Introduction to GeoDaSpace*

Contents

*First published: August/2012. Last revision: August/2012.

1 Introduction

GeoDaSpace is a free software to estimate spatial regressions developed by the GeoDa Center for Geostatistical Analysis and Computation[1]. The software provides a graphical user interface (GUI) to the spatial regression module of PySAL (spreg)[2] (Rey and Anselin 2007), an open source library for spatial analysis written in the object oriented language Python. The main estimation methods provided by GeoDaSpace are listed in Table 1. As shown in the table, GeoDaSpace is a tool for the estimation of models of Ordinary Least Squares (OLS), Two-Stage Least Squares (TSLS), GMM spatial lag, GMM spatial error and GMM spatial lag and error, including both the method proposed by Kelejian and Prucha (1998, 1999) (KP98/99) and Drukker et al. (2010) (KPD). All these estimation methods provide options for spatial and non-spatial diagnostics, non-spatial endogenous variables and the treatment of heteroskedasticity using HAC or White corrections, when applicable. In addition to these, GeoDaSpace offers a wide range of utilities to create and manipulate weights matrices, including contiguity, distance (bands, knn, inverse distance) and kernel.

Table 1 also shows the methods provided in GeoDaSpace that can be found in Stata and R. For this document we consider two different versions of the R package sphet. The first one, henceforth sphet1, is the released version of sphet (v. 1.1-12, published on CRAN on 2012-04-13). In addition to this version of sphet, we also use the alpha version from R-Forge, revision 56, published on 2012-07-22. This newer version of the code, which contains many additional methods and enhancements to sphet1, is henceforth referred as sphet2[3].

The remainder of this document provides a basic introduction to the use of GeoDaSpace (Section 2) . In addition to this, Section 3 presents a comparison of the estimation results using GeoDaSpace, Stata and R and explains the differences, when they occur.

[1] For information on the GeoDa Center, please check `https://geodacenter.asu.edu`

[2] More information on PySAL can be found at `http://pysal.geodacenter.org/`

[3] Given that it is an alpha version, the code is subject to change.

Table 1: Methods provided by GeoDaSpace and their availability in Stata or R

Method	**GeoDaSpace**	**Stata**	**R**[1]	**R**[2]
OLS	●	●	●	●
OLS with heteroskedasticity (White)	●	●	●	●
OLS with heteroskedasticity (HAC)	●		●	●
TSLS	●	●	●	●
TSLS with heteroskedasticity (White)	●	●	●	●
TSLS with heteroskedasticity (HAC)	●		●	●
Spatial lag	●	●	●	●
Spatial lag with het. (White)	●	●	●	●
Spatial lag with het. (HAC)	●		●	●
Spatial lag with endogenous var.	●	●		●
Spatial lag with endog. var. and het. (White)	●	●		●
Spatial lag with endog. var. and het. (HAC)	●			●
Spatial error (KP98/99)	●	●	●	●
Spatial error with endogenous var. (KP98/99)	●	●		
Spatial error and lag (KP98/99)	●	●	●	●
Spatial error and lag w/ endog. var. (KP98/99)	●	●		
Spatial error (KPD)	●	●		●
Spatial error with endogenous var. (KPD)	●	●		●
Spatial error and lag (KPD)	●	●		●
Spatial error and lag with endog. var. (KPD)	●	●		●
Spatial error with heteroskedasticity	●	●	●	●
Spatial error with endog. var. and het.	●	●		●
Spatial error and lag with het.	●	●	●	●
Spatial error and lag w/ endog. var. and het.	●	●		●

[1] Considering the packages spdep and sphet (v. 1.1-12, published on CRAN on 2012-04-13).

[2] Considering the packages spdep and sphet (revision 56, published on R-Forge on 2012-07-22).

2 Using GeoDaSpace

2.1 Opening a data file and specifying a model

GeoDaSpace is a stand-alone software that provides a graphical user interface (GUI) to the spatial regression module of PySAL (spreg). It is available in Windows and Mac OSx versions. The executable file can be downloaded for free from GeoDaCenter's website: `https://geodacenter.asu.edu/software`.

The main GUI window is shown in Figure 1. It can be divided in four sections: the menu icons (top), data and weights utilities (left), model specification (right) and model estimation (bottom).

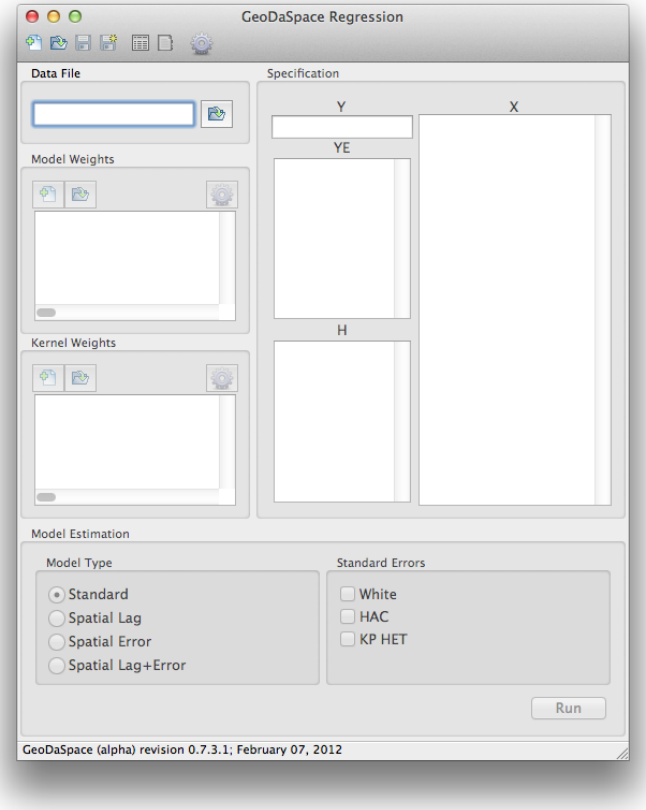


Figure 1: GeoDaSpace – Main window

The menu icons on the top of the window, from left to right, are:

- Create new model

- Open existing model
- Save model
- Save model as...
- Open the variables list
- Show results window
- Show advanced settings (see Section 2.4)

To start the specification of a new model, we need first to open the file containing the data. This can be done either by clicking on the first menu icon 'Create new model' or by clicking on the open folder icon within the data file section, as shown in Figure 2.

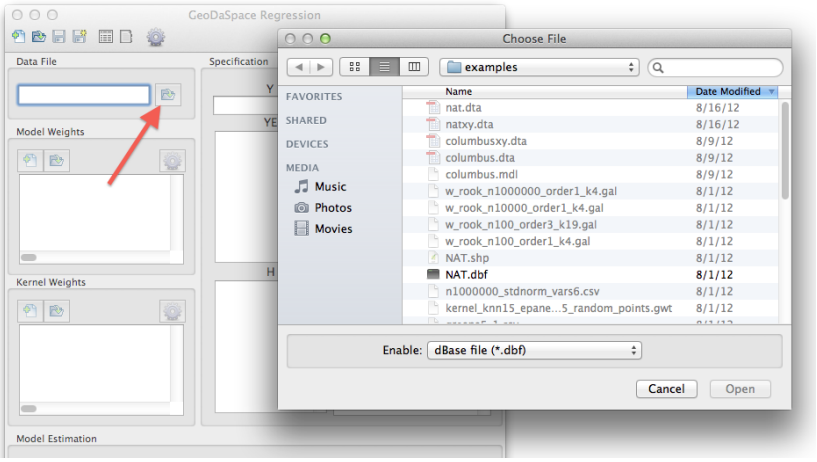


Figure 2: GeoDaSpace – Open data file

Currently, GeoDaSpace can open data files in DBF, CSV and TXT formats. For the examples in this document, we use the NAT.dbf file[4]. Once we open the data file, its variables are listed in a side window. This window can be retrieved at any time by clicking on the fifth menu icon 'Open the variable list'. To specify the model, click and drag the variables' name from the list to the respective boxes in the Specification section of the main window (Figure 3). The biggest box to the right, 'X' (required), must contain all independent variables of the model, or the right-hand side variables. The other boxes are 'Y' (required), for the dependent or left-hand side variable, 'YE' (optional), which should be used in case there are endogenous explanatory variables, and 'H' (optional),

[4]The data used in this document is available from PySAL's example data sets. It can be downloaded at `http://code.google.com/p/pysal/source/browse/#svn%2Ftrunk%2Fpysal%2Fexamples`

where the instruments for these endogenous explanatory variables should be specified.

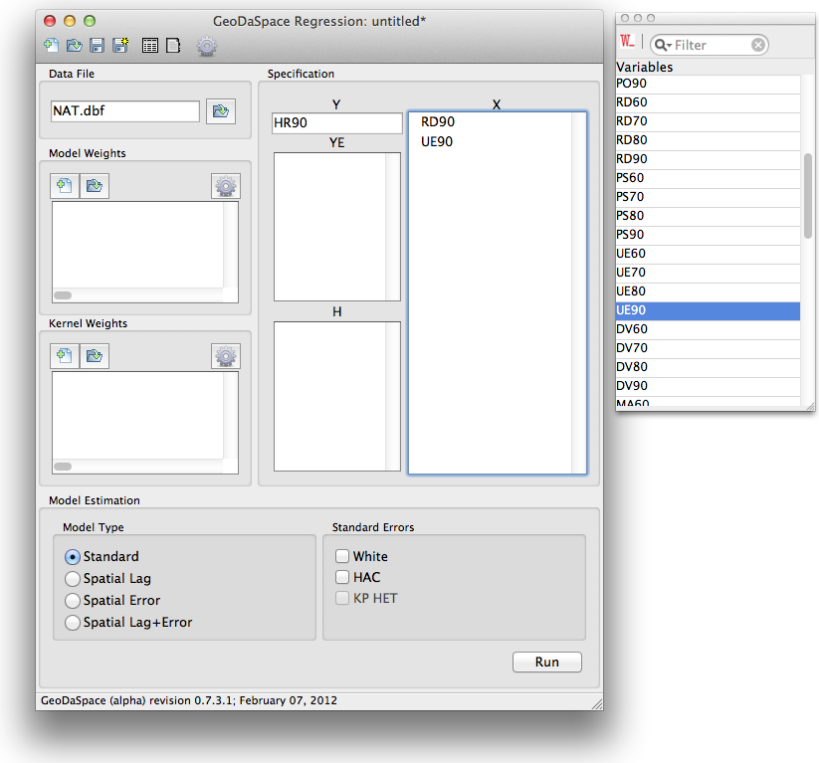


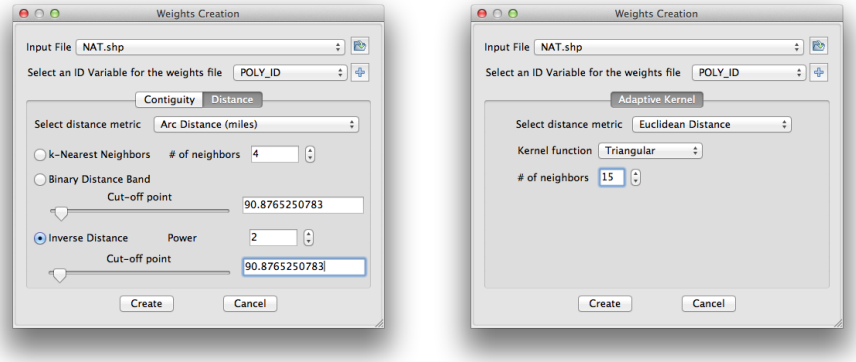
Figure 3: GeoDaSpace – Model specification

2.2 Weights creation

In the weights section, it is possible to create a new weights matrix or open an existing one. GeoDaSpace supports most of the common weights formats, including GAL, GWT and KWT (GeoDa weights), MAT (MatLab), TXT (Stata Text files), among others (clicking on the expandable menu of file type shows all options currently available). To create a new weights matrix we need to select the respective shapefile (SHP)[5] as 'Input File' (Figure 4). An ID variable for the weights matrix can either be selected from the database or added to it by clicking on the 'plus' sign. GeoDaSpace can create standard model weights or kernel weights, each type with their respective boxes in the Weights section of the main window. Within standard model weights, GeoDaSpace can create contiguity

[5]NAT.shp is available from PySAL's example data sets. It can be downloaded at `http://code.google.com/p/pysal/source/browse/#svn%2Ftrunk%2Fpysal%2Fexamples`

weights matrices of the Queen and Rook types, for any given order of contiguity, according to the spatial structure of the shapefile. Distance weights are also available (Figure 4a). Euclidean and Arc distances can be used as distance metrics. The types of distance weights available are: k-nearest neighbors, binary distance band and inverse distance. For the last two it is possible to select the cut-off point and, in the case of inverse distance, the power. If, for example, we want to create distance weights based on the inverse of the quadratic distance, all we have to do is choose 'Distance' when creating a new weights matrix, then check the 'Inverse Distance' bullet and set the power to 2 (Figure 4a). The cut-off points establish the maximum distance between neighbors. If the metric is 'Arc distance (miles)' and the cut-off is 100, no spatial unit farther than 100 miles from each other will be considered neighbors. The default cut-off point is calculated to ensure that all spatial units have at least 1 neighbor, thus preventing the creation of units with no neighbors, i.e. islands.



(a) Model weights

(b) Kernel weights

Figure 4: GeoDaSpace – Weights creation

In addition to standard model weights, kernel weights can also be created (Figure 4b). In GeoDaSpace, these are mainly used for the estimation of HAC models. Once again, Euclidean and Arc distances can be used as distance metrics. Currently, five Kernel functions are available: Uniform, Triangular, Epanechnikov, Quartic and Gaussian. It is also possible to specify the number of neighbors of each spatial unit.

The gear icon on the weights section opens a window showing the properties of the selected weights matrix. In this window, it is possible to transform the weights matrix into binary, row-standardized (each row summing 1), double-standardized (all row summing 1) and variance stabilized. The properties also include a list of the spatial units that have no neighbors, i.e. the islands, a list of the neighbors of any selected unit, the number of neighbors of each unit, i.e. the cardinalities, the unit's IDs and a histogram of units according to number of

neighbors. As an experimental feature, it is also possible to launch an interactive viewer displaying the neighborhood structure. Within the viewer, we can pass the mouse arrow over any spatial unit of the shapefile and its neighbors are instantly highlighted. Also, their IDs are listed along with their weights (Figure 5).

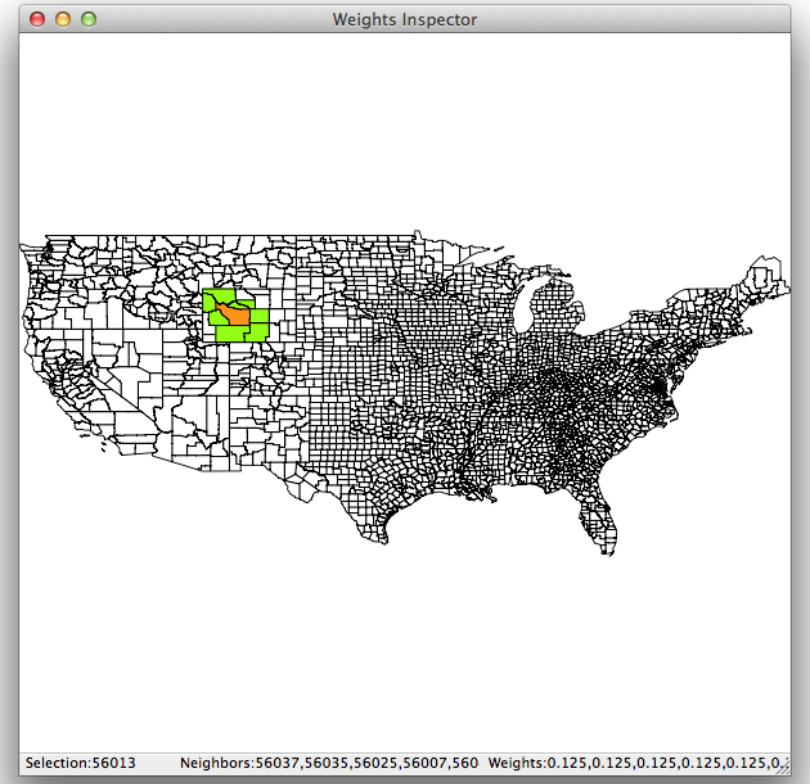


Figure 5: GeoDaSpace – Weights viewer

2.3 Model estimation

The estimation methods available in GeoDaSpace can be selected from the 'Model Estimation' section of the main window. There are four main model types: i) standard, ii) spatial lag, iii) spatial error and iv) spatial lag and error.

Standard

- Ordinary least squares (OLS)
- OLS with heteroskedasticity – White
- OLS with heteroskedasticity – HAC
- Two-stage least squares (TSLS)
- TSLS with heteroskedasticity – White
- TSLS with heteroskedasticity – HAC

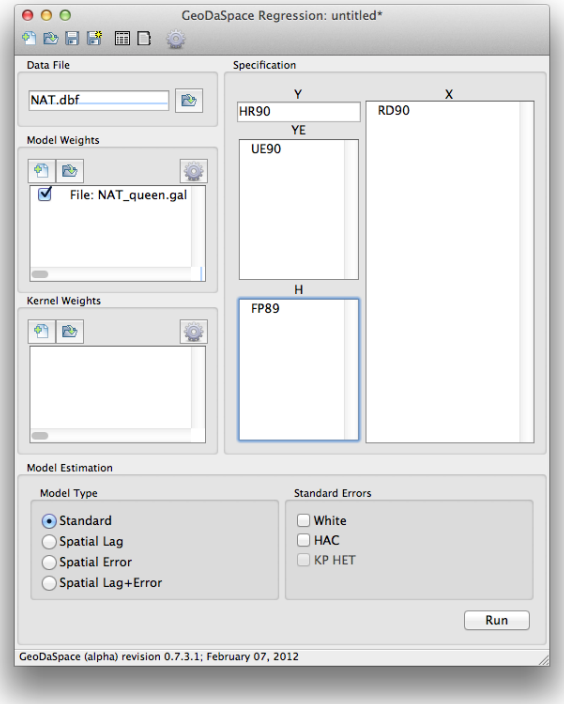


Figure 6: Estimation of a TSLS model

The standard model types are OLS and TSLS models:

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon, \tag{1}$$

in which $\mathbf{y}$ is a $n \times 1$ vector containing the dependent variable 'Y', $\mathbf{X}$ is a $n \times k$ matrix of observations on the explanatory variables 'X', β is a $k \times 1$ vector of coefficients, and ε is a $n \times 1$ vector of random errors.

These methods provide non-spatial diagnostics and spatial diagnostics, if a spatial weights matrix has been selected. In case there is no variable in the box 'YE', implying the absence of any explanatory endogenous variable, an OLS model is estimated. If instead there is any variable in the box 'YE', a TSLS is estimated. In this case, it is also required to provide in the box 'H' the instruments for the endogenous variable (Figure 6). White and HAC (requires kernel weights) corrections for heteroskedasticity are available for these methods. The 'KP HET' robust estimator assumes a spatial error structure, and therefore is not available for standard model types. Please see Section 2.4 for details on advanced settings for these methods.

Spatial lag

- Spatial lag
- Spatial lag with heteroskedasticity – White
- Spatial lag with heteroskedasticity – HAC
- Spatial lag with additional endogenous variables
- Spatial lag with additional endogenous variables and heteroskedasticity – White
- Spatial lag with additional endogenous variables and heteroskedasticity – HAC

The spatial lag type of model includes a lag of the dependent variable 'Y' on the right side of the equation:

$$\mathbf{y} = \rho\mathbf{W}\mathbf{y} + \mathbf{X}\beta + \varepsilon, \tag{2}$$

or, alternatively:

$$\mathbf{y} = (I - \rho\mathbf{W})^{-1}(\mathbf{X}\beta + \varepsilon), \tag{3}$$

in which $\mathbf{W}$ is the $n \times n$ spatial weights matrix and ρ is the spatial autoregressive scalar parameter.

As with the standard methods, White and HAC (requires kernel weights) corrections for heteroskedasticity are available for spatial lag models. Figure 7 shows an example of the estimation of a spatial lag model using HAC to correct the heteroskedasticity. The 'KP HET' robust estimator is not available for models with spatial lag and no spatial errors, since it assumes a spatial error structure. Please see Section 2.4 for details on advanced settings for these methods.

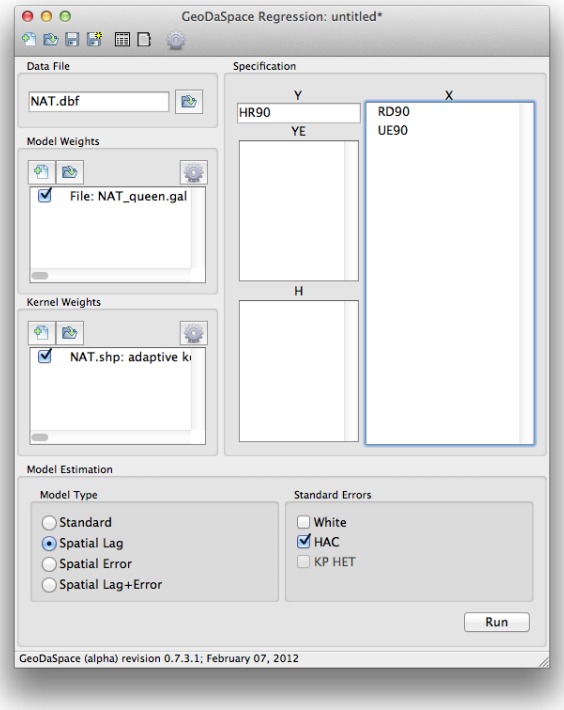


Figure 7: Estimation of a spatial lag model with heteroskedasticity – HAC

Spatial error

- Spatial error (KPD)
- Spatial error with endogenous variables (KPD)
- Spatial error (KP98/99)
- Spatial error with endogenous variables (KP98/99)
- Spatial error with heteroskedasticity
- Spatial error with endogenous variables and heteroskedasticity

The spatial error type of model estimates a spatial autoregressive parameter in the errors (λ):

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{u}, \tag{4}$$

$$\mathbf{u} = \lambda\mathbf{W}\mathbf{u} + \varepsilon, \tag{5}$$

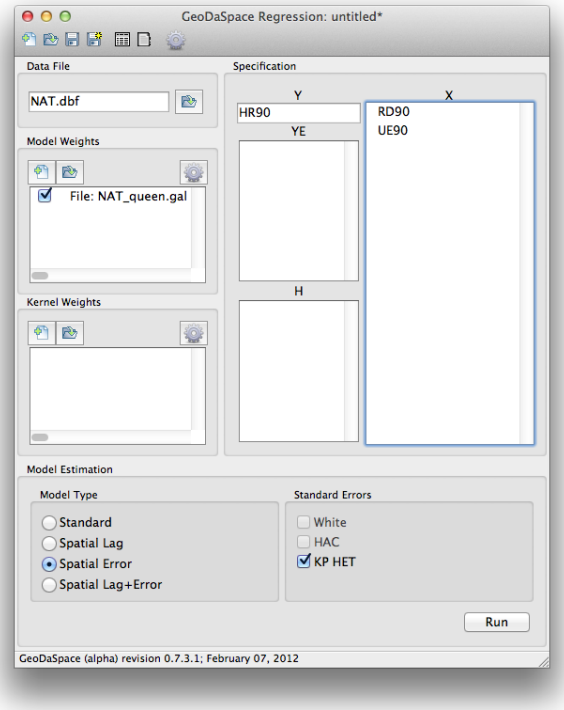


Figure 8: Estimation of spatial error models with heteroskedasticity

or, alternatively:

$$\mathbf{y} = \mathbf{X}\beta + (I - \lambda\mathbf{W})^{-1}\varepsilon. \tag{6}$$

GeoDaSpace provides two different estimators for spatial error models without heteroskedasticity. The default is the estimator proposed by Drukker et al. (2010), which we refer to as 'KPD'. The second option is the estimator proposed by Kelejian and Prucha (1998, 1999), here referred as 'KP98/99'. The choice of the estimator to be used can be changed in the Advanced Settings Panel (Section 2.4). Given the spatial structure of the error term, it is not possible to use White or HAC corrections for heteroskedasticity. Instead, GeoDaSpace can estimate the method proposed by Arraiz et al. (2010), which is robust to heteroskedasticity. To estimate this method, all we need to do is check both 'Spatial Error' model type and 'KP HET' standard errors (Figure 8) Please see Section 2.4 for details on advanced settings for these methods.

Spatial lag and error

- Spatial lag and error (KPD)

- Spatial lag and error with additional endogenous variables (KPD)
- Spatial lag and error (KP98/99)
- Spatial lag and error with additional endogenous variables (KP98/99)
- Spatial lag and error with heteroskedasticity
- Spatial lag and error with additional endogenous variables and heteroskedasticity

The spatial lag and error type of model estimates spatial autoregressive parameters both in the errors (λ) and for the spatial lag of the dependent variable (ρ):

$$\mathbf{y} = \rho\mathbf{W}\mathbf{y} + \mathbf{X}\beta + \mathbf{u}, \tag{7}$$

$$\mathbf{u} = \lambda\mathbf{W}\mathbf{u} + \varepsilon, \tag{8}$$

or, alternatively:

$$\mathbf{y} = (I - \rho\mathbf{W})^{-1}\mathbf{X}\beta + (I - \rho\mathbf{W})^{-1}(I - \lambda\mathbf{W})^{-1}\varepsilon. \tag{9}$$

All estimation methods for spatial error are also available in the presence of both spatial lag and spatial error. Once again, the default is the 'KPD', but the 'KP98/99' can be selected from the Advanced Settings Panel (Section 2.4). In the presence of heteroskedasticity, it is possible to estimate the method proposed by Arraiz et al. (2010) by checking both 'Spatial Lag+Error' model type and 'KP HET' standard errors. Figure 9 shows how to estimate a spatial error model with spatial lag and heteroskedasticity in GeoDaSpace.

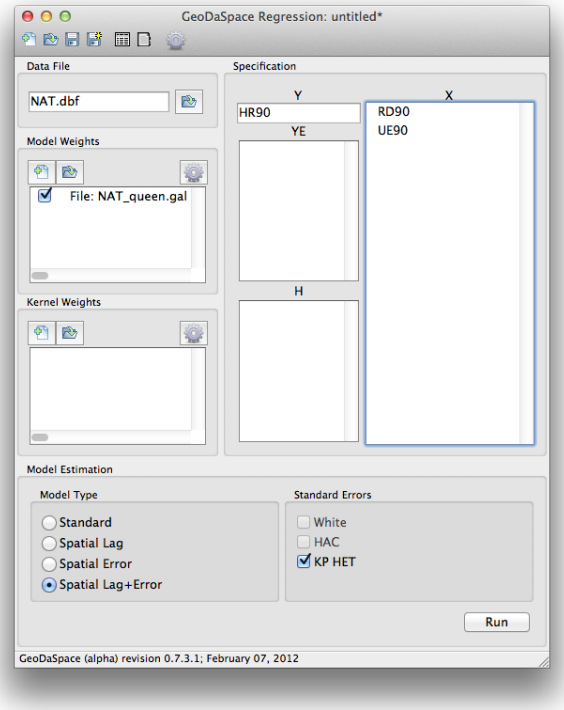


Figure 9: Estimation of spatial lag and error model with heteroskedasticity

2.4 Advanced settings

The advanced contains several preferences that affect the way the methods implemented in GeoDaSpace are estimated. To access this panel, click on the gear icon on the top menu. (Figure 10).

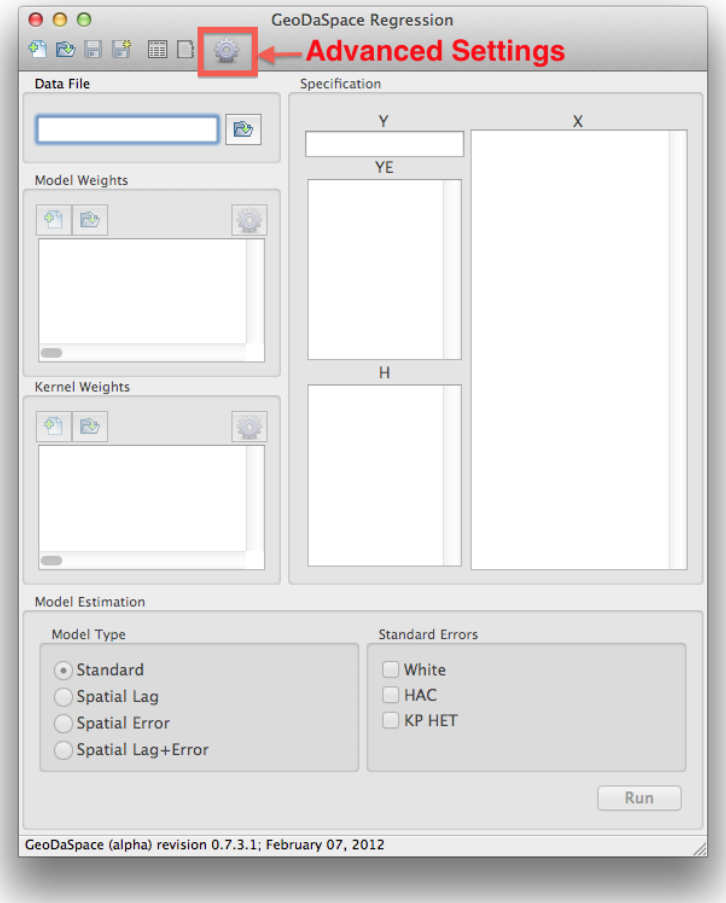


Figure 10: Advanced settings panel

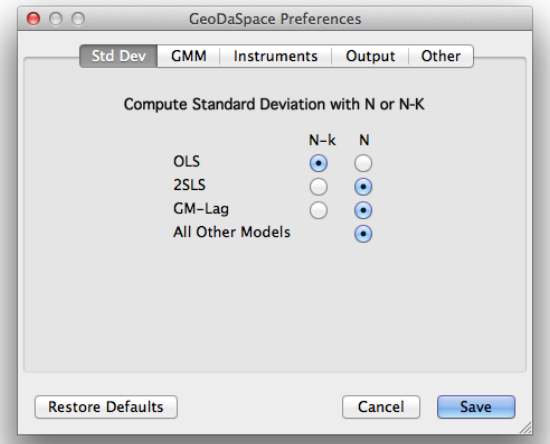


Figure 11: Advanced settings panel – Standard Deviations

The first tab of the advanced settings panel is related to the way GeoDaSpace computes the standard deviation for all methods (Figure 11). The formula for the normalization used in the calculation of the standard deviations can be selected for the OLS, TSLS, spatial lag and all other models. By default, only in the OLS case the denominator is $(N-k)$. For all others, the default is N. Of course, if N is very large, the difference between both choices is insignificant. However, the results for small samples will vary according to the methodology selected.

The second tab is related to the estimation of the GMM methods (Figure 12). Estimators such as the KPD spatial error model (Drukker et al. 2010) and spatial error with heteroskedasticity KP-HET (Arraiz et al. 2010) allow for several iterations to improve efficiency (see Anselin (2011) for details). The maximum number of iterations can be selected in this tab, as well as the convergence criterion. When any of these two criteria is achieved, the iteration process is finalized. In other words, when the difference between two subsequent estimations of λ is less or equal than the value assigned in the convergence criterion box or when the maximum number of iterations is reached, no more iterations are performed.

The second item in this tab refers to the inference on λ in spatial error or spatial lag and error models. When there is no heteroskedasticity, this box defines if the estimation method selected is the KP98/99 or the KPD. The default, when the box is checked, provides the inference on λ by estimating the KPD model. As originally proposed by Kelejian and Prucha (1998, 1999), the estimation of the KP98/99 does not provide this inference. The KP98/99 is the method used when the box is unchecked.

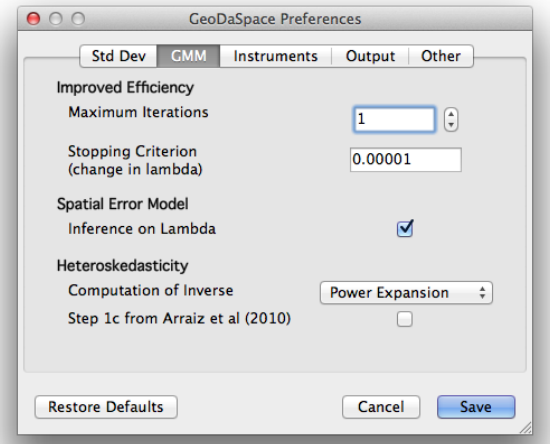


Figure 12: Advanced settings panel – GMM

The third and last item in the GMM tab is related to the estimation of spatial error models with heteroskedasticity. Here we can select the method for the computation of the inverse of the operations involving the $\mathbf{W}$ matrix. By default, a power expansion is performed in these cases, so that the computational time is decreased. If instead the true inverse is desired, this option is available from the drop-down list. In addition to this, we can also opt for having step 1c in the estimation of the spatial error model with heteroskedasticity, as proposed by Arraiz et al. (2010). Step 1c updates the initial consistent estimation of lambda using a weighted nonlinear least squares solution to the moments equations. This results in a consistent and efficient intermediate estimation of λ. Note however that a consistent estimation at this stage is already sufficient to obtain a consistent estimation of all parameters in the model. For this reason, the default in GeoDaSpace is to skip this step to save computational time.

The third tab is the Instruments tab (Figure 13). In this tab it is possible to change the way GeoDaSpace deals with instruments in case of a spatial lag or spatial lag and error models. The first item refers to the order of the spatial lags of the exogenous variables used as instruments of the spatial lag of the dependent variable. The default is to have only the first order lags of these variables, i.e. $\mathbf{WX}$, as instruments to $\mathbf{Wy}$. If instead we change the order of spatial for 2, we have that only $\mathbf{WX} + \mathbf{W^2X}$ will be used as instruments.

The second checkbox, checked by default, determines the inclusion of the spatial lag of the user-specified instruments in addition to the lag of the exogenous variables. Of course, this applies only if additional instruments are provided in the instruments box 'H' on the main GUI. The instruments in this case are $\mathbf{WX} + \mathbf{WH}$.

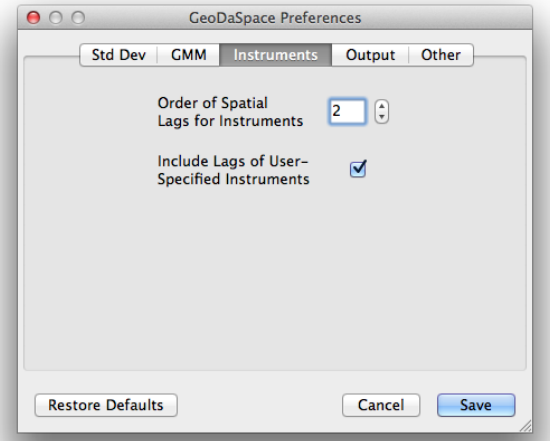


Figure 13: Advanced settings panel – Instruments

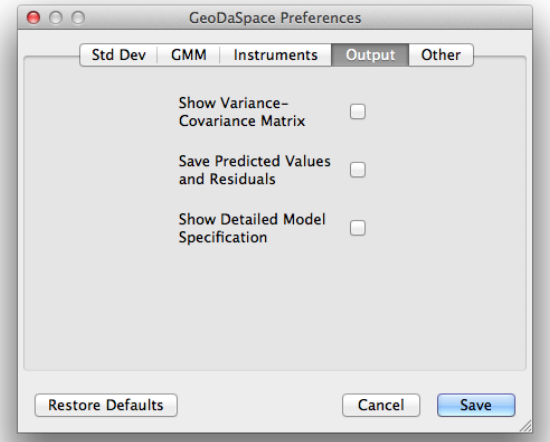


Figure 14: Advanced settings panel – Output

The fourth tab is related to the output that is printed by GeoDaSpace in the results window (Figure 14). The first checkbox controls the printing of the $k \times k$ variance-covariance matrix of the estimated parameters. When the box is checked (default), the variance-covariance matrix will be displayed below the estimation main results.

The second item allows us to save the predicted values of the dependent variable and residuals in a spreadsheet. If this option is checked, GeoDaSpace creates a CSV files containing this information. Before the estimation is performed, a pop-up window allows us to choose the folder and filename.

The last item in the Output box is "Show Detailed Model Specification". This option is currently unavailable and checking it will not affect the output.

The last tab contains other type of options. Its first item refers to the calculation of regression diagnostics. By default, non-spatial diagnostics are calculated when running an OLS model. These diagnostics include Jarque-Bera normality test, heteroskedasticity and multicollinearity diagnostics. Another item is related to the calculation of the Moran's I test for spatial dependence. Whenever a spatial matrix is selected when running standard models, LM tests will be calculated and their results will be printed on the results window. When the box for the Moran's I is check in the advanced settings panel, the Moran's I is also included. By default this option is disabled since the calculation of the Moran's I slows the computation time, especially for large samples.

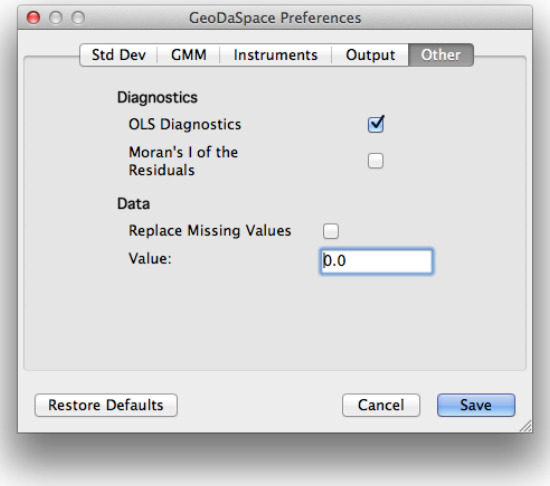


Figure 15: Advanced settings panel – Other

The second group of items in this tab refers to data manipulation. Currently GeoDaSpace cannot deal with missing values in the data. When these exist, this option allows us to replace all missing values by a given value (zero by default).

It is possible to save all options in the advanced settings panel. By clicking

on the 'Save' button, these chosen options will be saved for future sessions of GeoDaSpace, not only for the model currently being estimated. It is also possible to restore the default values at any time by clicking on the respective button and then saving these changes.

3 Comparing estimation results from GeoDaSpace, Stata and R: reasons for difference and pathway for matching

3.1 Introduction

In this section we compare the results of models with spatial error from 3 different programs: GeoDaSpace, R and Stata. Despite all the methods available in GeoDaSpace (see Section 1), here we show only those which yield discrepant results in GeoDaSpace in comparison with Stata. An explanation for the differences is provided. In addition, we show how to set the preferences in GeoDaSpace in order to get the same results from Stata. When this option is not available, we show how PySAL[6] (Rey and Anselin 2007), a Python library on which GeoDaSpace is based, can be used to match Stata.

For this document we use two different versions of the R package sphet. The first one, henceforth sphet1, is the released version of sphet (v. 1.1-12, published on CRAN on 2012-04-13). In addition to this version of sphet, we also use the alpha version from R-Forge, revision 56, published on 2012-07-22. This newer version of the code, which contains many additional methods and enhancements to sphet1, is henceforth referred as sphet2[7].

The following spatial error models (SEM) are discussed in this section:

- SEM with exogenous variables and no heteroskedasticity (KPD)
- SEM with endogenous variables and no heteroskedasticity (KPD)
- SEM with spatial lag and no heteroskedasticity (KPD)
- SEM with exogenous variables and heteroskedasticity
- SEM with endogenous variables and heteroskedasticity
- SEM with spatial lag and heteroskedasticity

3.2 Spatial Error Models without Heteroskedasticity

The default spatial error model without heteroskedasticity in GeoDaSpace is the KPD model (Drukker et al. 2010). Given the particular specification of the model when all variables as exogenous, as presented in Anselin (2011), the results from GeoDaSpace do not match those from Stata when this method is used. This discrepancy is due to an error in Stata for the case of a spatial error model with exogenous variables only. In Stata, the exogeneity is ignored and a two-stage least squares estimation is performed instead of the OLS.

Table 2 compares the results from GeoDaSpace, Stata and R. The KPD method is not available in the currently released version of sphet1 (v. 1.1-12).

[6] More information on PySAL can be found at `http://pysal.geodacenter.org/`

[7] Given that it is an alpha version, the code is subject to change.

Nonetheless, the results presented here can be obtained using sphet2, the alpha version from R-Forge. As Table 2 shows, the results from GeoDaSpace differ from those from sphet2.

Table 2: Comparison of the results of spatial error models with exogenous variables and no heteroskedasticity

Variable	**GeoDaSpace**	**sphet2**	**Stata**	**PySAL**[1]
CONSTANT	8.0259	6.6762	6.9884	6.9884
	(0.3601)	(0.3498)	(0.3605)	(0.3605)
RD90	4.3228	3.9450	3.9945	3.9945
	(0.1596)	(0.1553)	(0.1612)	(0.1612)
UE90	-0.2753	-0.0770	-0.1240	-0.1240
	(0.0479)	(0.0471)	(0.0490)	(0.0490)
lambda	0.4572	0.4149	0.4124	0.4124
	(0.0189)	(0.0194)	(0.0194)	(0.0194)

[1]PySAL using the code to match Stata as in Listing 1.

Despite the problems with Stata's estimator, PySAL allows us to match its results. To do so, we have to use PySAL's Base classes, that allows us to specify the model more freely. Since Stata performs a 2SLS estimation, in order to match its results using PySAL we have to specify X as both the endogenous variables and the instruments. Since PySAL requires at least one exogenous variable, we create a constant to use as such. In addition to the different treatment given to exogenous variables in Stata, the A1 matrix used to estimate the model is also different. In GeoDaSpace's code, the option was for the use of the matrix proposed by Arraiz et al. (2010) instead of Drukker et al. (2010) and Drukker et al. (2011). The details of this choice can be found in Anselin (2011). Listing 1 shows the command that will allow PySAL to match the results from Stata for the spatial error model[8].

[8]A walkthrough for the estimation of spatial error models without heteroskedasticity using PySAL can be found at `http://pysal.geodacenter.org/dev/library/spreg/error_sp_hom.html`

Listing 1: Using PySAL to match the results of spatial error models from Stata

```
import pysal
import numpy as np

w = pysal.open('NAT_queen.gal').read()
w.transform = 'r'
db = pysal.open('NAT.dbf')
hr90 = np.array([db.by_col('HR90')]).T
rd90 = np.array([db.by_col('RD90')]).T
ue90 = np.array([db.by_col('UE90')]).T
x = np.hstack((rd90,ue90))

ones = np.ones(crime.shape)
model = pysal.spreg.BaseGM_Endog_Error_Hom(hr90, ones,
        yend=x, q=x, w=w, A1='hom_sc')

print model.betas
print map(np.sqrt, model.vm.diagonal())
```

If we have endogenous variables, or a spatial lag, the problem in Stata's code no longer exists, since we now do have to run a 2SLS estimator. Nonetheless, the results from GeoDaSpace remain different from those from Stata and R, as shown in Table 3. The diffence is due to the choice of the A1 matrix used in the estimations and, for the spatial lag, the number of lags of the exogenous variables used as instruments.

Once again, PySAL offers the possibility of matching Stata. As shown in Listing 2, all that we have to do is to select the option 'hom_sc' for the argument A1. By doing so, we override the default A1='het', in which the matrix A1 is defined as in Arraiz et al. (2010) by opting for the A1 as used in Stata and presented in Drukker et al. (2010) and Drukker et al. (2011). For the case of a spatial lag, it is also important to set the amount of spatial lags of the exogenous variables to be used as instrument of the spatial lag of the dependent variable. The default used in GeoDaSpace is '1'. The value must be changed to '2' in order to match Stata's results. The code shown in Listing 2 continues from Listing 1.

Table 3: Comparison of the results of spatial error models with endogenous variables or spatial lag

Spatial error with UE90 as endogenous variable				
Variable	**GeoDaSpace**	**sphet2**	**Stata**	**PySAL**[1]
CONSTANT	21.0288	19.7052	21.0606	21.0606
	(1.5362)	(1.4194)	(1.5385)	(1.5385)
RD90	8.2376	8.0369	8.2420	8.2420
	(0.4881)	(0.4634)	(0.4888)	(0.4888)
UE90[2]	-2.2392	-2.0396	-2.2438	-2.2438
	(0.2286)	(0.2111)	(0.2290)	(0.2290)
lambda	0.4934	0.4856	0.4944	0.4944
	(0.0216)	(0.0220)	(0.0217)	(0.0217)
Spatial error with spatial lag				
Variable	**GeoDaSpace**[3]	**sphet2**	**Stata**	**PySAL**[1]
CONSTANT	6.9406	6.9362	6.9362	6.9362
	(0.5327)	(0.5120)	(0.5120)	(0.5120)
RD90	4.0074	4.0061	4.0061	4.0061
	(0.1758)	(0.1764)	(0.1764)	(0.1764)
UE90	-0.0957	-0.0978	-0.0978	-0.0978
	(0.0490)	(0.0481)	(0.0481)	(0.0481)
W_HR90	-0.0220	-0.0190	-0.0190	-0.0190
	(0.0543)	(0.0513)	(0.0513)	(0.0513)
lambda	0.5098	0.4364	0.4364	0.4364
	(0.0376)	(0.0421)	(0.0421)	(0.0421)

[1]PySAL using the code to match Stata as in Listing 2.

[2]UE90 instrumented by FP89 and all other exogenous variables.

[3]GeoDaSpace using 2 spatial lags for the instruments.

Note:I'm still not convinced I'm doing the endog model in R right.

I'll have to check with Gianfranco. He says he's results match Stata's.

Listing 2: Using PySAL to match the results of spatial error models with endogenous variables or spatial lag from Stata

```
#Spatial error model with spatial lag:
model = pysal.spreg.GM_Combo_Hom(hr90, x, w=w,
        A1='hom_sc', w_lags=2)
print model.summary

#Adding instrument 'FP89':
fp89 = np.array([db.by_col('FP89')]).T

#Spatial error model with UE90 as endogenous variable:
model = pysal.spreg.GM_Endog_Error_Hom(hr90, rd90,
        yend=ue90, q=fp89, w=w, A1='hom_sc')
print model.summary
```

3.3 Spatial Error Models with Heteroskedasticity

As in the case with no heteroskedasticity, Stata's code has an error in the estimation of the spatial error model with exogenous variables only. Therefore, it is not possible to match the results from Stata for this specification using GeoDaSpace. Table 4 compares GeoDaSpace's results against Stata and the package sphet from R. As already stated, we refer to the released version 1.1-12 of sphet as sphet1 and the updated alpha version of spreg available from R-Forge (revision 56 published on 2012-07-22) is referred as sphet2. Differently than sphet2, the version sphet1 does not allow us to skip one step in the estimation of the method (step1c), which is done by default in GeoDaSpace, Stata and sphet2. Please check Section 3.3.1 for more details on this.

Table 4: Comparison of the results of spatial error models with exogenous variables and heteroskedasticity

Variable	**GeoDaSpace**	**sphet1**	**sphet2**	**Stata**	**PySAL**[1]
CONSTANT	6.6586	6.5782	6.6586	6.9777	6.9777
	(0.4749)	(0.4594)	(0.4745)	(0.4622)	(0.4622)
RD90	3.9417	3.9275	3.9417	3.9911	3.9911
	(0.2602)	(0.2316)	(0.2599)	(0.2326)	(0.2325)
UE90	-0.0745	-0.0630	-0.0745	-0.1225	-0.1225
	(0.0611)	(0.0589)	(0.0611)	(0.0592)	(0.0592)
lambda	0.4753	0.4756	0.4740	0.4721	0.4721
	(0.0235)	(0.0237)	(0.0237)	(0.0236)	(0.0236)

[1]PySAL using the code to match Stata as in Listing 3.

In PySAL, it is possible to mimic the problem in Stata's code to estimate a

model that yields the same results[9]. The code is shown in Listing 3.

Listing 3: Using PySAL to match the results of spatial error models with heteroskedasticity from Stata

```
import pysal
import numpy as np

w = pysal.open('NAT_queen.gal').read()
w.transform = 'r'
db = pysal.open('NAT.dbf')
hr90 = np.array([db.by_col('HR90')]).T
rd90 = np.array([db.by_col('RD90')]).T
ue90 = np.array([db.by_col('UE90')]).T
x = np.hstack((rd90,ue90))

model = pysal.spreg.BaseGM_Endog_Error_Het(hr90, ones,
        yend=x, q=x, w=w)

print model.summary
```

When the spatial error model with heteroskedasticity contains a spatial lag, the default specification in GeoDaSpace does match the results from Stata. This is due to the order of the spatial lags of the exogenous variables used as instruments of the spatial lag of the dependent variable. The default in GeoDaSpace is a single lag. In Stata however the model is run lagging the exogenous variables twice. This option cannot be changed in Stata. In GeoDaSpace, we can choose the number of lags desired from the Preferences Panel (see Section 2.4). If we select '2' as the order of spatial lags for instruments, the results from GeoDaSpace match Stata's, as shown in Table 5.

If the spatial error model with heteroskedasticity contains other type of endogenous variables, but not a spatial lag, the results from GeoDaSpace match those from Stata without the need of any change (Table 6).

In PySAL, these models could be estimated using the code shown in Listing 4. This code is a continuation of Listing 3.

[9] A walkthrough for the estimation of spatial error models with heteroskedasticity using PySAL can be found at `http://pysal.geodacenter.org/dev/library/spreg/error_sp_het.html`

Table 5: Comparison of the results of spatial error models with spatial lag and heteroskedasticity

Variable	**GeoDaSpace**[1]	**sphet1**	**sphet2**	**Stata**	**PySAL**[2]
CONSTANT	6.9406	7.0196	6.9406	6.9406	6.9406
	(0.8600)	(0.8251)	(0.8600)	(0.8600)	(0.8600)
RD90	4.0074	4.0057	4.0074	4.0074	4.0074
	(0.3261)	(0.3212)	(0.3261)	(0.3261)	(0.3261)
UE90	-0.0957	-0.0643	-0.0957	-0.0957	-0.0957
	(0.0664)	(0.0640)	(0.0664)	(0.0664)	(0.0664)
W_HR90	-0.0220	-0.0702	-0.0220	-0.0220	-0.0220
	(0.0876)	(0.0839)	(0.0876)	(0.0876)	(0.0876)
lambda	0.5584	0.6399	0.5584	0.5584	0.5584
	(0.0507)	(0.0460)	(0.0507)	(0.0507)	(0.0507)

[1]GeoDaSpace using 2 spatial lags for the instruments.

[2]PySAL using the code to match Stata as in Listing 4.

Table 6: Comparison of the results of spatial error models with endogenous variable (UE90) and heteroskedasticity

Variable	**GeoDaSpace**	**sphet2**[1]	**Stata**	**PySAL**[2]
CONSTANT	21.0288	19.6812	21.0288	21.0288
	(2.5629)	(2.2653)	(2.5629)	(2.5629)
RD90	8.2376	8.0401	8.2376	8.2376
	(0.7817)	(0.7161)	(0.7817)	(0.7817)
UE90[3]	-2.2392	-2.0361	-2.2392	-2.2392
	(0.3902)	(0.3449)	(0.3902)	(0.3902)
lambda	0.4667	0.4614	0.4667	0.4667
	(0.0298)	(0.0295)	(0.0298)	(0.0298)

[1]sphet1 does not allow for endogenous variables.

[2]PySAL using the code to match Stata as in Listing 4.

[3]UE90 instrumented by FP89 and exogenous variables.

Note:I'm still not convinced I'm doing the endog model in R right.

I'll have to check with Gianfranco. He says he's results match Stata's.

Listing 4: Using PySAL to match the results of spatial error models with heteroskedasticity and endogenous variables or spatial lag from Stata

```
#Spatial error model with spatial lag and
#     heteroskedasticity:
model = pysal.spreg.GM_Combo_Het(hr90, x,
        w=w, w_lags=2)
print model.summary

#Adding instrument 'FP89':
fp89 = np.array([db.by_col('FP89')]).T

#Spatial error model with UE90 as endogenous variable
#     and heteroskedasticity:
model = pysal.spreg.GM_Endog_Error_Het(hr90, rd90,
        yend=ue90, q=fp89, w=w)
print model.summary
```

3.3.1 Initial Efficient Estimation of Lambda (Step1c)

In addition to the number of lags of the exogenous variables to be used as instruments, both GeoDaSpace and PySAL also offer the possibility to add the Step 1c in the estimation of the model as proposed by Arraiz et al. (2010). Step 1c updates the initial consistent estimation of lambda using a weighted nonlinear least squares solution to the moments equations. This results in a consistent and efficient intermediate estimation of lambda. Note however that a consistent estimation at this stage is already sufficient to obtain a consistent estimation of all parameters in the model. The option to run Step 1c can be found in the preferences panel in GeoDaSpace, as shown in Section 2.4. In PySAL, all we have to do to select this option is add 'step1c=True' to the arguments of the model (Listing 5).

Listing 5: Using PySAL to match the results of spatial error models with heteroskedasticity and endogenous variables or spatial lag from Stata

```
#Spatial error model with heteroskedasticity
#     (running Step1c):
model = pysal.spreg.GM_Error_Het(hr90, x,
        w=w, step1c=True)
print model.summary

#Spatial error model with spatial lag and
#     heteroskedasticity (running Step1c):
model = pysal.spreg.GM_Combo_Het(hr90, x,
        w=w, step1c=True)
print model.summary

#Spatial error model with HOVAL as endogenous variable
#     and heteroskedasticity (running Step1c):
model = pysal.spreg.GM_Endog_Error_Het(hr90, rd90,
        yend=ue90, q=fp89, w=w, step1c=True)
print model.summary
```

References

Anselin, L. (2011). GMM estimation of spatial error autocorrelation with and without heteroskedasticity. Technical report, GeoDa Center for Geospatial Analysis and Computation – Arizona State University. Available at `https://geodacenter.asu.edu/software/downloads/geodaspace`.

Arraiz, I., Drukker, D. M., Kelejian, H. H., and Prucha, I. R. (2010). A spatial Cliff-Ord-type model with heteroskedastic innovations: small and large sample results. *Journal of Regional Science*, 50:592–614.

Drukker, D. M., Egger, P., and Prucha, I. R. (2010). On two-step estimation of a spatial autoregressive model with autoregressive disturbances and endogenous regressors. *Working paper, Department of Economics, University of Maryland, College Park, MD*.

Drukker, D. M., Prucha, I. R., and Raciborski, R. (2011). A command for estimating spatial-autoregressive models with spatial-autoregressive disturbances and additional endogenous variables. *The Stata Journal*, 1:1–13.

Kelejian, H. H. and Prucha, I. R. (1998). A generalized spatial two-stage least squares procedures for estimating a spatial autoregressive model with autoregressive disturbances. *Journal of Real Estate Finance and Economics*, 17(1):99–121.

Kelejian, H. H. and Prucha, I. R. (1999). A generalized moments estimator for the autoregressive parameter in a spatial model. *International Economic Review*, 40(2):509–33.

Rey, S. and Anselin, L. (2007). PySAL, a python library of spatial analytical methods. *The Review of Regional Studies*, 37(1):5–27.