2009), Stata (Drukker, Prucha, and Raciborski 2013c; Drukker, Prucha, Peng, and Raciborski 2013a; Drukker, Prucha, and Raciborski 2013b) and PySal (Anselin and Rey 2014), among others. R (R Core Team 2023) is undoubtedly the open source environment that includes the largest variety of resources in spatial econometrics.

The oldest (and perhaps most famous) package is **spdep** (Bivand, Pebesma, and Gomez-Rubio 2013; Bivand and Wong 2018). The original version included functions to construct spatial weighting matrices and spatial lags, testing for spatial dependence, and a few additional

[1] For a deeper review see Bivand and Piras (2015) and Bivand, Millo, and Piras (2021).

routines to estimate the spatial lag and spatial error models through maximum likelihood (ML) and generalized method of moments (GMM). Few years later, **sphet** (Piras 2010; Piras and Postiglione 2022) was introduced to complement **spdep** with a complete treatment of a general cross-sectional model involving lags of the dependent variable, of the regressors, and of the error term. Dealing exclusively with GMM, **sphet** has the advantage that some of the regressors in the model can be endogenous. Recently, **spdep** split and gave rise to a new package named **spatialreg** (Bivand and Piras 2015). **spatialreg** inherited all the estimation functions that were originally included in **spdep** along with functions to compute impacts and properly interpret marginal effects in spatial models (LeSage and Pace 2009).[2] However, none of the packages mentioned allow for the estimation of models with limited dependent variables. This is very unfortunate given the fact that social agents are frequently faced with decisions that are intrinsically discrete.

The present paper introduces **spldv** (Sarrias and Piras 2023), a newly developed package that deals with GMM estimation of spatial models with a binary dependent variable. This package is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=spldv`. To the best of our knowledge, there are a few attempts to estimate spatial binary limited dependent variable models in R. The first is a linearized and one-step version of a GMM estimator available from the **McSpatial** package (McMillen 2013).[3] The second is the implementation available from **spatialprobit** based on a GIBBS sampler (Wilhelm and de Matos 2013). The package **ProbitSpatial** provides an approximate likelihood estimation (Martinetti and Geniaux 2021). Recently, Gómez-Rubio, Bivand, and Rue (2021) introduced the package R-**INLA** (available at `https://www.r-inla.org/download-install`), that implements an integrated nested Laplace approximation for the estimation of spatial models with binary outcomes. This paper is the first attempt to a systematic approach to estimate spatial binary models. The structure of the package follows the GMM methods put forth in Pinkse and Slade (1998), Klier and McMillen (2008) and Piras and Sarrias (2023), and includes a series of two-step estimators based on different choices of the weighting matrix for the moments conditions in the first step, and different estimators for the variance-covariance matrix of the estimated coefficients.[4]

The paper is organized as follows: Section 2 introduces the model and discusses the spatial effects and certain issues concerning inference on them. Specifically, Section 2.1 presents the general specification of the spatial autoregressive binary dependent model. Depending on the distributional assumption on the innovations, **spldv** allows for the estimation of probit as well as logit models. In Section 2.2, we introduce the spillover effects and two ways of approaching inference: one based on Monte Carlo simulation, and the other based on delta method. In Section 3 various one-step and two-step GMM estimator are reviewed and demonstrated using the main function of the package named `sbinaryGMM`. In the same section, we also illustrate

[2]The availability of software in R is not limited to cross-sectional methods. For example, the increased sophistication of methods for spatial panels encouraged the development of the package **splm** (Millo and Piras 2012). This package embeds a full treatment of static panel data models, with diagnostic tests and estimates of the impacts. There are also many other packages that are at various stages of development such as, for example, **spsur** (Angulo, Lopez, Minguez, and Mur 2022). For a more comprehensive review of those packages see Bivand *et al.* (2021).

[3]We included the code for the linearized GMM in our package since the original version does not allow for spatially lagged independent variables nor for the computation of proper marginal effects. **McSpatial** also allows to estimate other non-linear models using, for example, locally weighted regression, and semiparametric and conditionally parametric regression.

[4]Additional details will be given in later sections.

the function `sbinaryLGMM` that can be used to estimate a linearized version of the GMM estimator. We conclude this section by providing some guidance and recommendations on the use of GMM estimators. An empirical application is presented in Section 4 and Section 5 provides a comparison with other estimators available in R. Section 6 concludes the paper.

2. Spatial autoregressive binary dependent model

2.1. The model

The structural form of the spatial autoregressive specification of a binary dependent model (SARB) can be written in the following way:[5]

$$\begin{aligned} \mathbf{y}^* &= \mathbf{X}\boldsymbol{\beta} + \mathbf{W}\mathbf{X}\boldsymbol{\gamma} + \lambda\mathbf{W}\mathbf{y}^* + \boldsymbol{\epsilon}, \\ &= \mathbf{Z}\boldsymbol{\delta} + \lambda\mathbf{W}\mathbf{y}^* + \boldsymbol{\epsilon}, \\ \mathbf{y} &= \mathbb{1}\left[\mathbf{y}^* > 0\right], \end{aligned} \tag{1}$$

where $\mathbf{Z} = [\mathbf{X}, \mathbf{WX}]$, $\boldsymbol{\delta} = \left[\boldsymbol{\beta}^\top, \boldsymbol{\gamma}^\top\right]^\top$, $\mathbf{y}^*$ is an $n \times 1$ vector of latent (unobserved) continuous variable, $\mathbf{y}$ is the vector of observed binary variable, and $\mathbb{1}\left[\cdot\right]$ is the indicator function. In other words, the binary variable $y_i = 1$ if $y_i^* > 0$, and zero otherwise. The matrix $\mathbf{X}$ is an $n \times k$ matrix of explanatory variables whose first column is the intercept, $\mathbf{W}$ is a non-stochastic $n \times n$ spatial weighting matrix whose elements are w_{ij}, and $\mathbf{WX}$ is the $n \times l$ matrix of spatially lagged regressors. The matrix $\mathbf{WX}$ contains spatially lagged variables of (some of) the exogenous variables so that $l < k$.[6] The $n \times 1$ vector $\mathbf{Wy}^*$ is the spatial lag of the continuous but unobserved variable $\mathbf{y}^*$ which introduces (unobserved) endogeneity, and λ is the spatial autoregressive coefficient. The model in Equation 1 reduces to a conventional non-spatial binary model when λ and $\boldsymbol{\gamma}$ are both zero.

The full structural model is obtained once the distribution of the errors term $\boldsymbol{\epsilon}$ is postulated. In the probit SARB model, the errors $\boldsymbol{\epsilon}$ are assumed to follow a standard normal distribution, whereas in the logit SARB model, the errors are assumed to follow a standard logistic distribution with mean 0 and variance $\pi^2/3$. Thus:

1. Probit: $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2_\epsilon \mathbf{I}_n)$, with $\sigma^2_\epsilon = 1$.
2. Logit: $\boldsymbol{\epsilon} \sim L(\mathbf{0}, \sigma^2_\epsilon \mathbf{I}_n)$, with $\sigma^2_\epsilon = \pi^2/3$.

Under the usual assumption that all the diagonal elements of $\mathbf{W}$ are zero and that $|\lambda| < 1$, the reduced form equation for the SARB model in Equation 1 becomes:

$$\begin{aligned} \mathbf{y}^* &= \left(\mathbf{I}_n - \lambda\mathbf{W}\right)^{-1}\left(\mathbf{Z}\boldsymbol{\delta} + \boldsymbol{\epsilon}\right), \\ &= \mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta} + \mathbf{A}_\lambda^{-1}\boldsymbol{\epsilon}, \\ &= \mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta} + \mathbf{u}, \end{aligned}$$

[5] This model is also known as the spatial Durbin binary model (see LeSage and Pace 2009; Lacombe and LeSage 2018).

[6] Note that $l < k$ since $\mathbf{WX}$ cannot include the lagged intercept.

where $\mathbf{A}_\lambda = (\mathbf{I}_n - \lambda\mathbf{W})$, and $\mathbf{u} = \mathbf{A}_\lambda^{-1}\boldsymbol{\epsilon}$, so that $\mathbf{u} \sim (\mathbf{0}, \boldsymbol{\Sigma}_u)$, and:

$$\boldsymbol{\Sigma}_u = \mathsf{E}(\mathbf{u}\mathbf{u}^\top) = \sigma_\epsilon^2 (\mathbf{I}_n - \lambda\mathbf{W})^{-1}\left[(\mathbf{I}_n - \lambda\mathbf{W})^\top\right]^{-1} = \sigma_\epsilon^2\left(\mathbf{A}_\lambda^\top\mathbf{A}_\lambda\right)^{-1}, \tag{2}$$

which is a full matrix. Note that σ_ϵ^2 is held fixed at 1 or at $\pi^2/3$ in the probit and logit specification, respectively, for reasons related to the identification of the parameters. As a consequence, probit and logit estimates cannot be compared directly.

The expectation of the observed outcome, for all $i = 1, \ldots, n$, is

$$\begin{aligned}
\mathsf{E}(y_i) &= \mathsf{P}(y_i = 1), \\
&= \mathsf{P}\left(\{\mathbf{u}\}_i > -\left\{\mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta}\right\}_i\right), \\
&= F\left(\{\boldsymbol{\Sigma}_u\}_{ii}^{-1/2}\left\{\mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta}\right\}_i\right), \\
&= F(a_i),
\end{aligned} \tag{3}$$

where $F(\cdot)$ is either the normal or logistic standard cumulative distribution function (cdf), $\{\cdot\}_i$ is the ith element of the vector in brackets and $\{\cdot\}_{ii}$ is the ith diagonal element of the matrix in brackets. Let a_i in Equation 3 be the ith element of the following $n \times 1$ vector

$$\mathbf{a} = \mathbf{D}_\lambda^{-1}\mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta}, \tag{4}$$

where $\mathbf{D}_\lambda$ is an $n \times n$ diagonal matrix with diagonal elements representing the square root of the diagonal elements of the variance-covariance matrix of the error terms $\mathbf{u}$ given in Equation 2. Then, Equation 3 can also be written in vector form as

$$\mathsf{E}(\mathbf{y}) = F\left(\mathbf{D}_\lambda^{-1}\mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta}\right) = F(\mathbf{a}). \tag{5}$$

The parameters of the SARB model can be estimated by maximum likelihood (ML) under the assumption that the error term is distributed as $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma_\epsilon^2\mathbf{I}_n)$. In this case, the log-likelihood function can be expressed as

$$\ln L = \ln \Phi_n\left[\mathbf{Q}\mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta}; \mathbf{0}, \boldsymbol{\Sigma}_u\right], \tag{6}$$

where $\mathbf{Q}$ is a diagonal matrix with elements $2y_i - 1$, and Φ_n is an n-dimensional multivariate normal cumulative distribution with upper bounds corresponding to the first term in parenthesis, mean $\mathbf{0}$ and variance-covariance matrix $\boldsymbol{\Sigma}_u$. The main drawback of ML approach is that the evaluation of Equation 6 involves the computation of n-dimensional integrals and the inverse of matrix $\mathbf{A}_\lambda$, which is infeasible in practice.[7]

2.2. Spillover effects

Following Lacombe and LeSage (2018) and Billé and Arbia (2019), let $\mathbf{x}_{\cdot r} = (x_{1r}, x_{2r}, \ldots, x_{ir}, \ldots, x_{nr})^\top$ be an $n \times 1$ vector of observations of the rth regressor,

[7] Some progress has been made to reduce the dimensionality problem in the optimization of the log-likelihood function. Based on Vijverberg (1997), Beron and Vijverberg (2004) introduce the so-called recursive importance sampling (RIS) estimator and show how this estimator can be used to evaluate an n-dimensional normal probability. More recently, Billé and Leorato (2020), put forth a partial maximum likelihood estimator for a general spatial non-linear probit model, and perform a complete asymptotic analysis of their estimator.

$r = 1, \ldots, k$. Considering the expected value of the observed outcome in Equation 5, the marginal effects when the rth variable changes in all spatial units can be computed as[8]

$$
\begin{aligned}
\underset{(n\times n)}{\frac{\partial \mathsf{E}(\mathbf{y})}{\partial \mathbf{x}_{.r}^{\top}}} &= \begin{pmatrix} \frac{\partial \mathsf{E}(\mathbf{y})}{\partial x_{1r}} & \frac{\partial \mathsf{E}(\mathbf{y})}{\partial x_{2r}} & \cdots & \frac{\partial \mathsf{E}(\mathbf{y})}{\partial x_{nr}} \end{pmatrix}, \\
&= \begin{pmatrix} \frac{\partial \mathsf{E}(y_1)}{\partial x_{1r}} & \frac{\partial \mathsf{E}(y_1)}{\partial x_{2r}} & \cdots & \frac{\partial \mathsf{E}(y_1)}{\partial x_{nr}} \\ \frac{\partial \mathsf{E}(y_2)}{\partial x_{1r}} & \frac{\partial \mathsf{E}(y_2)}{\partial x_{2r}} & \cdots & \frac{\partial \mathsf{E}(y_2)}{\partial x_{nr}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathsf{E}(y_n)}{\partial x_{1r}} & \frac{\partial \mathsf{E}(y_n)}{\partial x_{2r}} & \cdots & \frac{\partial \mathsf{E}(y_n)}{\partial x_{nr}} \end{pmatrix}, \\
&= \operatorname{diag}(f(\mathbf{a}))\mathbf{D}_{\lambda}^{-1}\mathbf{A}_{\lambda}^{-1}\left(\mathbf{I}_n\beta_r + \mathbf{W}\gamma_r\right), \\
&= \mathbf{C}_r(\boldsymbol{\theta}),
\end{aligned} \tag{7}
$$

where $f(\cdot) = F'(\cdot)$ is the probability density function (pdf), $\operatorname{diag}(f(\mathbf{a}))$ is an operator that generates an $n \times n$ diagonal matrix with elements given by the $n \times 1$ vector $f(\mathbf{a})$, and $\boldsymbol{\theta} = (\boldsymbol{\delta}^{\top}, \lambda)^{\top}$ is the $(k + l + 1) \times 1$ vector of population parameters. Thus $\operatorname{diag}(f(\mathbf{a}))$ contains the pdf evaluated at the predictions for each of the observations on the diagonal.[9]

Every diagonal element of $\mathbf{C}_r(\boldsymbol{\theta})$ represents a direct effect. For example, the element $\mathbf{C}_{r,ii}(\boldsymbol{\theta})$ provides the partial change in the probability of observing $y_i = 1$ given a change of the same spatial unit i in x_r. This impact includes the effect of feedback loops where observation i affects observation j and observation j also affects observation i. The off-diagonal elements of $\mathbf{C}_{r,ij}(\boldsymbol{\theta})$ represent the indirect effect, that is, the partial change in the probability of observing $y_i = 1$ in unit i given a partial change in x_r for spatial unit j.

Since the change of each variable in each region implies n^2 potential marginal effects, LeSage and Pace (2009) propose the following scalar measures for the average total, direct and indirect effects:[10]

$$
\begin{aligned}
\mathrm{ATE}_r &= n^{-1}\boldsymbol{\imath}_n^{\top}\mathbf{C}_r\boldsymbol{\imath}_n, \\
\mathrm{ADE}_r &= n^{-1}\operatorname{tr}\left(\mathbf{C}_r\right), \\
\mathrm{AIE}_r &= \mathrm{ATE}_r - \mathrm{ADE}_r.
\end{aligned} \tag{8}
$$

Inference for the average total, direct and indirect effects can be approached in a couple of different ways: One can either use Monte Carlo (MC) methods to simulate a vector of effects using the sampling distribution of $\boldsymbol{\theta}$, or delta method.

A MC approximation to the average effects is obtained by generating a set of empirical values of the marginal effects evaluated at pseudo draws of $\boldsymbol{\theta}$ from the asymptotic distribution of the estimator. The algorithm can be summarized as follows:

1. Let $\widehat{\boldsymbol{\Omega}}_{\theta}$ be the estimated variance-covariance matrix of $\widehat{\boldsymbol{\theta}}$. Take a random draw of $\boldsymbol{\theta}$ (say, e.g., $\boldsymbol{\theta}^r$) from the normal distribution $N(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\Omega}}_{\theta})$.

2. Compute Equation 7 and the average effects in Equation 8, substituting $\widehat{\boldsymbol{\theta}}$ for $\boldsymbol{\theta}^r$.

3. Update $r = r + 1$, and go back to step 1.

[8]Unlike Billé and Arbia (2019), Lacombe and LeSage (2018) ignore $\mathbf{D}^{-1}$.

[9]Note that if $\lambda = 0$, so that $\mathbf{A}_{\lambda}^{-1} = \mathbf{I}_n$ and $\mathbf{A}_{ij,\lambda}^{-1} = 0$, we get the standard binary result where there are no spatial spillovers effects.

[10]See also Lacombe and LeSage (2018) for some simplification in the notation.

4. Repeat the previous steps R times, where R is a large number.
5. Calculate the empirical mean of the average marginal effects. The standard error of the average marginal effects across the R draws can be then used for inference.

The delta method computes the standard errors of the average marginal effects using asymptotic approximation. Let $f(\boldsymbol{\theta})$ be a 3×1 vector-valued function representing the average effects in Equation 8 for some variable. Then, the asymptotic variance-covariance matrix of $f(\boldsymbol{\theta})$ is approximated as:

$$V\left[f(\boldsymbol{\theta})\right] \approx \nabla f(\boldsymbol{\theta})^\top \boldsymbol{\Omega}_\theta \nabla f(\boldsymbol{\theta}),$$

where $\nabla f(\boldsymbol{\theta})$ is the $3 \times (k + l + 1)$ matrix of first derivatives evaluated at $\boldsymbol{\theta}$, also known as the Jacobian, and $\boldsymbol{\Omega}_\theta$ is the asymptotic variance-covariance matrix of the estimator. The standard errors of the average effects are then computed by taking the square root of the diagonal of $\widehat{V}\left[f(\widehat{\boldsymbol{\theta}})\right]$, where $\widehat{\boldsymbol{\theta}}$ is some consistent estimate of $\boldsymbol{\theta}$.

2.3. A simulated dataset

To show the capabilities of **spldv**, we create a simulated data set. The spatial layout of the observations is taken from the famous Boston data set which contains 506 spatial units corresponding to the Boston tracts (Bivand, Nowosad, and Lovelace 2023). The shapefile is loaded using `st_read` function from **sf** package (Pebesma 2018):

```
R> library("sf")
R> boston.tr <- st_read(system.file("shapes/boston_tracts.shp",
+    package = "spData")[1], quiet = TRUE)
```

Next, we create the spatial weighting matrix $\mathbf{W}$ based on contiguous boundaries using `poly2nb` and `nb2listw` functions from **spdep** (Bivand and Wong 2018):

```
R> library("spdep")
R> boston_nb <- poly2nb(boston.tr)
R> W <- nb2listw(boston_nb, style = "W")
```

The function `nb2listw` transforms a list of neighbors into an object of class '`listw`'. Since `style = "W"`, the spatial weighting matrix is row-standardized. The data generating process (DGP) is given by the following equations:

$$\begin{aligned} \mathbf{y}^* &= (\mathbf{I}_n - \lambda\mathbf{W})^{-1}\left(\beta_0\boldsymbol{\imath}_n + \beta_1\mathbf{x} + \beta_2\mathbf{W}\mathbf{x} + \beta_3\mathbf{z} + \boldsymbol{\epsilon}\right), \\ \mathbf{y} &= \mathbb{1}\left[\mathbf{y}^* > 0\right], \end{aligned} \tag{9}$$

where $\boldsymbol{\imath}_n$ in an $n \times 1$ vector of ones, the elements of the vector $\mathbf{x}$ are standard normal and $\mathbf{z}$ is uniformly distributed between 0 and 1. The true parameters are set to $\boldsymbol{\theta}_0 = (\beta_0, \beta_1, \beta_2, \beta_3, \lambda)^\top = (-0.5, 1, 1, 1, 0.6)^\top$. The error term is normally distributed with mean zero and standard deviation one yielding the SARB probit model.

After setting the seed, we create the random variables and the true coefficients. To generate $(\mathbf{I}_n - \lambda\mathbf{W})^{-1}$ we use the **spdep** function `invIrM`, while `lag.listw` creates the spatial lag of the explanatory variable $\mathbf{x}$:

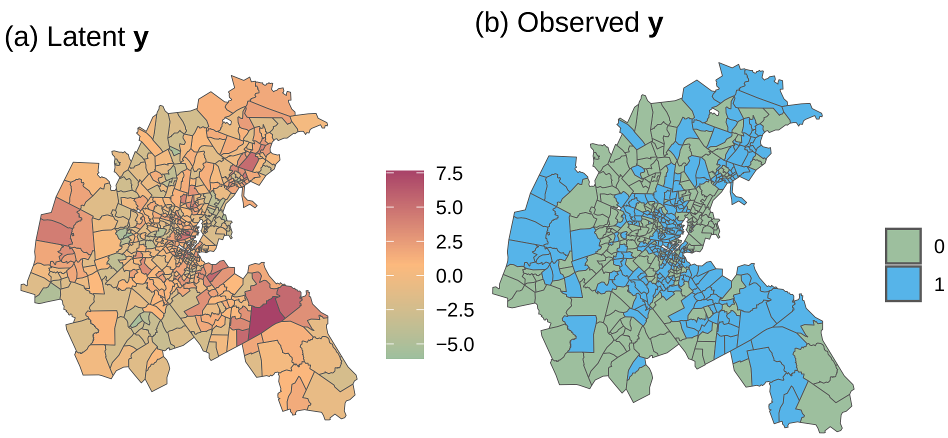

Figure 1: Spatial distribution of latent and binary dependent variable.

```
R> library("spatialreg")
R> set.seed(1)
R> n <- length(W$neighbours)
R> lambda <- 0.6
R> beta0 <- -0.5
R> beta1 <- beta2 <- beta3 <- 1
R> A_i <- invIrM(boston_nb, lambda)
R> x <- rnorm(n)
R> z <- runif(n)
R> Wx <- lag.listw(W, x)
R> epsilon <- rnorm(n)
```

The syntax to create the SARB probit model in Equation 9 is the following:

```
R> ystar <- A_i %*% (beta0 + beta1 * x + beta2 * Wx + beta3 * z + epsilon)
R> y <- as.numeric(ystar > 0)
R> data <- as.data.frame(cbind(y, x, z))
```

Figure 1 shows the spatial distribution of both $\mathbf{y}^*$ and $\mathbf{y}$ for the simulated data. For all those census tracts having $y_i^* > 0$ ($y_i^* \leq 0$), we observe that the dependent variables takes the value 1 (0). Since we know the true data generating process, we can also calculate the true average total, direct and indirect effects. The matrix $\mathbf{C}_r$ in Equation 7 for each variable $r = 1, 2$ is constructed as follows:[11]

```
R> Sigma_u <- tcrossprod(A_i)
R> sigma <- sqrt(diag(Sigma_u))
R> index <- beta0 + beta1 * x + beta2 * Wx + beta3 * z
R> a <- as.vector(A_i %*% index / sigma)
R> P <- diag(dnorm(a)) %*% A_i %*% diag(1 / sigma)
```

[11]The function `listw2mat` is available from **spdep** and transforms an object of class '`listw`' to a '`matrix`'.

```
R> C_x <- P %*% (diag(n) * beta1 + beta2 * listw2mat(W))
R> C_z <- P %*% (diag(n) * beta3)
```

Using Equation 8, the average marginal effects are:

```
R> TE_x <- (sum(C_x) / n)
R> DE_x <- (sum(diag(C_x))/ n)
R> IE_x <- TE_x - DE_x
R> TE_z <- (sum(C_z) / n)
R> DE_z <- (sum(diag(C_z))/ n)
R> IE_z <- TE_z - DE_z
R> Teffects <- rbind(cbind(TE_x, DE_x, IE_x), cbind(TE_z, DE_z, IE_z))
R> colnames(Teffects) <- c("ATE", "ADE", "AIE")
R> rownames(Teffects) <- c("x", "z")
R> Teffects
```

```
        ATE       ADE       AIE
x 0.9836068 0.2448900 0.7387169
z 0.4918034 0.2146375 0.2771659
```

For example, the ATE for x would be interpreted as: An increase of (approximately) 1 unit for x in all tracts increases the probability of observing $y_i = 1$ in the same unit and in units corresponding to other tracts by 98%, on average.

3. GMM estimators

3.1. Moment conditions

For the estimation of the SARB model using a GMM procedure we use population moment conditions based on the innovations. The main problem with the model in Equation 1 is that the error terms $\boldsymbol{\epsilon}$ are based on the unobserved dependent variables $\mathbf{y}^*$. For this reason, we have to rely on the so-called generalized residuals. The generalized residuals for spatial unit $i = 1, \dots, n$ are (see also Pinkse and Slade 1998)

$$\widetilde{u}_i(\boldsymbol{\theta}) = u_i \cdot \left[\frac{f(a_i)}{F(a_i)(1 - F(a_i))}\right], \tag{10}$$

where $u_i = y_i - F(a_i)$, a_i is the ith element of the vector $\mathbf{a}$ in Equation 4, and $\boldsymbol{\theta} = (\boldsymbol{\beta}^\top, \boldsymbol{\gamma}^\top, \lambda)^\top$ is the $(k + l + 1) \times 1$ vector of population parameters.

The $p \times 1$ population moment conditions are then

$$\mathsf{E}\left[\mathbf{h}_i \widetilde{u}_i(\boldsymbol{\theta})\right] = \mathbf{0}, \tag{11}$$

where $\mathbf{h}_i$ is a $p \times 1$ vector of instruments such that $p \geq k + l + 1$ for identification issues. The sample analog of the population moment conditions in Equation 11 in vector form is

$$\mathbf{g}(\boldsymbol{\theta}) = n^{-1}\mathbf{H}^\top \widetilde{\mathbf{u}}, \tag{12}$$

where $\tilde{\mathbf{u}}$ is the $n \times 1$ vector of the estimated generalized residuals. The $n \times p$ matrix of instrument is given by the independent columns of $\mathbf{H} = (\mathbf{Z}, \mathbf{WZ}, \mathbf{W}^2\mathbf{Z}, \ldots, \mathbf{W}^q\mathbf{Z})$ for some given q (Kelejian and Prucha 1998; Kelejian, Prucha, and Yuzefovich 2004). The $p \times p$ variance matrix of the moment conditions is given by

$$\begin{aligned} \mathbf{S}(\boldsymbol{\theta}) &= \mathsf{VAR}\left[\mathbf{g}(\boldsymbol{\theta})\right], \\ &= n^{-1}\mathbf{H}^\top\mathbf{T}\mathbf{H}, \end{aligned} \tag{13}$$

where $\mathbf{T}$ is a diagonal matrix whose elements are $f^2(a_i)/\left[F(a_i)(1 - F(a_i))\right]$.

3.2. General GMM estimator

Let $\widehat{\boldsymbol{\Psi}}$ be some $p \times p$ symmetric positive semidefinite moment-weighting matrix such that $\widehat{\boldsymbol{\Psi}} \xrightarrow{p} \boldsymbol{\Psi}$, then the corresponding GMM estimator is defined as

$$\widehat{\boldsymbol{\theta}}_{GMM} = \underset{\boldsymbol{\theta} \in \boldsymbol{\Theta}}{\operatorname{argmin}} \; J(\boldsymbol{\theta}) = \mathbf{g}^\top(\boldsymbol{\theta})\widehat{\boldsymbol{\Psi}}\mathbf{g}(\boldsymbol{\theta}), \tag{14}$$

where $\mathbf{g}$ is the $p \times 1$ vector of sample moments given in Equation 12. Under certain regularity conditions, the GMM estimator is consistent and asymptotically normally distributed with estimated variance-covariance matrix given by (see Pinkse and Slade 1998, p. 134)

$$\widehat{\mathbf{V}}_{GMM} = n\left[\left(\widehat{\mathbf{G}}^\top\mathbf{H}\right)\widehat{\boldsymbol{\Psi}}\left(\mathbf{H}^\top\widehat{\mathbf{G}}\right)\right]^{-1}\left[\left(\widehat{\mathbf{G}}^\top\mathbf{H}\right)\widehat{\boldsymbol{\Psi}}\widehat{\mathbf{S}}\widehat{\boldsymbol{\Psi}}\left(\mathbf{H}^\top\widehat{\mathbf{G}}\right)\right]\left[\left(\widehat{\mathbf{G}}^\top\mathbf{H}\right)\widehat{\boldsymbol{\Psi}}\left(\mathbf{H}^\top\widehat{\mathbf{G}}\right)\right]^{-1}, \tag{15}$$

where $\widehat{\mathbf{G}}$ is an $n \times (k + l + 1)$ matrix of first derivatives of the generalized residuals such that

$$\widehat{\mathbf{G}} = \left.\frac{\partial \tilde{\mathbf{u}}}{\partial \boldsymbol{\theta}^\top}\right|_{\widehat{\boldsymbol{\theta}}},$$

and the $p \times p$ matrix $\widehat{\mathbf{S}}$ is a consistent estimator of Equation 13.

Unlike the traditional GMM estimator for spatial models with continuous dependent variables (see for example Kelejian and Prucha 1998, 1999), the GMM estimator for the SARB model requires the computation of the inverse of the $n \times n$ matrix $\mathbf{A}_\lambda = (\mathbf{I}_n - \lambda\mathbf{W})$. Of course, the inversion slows down the optimization procedure increasing the computation time. However, these computation issues can be simplified using matrix approximation methods as proposed by Santos and Proença (2019).[12]

One-step GMM estimators

The one-step procedure estimates the model parameters based on an initial weight matrix $\widehat{\boldsymbol{\Psi}}$. Following Klier and McMillen (2008) and Pinkse and Slade (1998), we consider two types of one-step GMM estimators. The first estimator is obtained by setting $\widehat{\boldsymbol{\Psi}} = \left(n^{-1}\mathbf{H}^\top\mathbf{H}\right)^{-1}$, that is

[12] To reduce computation time, the current implementation of **spldv** takes advantage of sparse matrices routines available from the **Matrix** package (Bates, Maechler, and Jagan 2023). Users can also reduce the computation time when n is large by applying the expansion $\mathbf{A}_\lambda^{-1} = \sum_{q=0}^{\infty}(\lambda\mathbf{W})^q$ setting the arguments `approximation = TRUE` and fixing q with the argument `pw`. Of course, this approximation is effective as long as the spatial weighting matrix is sparse. See the examples in Section 5. Future releases of the package **spldv** will include other ways to speed up the computation such as, for example, the approximation methods proposed by Santos and Proença (2019).

$$\widetilde{\boldsymbol{\theta}}_{\text{OS},H} = \underset{\boldsymbol{\theta}\in\boldsymbol{\Theta}}{\text{argmin}} \;\; J(\boldsymbol{\theta}) = \left(\frac{1}{n}\widetilde{\mathbf{u}}^\top \mathbf{H}\right)\left(n^{-1}\mathbf{H}^\top\mathbf{H}\right)^{-1}\left(\frac{1}{n}\mathbf{H}^\top\widetilde{\mathbf{u}}\right). \tag{16}$$

This one-step estimator is a natural adaptation of the estimator proposed by Klier and McMillen (2008) to estimate a spatial lag binary dependent model with a logistically distributed error term. The variance-covariance matrix for $\widetilde{\boldsymbol{\theta}}_{\text{OS},H}$ can be estimated as:

$$\begin{aligned}\widehat{\mathbf{V}}\left(\widetilde{\boldsymbol{\theta}}_{\text{OS},H}\right) =& n\left[\widetilde{\mathbf{G}}^\top\mathbf{H}\left(\mathbf{H}^\top\mathbf{H}\right)^{-1}\mathbf{H}^\top\widetilde{\mathbf{G}}\right]^{-1}\left[\widetilde{\mathbf{G}}^\top\mathbf{H}\left(\mathbf{H}^\top\mathbf{H}\right)^{-1}\widetilde{\mathbf{S}}\left(\mathbf{H}^\top\mathbf{H}\right)^{-1}\mathbf{H}^\top\widetilde{\mathbf{G}}\right] \\ &\times\left[\widetilde{\mathbf{G}}^\top\mathbf{H}\left(\mathbf{H}^\top\mathbf{H}\right)^{-1}\mathbf{H}^\top\widetilde{\mathbf{G}}\right]^{-1},\end{aligned} \tag{17}$$

where the estimator for $\widetilde{\mathbf{S}}$ is

$$\widetilde{\mathbf{S}}\left(\widetilde{\boldsymbol{\theta}}_{\text{OS},H}\right) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{h}_i\left[\frac{\phi^2(\widetilde{a}_i)}{\Phi(\widetilde{a}_i)(1-\Phi(\widetilde{a}_i))}\right]\mathbf{h}_i^\top, \tag{18}$$

and $\widetilde{a}_i$ is the ith element of Equation 4 evaluated at $\widetilde{\boldsymbol{\theta}}_{\text{OS},H}$.

As in Pinkse and Slade (1998), the second one-step estimator sets $\widehat{\boldsymbol{\Psi}} = \mathbf{I}_p$ yielding

$$\widetilde{\boldsymbol{\theta}}_{\text{OS},\mathbf{I}} = \underset{\boldsymbol{\theta}\in\boldsymbol{\Theta}}{\text{argmin}} \;\; J(\boldsymbol{\theta}) = \mathbf{g}^\top\mathbf{g}. \tag{19}$$

This estimator can be viewed as an unweighted nonlinear least squares estimator in which $J(\boldsymbol{\theta})$ is the sum of p squared sample average of the moment conditions (Cameron and Trivedi 2005). The estimator of the variance-covariance matrix in this case is:

$$\widehat{\mathbf{V}}\left(\widetilde{\boldsymbol{\theta}}_{\text{OS},\mathbf{I}}\right) = n\left[\widehat{\mathbf{G}}^\top\mathbf{H}\mathbf{H}^\top\widehat{\mathbf{G}}\right]^{-1}\left[\widehat{\mathbf{G}}^\top\mathbf{H}\widehat{\mathbf{S}}\mathbf{H}^\top\widehat{\mathbf{G}}\right]\left[\widehat{\mathbf{G}}^\top\mathbf{H}\mathbf{H}^\top\widehat{\mathbf{G}}\right]^{-1}.$$

Under the assumptions made, the choice of the weight matrix $\widehat{\boldsymbol{\Psi}}$ should not affect the consistency of the one-step estimators. However, we should observe differences in finite samples (see Piras and Sarrias 2023).

Demonstration of one-step GMM estimators

The one-step GMM estimator $\widetilde{\boldsymbol{\theta}}_{\text{OS},H}$ given in Equation 16 can be fitted using the `sbinaryGMM` function from **spldv** and the simulated data from Section 2.3 as follows:

```
R> library("spldv")

R> os_R <- sbinaryGMM(y ~ x + z | x, link = "probit", listw = W,
+    nins = 2, data = data, type = "onestep", winitial = "optimal")

First-step GMM optimization based on optimal initial weight matrix
```

The `formula` argument of `sbinaryGMM` consists of two parts with the general form being `y ~ x | wx`, where `y` is the binary dependent variable, `x` are the exogenous regressors, and `wx` are the spatially lagged independent variables, which must appear also in the first part.[13] The argument `link` indicates whether a logit (`link = "logit"`) or probit (`link = "probit"`) model should be fitted. As in other packages dealing with spatial models in R, the argument `listw` handles the spatial weight matrix. The argument can be either of class '`listw`', '`matrix`', or '`Matrix`'. The argument `nins` defines the number of lags to be included in the instrument matrix. For example, if `nins = 2` (which is also the default value), then $\mathbf{H} = (\mathbf{Z}, \mathbf{WZ}, \mathbf{W}^2\mathbf{Z})$. The instruments are stored in the object `os_R` and can be retrieved as follows:

```
R> head(os_R$H)

6 x 8 Matrix of class "dgeMatrix"
  (Intercept)          x          z       lag_x      W*z     W*lag_x
1           1 -0.6264538 0.08492106 -0.17162223 0.5640233 -0.03326320
2           1  0.1836433 0.99477051 -0.17181750 0.4196241 -0.02199433
3           1 -0.8356286 0.40144412 -0.08190726 0.6560343 -0.29234035
4           1  1.5952808 0.89564630 -0.28573648 0.6423895  0.04205991
5           1  0.3295078 0.87685841 -0.02932424 0.5492231  0.01129188
6           1 -0.8204684 0.29648480  0.45128864 0.5417832 -0.03034083
       WW*z    WW*lag_x
1 0.4274437 -0.04584257
2 0.6010576 -0.07847112
3 0.5051597 -0.06991665
4 0.5416661 -0.08334591
5 0.5612784 -0.01820690
6 0.5327470  0.08178005
```

Note that the matrix of instruments does not contain the variable `WW*x` since `WW*x` and `W*lag_x` are linearly dependent.

The key arguments of `sbinaryGMM` for fitting different GMM estimators are `type` and `winitial`. The argument `type` is a string indicating whether the one-step (`type = "onestep"`), or two-step GMM (`type = "twostep"`) should be computed. The argument `winitial` is also a string indicating the initial moment-weighting matrix $\widehat{\boldsymbol{\Psi}}$ for the one-step estimator; it can be either `winitial = "optimal"` and then $\widehat{\boldsymbol{\Psi}} = \left(n^{-1}\mathbf{H}^\top\mathbf{H}\right)^{-1}$ as in Equation 16 (the default) or `winitial = "identity"` and then $\widehat{\boldsymbol{\Psi}} = \mathbf{I}_p$ as in Equation 19.

```
R> summary(os_R)

         ------------------------------------------------------------
                           SLM Binary Model by GMM
         ------------------------------------------------------------

Call:
sbinaryGMM(formula = y ~ x + z | x, data = data, listw = W, nins = 2,
    link = "probit", winitial = "optimal", type = "onestep")
```

[13] This rules out situations in which one of the regressors can be specified only in lagged form.

```
Coefficients:
             Estimate Std. Error z-value  Pr(>|z|)
(Intercept) -0.447141   0.124552 -3.5900 0.0003307 ***
x            0.907657   0.110266  8.2315 2.220e-16 ***
z            0.888341   0.244215  3.6375 0.0002753 ***
lag_x        1.002749   0.279634  3.5859 0.0003359 ***
lambda       0.605980   0.096286  6.2935 3.103e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample size: 506
```

The output generated from the `summary` method shows that the point estimates are close to the true population parameters $\boldsymbol{\theta}_0 = (\beta_0, \beta_1, \beta_2, \beta_3, \lambda) = (-0.5, 1, 1, 1, 0.6)$. The standard errors for the estimated coefficients are computed using the variance-covariance matrix in Equation 17.

Next, we compute the one-step GMM estimator $\widetilde{\boldsymbol{\theta}}_{\text{OS},I}$ given in Equation 19 by setting `winitial = "identity"` and `type = "onestep"`. A glance at the two outputs reveals that the two estimators $\widetilde{\boldsymbol{\theta}}_{\text{OS},H}$ and $\widetilde{\boldsymbol{\theta}}_{\text{OS},I}$ produce very similar results. This is not surprising given the sample size of the simulated data. However, we might expect larger differences for smaller sample sizes.

```
R> os_I <- sbinaryGMM(y ~ x + z | x, link = "probit", listw = W,
+    nins = 2, data = data, type = "onestep", winitial = "identity")

First-step GMM optimization based on identity initial weight matrix

R> summary(os_I)

        ------------------------------------------------------------
                       SLM Binary Model by GMM
        ------------------------------------------------------------

Call:
sbinaryGMM(formula = y ~ x + z | x, data = data, listw = W, nins = 2,
    link = "probit", winitial = "identity", type = "onestep")


Coefficients:
            Estimate Std. Error z-value  Pr(>|z|)
(Intercept) -0.48218    0.13291 -3.6278 0.0002859 ***
x            0.91262    0.11108  8.2157 2.220e-16 ***
z            0.95661    0.26043  3.6732 0.0002395 ***
lag_x        1.02183    0.29035  3.5193 0.0004326 ***
lambda       0.59996    0.10335  5.8053 6.425e-09 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample size: 506
```

The average marginal effects are calculated using the function `impacts` which is an S3 method. This function also allows to estimate the standard errors of the average total, direct and indirect effects using either MC approximation or delta method (see Section 2.2).

The average marginal effects for the model `os_R` can be computed via MC using `R = 100` draws by typing:[14]

```
R> set.seed(1)
R> summary(impacts(os_R, type = "mc", R = 100, approximation = TRUE,
+    pw = 6))

------------------------------------------------------
(a) Total effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.96139    0.06943  13.847  < 2e-16 ***
z 0.46340    0.13435   3.449 0.000562 ***

------------------------------------------------------
(b) Direct effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.23324    0.01376  16.953  < 2e-16 ***
z 0.20029    0.05240   3.822 0.000132 ***

------------------------------------------------------
(c) Indirect effects :
------------------------------------------------------
     dydx Std. error z value Pr(> z)
x 0.72815    0.06937  10.497 < 2e-16 ***
z 0.26312    0.09749   2.699 0.00696 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The point estimates of the average effects are computed using Equation 8 and are very close to the true average effects. The argument `approximation = TRUE` replace the inverse of $\mathbf{A}_\lambda$ by the expansion $\mathbf{A}_\lambda^{-1} = (\mathbf{I} - \lambda\mathbf{W})^{-1} = \sum_{q=0}^{\infty}(\lambda\mathbf{W})^q$. Even though it was not strictly necessary in this context, the approximation can be used to speed up the algorithm when the sample size is particularly large. The argument `pw = 6` indicates the number of powers used for the approximation. It is important to note that increasing the number of powers makes $\mathbf{W}$ denser and, consequently, the computation time is considerably larger.

[14]Since the MC approach uses random draws, we need to set the seed to make the results fully reproducible. Additionally, the precision increases with the number of draws. In this example, we fix the number of draws to 100, however, larger numbers should be considered in practice.

The delta method to estimate the standard errors can be used by setting `type = "delta"` in the `impacts` function as in the following example:

```
R> summary(impacts(os_R, type = "delta", approximation = TRUE, pw = 6))

------------------------------------------------------
(a) Total effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.96664    0.06827  14.160  < 2e-16 ***
z 0.44949    0.13570   3.312 0.000925 ***

------------------------------------------------------
(b) Direct effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.23562    0.01360  17.330  < 2e-16 ***
z 0.19945    0.05273   3.783 0.000155 ***

------------------------------------------------------
(c) Indirect effects :
------------------------------------------------------
     dydx Std. error z value Pr(> z)
x 0.73102    0.06894  10.604  <2e-16 ***
z 0.25004    0.09752   2.564  0.0104 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Lacombe and LeSage (2018) do not consider heteroskedasticity when computing the average marginal effects. This means that they ignore $\mathbf{D}^{-1}$ in Equation 7 so that:

$$\mathbf{C}_r(\boldsymbol{\theta}) = \operatorname{diag}\left(f\left(\mathbf{A}_\lambda^{-1}\mathbf{Z}\boldsymbol{\delta}\right)\right)\mathbf{A}_\lambda^{-1}\left(\mathbf{I}_n\beta_r + \mathbf{W}\gamma_r\right). \tag{20}$$

The function `impacts` accommodates this situation by simply setting the argument `het = FALSE`:

```
R> summary(impacts(os_R, type = "delta", approximation = TRUE,
+    pw = 6, het = FALSE))

------------------------------------------------------
(a) Total effects :
------------------------------------------------------
     dydx Std. error z value Pr(> z)
x 1.01436    0.08407  12.065 < 2e-16 ***
z 0.47168    0.14616   3.227 0.00125 **

------------------------------------------------------
```

```
(b) Direct effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.24707    0.01517  16.287  < 2e-16 ***
z 0.20924    0.05474   3.822 0.000132 ***

------------------------------------------------------
(c) Indirect effects :
------------------------------------------------------
     dydx Std. error z value Pr(> z)
x 0.76729    0.08139   9.427  <2e-16 ***
z 0.26244    0.10567   2.484   0.013 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The total effect for each spatial unit can be computed as $\boldsymbol{\imath}_n^\top \mathbf{C}_r(\widehat{\boldsymbol{\theta}})$. The following syntax shows how the user can obtain the total effect of `x` from the object `os_R`. First, we extract the estimated coefficients and the variables used to estimate the model:

```
R> theta.hat <- coef(os_R)
R> lambda.hat <- theta.hat["lambda"]
R> beta.hat <- theta.hat["x"]
R> wbeta.hat <- theta.hat["lag_x"]
R> X <- os_R$X
R> n <- nrow(X)
R> W <- os_R$listw
R> I <- Diagonal(n)
R> A_i <- solve(I - lambda.hat * W)
R> sigmas_u <- sqrt(diag(tcrossprod(A_i)))
```

Now, we compute the total effects as follows:

```
R> D_i <- Diagonal(x = 1 / sigmas_u)
R> a <- D_i %*% A_i %*% X %*% as.matrix(theta.hat[1:ncol(X)])
R> dfa <- Diagonal(x = dnorm(as.numeric(a)))
R> Chat_x <- dfa %*% D_i %*% A_i %*% (I * beta.hat + wbeta.hat * W)
R> TE_hat <- as.vector(t(rep(1, n)) %*% Chat_x)
```

Figure 2 shows the estimated total effect (red curve) for each observation (sorted from low-to-high) alongside with the true total effects (black curve). The graph shows that the two curves are very close to each other.

Two-step GMM estimator

It is widely known that one can gain efficiency by computing two-step estimators of the form:

$$\widehat{\boldsymbol{\theta}}_{\mathrm{TS}} = \underset{\boldsymbol{\theta}\in\boldsymbol{\Theta}}{\operatorname{argmin}}\; J(\boldsymbol{\theta}) = \mathbf{g}^\top(\boldsymbol{\theta})\widehat{\boldsymbol{\Psi}}\mathbf{g}(\boldsymbol{\theta}). \tag{21}$$

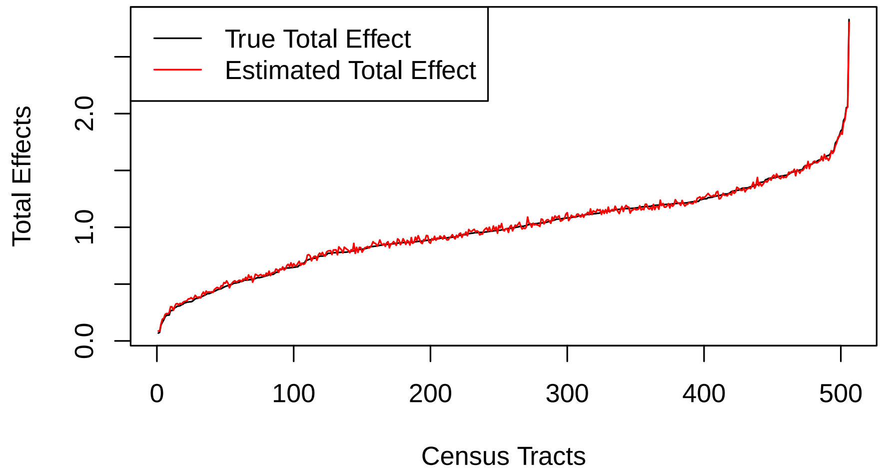

Figure 2: True vs estimated total effects (observational level).

where $\widehat{\boldsymbol{\Psi}} = \widetilde{\mathbf{S}}^{-1}$, and $\widetilde{\mathbf{S}}$ is an estimate of the variance-covariance matrix $\mathbf{S}$ based on either of the one-step estimators.

The procedure can be summarized by the following steps (Piras and Sarrias 2023):

1. First, minimize the objective function in Equation 14 by choosing either $\widehat{\boldsymbol{\Psi}} = \mathbf{I}_p$ or $\widehat{\boldsymbol{\Psi}} = (n^{-1}\mathbf{H}^\top\mathbf{H})^{-1}$ to obtain $\widetilde{\boldsymbol{\theta}}_{\text{OS}}$. Note that in either case, $\widetilde{\boldsymbol{\theta}}_{\text{OS}}$ is consistent as $n \to \infty$ but not fully efficient.

2. Second, use $\widetilde{\boldsymbol{\theta}}_{\text{OS}}$ to obtain the residuals from the first step and calculate $\widetilde{\mathbf{S}}$ using Equation 18. Set $\widehat{\boldsymbol{\Psi}} = \widetilde{\mathbf{S}}^{-1}$ and minimize Equation 21 to obtain the final round estimate $\widehat{\boldsymbol{\theta}}_{TS}$. The estimated asymptotic variance is given by:

$$\widehat{\mathbf{V}}_{EGMM} = n\left[\widehat{\mathbf{G}}^\top\mathbf{H}\widehat{\boldsymbol{\Psi}}\mathbf{H}^\top\widehat{\mathbf{G}}\right]^{-1}, \tag{22}$$

   where $\widehat{\boldsymbol{\Psi}} = \widetilde{\mathbf{S}}^{-1}$.[15] However, as pointed out by Cameron and Trivedi (2005) in finite samples the estimator in Equation 22 might be biased. In such cases, it could be better to use Equation 15, where $\widehat{\boldsymbol{\Psi}} = \widetilde{\mathbf{S}}^{-1}$, and $\widehat{\mathbf{S}}$ is computed using Equation 18 and $\widehat{\boldsymbol{\theta}}_{\text{TS}}$.

Demonstration of two-step GMM estimators

The two-step GMM estimators are obtained by setting `type = "twostep"`. The argument `winitial` allows the user to estimate $\widetilde{\boldsymbol{\theta}}_{OS,H}$ or $\widetilde{\boldsymbol{\theta}}_{OS,I}$ as the first-step estimator to construct the moment weighing matrix $\widetilde{\mathbf{S}}$ in Equation 18.

Thus, to fit a two-step GMM using $\widetilde{\boldsymbol{\theta}}_{OS,H}$ as first-step estimate we type:

```
R> ts_H <- sbinaryGMM(y ~ x + z | x, link = "probit", listw = W,
+    nins = 2, data = data, type = "twostep", winitial = "optimal")
```

[15] Since any consistent estimate of $\boldsymbol{\theta}$ can be used, an alternative is to evaluate $\mathbf{S}$ at $\widehat{\boldsymbol{\theta}}_{\text{TS}}$.

```
First-step GMM optimization based on optimal initial weight matrix

Second-step GMM optimization using S moment-weighing matrix

R> summary(ts_H)

        ------------------------------------------------------------
                      SLM Binary Model by GMM
        ------------------------------------------------------------

Call:
sbinaryGMM(formula = y ~ x + z | x, data = data, listw = W, nins = 2,
    link = "probit", winitial = "optimal", type = "twostep")


Coefficients:
             Estimate Std. Error z-value  Pr(>|z|)
(Intercept) -0.451177   0.124341 -3.6285 0.0002850 ***
x            0.909178   0.109571  8.2976 < 2.2e-16 ***
z            0.894382   0.243829  3.6681 0.0002444 ***
lag_x        1.015515   0.277896  3.6543 0.0002579 ***
lambda       0.602701   0.096319  6.2573 3.916e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample size: 506
```

By default, the `summary` method computes the standard errors using Equation 15, where $\widehat{\boldsymbol{\Psi}} = \widetilde{\mathbf{S}}^{-1}$, and $\widehat{\mathbf{S}}$ is computed using Equation 18 and $\widehat{\boldsymbol{\theta}}_{\mathrm{TS}}$. As an alternative, in order to estimate the standard errors using the more efficient variance-covariance matrix in Equation 22, we type

```
R> summary(ts_H, vce = "efficient")

        ------------------------------------------------------------
                      SLM Binary Model by GMM
        ------------------------------------------------------------

Call:
sbinaryGMM(formula = y ~ x + z | x, data = data, listw = W, nins = 2,
    link = "probit", winitial = "optimal", type = "twostep")


Coefficients:
             Estimate Std. Error z-value  Pr(>|z|)
(Intercept) -0.451177   0.124517 -3.6234 0.0002907 ***
x            0.909178   0.109736  8.2851 2.220e-16 ***
```

```
z            0.894382   0.244131   3.6635 0.0002488 ***
lag_x        1.015515   0.278442   3.6471 0.0002652 ***
lambda       0.602701   0.096452   6.2487 4.138e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample size: 506
```

Note that in this case, the overall statistical significance of the estimated coefficients do not change since the standard errors are only slightly different.

To incorporate the efficient VC matrix in the computation of the average marginal effects, we can use the following syntax:

```
R> summary(impacts(ts_H, type = "delta", vce = "efficient",
+    approximation = TRUE, pw = 6))

------------------------------------------------------
(a) Total effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.96667    0.06818  14.178  < 2e-16 ***
z 0.44920    0.13497   3.328 0.000874 ***

------------------------------------------------------
(b) Direct effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.23577    0.01356  17.389  < 2e-16 ***
z 0.20052    0.05267   3.807 0.000141 ***

------------------------------------------------------
(c) Indirect effects :
------------------------------------------------------
     dydx Std. error z value Pr(> z)
x 0.73090    0.06888  10.611  <2e-16 ***
z 0.24869    0.09697   2.565  0.0103 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.3. Linearized GMM estimator

One of the main drawback of the previous GMM estimators is that they require the inversion of the $n \times n$ matrix $\mathbf{A}_\lambda$ which can be very time consuming for large data sets. To overcome this problem, Klier and McMillen (2008) propose a linearized version of the one-step GMM estimator around the starting point $\lambda = 0$. When $\lambda = 0$, $\boldsymbol{\beta}$ is estimated consistently by standard Probit/Logit model and $\mathbf{A}_\lambda^{-1} = \mathbf{I}_n$ so that no matrices need to be inverted. Linearizing the generalized residuals around the initial estimates of $\boldsymbol{\theta}$ (i.e, $\boldsymbol{\theta}^0$), Klier and McMillen (2008) obtain $\widetilde{u}_i \approx \widetilde{u}_i^0 - \mathbf{G}(\boldsymbol{\theta} - \boldsymbol{\theta}^0)$. If we define $\upsilon_i = \widetilde{u}_i^0 + \mathbf{G}\boldsymbol{\theta}^0 - \mathbf{G}\boldsymbol{\theta}$ and letting $\boldsymbol{\Psi} = (\mathbf{H}^\top\mathbf{H})$, the objective function becomes $\boldsymbol{\upsilon}^\top\mathbf{H}\left(\mathbf{H}^\top\mathbf{H}\right)^{-1}\mathbf{H}^\top\boldsymbol{\upsilon}$.

The steps for the linearized spatial probit/logit model are the following (Klier and McMillen 2008, p. 462):

1. Estimate the spatial model by standard probit/logit model, in which spatial autocorrelation and heteroskedasticity are ignored. The estimated values are $\widehat{\boldsymbol{\beta}}_0$. Calculate the generalized residuals in Equation 10 assuming that $\lambda = 0$, and the gradient terms $\mathbf{G}_\beta = -\partial\tilde{\mathbf{u}}/\partial\boldsymbol{\beta}$ and $\mathbf{G}_\lambda = -\partial\tilde{\mathbf{u}}/\partial\lambda$.
2. The second step is a two-stage least squares estimator of the linearized model. Thus, regress $\mathbf{G}_\beta$ and $\mathbf{G}_\lambda$ on $\mathbf{H}$. The predicted values are $\widehat{\mathbf{G}} = \left[\widehat{\mathbf{G}}_\beta, \widehat{\mathbf{G}}_\lambda\right]$. Then regress $u_0 + \mathbf{G}_\beta^\top\widehat{\boldsymbol{\beta}}_0$ on $\widehat{\mathbf{G}}$. The resulting coefficients are the estimated values of $\boldsymbol{\beta}$ and λ.

The variance-covariance matrix can be computed using the heteroskedasticity correction based on the residuals of the (last) two-stage least squares estimator of the linearized model.

Demonstration of linearized GMM estimator

The linearized GMM estimator can be computed using the function `sbinaryLGMM` as follows:[16]

```
R> lgmm <- sbinaryLGMM(y ~ x + z | x, link = "probit", listw = W,
+    nins = 2, data = data)
R> summary(lgmm)

        ------------------------------------------------------------
                     SLM Binary Model by Linearized GMM
        ------------------------------------------------------------

Call:
sbinaryLGMM(formula = y ~ x + z | x, data = data, listw = W,
    nins = 2, link = "probit")


Coefficients:
            Estimate Std. Error z-value  Pr(>|z|)
(Intercept) -0.43962    0.12665 -3.4710 0.0005185 ***
x            0.67689    0.11133  6.0800 1.202e-09 ***
z            0.85513    0.22470  3.8057 0.0001414 ***
lag_x        0.70256    0.36642  1.9174 0.0551892 .
lambda       0.74306    0.17462  4.2553 2.088e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[16]The functions `spprobit` and `splogit` of **McSpatial** package (McMillen 2013) also implement the linearized GMM probit and logit model, respectively. However, they do not allow for spatially lagged independent variables nor for the computation of the marginal effects. See Section 5.

```
Sample size: 506
```

Similar to `sbinaryGMM`, the `formula` argument of `sbinaryLGMM` consists of two parts following the general form `y ~ x | wx`. The argument `link` indicates whether a logit (`link = "logit"`) or probit (`link = "probit"`) model should be fitted and the argument `nins` indicates the number of spatial lags of the exogenous variables to be used as instruments.

The average marginal effects can also be computed using the function `impacts`:

```
R> summary(impacts(lgmm, type = "delta", approximation = TRUE, pw = 6))

------------------------------------------------------
(a) Total effects :
------------------------------------------------------
    dydx Std. error z value  Pr(> z)
x 0.9865     0.1323   7.458 8.79e-14 ***
z 0.6116     0.2519   2.428   0.0152 *

------------------------------------------------------
(b) Direct effects :
------------------------------------------------------
     dydx Std. error z value  Pr(> z)
x 0.19795    0.02486   7.964 1.66e-15 ***
z 0.20848    0.05186   4.020 5.83e-05 ***

------------------------------------------------------
(c) Indirect effects :
------------------------------------------------------
    dydx Std. error z value Pr(> z)
x 0.7886     0.1380   5.715 1.1e-08 ***
z 0.4031     0.2290   1.760  0.0783 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.4. Properties of GMM estimators

Piras and Sarrias (2023) perform a Monte Carlo experiment comparing one- and two-step GMM, the LGMM, and the RIS estimators of the SARB model.[17] They found that the LGMM is the estimator that presents the largest bias and the highest standard deviation among all the considered estimators, particularly if the degree of spatial dependence is significantly large. Similar evidence was also reported by Calabrese and Elkink (2014) who compared five estimators: Expectation Maximization, GIBBS, RIS, one-step GMM and LGMM.

In terms of one-step GMM estimator, Piras and Sarrias (2023) also found that the initial optimal moment-weighting matrix, $\widehat{\boldsymbol{\Psi}} = \left(n^{-1}\mathbf{H}^\top\mathbf{H}\right)^{-1}$, outperforms the estimator based on

[17] As a referee correctly pointed out, users may need some guidance in choosing the various GMM estimators. Hence we added this subsection that is almost entirely based on the results obtained by Piras and Sarrias (2023).

the identity matrix, $\widehat{\boldsymbol{\Psi}} = \mathbf{I}_p$, in both efficiency and bias. This result may explain the poor performance of the one-step GMM in Calabrese and Elkink (2014)'s study. Finally, Piras and Sarrias (2023) show that the efficiency not always improves when a two-step estimator is adopted and the sample size is small. However, if the sample size is sufficiently large (i.e., in the order of 500 observations), then the two-step estimators achieve similar efficiency as the RIS.

4. An empirical example

In this section, we provide an empirical application. For this purpose, we use one of the Anselin (1988)'s dataset corresponding to a cross-section of 49 neighborhoods in Columbus, Ohio. The resulting model is used to explain the crime rate as a function of household income and housing values. In particular, the variables contained in the dataset are the following:

- `CRIME`: residential burglaries and vehicle thefts per thousand household in the neighborhood.
- `HOVAL`: housing value in USD 1,000.
- `INC`: household income in USD 1,000.

We start our analysis by loading the `columbus` shapefile and the corresponding spatial weights matrix into R:

```
R> columbus <- st_read(system.file("shapes/columbus.shp",
+     package = "spData")[1], quiet = TRUE)
R> col.nb <- read.gal(system.file("weights/columbus.gal",
+    package = "spData")[1])
R> W.col <- nb2listw(col.nb, style = "W")
```

We recode the dependent variable `CRIME` in order to obtain a binary dependent variable (`CRIMED`). The neighborhoods are classified as having "high-crime" rates if their residential burglaries and vehicle thefts are higher than 37 per thousand households, and "low-crime" rates otherwise. We choose this number in order to obtain a binary variable with a balanced number of zeros and ones.

```
R> columbus$CRIMED <- as.numeric(columbus$CRIME > 37)
```

The next step is to estimate different spatial GMM models for the `columbus` dataset. As a benchmark, we first estimate the classical probit model using the `glm` function. Then we move to the various GMM estimators presented in the previous section. In particular, we start by the linearized GMM, and then move to the one-step and two-step GMM estimators under different initial-weighting matrices — `winitial = "identity"` and `winitial = "optimal"`. One- and two-step GMM estimators are estimated using constrained optimization by setting the argument `cons.opt = TRUE` to take into account the parameter space of the spatial coefficient. In this case, λ is constrained to be in the interval $\lambda \in (\omega_{\min}^{-1}, \omega_{\max}^{-1})$, where $\omega_{\min}$ and $\omega_{\max}$ denote the smallest and largest (real) eigenvalues of $\mathbf{W}$, respectively.

	Probit	LGMM	OS-I	OS-O	TS-I	TS-O	TS-O-E
(Intercept)	3.331***	3.103***	4.705	4.252**	4.356***	4.304***	4.304***
	(0.886)	(0.952)	(5.670)	(1.764)	(1.420)	(1.405)	(1.408)
INC	−0.196***	−0.164**	−0.228	−0.216***	−0.209***	−0.207***	−0.207***
	(0.062)	(0.072)	(0.175)	(0.077)	(0.065)	(0.065)	(0.064)
HOVAL	−0.023	−0.023	−0.047	−0.040	−0.045*	−0.044*	−0.044*
	(0.015)	(0.017)	(0.098)	(0.030)	(0.026)	(0.026)	(0.026)
λ		0.746***	0.662	0.745***	0.753***	0.750***	0.750***
		(0.150)	(0.425)	(0.131)	(0.126)	(0.128)	(0.127)
N	49	49	49	49	49	49	49

Significance: $*** \equiv p < 0.01$; $** \equiv p < 0.05$; $* \equiv p < 0.1$

Table 1: SARB probit estimates for columbus example.

```
R> slm <- CRIMED ~ INC + HOVAL
R> probit <- glm(slm, family = binomial("probit"), data = columbus)
R> lgmm <- sbinaryLGMM(slm, link = "probit", listw = W.col,
+    data = columbus)
R> osI  <- sbinaryGMM(slm, link = "probit", listw = W.col,
+    data = columbus, type = "onestep", initial = "identity",
+    cons.opt = TRUE, verbose = FALSE)
R> osR  <- sbinaryGMM(slm, link = "probit", listw = W.col,
+    data = columbus, type = "onestep", winitial = "optimal",
+    cons.opt = TRUE, verbose = FALSE)
R> tsI  <- sbinaryGMM(slm, link = "probit", listw = W.col,
+    data = columbus, type = "twostep", winitial = "identity",
+    cons.opt = TRUE, verbose = FALSE)
R> tsR  <- sbinaryGMM(slm, link = "probit", listw = W.col,
+    data = columbus, type = "twostep", winitial = "optimal",
+    cons.opt = TRUE, verbose = FALSE)
R> tsR_E <- summary(tsR, vce = "efficient")
```

To have a better visualization of the results, we use the `mtable` function from **memisc** package (Elff 2023). Table 1 presents the estimates from the various models.

```
R> library("memisc")
R> table <- mtable("Probit" = probit, "LGMM" = lgmm, "OS-I" = osI,
+    "OS-O" = osR, "TS-I" = tsI, "TS-O" = tsR, "TS-O-E" = tsR_E,
+    summary.stats = c("N"),
+    signif.symbols = c("***" = .01, "**" = 0.05, "*" = 0.1))
R> table
```

The probit estimates are presented in the first column. The results show that an increase of neighborhood's income is correlated, on average, with a decrease in the propensity of having high crime. Housing value of the neighborhood, although negative, is not statistically significant. The linearized, one- and two-step GMM estimators provide similar estimates for

λ in terms of magnitude. The spatial autoregressive parameter λ is positive and it provides strong evidence of positive spatial autocorrelation on the propensity of having high-crime rates. The coefficients for `INC` and `HOVAL` are similar to the probit estimates, though slightly more negative (with the exception of the LGMM). The last column shows the estimates for $\widetilde{\boldsymbol{\theta}}_{TS,H}$ using the efficient variance-covariance matrix. As expected, the standard errors are slightly lower than the estimator that uses the robust variance-covariance matrix (Column 6).

For completeness, we also compute the average marginal effects for the last model in Table 1 using the delta method approach for the estimation of the standard errors.

```
R> me_delta <- impacts(tsR, type = "delta", vce = "efficient")
R> summary(me_delta)

------------------------------------------------------
(a) Total effects :
------------------------------------------------------
          dydx Std. error z value  Pr(> z)
INC   -0.09539    0.01748  -5.458 4.81e-08 ***
HOVAL -0.02027    0.01284  -1.578    0.114

------------------------------------------------------
(b) Direct effects :
------------------------------------------------------
           dydx Std. error z value  Pr(> z)
INC   -0.029356   0.007656  -3.834 0.000126 ***
HOVAL -0.006238   0.002767  -2.254 0.024185 *

------------------------------------------------------
(c) Indirect effects :
------------------------------------------------------
          dydx Std. error z value Pr(> z)
INC   -0.06604    0.02244  -2.943 0.00325 **
HOVAL -0.01403    0.01077  -1.302 0.19280
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results show that the variable that exerts the largest negative direct impact is `INC` (i.e., `INC` produces the largest reduction on the own-probability of having high crime). The point estimate indicates that, on average, every one thousand dollars in neighborhood's income reduces the probability of having high-crime rate of approximately 3%. Note that this average partial change also includes the effect of feedback loops. The average direct effect for `HOVAL` is also negative and significant at the 5%.

The indirect effects can be useful to identify which variable produces the largest spatial spillovers effect. In this example, negative indirect effects would represent a positive externality. That is, an increase in the housing value of spatial unit j would reduce the probability of having high-crime rates in j's neighbors. According to the results, the average indirect effect for `INC` and `HOVAL` are -0.066 and -0.014, respectively, accounting approximately for the 70% of the total effect. However, we cannot reject the null hypothesis that the indirect effect for HOVAL is zero.

Considering the variable INC, the sum of the direct and indirect effects accounts for a total negative effect of approximately 10%. In other words, an increase in income of one thousand dollars in all spatial units will generate effects that will transmit through the whole spatial system and result in a new equilibrium where the total probability of having high crime will reduce by 10%.

Finally, we use the same specification to estimate a spatial Durbin model (SDM) using the two-step GMM estimator and including the spatial lag of both INC and HOVAL:

```
R> sdm <- CRIMED ~ INC + HOVAL | INC + HOVAL
R> tsR_sdm  <- sbinaryGMM(sdm, link = "probit", listw = W.col,
+    data = columbus, type = "twostep", winitial = "optimal",
+    cons.opt = TRUE, verbose = FALSE)
R> summary(tsR_sdm, vce = "efficient")

        ------------------------------------------------------------
                      SLM Binary Model by GMM
        ------------------------------------------------------------

Call:
sbinaryGMM(formula = sdm, data = columbus, listw = W.col, link = "probit",
    winitial = "optimal", type = "twostep", cons.opt = TRUE,
    verbose = FALSE)


Coefficients:
             Estimate Std. Error z-value Pr(>|z|)
(Intercept)  9.296052   6.765445  1.3740   0.1694
INC         -0.110959   0.111052 -0.9992   0.3177
HOVAL       -0.058508   0.032221 -1.8158   0.0694 .
lag_INC     -0.470980   0.335828 -1.4024   0.1608
lag_HOVAL    0.018034   0.055994  0.3221   0.7474
lambda       0.083988   0.770141  0.1091   0.9132
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sample size: 49
```

The results for the SDM shows that none of the coefficients are statistically significant at the 5%. Hence, we test the hypothesis that the parameters for the spatially lagged variables are jointly zero using the function linearHypothesis from **car** package (Fox, Friendly, and Weisberg 2013):

```
R> library("car")
R> coefs <- names(coef(tsR_sdm))
```

```
R> linearHypothesis(tsR_sdm, coefs[grep("lag_", coefs)],
+    vcov = vcov(tsR_sdm, type = "efficient"))

Linear hypothesis test

Hypothesis:
lag_INC = 0
lag_HOVAL = 0

Model 1: restricted model
Model 2: CRIMED ~ INC + HOVAL | INC + HOVAL

Note: Coefficient covariance matrix supplied.

  Df  Chisq Pr(>Chisq)
1
2  2 2.7764     0.2495
```

The null hypothesis cannot be rejected at conventional levels.

5. Comparison with other functions

In this section, we use the simulated dataset from Section 2.3 and provide a comparison with other functions available in R in terms of estimation results and computation time.[18] We start the comparison using the functions `spprobit` (available from **McSpatial**) and `sbinaryLGMM`. Both functions allow to estimate the SARB probit model using the LGMM estimator:

```
R> library("McSpatial")
R> data$wx <- as.matrix(W) %*% x
R> sdm <- y ~ x + z + wx
R> mcS1gmm <- spprobit(sdm, data = data, wmat = as.matrix(W),
+    winst = ~ z + wx)
R> lgmm2 <- sbinaryLGMM(sdm, link = "probit", listw = W, data = data,
+    nins = 1)
```

The `spprobit` function does not allow to include spatially lagged explanatory variables as a separate `formula`. Hence, we manually create `wx` to estimate the SDM. By default, the `spprobit` function make use of the matrix of instruments $\mathbf{H} = (\mathbf{X}, \mathbf{WX})$. Thus, we need to indicate that only `z` and `wx` and their respective lags should be included as instruments. This is simply achieved by including the `winst` argument. To be consistent with `spprobit`, we set `nins = 1` in `sbinaryLGMM`.[19]

[18] We want to emphasize that in this section we are not comparing the statistical properties of different estimators, that would require a proper Monte Carlo experiment, but rather checking that the different implementations in R give similar results. The computation time is evaluated on different sample sizes but on a sigle dataset. We use the following version of each package: **spldv** 0.1.3, **McSpatial** 2.0, **ProbitSpatial** 1.1, **spatialprobit** 1.0, and R-**INLA** 23.04.02.

[19] The function `sbinaryLGMM` considers only the linearly independent columns of the matrix of instruments.

	LGMM		LGMM		OS-GMM		OS-GMM		OS-GMM		Bayes		ML		Laplace	
Cons	−0.452	0.127	−0.452	0.127	−0.448	0.120	−0.444	0.120	−0.445	0.125	−0.596	0.148	−0.455	0.013	−0.575	0.166
x	0.704	0.111	0.704	0.111	0.899	0.105	0.898	0.105	0.901	0.109	1.219	0.122	0.912	0.017	1.117	0.127
z	0.875	0.225	0.875	0.225	0.892	0.237	0.884	0.237	0.889	0.244	1.140	0.283	0.909	0.029	1.139	0.322
wx	0.782	0.365	0.782	0.365	1.075	0.279	1.067	0.280	1.048	0.305	1.319	0.309	1.035	0.037	1.001	0.299
λ	0.727	0.174	0.727	0.174	0.575	0.102	0.578	0.102	0.614	0.116	0.680	0.043	0.612	0.006	0.668	0.063

Table 2: Comparison across different functions: SARB Probit model.

Columns 1–4 in Table 2 show that our implementation and `spprobit` function provide the same coefficients and standard errors for the LGMM estimator.

Next, we compare the results for the one-step GMM estimator using the functions `gmmprobit` (available from **McSpatial**) and `sbinaryGMM`. The `gmmprobit` function uses the procedure outlined in Klier and McMillen (2008). Specifically, it minimizes the objective function given in Equation 16 and uses the variance-covariance matrix in Equation 17. An important caveat is that Klier and McMillen (2008) assumes homokedasticity, so that $\widetilde{\mathbf{S}} = \mathbf{H}^\top\widetilde{\mathbf{U}}\mathbf{H}$ where $\widetilde{\mathbf{U}}$ is a diagonal matrix with elements $\widehat{\widetilde{u}}_i^2$. The `sbinaryGMM` function allows also for this option by setting `s.matrix = "iid"`.

```
R> probit <- glm(sdm, family = binomial("probit"), data = data)
R> mcSgmm <- gmmprobit(sdm, data = data, wmat = as.matrix(W),
+    startb = coef(probit), startrho = 0, winst = ~ z + wx)
R> os_2 <- sbinaryGMM(sdm, link = "probit", listw = W, data = data,
+    type = "onestep", winitial = "optimal", s.matrix = "iid",
+    nins = 1, start = c(coef(probit), 0), reltol = 0.0001)
```

Note that we use the same starting values in both functions and we obtain similar results: the starting values for $\boldsymbol{\gamma}$ comes from a standard probit model, whereas the starting value of λ is set to 0.[20] Similarly to `spprobit`, `gmmprobit` uses $\mathbf{H} = (\mathbf{X}, \mathbf{WX})$ as instruments. As for the optimization algorithm, the default algorithm in `sbinaryGMM` is the BFGS algorithm, while `gmmprobit` uses a Gauss-Newton algorithm (that cannot be changed) with relative tolerance equals to 0.0001. The results in Columns 5–8 show that both functions provide very similar results.

In the following lines of code, we estimate the one-step GMM model with the `sbinaryGMM` function using the Taylor expansion $\mathbf{A}_\lambda^{-1} = (\mathbf{I} - \lambda\mathbf{W})^{-1} \approx \sum_{q=0}^{\infty}(\lambda\mathbf{W})^q$ setting $q = 4$. Columns 9–10 of Table 2 show that the results are close to our previous results in Section 3.2.1.

```
R> os_a <- sbinaryGMM(sdm, link = "probit", listw = W, data = data,
+    type = "onestep", winitial = "optimal", nins = 1,
+    approximation = TRUE, pw = 4, reltol = 0.0001)
```

Table 2 also includes the results of the function `sarprobit` from **spatialprobit** package and the function `ProbitSpatialFit` from **ProbitSpatial**. The `sarprobit` function provides a Bayesian estimation routine for the SARB probit model (see LeSage 2000; LeSage, Pace, Lam, Campanella, and Liu 2011), whereas the function `ProbitSpatialFit` is a conditional approximate likelihood estimation (see Martinetti and Geniaux 2017). The results are displayed in Columns 11–14. Overall, all the estimators seem to produce similar results for the spatial autoregressive parameter, λ. However, the `ProbitSpatialFit` provides the lowest standard errors across all estimators.

```
R> library("spatialprobit")
R> library("ProbitSpatial")
R> bayes <- sarprobit(sdm, W = W, data = data, showProgress = FALSE)
R> ml_cond <- ProbitSpatialFit(sdm, W = W, data = data, DGP = "SAR",
+    method = "conditional", varcov = "varcov")
```

[20]In practice this is not the optimal choice, but since the focus of this section is comparing different implementations the initial values are irrelevant as long as they are the same.

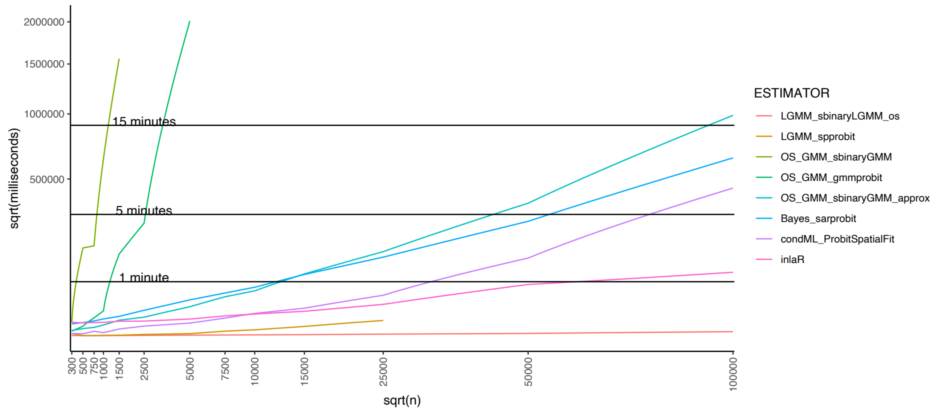

Figure 3: Comparison of computation time across different function and sample sizes: SARB Probit model.

Finally, we include the function `inla` from R-**INLA** package available at `https://www.r-inla.org/download-install`, which implements an integrated nested Laplace approximation.[21] Columns 15–16 show that the estimates from `inla` are similar to the OS-GMM, Bayes and ML estimates.

```
R> library("INLA")
R> n <- nrow(data)
R> data$idx <- 1:n
R> e <- eigen(W)$values
R> re.idx <- which(abs(Im(e)) < 1e-6)
R> rho.max <- 1 / max(Re(e[re.idx]))
R> rho.min <- 1 / min(Re(e[re.idx]))
R> mm <- model.matrix(sdm, data)
R> betaprec1 <- 0.001
R> Q.beta1 <- Diagonal(n = ncol(mm), x = 1)
R> Q.beta1 <- betaprec1 * Q.beta1
R> hyper.slm <- list(prec = list(initial = log(1), fixed = TRUE),
+    theta2 = list(prior = "logitbeta", param = c(1, 1)))
R> inlaR <- inla(y ~ -1 + f(idx, model = "slm",
+      args.slm = list(rho.min = rho.min, rho.max = rho.max, W = W,
+      X = mm, Q.beta = Q.beta1), hyper = hyper.slm),
+    data = data, family = "binomial",
+    control.family = list(link = "probit"),
+    control.compute = list(dic = TRUE, cpo = TRUE, config = TRUE))
```

Figure 3 illustrates the comparison in terms of computation time across the different functions presented above for different sample sizes. A glance at Figure 3 reveals that the two implementations of the one-step GMM (`sbinaryGMM` and `gmmprobit`) are much slower than any other functions. Interestingly, the approximated version of `sbinaryGMM` for sample sizes up

[21] For more information on the arguments of the function `inla` and how to specify them correctly we refer the reader to Gómez-Rubio *et al.* (2021) and the examples available at `https://github.com/becarioprecario/slm/blob/master/katrina-slm.R`. In this manuscript, we have also used the transformation of λ as suggested by the authors of R-**INLA**. See the replication code for further details.

to 10,000 observations takes less than a minute and it is very comparable to the Bayesian estimator in `sarprobit`. The conditional maximum likelihood approach in `ProbitSpatialFit` has a similar performance of `inla` for a sample size up to 15,000. However, for larger sample sizes `inla` performs much faster.[22] Finally, the linearized version proposed in this paper is the least computation intensive even compared with the original code in `spprobit`.

6. Conclusions

The current version of **spldv** implements most of GMM estimators available in the literature for spatial models with binary dependent variables. In particular, it allows to estimate one- and two-step GMM estimators, as well as the linearized version of the GMM procedure for Probit and Logit spatial autoregressive models. An important feature of **spldv** is that users can estimate the spatial Durbin model and compute the direct, indirect, and total effect in a friendly and flexible way.

In the paper we also give indications on the properties of the various GMM estimators and compare our implementation with other functions in R. From this comparison we conclude that there is clearly a trade-off between computation time and properties of estimators. For the benefit of the users, we try to summarize some of the findings. First of all, unless the GMM estimators become unfeasible (either because the sample size is too large or the spatial weighting matrix is too dense to even use the series expansion approximation), the linearized GMM should not be used because it presents the largest bias and the highest standard deviation. Second, the one-step estimators are similar in terms of computational time but the one using the identity matrix is outperformed by the one using optimal moment-weighting matrix. Moreover, the two-step GMM estimator are the slowest in terms of computational time but they are more efficient than the one-step. Using our simulated data, the series expansion approximation produces very similar results in the one-step estimators compared in the paper. A final indication for users that deals with large sample sizes and relatively sparse spatial weighting matrix would be to estimate their models using either of the two-step GMM estimators with the approximation.

There are of course plans to expand the package in the near future. One possible direction would be to incorporate additional (other than the spatial lag) endogenous variables. In fact, endogenous variables can be easily dealt with using GMM methods. Another direction to expand the package would be to consider panel data. Finally, the current version of the package is limited in that it considers only binary responses. An extension of the package to spatial models with multiple responses would certainly be very helpful for potential users.

Acknowledgments

We would like to express our gratitude to the three anonymous referees and the editor Roger Bivand whose comments greatly improved this paper and the package. Mauricio Sarrias thanks FONDECYT project #1230038 for full funding support.

[22]In this comparison we did not calculated the eigenvalues to set the limits for the spatial parameter in the function and simply used the interval $(-1, 1)$.

References

Angulo A, Lopez FA, Minguez R, Mur J (2022). ***spsur**: Spatial Seemingly Unrelated Regression Models.* R package version 1.0.2.5, URL https://CRAN.R-project.org/package=spsur.

Anselin L (1988). *Spatial Econometrics: Methods and Models.* Studies in Operational Regional Science. Springer-Verlag. doi:10.1007/978-94-015-7799-1.

Anselin L, Rey SJ (2014). *Modern Spatial Econometrics in Practice: A Guide to GeoDa, GeoDaSpace and PySal.* GeoDa Press LLC, Chicago.

Bates D, Maechler M, Jagan M (2023). ***Matrix**: Sparse and Dense Matrix Classes and Methods.* R package version 1.6-1, URL https://CRAN.R-project.org/package=Matrix.

Beron KJ, Vijverberg WPM (2004). "Probit in a Spatial Context: A Monte Carlo Analysis." In *Advances in Spatial Econometrics*, pp. 169–195. Springer-Verlag.

Billé AG, Arbia G (2019). "Spatial Limited Dependent Variable Models: A Review Focused on Specification, Estimation, and Health Economics Applications." *Journal of Economic Surveys*, **33**(5), 1531–1554. doi:10.1111/joes.12333.

Billé AG, Leorato S (2020). "Partial ML Estimation for Spatial Autoregressive Nonlinear Probit Model with Autoregressive Disturbances." *Econometric Reviews*, **39**(5), 437–475. doi:10.1080/07474938.2019.1682314.

Bivand R, Millo G, Piras G (2021). "A Review of Software for Spatial Econometrics in R." *Mathematics*, **9**(11). ISSN 2227-7390. doi:10.3390/math9111276.

Bivand R, Nowosad J, Lovelace R (2023). ***spData**: Datasets for Spatial Analysis.* R package version 2.3.0, URL https://CRAN.R-project.org/package=spData.

Bivand R, Pebesma E, Gomez-Rubio V (2013). *Applied Spatial Data Analysis with R.* 2nd edition. Springer-Verlag. URL https://asdar-book.org/.

Bivand R, Piras G (2015). "Comparing Implementations of Estimation Methods for Spatial Econometrics." *Journal of Statistical Software*, **63**(18), 1–36. doi:10.18637/jss.v063.i18.

Bivand R, Wong DWS (2018). "Comparing Implementations of Global and Local Indicators of Spatial Association." *TEST*, **27**(3), 716–748. doi:10.1007/s11749-018-0599-x.

Calabrese R, Elkink JA (2014). "Estimators of Binary Spatial Autoregressive Models: A Monte Carlo Study." *Journal of Regional Science*, **54**(4), 664–687. doi:10.1111/jors.12116.

Cameron AC, Trivedi PK (2005). *Microeconometrics: Methods and Applications.* Cambridge University Press.

Drukker DM, Prucha IR, Peng H, Raciborski R (2013a). "Creating and Managing Spatial-Weighting Matrices with the spmat Command." *Stata Journal*, **13**(2), 242–286. doi:10.1177/1536867x1301300202.

Drukker DM, Prucha IR, Raciborski R (2013b). "A Command for Estimating Spatial-Autoregressive Models with Spatial-Autoregressive Disturbances and Additional Endogenous Variables." *Stata Journal*, **13**(2), 287–301. doi:10.1177/1536867x1301300203.

Drukker DM, Prucha IR, Raciborski R (2013c). "Maximum Likelihood and Generalized Spatial Two-Stage Least-Squares Estimators for a Spatial-Autoregressive Model with Spatial-Autoregressive Disturbances." *Stata Journal*, **13**(2), 221–241. doi:10.1177/1536867x1301300201.

Elff M (2023). ***memisc**: Management of Survey Data and Presentation of Analysis Results.* R package version 0.99.31.6, URL https://CRAN.R-project.org/package=memisc.

Fox J, Friendly M, Weisberg S (2013). "Hypothesis Tests for Multivariate Linear Models Using the **car** Package." *The R Journal*, **5**(1), 39. doi:10.32614/rj-2013-004.

Gómez-Rubio V, Bivand R, Rue H (2021). "Estimating Spatial Econometrics Models with Integrated Nested Laplace Approximation." *Mathematics*, **9**(17), 2044. doi:10.3390/math9172044.

Kelejian HH, Prucha IR (1998). "A Generalized Spatial Two-Stage Least Squares Procedure for Estimating a Spatial Autoregressive Model with Autoregressive Disturbances." *The Journal of Real Estate Finance and Economics*, **17**(1), 99–121. doi:10.1023/a:1007707430416.

Kelejian HH, Prucha IR (1999). "A Generalized Moments Estimator for the Autoregressive Parameter in a Spatial Model." *International Economic Review*, **40**(2), 509–533. doi:10.1111/1468-2354.00027.

Kelejian HH, Prucha IR, Yuzefovich Y (2004). "Instrumental Variable Estimation of a Spatial Autoregressive Model with Autoregressive Disturbances: Large and Small Sample Results." *Advances in Econometrics: Spatial and Spatio-Temporal Econometrics*, pp. 163–198. doi:10.1016/s0731-9053(04)18005-5.

Klier T, McMillen DP (2008). "Clustering of Auto Supplier Plants in the United States: Generalized Method of Moments Spatial Logit for Large Samples." *Journal of Business & Economic Statistics*, **26**(4), 460–471. doi:10.1198/073500107000000188.

Lacombe DJ, LeSage JP (2018). "Use and Interpretation of Spatial Autoregressive Probit Models." *The Annals of Regional Science*, **60**(1), 1–24. doi:10.1007/s00168-015-0705-x.

LeSage J, Pace RK (2009). *Introduction to Spatial Econometrics*. Chapman & Hall/CRC, Boca Raton.

LeSage JP (2000). "Bayesian Estimation of Limited Dependent Variable Spatial Autoregressive Models." *Geographical Analysis*, **32**(1), 19–35. doi:10.1111/j.1538-4632.2000.tb00413.x.

LeSage JP, Pace KR, Lam N, Campanella R, Liu X (2011). "New Orleans Business Recovery in the Aftermath of Hurricane Katrina." *Journal of the Royal Statistical Society A*, **174**(4), 1007–1027. doi:10.1111/j.1467-985x.2011.00712.x.

Martinetti D, Geniaux G (2017). "Approximate Likelihood Estimation of Spatial Probit Models." *Regional Science and Urban Economics*, **64**, 30–45. doi:10.1016/j.regsciurbeco.2017.02.002.

Martinetti D, Geniaux G (2021). ***ProbitSpatial**: Probit with Spatial Dependence, SAR, SEM and SARAR Models*. R package version 1.1, URL https://CRAN.R-project.org/package=ProbitSpatial.

McMillen D (2013). ***McSpatial**: Nonparametric Spatial Data Analysis*. R package version 2.0, URL https://CRAN.R-project.org/src/contrib/Archive/McSpatial/.

Millo G, Piras G (2012). "**splm**: Spatial Panel Data Models in R." *Journal of Statistical Software*, **47**, 1–38. doi:10.18637/jss.v047.i01.

Pebesma E (2018). "Simple Features for R: Standardized Support for Spatial Vector Data." *The R Journal*, **10**(1), 439–446. doi:10.32614/rj-2018-009.

Pinkse J, Slade ME (1998). "Contracting in Space: An Application of Spatial Statistics to Discrete-Choice Models." *Journal of Econometrics*, **85**(1), 125–154. doi:10.1016/s0304-4076(97)00097-3.

Piras G (2010). "**sphet**: Spatial Models with Heteroskedastic Innovations in R." *Journal of Statistical Software*, **35**(1), 1–21. doi:10.18637/jss.v035.i01.

Piras G, Postiglione P (2022). "A Deeper Look at Impacts in Spatial Durbin Model with **sphet**." *Geographical Analysis*, **54**(3), 664–684. doi:10.1111/gean.12318.

Piras G, Sarrias M (2023). "One or Two-Step? Evaluating GMM Efficiency for Spatial Binary Probit Models." *Journal of Choice Modelling*, **48**, 100432. doi:10.1016/j.jocm.2023.100432.

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Santos LS, Proença I (2019). "The Inversion of the Spatial Lag Operator in Binary Choice Models: Fast Computation and a Closed Formula Approximation." *Regional Science and Urban Economics*, **76**, 74–102. doi:10.1016/j.regsciurbeco.2019.01.003.

Sarrias M, Piras G (2023). ***spldv**: Spatial Models for Limited Dependent Variables*. R package version 0.1.3, URL https://CRAN.R-project.org/package=spldv.

Vijverberg WP (1997). "Monte Carlo Evaluation of Multivariate Normal Probabilities." *Journal of Econometrics*, **76**, 281–307. doi:10.1016/0304-4076(95)01792-5.

Wilhelm S, de Matos MG (2013). "Estimating Spatial Probit Models in R." *The R Journal*, **5**(1), 130–143. doi:10.32614/rj-2013-013.

Affiliation:

Gianfranco Piras
Department of Economics
School of Arts and Sciences
The Catholic University of America
and
Department of Economic Studies
University "G.d'Annunzio" of Chieti – Pescara
E-mail: gpiras@mac.com
URL: https://orcid.org/0000-0003-0225-6061

Mauricio Sarrias
Facultad de Economía y Negocios (FEN)
Universidad de Talca
sn Avenida Lircay
Talca, Chile
E-mail: mauricio.sarrias@utalca.cl
URL: https://orcid.org/0000-0001-5932-4817

Journal of Statistical Software https://www.jstatsoft.org/
published by the Foundation for Open Access Statistics https://www.foastat.org/
October 2023, Volume 107, Issue 8 *Submitted:* 2022-04-06
doi:10.18637/jss.v107.i08 *Accepted:* 2023-04-15