STATA SPATIAL AUTOREGRESSIVE MODELS REFERENCE MANUAL

RELEASE 16

A Stata Press Publication
StataCorp LLC
College Station, Texas

Version 16

Published by Stata Press, 4905 Lakeway Drive, College Station, Texas 77845
Typeset in TEX

ISBN-10: 1-59718-296-6
ISBN-13: 978-1-59718-296-6

The suggested citation for this software is

StataCorp. 2019. *Stata: Release 16*. Statistical Software. College Station, TX: StataCorp LLC.

Contents

Cross-referencing the documentation

When reading this manual, you will find references to other Stata manuals, for example, **[U] 27 Overview of Stata estimation commands**; [R] **regress**; and [D] **reshape**. The first example is a reference to chapter 27, *Overview of Stata estimation commands*, in the *User's Guide*; the second is a reference to the `regress` entry in the *Base Reference Manual*; and the third is a reference to the `reshape` entry in the *Data Management Reference Manual*.

All the manuals in the Stata Documentation have a shorthand notation:

[GSM]	*Getting Started with Stata for Mac*
[GSU]	*Getting Started with Stata for Unix*
[GSW]	*Getting Started with Stata for Windows*
[U]	*Stata User's Guide*
[R]	*Stata Base Reference Manual*
[BAYES]	*Stata Bayesian Analysis Reference Manual*
[CM]	*Stata Choice Models Reference Manual*
[D]	*Stata Data Management Reference Manual*
[DSGE]	*Stata Dynamic Stochastic General Equilibrium Models Reference Manual*
[ERM]	*Stata Extended Regression Models Reference Manual*
[FMM]	*Stata Finite Mixture Models Reference Manual*
[FN]	*Stata Functions Reference Manual*
[G]	*Stata Graphics Reference Manual*
[IRT]	*Stata Item Response Theory Reference Manual*
[LASSO]	*Stata Lasso Reference Manual*
[XT]	*Stata Longitudinal-Data/Panel-Data Reference Manual*
[META]	*Stata Meta-Analysis Reference Manual*
[ME]	*Stata Multilevel Mixed-Effects Reference Manual*
[MI]	*Stata Multiple-Imputation Reference Manual*
[MV]	*Stata Multivariate Statistics Reference Manual*
[PSS]	*Stata Power, Precision, and Sample-Size Reference Manual*
[P]	*Stata Programming Reference Manual*
[RPT]	*Stata Reporting Reference Manual*
[SP]	*Stata Spatial Autoregressive Models Reference Manual*
[SEM]	*Stata Structural Equation Modeling Reference Manual*
[SVY]	*Stata Survey Data Reference Manual*
[ST]	*Stata Survival Analysis Reference Manual*
[TS]	*Stata Time-Series Reference Manual*
[TE]	*Stata Treatment-Effects Reference Manual: Potential Outcomes/Counterfactual Outcomes*
[I]	*Stata Glossary and Index*
[M]	*Mata Reference Manual*

Title

Intro — Introduction to spatial data and SAR models

Description

The Sp commands manage data and fit regressions accounting for spatial relationships. Sp fits SAR models that include spatial lags of dependent and independent variables with spatial autoregressive errors on lattice and areal data, which includes nongeographic data such as social network nodes.

Different fields use different jargon for spatial concepts. SAR stands for (take your pick) spatial autoregressive or simultaneous autoregressive.

Eight short introductions will turn you into an expert on the Sp software. In these introductions, you will learn about spatial weighting matrices and how to create them as you prepare your data for analysis. You will learn about three estimation commands—`spregress`, `spivregress`, and `spxtregress`—for fitting SAR models. You will also find a worked example that includes data preparation, model fitting, and interpretation. Read the introductions first and read them sequentially.

The introductions and the commands of interest with spatial data are listed below, and each command is described in detail in its respective manual entry.

Learning the system

[SP] **Intro 1**	A brief introduction to SAR models
[SP] **Intro 2**	The $\mathbf{W}$ matrix
[SP] **Intro 3**	Preparing data for analysis
[SP] **Intro 4**	Preparing data: Data with shapefiles
[SP] **Intro 5**	Preparing data: Data containing locations (no shapefiles)
[SP] **Intro 6**	Preparing data: Data without shapefiles or locations
[SP] **Intro 7**	Example from start to finish
[SP] **Intro 8**	The Sp estimation commands

Preparing data

[D] **zipfile**	Compress and uncompress files in zip archive format
[SP] **spshape2dta**	Translate shapefile to Stata format
[SP] **spset**	Declare data to be Sp spatial data
[SP] **spbalance**	Make panel data strongly balanced
[SP] **spcompress**	Compress Stata-format shapefile

Looking at data

[SP] **grmap**	Graph choropleth maps
[SP] **spdistance**	Calculator for distance between places

Setting the spatial weighting matrix

[SP] **spmatrix**	Create, manipulate, and import/export weighting matrices
[SP] **spgenerate**	Generate spatial lag ($\mathbf{W} \times \mathbf{x}$) variables

Fitting models

[SP] **spregress**	Fit cross-sectional SAR models
[SP] **spivregress**	Fit cross-sectional SAR models with endogenous covariates
[SP] **spxtregress**	Fit panel-data SAR models

Postestimation

[SP] **estat moran**	Moran's test after regress
[SP] **spregress postestimation**	Postestimation tools for spregress
[SP] **spivregress postestimation**	Postestimation tools for spivregress
[SP] **spxtregress postestimation**	Postestimation tools for spxtregress

Glossary

[SP] **Glossary**	Jargon

Remarks and examples

The sections below provide more information about SAR models.

References for learning SAR models

Spatial models have been applied in a variety of disciplines, such as criminology, demography, economics, epidemiology, political science, and public health. Cressie (1993), Darmofal (2015), LeSage and Pace (2009), and Waller and Gotway (2004) provide textbook introductions.

Darmofal (2015, chap. 2) gives an introduction to spatial weighting matrices.

LeSage and Pace (2009, sec. 2.7) define total, direct, and indirect impacts.

Anselin (1988) gives a classic introduction to the subject.

Technical references on the development and fitting of SAR models

SAR models date back to the work of Whittle (1954) and Cliff and Ord (1973, 1981).

The GS2SLS estimator was derived by Kelejian and Prucha (1998, 1999, 2010) and extended by Arraiz et al. (2010) and Drukker, Egger, and Prucha (2013).

The formulas for the GS2SLS without higher-order spatial weighting matrices were published in Drukker, Prucha, and Raciborski (2013a). For the higher-order models, `spregress, gs2sls` implements the estimator derived in Badinger and Egger (2011) and Prucha, Drukker, and Egger (2016).

The properties of the ML estimator were proven by Lee (2004), who also provides the formulas for the robust estimator of the VCE.

Kelejian and Prucha (2010) give a technical discussion of how normalizing spatial weighting matrices affects parameter definition.

Lee and Yu (2011) give formulas and theory for SAR panel models.

Acknowledgments

We thank Ingmar Prucha of the University of Maryland for his work with us on spatial methods and econometrics that led to the methods implemented here.

We also thank Irani Arraiz of the Inter-American Development Bank, Badi Baltagi of Syracuse University, Petter Egger of ETH Zurich, and Harry Kelejian of the University of Maryland for their helpful comments and guidance.

We are grateful to Maurizio Pisati of the Università degli Studi di Milano-Bicocca for allowing us to include `grmap`, a lightly adapted version of his `spmap` command (Pisati 2007), which was preceded by his `tmap` command (Pisati 2004).

We thank Stata users for their contributions on spatial data management and spatial analysis that were published in the *Stata Journal*. We thank Belotti, Hughes, and Piano Mortari for "Spatial panel-data models using Stata". We thank Brophy, Daniels, and Musundwa for "gpsbound: A command for importing and verifying geographical information from a user-provided shapefile". We thank Neumayer and Plümper for "Making spatial analysis operational: Commands for generating spatial-effect variables in monadic and dyadic data". We thank Müller for "Stata in space: Econometric analysis of spatially explicit raster data".

StataCorp's Sp commands are based on earlier versions published in Drukker, Prucha, and Raciborski (2013a, 2013b) and Drukker et al. (2013).

References

Anselin, L. 1988. *Spatial Econometrics: Methods and Models*. New York: Springer.

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

Badinger, H., and P. H. Egger. 2011. Estimation of higher-order spatial autoregressive cross-section models with heteroscedastic disturbances. *Papers in Regional Science* 90: 213–235.

Cliff, A. D., and J. K. Ord. 1973. *Spatial Autocorrelation*. London: Pion.

——. 1981. *Spatial Processes: Models and Applications*. London: Pion.

Cressie, N. 1993. *Statistics for Spatial Data*. Rev. ed. New York: Wiley.

Darmofal, D. 2015. *Spatial Analysis for the Social Sciences*. New York: Cambridge University Press.

Drukker, D. M., P. H. Egger, and I. R. Prucha. 2013. On two-step estimation of a spatial autoregressive model with autoregressive disturbances and endogenous regressors. *Econometric Reviews* 32: 686–733.

Drukker, D. M., H. Peng, I. R. Prucha, and R. Raciborski. 2013. Creating and managing spatial-weighting matrices with the spmat command. *Stata Journal* 13: 242–286.

Drukker, D. M., I. R. Prucha, and R. Raciborski. 2013a. Maximum likelihood and generalized spatial two-stage least-squares estimators for a spatial-autoregressive model with spatial-autoregressive disturbances. *Stata Journal* 13: 221–241.

——. 2013b. A command for estimating spatial-autoregressive models with spatial-autoregressive disturbances and additional endogenous variables. *Stata Journal* 13: 287–301.

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

——. 1999. A generalized moments estimator for the autoregressive parameter in a spatial model. *International Economic Review* 40: 509–533.

——. 2010. Specification and estimation of spatial autoregressive models with autoregressive and heteroskedastic disturbances. *Journal of Econometrics* 157: 53–67.

Lee, L.-F. 2004. Asymptotic distributions of quasi-maximum likelihood estimators for spatial autoregressive models. *Econometrica* 72: 1899–1925.

Lee, L.-F., and J. Yu. 2011. Estimation of spatial panels. *Foundations and Trends in Econometrics* 4(1–2): 1–164.

LeSage, J., and R. K. Pace. 2009. *Introduction to Spatial Econometrics*. Boca Raton, FL: Chapman & Hall/CRC.

Pisati, M. 2004. Simple thematic mapping. *Stata Journal* 4: 361–378.

——. 2007. spmap: Stata module to visualize spatial data. Statistical Software Components S456812, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s456812.html.

Prucha, I. R., D. M. Drukker, and P. H. Egger. 2016. Simultaneous equations models with higher-order spatial or social network interactions. Working paper, Department of Economics, University of Maryland. http://econweb.umd.edu/~prucha/papers/WP_IRP_PHE_DMD_2016.pdf.

Waller, L. A., and C. A. Gotway. 2004. *Applied Spatial Statistics for Public Health Data*. Hoboken, NJ: Wiley.

Whittle, P. 1954. On stationary processes in the plane. *Biometrika* 434–449.

Title

Intro 1 — A brief introduction to SAR models

Description Remarks and examples Also see

Description

Sp can fit models with

1. spatial lags of dependent variables,
2. spatial lags of independent variables, and
3. spatially autoregressive errors.

The spatial features can be used in any combination.

This entry describes the above features and describes SAR models in general.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 2	The $\mathbf{W}$ matrix
Intro 3	Preparing data for analysis
Intro 4	Preparing data: Data with shapefiles
Intro 5	Preparing data: Data containing locations (no shapefiles)
Intro 6	Preparing data: Data without shapefiles or locations
Intro 7	Example from start to finish
Intro 8	The Sp estimation commands

Remarks and examples

SAR models are fit using datasets that contain observations on spatial units such as countries, districts, or even nongeographical units such as social network nodes. For simplicity, we refer to these spatial units as areas. Datasets contain at a minimum a continuous outcome variable, such as incidence of disease, output of farms, or crime rates, along with the other variables assumed to predict the chosen outcome. The dataset could be used to fit a linear regression of the form

$$y_i = \beta_0 + x_{i,1}\beta_1 + x_{i,2}\beta_2 + \cdots + x_{i,k}\beta_k + \epsilon_i$$

This linear regression is provided as a starting point; it is not a SAR model. To give this starting point a spatial feel, we will call the observations areas. The variables contain characteristics of the areas. The notation we will use is

i	area (observation), numbered 1 to N
y_i	dependent (outcome) variable in area i
$x_{i,1}$	1st independent variable in area i
	$\vdots$
$x_{i,j}$	jth independent variable in area i
	$\vdots$
$x_{i,k}$	last independent variable in area i
ϵ_i	error (residual) in area i

The linear regression model can be written in column-vector notation:

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \cdots + \beta_k\mathbf{x}_k + \boldsymbol{\epsilon}$$

The boldfaced variables are each $N \times 1$ vectors.

You could fit the above model in Stata by typing

```
regress y x1 x2 ... xk
```

SAR models extend linear regression by allowing outcomes in one area to be affected by

1. outcomes in nearby areas,
2. covariates from nearby areas, and
3. errors from nearby areas.

Said in the spatial jargon, models can contain

1. spatial lags of the outcome variable,
2. spatial lags of covariates, and
3. spatially autoregressive errors.

These terms are borrowed from the time-series literature. In time series, an autoregressive AR(1) process is

$$y_t = \gamma_0 + \gamma_1 y_{t-1} + \epsilon_t$$

where y_{t-1} is called the lag of y. In vector notation, L. is the lag operator, and the above equation could be written as

$$\mathbf{y} = \gamma_0 + \gamma_1 \mathrm{L}.\mathbf{y} + \boldsymbol{\epsilon}$$

Sometimes, AR(1) models also include autoregressive errors:

$$\mathbf{y} = \gamma_0 + \gamma_1 \mathrm{L}.\mathbf{y} + \mathbf{u}$$

where $\mathbf{u} = \rho\mathrm{L}.\mathbf{u} + \boldsymbol{\epsilon}$. In that case, the equation becomes

$$\mathbf{y} = \gamma_0 + \gamma_1 \mathrm{L}.\mathbf{y} + (\mathbf{I} - \rho\mathrm{L}.)^{-1}\boldsymbol{\epsilon}$$

The parameter ρ measures the correlation in the errors and is a parameter to be estimated along with γ_0 and γ_1.

The time-series notation and jargon can be translated to the spatial domain. The lag operator becomes an $N \times N$ matrix $\mathbf{W}$. What was L.$\mathbf{y}$ becomes $\mathbf{Wy}$, which means matrix $\mathbf{W}$ multiplied by vector $\mathbf{y}$. The SAR model corresponding to the above time-series equation is

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{Wy} + \boldsymbol{\epsilon}$$

The SAR model corresponding to the time-series equation with autoregressive errors is

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{Wy} + (\mathbf{I} - \rho\mathbf{W})^{-1}\boldsymbol{\epsilon}$$

$\mathbf{W}$ is called the spatial weighting matrix. The values in the matrix characterize the spatial relationships between areas. $\mathbf{W}$ is the spatial analog of L.$\mathbf{y}$. Whereas L.$\mathbf{y}$ measures the potential spillover from time $t-1$ to t, elements W_{i_1,i_2} specify how much potential spillover there is from area i_2 to i_1. W_{i_1,i_2} is zero if area i_2 can have no effect on i_1. The more potential spillover there is, the larger W_{i_1,i_2} is. The elements of $\mathbf{W}$ are specified before the model is fit.

In the mathematics of SAR models:

- $\mathbf{Wy}$ is the spatial equivalent of L.$\mathbf{y}$. Either way, it is the lag of the dependent variable.
- $\mathbf{Wx}_j$ is the spatial equivalent of L.$\mathbf{x}_j$. Either way, it is the lag of the jth independent variable.
- $(\mathbf{I} - \rho\mathbf{W})^{-1}\boldsymbol{\epsilon}$ is the spatial equivalent of $(1 - \rho\text{L.})^{-1}\boldsymbol{\epsilon}$. Either way, it is an autoregressive error.

Any of the above could be included in a SAR model.

Recall that the linear regression model we started with was

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \cdots + \beta_k\mathbf{x}_k + \boldsymbol{\epsilon}$$

We will keep the first two explanatory variables and drop the rest. The equation becomes

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \boldsymbol{\epsilon}$$

We could fit the shortened model by typing

```
regress y x1 x2
```

We could add $\mathbf{Wy}$ to the model:

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \beta_3\mathbf{Wy} + \boldsymbol{\epsilon}$$

We could fit this model by typing

```
spregress y x1 x2, gs2sls dvarlag(W)
```

The result would be that $\beta_3\mathbf{W}$ would measure the amount that outcomes are affected by nearby outcomes.

We could add $\mathbf{Wx}_1$ to the model:

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \beta_3\mathbf{Wy} + \beta_4\mathbf{Wx}_1 + \boldsymbol{\epsilon}$$

To fit this model, we would type

```
spregress y x1 x2, gs2sls dvarlag(W) ivarlag(W:x1)
```

The result would be that we would estimate an extra coefficient β_4 and that $\beta_4\mathbf{W}$ would measure the spillover of $\mathbf{x}_1$.

Spatial models can have more than one weighting matrix. If we had a second weighting matrix $\mathbf{V}$ in addition to $\mathbf{W}$, we could fit the model

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \beta_3\mathbf{W}\mathbf{y} + \beta_4\mathbf{W}\mathbf{x}_1 + \beta_5\mathbf{V}\mathbf{x}_1 + \boldsymbol{\epsilon}$$

by typing

```
spregress y x1 x2, gs2sls dvarlag(W) ivarlag(W:x1) ivarlag(V:x1)
```

We might do this if we were uncertain how spillover from nearby areas affects outcomes. We might be reasonably certain that there are spillover effects from adjacent areas and even from areas adjacent to adjacent areas. Let's call the adjacent areas "first-order neighbors" and the areas adjacent to adjacent areas "second-order neighbors". If we thought half the amount spilled over from second-order neighbors as from first-order neighbors, we would define $\mathbf{W}$ to constrain that by making W_{i_1,i_2} for second-order neighbors half that of first-order neighbors. If we were uncertain about the one-half assumption, we could define $\mathbf{W}$ to allow spillovers only from first-order neighbors and $\mathbf{V}$ to allow spillovers only from second-order neighbors. The spillover effect from $\mathbf{x}_1$ would be $\beta_4\mathbf{W} + \beta_5\mathbf{V}$. The ratio of second- to first-order spillovers would then be β_5/β_4.

Fitting models that estimate instead of imposing such second-order effects is asking a lot of the data. But if you have a sufficient amount of data that support this model, the approach works well.

To keep our model simple, we will remove the second-order lag so that the model reverts to

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \beta_3\mathbf{W}\mathbf{y} + \beta_4\mathbf{W}\mathbf{x}_1 + \boldsymbol{\epsilon}$$

If we added the spatial lag of $\mathbf{x}_2$ to the model, it would become

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \beta_3\mathbf{W}\mathbf{y} + \beta_4\mathbf{W}\mathbf{x}_1 + \beta_5\mathbf{W}\mathbf{x}_2 + \boldsymbol{\epsilon}$$

We could fit this model by typing

```
spregress y x1 x2, gs2sls dvarlag(W) ivarlag(W:x1 x2)
```

Whatever other lags we include in the model, we could specify autoregressive errors. The model becomes

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x}_1 + \beta_2\mathbf{x}_2 + \beta_3\mathbf{W}\mathbf{y} + \beta_4\mathbf{W}\mathbf{x}_1 + \beta_5\mathbf{W}\mathbf{x}_2 + (\mathbf{I} - \rho\mathbf{W})^{-1}\boldsymbol{\epsilon}$$

To fit this model, we would type

```
spregress y x1 x2, gs2sls dvarlag(W) ivarlag(W:x1 x2) errorlag(W)
```

The parameters to be fit in the model are β_0 though β_5 and ρ, where ρ is the correlation parameter of the residuals.

This is another model that is asking a lot of the data. Distinguishing correlated residuals from lagged dependent variables is especially tricky.

The machinery underlying `spregress` is complex. The `spregress` command with the `gs2sls` estimator uses a generalized method-of-moments estimator, which allows higher-order dependent variable lags and higher-order autoregressive error terms to be fit. `spregress` has an `ml` option for fitting a maximum likelihood estimator should you wish to fit a model under the assumption of normally distributed errors. You can read *Methods and formulas* in [SP] **spregress** for more details if you are curious.

You can fit SAR models for panel data with `spxtregress`, and you can fit SAR models with endogenous covariates using `spivregress`. These commands also incorporate spatial features like the ones described here. For details, see *Methods and formulas* in [SP] **spxtregress** and *Methods and formulas* in [SP] **spivregress**

See [SP] **Intro 8** for a brief tour of the Sp estimation commands.

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **Intro 2** — The W matrix

[SP] **Intro 7** — Example from start to finish

[SP] **Intro 8** — The Sp estimation commands

[SP] **spivregress** — Spatial autoregressive models with endogenous covariates

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spregress** — Spatial autoregressive models

[SP] **spxtregress** — Spatial autoregressive models for panel data

Title

Intro 2 — The W matrix

Description Remarks and examples Reference Also see

Description

Spatial lags and spatially autoregressive errors are defined by the spatial weighting matrix $\mathbf{W}$. This entry describes the weighting matrix.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 1	A brief introduction to SAR models
Intro 3	Preparing data for analysis
Intro 4	Preparing data: Data with shapefiles
Intro 5	Preparing data: Data containing locations (no shapefiles)
Intro 6	Preparing data: Data without shapefiles or locations
Intro 7	Example from start to finish
Intro 8	The Sp estimation commands

Remarks and examples

Remarks are presented under the following headings:

Understanding the W matrix

You will usually construct $\mathbf{W}$ on the basis of shapefiles (maps) that you obtain over the web or from other sources. It is so easy to do that you might think you can ignore the details of $\mathbf{W}$. You cannot. You need to understand $\mathbf{W}$ to interpret results from the models you fit. Moreover, those models are conditioned on $\mathbf{W}$, and the matrices you use are as much a part of your model as are the variables you include or intentionally exclude.

You use $\mathbf{W}$ in your models in three ways:

1. You include $\lambda\mathbf{W}\mathbf{y}$ to allow nearby outcomes to affect outcomes.
2. You include $\gamma\mathbf{W}\mathbf{x}$ to allow nearby covariates to affect outcomes.
3. You include autoregressive errors $(\mathbf{I}-\rho\mathbf{W})^{-1}\boldsymbol{\epsilon}$ to allow nearby errors to affect outcomes.

You can think of $\mathbf{W}$ as specifying the potential spillover as long as you realize that the actual spillovers are as follows:

1. The effect that y_i of area i has on nearby y's from the term $\lambda\mathbf{Wy}$.
2. The effect that x_i has on nearby y's, both from the term $\gamma\mathbf{Wx}$ and from the effect that x_i has on y_i, which in turn affects nearby y's.
3. The effect of including an autoregressive error.

The weighting matrix $\mathbf{W}$ is effectively a constraint placed on the individual spillovers formulated as part of the model specification.

For instance, if $W_{1,3}$ is 0, then there will be no spillover from 3 to 1 contributing to the total. It is constrained to be 0. If $W_{2,6}$ and $W_{4,7}$ are both 1, then individual spillovers from 6 to 2 and from 7 to 4 will be constrained to be equal. If $W_{5,7}$ is 2, then the spillover from 7 to 5 will be twice that of 7 to 4.

To see how this works, we will consider the matrix for four fictional places:

- Mordor, a dark land in J. R. R. Tolkien's *The Lord of the Rings*.
- Bree, a village from the same story.
- Hogsmead, a village from J. K. Rowling's *Harry Potter* novels.
- Hogwarts, a school near Hogsmead in the *Harry Potter* novels.

Spatial weighting matrices have 0s down the diagonal:

Spatial weighting $\mathbf{W}$

	Mordor	Bree	Hogsmead	Hogwarts
Mordor	0			
Bree		0		
Hogsmead			0	
Hogwarts				0

The 0s down the diagonal may surprise you. Perhaps you expected 1s. $W_{i,j}$ is the spillover from j to i, so $W_{i,i}$ is the spillover from i onto itself. Surely, geographic area i affects itself. Your thinking is correct, but you forgot that the purpose of $\mathbf{W}$ is to specify the effect of nearby areas. You will measure the effects of i on itself by adding other variables, such as $\mathbf{x}$, to your model:

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x} + \beta_2\mathbf{Wx} + \cdots$$

In this model, β_1 measures the effect of x_i on y_i, and $\beta_2\mathbf{W}$ measures the effect of $x_{i'}$ from other areas $i' \neq i$ on y_i. $\mathbf{W}$ has 0s down the diagonal so that $\mathbf{W}$ serves its intended purpose.

A $\mathbf{W}$ matrix could contain all 0s:

Spatial weighting $\mathbf{W}$

	Mordor	Bree	Hogsmead	Hogwarts
Mordor	0	0	0	0
Bree	0	0	0	0
Hogsmead	0	0	0	0
Hogwarts	0	0	0	0

If the matrix contains all 0s, there are no spatial effects. The observations are independent, and you may as well use `regress` to fit the model.

You use Sp estimation commands when some elements of $\mathbf{W}$ are nonzero. Zeros are nonetheless a reasonable value for many of the elements. For instance, Mordor and Bree are from one set of novels, while Hogsmead and Hogwarts are from another. It would be reasonable to assume (to constrain) that there are no spillover effects between them. We would have the following matrix:

Spatial weighting $\mathbf{W}$

	Mordor	Bree	Hogsmead	Hogwarts
Mordor	0	?	0	0
Bree	?	0	0	0
Hogsmead	0	0	0	?
Hogwarts	0	0	?	0

In the above matrix, we are specifying that Mordor and Bree are independent of Hogsmead and Hogwarts, and vice versa. The question marks stand in for the values left to be filled in, which are

- $W_{1,2}$, the potential spillover of Bree on Mordor.
- $W_{2,1}$, the potential spillover of Mordor on Bree.
- $W_{3,4}$, the potential spillover of Hogwarts on Hogsmead.
- $W_{4,3}$, the potential spillover of Hogsmead on Hogwarts.

Nonzero values in $\mathbf{W}$ must be positive. The larger the value in W_{i_1,i_2}, the more the potential spillover.

How shall we measure spillover? It turns out not to matter so long as we are consistent. Said differently, only ratios of elements in the matrix matter. Remember how spatial lags are used:

$$\mathbf{y} = \beta_0 + \beta_1\mathbf{x} + \beta_2\mathbf{W}\mathbf{x} + \cdots$$

Fitted coefficient β_2 measures the effect of the spatial lag. If we replaced $\mathbf{W}$ with $2\mathbf{W}$, the result would be to halve β_2, just as β_1 would halve if we doubled $\mathbf{x}$.

We will set $W_{3,4}$, the potential spillover of Hogwarts on Hogsmead, to 1, and in setting this first nonzero value, we have decided on the units. The units are Hogwarts on Hogsmead. If we set an element to 2, then we are setting the potential spillover to be twice that of Hogwarts on Hogsmead. If we set an element to $1/2$, then we are setting the potential spillover to be half that of Hogwarts on Hogsmead.

If we also set $W_{4,3} = 1$, we will be constraining the potential spillover of Hogsmead on Hogwarts to be the same as Hogwarts on Hogsmead. Our matrix would be

Spatial weighting $\mathbf{W}$

	Mordor	Bree	Hogsmead	Hogwarts
Mordor	0	?	0	0
Bree	?	0	0	0
Hogsmead	0	0	0	1
Hogwarts	0	0	1	0

What should we make the spillovers of Bree on Mordor and of Mordor on Bree? In the *Lord of the Rings* story, Mordor is far from Bree, larger than Bree, and actively exporting evil at the speed of magic. Bree, meanwhile, is a speck that Mordor could brush away with little effort.

We will set $W_{2,1}$, the spillover of Mordor on Bree, to 4 and $W_{1,2}$, the spillover of Bree on Mordor, to 0.1. The spillover of Mordor on Bree will be four times that of Hogwarts on Hogsmead. Meanwhile, the spillover of Bree on Mordor will be one-tenth that of Hogwarts on Hogsmead.

You might well question the numbers we have chosen. Why is the spillover of Mordor on Bree 4 and not 5? Or 10? We have no satisfactory answer, and that is why in real problems researchers often set potential spillovers to 1 for adjacent areas and to 0 elsewhere, or set potential spillovers to the inverse of the distance between the locations. Both seem more defensible, although defending them can be problematic. Do second-order neighbors really have no effect? Or in the case of inverse distance, why not inverse distance squared? Sometimes theory can provide an answer. The spillover of a light bulb is inverse distance squared. In other cases, there are no satisfactory answers except that making partially justified assumptions and accounting for spillover effects is preferable to assuming that spillover effects are all 0.

So we have

Spatial weighting $\mathbf{W}$

	Mordor	Bree	Hogsmead	Hogwarts
Mordor	0	0.1	0	0
Bree	4	0	0	0
Hogsmead	0	0	0	1
Hogwarts	0	0	1	0

And the final $\mathbf{W}$ matrix is

$$\mathbf{W} = \begin{bmatrix} 0 & 0.1 & 0 & 0 \\ 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

We could enter this matrix into Sp and proceed to estimation. It would be easy enough to do; see [SP] **spmatrix userdefined**, [SP] **spmatrix fromdata**, and [SP] **spmatrix spfrommata**.

Although we could do this, in reality you will not. You will have more than four geographical units—you might have 3,000 counties. To say nothing of the misery of entering a 3000×3000 matrix, you are not going to carefully consider and research all 8,997,000 pairs of counties. You are going to assume that only adjacent counties affect each other—called "contiguity" in the literature—or that spillover effects are proportional to the inverse of distance between counties. You are going to do that because you can create such $\mathbf{W}$ matrices by typing a single command such as

```
. spmatrix create contiguity Wc          // contiguity
. spmatrix create idistance  Widist      // inverse distance
```

We told you that the units in which the weights are measured do not matter, but that is not exactly true. They do not matter if you only include spatially lagged covariates. If, however, you use the spatial weighting matrix to lag the dependent variable ($\lambda\mathbf{Wy}$) or for autoregressive errors [$(\mathbf{I} - \rho\mathbf{W})^{-1}\boldsymbol{\epsilon}$], then $\widehat{\lambda}$ and $\widehat{\rho}$ will be easier to interpret if the matrix is scaled appropriately. In that case, $\widehat{\lambda}$ and $\widehat{\rho}$ should be between -1 and 1 unless the solution is explosive.

Explosive solutions can arise in spatial analysis for the same reasons they arise in time-series analysis. If A affects B and B affects A, and if the coefficients are large enough, then feedback becomes amplified. A sends a large value to B, which B receives, amplifies, and sends back to A, whereupon the procedure repeats, and eventually, the process explodes in a mess of infinities.

To fit models, the Sp software virtually requires that you scale the matrices used to produce lags of the dependent variable or autoregressive errors, and you ought to scale the other matrices too. The software produces more accurate results when inputs are scaled.

Scaling is so important that when you type the commands

```
. spmatrix create contiguity Wc
. spmatrix create idistance Widist
```

scaling is performed automatically, and you have to go out of your way to prevent it, which you can do by typing

```
. spmatrix create contiguity Wc, normalize(none)
. spmatrix create idistance Widist, normalize(none)
```

By default, weighting matrices are scaled so that their largest eigenvalue is 1. See [SP] **spmatrix create** and *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

Missing values, dropped observations, and the W matrix

Missing values sometimes appear in data. When fitting models with such data, the usual solution is to omit the observations from the estimation sample. That can be justified when observations are independent, but observations are not independent in SAR models.

Spatial models allow for spillover effects from nearby areas. In spatial data, observations are areas. Omitting some areas means that the spillovers from them are no longer being included in the fitted model. Consider two adjacent counties and assume that one of them is dropped from the estimation. Then, the spillover from the dropped county to its neighbor—a neighbor still in the data—becomes 0 even though there really is spillover. It is just unobserved spillover.

Thus, Sp estimation commands handle missing observations differently from Stata's other estimation commands. If an area is defined in a spatial weighting matrix and that area is not observed in the data, Sp refuses to fit the model unless you specify option `force`.

Imagine that you type

```
. spregress y x, gs2sls ivarlag(W: x)
```

and that observation 4 contains a missing value for `x`. Most Stata estimation commands would omit the observation from the estimation sample and proceed with estimation. `spregress` will issue an error and mention the `force` option.

```
. spregress y x, gs2sls ivarlag(W: x)
  (1412 observations)
  (1 observation excluded due to missing values)
  (1411 observations (places) used)
  (weighting matrix defines 1412 places)
weighting matrix defines places not in estimation sample
    Excluding observations excludes the spillovers from those observations to
    other observations which are not excluded. You must determine whether this
    is appropriate in this case and, if it is, specify option force.
r(459);
```

You would be on firm theoretical ground to specify the `force` option if the fourth column of `W` contained only 0 values, because in that case there are no spillovers from observation 4 to the other areas. Meanwhile, the fourth row does not have to be all 0s. The other areas might spillover to observation 4, but that will not bias results if observation 4 is omitted. It is the unobserved spillovers from observation 4 that cause bias.

You would be on muddy theoretical ground—which most applied researchers consider to be firm enough—if the fourth column of `W` contained only small values.

You would be sinking in a swamp if the fourth column of `W` contained any large values. We at StataCorp might go there, but if we did, we would afterward try replacing `x[4]` with various reasonable values to determine how sensitive our forced results would be to the missing observation.

Reference

Liu, D. 2017. How to create animated graphics to illustrate spatial spillover effects. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2018/03/06/how-to-create-animated-graphics-to-illustrate-spatial-spillover-effects/.

Also see

[SP] **Intro 3** — Preparing data for analysis

[SP] **Intro 7** — Example from start to finish

[SP] **spmatrix** — Categorical guide to the spmatrix command

Title

Intro 3 — Preparing data for analysis

Description Remarks and examples Also see

Description

Before you can use the Sp estimation commands—`spregress`, `spivregress`, and `spxtregress`—to fit SAR models, you need to prepare your data. This entry describes the various types of Sp data and their characteristics.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 1	A brief introduction to SAR models
Intro 2	The $\mathbf{W}$ matrix
Intro 4	Preparing data: Data with shapefiles
Intro 5	Preparing data: Data containing locations (no shapefiles)
Intro 6	Preparing data: Data without shapefiles or locations
Intro 7	Example from start to finish
Intro 8	The Sp estimation commands

Remarks and examples

Remarks are presented under the following headings:

Three types of Sp data

The Sp commands categorize the data that you use as being

- data with shapefiles,
- data without shapefiles but including location information, or
- data without shapefiles or location information.

Shapefiles are maps and are easily found on the web. One way that the Sp commands use shapefiles is to obtain (x, y) coordinates of places, which makes creating $\mathbf{W}$ matrices easy.

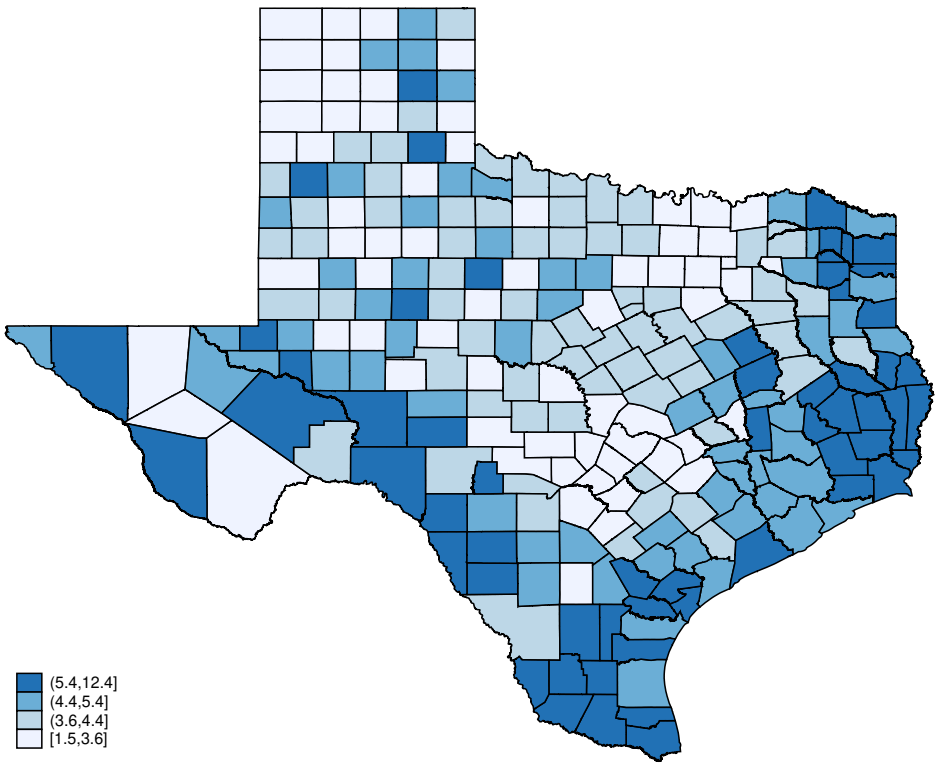
Alternatively, your data could contain the (x, y) coordinates, and then you do not need shapefiles. However, you still might want them because then you can draw choropleth maps.

Finally, your data might not contain (x, y) coordinates at all. Your data might not be geographic. Whether your data are geographic or a social network, it is the $\mathbf{W}$ matrix that defines the "spatial" relationships.

Type 1: Data with shapefiles

The first and best approach with geographic data is to use shapefiles. Shapefiles are easily found and downloaded from the web. Shapefiles make setting the **W** matrix easy, and you can draw choropleth maps such as

```
. grmap unemployment
```



Type 2: Data without shapefiles but including location information

If your data already contain the locations of the observations, you do not need shapefiles. You can proceed almost directly to analysis.

Setting the spatial weighting matrix is almost as easy as it is when you have a shapefile. You lose the easy construction of contiguity matrices—matrices in which only adjacent areas spill over to one another—but you can still set **W** on the basis of distance.

Without shapefiles, you lose the ability to draw choropleth maps.

Type 3: Data without shapefiles or location information

When you do not have location information, you must construct and enter the spatial weighting matrix **W** manually, just as we did in [SP] **Intro 2** with Mordor, Bree, Hogsmead, and Hogwarts.

SAR models can be fit to data that are not spatial, such as social networks. The elements of $\mathbf{W}$ record spillover from j to i, whether that is place j to i, imaginary universe j to i, or network node j to i. In the case of networks, you may already have a $\mathbf{W}$ from an official source. You can use `spmatrix import` to import it; see [SP] **spmatrix import**.

If your data are spatial, on the other hand, we strongly suggest finding a shapefile on the web or finding and entering each observation's location.

Sp can be used with cross-sectional data or panel data

Whether the data contain shapefiles, locations, or neither is one aspect of Sp data. The other is whether the data are cross-sectional or panel.

Cross-sectional data contain one observation per geographical unit, such as country, state, county, or zip code. A cross-sectional dataset might look like this:

area_id	area_name	v1	v2	...
1	Brazos	...		
2	Travis	...		
3	Grimes	...		

`v1`, `v2`, `...` contain values for each area.

Panel data contain multiple observations per geographical unit. Panel data look like this:

area_id	area_name	year	v1	v2	...
1	Brazos	1990	...		
1	Brazos	2000	...		
1	Brazos	2014	...		
2	Travis	1990	...		
2	Travis	2000	...		
2	Travis	2014	...		
3	Grimes	1990	...		
3	Grimes	2000	...		
3	Grimes	2014	...		

`v1`, `v2`, `...` contain values by year for each area.

Detailed instructions for preparing cross-sectional and panel data will be provided in [SP] **Intro 4**, [SP] **Intro 5**, and [SP] **Intro 6**. First, we need to tell you about the numeric ID variables that Sp will need.

ID variables for cross-sectional data

Sp requires that cross-sectional data contain an ID variable that uniquely identifies the observations. Both `area_id` and `area_name` do that in the following data:

area_id	area_name	v1	v2	...
1	Brazos	...		
2	Travis	...		
3	Grimes	...		

`v1`, `v2`, `...` contain average values within area.

Because Sp requires that the ID variable be numeric, `area_id` would be our ID variable. `area_id` contains $1, 2, \ldots$, but that is not required. Another dataset might contain U.S. Census FIPS county codes:

fips	area_name	v1	v2	...
48041	Brazos	...		
48453	Travis	...		
48185	Grimes	...		

The ID variable then would be `fips`.

If the data do not contain a numeric ID but do contain a string ID variable, such as `area_name`, you can create a numeric ID from it by typing

```
. sort area_name
. generate id = _n
```

We sorted by `area_name` to align the names and code, but that is not necessary. If you had no identification variable whatsoever, you could type

```
. generate id = _n
```

ID variables for panel data

We have a lot more to say about ID variables in panel data, and there are substantive issues as well. To remind you, panel data look like this:

area_id	area_name	year	v1	v2	...
1	Brazos	1990	...		
1	Brazos	2000	...		
1	Brazos	2014	...		
2	Travis	1990	...		
2	Travis	2000	...		
2	Travis	2014	...		
3	Grimes	1990	...		
3	Grimes	2000	...		
3	Grimes	2014	...		

`v1`, `v2`, `...` contain average values within year for each area.

Panel data have two identifiers. Generically, they are called the first- and second-level IDs. In these data, those IDs are

First-level ID	Second-level ID
area_id	year

`area_name` could be the first-level ID, but because Sp requires that ID variables be numeric, we put `area_id` in the table. If the data contained `area_name` but not `area_id`, we would create `area_id` by typing

```
. egen area_id = group(area_name)
```

The first-level ID corresponds to the ID in cross-sectional data. As with cross-sectional data, that first-level ID is not required to contain 1, 2, ... It could contain FIPS codes or whatever else.

Sp assumes that the first-level ID corresponds to area.

Sp assumes that the second-level ID corresponds to time.

Concerning the second-level variable, we call it time because it usually is time. The spatial fixed- and random-effects estimators that Sp provides are appropriate for use with panels over time. The estimators are appropriate in other cases, too, but not all other cases. Whether they are appropriate hinges on whether spatial lags have a meaningful interpretation.

Sp defines panel-data spatial lags as being across area at the same time or, equivalently, across first-level ID for the same values of the second-level ID:

Meaning of spatial lag, observation by observation

1st-level ID `area_id`	2nd-level ID year	Spatial lag means `area_id` for year	
1	1990	*	1990
1	2000	*	2000
1	2014	*	2014
2	1990	*	1990
2	2000	*	2000
2	2014	*	2014
3	1990	*	1990
3	2000	*	2000
3	2014	*	2014

When the second-level identifier is time, defining spillovers as coming from nearby areas at the same time is just what you want. It is sometimes what you want when the second-level identifier is not time, too.

On the other hand, here is an example in which the second-level identifier is not time and the data are not appropriate for use with Sp. We have data on school districts in counties:

`area_id`	`area_name`	`district`	`district_name`	`v1`	`v2`	...
1	Brazos	1	BISD	...		
1	Brazos	2	CSISD	...		
1	Brazos	3	NISD	...		
2	Travis	1	AISD	...		
2	Travis	2	HISD	...		
2	Travis	3	RRISD	...		
3	Grimes	1	ASISD	...		
3	Grimes	2	MISD	...		
3	Grimes	3	RISD	...		

`v1`, `v2`, ... contain average values within district for each area.

Spatial lags would be meaningless with these data because they would be calculated across area for equal values of `district`. Independent school districts run schools in subareas of counties. Those independent school districts have names like BISD and CSISD. ISD stands for Independent School

District. BISD stands for Bryan ISD, and CSISD stands for College Station ISD. Bryan and College Station are two different areas of Brazos County.

Let's consider the meaning of a spatial lag for the first observation in the data. It would be calculated across area for `district` = 1. Across area is just what we want, but matching BISD with AISD with ASISD is senseless.

Data on county–school type, however, could be meaningfully matched:

area_id	area_name	type	meaning	v1	v2	...
1	Brazos	1	elementary	...		
1	Brazos	2	middle	...		
1	Brazos	3	high school	...		
2	Travis	1	elementary	...		
2	Travis	2	middle	...		
2	Travis	3	high school	...		
3	Grimes	1	elementary	...		
3	Grimes	2	middle	...		
3	Grimes	3	high school	...		

`v1`, `v2`, `...` contain average values within type of school for each area.

In these data, a spatial lag would be nearby counties for schools of the same type.

In the rest of this manual, we will write as if all panel datasets are location–time datasets, but remember that time is not required to be time. If it is not time, however, you must ensure that the spatial comparisons are reasonable.

Because of the matching required in calculating spatial lags, Sp's fixed- and random-effects estimators require that the data be strongly balanced. Strongly balanced means that each panel has the same number of observations and that the panels record data for the same set of times. Later, we will tell you about the `spbalance` command. It will balance the data for you by dropping observations for times not defined in all panels. See [SP] **spbalance**.

Also see

[SP] **Intro 7** — Example from start to finish

[SP] **spbalance** — Make panel data strongly balanced

[SP] **spset** — Declare data to be Sp spatial data

Title

Intro 4 — Preparing data: Data with shapefiles

Description Remarks and examples Also see

Description

To perform a spatial analysis, you do the following steps:

1. Prepare data for use by Sp.
2. Define weighting matrices.
3. Fit models using `spregress`, `spivregress`, or `spxtregress`.

Step-by-step instructions for step 1 are provided below. These instructions are for preparing data with shapefiles.

Shapefiles define maps. You obtain them over the web. After translation into Sp format, you merge the translated result with a `.dta` file or files you already have.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 1	A brief introduction to SAR models
Intro 2	The **W** matrix
Intro 3	Preparing data for analysis
Intro 5	Preparing data: Data containing locations (no shapefiles)
Intro 6	Preparing data: Data without shapefiles or locations
Intro 7	Example from start to finish
Intro 8	The Sp estimation commands

Remarks and examples

Remarks are presented under the following headings:

Overview

Shapefile is jargon for computer files that store a map. A shapefile might store the map for the 3,000-plus counties in the United States.

Shapefiles are optional. If you have one, Sp can determine which places (counties) are neighbors (share a border), and Sp will know the distances between the centroids of the places. You will be able to create spatial weighting matrices of first-order neighbors by typing

```
. spmatrix create contiguity Wc
```

and to create inverse-distance weighting matrices by typing

```
. spmatrix create idistance Wd
```

and to graph choropleth maps by typing

```
. grmap ue_rate
```

You find and download shapefiles on the web, and translate them to Stata format. For example,

1. You find and download `tl_2016_us_county.zip` for U.S. counties.
2. You convert `tl_2016_us_county.zip` to Stata format, creating two new datasets: `tl_2016_us_county.dta` and `tl_2016_us_county_shp.dta`.

For information on how to find `tl_2016_us_county.zip` on the web, see *Finding a shapefile for Texas counties* in [SP] **Intro 7**. You can download this file if you want to follow along with the commands in this introduction.

Let's suppose you have downloaded the U.S. counties file and unzipped it. You also have two county-data datasets, `project_cs.dta` and `project_panel.dta`, containing observations on the 3,000-plus counties. These datasets are available by typing

```
. copy https://www.stata-press.com/data/r16/project_cs.dta .
. copy https://www.stata-press.com/data/r16/project_panel.dta .
```

They are standard Stata datasets. You could use them with `regress` or, in the case of `project_panel.dta`, which contains panel data, `xtreg`, but the datasets are not yet suitable for use with `spregress` or `spxtregress`.

To make the project datasets work with Sp, you merge each one with the Stata-format shapefiles. We will show you how to create these shape files in *Creating Stata-format shapefiles*. Merging your data with a shapefile will add three variables to your data: `_ID`, `_CX`, and `_CY`.

`project_cs.dta` is a cross-sectional dataset, so when the shapefile is prepared, you could type

```
. use project_cs, clear
. merge 1:1 fips using tl_2016_us_county
. keep if merge==3
. drop _merge
```

If all goes well, no observations from `project_cs.dta` will be dropped. You keep the matches because there are sometimes observations in the shapefile that are not in `project_cs.dta`.

`project_panel.dta` is a panel dataset, so you could type

```
. use project_panel, clear
. xtset fips time
. spbalance
. merge m:1 fips using tl_2016_us_county
. keep if _merge==3
. drop _merge
```

Merging panel data requires extra steps because 1) the data must be `xtset` and 2) Sp requires that the panels be strongly balanced. This was discussed in [SP] **Intro 3**.

How to find and download shapefiles on the web

Shapefiles contain more than a map. They sometimes contain data relevant to specific research problems. You can find shapefiles that contain climate or economic or epidemiological data. You might think that you need to find the shapefile relevant to your research problem, but you do not. You need to find shapefiles defining the geographic units that you will be analyzing, such as U.S. counties. In addition to the map, shapefiles include the names and standard codes for the geographic units. You will later use those variables to merge the shapefile with data you already have or that you obtain from other sources.

To find appropriate shapefiles, use your browser and search for them. You could search for

```
shapefiles

shapefiles europe
shapefiles deutschland
shapefiles deutschland bundesländer
shapefiles schweiz kantone
shapefiles uk
shapefiles uk county

shapefiles us
shapefiles us census
shapefiles us census county
shapefiles us census blocks

shapefiles us census tiger     // TIGER/Line is especially popular
```

It is best to choose a shapefile from official sources. If such a shapefile is not available, choose one that is from a reputable source.

Find the appropriate shapefile and download it.

Standard-format shapefiles

The shapefile you just loaded is known as a standard-format shapefile. The word shapefile itself is confusing because a shapefile is actually a collection of related files. For example, a shapefile could be any of the following:

File	Contents
name`.shp`	shapes and locations of geographic units
name`.dbf`	other attributes of the geographic units
name`.*`	other information, not needed by Sp
name`.zip`	compressed file containing all the files above

name`.zip` is often called a shapefile even though it is a zip file containing the shapefiles.

name`.shp` really is a shapefile—it contains the map of the geographic units, which could be countries of the world, counties of the United States, etc.

name.dbf contains data (called attributes) about the geographic units. The .dbf stands for database file. It is a dataset containing variables and observations. Among the variables are usually variables for the names and numeric identification codes of the geographic units. The file sometimes contains other variables, such as temperature, rainfall, or unemployment. After translation to Sp format, you can use the variables, ignore them, or drop them.

In addition to *name*.shp and *name*.dbf, there are other files. Stata ignores them, and you can erase them if you wish. After translation, you can erase all the files that were in the original .zip file.

Stata-format shapefiles

You will translate the standard-format shapefiles to Stata format. It is easy to do:

```
. unzipfile name.zip
. spshape2dta name
```

This will create two Stata-format datasets, *name*.dta and *name*_shp.dta.

Stata-format file	Corresponding standard-format file
name.dta	*name*.dbf
*name*_shp.dta	*name*.shp

name.dta contains

Variable name	Contents
_ID	ID variable with values $1, 2, \ldots, N$
_CX	x coordinate of centroid of geographic unit
_CY	y coordinate of centroid of geographic unit
other variables	attributes of the geographic units from *name*.dbf

Notes: 1. The dataset will have N observations, one for each geographic unit.
2. You will learn later that Sp data must be spset. spshape2dta handles that for you. *name*.dta is spset.
3. Variable _ID links observations in *name*.dta with the map stored in *name*_shp.dta.
4. You may rename, modify, or drop any of the variables except _ID, _CX, and _CY.
5. You merge your .dta files with *name*.dta to use them in Sp.

*name*_shp.dta contains

Variable name	Contents
_ID	ID variable with values $1, 2, \ldots, N$
other variables	descriptions of the map

Notes: 1. This file has many more than N observations. Each observation describes a line segment that when combined draws the map.
2. You do not use or modify this dataset. Sp uses the dataset behind the scenes.
3. *name*.dta and *name*_shp.dta must be in the same directory.

Creating Stata-format shapefiles

There are six steps to preparing shapefiles for use:

1. Find and download a standard-format shapefile.
2. Translate the shapefile to Stata format.
3. Look at the translated data.
4. Create a common ID variable for use with other data.
5. Optionally, tell Sp to use the common ID variable.
6. Set the units of the coordinates, if necessary.

These steps are not independent; that is, you cannot jump ahead to, say, step 4.

Below, we start at step 1, finding and downloading

```
tl_2016_us_county.zip
```

and finish with step 6, having created

```
tl_2016_us_county.dta
tl_2016_us_county_shp.dta
```

These are the same files we used in *Overview*.

We discuss each step below. Here is a preview of the code for the steps:

Step 1: Find and download a standard-format shapefile

```
. * do this on the web
```

Step 2: Translate the shapefile to Stata format

```
. copy ~/Downloads/tl_2016_us_county.zip .
. unzipfile tl_2016_us_county.zip
. spshape2dta tl_2016_us_county
```

Step 3: Look at the translated data

```
. use tl_2016_us_county, clear
. describe
. list in 1/5
```

Step 4: Create a common ID variable for use with other data

```
. generate long fips = real(STATEFP + COUNTYFP)
. bysort fips: assert _N==1
. assert fips != .
```

Step 5: Optionally, tell Sp to use the common ID variable

```
. spset fips, modify replace
```

Step 6: Set the units of the coordinates, if necessary

```
. spset, modify coordsys(latlong, miles)
. save, replace
```

Step 1: Find and download a shapefile

Use your browser. We did, and we found and downloaded `tl_2016_us_county.zip` as described in *Finding a shapefile for Texas counties* in [SP] **Intro 7**. Our browser stored the file in our `Downloads` directory, which is `~/Downloads/` on our computer. `~` is Stata syntax for home directory.

Step 2: Translate the shapefile to Stata format

We entered Stata and changed to the directory containing the project datasets. We typed

```
. copy ~/Downloads/tl_2016_us_county.zip .
. unzipfile tl_2016_us_county.zip
    inflating: tl_2016_us_county.cpg
    inflating: tl_2016_us_county.dbf
    inflating: tl_2016_us_county.prj
    inflating: tl_2016_us_county.shp
    inflating: tl_2016_us_county.shp.ea.iso.xml
    inflating: tl_2016_us_county.shp.iso.xml
    inflating: tl_2016_us_county.shp.xml
    inflating: tl_2016_us_county.shx
successfully unzipped tl_2016_us_county.zip to current directory
. spshape2dta tl_2016_us_county
  (importing .shp file)
  (importing .dbf file)
  (creating _ID spatial-unit id)
  (creating _CX coordinate)
  (creating _CY coordinate)
  file tl_2016_us_county_shp.dta created
  file tl_2016_us_county.dta     created
```

`spshape2dta` translated the files to Stata format. It did not load them into memory. You will never load the `*_shp.dta` file, but Sp will use it behind the scenes. The file is linked to `tl_2016_us_county.dta`, which you will directly use. Keep them both in the same directory.

Step 3: Look at the translated data

Look at the data you have just created. The data are already `spset`, but we can type `spset` to find out how:

```
. use tl_2016_us_county, clear
. spset
  Sp dataset tl_2016_us_county.dta
                data:  cross sectional
   spatial-unit id:  _ID
       coordinates:  _CX, _CY (planar)
  linked shapefile:  tl_2016_us_county_shp.dta
```

Look at the variables, too:

```
. describe
  (output omitted)
. list in 1/5
  (output omitted)
```

You need to understand the data and its variables. Some of them you will not need. You may drop them, but do not drop `_ID`, `_CX`, and `_CY`. They were created by `spshape2dta`, and you will need them later.

In the unlikely event that you find all the variables you need for your intended analysis, you can use tl_2016_us_county.dta as your analysis dataset. You are ready to go, except you might need to set the coordinate system. Skip to step 6, and stop after that.

Step 4: Create a common ID variable for use with other data

We continue with step 4 because we did not find the analysis variables we needed, nor did we expect to find them. We have project_cs.dta containing our analysis variables. The problem is that we will need to merge project_cs.dta with the Stata-format shapefiles, and to do that, they will need to have an ID variable in common. project_cs.dta has a variable named fips containing standard county codes. We hope to find the same variable in tl_2016_us_county.dta.

We looked but did not find the FIPS-code variable. We did discover the variable NAME containing county names. That variable could work for us. project_cs.dta also has a variable named countyname. If we rename NAME to countyname in tl_2016_us_county.dta, we could merge datasets.

However, we have had bad experiences merging on string variables. Names in the two datasets can differ for trivial reasons, such as capitalization. Before we resigned ourselves to the string-variable solution, we looked again. Numeric ID variables are better.

We discovered variables STATEFP and COUNTYFP. They were recorded as string variables, but appeared to contain two- and three-digit numeric codes. We read about FIPS codes on the web and learned there are two-digit state codes, three-digit county-within-state codes, and five-digit county codes, which are nothing more than the two- and three-digit codes run together. If STATEFP is 01 and COUNTYFP is 001, then the five-digit code is 01001.

We create the new numeric variable fips containing the run-together code by typing

```
. generate long fips = real(STATEFP + COUNTYFP)
```

The variable we created did not have to be numeric, but fips is numeric in project_cs.dta, and numeric is better for reasons to be explained in step 5.

In any case, we were pleased when we listed the value of variable NAME for fips = 1001 and it was Autauga.

We also verify that new variable fips really does uniquely identify the observations in tl_2016_us_county.dta by typing

```
. bysort fips: assert _N==1
. assert fips != .
```

Step 5: Optionally, tell Sp to use the common ID variable

This step is optional but worth doing if you found or created a numeric ID variable in the previous step. Because we created fips in step 4, we will type

```
. spset fips, modify replace
  (_shp.dta file saved)
  (data in memory saved)
  Sp dataset tl_2016_us_county.dta
                data:  cross sectional
   spatial-unit id:  _ID (equal to fips)
       coordinates:  _CX, _CY (planar)
  linked shapefile:  tl_2016_us_county_shp.dta
```

The above resets _ID. `spset` verifies that `fips` is numeric and would make an appropriate ID code. If it does, `spset` copies `fips` to Sp's _ID variable, the variable that officially identifies the observations. Sp then reindexes both `tl_2016_us_county.dta` and `tl_2016_us_county_shp.dta` on the new _ID values.

You should do this step because, if _ID is a common code, the spatial weighting matrices you create will be sharable with other projects and researchers. The rows and columns of the matrices will be identified by the common code rather than the arbitrary code _ID previously contained.

Step 6: Set the units of the coordinates, if necessary

The coordinates recorded in shapefiles historically were required to be in planar units. These days, shapefiles are just as likely to contain latitude and longitude. Usage is running ahead of file-format standards, and so you must determine which coordinate system is being used.

When Sp converts a shapefile as we did in step 2, it assumes coordinates are in planar units. If they are actually recorded in degrees latitude and longitude, you need to type

```
. spset, modify coordsys(latlong, miles)
```

or

```
. spset, modify coordsys(latlong, kilometers)
```

Whether you specify miles or kilometers is of little importance—that setting merely determines the units in which Sp will report distances. It is important, however, that you specify the coordinate system is `latlong` when it is latitude and longitude if distances are to be measured accurately.

The distributor of the shapefile may provide documentation that tells you whether the file uses planar units or latitude and longitude. If you are unable to find this information, you can do some detective work to figure it out.

Here is how to determine the units. Coordinates (centroids) are stored in variables _CX and _CY. We listed some of them and discovered that Brazos County, Texas, is recorded as being at

$$\_\mathtt{CX} = -96.302386 \quad \text{and} \quad \_\mathtt{CY} = 30.6608$$

We looked on the web and found that College Station, a city in Brazos County, is located at latitude 30.601389 and longitude −96.314444. We checked two other cities and counties and found similar agreement. (Note that latitude is stored in _CY and longitude in _CX. It will always be that way.)

Thus, we type

```
. spset, modify coordsys(latlong, miles)
  Sp dataset tl_2016_us_county.dta
                data:  cross sectional
   spatial-unit id:  _ID (equal to fips)
       coordinates:  _CY, _CX (latitude-and-longitude, miles)
  linked shapefile:  tl_2016_us_county_shp.dta
```

We are finished preparing our shapefile, so we save `tl_2016_us_county.dta`.

```
. save, replace
file tl_2016_us_county.dta saved
```

Preparing your data

We now have

```
tl_2016_us_county.dta
tl_2016_us_county_shp.dta
```

These are the same datasets we used in *Overview*.

You should keep these two files around, just as they are. You can use them in the future whenever you have a county dataset that you want to use with Sp.

Step 7a: Merge your cross-sectional data with the Stata-format shapefiles

We showed you how to do this in the *Overview*, but we will do it again now that we have our Stata-format shapefiles so that you can see the output. To make the cross-sectional data in `project_cs.dta` work with Sp, type

```
. use project_cs, clear
. merge 1:1 fips using tl_2016_us_county
. keep if _merge==3
. drop _merge
. save, replace
```

The result is

```
. use project_cs, clear
(My cross-sectional data)
. merge 1:1 fips using tl_2016_us_county
    Result                           # of obs.
    -----------------------------------------
    not matched                             91
        from master                          0  (_merge==1)
        from using                          91  (_merge==2)
    matched                              3,142  (_merge==3)
    -----------------------------------------

. keep if _merge==3
(91 observations deleted)
. drop _merge
. save, replace
file project_cs.dta saved
```

Note that all observations from the master were matched. Had observations been dropped from the master, we would have found out why `project_cs.dta` contained counties not in `tl_2016_us_county.dta`.

We have not discussed the `spset` command, the other way to turn regular Stata datasets into Sp datasets. We will discuss `spset` in [SP] **Intro 5** and [SP] **Intro 6**. Merging regular data (`project_cs.dta`) with `spset` data (`tl_2016_us_county.dta`, because it was created by `spshape2dta`) produces an `spset` result. `project_cs.dta` was not `spset` before the merge, but it is now:

```
. spset
  Sp dataset project_cs.dta
                data:  cross sectional
   spatial-unit id:  _ID (equal to fips)
       coordinates:  _CY, _CX (latitude-and-longitude, miles)
  linked shapefile:  tl_2016_us_county_shp.dta
```

Step 7b: Merge your panel data with the Stata-format shapefiles

Because `project_panel.dta` is panel data, you still merge with `tl_2016_us_county.dta`, but you go about it a little differently. You type

```
. use project_panel, clear
. xtset fips time
. spbalance
. merge m:1 fips using tl_2016_us_county
. keep if _merge==3
. drop _merge
. save, replace
```

The result is

```
. use project_panel, clear
(My panel data)
. xtset fips time
       panel variable:  fips (strongly balanced)
        time variable:  time, 1 to 3
                delta:  1 unit
. spbalance
  (data strongly balanced)
. merge m:1 fips using tl_2016_us_county
    Result                           # of obs.
    -----------------------------------------
    not matched                             91
        from master                          0  (_merge==1)
        from using                          91  (_merge==2)
    matched                              9,426  (_merge==3)
    -----------------------------------------
. keep if _merge==3
(91 observations deleted)
. drop _merge
. save, replace
file project_panel.dta saved
```

project_panel.dta is now spset:

```
. spset
  Sp dataset project_panel.dta
                data:  panel
   spatial-unit id:  _ID (equal to fips)
           time id:  time (see xtset)
       coordinates:  _CY, _CX (latitude-and-longitude, miles)
  linked shapefile:  tl_2016_us_county_shp.dta
```

The data are still xtset, but Sp modified the setting. The data were set on fips and time. They are now set on _ID and time:

```
. xtset
      panel variable:  _ID (strongly balanced)
       time variable:  time, 1 to 3
               delta:  1 unit
```

Sp changed the setting because spset and xtset must agree on the panel identifier.

Rules for working with Sp data, whether cross-sectional or panel

The data whether cross-sectional, as in project_cs.dta, or panel, as in project_panel.dta, is now Sp. It is a Stata dataset with one special feature: its observations are linked to the Stata-format shapefile tl_2016_us_shp.dta. Because of the linkage, there are rules for using either project_cs.dta or project_panel.dta.

Rule 1: Do not drop or modify variables _ID, _CX, or _CY.

You may drop other variables in the file.

Rule 2:

Cross-sectional data:

Do not add new observations.

Panel data:

Do not add new observations with new values of _ID.

The rule that handles both cross-sectional and panel data is that you may not add observations that have no corresponding definition in tl_2016_us_shp.dta.

For cross-sectional data, the rule reduces to "do not add new observations".

For panel data, the rule said positively is that you can add new observations, but only for new time periods within panels.

You may drop observations from cross-sectional data, and observations for entire panels from panel data. Dropping is allowed because unnecessary definitions in tl_2016_us_shp.dta are ignored.

Be careful when performing merges with other datasets. If you type

Cross-sectional data:

```
. merge 1:1 fips using anotherdataset
```

Panel data:

```
. merge 1:1 fips time using anotherdataset
```

or

```
. merge m:1 fips using anotherdataset
```

you must then either

```
. keep if _merge==3
```

or

```
. keep if _merge==1
```

Rule 3: Do not erase, modify, or rename file tl_2016_us_shp.dta.

Even if you rename `project_cs.dta` or `project_panel.dta`, do not rename `tl_2016_us_shp.dta`.

Rule 4: project_cs.dta or project_panel.dta and tl_2016_us_shp.dta must be stored in the same directory.

If you copy `project_cs.dta` or `project_panel.dta` to a different directory, copy `tl_2016_us_shp.dta` to the same directory.

That is the end of the prohibitions. The following rule need not be stated, because that which is not prohibited is allowed, but it is reassuring:

Rule 5: You may save copies of project_cs.dta or project_panel.dta under new names.

New files will inherit the linkage to `tl_2016_us_shp.dta`. For example, you could type

```
. copy project_cs.dta newname.dta
```

Afterward, if you wished, you could type

```
. erase project_cs.dta
```

Here is one way making copies can be useful:

```
. use project_cs
. keep if state=="Texas"
. save texas
```

Also see

[SP] **Intro 7** — Example from start to finish

[SP] **spset** — Declare data to be Sp spatial data

[SP] **spshape2dta** — Translate shapefile to Stata format

Title

Intro 5 — Preparing data: Data containing locations (no shapefiles)

Description Remarks and examples Also see

Description

If you have data that already contain the coordinates of the geographical units, you are not required to use the shapefiles discussed in [SP] **Intro 4**. You may, however, want to use shapefiles. Without them, you cannot create contiguity weighting matrices (matrices in which spillovers occur only among adjacent places), nor can you draw choropleth maps.

This entry provides steps to prepare data with no shapefiles for use by Sp.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 1	A brief introduction to SAR models
Intro 2	The **W** matrix
Intro 3	Preparing data for analysis
Intro 4	Preparing data: Data with shapefiles
Intro 6	Preparing data: Data without shapefiles or locations
Intro 7	Example from start to finish
Intro 8	The Sp estimation commands

Remarks and examples

Remarks are presented under the following headings:

Preparation of cross-sectional data
Preparation of panel data
There are no rules as there are with shapefiles

Preparation of cross-sectional data

We will assume that you have file `project_cs2.dta`, which is a cross-sectional dataset on U.S. counties over time, variable `fips` containing the standard county codes, and variables `locx` and `locy` identifying the location of each county.

To turn `project_cs2.dta` into Sp data, do the following:

Step 1: Load the dataset

```
. use project_cs2, clear
```

Step 2: Verify that `fips` is an ID variable

```
. assert fips!=.
. bysort fips: assert _N==1
```

Step 3: spset the data

```
. spset fips, coord(locx locy)
```

Step 4: Set the coordinate units, if necessary

```
. spset, coordsys(latlong, miles)
```

Step 5: Save the data

```
. save, replace
```

That is all there is to it.

In step 3, we specified option coord(locx locy). spset will create new variables _ID, _CX, and _CY. It will copy fips into _ID, and locx and locy into _CX and _CY.

In step 4, we set the coordinate system to degrees latitude and longitude because that was necessary in this case. We discussed in [SP] **Intro 4** how to determine the coordinate system.

In step 5, we saved project_cs2.dta over itself. The new dataset differs from the old in that it has three new variables and is spset. No changes or deletions were made to the data.

Preparation of panel data

This time, suppose project_panel2.dta is a panel dataset on U.S. counties over time. Perhaps it is already xtset on fips and time. The dataset also includes variables locx and locy identifying the location of each county.

To turn project_panel2.dta into Sp data, do the following:

Step 1: Load the dataset

```
. use project_panel2, clear
```

Step 2: Verify that fips and time jointly identify the observations

```
. assert fips!=.
. assert time!=.
. bysort fips time: assert _N==1
```

Step 2a: xtset the data and verify that locx and locy are constant within panel

```
. xtset, clear
. xtset fips time
. bysort fips (time): assert locx == locx[1]
. bysort fips (time): assert locy == locy[1]
```

Step 3: Balance and spset the data

```
. spbalance
. spset fips, coord(locx locy)
```

Step 4: Set the coordinate units, if necessary

```
. spset, coordsys(latlong)   // optional
```

Step 5: Save the data

```
. save, replace     or     save newfilename, replace
```

Concerning step 5, type `save, replace` only if step 3 did not involve dropping data.

In step 3, we `spset` the data, but not before verifying that they are strongly balanced. If the data are not strongly balanced, `spbalance` will issue an error and suggest that you type

```
. spbalance, balance
```

If you type that, `spbalance` will balance the data.

Then we `spset` the data. This creates the new variables `_ID`, `_CX`, and `_CY`. `spset` copies `fips` into `_ID` and copies `locx` and `locy` into `_CX` and `_CY`.

In step 4, we set coordinate units to degrees latitude and longitude. We discussed how to determine coordinate units in [SP] **Intro 4**.

There are no rules as there are with shapefiles

There are no special rules for working with the data created here as there were when working with data and shapefiles. The rules in [SP] **Intro 4** arose because of the linkage between the data file and its linked `*_shp.dta` file.

Also see

[SP] **spbalance** — Make panel data strongly balanced

[SP] **spset** — Declare data to be Sp spatial data

Title

Intro 6 — Preparing data: Data without shapefiles or locations

Description Remarks and examples Also see

Description

This entry outlines the preparation of data without shapefiles or locations. Such data arise when spillover effects are based not on physical proximity but on proximity in other metrics.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 1	A brief introduction to SAR models
Intro 2	The **W** matrix
Intro 3	Preparing data for analysis
Intro 4	Preparing data: Data with shapefiles
Intro 5	Preparing data: Data containing locations (no shapefiles)
Intro 7	Example from start to finish
Intro 8	The Sp estimation commands

Remarks and examples

Remarks are presented under the following headings:

Nongeographic spatial data

Spatial analysis is about accounting for spillover effects. Consider an analysis of test scores of students. There may be spillover effects among friends for no other reason than friends share similar but relevant unmeasured characteristics. Or you might hypothesize more direct effects. Such data are known as social network data.

Consider the dollar value of trade between countries. Effects may spillover from one country to the next based on closeness measured by industry and the development level. Closeness might be based on the dissimilarity of industry (providing a reason to trade) and similarity of development level.

In these cases, the construction of the **W** spatial weighting matrices is often a substantive research problem in and of itself. As a result, researchers share weighting matrices. If you are analyzing such data, see [SP] **spmatrix import**. If you create such matrices, see [SP] **spmatrix userdefined**, [SP] **spmatrix fromdata**, [SP] **spmatrix spfrommata**, and [SP] **spmatrix export**.

First, however, you must prepare the data for use by Sp.

Preparation of cross-sectional data

We will assume that you have a dataset named `project_cs3.dta` that contains observations on nodes with variable `node_id` containing the standard codes for them.

To turn `project_cs3.dta` into Sp data, do the following:

Step 1: Load the data

```
. use project_cs3, clear
```

Step 2: Verify that `node_id` is an ID variable

```
. assert node_id!=.
. bysort node_id: assert _N==1
```

Step 3: `spset` the data

```
. spset node_id
```

Step 4: Save the data

```
. save, replace
```

In step 3, when we `spset` the data, `spset` created the new variable `_ID` containing a copy of the values in `node_id`. Variables `_CX` and `_CY` will not be created as they were in [SP] **Intro 4** and [SP] **Intro 5**, because these data do not contain location information.

In step 4, we save `project_cs3.dta` over itself. The new dataset differs from the old in that it has a new variable and it is `spset`. No changes or deletions were made to the data.

Preparation of panel data

We will now assume that you have `project_panel3.dta`, which is a panel dataset based on `node_id` and `time`.

To turn `project_panel3.dta` into Sp data, do the following:

Step 1: Load the dataset

```
. use project_panel3, clear
```

Step 2: Verify that `node_id` and `time` are jointly an ID variable

```
. assert node_id!=.
. assert time!=.
. bysort node_id time: assert _N==1
```

Step 2a: `xtset` the data

```
. xtset, clear
. xtset node_id time
```

Step 3: Balance and `spset` the data

```
. spbalance
. spset node_id
```

Step 4: Save the data

`. save, replace` *or* `save` *newfilename*

Concerning step 4, type `save, replace` only if step 3 did not involve dropping data.

There are no rules as there are with shapefiles

There are no special rules for working with the data created here as there were when working with data and shapefiles. The rules in [SP] **Intro 4** arose because of the linkage between the data file and its `*_shp.dta` file.

Also see

[SP] **spbalance** — Make panel data strongly balanced

[SP] **spset** — Declare data to be Sp spatial data

Title

Intro 7 — Example from start to finish

Description Remarks and examples Also see

Description

This entry comprises an example from start to finish.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 1	A brief introduction to SAR models
Intro 2	The **W** matrix
Intro 3	Preparing data for analysis
Intro 4	Preparing data: Data with shapefiles
Intro 5	Preparing data: Data containing locations (no shapefiles)
Intro 6	Preparing data: Data without shapefiles or locations
Intro 8	The Sp estimation commands

Remarks and examples

Remarks are presented under the following headings:

Research plan

We are going to analyze unemployment in counties of Texas. We are going to use `texas_ue.dta`. The data contain unemployment rates and college graduation rates for Texas counties, but they do not include the locations of the counties or a map. The data can be used to fit models with `regress`, but they do not contain the information necessary to fit models with `spregress` that could account for spillover effects.

We will

1. Find and download a U.S. counties shapefile.
2. Translate the downloaded file to Stata format.
3. Merge the translated file with our existing data.
4. Analyze the merged data.

Please keep in mind that this is just an example in a computer software manual. We will model the unemployment rate as a function of college graduation rate only, though we ought to include other explanatory variables. We analyze data for Texas only, though we should use the entire United States. We will draw conclusions that are unjustified, and we will not qualify them appropriately. We will, however, show you how to use `spregress` and interpret its output.

Finding and preparing data

We first find and download an appropriate shapefile from the web. Then, we will prepare it as described in [SP] **Intro 4**.

Finding a shapefile for Texas counties

We looked for a county shapefile for Texas but could not find one. We did find shapefiles for the entire United States, however. We used our browser to search for "shapefile U.S. counties census". From the results, we selected *TIGER/Line Shapefile, 2016, nation, U.S., Current County and Equivalent National Shapefile*. On the resulting page, we clicked to download the **Shapefile Zip File** from the **Downloads & Resources** section. File `tl_2016_us_county.zip` was downloaded to the `Downloads` directory on our computer.

Creating the Stata-format shapefile

We found a standard-format shapefile, `tl_2016_us_county.zip`. We now follow the instructions in [SP] **Intro 4** to create a Stata-format shapefile. Here is the result:

```
. // ------------------------------------------------------------
. // [SP] intro 4, step 2:  Translate the shapefile
.
. copy ~/Downloads/tl_2016_us_county.zip .
. unzipfile tl_2016_us_county.zip
    inflating: tl_2016_us_county.cpg
    inflating: tl_2016_us_county.dbf
    inflating: tl_2016_us_county.prj
    inflating: tl_2016_us_county.shp
    inflating: tl_2016_us_county.shp.ea.iso.xml
    inflating: tl_2016_us_county.shp.iso.xml
    inflating: tl_2016_us_county.shp.xml
    inflating: tl_2016_us_county.shx
successfully unzipped tl_2016_us_county.zip to current directory
```

```
. spshape2dta tl_2016_us_county
  (importing .shp file)
  (importing .dbf file)
  (creating _ID spatial-unit id)
  (creating _CX coordinate)
  (creating _CY coordinate)
  file tl_2016_us_county_shp.dta created
  file tl_2016_us_county.dta     created
.
. // -------------------------------------------------------
. // [SP] intro 4, step 3: Look at the data
.
. use tl_2016_us_county, clear
. describe
Contains data from tl_2016_us_county.dta
  obs:         3,233
 vars:            20                          14 Apr 2019 09:51
```

variable name	storage type	display format	value label	variable label
_ID	int	%12.0g		
_CX	double	%10.0g		x-coordinate of area centroid
_CY	double	%10.0g		y-coordinate of area centroid
STATEFP	str2	%9s		STATEFP
COUNTYFP	str3	%9s		COUNTYFP
COUNTYNS	str8	%9s		COUNTYNS
GEOID	str5	%9s		GEOID
NAME	str21	%21s		NAME
NAMELSAD	str33	%33s		NAMELSAD
LSAD	str2	%9s		LSAD
CLASSFP	str2	%9s		CLASSFP
MTFCC	str5	%9s		MTFCC
CSAFP	str3	%9s		CSAFP
CBSAFP	str5	%9s		CBSAFP
METDIVFP	str5	%9s		METDIVFP
FUNCSTAT	str1	%9s		FUNCSTAT
ALAND	double	%14.0f		ALAND
AWATER	double	%14.0f		AWATER
INTPTLAT	str11	%11s		INTPTLAT
INTPTLON	str12	%12s		INTPTLON

```
Sorted by: _ID
. list in 1/2
```

1.

_ID	_CX	_CY	STATEFP	COUNTYFP	COUNTYNS	GEOID
1	-96.7874	41.916403	31	039	00835841	31039

NAME	NAMELSAD	LSAD	CLASSFP	MTFCC	CSAFP	CBSAFP
Cuming	Cuming County	06	H1	G4020		

METDIVFP	FUNCSTAT	ALAND	AWATER	INTPTLAT
	A	1477895811	10447360	+41.9158651

INTPTLON
-096.7885168

```
  2.
```

_ID	_CX	_CY	STATEFP	COUNTYFP	COUNTYNS	GEOID
2	-123.43347	46.291134	53	069	01513275	53069

NAME	NAMELSAD	LSAD	CLASSFP	MTFCC	CSAFP	CBSAFP
Wahkiakum	Wahkiakum County	06	H1	G4020		

METDIVFP	FUNCSTAT	ALAND	AWATER	INTPTLAT
	A	680956787	61588406	+46.2946377

INTPTLON
-123.4244583

```
.
. // ------------------------------------------------------------
. // [SP] intro 4, step 4: Create standard ID variable
.
. generate long fips = real(STATEFP + COUNTYFP)
. bysort fips: assert _N==1
. assert fips != .
.
. // ------------------------------------------------------------
. // [SP] intro 4, step 5: Tell Sp to use standard ID variable
.
. spset fips, modify replace
  (_shp.dta file saved)
  (data in memory saved)
  Sp dataset tl_2016_us_county.dta
                data:  cross sectional
     spatial-unit id:  _ID (equal to fips)
         coordinates:  _CX, _CY (planar)
    linked shapefile:  tl_2016_us_county_shp.dta
.
. // ------------------------------------------------------------
. // [SP] intro 4, step 6: Set coordinate units
.
. spset, modify coordsys(latlong, miles)
  Sp dataset tl_2016_us_county.dta
                data:  cross sectional
     spatial-unit id:  _ID (equal to fips)
         coordinates:  _CY, _CX (latitude-and-longitude, miles)
    linked shapefile:  tl_2016_us_county_shp.dta

. save, replace
file tl_2016_us_county.dta saved
. // ------------------------------------------------------------
```

Merging our data with the Stata-format shapefile

Recall that we are going to use `texas_ue.dta` containing unemployment rates and college graduation rates for Texas counties. We follow the instructions in [SP] **Intro 4**, *Step 7a* to merge our existing data with the Stata-format shapefile.

```
. copy https://www.stata-press.com/data/r16/texas_ue.dta .
. use texas_ue, clear
. describe
Contains data from texas_ue.dta
  obs:           254
 vars:             4                          10 Feb 2019 12:36
                                               (_dta has notes)
------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
------------------------------------------------------------------------------
fips            float   %9.0g                 FIPS
college         float   %9.0g                * Percent college degree
income          long    %12.0g                Median household income
unemployment    float   %9.0g                 Unemployment rate
                                             * indicated variables have notes
------------------------------------------------------------------------------
Sorted by: fips
. merge 1:1 fips using tl_2016_us_county
(note: variable fips was float, now double to accommodate using data's
       values)

    Result                           # of obs.
    -----------------------------------------
    not matched                         2,979
        from master                         0  (_merge==1)
        from using                      2,979  (_merge==2)

    matched                               254  (_merge==3)
    -----------------------------------------

. keep if _merge==3
(2,979 observations deleted)
. drop _merge
```

At this point, we type `describe` again and discover that `texas_ue.dta` has lots of unnecessary, leftover variables from `tl_2016_us_county.dta`, so we drop them. There is another variable that we rather like—the names of the counties—and we rename it.

```
. rename NAME countyname
. drop STATEFP COUNTYFP COUNTYNS GEOID
. drop NAMELSAD LSAD CLASSFP MTFCC CSAFP
. drop CBSAFP METDIVFP FUNCSTAT
. drop ALAND AWATER INTPTLAT INTPTLON
. save, replace
file texas_ue.dta saved
```

Analyzing texas_ue.dta

File texas_ue.dta is our updated analysis dataset that can be used with Sp commands.

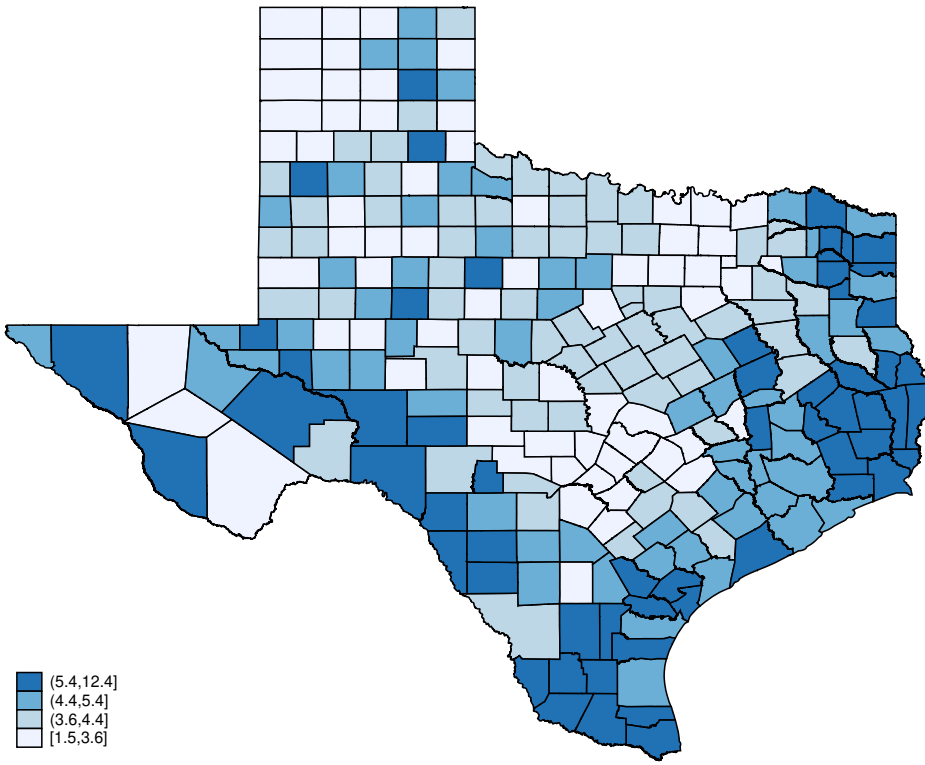
```
. describe
Contains data from texas_ue.dta
  obs:           254
 vars:             8                          26 Feb 2019 16:10
                                               (_dta has notes)
-------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
-------------------------------------------------------------------------------
fips            double  %9.0g                 FIPS
college         float   %9.0g                * Percent college degree
income          long    %12.0g                Median household income
unemployment    float   %9.0g                 Unemployment rate
_ID             long    %12.0g                Spatial-unit ID
_CX             double  %10.0g                x-coordinate of area centroid
_CY             double  %10.0g                y-coordinate of area centroid
countyname      str21   %21s                  NAME
                                             * indicated variables have notes
-------------------------------------------------------------------------------
Sorted by:
```

Our example uses the unemployment rate. It varies between 1.5% and 12.4% across the counties of Texas:

```
. summarize unemployment
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
unemployment |        254    4.731102    1.716514         1.5       12.4
```

Because `texas_ue.dta` has been `spset` and has a shapefile, we can draw choropleth maps, such as this one of the unemployment rate:

```
. grmap unemployment
```



Unemployment appears to be clustered, which suggests that there are spillover effects between counties.

Testing whether ordinary regression is adequate

These data are suitable for both spatial and nonspatial analysis. (Spatial data always are.) We will fit a linear regression of the unemployment rate on the college graduation rate, mostly for illustrative purposes. After fitting the linear regression, we will use an Sp command to determine whether the residuals of the model are spatially correlated, and we find that they are.

Here is the regression:

```
. regress unemployment college
```

Source	SS	df	MS	Number of obs	=	254
				F(1, 252)	=	57.92
Model	139.314746	1	139.314746	Prob > F	=	0.0000
Residual	606.129539	252	2.40527595	R-squared	=	0.1869
				Adj R-squared	=	0.1837
Total	745.444285	253	2.9464201	Root MSE	=	1.5509

unemployment	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
college	-.1008791	.0132552	-7.61	0.000	-.1269842	-.0747741
_cons	6.542796	.2571722	25.44	0.000	6.036316	7.049277

The results of this oversimplified model indicate that the college graduation rate reduces unemployment markedly.

Are we done? If the residuals show no signs of being spatially clustered, then we are. We can perform a statistical test.

Sp provides the Moran test for determining whether the residuals of a model fit by `regress` are correlated with nearby residuals. To use it, we must define "nearby". We do that by defining a spatial weighting matrix, which is created by the `spmatrix` command. We will define a contiguity matrix.

```
. spmatrix create contiguity W
```

This contiguity matrix sets "nearby" to mean "shares a border".

`spmatrix` can create other types of weighting matrices. It even allows you to create custom matrices or to import matrices. See [SP] **spmatrix**.

We can now run the Moran test.

```
. estat moran, errorlag(W)

Moran test for spatial dependence
         Ho: error is i.i.d.
         Errorlags:  W

         chi2(1)      =    94.06
         Prob > chi2  =   0.0000
```

The test reports that we can reject that the residuals from the model above are independent and identically distributed (i.i.d.). In particular, the test considered the alternative hypothesis that residuals are correlated with nearby residuals as defined by $\mathbf{W}$.

spregress can reproduce regress results

`spregress` is the spatial autoregression command. `spregress` fits models in which the observations are not independent, as defined by the $\mathbf{W}$ weighting matrix.

Above, we fit a model under the assumption that the counties are independent. We used `regress`, Stata's ordinary linear regression command. We typed

```
. regress unemployment college
```

We could have fit the same model and obtained the same results by using `spregress`. We would have typed

```
. spregress unemployment college, gs2sls
```

or

```
. spregress unemployment college, ml
```

`spregress` is seldom used for fitting models without spatial lags or autocorrelated errors, but when it is, it reports the same linear regression results that `regress` reports, although there are some differences. Standard errors are slightly different, and `spregress` reports Z and χ^2 statistics instead of t and F statistics. `spregress` does not include the finite-sample adjustments that `regress` does because it does not expect to be used in situations where those adjustments would be appropriate.

Fitting models with a spatial lag of the dependent variable

We will use `spregress` to fit the same model we fit using `regress` but with the addition of a spatial lag of unemployment. The model we fit will be

$$\mathbf{y}_{\text{ue}} = \beta_0 + \beta_1 \mathbf{x}_{\text{cr}} + \beta_2 \mathbf{W}\mathbf{y}_{\text{ue}} + \boldsymbol{\epsilon}$$

$\mathbf{y}_{\text{ue}}$ is the unemployment rate corresponding to variable `unemployment` in our data. $\mathbf{x}_{\text{cr}}$ is the college graduation rate corresponding to variable `college`.

The model we fit will include the term $\beta_2 \mathbf{W}\mathbf{y}_{\text{ue}}$, meaning that we will assume the unemployment rate spills over from nearby counties. There is a real logic to such a model. One would expect workers in high unemployment counties to seek employment nearby.

`spregress` provides two ways of fitting models: generalized spatial two-stage least squares (`gs2sls`) and maximum likelihood (`ml`). To fit the above model, we could type

```
. spregress unemployment college, gs2sls dvarlag(W)
```

or

```
. spregress unemployment college, ml dvarlag(W)
```

`spregress, ml` is statistically more efficient than `gs2sls` when the errors are normally distributed. Efficiency is desirable, so we should use `ml`, right? That same property said differently is that `gs2sls` is robust to violations of normality. Robustness is desirable, too. So now the choice between them hinges on whether we believe the normality assumption. That said, `ml` will provide standard errors that are also robust to violations of normality if we specify its `vce(robust)` option. Finally, `ml` takes longer to run, and that computation time increases as the number of observations increases. We will use `gs2sls`.

```
. spregress unemployment college, gs2sls dvarlag(W)
  (254 observations)
  (254 observations (places) used)
  (weighting matrix defines 254 places)
Spatial autoregressive model                    Number of obs     =        254
GS2SLS estimates                                Wald chi2(2)      =      67.66
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1453
```

unemployment	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
unemployment						
college	-.0939834	.0131033	-7.17	0.000	-.1196653	-.0683015
_cons	5.607379	.5033813	11.14	0.000	4.620769	6.593988
W						
unemployment	.2007728	.0942205	2.13	0.033	.016104	.3854415

Wald test of spatial terms: chi2(1) = 4.54 Prob > chi2 = 0.0331

Results for β_0 and β_1 are similar to those reported by `regress`, but that is a fluke of this example. Usually, when spillover effects are significant, other parameters change. Meanwhile, we find that β_2 (which multiplies $\mathbf{W}\mathbf{y}_{\rm ue}$) is significant, but it is not sharply estimated. The 95% confidence interval places β_2 in the range $[0.02, 0.39]$.

Interpreting models with a spatial lag of the dependent variable

You might be tempted to think of β_1 as the direct effect of education and β_2 as the spillover effect, but they are not. They are ingredients into a recursive calculation of those effects. The model we fit is

$$\mathbf{y}_{\rm ue} = \beta_0 + \beta_1 \mathbf{x}_{\rm cr} + \beta_2 \mathbf{W}\mathbf{y}_{\rm ue} + \boldsymbol{\epsilon}$$

If $\mathbf{x}_{\rm cr}$ increases, that reduces $\mathbf{y}_{\rm ue}$ by β_1, and that reduction in $\mathbf{y}_{\rm ue}$ spills over to produce a further reduction in $\mathbf{y}_{\rm ue}$ of $\beta_2\mathbf{W}$, and that reduction spills over to produce yet another reduction in $\mathbf{y}_{\rm ue}$, and so on.

`estat impact` reports the average effects from the recursive process.

```
. estat impact
progress   :100%
Average impacts                                 Number of obs     =        254
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
college	-.0945245	.0130576	-7.24	0.000	-.120117	-.0689321
indirect						
college	-.0195459	.010691	-1.83	0.068	-.0405	.0014081
total						
college	-.1140705	.0171995	-6.63	0.000	-.1477808	-.0803602

In these data, both the unemployment and the graduation rates are measured in percentage points. A change of 1 is a change of 1 percentage point. The table above reports derivatives, but we can be

forgiven for interpreting the results as if they were for a one-unit change. Everybody does it, and sometimes it is even justifiable, for example, if the model is linear in the variables as this one is. Even if the model were nonlinear, it would be a tolerable approximation to the truth as long as a one-unit change were small.

The table reports average changes for a 1-percentage-point increase in the college graduation rate. The direct effect is the effect of the change within the county, ignoring spillover effects. The own-county direct effect is to reduce the unemployment rate by 0.09 percentage points.

The indirect effect is the spillover effect. A 1-percentage-point increase in the college graduation rate reduces unemployment, and that reduction spills over to further reduce unemployment. The result is a 0.02 reduction in unemployment.

The total effect is the sum of the direct and indirect effects, which is $-0.09 + -0.02 = -0.11$.

You must use `estat impact` to interpret effects. Do not try to judge them from the coefficients that `spregress` reports because they can mislead you. For instance, if we multiplied variable `unemployment` by 100, that would not substantively change anything about the model, yet the effect on the coefficients that `spregress` estimates is surprising.

Summary of `spregress` results

Regression of `unemployment` and `100*unemployment` on `college` and `W*unemployment`

	`unemployment`	`100*unemployment`
`college`	−0.094	−9.4
`W*unemployment`	0.201	0.201

Notes: Column 1 from `spregress` output above.
Column 2 from:

```
generate ue100 = 100*unemployment
spregress unemployment college, gs2sls dvarlag(W)
```

The effect of the change in units is to multiply the coefficient on `college` (β_1) by 100 just as you would expect. Yet β_2, the coefficient on $\mathbf{W}\mathbf{y}_{\text{ue}}$, is unchanged! Comparing these two models, you might mislead yourself into thinking that the ratio of the indirect-to-direct effects is smaller in the second model, but it is not. `estat impact` continues to report the same results as it did previously, multiplied by 100:

```
. estat impact
progress   :100%
Average impacts                                 Number of obs     =        254
```

	Delta-Method dy/dx	Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
college	-9.452455	1.30576	-7.24	0.000	-12.0117	-6.893213
indirect						
college	-1.954593	1.069105	-1.83	0.068	-4.05	.1408134
total						
college	-11.40705	1.719946	-6.63	0.000	-14.77808	-8.036016

Fitting models with a spatial lag of independent variables

We fit a model above with a spatial lag of the dependent variable:

$$\mathbf{y}_{\text{ue}} = \beta_0 + \beta_1 \mathbf{x}_{\text{cr}} + \beta_2 \mathbf{W}\mathbf{y}_{\text{ue}} + \boldsymbol{\epsilon}$$

We could instead fit a model with a spatial lag of the independent variable:

$$\mathbf{y}_{\text{ue}} = \beta_0 + \beta_1 \mathbf{x}_{\text{cr}} + \beta_2 \mathbf{W}\mathbf{x}_{\text{cr}} + \boldsymbol{\epsilon}$$

We do that by typing

```
. spregress unemployment college, gs2sls ivarlag(W:college)
  (254 observations)
  (254 observations (places) used)
  (weighting matrix defines 254 places)
Spatial autoregressive model                    Number of obs     =        254
GS2SLS estimates                                Wald chi2(2)      =      81.13
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.2421
```

unemployment	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
unemployment						
college	-.077997	.0138127	-5.65	0.000	-.1050695	-.0509245
_cons	7.424453	.3212299	23.11	0.000	6.794854	8.054053
W						
college	-.0823959	.0191586	-4.30	0.000	-.1199461	-.0448458

```
Wald test of spatial terms:          chi2(1) = 18.50      Prob > chi2 = 0.0000
```

Interpreting models with a spatial lag of the independent variables

Just as with lags of the dependent variable, the easy way to obtain the direct and indirect effects of independent variables is to use `estat impact`.

```
. estat impact
progress   :100%
Average impacts                                 Number of obs     =        254
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
college	-.077997	.0138127	-5.65	0.000	-.1050695	-.0509245
indirect						
college	-.0715273	.0166314	-4.30	0.000	-.1041243	-.0389303
total						
college	-.1495243	.0170417	-8.77	0.000	-.1829255	-.1161231

The table reports that the own-county direct effect of a 1-percentage-point increase in the college graduation rate is to reduce unemployment by 0.078 percentage points.

The across-county spillover effect of a 1-percentage-point increase in the college graduation rate is to reduce unemployment by 0.072 percentage points on average.

For those curious how the results were calculated, here are the details.

- The direct effect of college graduation rate is $\beta_1 \mathbf{x}_{\text{cr}}$.
- The indirect effect of college graduation rate is $\beta_2 \mathbf{W}\mathbf{x}_{\text{cr}}$.
- The direct effect of increasing $\mathbf{x}_{\text{cr}}$ by 1 in all counties is

$$\Delta \mathbf{y}_{\text{ue}} = \beta_1(\mathbf{x}_{\text{cr}} + \mathbf{1}) - \beta_1 \mathbf{x}_{\text{cr}} = \beta_1 \mathbf{1}$$

  where $\mathbf{1}$ is an $N \times 1$ vector of 1s.
- The direct effect is that $\mathbf{y}_{\text{ue}}$ increases by β_1 in each county.
- The indirect effect follows the same logic:

$$\Delta \mathbf{y}_{\text{ue}} = \beta_2 \mathbf{W}(\mathbf{x}_{\text{cr}} + \mathbf{1}) - \beta_2 \mathbf{W}\mathbf{x}_{\text{cr}} = \beta_2 \mathbf{W}\mathbf{1}$$

This result states that $\mathbf{y}_{\text{ue}}$ increases by $(\beta_2 \mathbf{W}\mathbf{1})_i$ in county i. For different counties, there are different effects because each county is affected by its own neighbors. The average effect across counties is the average of $\beta_2 \mathbf{W}\mathbf{1}$.

Fitting models with spatially autoregressive errors

We have fit models with a spatial lag of the dependent variable and with a spatial lag of the independent variable.

$$\mathbf{y}_{\text{ue}} = \beta_0 + \beta_1 \mathbf{x}_{\text{cr}} + \beta_2 \mathbf{W}\mathbf{y}_{\text{ue}} + \boldsymbol{\epsilon}$$
$$\mathbf{y}_{\text{ue}} = \beta_0 + \beta_1 \mathbf{x}_{\text{cr}} + \beta_2 \mathbf{W}\mathbf{x}_{\text{cr}} + \boldsymbol{\epsilon}$$

We could instead fit a model with a spatial lag of the error:

$$\mathbf{y}_{\text{ue}} = \beta_0 + \beta_1 \mathbf{x}_{\text{cr}} + (\mathbf{I} - \rho \mathbf{W})^{-1} \boldsymbol{\epsilon}$$

We do that by typing

```
. spregress unemployment college, gs2sls errorlag(W)
  (254 observations)
  (254 observations (places) used)
  (weighting matrix defines 254 places)
Estimating rho using 2SLS residuals:
initial:       GMM criterion =  .71251706
alternative:   GMM criterion =  .04381608
rescale:       GMM criterion =  .02453154
Iteration 0:   GMM criterion =  .02453154
Iteration 1:   GMM criterion =  .00420723
Iteration 2:   GMM criterion =   .0002217
Iteration 3:   GMM criterion =  .00021298
Iteration 4:   GMM criterion =  .00021298
Estimating rho using GS2SLS residuals:
Iteration 0:   GMM criterion =  .00566696
Iteration 1:   GMM criterion =  .00486118
Iteration 2:   GMM criterion =  .00486066
Iteration 3:   GMM criterion =  .00486066
Spatial autoregressive model                    Number of obs     =        254
GS2SLS estimates                                Wald chi2(1)      =      37.76
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1869
------------------------------------------------------------------------------
unemployment |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
unemployment |
     college |  -.0759125   .0123532    -6.15   0.000    -.1001243   -.0517008
       _cons |   6.292997   .2968272    21.20   0.000     5.711227    6.874768
-------------+----------------------------------------------------------------
W            |
e.unemploy~t |   .7697395   .0690499    11.15   0.000     .6344043    .9050748
------------------------------------------------------------------------------
Wald test of spatial terms:          chi2(1) = 124.27     Prob > chi2 = 0.0000
```

The estimated value of the spatial autocorrelation parameter ρ is presented on the line above the Wald test: $\widehat{\rho} = 0.77$. It is estimated to be large and significant.

ρ is called the autocorrelation parameter because it is not a correlation coefficient, although it does share some characteristics with correlation coefficients. It is theoretically bounded by -1 and 1, and $\rho = 0$ means that the autocorrelation is 0.

`estat impact` does not report ρ:

```
. estat impact
progress   :100%
Average impacts                                 Number of obs     =        254
```

	Delta-Method dy/dx	Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
college	-.0759125	.0123532	-6.15	0.000	-.1001243	-.0517008
indirect						
college	0	(omitted)				
total						
college	-.0759125	.0123532	-6.15	0.000	-.1001243	-.0517008

The above output is an example of what `estat impact` produces when there are no lagged dependent or independent variables. There are no spillover effects. Spatially correlated errors do not induce spillover effects in the covariates.

Models can have all three kinds of spatial lag terms

We have shown models with each type of spatial lag term, but models can have more than one. Use `estat impact` to estimate the effects of covariates when you have lagged variables, whether dependent, independent, or both. If you include spatially correlated errors, check the size and significance of the estimated ρ.

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spregress** — Spatial autoregressive models

[SP] **spregress postestimation** — Postestimation tools for spregress

[SP] **spset** — Declare data to be Sp spatial data

Title

Intro 8 — The Sp estimation commands

Description Remarks and examples Reference Also see

Description

There are three Sp estimation commands for spatial data:

- `spregress`—linear regression for cross-sectional data
- `spivregress`—instrumental-variables linear regression for cross-sectional data
- `spxtregress`—fixed- and random-effects linear regression models for panel data

This entry provides an overview of these estimation commands.

You may also be interested in introductions to other aspects of Sp. Below, we provide links to those other introductions.

Intro 1	A brief introduction to SAR models
Intro 2	The **W** matrix
Intro 3	Preparing data for analysis
Intro 4	Preparing data: Data with shapefiles
Intro 5	Preparing data: Data containing locations (no shapefiles)
Intro 6	Preparing data: Data without shapefiles or locations
Intro 7	Example from start to finish

Remarks and examples

Remarks are presented under the following headings:

spregress, gs2sls

`spregress` is the equivalent of `regress` for spatial data. You have two choices of estimator: `gs2sls` or `ml`.

The `gs2sls` estimator is a generalized method-of-moments estimator. With `gs2sls`, you can fit multiple spatial lags of the dependent variable (that is, multiple spatial weighting matrices), multiple spatial autoregressive error terms, and multiple spatial lags of covariates. To fit a model, you issue a command like

```
spregress y x1 x2, gs2sls dvarlag(W) errorlag(W) ivarlag(M: x1 x2)
```

where `W` and `M` are weighting matrices. See [SP] **spregress**.

To interpret your results after fitting the model, it is essential that you run `estat impact`. `estat impact` works after all the Sp estimation commands. Explanations and examples are given in [SP] **Intro 7**, example 1 of [SP] **spregress**, [SP] **spivregress postestimation**, [SP] **spregress postestimation**, and [SP] **spxtregress postestimation**.

The `gs2sls` estimator assumes that the errors are independent and identically distributed (i.i.d.) but does not require normality. The i.i.d. requirement is relaxed when you use the `heteroskedastic` option; only independence is required.

```
spregress y x1 x2, gs2sls heteroskedastic dvarlag(W) errorlag(W) ///
   ivarlag(M: x1 x2)
```

The `heteroskedastic` option uses different formulas for the spatial autoregressive error correlations and the standard errors. See *Methods and formulas* in [SP] **spregress**.

spregress, ml

The `spregress, ml` estimator is a maximum likelihood (ML) estimator. With `ml`, you can fit only one spatial lag of the dependent variable and only one spatial autoregressive error term, but you can fit multiple spatial lags of covariates. To fit a model, type

```
spregress y x1 x2, ml dvarlag(W) errorlag(W) ivarlag(W: x1 x2) ///
   ivarlag(M: x1 x2)
```

The `ml` estimator assumes that the errors are normal and i.i.d. The command `spregress, ml` is typically slower than `spregress, gs2sls`, but `spregress, ml` may be more efficient (smaller standard errors) when errors are normal.

The requirement of normality is removed if you use the `vce(robust)` option, just as it is for Stata's other ML estimators that allow this option:

```
spregress y x1 x2, ml vce(robust) dvarlag(W) errorlag(W) ///
     ivarlag(M: x1 x2)
```

See *Methods and formulas* in [SP] **spregress**.

spivregress

`spivregress` is the equivalent of `ivregress` for spatial data. `spivregress` uses the same estimator as `spregress, gs2sls`, but it allows endogenous regressors. You can fit multiple spatial lags of the dependent variable, multiple spatial autoregressive error terms, and multiple spatial lags of included exogenous regressors. You cannot specify a spatial lag for the endogenous regressors or for the excluded exogenous regressors. See *Remarks and examples* in [SP] **spivregress**.

To fit a model using `spivregress`, you would issue a command like

```
spivregress y x1 x2 (z = x3), dvarlag(W) errorlag(W) ivarlag(M: x1 x2)
```

`spivregress` also has a `heteroskedastic` option that provides the same properties it does when used with `spregress, gs2sls`.

spxtregress

`spxtregress` is the Sp estimation command for panel data. It fits fixed-effects (`fe`) and random-effects (`re`) models. `spxtregress, fe` and `re` are the spatial data equivalent of `xtreg, fe` and `re`. To use `spxtregress`, you must have strongly balanced data, and your data must be `xtset`. See [SP] **Intro 3**, [SP] **Intro 7**, and [SP] **spbalance**.

With `spxtregress, fe` and `re`, you can fit only one spatial lag of the dependent variable and only one spatial autoregressive error term. You can fit multiple spatial lags of covariates.

spxtregress, re

The random-effects model is fit using a maximum likelihood estimator. It assumes that the panel-level effects are normal i.i.d. across the panels and that the errors are normal i.i.d. across panels and time.

To fit this model, you issue a command like

```
spxtregress y x1 x2, re dvarlag(W) errorlag(W) ivarlag(M: x1 x2)
```

`spxtregress, re` has a `sarpanel` option that uses a different formulation of the random-effects estimator due to Kapoor, Kelejian, and Prucha (2007). The panel-level effects are considered a disturbance in the error equation, and the panel-level effects have the same autoregressive form as the time-level errors. To fit such models, you issue a command like

```
spxtregress y x1 x2, re sarpanel dvarlag(W) errorlag(W) ///
   ivarlag(M: x1 x2)
```

spxtregress, fe

The fixed-effects model also uses a maximum likelihood estimator. In this estimator, panel effects and effects that are constant within time are conditioned out of the likelihood. No distributional assumptions are made about the panel effects. Only covariates that vary across both panels and time can be fit with this estimator.

To fit this model, you issue a command like

```
spxtregress y x1 x2, fe dvarlag(W) errorlag(W) ivarlag(M: x1 x2)
```

See *Methods and formulas* in [SP] **spxtregress**.

Reference

Kapoor, M., H. H. Kelejian, and I. R. Prucha. 2007. Panel data models with spatially correlated error components. *Journal of Econometrics* 140: 97–130.

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spivregress** — Spatial autoregressive models with endogenous covariates

[SP] **spregress** — Spatial autoregressive models

[SP] **spxtregress** — Spatial autoregressive models for panel data

Title

estat moran — Moran's test of residual correlation with nearby residuals

Description

`estat moran` is a postestimation test that can be run after fitting a model using `regress` with spatial data. It performs the Moran test for spatial correlation among the residuals.

Quick start

Linear regression of `y` on `x1` and `x2`, then testing for spatial correlation among the residuals using the spatial weighting matrix `W`

```
regress y x1 x2
estat moran, errorlag(W)
```

After the same `regress` command, add another spatial weighting matrix

```
estat moran, errorlag(W) errorlag(M)
```

After `regress` with no independent variables

```
regress y
estat moran, errorlag(W)
```

Menu for estat

Statistics > Postestimation

Syntax

`estat moran, errorlag(`*spmatname*`) [errorlag(`*spmatname*`) ...]`

Option

`errorlag(`*spmatname*`)` specifies a spatial weighting matrix that defines the error spatial lag that will be tested. `errorlag()` is required. This option is repeatable to allow testing of higher-order error lags.

Remarks and examples

If you have not read [SP] **Intro 1**–[SP] **Intro 8**, you should do so before using `estat moran`.

To use `estat moran`, your data must be cross-sectional Sp data. See [SP] **Intro 3** for instructions on how to prepare your data.

To specify the form of the spatial correlation to be tested, you will need to have one or more spatial weighting matrices. See [SP] **Intro 2** and [SP] **spmatrix** for an explanation of the types of weighting matrices and how to create them.

Before fitting a spatial autoregressive (SAR) model with `spregress`, you may want to fit the model with `regress` and then run `estat moran`. If the Moran test is significant, you will likely want to fit the model with `spregress`. If the test is not significant, you may question the need to fit a SAR model.

`regress` can be used with a single variable before running `estat moran`. This is a test of the spatial correlation of the variable.

▷ Example 1: A test for spatial correlation

We have data on the homicide rate in counties in southern states of the U.S. `homicide1990.dta` contains `hrate`, the county-level homicide rate per year per 100,000 persons; `ln_population`, the logarithm of the county population; `ln_pdensity`, the logarithm of the population density; and `gini`, the Gini coefficient for the county, a measure of income inequality where larger values represent more inequality (Gini 1909). The data are an extract of the data originally used by Messner et al. (2000); see Britt (1994) for a literature review of the topic. This dataset is also used for the examples in [SP] **spregress**.

We used `spshape2dta` in the usual way to create the datasets `homicide1990.dta` and `homicide1990_shp.dta`. The latter file contains the boundary coordinates for U.S. southern counties. See [SP] **Intro 4**, [SP] **Intro 7**, [SP] **spshape2dta**, and [SP] **spset**.

Because the analysis dataset and the Stata-formatted shapefile must be in our working directory to `spset` the data, we first save both `homicide1990.dta` and `homicide1990_shp.dta` to our working directory by using the `copy` command. We then load the data and type `spset` to display the Sp attributes of the data.

```
. copy https://www.stata-press.com/data/r16/homicide1990.dta .
. copy https://www.stata-press.com/data/r16/homicide1990_shp.dta .
. use homicide1990
(S.Messner et al.(2000), U.S southern county homicide rates in 1990)
. spset
  Sp dataset homicide1990.dta
                data:  cross sectional
    spatial-unit id:  _ID
        coordinates:  _CX, _CY (planar)
  linked shapefile:  homicide1990_shp.dta
```

We plot the homicide rate on a map of the counties by using the `grmap` command; see [SP] **grmap**. Figure 1 is the result.

```
. grmap hrate
```

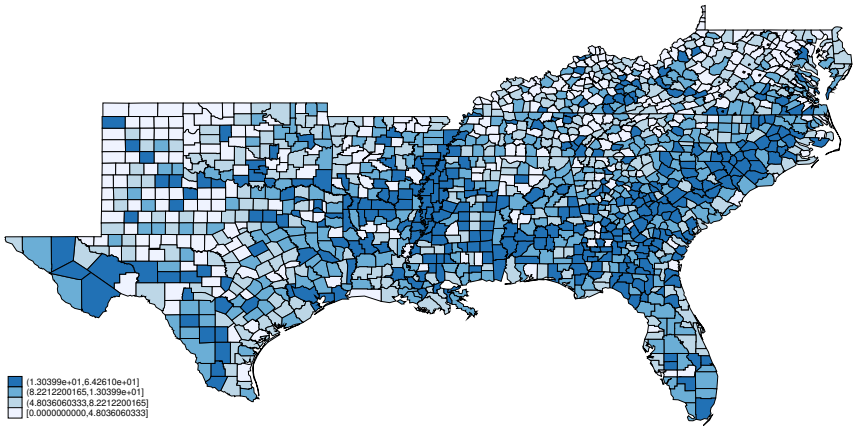


Figure 1: Homicide rate in 1990 for southern U.S. counties

The homicide rate appears to be spatially dependent because the high homicide-rate counties appear to be clustered together.

To conduct the Moran test, we need a spatial weighting matrix. We will create a contiguity matrix and use the default spectral normalization for this matrix. See [SP] **Intro 2** and [SP] **spmatrix create** for details. We type

```
. spmatrix create contiguity W
```

Now, we run `regress` and then `estat moran`:

```
. regress hrate
```

Source	SS	df	MS	Number of obs	=	1,412
				F(0, 1411)	=	0.00
Model	0	0	.	Prob > F	=	.
Residual	69908.59	1,411	49.5454217	R-squared	=	0.0000
				Adj R-squared	=	0.0000
Total	69908.59	1,411	49.5454217	Root MSE	=	7.0389

hrate	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
_cons	9.549293	.1873201	50.98	0.000	9.181837	9.916749

```
. estat moran, errorlag(W)

Moran test for spatial dependence
        Ho: error is i.i.d.
        Errorlags:  W

        chi2(1)      =   265.84
        Prob > chi2  =   0.0000
```

The test reports that we can reject that the errors are i.i.d. This is not surprising based on our visual appraisal of the data.

`estat moran` can be used with more than one weighting matrix. In this case, it produces a joint test of whether any of the weighting matrices specify a spatial dependence.

```
. spmatrix create idistance M
. estat moran, errorlag(W) errorlag(M)
Moran test for spatial dependence
        Ho: error is i.i.d.
        Errorlags:  W M
        chi2(2)      =   898.62
        Prob > chi2  =   0.0000
```

We can also use `estat moran` after a linear regression with independent variables:

```
. regress hrate ln_population ln_pdensity gini
```

Source	SS	df	MS	Number of obs	=	1,412
				F(3, 1408)	=	96.78
Model	11950.8309	3	3983.61032	Prob > F	=	0.0000
Residual	57957.7591	1,408	41.1631812	R-squared	=	0.1709
				Adj R-squared	=	0.1692
Total	69908.59	1,411	49.5454217	Root MSE	=	6.4159

hrate	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
ln_populat~n	.5559273	.2574637	2.16	0.031	.0508736	1.060981
ln_pdensity	.8231517	.2304413	3.57	0.000	.3711065	1.275197
gini	84.33136	5.169489	16.31	0.000	74.19063	94.47209
_cons	-32.46353	2.891056	-11.23	0.000	-38.13477	-26.79229

```
. estat moran, errorlag(W)
Moran test for spatial dependence
        Ho: error is i.i.d.
        Errorlags:  W
        chi2(1)      =   186.72
        Prob > chi2  =   0.0000
```

The Moran test is significant. We fit a SAR model using `spregress, gs2sls`:

```
. spregress hrate ln_population ln_pdensity gini, gs2sls errorlag(W)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrix defines 1412 places)
  (output omitted)
Spatial autoregressive model                    Number of obs     =      1,412
GS2SLS estimates                                Wald chi2(3)      =     243.84
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1686
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	.3184462	.2664379	1.20	0.232	-.2037625	.8406549
ln_pdensity	.8156068	.2469074	3.30	0.001	.3316771	1.299537
gini	88.44808	5.925536	14.93	0.000	76.83425	100.0619
_cons	-31.81189	3.115188	-10.21	0.000	-37.91755	-25.70624
W						
e.hrate	.5250879	.0326974	16.06	0.000	.4610021	.5891736

```
Wald test of spatial terms:          chi2(1) = 257.89     Prob > chi2 = 0.0000
```

See [SP] **spregress**.

Stored results

`estat moran` stores the following in `r()`:

Scalars

`r(chi2)`	χ^2
`r(df)`	degrees of freedom of χ^2
`r(p)`	p-value for model test

Macros

`r(elmat)`	weighting matrices used to specify error lag

Methods and formulas

Consider the model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{u}$$

where $\mathbf{y}$ is the $n \times 1$ dependent-variable vector, $\mathbf{X}$ is the $n \times K$ matrix of covariates, $\boldsymbol{\beta}$ is the $K \times 1$ vector of regression parameters, and $\mathbf{u}$ is the $n \times 1$ vector of disturbances. We assume that u_i are identically distributed with $E(u_i) = 0$ and $E(u_i^2) = \sigma^2$. We want to test the hypothesis that u_i are uncorrelated; that is, we want to test

$$H_0 : E(\mathbf{u}\mathbf{u}') = \sigma^2\mathbf{I}$$

Consider the case where the researcher believes that the spatial weighting matrix $\mathbf{W}_1$ gives a proper representation of spatial links for the disturbances $\mathbf{u}$. In this case, the researcher could test H_0 using the standard Moran I test statistic (Moran 1950),

$$I = \frac{\widehat{\mathbf{u}}'\mathbf{W}_1\widehat{\mathbf{u}}}{\widehat{\sigma}^2\left[\mathrm{tr}\left\{(\mathbf{W}_1' + \mathbf{W}_1)\mathbf{W}_1\right\}\right]^{1/2}}$$

where $\widehat{\mathbf{u}} = \mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}$ are the estimated residuals and $\widehat{\sigma}^2 = \widehat{\mathbf{u}}'\widehat{\mathbf{u}}/n$ is the corresponding estimator for σ^2. Under appropriate assumptions, it follows from Kelejian and Prucha (2001) that $I \sim N(0,1)$ and $I^2 \sim \chi^2(1)$.

Next, consider the case where the researcher is not sure whether any of the weighting matrices $\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_q$ properly model the spatial interdependence between u_i. In this case, the researcher can test H_0 using the $I(q)^2$ test statistic:

$$I(q)^2 = \begin{bmatrix} \widehat{\mathbf{u}}'\mathbf{W}_1\widehat{\mathbf{u}}/\widehat{\sigma}^2 \\ \vdots \\ \widehat{\mathbf{u}}'\mathbf{W}_q\widehat{\mathbf{u}}/\widehat{\sigma}^2 \end{bmatrix}' \Phi^{-1} \begin{bmatrix} \widehat{\mathbf{u}}'\mathbf{W}_1\widehat{\mathbf{u}}/\widehat{\sigma}^2 \\ \vdots \\ \widehat{\mathbf{u}}'\mathbf{W}_q\widehat{\mathbf{u}}/\widehat{\sigma}^2 \end{bmatrix}$$

where $\Phi = (\phi_{rs})$ and $r, s = 1, \ldots, q$:

$$\phi_{rs} = \frac{1}{2}\mathrm{tr}\left\{(\mathbf{W}_r + \mathbf{W}_r')(\mathbf{W}_s + \mathbf{W}_s')\right\}$$

It follows from Kelejian and Prucha (2001) and Drukker and Prucha (2013) that $I(q)^2 \sim \chi^2(q)$ under H_0.

References

Britt, C. L. 1994. Crime and unemployment among youths in the United States, 1958–1990: A time series analysis. *American Journal of Economics and Sociology* 53: 99–109.

Drukker, D. M., and I. R. Prucha. 2013. On the $I^2(q)$ test statistic for spatial dependence: Finite sample standardization and properties. *Spatial Economic Analysis* 8: 271–292.

Gini, C. 1909. Concentration and dependency ratios (in Italian). English translation in *Rivista di Politica Economica 1997* 87: 769–789.

Kelejian, H. H., and I. R. Prucha. 2001. On the asymptotic distribution of the Moran I test statistic with applications. *Journal of Econometrics* 104: 219–257.

Messner, S. F., L. Anselin, D. F. Hawkins, G. Deane, S. E. Tolnay, and R. D. Baller. 2000. An Atlas of the Spatial Patterning of County-Level Homicide, 1960–1990. Pittsburgh: National Consortium on Violence Research.

Moran, P. A. P. 1950. Notes on continuous stochastic phenomena. *Biometrika* 37: 17–23.

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spmatrix create** — Create standard weighting matrices

[SP] **spregress** — Spatial autoregressive models

[R] **regress** — Linear regression

Title

grmap — Graph choropleth maps

Description

`grmap` draws choropleth maps. Choropleth maps are maps in which shading or coloring is used to indicate values of variables within areas.

Type `help grmap` for syntax.

Quick start

A choropleth map of `x` using `spset` data

```
grmap x
```

Menu

Statistics > Spatial autoregressive models

Remarks and examples

`grmap` is lightly adapted from `spmap`, which was written by Maurizio Pisati (2007) of the Università degli Studi di Milano-Bicocca and which was preceded by his `tmap` command (2004). `grmap` differs from `spmap` in that it works with `spset` data. StataCorp expresses its gratitude to Maurizio for allowing us to use it.

References

Pisati, M. 2004. Simple thematic mapping. *Stata Journal* 4: 361–378.

——. 2007. spmap: Stata module to visualize spatial data. Statistical Software Components S456812, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s456812.html.

Also see

[SP] **spcompress** — Compress Stata-format shapefile

Title

spbalance — Make panel data strongly balanced

Description

`spbalance` reports whether panel data are strongly balanced and, optionally, makes them balanced if they are not.

The data are required to be `xtset`.

Quick start

Determine whether data are strongly balanced

```
spbalance
```

Make data strongly balanced

```
spbalance, balance
```

Menu

Statistics > Spatial autoregressive models

Syntax

Query whether data are strongly balanced

```
spbalance
```

Make data strongly balanced if they are not

```
spbalance, balance
```

Remarks and examples

Sp works with panel data but requires that they be strongly balanced. Panels are strongly balanced when each has the same number of observations and defines the same set of times. You can use `spbalance` before data are `spset` or after. Setting the data after is important because Sp data that were balanced can become unbalanced after merging additional data.

The data must be `xtset` before you can use `spbalance`:

```
. use https://www.stata-press.com/data/r16/counties
. spbalance
data not xtset
r(459);
. xtset fips time
       panel variable:  fips (unbalanced)
        time variable:  time, 1 to 5, but with a gap
                delta:  1 unit
. spbalance
  (data not strongly balanced)
    Type spbalance, balance to make the data strongly balanced by dropping
    observations.
```

Type `spbalance, balance` to make the data strongly balanced by dropping observations.

```
. spbalance, balance
  balancing data ...
    2,999 observations dropped.  Dropped was time == 3.  Data are now
    strongly balanced.
```

The dataset we started with contained data on five time periods for more than 3,000 U.S. counties. Evidently, some of the panels did not have an observation for time 3. Now, none of the panels have data on time 3. If some panels had no observations on time 4, then all observations for time 4 would have been dropped too.

Balancing by dropping spatial units

`spbalance` balances data by dropping observations for time periods that do not appear in all panels. `spbalance` does not consider the alternative of balancing by dropping spatial units, but you may want to. Here's an example.

We downloaded shapefiles for all U.S. counties in 2010. We use `spshape2dta` to create Stata Sp datasets:

```
. spshape2dta County_2010Census_DP1
  (importing .shp file)
  (importing .dbf file)
  (creating _ID spatial-unit id)
  (creating _CX coordinate)
  (creating _CY coordinate)
  file County_2010Census_DP1_shp.dta created
  file County_2010Census_DP1.dta     created
```

Our analysis dataset is `cbp05_14co.dta` consisting of U.S. Census County Business Patterns data for the years 2005–2014. We load this dataset and merge into it the Sp dataset `County_2010Census_DP1.dta` created by `spshape2dta`.

```
. copy https://www.stata-press.com/data/r16/cbp05_14co.dta .
. use cbp05_14co, clear
```

```
. merge m:1 GEOID10 using County_2010Census_DP1
    Result                           # of obs.
    -----------------------------------------
    not matched                             444
        from master                         327  (_merge==1)
        from using                          117  (_merge==2)
    matched                              31,035  (_merge==3)
    -----------------------------------------

. keep if _merge == 3
(444 observations deleted)
. drop _merge
. save cbp05_14co_census
file cbp05_14co_census.dta saved
```

We `xtset` the data and check to see if it is balanced.

```
. xtset _ID year
       panel variable:  _ID (unbalanced)
        time variable:  year, 2005 to 2014
                delta:  1 unit
. spbalance
  (data not strongly balanced)
    Type spbalance, balance to make the data strongly balanced by dropping
    observations.
```

Both `xtset` and `spbalance` tell us the same thing: the data are unbalanced. We use `spbalance, balance` to balance it.

```
. spbalance, balance
  balancing data ...
    15,515 observations dropped.  Dropped were year == 2005, 2006, 2007,
    2008, 2009.  Data are now strongly balanced.
```

What? It dropped all the years 2005–2009.

Let's go back and see what was causing the data to be unbalanced.

```
. use cbp05_14co_census, clear
. bysort _ID: gen npanel = _N
. tabulate npanel
```

npanel	Freq.	Percent	Cum.
5	5	0.02	0.02
10	31,030	99.98	100.00
Total	31,035	100.00	

Every value of `_ID` has data for 10 years except one. The one exception has data for only 5 years. We list it.

```
. list _ID state countyname year npanel if npanel != 10, noobs
```

_ID	state	countyname	year	npanel
400	ND	Slope County	2010	5
400	ND	Slope County	2011	5
400	ND	Slope County	2012	5
400	ND	Slope County	2013	5
400	ND	Slope County	2014	5

Evidently, in the 2010 Census, North Dakota got a new county named Slope County. If we drop it, our data will be balanced.

```
. drop if _ID == 400
(5 observations deleted)
. xtset _ID year
       panel variable:  _ID (strongly balanced)
        time variable:  year, 2005 to 2014
                delta:  1 unit
. spbalance
  (data strongly balanced)
```

There are consequences to this. We dropped a county in the years 2010–2014, and now there is a "hole" in the spatial map for 2010–2014. The county we dropped was part of a larger county before 2010. The spatial maps for this part of North Dakota do not match pre- and post-2010. We might not care about it and just go ahead with our analysis. Or, we might do more work to match up the spatial maps.

This is why `spbalance` always drops times. When it does that, the spatial maps are always the same for the remaining times.

Stored results

`spbalance` without the `balance` option stores the following in `r()`:

Scalars

`r(balanced)`	`1` if strongly balanced, `0` otherwise

`spbalance, balance` stores the following in `r()`:

Scalars

`r(balanced)`	`1`
`r(Ndropped)`	number of observations dropped

Matrices

`r(T)`	$1 \times$ `r(Ndropped)` vector of the times dropped if `r(Ndropped)` > 0

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spset** — Declare data to be Sp spatial data

[SP] **spregress** — Spatial autoregressive models

[SP] **spxtregress** — Spatial autoregressive models for panel data

[XT] **xtset** — Declare data to be panel data

Title

spcompress — Compress Stata-format shapefile

Description

`spcompress` creates a new Stata-format shapefile omitting places (geographical units) that do not appear in the Sp data in memory. The new shapefile will be named after the data in memory.

Quick start

Create new file `new_shp.dta` containing only cases identified by `mysample` from `old_shp.dta`

```
use old
keep if mysample
save new
spcompress
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spcompress [, force]
```

Option

`force` allows replacing an existing shapefile. `force` is the option name StataCorp uses when you should think twice before specifying it. In most cases, you want to create a new shapefile.

Remarks and examples

Remarks are presented under the following headings:

Introduction

In [SP] **Intro 4** and [SP] **Intro 7**, we discussed how to find and prepare the analysis dataset, `tl_2016_us_county.dta`, and the shapefile dataset, `tl_2016_us_county_shp.dta`. We again use those datasets here.

You sometimes want to analyze a subset of the data. In those cases, you might type

```
. use tl_2016_us_county            // use all the data
. keep if STATEFP == "48"          // keep the subset of interest
. save texas                       // save under a different name
```

All will work fine. File `texas.dta` is linked to `tl_2016_us_county_shp.dta`, which contains a lot of unnecessary information, but that will cause Sp no difficulty.

Next, you can type

```
. spcompress
```

Now, files `tl_2016_us_county.dta` and `tl_2016_us_county_shp.dta` remain unchanged, and file `texas_shp.dta` was created. `texas.dta` was resaved so that the copy on disk would reflect that it is now linked to `texas_shp.dta` instead of `tl_2016_us_county_shp.dta`.

Sp will run a little faster if we compress the shapefile. We say a little because only `grmap` will run faster.

Using the force option

Above, we showed an example. Here is what would have happened had we omitted the line `save texas`:

```
. use tl_2016_us_county
. keep if STATEFP == "48"
(2,979 observations deleted)
. * save texas              // save texas intentionally commented out
. spcompress
file tl_2016_us_county_shp.dta already exists
r(602);
```

Whether you type `save texas` makes all the difference. Do you really want to replace `tl_2016_us_county_shp.dta`? If so, specify `force`.

The option is called `force` because Stata wonders whether you really meant to type

```
. use tl_2016_us_county, clear
. keep if STATEFP == "48"
(2,979 observations deleted)
. save texas
file texas.dta saved
. spcompress
  (texas_shp.dta created with 254 spatial units, 2,979 fewer than previously)
  (texas_shp.dta saved)
  (texas.dta saved)
```

Even if you intended to discard all but Texas from `tl_2016_us_county.dta` and `tl_2016_us_county_shp.dta`, we would recommend that you type

```
. use tl_2016_us_county
. keep if STATEFP == "48"
. save texas
. spcompress
. erase tl_2016_us_county.dta
. erase tl_2016_us_county_shp.dta
```

Stored results

`spcompress` stores the following in `r()`:

Scalars

`r(num_drop_ids)`	# of spatial units dropped
`r(num_ids)`	# of spatial units remaining

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[D] **compress** — Compress data in memory

Title

spdistance — Calculator for distance between places

Description

`spdistance` $\#_1$ $\#_2$ reports the distance between the areas `_ID` $= \#_1$ and `_ID` $= \#_2$.

Quick start

Obtain distance between `_ID` $= 48201$ and `_ID` $= 48041$

```
spdistance 48201 48041
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spdistance #1 #2
```

$\#_1$ and $\#_2$ are two `_ID` values.

Remarks and examples

Remarks are presented under the following headings:

Are coordinates really planar and not latitude and longitude?

The purpose of `spdistance` is to help in understanding the units in which distances are measured when coordinates are recorded in planar units. Before turning to that issue, however, let us ask another question: Are the coordinates recorded in your data really planar? Sp assumes that they are. It is your responsibility to change a setting if they are in fact degrees latitude and longitude. You change the setting by typing

```
. spset, modify coordsys(latlong, kilometers)
```

or

```
. spset, modify coordsys(latlong, miles)
```

If coordinates are latitude and longitude, type one of those commands and then distances will be reported in kilometers or miles and you can dispense with `spdistance` for determining units.

Why does Sp assume that coordinates are planar when they might be latitude and longitude? How can you tell whether your data contain latitudes and longitudes?

Sp assumes that coordinates are planar because coordinates obtained from shapefiles are supposed to be planar. Usage is running ahead of standards, however, and these days many shapefile providers are providing latitude and longitude.

Before answering the second question, let us answer a third question you may be asking yourself: "Do I care? How bad would it be to treat degrees as if they were planar?" If all the locations in your data are near the equator, there is no reason you should care. But this is not likely to be the case, and because degrees longitude are not a fixed distance, your calculations will be incorrect if you treat degrees as if they were planar. See *Latitude and longitude coordinates* below for more details.

So how do you tell the units of measure? The documentation for your shapefile may tell you. If not, you can inspect the data. Sp datasets record the coordinates in variables `_CX` and `_CY`. You look at those variables and compare them with latitudes and longitudes for the same places or nearby places, which you can easily find on the web. If coordinates are latitude and longitude, then

- `_CX` will be the longitude value
- `_CY` will be the latitude value

Reverse engineering planar distances

Planar coordinates have no predetermined scale. Two places might be 5 apart. How far is that? One way to find out is to reverse engineer the scale. Take two places that you know the distance between, use `spdistance` to obtain the planar distance, and divide.

For instance, we have a city dataset in which Los Angeles and New York have `_ID` values 1 and 79. Using `spdistance`, we obtain the distance between them.

```
. spdistance 1 79
  (data currently use planar coordinates)
       _ID       (x, y) (planar)
------------------------------------------
         1       (0, 0)
        79       (4.97, 1)
------------------------------------------
  distance       5.0696053 planar units
```

The distance between the cities is roughly 2,400 miles, so we know that one planar unit equals $2400/5.07 \approx 473$ miles.

More than you want to know about coordinates

Planar coordinates

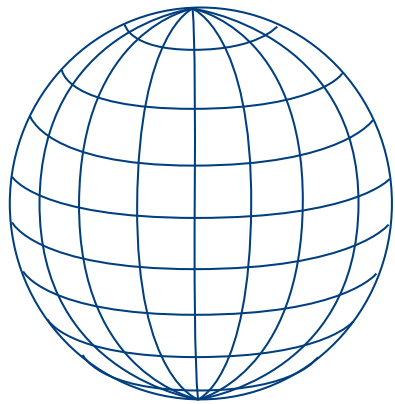
Planar coordinates, also known as rectangular coordinates, are coordinates on a plane. The formula for the distance between two places (x_1, y_1) and (x_2, y_2) is given by the Euclidean distance formula:

$$\text{distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Distance will be measured in miles if x and y are measured in miles.

Latitude and longitude coordinates

Latitude and longitude are measured in degrees on a globe. The common illustration looks like this:



The vertical lines passing through the poles are lines of equal longitude. Longitude indicates which vertical line you are on. the vertical lines. It measures which vertical line you are on. Longitude is an east–west measure. Examples of longitude include 150° east and 96° west, with east and west referring to east and west of Greenwich, UK.

The horizontal rings are lines of equal latitude. Different horizontal lines are different latitudes. Latitude is a north–south measure. Examples of latitude include 30° north and 33° south, with north and south referring to north and south of the equator.

College Station, USA, and Sydney, Australia, are located at

City	Latitude	Longitude
College Station	30°36′05″ N	96°18′52″ W
Sydney, Australia	33°51′54″ S	151°12′34″ E

Computers use signed values to indicate direction and fractions of degrees instead of minutes and seconds. The above table can equivalently be written as

City	Latitude	Longitude
College Station	30.601	−96.314
Sydney, Australia	−33.865	151.209

If 1° equaled a fixed distance, we could use the Euclidean formula for calculating the distance between College Station and Sydney.

One degree of latitude does equal a fixed distance, namely, 69 miles, if the Earth were a sphere.

One degree of longitude, however, measures a distance that varies from 69 miles at the equator to 0 miles at the North and South Poles:

At latitude	1° longitude equals	
±0	69 miles	(equator)
±10	68	
±20	65	
±30	60	
±40	53	
±50	44	
±60	35	
±70	24	
±80	12	
±89	1	
±90	0	(N or S pole)

There are formulas for calculating distances from latitude and longitude, and Sp will use them if you tell it that the coordinates are degrees latitude and longitude. If you do not, Sp makes calculations using the Euclidean formula, and that will result in incorrect distances except at the equator. At the equator, 1° longitude equals 1° latitude equals 69 miles. The farther north or south you make the calculation, the more will be the error. East–west distances will be exaggerated relative to north–south distances, resulting in places being calculated as being farther apart than they really are.

The 48 contiguous states of the United States lies between 25° and 50° latitude, a region in which 1° longitude varies between 66 and 44 miles.

Europe lies between 35° and 70° latitude, a region in which 1° longitude varies between 56 and 24 miles.

Stored results

`spdistance` stores the following in `r()`:

Scalars
- `r(dist)` distance between

Macros
- `r(coordsys)` `planar` or `latlong`
- `r(dunits)` `miles` or `kilometers` if `r(coordsys) = latlong`

Methods and formulas

If coordinates are planar, the distance between (x_1, y_1) and (x_2, y_2) is

$$d = \sqrt{(x_2 - x_1)^2 + (y_1 - y_1)^2}$$

If coordinates are latitude and longitude, let (t_1, n_1) and (t_2, n_2) be the two coordinate pairs, where t represents latitude and n represents longitude converted from degrees to radians.

Let $\Delta t = (t_2 - t_1)$ and $\Delta n = (n_2 - n_1)$. Then the distance between the two points is

$$d = r \text{ invhav}\{r \text{ hav}(\Delta t) + \cos t_1 \cos t_2 \text{ hav}(\Delta n)\}$$

where r is the radius of the Earth measured in the desired units (miles or kilometers) and

$$\text{hav}(\theta) = \frac{1 - \cos \theta}{2}$$

$$\text{invhav}(h) = 2 \text{ asin}(\sqrt{h})$$

Reference

Weber, S., and M. Péclat. 2017. A simple command to calculate travel distance and travel time. *Stata Journal* 17: 962–971.

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spset** — Declare data to be Sp spatial data

Title

spgenerate — Generate variables containing spatial lags

Description

`spgenerate` creates new variables containing $\mathbf{Wx}$. These are the same spatial lag variables that you include in models that you fit with the Sp estimation commands.

Quick start

Create variable `x_nearby` equal to `Wc*x`, the spatial lag of `x` using spatial weighting matrix `Wc`

```
spgenerate x_nearby = Wc*x
```

Menu

Statistics > Spatial autoregressive models

Syntax

`spgenerate` [*type*] *newvar* `=` *spmatname*`*`*varname* [*if*] [*in*]

Remarks and examples

Remarks are presented under the following headings:

Use with Sp data
Use with other datasets

Use with Sp data

The $\mathbf{Wx}$ variables that `spgenerate` creates are literally the variables that the Sp estimation commands include in the models when $\mathbf{x}$ is not the dependent variable. Nonetheless, do not type

```
. spmatrix create contiguity W
. spgenerate Wcollege = W*college
. spregress unemployment college Wcollege, gs2sls
```

Instead, type

```
. spmatrix create contiguity W
. spregress unemployment college, gs2sls ivarlag(W:college)
```

spregress will report the same result either way because college is an exogenous variable. But some postestimation commands will produce incorrect results because they will not know that Wcollege is W*college.

You can use Wcollege after fitting models, however, to better understand results.

In an example in *Fitting models with a spatial lag of independent variables* of [SP] **Intro 7**, we fit the model

```
. use texas_ue
. spmatrix create contiguity W
. spregress unemployment college, gs2sls ivarlag(W:college)
  (254 observations)
  (254 observations (places) used)
  (weighting matrix defines 254 places)
Spatial autoregressive model                    Number of obs     =        254
GS2SLS estimates                                Wald chi2(2)      =      81.13
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.2421
------------------------------------------------------------------------------
unemployment |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
unemployment |
     college |   -.077997   .0138127    -5.65   0.000    -.1050695   -.0509245
       _cons |   7.424453   .3212299    23.11   0.000     6.794854    8.054053
-------------+----------------------------------------------------------------
W            |
     college |  -.0823959   .0191586    -4.30   0.000    -.1199461   -.0448458
------------------------------------------------------------------------------
Wald test of spatial terms:          chi2(1) = 18.50      Prob > chi2 = 0.0000
```

Matrix W is the contiguity matrix for first-order neighbors.

If W*college is something of a mystery to you, you can use spgenerate to create the variable and explore it. Type

```
. spgenerate Wcollege = W*college
```

In this example, variables college and Wcollege have similar summary statistics. They usually do.

```
. summarize unemployment college Wcollege
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
unemployment |        254    4.731102    1.716514         1.5       12.4
     college |        254    17.95906    7.355919         2.6       49.4
    Wcollege |        254    15.68765    5.303385    1.279117   36.43961
```

It turns out that variables college and Wcollege have a surprisingly low correlation, which is not typical:

```
. correlate unemployment college Wcollege
(obs=254)
             | unempl~t  college Wcollege
-------------+---------------------------
unemployment |   1.0000
     college |  -0.4323   1.0000
    Wcollege |  -0.3833   0.3852   1.0000
```

You can use Wcollege to assess practical significance. We know from the regression output that the coefficient on W*college is −0.0824 and statistically significant. Is −0.0824 practically significant? From the summarize output, we know that the mean of Wcollege is 15.69. Thus at its average, W*college is contributing $-0.0824 \times 15.69 = -1.29$ to unemployment, which itself has mean 4.73.

Use with other datasets

Consider another analysis that has nothing to do with the spatial analyses discussed in this manual. You are fitting a logistic regression model using outcome.dta. The dataset contains observations on thousands of people whom you call subjects. It has lots of variables, too, among which is fips, the county code in which each subject resides. You want to include the county unemployment rate as an exogenous variable in your model, but outcome.dta does not have that variable.

Obtaining unemployment would be easy enough if you had another dataset containing it, and you do. You have ue_texas.dta, the Sp dataset you used to fit the spatial model above. It is irrelevant that the dataset is spatial; you just want to borrow its county unemployment variable. You could type

```
. use texas_ue, clear
. keep fips unemployment
. save unemploymentvar

. use outcome, clear
. sort fips
. merge m:1 fips using unemploymentvar, keep(master)

. erase unemploymentvar.dta

. logistic outcome ... unemployment ...
```

You had to perform an m:1 merge because outcome.dta might contain multiple subjects living in the same county. You had to keep(master) because there might be some counties in which no one in the data lived. None of that bothers you—you just want the unemployment for the county in which each subject resides, and now you have it, and you fit your model.

What you may not know is that you can include spatial lags of unemployment as an exogenous variable in your logistic model and be on firm statistical ground. A spatial lag is W*unemployment, and W is fixed and unemployment is exogenous in your logistic model. To do that, you would type

```
. use texas_ue, clear
. spmatrix create contiguity W
. spgenerate Wunemployment = W*unemployment
. keep fips unemployment Wunemployment
. save unemploymentvar

. use outcome, clear
. sort fips
. merge m:1 fips using unemploymentvar, keep(master)

. erase unemploymentvar.dta

. logistic outcome ... unemployment Wunemployment ...
```

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spmatrix create** — Create standard weighting matrices

[SP] **spregress** — Spatial autoregressive models

Title

spivregress — Spatial autoregressive models with endogenous covariates

Description

`spivregress` is the equivalent of `ivregress` for spatial data. `spivregress` fits spatial autoregressive (SAR) models, also known as simultaneous autoregressive models, where the models may contain additional endogenous variables as well as exogenous variables. These models can be used to account for possible dependence between the outcome variable and the unobserved errors.

For models without endogenous regressors, see [SP] **spregress**.

If you have not read [SP] **Intro 1**–[SP] **Intro 8**, you should do so before using `spivregress`. Your data must be Sp data to use `spivregress`. See [SP] **Intro 3** for instructions on how to prepare your data.

To specify spatial lags, you will need to have one or more spatial weighting matrices. See [SP] **Intro 2** and [SP] **spmatrix** for an explanation of the types of weighting matrices and how to create them.

Quick start

Spatial autoregressive model of `y1` regressed on `x1`, `x2`, endogenous regressor `y2`, which uses `z1` as an instrument, and a spatial lag for `y1` specified by the weighting matrix `W`

```
spivregress y1 x1 x2 (y2 = z1), dvarlag(W)
```

Add an autoregressive error term with the lag given by `M`

```
spivregress y1 x1 x2 (y2 = z1), dvarlag(W) errorlag(M)
```

Add a spatial lag for the exogenous variable `x1` based on `W`

```
spivregress y1 x1 x2 (y2 = z1), dvarlag(W) errorlag(M) ivarlag(W: x1)
```

Add a second spatial lag for the outcome variable based on the weighting matrix `M`

```
spivregress y1 x1 x2 (y2 = z1), dvarlag(W) errorlag(M)              ///
     dvarlag(M) ivarlag(W: x1)
```

Add interaction between `x1` and `x2` and add categorical instrument `z2` using factor variable notation

```
spivregress y1 x1 x2 c.x1#c.x2 (y2 = z1 i.z2), dvarlag(W)          ///
     errorlag(M) dvarlag(M) ivarlag(W: x1 x2 c.x1#c.x2)
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spivregress depvar [varlist1] (varlist2 = varlistiv) [if] [in] [, options]
```

$varlist_1$ is the list of included exogenous regressors.

$varlist_2$ is the list of endogenous regressors.

$varlist_{iv}$ is the list of excluded exogenous regressors used with $varlist_1$ as instruments for $varlist_2$.

options	Description
Model	
`dvarlag(`*spmatname*`)`	spatially lagged dependent variable; repeatable
`errorlag(`*spmatname*`)`	spatially lagged errors; repeatable
`ivarlag(`*spmatname* `:` *varlist*`)`	spatially lagged exogenous variables from $varlist_1$; repeatable
`noconstant`	suppress constant term
`heteroskedastic`	treat errors as heteroskedastic
`force`	allow estimation when estimation sample is a subset of the sample used to create the spatial weighting matrix
`impower(#)`	order of instrumental-variable approximation
Reporting	
`level(#)`	set confidence level; default is `level(95)`
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Optimization	
optimization_options	control the optimization process; seldom used
`coeflegend`	display legend instead of statistics

$varlist_1$, $varlist_2$, $varlist_{iv}$, and *varlist* specified in `ivarlag()` may contain factor variables; see [U] **11.4.3 Factor variables**.

`coeflegend` does not appear in the dialog box.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options

Model

`dvarlag(`*spmatname*`)` specifies a spatial weighting matrix that defines a spatial lag of the dependent variable. This option is repeatable to allow higher-order models. By default, no spatial lags of the dependent variable are included.

`errorlag(`*spmatname*`)` specifies a spatial weighting matrix that defines a spatially lagged error. This option is repeatable to allow higher-order models. By default, no spatially lagged errors are included.

`ivarlag(`*spmatname* `:` *varlist*`)` specifies a spatial weighting matrix and a list of exogenous variables that define spatial lags of the variables. The variables in *varlist* must be a subset of the exogenous variables in $varlist_1$. This option is repeatable to allow spatial lags created from different matrices. By default, no spatial lags of the exogenous variables are included.

`noconstant`; see [R] **Estimation options**.

`heteroskedastic` specifies that the estimator treat the errors as heteroskedastic instead of homoskedastic, which is the default; see *Methods and formulas* in [SP] **spregress**.

`force` requests that estimation be done when the estimation sample is a proper subset of the sample used to create the spatial weighting matrices. The default is to refuse to fit the model. Weighting matrices potentially connect all the spatial units. When the estimation sample is a subset of this space, the spatial connections differ and spillover effects can be altered. In addition, the normalization of the weighting matrix differs from what it would have been had the matrix been normalized over the estimation sample. The better alternative to `force` is first to understand the spatial space of the estimation sample and, if it is sensible, then create new weighting matrices for it. See [SP] **spmatrix** and *Missing values, dropped observations, and the W matrix* in [SP] **Intro 2**.

`impower(#)` specifies the order of an instrumental-variable approximation used in fitting the model. The derivation of the estimator involves a product of # matrices. Increasing # may improve the precision of the estimation and will not cause harm, but will require more computer time. The default is `impower(2)`. See *Methods and formulas* for additional details on `impower(#)`.

Reporting

`level(#)`; see [R] **Estimation options**.

display_options: `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(#)`, `fvwrapon(`*style*`)`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

Optimization

optimization_options: `iterate(#)`, `[no]log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(#)`, `ltolerance(#)`, `nrtolerance(#)`, and `nonrtolerance`; see [M-5] **optimize( )**.

The following option is available with `spivregress` but is not shown in the dialog box:

`coeflegend`; see [R] **Estimation options**.

Remarks and examples

See [SP] **Intro** for an overview of SAR models.

`spivregress` fits spatial autoregressive models that include endogenous regressors. The `spivregress` command is for use with cross-sectional data. It requires each observation to represent one unique spatial unit. See [SP] **Intro 3** and the introductory sections that follow for instructions with examples on how to prepare your data for analysis with `spivregress`.

`spivregress` fits models like the following:

```
spivregress y1 x1 x2 (y2 y3 = z1 z2 z3), dvarlag(W) errorlag(M) ///
     ivarlag(W: x1)
```

`dvarlag(W)` specifies a spatial lag of the dependent variable `y1`, with the formulation of the lag given by the spatial weighting matrix `W`. You can include multiple `dvarlag()` options, each with different weighting matrices, to model higher-order spatial lags of the dependent variable.

`errorlag(M)` specifies an autoregressive error term based on the weighting matrix `M`. You can include multiple `errorlag()` options.

`ivarlag(W: x1)` specifies a spatial lag of the exogenous variable `x1`. You cannot include in the model spatial lags of the endogenous regressors `y2` and `y3` or spatial lags of the excluded exogenous regressors `z1`, `z2`, and `z3`.

`spivregress` uses a generalized method of moments estimator known as generalized spatial two-stage least squares (GS2SLS), the same estimator used by `spregress, gs2sls`. See *Methods and formulas*. Also see *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about the GS2SLS estimator.

▷ Example 1: SAR models with endogenous regressors

Suppose we want to know whether prohibiting alcohol sales in a county decreases the rate of arrests for driving under the influence (DUI). We use the artificial dataset `dui_southern.dta`, containing DUI rates in counties in southern states of the United States.

Because the analysis dataset and the Stata-formatted shapefile must be in our working directory to `spset` the data, we first save both `dui_southern.dta` and `dui_southern_shp.dta` to our working directory by using the `copy` command. We then load the data and type `spset` to see the Sp settings.

```
. copy https://www.stata-press.com/data/r16/dui_southern.dta .
. copy https://www.stata-press.com/data/r16/dui_southern_shp.dta .
. use dui_southern
. spset
  Sp dataset dui_southern.dta
                data:  cross sectional
   spatial-unit id:  _ID
       coordinates:  _CX, _CY (planar)
  linked shapefile:  dui_southern_shp.dta
```

The outcome of interest is `dui`, which is the alcohol-related arrest rate per 100,000 daily vehicle miles traveled (DVMT). Explanatory variables include `police`, the number of sworn officers per 100,000 DVMT; `nondui`, the nonalcohol-related arrest rate per 100,000 DVMT; `vehicles`, the number of registered vehicles per 1,000 residents; and `dry`, a variable that indicates whether a county prohibits the sale of alcohol within its borders.

Because the size of the police force may be a function of `dui` and `nondui` arrest rates, we treat `police` as endogenous. We assume the variable `election` is a valid instrument, where `election` is 1 if the county government faces an election and is 0 otherwise.

We believe the DUI arrest rate to be spatially correlated, with the rate in a county affecting the rates in neighboring counties. Formally, the model we want to fit is

$$\texttt{dui} = \beta_0 + \beta_1 \times \texttt{nondui} + \beta_2 \times \texttt{dry} + \beta_3 \times \texttt{vehicles} + \pi_1 \times \texttt{police} + \lambda \mathbf{W} \times \texttt{dui} + \mathbf{u}$$

$$\mathbf{u} = \rho \mathbf{W}\,\mathbf{u} + \boldsymbol{\epsilon}$$

The term $\mathbf{W} \times$ `dui` defines a spatial lag of `dui`. See [SP] **Intro 2** for an explanation of how spatial lags are defined by weighting matrices, and see *Choosing weighting matrices and their normalization* in [SP] **spregress**. The equation for $\mathbf{u}$ gives the error an autoregressive form also specified by the weighting matrix $\mathbf{W}$. The variable `police` is endogenous and may be correlated with the error $\mathbf{u}$. We instrument it with the variable `election`. See *Methods and formulas* for how the endogeneity of `police` is handled by the estimator.

Before we can fit the model, we must create the weighting matrix $\mathbf{W}$. We will create one that puts the same positive weight on contiguous counties and a 0 weight on all other counties—a matrix known as a contiguity matrix. We will use the default spectral normalization for the matrix. See [SP] **Intro 2** and [SP] **spmatrix create** for details. We type

```
. spmatrix create contiguity W
```

We fit the model by typing

```
. spivregress dui nondui vehicles i.dry (police = elect), dvarlag(W) errorlag(W)
  (1422 observations)
  (1422 observations (places) used)
  (weighting matrix defines 1422 places)
Estimating rho using 2SLS residuals:
initial:       GMM criterion =  .00254902
alternative:   GMM criterion =  .00377532
rescale:       GMM criterion =  .00009468
Iteration 0:   GMM criterion =  .00009468
Iteration 1:   GMM criterion =  .00001513
Iteration 2:   GMM criterion =  .00001512
Estimating rho using GS2SLS residuals:
Iteration 0:   GMM criterion =  .00086665
Iteration 1:   GMM criterion =  .00085487
Iteration 2:   GMM criterion =  .00085486
Spatial autoregressive model                    Number of obs     =      1,422
GS2SLS estimates                                Wald chi2(5)      =    4393.21
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.7378
```

dui	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
dui						
police	-1.283189	.1138994	-11.27	0.000	-1.506428	-1.059951
nondui	-.001833	.0025467	-0.72	0.472	-.0068245	.0031585
vehicles	.0906069	.0045059	20.11	0.000	.0817755	.0994384
dry						
Yes	.4631025	.076754	6.03	0.000	.3126674	.6135377
_cons	8.714745	1.060428	8.22	0.000	6.636345	10.79315
W						
dui	.3859225	.0194397	19.85	0.000	.3478214	.4240235
e.dui	.2169234	.0496595	4.37	0.000	.1195926	.3142541

```
Wald test of spatial terms:          chi2(2) = 408.78     Prob > chi2 = 0.0000
Instrumented:     police (W*dui)
Raw instruments:  nondui vehicles 1.dry election dui:_cons
```

When a spatial lag of the dependent variable is included in the model, covariates have both direct and indirect effects. See example 1 of [SP] **spregress** for a discussion. To obtain the direct, indirect, and total effects of the covariates, we must use `estat impact`:

```
. estat impact
progress   : 20%  40%  60%  80% 100%
Average impacts                                 Number of obs     =      1,422
```

	Delta-Method dy/dx	Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
police	-1.313426	.1198948	-10.95	0.000	-1.548416	-1.078437
nondui	-.0018762	.0026073	-0.72	0.472	-.0069864	.003234
vehicles	.092742	.0048427	19.15	0.000	.0832504	.1022336
dry						
Yes	.4740151	.0788695	6.01	0.000	.3194336	.6285966
indirect						
police	-.6465736	.1063216	-6.08	0.000	-.8549601	-.4381871
nondui	-.0009236	.0012928	-0.71	0.475	-.0034576	.0016103
vehicles	.045655	.0057216	7.98	0.000	.0344409	.0568692
dry						
Yes	.2333482	.0464145	5.03	0.000	.1423774	.3243189
total						
police	-1.96	.2258604	-8.68	0.000	-2.402678	-1.517322
nondui	-.0027998	.0038989	-0.72	0.473	-.0104416	.0048419
vehicles	.138397	.0105248	13.15	0.000	.1177688	.1590253
dry						
Yes	.7073633	.123289	5.74	0.000	.4657213	.9490052

While it is running, `estat impact` prints percentages at the top of the output to indicate progress. Calculation of the standard errors of the effects can be intensive and take time, so it reports its progress as it does the computations.

The average direct, or own-county, effect of going from a wet county to a dry county on alcohol-related arrest rates is positive. The average indirect, or spillover, effect of going from a wet county to a dry county on alcohol-related arrest rates is also positive. The total effects are the sum of the direct and indirect effects, so these are also positive.

◁

▷ Example 2: SAR models with endogenous regressors and covariate lags

Continuing with example 1, we found that dry, we now add a spatial lag of the covariate dry.

```
. spivregress dui nondui vehicles i.dry (police = elect), dvarlag(W)
> errorlag(W) ivarlag(W: i.dry)
  (1422 observations)
  (1422 observations (places) used)
  (weighting matrix defines 1422 places)
note: exog*W:0b.dry omitted because of collinearity
```

(*output omitted*)

```
Spatial autoregressive model                    Number of obs     =      1,422
GS2SLS estimates                                Wald chi2(6)      =    4300.29
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.7337
```

dui	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
dui						
police	-1.301634	.1155866	-11.26	0.000	-1.52818	-1.075089
nondui	-.0018725	.0025746	-0.73	0.467	-.0069187	.0031737
vehicles	.091364	.0045754	19.97	0.000	.0823965	.1003316
dry						
Yes	.4754855	.078153	6.08	0.000	.3223085	.6286626
_cons	8.853401	1.07409	8.24	0.000	6.748223	10.95858
W						
dry						
Yes	.2868458	.2209814	1.30	0.194	-.1462697	.7199613
dui	.38758	.0196366	19.74	0.000	.349093	.4260669
e.dui	.2196418	.0497708	4.41	0.000	.1220929	.3171908

```
Wald test of spatial terms:          chi2(3) = 405.90     Prob > chi2 = 0.0000
Instrumented:     police (W*dui)
Raw instruments:  nondui vehicles 1.dry election (W*0b.dry) (W*1.dry)
                  dui:_cons
```

We use estat impact to see the effects:

```
. estat impact
progress   : 20%  40%  60%  80% 100%
Average impacts                                 Number of obs     =      1,422
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
police	-1.332603	.1217453	-10.95	0.000	-1.571219	-1.093986
nondui	-.001917	.0026364	-0.73	0.467	-.0070844	.0032503
vehicles	.0935378	.0049201	19.01	0.000	.0838945	.1031811
dry						
Yes	.5044067	.0833742	6.05	0.000	.3409963	.667817
indirect						
police	-.6601862	.1089584	-6.06	0.000	-.8737408	-.4466316
nondui	-.0009497	.0013158	-0.72	0.470	-.0035287	.0016293
vehicles	.0463396	.0058501	7.92	0.000	.0348737	.0578055
dry						
Yes	.6165397	.3004056	2.05	0.040	.0277555	1.205324
total						
police	-1.992789	.2303197	-8.65	0.000	-2.444207	-1.541371
nondui	-.0028668	.003951	-0.73	0.468	-.0106106	.0048771
vehicles	.1398774	.0107284	13.04	0.000	.1188501	.1609047
dry						
Yes	1.120946	.3442805	3.26	0.001	.446169	1.795724

The direct effect of dry is little changed when we added a lag of dry, going from 0.47 to 0.50. But the indirect effects of dry go from 0.23 to 0.62. In these fictional data, the indirect effects of dry become larger than the direct effects when there is a lag of dry in the model.

Note that spivregress does not allow the fitting of spatial lags for police, our endogenous regressor, nor for election, its instrument.

◁

▷ Example 3: SAR models with endogenous regressors and higher-order lags

In the previous models, we specified all the spatial lags with a single weighting matrix W, a contiguity weighting matrix with the default spectral normalization. Many researchers use a spatial weighting matrix whose (i, j)th element is the inverse of the distance between units i and j. With the GS2SLS estimator used by spivregress, we can include spatial lags using two spatial weighting matrices. This can be done to model a "higher-order" approximation to the true spatial process. We will now add lags specified by an inverse-distance matrix, using again a spectral normalization of the matrix.

We create the inverse-distance matrix M and use spmatrix dir to list our Sp matrices.

```
. spmatrix create idistance M
. spmatrix dir
```

Weighting matrix name	N x N	Type	Normalization
M	1422 x 1422	idistance	spectral
W	1422 x 1422	contiguity	spectral

We fit the model including both weighting matrices for all the lags:

```
. spivregress dui nondui vehicles i.dry (police = elect), dvarlag(W)
> errorlag(W) ivarlag(W: i.dry) dvarlag(M) errorlag(M) ivarlag(M: i.dry)
  (1422 observations)
  (1422 observations (places) used)
  (weighting matrices define 1422 places)
note: exog*W:0b.dry omitted because of collinearity
note: exog*M:0b.dry omitted because of collinearity
```

(*output omitted*)

```
Spatial autoregressive model                    Number of obs     =      1,422
GS2SLS estimates                                Wald chi2(8)      =    6447.62
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.8058
```

dui	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
dui						
police	-.9762244	.0782512	-12.48	0.000	-1.129594	-.8228549
nondui	-.0010538	.002093	-0.50	0.615	-.005156	.0030483
vehicles	.0786503	.0031164	25.24	0.000	.0725423	.0847582
dry						
Yes	.4207535	.0631503	6.66	0.000	.2969811	.5445258
_cons	6.067724	.7490414	8.10	0.000	4.59963	7.535818
W						
dry						
Yes	.2353895	.2272276	1.04	0.300	-.2099684	.6807474
dui	.3335312	.0134259	24.84	0.000	.3072169	.3598455
e.dui	.2206942	.0630468	3.50	0.000	.0971248	.3442636
M						
dry						
Yes	-.0923513	2.70903	-0.03	0.973	-5.401952	5.217249
dui	.0005204	.0112677	0.05	0.963	-.0215639	.0226046
e.dui	-.1069363	.5910148	-0.18	0.856	-1.265304	1.051431

```
Wald test of spatial terms:          chi2(6) = 649.11     Prob > chi2 = 0.0000
Instrumented:     police (W*dui) (M*dui)
Raw instruments:  nondui vehicles 1.dry election (W*0b.dry) (W*1.dry)
                  (M*0b.dry) (M*1.dry) dui:_cons
```

All the spatial lags specified by the inverse-distance matrix M are nonsignificant. We conclude that there are no inverse-distance-type effects after we account for contiguity-type effects.

◁

Stored results

`spivregress` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k)`	number of parameters
`e(df_m)`	model degrees of freedom
`e(df_c)`	degrees of freedom for comparison test
`e(iterations)`	number of generalized method of moments iterations
`e(iterations_2sls)`	number of two-stage least-squares iterations
`e(rank)`	rank of `e(V)`
`e(r2_p)`	pseudo-R^2
`e(chi2)`	χ^2
`e(chi2_c)`	χ^2 for comparison test
`e(p)`	p-value for model test
`e(p_c)`	p-value for test of spatial terms
`e(converged)`	`1` if generalized method of moments converged, `0` otherwise
`e(converged_2sls)`	`1` if two-stage least-squares converged, `0` otherwise

Macros

`e(cmd)`	`spivregress`
`e(cmdline)`	command as typed
`e(depvar)`	name of dependent variable
`e(indeps)`	names of independent variables
`e(idvar)`	name of ID variable
`e(estimator)`	`gs2sls`
`e(title)`	title in estimation output
`e(constant)`	`hasconstant` or `noconstant`
`e(exogr)`	exogenous regressors
`e(dlmat)`	names of spatial weighting matrices applied to *depvar*
`e(elmat)`	names of spatial weighting matrices applied to errors
`e(het)`	`heteroskedastic` or `homoskedastic`
`e(chi2type)`	`Wald`; type of model χ^2 test
`e(properties)`	`b V`
`e(estat_cmd)`	program used to implement `estat`
`e(predict)`	program used to implement `predict`
`e(marginsok)`	predictions allowed by `margins`
`e(marginsnotok)`	predictions disallowed by `margins`
`e(asbalanced)`	factor variables `fvset` as `asbalanced`
`e(asobserved)`	factor variables `fvset` as `asobserved`

Matrices

`e(b)`	coefficient vector
`e(delta_2sls)`	two-stage least-squares estimates of coefficients in spatial lag equation
`e(rho_2sls)`	generalized method of moments estimates of coefficients in spatial error equation
`e(V)`	variance–covariance matrix of the estimators

Functions

`e(sample)`	marks estimation sample

Methods and formulas

We consider a cross-sectional spatial autoregressive model with possible endogenous covariates and spatial autoregressive disturbances (SARAR), allowing for higher-order spatial dependence in the dependent variable, the exogenous variables, and the spatial errors. The model is

$$\begin{aligned}\mathbf{y} &= \sum_{j=1}^{J} \pi_j \widetilde{\mathbf{y}}_j + \sum_{k=1}^{K} \beta_k \mathbf{x}_k + \sum_{p=1}^{P} \gamma_p \mathbf{W}_p \mathbf{x}_p + \sum_{r=1}^{R} \lambda_r \mathbf{W}_r \mathbf{y} + \mathbf{u} \\ \mathbf{u} &= \sum_{s=1}^{S} \rho_s \mathbf{M}_s \mathbf{u} + \boldsymbol{\epsilon}\end{aligned} \tag{1}$$

where

$\mathbf{y}$ is an $n \times 1$ vector of observations on the dependent variable;

$\widetilde{\mathbf{y}}_j$ is an $n \times 1$ vector of observations on the jth endogenous variable; π_j is the corresponding scalar parameter;

$\mathbf{x}_k$ is an $n \times 1$ vector of observations on the kth exogenous variable; β_k is the corresponding scalar parameter;

$\mathbf{W}_p$, $\mathbf{W}_r$, and $\mathbf{M}_s$ are $n \times n$ spatial weighting matrices;

$\mathbf{W}_p\,\mathbf{x}_p$, $\mathbf{W}_r\,\mathbf{y}$, and $\mathbf{M}_s\mathbf{u}$ are $n \times 1$ spatial lags for the exogenous variable, dependent variable, and error terms; γ_p, λ_r, and ρ_s are scalar parameters; and

$\boldsymbol{\epsilon}$ is an $n \times 1$ vector of innovations.

The J endogenous variables $\widetilde{\mathbf{y}}_j$ are correlated with the errors $\mathbf{u}$. To estimate the model parameters, we need Q instrumental variables $\mathbf{x}^e_1, \mathbf{x}^e_2, \ldots, \mathbf{x}^e_Q$ with $Q \geq J$ that are correlated with the endogenous variables in $\widetilde{\mathbf{y}}_j$ and uncorrelated with the errors $\mathbf{u}$.

The model in (1) is frequently referred to as a higher-order spatial autoregressive model with spatial autoregressive disturbances, or namely, a SARAR(R, S) model.

The innovations $\boldsymbol{\epsilon}$ are assumed to be independent and identically distributed or independent but heteroskedastically distributed, where the heteroskedasticity is of unknown form. The generalized spatial two-stage least-squares (GS2SLS) estimator implemented in `spivregress` produces consistent estimates in both cases when the `heteroskedastic` option is specified.

For the first-order SARAR model, `spivregress` implements the GS2SLS estimator discussed in Arraiz et al. (2010) and Drukker, Egger, and Prucha (2013). This estimation strategy builds on Kelejian and Prucha (1998, 1999, 2010) and references cited therein. For higher-order SARAR(R, S) models, `spivregress` implements an extension of GS2SLS in Badinger and Egger (2011) to allow endogenous covariates.

Let's first rewrite (1) in a compact form.

$$\begin{aligned} \mathbf{y} &= \mathbf{Z}\delta + \mathbf{u} \\ \mathbf{u} &= \overline{\mathbf{U}}\rho + \varepsilon \end{aligned} \tag{2}$$

where

$\mathbf{Z}$ is the matrix of observations on all the variables in the equation for $\mathbf{y}$; $\mathbf{Z}$ contains the endogenous covariates $\widetilde{\mathbf{y}}_1, \ldots, \widetilde{\mathbf{y}}_J$, the exogenous covariates $\mathbf{x}_1, \ldots, \mathbf{x}_K$, the spatially lagged exogenous covariates $\mathbf{W}\mathbf{x}_1, \ldots, \mathbf{W}\mathbf{x}_P$, and the spatially lagged dependent variables $\mathbf{W}\mathbf{y}_1, \ldots, \mathbf{W}\mathbf{y}_R$;

$\overline{\mathbf{U}}$ contains all the spatial lags of the errors $\mathbf{u}$ that appear in (1); $\overline{\mathbf{U}}$ contains $\mathbf{M}_1\mathbf{u}, \ldots, \mathbf{M}_S\mathbf{u}$;

$\boldsymbol{\delta} = (\pi_1, \ldots \pi_J, \beta_1, \ldots, \beta_K, \gamma_1, \ldots, \gamma_P, \lambda_1, \ldots, \lambda_R)'$ is a vector of all the coefficients on the variables in the equation for $\mathbf{y}$; and

$\boldsymbol{\rho} = (\rho_1, \ldots, \rho_S)$ is the vector of coefficients on the spatially lagged errors.

Given these definitions, the estimator implemented in `spivregress` is a simple extension to the GS2SLS estimator documented in the *Methods and formulas* of `spregress`.

Specifically, after adding the instrumental variables $\mathbf{x}^e_1, \mathbf{x}^e_2, \ldots, \mathbf{x}^e_Q$ to the list of exogenous variables $\mathbf{X}_f$ used to create the matrix of instruments $\mathbf{H}_1$ in `spregress`, the other formulas in `spregress` specify how the estimator implemented in `spivregress` works. See *Methods and formulas* in [SP] **spregress** for further details.

References

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

Badinger, H., and P. H. Egger. 2011. Estimation of higher-order spatial autoregressive cross-section models with heteroscedastic disturbances. *Papers in Regional Science* 90: 213–235.

Drukker, D. M., P. H. Egger, and I. R. Prucha. 2013. On two-step estimation of a spatial autoregressive model with autoregressive disturbances and endogenous regressors. *Econometric Reviews* 32: 686–733.

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

——. 1999. A generalized moments estimator for the autoregressive parameter in a spatial model. *International Economic Review* 40: 509–533.

——. 2010. Specification and estimation of spatial autoregressive models with autoregressive and heteroskedastic disturbances. *Journal of Econometrics* 157: 53–67.

Also see

[SP] **spivregress postestimation** — Postestimation tools for spivregress

[SP] **estat moran** — Moran's test of residual correlation with nearby residuals

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spregress** — Spatial autoregressive models

[SP] **spxtregress** — Spatial autoregressive models for panel data

[R] **ivregress** — Single-equation instrumental-variables regression

[U] **20 Estimation and postestimation commands**

Title

spivregress postestimation — Postestimation tools for spivregress

Postestimation commands

The following postestimation command is of special interest after `spivregress`:

Command	Description
`estat impact`	direct, indirect, and total impacts

The following postestimation commands are also available:

Command	Description
`contrast`	contrasts and ANOVA-style joint tests of estimates
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estimates`	cataloging estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of coefficients
`margins`	marginal means, predictive margins, marginal effects, and average marginal effects
`marginsplot`	graph the results from margins (profile plots, interaction plots, etc.)
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
`predict`	predictions, residuals, influence statistics, and other diagnostic measures
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of estimates
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

predict

Description for predict

`predict` creates a new variable containing predictions such as the reduced-form mean, the direct mean, the indirect mean, the limited-information mean, the full-information mean, the naïve-form prediction, the linear prediction, the residuals, or the uncorrelated residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] *newvar* [*if*] [*in*] [, *statistic*]

statistic	Description
Main	
`rform`	reduced-form mean; the default
`direct`	direct mean
`indirect`	indirect mean
`limited`	limited-information mean
`full`	full-information mean
`naive`	naïve-form prediction
`xb`	linear prediction
`residuals`	residuals
`ucresiduals`	uncorrelated residuals

These statistics are only available in a subset of the estimation sample.

Options for predict

Main

`rform`, the default, calculates the reduced-form mean. It is the predicted mean of the dependent variable conditional on the independent variables and any spatial lags of the independent variables. See *Methods and formulas*.

`direct` calculates the direct mean. It is a unit's predicted contribution to its own reduced-form mean. The direct and indirect means sum to the reduced-form mean.

`indirect` calculates the indirect mean. It is the predicted sum of the other units' contributions to a unit's reduced-form mean.

`limited` calculates the limited-information mean. It is the predicted mean of the dependent variable conditional on the independent variables, any spatial lags of the independent variables, and any spatial lags of the dependent variable. `limited` is not available when the `heteroskedastic` option is used with `spivregress`.

`full` calculates the full-information mean. It is the predicted mean of the dependent variable conditional on the independent variables, any spatial lags of the independent variables, and the other units' values of the dependent variable. `full` is not available when the `heteroskedastic` option is used with `spivregress`.

`naive` calculates the naïve-form prediction. It is the predicted linear combination of the independent variables, any spatial lags of the independent variables, and any spatial lags of the dependent variable. It is not a consistent estimator of an expectation. See *Methods and formulas*.

`xb` calculates the predicted linear combination of the independent variables.

`residuals` calculates the residuals, including any autoregressive error term.

`ucresiduals` calculates the uncorrelated residuals, which are estimates of the uncorrelated error term.

margins

Description for margins

`margins` estimates margins of response for reduced-form mean, direct mean, indirect mean, and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

`margins` [*marginlist*] [`,` *options*]

`margins` [*marginlist*] `,` `predict(`*statistic* ...`)` [`predict(`*statistic* ...`)` ...] [*options*]

statistic	Description
`rform`	reduced-form mean; the default
`direct`	direct mean
`indirect`	indirect mean
`xb`	linear prediction
`limited`	not allowed with `margins`
`full`	not allowed with `margins`
`naive`	not allowed with `margins`
`residuals`	not allowed with `margins`
`ucresiduals`	not allowed with `margins`

Statistics not allowed with `margins` are functions of stochastic quantities other than `e(b)`.

For the full syntax, see [R] **margins**.

Remarks for margins

The computations that `margins` must do to calculate standard errors can sometimes be time consuming. Time will depend on the complexity of the spatial model and the number of spatial units in the data. You may want to fit your model with a subsample of your data, run `margins`, and extrapolate to estimate the time required to run `margins` on the full sample. See [P] **timer** and [P] **rmsg**.

estat impact

Description for estat impact

`estat impact` estimates the mean of the direct, indirect, and total impacts of independent variables on the reduced-form mean of the dependent variable.

Syntax for estat impact

`estat impact` [*varlist*] [*if*] [*in*] [`, nolog vce(`*vcetype*`)`]

varlist is a list of independent variables, including factor variables, taken from the fitted model. By default, all independent variables from the fitted model are used.

Options for estat impact

Main

`nolog` suppresses the calculation progress log that shows the percentage completed. By default, the log is displayed.

VCE

`vce(`*vcetype*`)` specifies how the standard errors of the impacts are calculated.

`vce(delta)`, the default, is the delta method and treats the independent variables as fixed.

`vce(unconditional)` specifies that standard errors account for sampling variance in the independent variables. This option is not available when `if` or `in` is specified with `estat impact`.

Remarks for estat impact

`estat impact` is essential for interpreting the output of `spivregress`. See [SP] **Intro 7** and example 1 of [SP] **spregress** for explanations and examples.

Stored results for estat impact

`estat impact` stores the following in `r()`:

Scalars

`r(N)`	number of observations

Macros

`r(vce)`	*vcetype* specified in `vce()`
`r(xvars)`	names of independent variables

Matrices

`r(b_direct)`	vector of estimated direct impacts
`r(Jacobian_direct)`	Jacobian matrix for direct impacts
`r(V_direct)`	estimated variance–covariance matrix of direct impacts
`r(b_indirect)`	vector of estimated indirect impacts
`r(Jacobian_indirect)`	Jacobian matrix for indirect impacts
`r(V_indirect)`	estimated variance–covariance matrix of indirect impacts
`r(b_total)`	vector of estimated total impacts
`r(Jacobian_total)`	Jacobian matrix for total impacts
`r(V_total)`	estimated variance–covariance matrix of total impacts

Methods and formulas

Methods and formulas are presented under the following headings:

Predictions

To motivate the predictions, consider the vector form of a spatial autoregressive model

$$\mathbf{y} = \lambda \mathbf{W}\mathbf{y} + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{1}$$

where

$\mathbf{y}$ is the vector containing each unit's dependent-variable observation,

$\mathbf{W}\mathbf{y}$ is a spatial lag of $\mathbf{y}$,

$\mathbf{X}$ is the matrix of independent-variable observations,

$\boldsymbol{\epsilon}$ is a vector of errors, and

λ and $\boldsymbol{\beta}$ are the coefficients.

Any spatial lags of the independent variables are assumed to be in $\mathbf{X}$. Spatial lags of the error do not affect the reduced-form, direct, or indirect means, so they are not included in (1) for simplicity.

Reduced-form mean

Equation (1) represents the spatial autoregressive model as a system of equations. The solution

$$\mathbf{y} = (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}) \tag{2}$$

implies that the mean of $\mathbf{y}$ given the independent variables and the spatial weighting matrix is

$$E(\mathbf{y} \mid \mathbf{X}, \mathbf{W}) = (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}\boldsymbol{\beta}) \tag{3}$$

This is known as the reduced-form mean because the solution in (2) is known as the reduced form of the model. The predicted reduced-form mean substitutes estimates of λ and $\boldsymbol{\beta}$ into (3).

Direct and indirect means

To define the direct mean and the indirect mean, let

$$\mathbf{S} = (\mathbf{I} - \lambda\mathbf{W})^{-1}$$

and let $\mathbf{S}_d$ be a matrix with diagonal elements of $\mathbf{S}$ on its diagonal and with off-diagonal elements set to 0.

The direct means are

$$\mathbf{S}_d\mathbf{X}\boldsymbol{\beta}$$

which capture the contributions of each unit's independent variables on its own reduced-form mean. Substituting estimates of λ and $\boldsymbol{\beta}$ produces the predictions.

The indirect means capture the contributions of the other units' independent variables on a unit's reduced-form prediction, and they are

$$\left\{(\mathbf{I} - \lambda\mathbf{W})^{-1} - \mathbf{S}_d\right\}\mathbf{X}\boldsymbol{\beta}$$

Limited-information mean

Instead of solving for the reduced form, the limited-information mean conditions on the spatial lag of y for observation i, which we denote by $(\mathbf{W}\,\mathbf{y})_i$, which yields

$$E\{y_i \mid \mathbf{X}, \mathbf{W}, (\mathbf{W}\,\mathbf{y})_i\} = \mathbf{x}_i\boldsymbol{\beta} + \lambda(\mathbf{W}\,\mathbf{y})_i + u_i \tag{4}$$

where u_i is the predictable part of the error term given $(\mathbf{W}\,\mathbf{y})_i$. See Kelejian and Prucha (2007) and Drukker, Prucha, and Raciborski (2013).

Full-information mean

The full-information mean conditions on the dependent-variable values of all the other units instead of conditioning on the spatial lag of the dependent variable, as does the limited-information mean. The additional information produces a better prediction of the error term when a spatial lag of the errors is in the model. See Kelejian and Prucha (2007).

Naïve-form predictor

The naïve-form predictor sets u_i to 0 in (4). It is not consistent for $E\{y_i \mid \mathbf{X}, \mathbf{W}, (\mathbf{W}\,\mathbf{y})_i\}$ because it ignores u_i.

Linear predictor

The linear predictor is $\mathbf{X}\beta$.

Residuals

The residuals are u_i from (4).

Uncorrelated residuals

The uncorrelated residuals are

$$\widehat{\boldsymbol{\epsilon}} = (\mathbf{I} - \widehat{\rho}\mathbf{M})^{-1}\,\mathbf{u}$$

where $\mathbf{u}$ is the vector of u_i's, $\mathbf{M}$ is the spatial weighting matrix for the autoregressive error term, and $\widehat{\rho}$ is the estimated correlation of $\mathbf{u}$.

Impacts

The total impact of an independent variable $\mathbf{x}$ is the average of the marginal effects it has on the reduced-form mean,

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})}{\partial x_j}$$

where $E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})$ is the ith element of the vector $E(\mathbf{y} \mid \mathbf{X}, \mathbf{W})$, whose formula is given in (2), and x_j is the jth unit's value for $\mathbf{x}$.

The direct impact of an independent variable $\mathbf{x}$ is the average of the direct, or own, marginal effects

$$\frac{1}{n}\sum_{i=1}^{n}\frac{\partial E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})}{\partial x_i}$$

The indirect impact of an independent variable $\mathbf{x}$ is the average of the indirect, or spillover, marginal effects.

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\frac{\partial E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})}{\partial x_j}$$

LeSage and Pace (2009, 36–37) call the average direct impact the "average total direct impact", and they call the average indirect impact the "average total indirect impact".

`estat impact` with the default `vce(delta)` uses the delta method to calculate the estimated variance of the impacts. This variance is conditional on the values of the independent variables in the model.

`estat impact` with `vce(unconditional)` uses the generalized method of moments estimation strategy to estimate the unconditional variance of the impacts. It accounts for sampling variance of the independent variables in the model.

References

Drukker, D. M., I. R. Prucha, and R. Raciborski. 2013. Maximum likelihood and generalized spatial two-stage least-squares estimators for a spatial-autoregressive model with spatial-autoregressive disturbances. *Stata Journal* 13: 221–241.

Kelejian, H. H., and I. R. Prucha. 2007. The relative efficiencies of various predictors in spatial econometric models containing spatial lags. *Regional Science and Urban Economics* 37: 363–374.

LeSage, J., and R. K. Pace. 2009. *Introduction to Spatial Econometrics*. Boca Raton, FL: Chapman & Hall/CRC.

Liu, D. 2017. How to create animated graphics to illustrate spatial spillover effects. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2018/03/06/how-to-create-animated-graphics-to-illustrate-spatial-spillover-effects/.

Also see

[SP] **spivregress** — Spatial autoregressive models with endogenous covariates

[U] **20 Estimation and postestimation commands**

Title

spmatrix — Categorical guide to the spmatrix command

Description

The `spmatrix` command creates, imports, manipulates, and exports **W** spatial weighting matrices. Listed below are the sections describing the `spmatrix` command.

Creating standard weighting matrices

`spmatrix create`	Create standard matrix
`spdistance`	Calculator for distance between places

Creating custom weighting matrices

`spmatrix userdefined`	Custom creation using a user-defined function
`spmatrix fromdata`	Custom creation based on variables in the dataset
`spmatrix spfrommata`	Get weighting matrix from Mata
`spmatrix matafromsp`	Copy weighting matrix to Mata
`spmatrix normalize`	Normalize matrix

Manipulating weighting matrices

`spmatrix dir`	List names of weighting matrices in memory
`spmatrix summarize`	Details of weighting matrix stored in memory
`spmatrix drop`	Drop weighting matrix from memory
`spmatrix copy`	Copy weighting matrix to new name
`spmatrix save`	Save spatial weighting matrix to file
`spmatrix use`	Load spatial weighting matrix from file
`spmatrix note`	Set or list note
`spmatrix clear`	Drop all weighting matrices from memory

Importing and exporting weighting matrices

`spmatrix export`	Export weighting matrix in standard format
`spmatrix import`	Import weighting matrix in standard format

Also see

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix copy — Copy spatial weighting matrix stored in memory

Description Quick start Menu Syntax
Also see

Description

`spmatrix copy` copies weighting matrices stored in memory to new names, also stored in memory.

Quick start

Copy existing matrix `Wd` to `Wdistance`

```
spmatrix copy Wd Wdistance
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix copy spmatname1 spmatname2
```

spmatname$_1$ is the name of an existing weighting matrix.

spmatname$_2$ is a name of a weighting matrix that does not exist.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix create — Create standard weighting matrices

Description

`spmatrix create` creates standard-format spatial weighting matrices.

Quick start

Create contiguity spatial weighting matrix `M` with default spectral normalization

```
spmatrix create contiguity M
```

Same as above

```
spmatrix create contiguity M, normalize(spectral)
```

Create row-standardized contiguity spatial weighting matrix `M`

```
spmatrix create contiguity M, normalize(row)
```

Create contiguity spatial weighting matrix `M` without normalization

```
spmatrix create contiguity M, normalize(none)
```

Create spectral-normalized inverse-distance spatial weighting matrix `W`

```
spmatrix create idistance W
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix create contiguity spmatname [if] [in] [, contoptions stdoptions]

spmatrix create idistance spmatname [if] [in] [, idistoption stdoptions]
```

spmatname is a weighting matrix name.

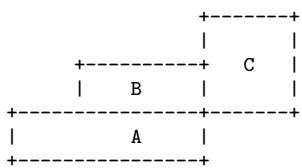
contoptions	Description
`rook`	share a border and not just a vertex
`first`	first-order neighbors
`second`[`(#)`]	second-order neighbors

idistoption	Description
`vtruncate(#)`	set (i, j) element to 0 if $1/\text{distance} \leq \#$

stdoptions	Description
`normalize(`*normalize*`)`	type of normalization; default is `normalize(spectral)`
`replace`	replace existing weighting matrix

Options for spmatrix create contiguity

`rook` specifies that areas that share just a vertex not be treated as neighbors. For instance, consider the following map:



If `rook` is not specified, A and C are neighbors because they have a vertex (corner) in common. If `rook` is specified, A and C are not neighbors. Regardless of whether `rook` is specified, A and B are neighbors and B and C are neighbors because they share a border (line segment).

`first` specifies that first-order neighbors be assigned 1. If areas i and j are neighbors, then $\textit{spmatname}_{i,j} = \textit{spmatname}_{j,i} = 1$. `first` is the default unless `second` or `second(#)` is specified.

`second`[`(#)`] specifies that the second-order neighbors—neighbors of neighbors—be assigned a nonzero value. `second` specifies that they be assigned 1. `second(#)` specifies that they be assigned #.

If you also specify option `first`, then the matrix created will set first-order neighbors to contain 1. For instance, if you specify `first second`, both kinds of neighbors will be set to 1. If you specify `first second(.5)`, first-order neighbors are set to 1 and second-order neighbors are set to 0.5.

Option for spmatrix create idistance

`vtruncate(#)` specifies that areas farther apart than # be set to 0. Type `spset` without arguments to determine the units in which # is specified. The `coordinates` line of `spset`'s output will be one of the following:

```
coordinates:  _CX, _CY (planar)
coordinates:  _CY, _CX (latitude and longitude, kilometers)
coordinates:  _CY, _CX (latitude and longitude, miles)
```

Units of # will be planar, kilometers, or miles. If planar, see [SP] **spdistance** for advice on determining the units.

If `spset` reports

```
coordinates:  none
```

then you cannot use `spmatrix create`.

Options for both contiguity and idistance

`normalize(`*normalize*`)` specifies how the resulting matrix is to be scaled.

`normalize(spectral)` is the default. The matrix will be normalized so that its largest eigenvalue is 1.

`normalize(minmax)` specifies that the matrix elements be divided by the smaller of the largest row or column sum of absolute values. The min–max calculation is much quicker than the spectral calculation and in most cases gives similar results as the spectral normalization.

`normalize(row)` specifies that each row of the matrix be divided by the row's sum (not absolute values). This adjustment can be performed even more quickly than the min–max adjustment.

`normalize(none)` specifies that the matrix not be rescaled. This option has one use: To store the matrix in unadjusted form so that you can fetch it later, make changes to it while the matrix is still in its original units, and then repost the matrix, at which point it will be rescaled. See *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

`replace` specifies that matrix *spmatname* may be replaced if it already exists.

Remarks and examples

See [SP] **Intro 1** about the role spatial weighting matrices play in SAR models and see [SP] **Intro 2** for a thorough discussion of the matrices. To remind you, the (i, j) element of a weighting matrix specifies the potential spillover from area j to i.

Remarks are presented under the following headings:

Creating contiguity matrices

`spmatrix create contiguity` is mostly used to create matrices with elements equal to 1 or 0 (before normalization). *spmatname*$_{i,j}$ is 1 when areas i and j are neighbors. The matrix is symmetric.

Creation of contiguity matrices requires that the Sp data in memory be linked to a shapefile. The data must look like this:

```
. spset
  Sp dataset
            data:  cross sectional or panel
 spatial-unit id:  _ID
     coordinates:  _CY, _CX (latitude and longitude, miles)
linked shapefile:  irrelevant
```

Determining whether places are neighbors requires a linked shapefile. Knowing their locations is not sufficient.

To create a contiguity matrix named `F` of first-order neighbors, type

```
. spmatrix create contiguity F
```

The contiguity matrix `F` is automatically normalized using the spectral normalization; see *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

In [SP] **Intro 1**, we discussed a spatial weighting matrix containing 1s for first-order neighbors and 0.5s for second-order neighbors. Such a matrix could be created by typing

```
. spmatrix create contiguity W, first second(0.5)
```

Also in the introduction, we considered making two weighting matrices, `W` for first-order neighbors and `V` for second. To create `W` and `V`, type

```
. spmatrix create contiguity W
. spmatrix create contiguity V, second
```

The syntax of the `spmatrix create contiguity` command is as follows:

Command	Meaning
1. `spmatrix create contiguity`	1st-order neighbors
2. `spmatrix create contiguity, first`	same as command 1
3. `spmatrix create contiguity, second`	2nd-order neighbors
4. `spmatrix create contiguity, second(1)`	same as command 3
5. `spmatrix create contiguity, first second`	1st- and 2nd-order neighbors
6. `spmatrix create contiguity, first second(1)`	same as command 5
7. `spmatrix create contiguity, first second(0.5)`	1st- and 2nd-order neighbors, 1st set to 1, 2nd set to 0.5

Creating inverse-distance matrices

`spmatrix create idistance` creates matrices with elements equal to the reciprocal of distance between places (before normalization). The matrix is symmetric.

Creation of inverse-distance matrices requires that the Sp data have coordinates, but a shapefile is not required. The data must look like this:

```
. spset
  Sp dataset
            data:  cross sectional or panel
 spatial-unit id:  _ID
     coordinates:  _CY, _CX (latitude and longitude, miles)
linked shapefile:  irrelevant
```

Coordinates must be defined, although they are not required to be latitude and longitude. If they are latitude and longitude, however, Sp needs to know; see [SP] **Intro 4**. Whether units are miles or kilometers is irrelevant.

To create an inverse-distance matrix named `Idist`, type

```
. spmatrix create idistance Idist
```

The inverse-distance matrix `Idist` is automatically normalized using the spectral normalization; see *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

`spmatrix create idistance` allows option `vtruncate(#)`, which sets spillovers less than or equal to # to 0. To create an inverse-distance matrix `I0` with places more than 100 apart, you could type

```
. spmatrix create contiguity I0, vtruncate(.01)
```

Note that you specify $\# = 1/\text{distance}$.

See the description of the `vtruncate()` option above for the meaning of how far apart 100 means.

Creating inverse-distance contiguity matrices

An inverse-distance contiguity matrix is a weighting matrix that contains inverse distance for neighbors and 0 otherwise. Here is how you create such a matrix:

1. Create the inverse-distance and contiguity matrices separately.
2. Multiply them element by element in Mata. The result is a matrix containing inverse distance for neighbors because the contiguity matrix contains 1s and 0s.
3. Store the Mata result as an Sp spatial weighting matrix.

We do that below to create an inverse-distance first-order neighbor matrix named `CN`.

```
. // -------------------------- create the matrices separately ---
.
. spmatrix create idistance  N, normalize(none)          // note 1
. spmatrix create contiguity C, first normalize(none)    // note 2
.
. // ------------------------------------ load them into Mata ---
.
. spmatrix matafromsp Wn v = N
. spmatrix matafromsp Wc v = C
.
.
. // ------------------------ multiply them element by element ---
. mata: Wcn = Wc :* Wn                                   // note 3
.
.
```

```
. // ---------------------------------- save the result in Sp ---
.
. spmatrix spfrommata CN = Wcn v                          // note 4
.
.
. // ------------------------------------------------ clean up ---
.
. mata: mata drop Wcn Wc Wn                                // note 5
. spmatrix drop C
. spmatrix drop N
. // ---------------------------------------- the final result ---
.
. spmatrix dir
```

Weighting matrix name	N x N	Type	Normalization
CN	254 x 254	custom	spectral

Notes:

1. We specify `normalize(none)` when we create the matrices separately for speed, not because it is necessary. Normalization amounts to multiplying the matrices by a constant, and that will not matter. Calculating the constant, however, takes considerable time.
2. We created `C` to be first-order neighbors. We could have included second-order neighbors as well by adding option `second` to the command.
3. Colon-asterisk (`:*`) is Mata's element-by-element matrix multiplication operator. It is called colon-multiply.
4. `spmatrix spfrommata` allows the `normalize()` option and defaults to `normalize(spectral)`, just as `spmatrix create` does. Thus, the matrix stored in Sp is normalized.
5. Do not skip the clean-up step. Spatial weighting matrices are $N \times N$ and can consume considerable amounts of memory.

   It is also important that we cleared the Mata matrices by dropping them and not by typing `clear mata`. Sp stores the matrices you create in Mata and, if you cleared Mata, the new weighting matrix `CN` would also be dropped!

See [SP] **spmatrix matafromsp** and [SP] **spmatrix spfrommata**.

The normalize() option

We have hardly mentioned the `normalize()` option so far, because `spmatrix create` normalizes matrices by default. Normalization is important. All the Sp commands that create spatial weighting matrices normalize by default and include the `normalize()` option for cases in which you want to modify how or whether it is done.

Sp provides three normalizations:

`normalize(spectral)`	the default
`normalize(minmax)`	min–max
`normalize(row)`	row

`normalize()` provides a fourth setting to skip normalization altogether:

`normalize(none)`	do not perform normalization

The Sp commands are so determined you do not forget to normalize spatial weighting matrices at the last step that you must not forget to specify `normalize(none)` when you are building a custom matrix from ingredients. The spectral and min–max normalizations merely change the scale of the matrix.

`normalize(row)`, however, is a normalization of a different ilk from the others. The others merely change the scale of the matrix. Changing a matrix's scale is performed by dividing the elements by a constant. `normalize(row)` divides each row by a different constant. Doing this transformation on the matrix changes the model specification.

See [SP] **spmatrix normalize** and *Choosing weighting matrices and their normalization* in [SP] **spregress** for details.

Panel data

If you have panel data and want to create a weighting matrix, you must use an `if` statement with `spmatrix create` to restrict the data to a single time value.

Here is an example. We load an Sp panel dataset and type `spset` to see the Sp settings:

```
. copy https://www.stata-press.com/data/r16/homicide_1960_1990.dta .
. copy https://www.stata-press.com/data/r16/homicide_1960_1990_shp.dta .
. use homicide_1960_1990
(S.Messner et al.(2000), U.S southern county homicide rate in 1960-1990)
. xtset _ID year
       panel variable:  _ID (strongly balanced)
        time variable:  year, 1960 to 1990, but with gaps
                delta:  1 unit
. spset
  Sp dataset homicide_1960_1990.dta
                data:  panel
   spatial-unit id:  _ID
           time id:  year (see xtset)
       coordinates:  _CX, _CY (planar)
  linked shapefile:  homicide_1960_1990_shp.dta
```

If we tried to create a weighting matrix the usual way, we would not be successful:

```
. spmatrix create contiguity W
variable _ID does not uniquely identify observations in the master data
r(459);
```

We get an error message because `spmatrix create` needs to know which observations to use. We must restrict `spmatrix create` to one observation per panel, which is easy to do using an `if` statement:

```
. spmatrix create contiguity W if year == 1990
```

Do not misinterpret the purpose of `if year == 1990`. The matrix created will be appropriate for creating spatial lags for any year, because if two spatial units share a border in 1990, they will share it in the other years too. The map does not change.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **Intro** — Introduction to spatial data and SAR models

Mata Reference Manual

Title

spmatrix drop — List and delete weighting matrices stored in memory

Description

`spmatrix dir` lists the Sp weighting matrices stored in memory.

`spmatrix drop` deletes a single Sp matrix from memory.

`spmatrix clear` deletes all Sp matrices from memory.

Quick start

List weighting matrices stored in memory

```
spmatrix dir
```

Drop weighting matrix `Wd`

```
spmatrix drop Wd
```

Drop all weighting matrices

```
spmatrix clear
```

Menu

Statistics > Spatial autoregressive models

Syntax

List the Sp weighting matrices stored in memory

```
spmatrix dir
```

Drop an Sp matrix from memory

```
spmatrix drop spmatname
```

Drop all weighting matrices from memory

```
spmatrix clear
```

spmatname is the name of an Sp weighting matrix stored in memory.

Remarks and examples

Remarks are presented under the following headings:

spmatrix dir

The spatial weighting matrices that you create are stored in memory. You create them with the following commands:

`spmatrix create`

`spmatrix import`

`spmatrix fromdata`

`spmatrix userdefined`

`spmatrix spfrommata`

`spmatrix dir` lists spatial weighting matrices names:

```
. spmatrix dir
```

Weighting matrix name	N x N	Type	Normalization
Wc	254 x 254	contiguity	spectral
Wd	254 x 254	idistance	spectral

When `spmatrix dir` reports that a matrix is a contiguity matrix, as it does with `Wc`, contiguity is used in its ex post sense. See [SP] **spmatrix summarize** or the [SP] **Glossary** for the definition of ex post contiguity matrices.

Save and drop matrices you are not using

Spatial weighting matrices are stored in memory, and they can consume a lot of it. The ones above consume a mere $254 \times 254 = 517128$ bytes each. Had the matrices been 3000×3000, they would have consumed 69 megabytes each.

Spatial weighting matrices can be saved on disk. Any that you are not currently using, you can save to disk and drop from memory:

```
. spmatrix save Wc using wc
  (file wc.stswm saved)
. spmatrix drop Wc
```

All spatial weighting matrices are dropped when you type

```
. spmatrix clear
```

or

```
. clear mata
```

or

```
. clear all
```

The `clear mata` command also clears any Mata functions or objects in memory. The `clear all` command also clears any data in memory.

Stored results

`spmatrix dir` stores the following in `r()`:

Macros

`r(names)`	space-separated list of matrix names

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix summarize** — Summarize weighting matrix stored in memory

[SP] **Intro** — Introduction to spatial data and SAR models

[D] **clear** — Clear memory

Mata Reference Manual

Title

spmatrix export — Export weighting matrix to text file

Description

`spmatrix export` saves one weighting matrix in a text file that you can use for sending to other researchers.

Stata users can import text files created by `spmatrix export`; see [SP] **spmatrix import**.

Quick start

Create file `wmat.txt` containing weighting matrix `Wme`

```
spmatrix export Wme using wmat.txt
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix export spmatname using filename [ , replace ]
```

spmatname is the name of a weighting matrix stored in memory.

filename is the name of a file with or without the default `.txt` suffix.

Option

`replace` specifies that *filename* may be overwritten if it already exists.

Remarks and examples

Remarks are presented under the following headings:

Using spmatrix export
The spmatrix export text-file format

Using spmatrix export

`spmatrix export` creates files containing spatial weighting matrices that you can send to other users who are not using Stata. If you want to send to Stata users, it is easier and better if you send Stata `.stswm` files created using `spmatrix save`. `spmatrix export` produces a text-based format that is easy for non-Stata users to read.

To send a contiguity matrix, for instance, you could type

```
. spmatrix create contiguity Wc
. spmatrix export Wc using contig.txt
  (matrix Wc saved in file contig.txt)
```

You could then email the file `contig.txt`.

The spmatrix export text-file format

An `spmatrix export` file contains values of the matrix and the `_ID` values to which the matrix's rows and columns correspond.

A small sample file is shown below. It corresponds to a 4×4 weighting matrix for U.S. counties 3137, 960, 298, and 707. If others are to be able to interpret this information, the counties need to be a standard code. We are using the standard FIPS code because, before creating spatial weighting matrices in [SP] **Intro 4**, we used `spset, modify id(fips)`.

To create the file listed below, we typed

```
. spmatrix create idistance Idist
. spmatrix export Idist using small.txt
  (matrix Idist saved in file small.txt)
```

We did this after keeping four observations so that we would have a small file to show you.

The resulting file is

```
. type small.txt
4
20029 0 .225898983673981 .259698923068494 .746562405514367
33003 .225898983673981 0 .123515701241913 .187089086384635
41021 .259698923068494 .123515701241913 0 .264715523882705
48227 .746562405514367 .187089086384635 .264715523882705 0
```

The file records a 4×4 spatial weighting matrix. Real examples would record much larger matrices. $N \times N$ matrices are recorded in $N + 1$ lines.

The first line states that $N = 4$. The matrix is 4×4.

The second and subsequent lines each record $N + 1$ values with spaces between them. The first value, 20029, is the `_ID` (FIPS) value corresponding to the first row of the weighting matrix. The remaining N values on the line are the first row of the matrix.

The remaining lines are repeats for the second row, third row, and so on. The first value is an `_ID` value and the rest are that `_ID`'s row of the matrix.

It is a simple and easy-to-read file.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix import** — Import weighting matrix from text file

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix fromdata — Create custom weighting matrix from data

Description

`spmatrix fromdata` creates custom spatial weighting matrices from Sp data.

There are two other ways to create custom weighting matrices: `spmatrix userdefined` and `spmatrix spfrommata`. Those ways may require less work, but they require knowledge of Mata.

Quick start

Create spectral-normalized spatial weighting matrix `Wnew` from the $N \times N$ "matrix" stored in variables `x1`, `x2`, ..., `xn`

```
spmatrix fromdata Wnew = x1 - xn
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix fromdata spmatname = varlist [ , options ]
```

spmatname is the name of the spatial weighting matrix to be created.

options	Description
`idistance`	store reciprocal of elements
`normalize(`*normalize*`)`	type of normalization; default is `normalized(spectral)`
`replace`	replace existing weighting matrix

Options

`idistance` converts distance to inverse distance by storing the reciprocal of the elements.

`normalize(`*normalize*`)` specifies how the resulting matrix is to be scaled. `normalize(spectral)` is the default. `normalize(minmax)`, `normalize(row)`, and `normalize(none)` are also allowed. See [SP] **spmatrix create** for full details of the option and *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

`replace` specifies that matrix *spmatname* be overwritten if it already exists.

Remarks and examples

The `fromdata` in `spmatrix fromdata` means that the matrix itself is stored as variables in the data. Some researchers are used to working this way, and if you are among them, `spmatrix fromdata` is for you.

If the matrix is stored with the variables because you created it using the data, you may want to consider using `spmatrix userdefined` and `spmatrix spfrommata` instead. Both require knowledge of Mata, so that is a disadvantage if you do not already know Mata. On the other hand, `spmatrix userdefined` does not require much knowledge and handles the creation of most custom weighting matrices simply and elegantly. `spmatrix spfrommata` requires more extensive knowledge of Mata, but it will handle problems that no other method can.

The problem with `spmatrix fromdata` is not that the matrix is stored in the data but that filling in the matrix is more work than it needs to be. Stata draws a distinction between rows and columns. Rows are observations and columns are variables. Stata is perfectly willing to sweep down observations, but few Stata commands will sweep across variables. Mata, being a matrix language, draws no such distinction.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix spfrommata** — Copy Mata matrix to Sp

[SP] **spmatrix userdefined** — Create custom weighting matrix

[SP] **Intro** — Introduction to spatial data and SAR models

Mata Reference Manual

Title

spmatrix import — Import weighting matrix from text file

Description

`spmatrix import` reads files created by `spmatrix export`.

Quick start

Create spatial weighting matrix `Wme` by importing file `wmat.txt`

```
spmatrix import Wme using wmat.txt
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix import spmatname using filename [ , replace ]
```

spmatname will be the name of the weighting matrix that is created.

filename is the name of a file with or without the default `.txt` suffix.

Option

`replace` specifies that weighting matrix *spmatname* in memory be overwritten if it already exists.

Remarks and examples

`spmatrix import` reads files written in a particular text-file format. The format is described in [SP] **spmatrix export**. Such a file might be named `contig.txt`. To read the file and store the matrix in Sp spatial weighting matrix `Wcontig`, type

```
. spmatrix import Wcontig using contig.txt
```

or

```
. spmatrix import Wcontig using contig
```

The file extension `.txt` is assumed.

The file is read and stored as is. Presumably, the user who created the matrix normalized it, but if not, you can normalize it by typing

```
. spmatrix normalize Wcontig
```

By default, `spmatrix normalize` uses spectral normalization, but you can specify a different normalization using the `normalize()` option. See [SP] **spmatrix normalize**.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix export** — Export weighting matrix to text file

[SP] **spmatrix normalize** — Normalize weighting matrix

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix matafromsp — Copy weighting matrix to Mata

Description

`spmatrix matafromsp` copies weighting matrix *spmatname* from Sp to Mata. Weighting matrix *spmatname* remains unchanged.

Quick start

Create weighting matrix `W` and ID vector `id` in Mata from spatial weighting matrix `C`

```
spmatrix matafromsp W id = C
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix matafromsp matamatrix matavec = spmatname
```

Remarks and examples

Remarks are presented under the following headings:

Getting W and id

The command

```
. spmatrix matafromsp W id = C
```

copies spatial weighting matrix `C` to a Mata matrix named `W` and copies `C`'s `_ID` vector to a Mata vector named `id`.

What is id? When a spatial weighting matrix such as C is created, stored along with it are the _ID values. Those _ID values identify the meaning of the rows and columns.

Consider a spatial weighting matrix created by spmatrix create. We use the datasets downloaded in [SP] **estat moran**.

```
. use homicide1990
(S.Messner et al.(2000), U.S southern county homicide rates in 1990)
. spset
  (output omitted)
. spmatrix create contiguity C
```

What is the meaning of element $c_{1,2}$? It is the spillover from _ID[2] to _ID[1]. If the data were currently spset on fips, _ID[1] might equal 48507 and _ID[2] might equal 48003, and thus it would be the spillover from Andrews to Zavala county in Texas. Sp keeps a copy of the _ID vector so that later, when the data are in a different order, $c\{1, 2\}$ will still mean the spillover from Andrews to Zavala county.

You need not concern yourself with id if you plan on doing something with W that does not involve the data in memory. If what you need to do involves the data in memory, you will need to address the problem that the order of the data in memory now is not the same as it was when W was created.

Using W without involving the data in memory

Say that you wish to fetch C from Sp just so you can change values greater than or equal to 0.8 to 0.5. Doing that does not involve the data in memory. You type

```
. spmatrix matafromsp W id = C
. mata:
------------------------------------------------- mata (type end to exit) -----
: for (i=1; i<=rows(W); i++) {
>         for (j=1; j<=cols(W); j++) {
>                 if (W[i,j] >= 0.8) W[i,j] = 0.5
>         }
> }
: end
--------------------------------------------------------------------------------
```

You might now store the W back into C by typing

```
. spmatrix spfrommata C = W id, replace
```

You specify the same id vector you received because you have not changed the ordering of the rows or columns of the matrix.

Using W involving the data in memory

If you intend to use W and the data in memory together, you need to align the data and W. The instructions presented here work with cross-sectional data but not panel data.

First, check whether the data and W are conformable:

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: ID = st_data(., "_ID")
: assert( sort(ID, 1) == sort(id, 1) )
: end
--------------------------------------------------------------------------------
```

If Mata reports that the assertion is false, then the data and W are not conformable. This has nothing to do with observations and rows and columns being in different order. Not conformable means that one, the other, or both are missing _ID values that the other one has.

Let's imagine that Mata responds with silence to the assertion. Thus, the data are conformable. If they are also in the same order, you can use the data and W together, so find out if they are.

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: assert( ID == id )
: end
-------------------------------------------------------------------------------
```

If they are in the same order, row/column 1 of the matrix corresponds to observation 1 of the data, row/column 2 of the matrix corresponds to observation 2 of the data, and so on.

If Mata reports that the assertion is false, you have to put the data in the same order. Here is how:

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: p  = order(id, 1)
: W  =  W[p, p]          // put W in ascending order of id
: id = id[p]             // put id in ascending order of id
: end
-------------------------------------------------------------------------------

. sort _ID               // put the data in ascending order of _ID
```

You can now do whatever with W and the data. Row/column 1 of W corresponds to observation 1 of the data, row/column 2 of W corresponds to observation 2 of the data, and so on.

Perhaps whatever you will do involves, as a last step, posting the matrix back to Sp. In that case, use the id variable you updated:

```
. spmatrix spfrommata C = W id, replace
```

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix spfrommata** — Copy Mata matrix to Sp

[SP] **Intro** — Introduction to spatial data and SAR models

Mata Reference Manual

Title

spmatrix normalize — Normalize weighting matrix

Description

`spmatrix normalize` normalizes a spatial weighting matrix. It is mostly used after `spmatrix import`.

Quick start

Normalize spatial weighting matrix `W` using the default spectral normalization

```
spmatrix normalize W
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix normalize spmatname [ , normalize(normalize) ]
```

spmatname is the name of an existing spatial weighting matrix stored in memory.

normalize	Description
`spectral`	spectral; the default
`minmax`	min–max
`row`	row
`none`	do not normalize; leave matrix as is

Option

`normalize(`*normalize*`)` specifies how the resulting matrix is to be scaled. `normalize(spectral)` is the default. `normalize(minmax)`, `normalize(row)`, and `normalize(none)` are also allowed. See [SP] **spmatrix create** for full details of the option and *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

Remarks and examples

Remarks are presented under the following headings:

Using spmatrix normalize after spmatrix import

With one exception, the commands that create spatial weighting matrices provide a `normalize()` option and default to `normalize(spectral)`. `spmatrix import` is the exception. You can use `spmatrix normalize` after importing; see [SP] **spmatrix import**.

Using spmatrix normalize after other commands

If you create a matrix using `normalize(none)`, you can use `spmatrix normalize` to normalize the matrix subsequently. For instance,

```
. spmatrix create contiguity Wc, normalize(none)
. spmatrix normalize Wc
```

Using spmatrix normalize to change normalization

Sp provides three normalizations:

`normalize(spectral)`	the default
`normalize(minmax)`	min–max
`normalize(row)`	row

Concerning the first two, you can use `spmatrix normalize` to change the normalization.

1. If `W` is normalized spectrally, no matter how you created it, normalizing it again spectrally leaves the matrix unchanged.
2. The same applies to the min–max normalization. If `W` is normalized using min–max, normalizing it again leaves the matrix unchanged.
3. If `W` is normalized spectrally and you renormalize using min–max, the result is the same as you would have obtained had `W` been normalized using min–max at the outset.
4. The same applies if the roles of min–max and spectral are reversed. If `W` is normalized using min–max and you renormalize it spectrally, the result is the same as if you had normalized it spectrally at the outset.

Row normalization, meanwhile, is unique. You can apply row normalization repeatedly to an already row-normalized matrix and obtain the same results, but you cannot change normalizations.

See *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix import** — Import weighting matrix from text file

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix note — Put note on weighting matrix, or display it

Description

`spmatrix note` *spmatname*`:` *text* puts or replaces the note on weighting matrix *spmatname* stored in memory.

`spmatrix note` *spmatname* displays the note.

Quick start

Place or replace note on spatial weighting matrix `W`

```
spmatrix note W: inverse-distance 1st-order contiguity matrix
```

Display note on spatial weighting matrix `W`

```
spmatrix note W
```

Clear note on spatial weighting matrix `W`

```
spmatrix note W:
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix note spmatname : text
spmatrix note spmatname
```

spmatname is the name of an existing weighting matrix.

Remarks and examples

See [SP] **spmatrix save** for an example using `spmatrix note`.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix save** — Save spatial weighting matrix to file

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix save — Save spatial weighting matrix to file

Description

`spmatrix save` saves the specified spatial weighting matrix to disk. You can later load the matrix using `spmatrix use`.

spmatname is saved to disk but is not dropped from memory. If you wish to eliminate the matrix from memory, see [SP] **spmatrix drop**.

Quick start

Save spatial weighting matrix `W` in file `w.stswm`

```
spmatrix save W using w.stswm
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix save spmatname using filename [ , replace ]
```

spmatname is the name of an existing weighting matrix.

filename is the name of a file with or without the `.stswm` suffix.

Option

`replace` specifies that *filename* be overwritten if it already exists.

Remarks and examples

Saving spatial weighting matrices in files allows you to use them from one session to the next.

It is easy to lose track of which files contain which matrices. It can be useful to set the weighting matrix's note as a reminder:

```
. spmatrix note Wme: inverse-distance first-order contiguity matrix
. spmatrix save Wme using wme
  (matrix Wme saved in file wme.stswm)
```

`spmatrix use` will display the note when it loads the file:

```
. spmatrix use W1 using wme
  (inverse-distance first-order contiguity matrix)
```

The name you specify when you use the matrix is not required to match the name you used when you saved it.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix export** — Export weighting matrix to text file

[SP] **spmatrix use** — Load spatial weighting matrix from file

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix spfrommata — Copy Mata matrix to Sp

Description

`spmatrix spfrommata` copies a weighting matrix and an ID vector from Mata to an Sp spatial weighting matrix.

Quick start

Create Sp spatial weighting matrix `Wnew` from Mata matrix `W` and vector `v` with the default spectral normalization

```
spmatrix spfrommata Wnew = W v
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix spfrommata spmatname = matamatrix matavec [ , options ]
```

options	Description
`normalize(`*normalize*`)`	type of normalization; default is `normalize(spectral)`
`replace`	replace existing weighting matrix

Options

`normalize(`*normalize*`)` specifies how the resulting matrix is to be scaled. `normalize(spectral)` is the default. `normalize(minmax)`, `normalize(row)`, and `normalize(none)` are also allowed. See [SP] **spmatrix create** for full details of the option and *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

`replace` specifies that matrix *spmatname* be overwritten if it already exists.

Remarks and examples

Remarks are presented under the following headings:

W and v
Simple use
Advanced use

W and v

Two components are required to set an Sp spatial weighting matrix: the spatial weighting matrix itself and its vector of _ID values. Let's call them W and v, respectively. v states that the first row and column of W correspond to _ID==v[1], the second row and column correspond to _ID==v[2], and so on. The purpose of v and how it works is explained in [SP] **spmatrix matafromsp**.

Examples of spmatrix spfrommata can be found in [SP] **spmatrix create** and [SP] **spmatrix matafromsp**.

Simple use

We are going to show you how Mata can be used to construct complicated spatial weighting matrices. However, we will start with a simple case in which the values of the weighting matrix $W_{i,j}$ are a function of variables in observations i and j of the data in memory. Inverse-distance matrices are an example of this. The distance between i and j is a function of the values of Stata variables _CX and _CY in observations i and j.

We start by loading the Sp data into memory:

```
. use your_sp_data
```

The Mata solution is

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: id = st_data(., "_ID")
: location = st_data(., ("_CX", "_CY"))
: N = st_nobs()
: W = J(N, N, 0)
: for (i=1; i<=N; i++) {
>         for (j=1; j<i; j++) {
>                 delta = location[i,.] - location[j,.]
>                 W[i,j] = W[j,i] = 1/sqrt(delta*delta')
>         }
> }
: end
--------------------------------------------------------------------------------

. spmatrix spfrommata myIdist = W id
```

We just created an inverse-distance matrix. If you wanted to create such matrices, it would obviously be easier to type

```
. spmatrix create idistance myIdist
```

The Mata solution has the advantage that you can substitute different inverse-distance functions, such as inverse-distance squared. There is an easier solution for that case too, namely, the one outlined in [SP] **spmatrix userdefined**.

Now that you know how Mata and spmatrix spfrommata work in simple cases, we can show you an example that could be done no other way than by direct use of Mata.

Advanced use

You have `export.dta`, a cross-sectional Sp dataset on countries and their characteristics. Its `_ID` variable contains standard country codes. You need to construct a spatial weighting matrix to use with it.

`export.dta` does not itself contain sufficient information to construct the matrix you want to use. Instead, you have a second dataset, which is not Sp. It contains

```
. copy https://www.stata-press.com/data/r16/exports.dta .
. use exports
(country export data)
. describe
Contains data from exports.dta
  obs:        38,220                          country export data
 vars:             4                          17 Apr 2019 08:26
-------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
-------------------------------------------------------------------------------
from            int     %9.0g                 from (country code)
to              int     %9.0g                 to (country code)
exports         float   %9.0g                 exports ($ value)
gdp             float   %9.0g                 GDP of producer
-------------------------------------------------------------------------------
Sorted by:
```

The data record exports from one country to another for all 196 countries of the world. We say exports, but we can just as well interpret the data as imports by reversing the roles of variables `from` and `to`.

To simplify the problem, we are going to assume the country codes recorded in variables `from` and `to` are 1, 2, ..., 196 and that the same codes are recorded in variable `_ID` of `exports.dta`. If the true country codes were not 1, 2, ..., 196, you could easily construct such country codes.

We are going to create a spatial weighting matrix from these data. Potential spillover from j to i will be

$$W_{i,j} = \frac{(\text{exports from } i \text{ to } j) + (\text{exports from } j \text{ to } i)}{i\text{'s GDP}}$$

This weighting matrix is a near cousin to one developed by Badinger and Egger (2008).

$\mathbf{W}$ would be easy to calculate if we had a matrix $\mathbf{E}$ recording exports and a vector $\mathbf{g}$ recording GDP. $E_{i,j}$ would be the exports from i to j, and g_i would record GDP of country i. The formula for $W_{i,j}$ would then be

$$W_{i,j} = (E_{i,j} + E_{j,i})/g_i$$

E and **g** can be easily created in Mata:

```
. tomata from to export gdp
. mata:
------------------------------------------------- mata (type end to exit) -----
: g = J(196, 1, 0)
: E = J(196, 196, 0)
: for (k=1; k<=length(exports); k++) {
>         i = from[k]
>         j = to[k]
>         g[i] = gdp[k]
>         E[i,j] = exports[k]
> }
: end
--------------------------------------------------------------------------------
```

`tomata` (Gould 2006) is a community-contributed command that makes it easy to create Mata matrix views of individual Stata variables. Type `search tomata` for details. The matrices will have the same names as the variables.

We can now calculate the weighting matrix:

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: W = (E + E') :/ g
: end
--------------------------------------------------------------------------------
```

Finally, we post the result to Sp. We create column vector `id` containing 1, 2, . . . , 196 because row/column 1 of `W` corresponds to country code 1, row/column 2 of `W` corresponds to country code 2, and so on.

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: id = 1::196
: end
--------------------------------------------------------------------------------
. spmatrix spfrommata Wt = W id
```

We could now use `exports.dta` and fit a model using `Wt` to create spatial lags.

References

Badinger, H., and P. H. Egger. 2008. Intra- and inter-industry productivity spillovers in OECD manufacturing: A spatial econometric perspective. Working paper 2181, CESifo Group, Munich, Germany. http://www.cesifo-group.de/DocDL/cesifo1_wp2181.pdf.

Gould, W. W. 2006. Stata tip 35: Detecting whether data have changed. *Stata Journal* 6: 428–429.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix create** — Create standard weighting matrices

[SP] **spmatrix matafromsp** — Copy weighting matrix to Mata

[SP] **Intro** — Introduction to spatial data and SAR models

Mata Reference Manual

Title

spmatrix summarize — Summarize weighting matrix stored in memory

Description

`spmatrix summarize` reports the summary values of the elements of a weighting matrix.

Quick start

Display summary statistics for spatial weighting matrix `Wd`

```
spmatrix summarize Wd
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix summarize spmatname [ , generate(newvar) ]
```

spmatname is the name of a weighting matrix.

Option

`generate(`*newvar*`)` adds new variable *newvar* to the data. It contains the number of neighbors for each observation. `generate()` may be specified only when `spmatrix summarize` or `spmatrix dir` report that the matrix is a contiguity matrix. See [SP] **Glossary** for a definition of ex post contiguity matrices.

Remarks and examples

We will again use the data from [SP] **Intro 7**. `spmatrix summarize` produces output such as

```
. use tl_2016_us_county
. keep if STATEFP=="48"
  (output omitted)
. spmatrix create idistance Wd
. spmatrix create contiguity Wc
. spmatrix summarize Wd
Weighting matrix  Wd
```

Type	idistance
Normalization	spectral
Dimension	254 x 254
Elements	
minimum	0
minimum > 0	.0008812
mean	.0038122
max	.0512134

```
. spmatrix summarize Wc
Weighting matrix  Wc
```

Type	contiguity
Normalization	spectral
Dimension	254 x 254
Elements	
minimum	0
minimum > 0	.1522758
mean	.0034177
max	.1522758
Neighbors	
minimum	1
mean	5.700787
maximum	9

When a matrix is a contiguity matrix, a summary of the number of neighbors is added to the output. By contiguity matrix, we mean a contiguity matrix in the sense we describe below. A matrix created by `spmatrix create contiguity` does not necessarily qualify, and matrices created by other commands sometimes do.

We call this definition ex post contiguity. Such matrices 1) are symmetric and 2) have all elements equal to one of two values: 0 or c. In this case, c happens to be 0.1522758, but that is not important. What is important is that there are two values, one zero and the other nonzero. Spatial weighting matrices do not have a scale. If there are only two values, the matrix can be fully described as containing values such that "there is spillover" or "there is no spillover". Those with spillover are what we call neighbors.

Matrices created by `spmatrix create contiguity` are not necessarily ex post contiguity matrices. For instance, typing

```
. spmatrix create contiguity W2, first second(.5)
```

would create a matrix containing three values—0, 0.05, and 1—before normalization and different values after normalization. If we wanted to count neighbors, we would need to count first- and second-order neighbors separately. Meanwhile, typing

```
. spmatrix create contiguity W1
```

and

```
. spmatrix create contiguity W12, first second
```

would produce ex post contiguity matrices.

`spmatrix dir` uses the word contiguity in the same way as `spmatrix summarize`, namely, ex post contiguity:

```
. spmatrix dir
```

Weighting matrix name	N x N	Type	Normalization
Wc	254 x 254	contiguity	spectral
Wd	254 x 254	idistance	spectral

Matrix `Wc` is an ex post contiguity matrix regardless of how it was created.

Normalization does not interfere with ex post contiguity. Normalization is performed on a spatial weighting matrix by dividing its elements by a constant, and thus a matrix that starts out with two distinct values still has two distinct values after normalization. Row normalization—`normalize(row)`—works differently. Each row is divided by potentially a different constant and thus does not satisfy the definition of ex post contiguity.

Stored results

`spmatrix summarize` stores the following in `r()`:

Scalars

`r(n)`	number of rows (columns)
`r(min)`	elements: minimum value
`r(mean)`	elements: mean value
`r(min0)`	elements: minimum of elements>0
`r(max)`	elements: maximum value

Macros

`r(type)`	type of matrix: `contiguity`, `idistance`, or `custom`
`r(normalization)`	type of normalization

If `r(type)` = `contiguity`, also stored are

Scalars

`r(n_min)`	neighbors: minimum value
`r(n_mean)`	neighbors: mean value
`r(n_max)`	neighbors: maximum values

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix use — Load spatial weighting matrix from file

Description

`spmatrix use` loads the spatial weighting matrix previously saved using `spmatrix save`.

Quick start

Use spatial weighting matrix `W` stored in file `wme.stswm`

```
spmatrix use W using wme.stswm
```

Menu

Statistics > Spatial autoregressive models

Syntax

```
spmatrix use spmatname using filename [ , replace ]
```

spmatname is a weighting matrix name.

filename is the name of a file with or without the `.stswm` suffix.

Option

`replace` specifies that weighting matrix *spmatname* be overwritten if it already exists.

Remarks and examples

See [SP] **spmatrix save** for an example of `spmatrix use`.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix import** — Import weighting matrix from text file

[SP] **spmatrix save** — Save spatial weighting matrix to file

[SP] **Intro** — Introduction to spatial data and SAR models

Title

spmatrix userdefined — Create custom weighting matrix

Description

`spmatrix userdefined` is one way of creating custom spatial weighting matrices. The function you write need not be based on coordinate locations.

Quick start

Having written the Mata function `SinvD()`, create new spatial weighting matrix `C` with default spectral normalization

```
spmatrix userdefined C = SinvD(_CX _CY)
```

Menu

Statistics > Spatial autoregressive models

Syntax

`spmatrix userdefined` *Wmatname* `=` *fcnname*`(`*varlist*`)` [*if*] [*in*] [`,` *options*]

Wmatname is a weighting matrix name, such as `W`.

fcnname is the name of a Mata function you have written, such as `SinvD` or `Awind`.

1. *fcnname* must start with the letter `S` or `A`, which indicates whether the function produces a symmetric or an asymmetric result.
2. *fcnname* receives two row-vector arguments and returns a scalar result. For example,

```
function SinvD(v1, v2)
{
        return(1/sqrt((v1-v2)*(v1-v2)'))
}
```

Function `SinvD()` starts with `S` because for all x and y, `SinvD(`x`,` y`)` = `SinvD(`y`,` x`)`.

options	Description
`normalize(`*normalize*`)`	type of normalization; default is `normalize(spectral)`
`replace`	replace existing weighting matrix

Options

`normalize(`*normalize*`)` specifies how the resulting matrix is to be scaled. `normalize(spectral)` is the default. `normalize(minmax)`, `normalize(row)`, and `normalize(none)` are also allowed. See [SP] **spmatrix create** for full details of the option and *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

`replace` specifies to overwrite matrix *spmatname* if it already exists.

Remarks and examples

Sp provides five ways to create spatial weighting matrices:

1. [SP] **spmatrix create** creates standard weighting matrices. No programming and little effort is required.
2. [SP] **spmatrix import** imports weighting matrices produced by others.
3. [SP] **spmatrix fromdata** lets you create custom weighting matrices without Mata programming.
4. [SP] **spmatrix userdefined** lets you create custom weighting matrices. Some Mata programming is required.
5. [SP] **spmatrix spfrommata** leaves it to you to create matrices from start to finish, which you can do in Mata with or without programs. Once created, you use `spmatrix spfrommata` to store it.

This manual entry concerns method 4. Remarks are presented under the following headings:

Overview

Consider a cross-sectional spatial dataset of N observations. Each observation contains data on a place. We say place, but it need not be a physical place such as census block 060670011011085, zip code 77845, city College Station, county Brazos, state Texas, or country United States. It could be a network node or anything else.

A weighting matrix $\mathbf{W}$ is $N \times N$. Its elements $W_{i,j}$ record the potential spillover from place j to i. For simplicity and without loss of generality, we will assume that places i and j correspond to observations i and j.

`spmatrix userdefined` handles situations when $W_{i,j}$ is a function of $\mathbf{v}_i$ and $\mathbf{v}_j$, where $\mathbf{v}$ is a vector. $W_{i,j}$ could be a function of all the variables in observations i and j, but probably it is a function of a subset of them. For instance, if the spatial weighting matrix were based on locations, $W_{i,j}$ would be a function of variables `_CX` and `_CY` in the two observations. Or if the weighting were based on industry output, it might be a function of variables `f1`, `f2`, ..., `f12` in observations i and j. The variables might contain the fraction of output within industrial group and so sum to 1, or they might record total dollar output.

Whatever the relevant variables are, let's just call them *varlist*. Then

$\mathbf{v}_i$ = row vector of values of *varlist* in observation i

$\mathbf{v}_j$ = row vector of values of *varlist* in observation j

The formula for the elements of a spatial weighting matrix is

$$W_{i,j} = 0 \quad \text{if } i = j$$
$$f(v_i, v_j) \quad \text{otherwise}$$

`spmatrix userdefined` handles this problem when you type

`spmatrix userdefined` *Wmatname* `=` *fcnname*`(`*varlist*`)`

The mapping from the command's syntax to the mathematics is

Syntax element	Corresponding mathematical element
Wmatname	$\mathbf{W}$
fcnname`()`	$f(\cdot)$
varlist	$\mathbf{v}_i$, $\mathbf{v}_j$

Here is an example:

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: function SinvD(vi, vj)
> {
>         return (1/sqrt( (vi-vj)*(vi-vj)' ) )
> }
: end
--------------------------------------------------------------------------------

. spmatrix userdefined W = SinvD(_CX _CY)
```

The above produces the matrix that could also be created by typing

```
. spmatrix create idistance W
```

Here is an example that `spmatrix create` cannot duplicate:

```
. mata:
------------------------------------------------- mata (type end to exit) -----
: function Sdistance(vi, vj)
> {
>         return ( sqrt( (vi-vj)*(vi-vj)' ) )
> }
: end
--------------------------------------------------------------------------------

. spmatrix userdefined W = Sdistance(f*)
```

The above code calculates the distance between the `f*` variables for all i and j. The "farther" apart industrial output shares are, the greater is the incentive for places i and j to engage in trade.

Sfcnname() versus Afcnname()

The Mata functions you write start with the letter `S` when

`S`*fcnname*`(`v_i`, `v_j`) == S`*fcnname*`(`v_j`, `v_i`)`

Commutative functions produce symmetric matrices. `spmatrix userdefined` runs faster in this case because when it calls the function to calculate $W_{i,j}$, it also stores the result in $W_{j,i}$.

If the function is not commutative, it produces asymmetric matrices. Name such functions `A`*fcnname*`()`. Then, the function will be called separately to calculate $W_{i,j}$ and $W_{j,i}$.

Programming style

We wrote inverse distance as

```
. mata:
: program SinvD(vi, vj)
> {
>         return( 1/sqrt( (vi-vj)*(vi-vj)' ) )
> }
: end
```

There are other programming styles we could have used. The above used vector and matrix notation. Here is the same calculation written even more densely:

```
: program SinvD(vi, vj)
> {
>         delta = vi - vj
>         return( 1/sqrt( delta*delta' ) )
> }
: end
```

And here is the calculation again in more traditional scalar notation:

```
. mata:
: program SinvD(vi, vj)
> {
>         delta_x = vi[1] - vj[1]
>         delta_y = vi[2] - vj[2]
>         return( 1/sqrt(delta_x^2 + delta_y^2) )
> }
: end
```

You can write code in whichever style you find easiest.

Advanced programs

The Mata program you write is not required to be simple. If you were an epidemiologist, you could write a program that accounted for prevailing wind direction so that communicable diseases were more likely to spillover when location j is west of i and j's prevailing winds are out of the west. The program would look something like this:

```
: mata
: program Awind(vi, vj)
> {
>         locj      = (vj[1], vj[2])
>         loci      = (vi[1], vi[2])
>         windfromi =  vi[3]    // 1=N, 2=E, 3=S, 4=W
>
>         j_rel_i   = ...       // 1 if j N of i,
>                               // 2 if j E of i,
>                               // ..
>
>         if (j_rel_i == windfrom)  c = 1.5
>         else                      c = 0.5
>
>         return(SinvD(loci, locj)*c)
> }
: end
```

We omitted lines, and if we were going to use this approach, we would further complicate the program by considering the directions N, NE, E, SE, S, SW, W, and NW.

However complicated the code might be, the weighting matrix would be calculated by typing

```
. spmatrix userdefined Wadj = Awind(_CX _CY winddir)
```

`Wadj` would contain a wind-adjusted distance matrix, which will also be spectral normalized.

And then, we would be tempted to convert `Wadj` to be a wind-adjusted distance matrix for areas that bordered on each other. Let us show you how.

Mixed approaches

You do not have to calculate everything in your program. Let's imagine that in the above example, the researcher only wants to use the wind-adjusted calculation when i and j are first-order neighbors. Otherwise, the spillover is to be 0. We can use the same approach explained in more detail in [SP] **spmatrix create**:

1. Create the wind-adjusted matrix as shown above, but do not normalize it.
2. Create the first-order neighbor matrix using `spmatrix create contiguity`, also unnormalized.
3. Multiply the matrices element by element in Mata.
4. Store the Mata result in Sp.

The code is

```
. // -------------------------- create the matrices separately ---
. spmatrix userdefined Wadj = Awind(_CX _CY winddir), normalize(none)
. spmatrix create contiguity C, first normalize(none)
.
.
. // ------------------------------------ load them into Mata ---
. spmatrix matafromsp W1 v = Wadj
. spmatrix matafromsp W2 v = C
.
.
. // -------------------------multiply them element by element ---
. mata: W3 = W1 :* W2
.
```

```
. // ------------------------------------save the result in Sp ---
.
. spmatrix spfrommata Wfinal = W3 v
.
. // ------------------------------------------------clean up ---
.
. mata: mata drop  W1 W2 W3
. spmatrix drop Wadj
. spmatrix drop C
.
. // ---------------------------------------------final result ---
. spmatrix dir
```

Weighting matrix name	N x N	Type	Normalization
Wfinal	254 x 254	custom	spectral

In the above, `:*` (colon-asterisk) is Mata's element-by-element multiply function. See [SP] **spmatrix create** for more explanation.

Also see

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spmatrix spfrommata** — Copy Mata matrix to Sp

[SP] **Intro** — Introduction to spatial data and SAR models

Mata Reference Manual

Title

spregress — Spatial autoregressive models

Description

`spregress` is the equivalent of `regress` for spatial data. `spregress` fits spatial autoregressive (SAR) models, also known as simultaneous autoregressive models. If you have not read [SP] **Intro 1**–[SP] **Intro 8**, you should do so before using `spregress`.

To use `spregress`, your data must be Sp data. See [SP] **Intro 3** for instructions on how to prepare your data.

To specify spatial lags, you will need to have one or more spatial weighting matrices. See [SP] **Intro 2** and [SP] **spmatrix** for an explanation of the types of weighting matrices and how to create them.

Quick start

Spatial autoregressive model of `y` on `x1` and `x2` with a spatial lag of `y` specified by the spatial weighting matrix `W` using the GS2SLS estimator

```
spregress y x1 x2, gs2sls dvarlag(W)
```

Add a spatially lagged error term also specified by `W`

```
spregress y x1 x2, gs2sls dvarlag(W) errorlag(W)
```

Add spatial lags of covariates `x1` and `x2`

```
spregress y x1 x2, gs2sls dvarlag(W) errorlag(W) ivarlag(W: x1 x2)
```

Add a higher-order spatial lag of `y` specified by another weighting matrix `M`

```
spregress y x1 x2, gs2sls dvarlag(W) errorlag(W) ivarlag(W: x1 x2) ///
     dvarlag(M)
```

Use the ML estimator and include spatial lags of `y`, `x1`, `x2` and the error term specified by `W`

```
spregress y x1 x2, ml dvarlag(W) errorlag(W) ivarlag(W: x1 x2)
```

Add an additional spatial lag of the covariates specified by the matrix `M`

```
spregress y x1 x2, ml dvarlag(W) errorlag(W) ivarlag(W: x1 x2)     ///
     ivarlag(M: x1 x2)
```

Same model fit by GS2SLS

```
spregress y x1 x2, gs2sls dvarlag(W) errorlag(W) ivarlag(W: x1 x2)  ///
     ivarlag(M: x1 x2)
```

Model fit by GS2SLS with spatial lags of `y` and of the error term and treating the errors as heteroskedastic

```
spregress y x1 x2, gs2sls heteroskedastic dvarlag(W) errorlag(W)
```

Menu

Statistics > Spatial autoregressive models

Syntax

Generalized spatial two-stage least squares

```
spregress depvar [indepvars] [if] [in], gs2sls [gs2sls_options]
```

Maximum likelihood

```
spregress depvar [indepvars] [if] [in], ml [ml_options]
```

gs2sls_options	Description
Model	
* gs2sls	use generalized spatial two-stage least-squares estimator
dvarlag(*spmatname*)	spatially lagged dependent variable; repeatable
errorlag(*spmatname*)	spatially lagged errors; repeatable
ivarlag(*spmatname* : *varlist*)	spatially lagged independent variables; repeatable
noconstant	suppress constant term
heteroskedastic	treat errors as heteroskedastic
force	allow estimation when estimation sample is a subset of the sample used to create the spatial weighting matrix
impower(#)	order of instrumental-variable approximation
Reporting	
level(#)	set confidence level; default is level(95)
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Optimization	
optimization_options	control the optimization process; seldom used
coeflegend	display legend instead of statistics

ml_options	Description
Model	
* `ml`	use maximum likelihood estimator
`dvarlag(`*spmatname*`)`	spatially lagged dependent variable; not repeatable
`errorlag(`*spmatname*`)`	spatially lagged errors; not repeatable
`ivarlag(`*spmatname* `:` *varlist*`)`	spatially lagged independent variables; repeatable
`noconstant`	suppress constant term
`constraints(`*constraints*`)`	apply specified linear constraints
`force`	allow estimation when estimation sample is a subset of the sample used to create the spatial weighting matrix
`gridsearch(#)`	resolution of the initial-value search grid; seldom used
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `oim` or `robust`
Reporting	
`level(#)`	set confidence level; default is `level(95)`
`nocnsreport`	do not display constraints
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process; seldom used
`coeflegend`	display legend instead of statistics

* You must specify either `gs2sls` or `ml`.

indepvars and *varlist* specified in `ivarlag()` may contain factor variables; see [U] **11.4.3 Factor variables**.

`coeflegend` does not appear in the dialog box.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options for spregress, gs2sls

Model

`gs2sls` requests that the generalized spatial two-stage least-squares estimator be used.

`dvarlag(`*spmatname*`)` specifies a spatial weighting matrix that defines a spatial lag of the dependent variable. This option is repeatable to allow higher-order models. By default, no spatial lags of the dependent variable are included.

`errorlag(`*spmatname*`)` specifies a spatial weighting matrix that defines a spatially lagged error. This option is repeatable to allow higher-order models. By default, no spatially lagged errors are included.

`ivarlag(`*spmatname* `:` *varlist*`)` specifies a spatial weighting matrix and a list of independent variables that define spatial lags of the variables. This option is repeatable to allow spatial lags created from different matrices. By default, no spatial lags of the independent variables are included.

`noconstant`; see [R] **Estimation options**.

`heteroskedastic` specifies that the estimator treat the errors as heteroskedastic instead of homoskedastic, which is the default; see *Methods and formulas*.

force requests that estimation be done when the estimation sample is a proper subset of the sample used to create the spatial weighting matrices. The default is to refuse to fit the model. Weighting matrices potentially connect all the spatial units. When the estimation sample is a subset of this space, the spatial connections differ and spillover effects can be altered. In addition, the normalization of the weighting matrix differs from what it would have been had the matrix been normalized over the estimation sample. The better alternative to force is first to understand the spatial space of the estimation sample and, if it is sensible, then create new weighting matrices for it. See [SP] **spmatrix** and *Missing values, dropped observations, and the W matrix* in [SP] **Intro 2**.

impower(#) specifies the order of an instrumental-variable approximation used in fitting the model. The derivation of the estimator involves a product of # matrices. Increasing # may improve the precision of the estimation and will not cause harm, but will require more computer time. The default is impower(2). See *Methods and formulas* for additional details on impower(#).

Reporting

level(#); see [R] **Estimation options**.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(*style*), cformat(%*fmt*), pformat(%*fmt*), sformat(%*fmt*), and nolstretch; see [R] **Estimation options**.

Optimization

optimization_options: iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), and nonrtolerance; see [M-5] **optimize( )**.

The following option is available with spregress, gs2sls but is not shown in the dialog box:

coeflegend; see [R] **Estimation options**.

Options for spregress, ml

Model

ml requests that the maximum likelihood estimator be used.

dvarlag(*spmatname*) specifies a spatial weighting matrix that defines a spatial lag of the dependent variable. Only one dvarlag() option may be specified. By default, no spatial lags of the dependent variable are included.

errorlag(*spmatname*) specifies a spatial weighting matrix that defines a spatially lagged error. Only one errorlag() option may be specified. By default, no spatially lagged errors are included.

ivarlag(*spmatname* : *varlist*) specifies a spatial weighting matrix and a list of independent variables that define spatial lags of the variables. This option is repeatable to allow spatial lags created from different matrices. By default, no spatial lags of the independent variables are included.

noconstant, constraints(*constraints*); see [R] **Estimation options**.

force requests that estimation be done when the estimation sample is a proper subset of the sample used to create the spatial weighting matrices. The default is to refuse to fit the model. This is the same force option described for use with spregress, gs2sls.

gridsearch(#) specifies the resolution of the initial-value search grid. The default is gridsearch(0.1). You may specify any number between 0.001 and 0.1 inclusive.

SE/Robust

vce(*vcetype*) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim) and that are robust to nonnormal independent and identically distributed (i.i.d.) disturbance (robust). See [R] ***vce_option***.

Reporting

level(#), nocnsreport; see [R] **Estimation options**.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(*style*), cformat(%*fmt*), pformat(%*fmt*), sformat(%*fmt*), and nolstretch; see [R] **Estimation options**.

Maximization

maximize_options: difficult, technique(*algorithm_spec*), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), and nonrtolerance; see [R] **Maximize**.

The following option is available with spregress, ml but is not shown in the dialog box:

coeflegend; see [R] **Estimation options**.

Remarks and examples

Remarks are presented under the following headings:

Introduction

See [SP] **Intro 1**–[SP] **Intro 8** for an overview of SAR models. The introductions also describe, in detail and with examples, how to prepare your data for analysis with spregress and the other Sp estimation commands.

Datasets for SAR models contain observations on geographical areas or other units; all that is required is that there be some measure of distance that distinguishes which units are close to each other. The spregress command models cross-sectional data. It requires each observation to represent one unique spatial unit. For data with multiple observations on each unit—namely, panel data—see [SP] **spxtregress**.

To fit models with endogenous regressors for cross-sectional data, see [SP] **spivregress**.

spregress, gs2sls uses a generalized method of moments estimator known as generalized spatial two-stage least squares (GS2SLS). spregress, ml uses a maximum likelihood (ML) estimator. For normally distributed data, ml is theoretically more efficient than gs2sls, but when data are i.i.d., spregress, gs2sls produces results that are not appreciably different from those of spregress, ml. See *Methods and formulas*.

The vce(robust) variance estimator can be used with spregress, ml to produce standard errors that are robust to nonnormal i.i.d. errors; see [R] ***vce_option***. spregress, ml can produce inconsistent estimates with data that are not identically distributed.

spregress, gs2sls has a heteroskedastic option that relaxes the assumption that errors are i.i.d. With the heteroskedastic option, errors only need to be independent; see example 2.

Choosing weighting matrices and their normalization

Weighting matrices

It is important to understand that the choice of weighting matrices is part of your SAR model specification.

The choice of weighting matrix should be based on the formulation of your research question. Does it make sense to define spatial lags based on only neighboring areas? Or do you want to model effects across distances that decrease with increasing distance? Or do you want to model spatial lags based on some measure in your data, for example, the value of imports and exports between countries?

The Sp system has the spmatrix create command, which can create contiguity matrices and inverse-distance matrices. For instance, typing

```
spmatrix create contiguity W
```

creates a symmetric weighting matrix, W, that has the same positive weight for contiguous spatial units and, by default, a zero weight for all other units, with an option to include nonzero weights for second-order neighbors (neighbors of neighbors). There are also Sp commands for creating custom weighting matrices. See [SP] **Intro 2** and [SP] **spmatrix** for details.

Both spregress, gs2sls and spregress, ml can fit models with multiple spatial lags of the independent variables. You can specify multiple ivarlag() options with different spatial weighting matrices for the same or different variables.

With the gs2sls estimator, you can also include dependent-variable spatial lags and autoregressive error terms specified by two or more spatial weighting matrices. You do this by specifying multiple dvarlag() options or multiple errorlag() options. Multiple weighting matrices can be viewed as providing a "higher-order" approximation to the true dependent variable or error spatial dependence, and they allow testing of the formulation of the spatial lag.

With the ml estimator, you can include only one dvarlag() and one errorlag(), but each can have its own, possibly different, spatial weighting matrix.

Normalization of weighting matrices

spmatrix create by default normalizes the weighting matrix it creates by dividing the entries by the absolute value of the largest eigenvalue of the matrix; this is the normalize(spectral) option. The normalize(minmax) option scales the matrix using either the maximum of column sums or the maximum of the row sums, whichever is smaller. The normalize(row) option scales each row of the matrix by its row sum, so that each row sums to one.

You may have also created your own weighting matrix with good properties for the estimator. In this case, you may want to leave the matrix unnormalized using the normalize(none) option.

What are the differences among the three normalizations?

There are two reasons to normalize: interpretability of the spatial lag coefficients and estimability.

normalize(spectral) and normalize(minmax) produce matrices that differ from the original only by a scalar multiple. This not true for normalize(row), so let's discuss it first.

Row normalization, `normalize(row)`, has a long history and is popular in applied work. Row normalization can potentially multiply different rows by different scalars, and if it does so, that changes the model specification given by the weighting matrix. For example, if you start with a contiguity matrix, and the first row has two 1s and the second row has four 1s, then after row normalization, the first row contains two halves and the second four quarters. This amounts to spreading the potential spillover effects of each spatial unit equally across its neighbors, whereas the original unnormalized contiguity matrix modeled equal potential spillover effects for each neighbor regardless of the number of neighbors. `normalize(row)` also transforms a symmetric contiguity matrix into an asymmetric matrix. Row normalization should be used when the spatial lags it specifies are appropriate for your research question and when the lags of the original matrix are not.

When the unnormalized matrix has been formulated to match your research question, there is the choice of `normalize(spectral)`, `normalize(minmax)`, or `normalize(none)`. The choice affects the interpretation of the spatial lag coefficients.

Because dependent-variable spatial lags enter the model as $\lambda\mathbf{Wy}$, covariate lags enter as $\gamma\mathbf{Wx}$, and the autoregressive errors are modeled using $\rho\mathbf{We}$, we would expect the spatial lag coefficient estimates to scale inversely by the scale of $\mathbf{W}$. If $\mathbf{W}$ is scaled by c to become $\mathbf{W}/c$, then $\widehat{\lambda}$ becomes $c\widehat{\lambda}$, $\widehat{\gamma}$ becomes $c\widehat{\gamma}$, and $\widehat{\rho}$ becomes $c\widehat{\rho}$.

For example, if an unnormalized matrix results in an estimation of $\widehat{\rho}_{\text{unnorm}} = 0.1$, and if the matrix is then scaled by $c = 5$, the estimation using the normalized matrix would yield $\widehat{\rho}_{\text{norm}} = 0.5$. So what we want for the interpretation of the parameter estimate is a scaling where $\widehat{\rho}_{\text{norm}}$ is typically in the range -1 to 1. Recall from the discussion in [SP] **Intro 2** and [SP] **Intro 7** that ρ is not a true correlation, only something like a correlation. There is no guarantee that the estimate for it will be between -1 and 1. In an explosive model, the estimate will be outside this range.

The scaling factor c from `normalize(spectral)` is always less than or equal to the scaling factor from `normalize(minmax)`. So for the same model run with different normalizations, `minmax` will result in an estimate $\widehat{\rho}_{\text{minmax}}$ that is larger than $\widehat{\rho}_{\text{spectral}}$, the estimate resulting from using `spectral`. So the spectral normalization is more likely to produce estimates of ρ in the range -1 to 1.

The second reason for normalization is estimability. The scaling from `normalize(spectral)` guarantees nonsingularity of certain terms in the model estimation; see *Methods and formulas*. The bigger scaling of `normalize(minmax)`, of course, also guarantees nonsingularity, but it is a bigger scaling than necessary.

Row normalization also guarantees nonsingularity, but because it is not a scalar multiple of the unnormalized matrix, we cannot in general say how it will change the spatial lag coefficient estimates relative to the estimates produced using the unnormalized matrix. Row normalization, as we said earlier, results in a different model specification.

You may have created your own weighting matrix, and you know that based on its properties and the form of the estimator that it will not yield singularities. In this case, you need not normalize. If an unnormalized matrix, however, causes a singularity in the estimation, you may get "wrong" estimation results, that is, ones differing by other than a scale factor from those using a spectral or min–max normalization.

`spmatrix create` and other Sp matrix commands use spectral normalization by default because it is the smallest scaling that in general guarantees nonsingularity without changing the model specification of the original matrix. However, `normalize(spectral)` is computationally expensive. It can take a long time for large matrices. If this is a consideration, `normalize(minmax)` is faster to compute and will yield results that are close to those of `normalize(spectral)`.

Direct and indirect effects and normalization

Direct and indirect, also called spillover, effects were discussed in [SP] **Intro 1** and [SP] **Intro 2**. In example 1 below, we show how to get these estimates using the `estat impact` command.

The scaling property between the spectral and min–max normalizations and the spatial lag coefficient estimates that we described in the previous section implies that the estimates of the direct and indirect effects should be scale invariant. `spregress, ml` has this scaling property and gives scale-invariant effects. When there is no autoregressive error term, `spregress, gs2sls` also has this scaling property and gives scale-invariant effects. When there is an autoregressive error term, however, the GS2SLS estimator is only asymptotically scale invariant.

Practically speaking, this means when you use `estat impact` to look at the direct and indirect effects of the covariates after `spregress, ml` in all cases, or `spregress, gs2sls` with no `errorlag()`, you will get results differing only by numerical precision whether you used `normalize(spectral)`, `normalize(minmax)`, or an unnormalized matrix with sound numerical properties.

The GS2SLS estimator, however, is a nonlinear function of the weighting matrix when an autoregressive error term is included. For this nonlinear GS2SLS estimator, models are well defined only if the coefficient on the spatial lag of the dependent variable and the coefficient on the spatially lagged error lie within certain intervals. Normalizing the weighting matrix by the spectral normalization or the row normalization puts the estimates in these intervals when there are no higher-order lags. Because min–max normalization is a close approximation to spectral normalization, the resulting estimates should be close.

Again, practically speaking, this means that even though `normalize(spectral)` and `normalize(minmax)` both simply multiply the original matrix by a scalar, and the scalars are similar in size, `estat impact` may give slightly different estimates depending on the normalization for the GS2SLS estimator with an autoregressive error term. This is especially the case in small samples, and the differences will decrease as the sample size increases.

Of course, the `normalize(row)` normalization will yield different estimates of effects compared with the other normalizations or with no normalization because row normalization results in a different model specification.

In higher-order models with GS2SLS and autoregressive error terms, the estimator is a nonlinear function of multiple weighting matrices. The sets of spatial lag coefficients for which the models are well defined are multidimensional regions, but the same normalizations are used, and the tradeoffs mentioned above still apply.

Examples

▷ Example 1: A spatial autoregressive model

We want to model the homicide rate in counties in southern states of the United States. `homicide1990.dta` contains `hrate`, the county-level homicide rate per year per 100,000 persons; `ln_population`, the logarithm of the county population; `ln_pdensity`, the logarithm of the population density; and `gini`, the Gini coefficient for the county, a measure of income inequality where larger values represent more inequality (Gini 1909). The data are an extract of the data originally used by Messner et al. (2000); see Britt (1994) for a literature review of the topic.

We used `spshape2dta` to create `homicide1990.dta` and `homicide1990_shp.dta`. The latter file contains the boundary coordinates for U.S. southern counties. See [SP] **Intro 4**, [SP] **Intro 7**, [SP] **spshape2dta**, and [SP] **spset**.

Because the analysis dataset and the Stata-formatted shapefile must be in our working directory to spset the data, we first save both homicide1990.dta and homicide1990_shp.dta to our working directory by using the copy command. We then load the data and type spset to see the Sp settings.

```
. copy https://www.stata-press.com/data/r16/homicide1990.dta .
. copy https://www.stata-press.com/data/r16/homicide1990_shp.dta .
. use homicide1990
(S.Messner et al.(2000), U.S southern county homicide rates in 1990)
. spset
  Sp dataset homicide1990.dta
                data:  cross sectional
    spatial-unit id:  _ID
        coordinates:  _CX, _CY (planar)
  linked shapefile:  homicide1990_shp.dta
```

We plot the homicide rate on a map of the counties by using the grmap command; see [SP] **grmap**. Figure 1 is the result.

```
. grmap hrate
```

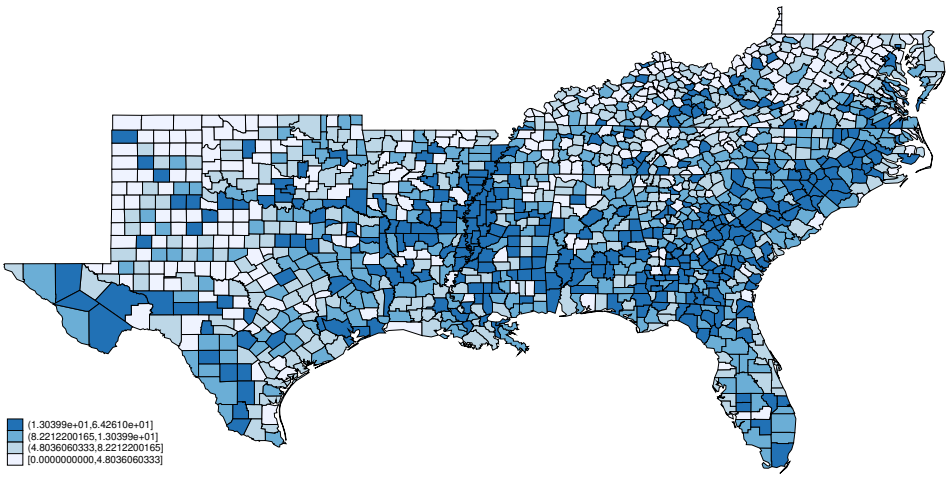


Figure 1: Homicide rate in 1990 for southern U.S. counties

The homicide rate appears to be spatially dependent because the high homicide-rate counties appear to be clustered together. As described in [SP] **Intro 7**, we can fit an ordinary linear regression and test whether the errors are spatially correlated using the Moran test.

To conduct the test, we need a spatial weighting matrix. We will create one that puts the same positive weight on contiguous counties and a zero weight on all other counties—a matrix known as a contiguity matrix. We will use the default spectral normalization for the matrix. See [SP] **Intro 2**, [SP] **spmatrix create**, and *Choosing weighting matrices and their normalization* above for details. We type

```
. spmatrix create contiguity W
```

To create W, spmatrix used the coordinate data in homicide1990_shp.dta behind the scenes.

Now, we run `regress` and then `estat moran`.

```
. regress hrate
```

Source	SS	df	MS	Number of obs	=	1,412
				F(0, 1411)	=	0.00
Model	0	0	.	Prob > F	=	.
Residual	69908.59	1,411	49.5454217	R-squared	=	0.0000
				Adj R-squared	=	0.0000
Total	69908.59	1,411	49.5454217	Root MSE	=	7.0389

hrate	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
_cons	9.549293	.1873201	50.98	0.000	9.181837	9.916749

```
. estat moran, errorlag(W)

Moran test for spatial dependence
        Ho: error is i.i.d.
        Errorlags:  W

        chi2(1)      =   265.84
        Prob > chi2  =   0.0000
```

The test reports that we can reject that the errors are i.i.d. and confirms our visual appraisal of the data.

To model the homicide rate `hrate`, we will use the GS2SLS estimator and specify the option `dvarlag(W)` to fit a model with a spatial lag of `hrate` based on `W`.

```
. spregress hrate ln_population ln_pdensity gini, gs2sls dvarlag(W)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrix defines 1412 places)
```

Spatial autoregressive model		Number of obs	=	1,412
GS2SLS estimates		Wald chi2(4)	=	328.40
		Prob > chi2	=	0.0000
		Pseudo R2	=	0.1754

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	.195714	.2654999	0.74	0.461	-.3246563	.7160843
ln_pdensity	1.060728	.2303736	4.60	0.000	.6092043	1.512252
gini	77.10293	5.330446	14.46	0.000	66.65544	87.55041
_cons	-28.79865	2.945944	-9.78	0.000	-34.57259	-23.02471
W						
hrate	.2270154	.0607158	3.74	0.000	.1080146	.3460161

Wald test of spatial terms: chi2(1) = 13.98 Prob > chi2 = 0.0002

The estimated coefficient on the spatial lag of `hrate` is 0.23, indicating positive correlation between the homicide rate in one county and the homicide rate in a neighboring county.

As we discussed in [SP] **Intro 7** the coefficients cannot be interpreted as they are in standard regression models. We can use `estat impact` to interpret results, but first we will illustrate how to fit other SAR models.

We now include a spatial autoregressive error term by adding `errorlag(W)`.

```
. spregress hrate ln_population ln_pdensity gini, gs2sls dvarlag(W) errorlag(W)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrix defines 1412 places)
Estimating rho using 2SLS residuals:
initial:       GMM criterion =  16.837319
alternative:   GMM criterion =  10.842722
rescale:       GMM criterion =  1.1522691
Iteration 0:   GMM criterion =  1.1522691
Iteration 1:   GMM criterion =  1.1386586
Iteration 2:   GMM criterion =  1.1386578
Iteration 3:   GMM criterion =  1.1386578
Estimating rho using GS2SLS residuals:
Iteration 0:   GMM criterion =  .02771702
Iteration 1:   GMM criterion =   .0262056
Iteration 2:   GMM criterion =  .02606375
Iteration 3:   GMM criterion =  .02601873
Iteration 4:   GMM criterion =  .02601004
Iteration 5:   GMM criterion =  .02600789
Iteration 6:   GMM criterion =  .02600742
Iteration 7:   GMM criterion =  .02600731
Iteration 8:   GMM criterion =  .02600729
Spatial autoregressive model                    Number of obs     =      1,412
GS2SLS estimates                                Wald chi2(4)      =     276.72
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1736
------------------------------------------------------------------------------
       hrate |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
hrate        |
ln_populat~n |   .1034997   .2810656     0.37   0.713    -.4473787    .6543781
 ln_pdensity |   1.081404   .2520505     4.29   0.000     .5873939    1.575414
        gini |    82.0687   5.658372    14.50   0.000      70.9785     93.1589
       _cons |  -29.63033   3.070332    -9.65   0.000    -35.64807   -23.61259
-------------+----------------------------------------------------------------
W            |
       hrate |   .1937419   .0654322     2.96   0.003     .0654972    .3219867
     e.hrate |   .3555443   .0786465     4.52   0.000        .2014    .5096887
------------------------------------------------------------------------------
Wald test of spatial terms:          chi2(2) = 226.21     Prob > chi2 = 0.0000
. estimates store gs2sls_model
```

Note that when an autoregressive error term is included, the estimation procedure becomes an iterative generalized method of moments procedure.

We keep the SAR error term `e.hrate` in our model and add terms representing spatial lags of the independent variables by using `ivarlag(W: ...)`.

```
. spregress hrate ln_population ln_pdensity gini, gs2sls dvarlag(W) errorlag(W)
> ivarlag(W: ln_population ln_pdensity gini)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrix defines 1412 places)
```

(*output omitted*)

```
Spatial autoregressive model                    Number of obs     =      1,412
GS2SLS estimates                                Wald chi2(7)      =     394.61
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1866
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	-.3489221	.3050009	-1.14	0.253	-.9467129	.2488687
ln_pdensity	1.210485	.3015442	4.01	0.000	.6194695	1.801501
gini	89.17773	6.454876	13.82	0.000	76.5264	101.8291
_cons	-28.80191	3.178656	-9.06	0.000	-35.03196	-22.57186
W						
ln_populat~n	1.918436	.4598247	4.17	0.000	1.017196	2.819676
ln_pdensity	-1.260725	.5326521	-2.37	0.018	-2.304704	-.2167459
gini	-43.4606	8.607378	-5.05	0.000	-60.33075	-26.59045
hrate	.5071798	.1139532	4.45	0.000	.2838356	.730524
e.hrate	-.3135187	.1396411	-2.25	0.025	-.5872103	-.0398271

```
Wald test of spatial terms:          chi2(5) = 61.81      Prob > chi2 = 0.0000
```

The coefficients for the lagged variables and the autoregressive error term are significant.

We are often unsure which spatial weighting matrix should be used to compute spatial lags. Many researchers use a spatial weighting matrix whose (i, j)th element is the inverse of the distance between units i and j. This inverse-distance matrix has many nice properties and a long history in spatial analysis; see [SP] **spmatrix** and *Choosing weighting matrices and their normalization* above.

With the GS2SLS estimator, we can include spatial lags using two spatial weighting matrices, in which case we might view them as together providing a "higher-order" approximation to the true spatial process. We had in our model a spatial lag of the dependent variable using a contiguity matrix alone. Now, we will include that and another lag of the dependent variable using an inverse-distance matrix.

We create the inverse-distance matrix `M` with the default spectral normalization and use `spmatrix dir` to list our Sp matrices.

```
. spmatrix create idistance M
. spmatrix dir
```

Weighting matrix name	N x N	Type	Normalization
M	1412 x 1412	idistance	spectral
W	1412 x 1412	contiguity	spectral

Now, we add dvarlag(M) to our model.

```
. spregress hrate ln_population ln_pdensity gini, gs2sls dvarlag(W) errorlag(W)
> ivarlag(W: ln_population ln_pdensity gini) dvarlag(M)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrices define 1412 places)
```

(*output omitted*)

```
Spatial autoregressive model                    Number of obs     =      1,412
GS2SLS estimates                                Wald chi2(8)      =    1323.43
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1121
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	-.6245268	.2830848	-2.21	0.027	-1.179363	-.0696908
ln_pdensity	1.266528	.2831372	4.47	0.000	.7115889	1.821466
gini	69.30288	5.64501	12.28	0.000	58.23887	80.3669
_cons	-19.77152	2.753498	-7.18	0.000	-25.16827	-14.37476
W						
ln_populat~n	2.590823	.3806543	6.81	0.000	1.844754	3.336892
ln_pdensity	-2.63202	.4261689	-6.18	0.000	-3.467296	-1.796744
gini	-59.75958	6.438899	-9.28	0.000	-72.37959	-47.13957
hrate	.9269411	.0492867	18.81	0.000	.830341	1.023541
e.hrate	-.8531151	.0914652	-9.33	0.000	-1.032384	-.6738465
M						
hrate	.2289787	.0755038	3.03	0.002	.080994	.3769634

```
Wald test of spatial terms:          chi2(6) = 676.93     Prob > chi2 = 0.0000
```

The hrate lag specified by M is significant in addition to the hrate lag specified by W. We may well want to include both in our final model.

We could repeat the process, fitting a model with errorlag(M) in addition to errorlag(W), and another model with ivarlag(M: ...) in addition to ivarlag(W: ...). One issue is that we have "only" $N = 1412$ spatial units (observations) in this example. To fit higher-order lags, one needs lots of spatial units, so we need to exercise judgment just as in any other model-building process. In our final model, we keep a single weighting matrix for each term. We use W for dvarlag() and ivarlag(), but M for errorlag().

```
. spregress hrate ln_population ln_pdensity gini, gs2sls dvarlag(W) errorlag(M)
> ivarlag(W: ln_population ln_pdensity gini)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrices define 1412 places)
```

(*output omitted*)

```
Spatial autoregressive model                    Number of obs     =      1,412
GS2SLS estimates                                Wald chi2(7)      =     357.06
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1241
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	-.0475582	.3295548	-0.14	0.885	-.6934737	.5983573
ln_pdensity	.8989538	.3211524	2.80	0.005	.2695066	1.528401
gini	89.91969	6.409286	14.03	0.000	77.35772	102.4817
_cons	-32.21599	3.590014	-8.97	0.000	-39.25229	-25.17969
W						
ln_populat~n	2.679931	.5218152	5.14	0.000	1.657192	3.702669
ln_pdensity	-2.468953	.6209688	-3.98	0.000	-3.686029	-1.251876
gini	-57.38302	9.418108	-6.09	0.000	-75.84217	-38.92387
hrate	.6818566	.1141573	5.97	0.000	.4581125	.9056007
M						
e.hrate	.9533048	.1324392	7.20	0.000	.6937289	1.212881

```
Wald test of spatial terms:          chi2(5) = 169.23     Prob > chi2 = 0.0000
. estimates store model_ex1_last
```

In [SP] **Intro 7**, we cautioned that interpreting covariate effects based on their coefficient estimates is difficult when there is a dependent-variable lag or an independent-variable lag in the model.

The spatial lag of `hrate` modifies the covariate effects. A change in `gini` in a county changes the conditional mean of `hrate` in that county, and that change in `hrate` changes the conditional mean of `hrate` in all contiguous counties. The change in `hrate` in these counties then affects `hrate` in all counties contiguous to them, and so on, until all counties linked by a chain of contiguous counties are affected.

The effects of a covariate vary over the counties because of how the spatial lag of `hrate` modifies the covariate effects. There are as many effects of a covariate as there are spatial units. As discussed by LeSage and Pace (2009, sec. 2.7), we define the average of these spatial unit-level effects to be the covariate effect.

The effect of `gini` on the conditional mean of `hrate` in other counties is called an indirect, or spillover, effect.

Because a spatial lag of `gini` is included in the model, there is a second indirect effect. The equation for `hrate` includes a term for `gini` in neighboring counties, so a change in `gini` in one county changes the conditional mean of `hrate` in neighboring counties.

The effect of `gini` on the conditional mean of `hrate` in the same county is called a direct, or own, effect. The sum of the direct and indirect effects is called the total effect.

We use estat impact to estimate the magnitude of these effects.

```
. estat impact
progress   : 33%  67% 100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
ln_populat~n	.3149608	.3545409	0.89	0.374	-.3799266	1.009848
ln_pdensity	.6448149	.3426066	1.88	0.060	-.0266817	1.316311
gini	90.45773	6.380729	14.18	0.000	77.95173	102.9637
indirect						
ln_populat~n	5.856241	2.256561	2.60	0.009	1.433463	10.27902
ln_pdensity	-4.105437	1.883462	-2.18	0.029	-7.796956	-.413919
gini	8.691593	19.58268	0.44	0.657	-29.68975	47.07294
total						
ln_populat~n	6.171202	2.411894	2.56	0.011	1.443976	10.89843
ln_pdensity	-3.460622	2.029163	-1.71	0.088	-7.437708	.5164636
gini	99.14932	21.03394	4.71	0.000	57.92356	140.3751

See the percentages at the top of the output? For large datasets, calculating standard errors of the effects can be time consuming, so estat impact reports its progress as it does the computations.

The direct effect of gini is positive because the coefficient of gini is positive. The indirect effect of gini due to the spatial lag of hrate is positive because the coefficient of the dependent-variable lag is positive and the coefficient of gini is positive. The indirect effect of gini due to its spatial lag, however, is negative because the coefficient of its lag is negative. estat impact shows that the two indirect effects of gini sum to a net positive indirect effect, although the sum is not significantly different from 0.

Note that the normalization of W affects the size of the coefficient estimates for the lags of the covariates. For the GS2SLS estimator, the normalization of W (except for the case of row normalization) does not affect the asymptotic estimates of the covariate effects. In finite samples, this means that the normalization of W may have a small effect on the estimates produced by estat impact—small compared with the effect's standard error. For the ML estimator, the normalization does not affect the size of estimated effects shown by estat impact. See *Choosing weighting matrices and their normalization*.

Running estat impact after spregress is essential for proper interpretation of the model. The output of estat impact can be read directly as the change in the metric of the dependent variable per incremental change of the covariate averaged across all the spatial units (observations).

estat impact shows marginal (incremental change) effects. We might want to see the total effect of a discrete change in a covariate. The expectation of the dependent variable is linear in the covariates in this example. We did not fit polynomial or other nonlinear terms. We could just multiply the incremental change by the discrete change of the covariate. Or, we could use the margins command, which works for both linear and nonlinear terms; see [R] **margins**.

The median of gini is 0.39, its 25th percentile is 0.37, and its 75th percentile is 0.41. So it is reasonable to ask how a change of ± 0.02 in the Gini coefficient affects the homicide rate. Here's how to get the answer by using margins:

```
. margins, at(gini = generate(gini - 0.02)) at(gini = generate(gini))
> at(gini = generate(gini + 0.02))
Predictive margins                              Number of obs     =      1,412
Expression   : Reduced-form mean, predict()
1._at        : gini            = gini - 0.02
2._at        : gini            = gini
3._at        : gini            = gini + 0.02
```

	Margin	Delta-method Std. Err.	z	P>|z|	[95% Conf.	Interval]
_at						
1	2.550868	2.651383	0.96	0.336	-2.645746	7.747482
2	4.533855	2.584986	1.75	0.079	-.5326253	9.600334
3	6.516841	2.586198	2.52	0.012	1.447986	11.5857

A change of ± 0.02 in the Gini coefficient causes the homicide rate to change by roughly ± 2.0 per 100,000 persons per year.

The computations that `margins` must do to calculate standard errors can sometimes be time consuming. Time will depend on the complexity of the spatial model and the number of spatial units in the data. You may want to fit your model with a subsample of your data, run `margins`, and extrapolate to estimate the time required to run `margins` on the full sample. See [P] **timer** and [P] **rmsg**.

◁

▷ Example 2: spregress, gs2sls heteroskedastic

The `spregress, gs2sls` command has a `heteroskedastic` option that requires the errors to be independent but not necessarily identically distributed. Practically speaking, this option causes the estimates of the spatial autoregressive error correlations and the standard errors to change. In models without spatially autoregressive errors, only standard errors will change. See *Methods and formulas*.

If we add the `heteroskedastic` option to the last model we fit in example 1, we get

```
. spregress hrate ln_population ln_pdensity gini, gs2sls heteroskedastic
> dvarlag(W) errorlag(M) ivarlag(W: ln_population ln_pdensity gini)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrices define 1412 places)
  (output omitted)
```

```
Spatial autoregressive model                    Number of obs     =      1,412
GS2SLS estimates                                Wald chi2(7)      =     248.74
                                                Prob > chi2       =     0.0000
                                                Pseudo R2         =     0.1241
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	-.0475582	.3545931	-0.13	0.893	-.7425479	.6474315
ln_pdensity	.8989538	.4016155	2.24	0.025	.1118019	1.686106
gini	89.91969	10.71501	8.39	0.000	68.91866	110.9207
_cons	-32.21599	5.013344	-6.43	0.000	-42.04197	-22.39002
W						
ln_populat~n	2.679931	.5247129	5.11	0.000	1.651512	3.708349
ln_pdensity	-2.468953	.6786844	-3.64	0.000	-3.79915	-1.138756
gini	-57.38302	9.719208	-5.90	0.000	-76.43232	-38.33372
hrate	.6818566	.13258	5.14	0.000	.4220047	.9417085
M						
e.hrate	.9614507	.1554489	6.18	0.000	.6567764	1.266125

```
Wald test of spatial terms:          chi2(5) = 156.95     Prob > chi2 = 0.0000
. estimates store heterosk_model
```

We used `estimates store` to store the results of the earlier model, and we stored this model, too. We can now use `estimates table` to display coefficient estimates with their standard errors side by side. See [R] **estimates store** and [R] **estimates table**.

```
. estimates table model_ex1_last heterosk_model, b(%6.3f) se(%6.3f)
```

Variable	model~t	heter~l
hrate		
ln_populat~n	-0.048	-0.048
	0.330	0.355
ln_pdensity	0.899	0.899
	0.321	0.402
gini	89.920	89.920
	6.409	10.715
_cons	-32.216	-32.216
	3.590	5.013
W		
ln_populat~n	2.680	2.680
	0.522	0.525
ln_pdensity	-2.469	-2.469
	0.621	0.679
gini	-57.383	-57.383
	9.418	9.719
hrate	0.682	0.682
	0.114	0.133
M		
e.hrate	0.953	0.961
	0.132	0.155

legend: b/se

We see that standard errors are larger, especially those for the direct-effect coefficients of the covariates. We also see that the estimate of ρ, the SAR error correlation labeled as `e.hrate`, differs between the two estimators.

◁

▷ Example 3: spregress, ml

SAR models can be fit using ML estimation. Here's the second model we fit in example 1 estimated using `ml` in place of `gs2sls`.

```
. spregress hrate ln_population ln_pdensity gini, ml dvarlag(W) errorlag(W)
  (1412 observations)
  (1412 observations (places) used)
  (weighting matrix defines 1412 places)
Performing grid search ... finished
Optimizing concentrated log likelihood:
Iteration 0:   log likelihood =  -4557.201
Iteration 1:   log likelihood =  -4556.763
Iteration 2:   log likelihood = -4556.7539
Iteration 3:   log likelihood = -4556.7539
Optimizing unconcentrated log likelihood:
Iteration 0:   log likelihood = -4556.7539
Iteration 1:   log likelihood = -4556.7539  (backed up)
```

```
Spatial autoregressive model                    Number of obs     =      1,412
Maximum likelihood estimates                    Wald chi2(4)      =     240.21
                                                Prob > chi2       =     0.0000
Log likelihood = -4556.7539                     Pseudo R2         =     0.1590
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	.5268247	.3038837	1.73	0.083	-.0687763	1.122426
ln_pdensity	.5269135	.3136226	1.68	0.093	-.0877755	1.141603
gini	91.44471	6.263932	14.60	0.000	79.16763	103.7218
_cons	-32.8348	3.205075	-10.24	0.000	-39.11663	-26.55297
W						
hrate	-.1850846	.1218453	-1.52	0.129	-.423897	.0537279
e.hrate	.6244211	.0897639	6.96	0.000	.4484871	.8003551
var(e.hrate)	34.79054	1.599235			31.79315	38.07052

```
Wald test of spatial terms:          chi2(2) = 227.84     Prob > chi2 = 0.0000
. estimates store ml_model
```

We stored the estimation results with `estimates store`, as we did with the same model fit with `gs2sls`, and now we use `estimates table` to compare coefficient estimates and their standard errors.

```
. estimates table gs2sls_model ml_model, b(%6.3f) se(%6.3f)
```

Variable	gs2sl~l	ml_mo~l
hrate		
ln_populat~n	0.103	0.527
	0.281	0.304
ln_pdensity	1.081	0.527
	0.252	0.314
gini	82.069	91.445
	5.658	6.264
_cons	-29.630	-32.835
	3.070	3.205
W		
hrate	0.194	-0.185
	0.065	0.122
e.hrate	0.356	0.624
	0.079	0.090
var(e.hrate)		34.791
		1.599

legend: b/se

There are meaningful differences in the results. The coefficient of `ln_pdensity` was significant in the GS2SLS model but is nonsignificant in the ML model. The coefficient estimates for `gini`, however, are similar, as are their standard errors. The coefficient of the lag of `hrate` becomes negative in the ML model, and the SAR error correlation increases from $\rho = 0.36$ to $\rho = 0.62$.

We note that the ML estimator is not consistent under heteroskedasticity; for consistency, the error distribution needs to be i.i.d., although it need not be normal. Heteroskedasticity may be the reason why the estimates differ as they do. See Arraiz et al. (2010).

◁

Stored results

`spregress, gs2sls` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k)`	number of parameters
`e(df_m)`	model degrees of freedom
`e(df_c)`	degrees of freedom for test of spatial terms
`e(iterations)`	number of generalized method of moments iterations
`e(iterations_2sls)`	number of two-stage least-squares iterations
`e(rank)`	rank of `e(V)`
`e(r2_p)`	pseudo-R^2
`e(chi2)`	χ^2
`e(chi2_c)`	χ^2 for test of spatial terms
`e(p)`	p-value for model test
`e(p_c)`	p-value for test of spatial terms
`e(converged)`	`1` if generalized method of moments converged, `0` otherwise
`e(converged_2sls)`	`1` if two-stage least-squares converged, `0` otherwise

Macros

`e(cmd)`	`spregress`
`e(cmdline)`	command as typed
`e(depvar)`	name of dependent variable
`e(indeps)`	names of independent variables
`e(idvar)`	name of ID variable
`e(estimator)`	`gs2sls`
`e(title)`	title in estimation output
`e(constant)`	`hasconstant` or `noconstant`
`e(exogr)`	exogenous regressors
`e(dlmat)`	names of spatial weighting matrices applied to *depvar*
`e(elmat)`	names of spatial weighting matrices applied to errors
`e(het)`	`heteroskedastic` or `homoskedastic`
`e(chi2type)`	`Wald`; type of model χ^2 test
`e(properties)`	`b V`
`e(estat_cmd)`	program used to implement `estat`
`e(predict)`	program used to implement `predict`
`e(marginsok)`	predictions allowed by `margins`
`e(marginsnotok)`	predictions disallowed by `margins`
`e(asbalanced)`	factor variables `fvset` as `asbalanced`
`e(asobserved)`	factor variables `fvset` as `asobserved`

Matrices

`e(b)`	coefficient vector
`e(delta_2sls)`	two-stage least-squares estimates of coefficients in spatial lag equation
`e(rho_2sls)`	generalized method of moments estimates of coefficients in spatial error equation
`e(V)`	variance–covariance matrix of the estimators

Functions

`e(sample)`	marks estimation sample

`spregress, ml` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k)`	number of parameters
`e(df_m)`	model degrees of freedom
`e(df_c)`	degrees of freedom for test of spatial terms
`e(ll)`	log likelihood
`e(iterations)`	number of maximum log-likelihood estimation iterations
`e(rank)`	rank of `e(V)`
`e(r2_p)`	pseudo-R^2
`e(chi2)`	χ^2
`e(chi2_c)`	χ^2 for test of spatial terms
`e(p)`	p-value for model test

e(p_c)	p-value for test of spatial terms
e(converged)	1 if converged, 0 otherwise
Macros	
e(cmd)	spregress
e(cmdline)	command as typed
e(depvar)	name of dependent variable
e(indeps)	names of independent variables
e(idvar)	name of ID variable
e(estimator)	ml
e(title)	title in estimation output
e(constant)	hasconstant or noconstant
e(dlmat)	name of spatial weighting matrix applied to *depvar*
e(elmat)	name of spatial weighting matrix applied to errors
e(chi2type)	Wald; type of model χ^2 test
e(vce)	*vcetype* specified in vce()
e(vcetype)	title used to label Std. Err.
e(ml_method)	type of ml method
e(technique)	maximization technique
e(properties)	b V
e(estat_cmd)	program used to implement estat
e(predict)	program used to implement predict
e(marginsok)	predictions allowed by margins
e(marginsnotok)	predictions disallowed by margins
e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved
Matrices	
e(b)	coefficient vector
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(Hessian)	Hessian matrix
e(V)	variance–covariance matrix of the estimators
Functions	
e(sample)	marks estimation sample

Methods and formulas

SAR models date back to the works of Whittle (1954) and Cliff and Ord (1973, 1981). Cressie (1993), LeSage and Pace (2009), and Waller and Gotway (2004) provide textbook introductions. Spatial models have been applied in a variety of disciplines, such as criminology, demography, economics, epidemiology, political science, and public health. See Darmofal (2015), Waller and Gotway (2004), Kelejian and Prucha (2010), Drukker, Egger, and Prucha (2013), and Lee, Liu, and Lin (2010) for examples in economics, social science, and public health, including examples of nongeographic models such as social interactions and social networks.

The GS2SLS estimator was derived by Kelejian and Prucha (1998, 1999, 2010) and extended by Arraiz et al. (2010) and Drukker, Egger, and Prucha (2013).

The formulas for the GS2SLS without higher-order spatial weighting matrices were published in Drukker, Prucha, and Raciborski (2013). For the higher-order models, spregress, gs2sls implements the estimator derived in Badinger and Egger (2011) and Prucha, Drukker, and Egger (2016).

The properties of the ML estimator were proven by Lee (2004), which also provides the formulas for the robust estimator of the VCE.

Methods and formulas are presented under the following headings:

Model

We consider a cross-sectional spatial autoregressive model with autoregressive disturbances (SARAR), allowing for higher-order spatial dependence in the dependent variable, exogenous independent variables, and spatial errors. The model is

$$\begin{aligned}\mathbf{y} &= \sum_{k=1}^{K}\beta_k\mathbf{x}_k + \sum_{p=1}^{P}\gamma_p\mathbf{W}_p\,\mathbf{x}_p + \sum_{r=1}^{R}\lambda_r\mathbf{W}_r\,\mathbf{y} + \mathbf{u}\\ \mathbf{u} &= \sum_{s=1}^{S}\rho_s\mathbf{M}_s\mathbf{u} + \boldsymbol{\epsilon}\end{aligned} \tag{1}$$

where

$\mathbf{y}$ is an $n \times 1$ vector of observations on the dependent variable;

$\mathbf{x}_k$ is an $n \times 1$ vector of observations on the exogenous variable; β_k is the corresponding scalar parameter;

$\mathbf{W}_p$, $\mathbf{W}_r$, and $\mathbf{M}_s$ are $n \times n$ spatial weighting matrices with 0 diagonal elements;

$\mathbf{W}_p\,\mathbf{x}_p$, $\mathbf{W}_r\,\mathbf{y}$, and $\mathbf{M}_s\mathbf{u}$ are $n \times 1$ vectors typically referred to as spatial lags for the exogenous variable, dependent variable, and error term; γ_p, λ_r, and ρ_s are scalar parameters; and

$\boldsymbol{\epsilon}$ is an $n \times 1$ vector of innovations (i.i.d. disturbances).

The model in (1) is frequently referred to as a higher-order spatial autoregressive model with spatial autoregressive disturbances, or namely, a SARAR(R, S) model.

The spatial weighting matrices $\mathbf{W}_p$, $\mathbf{W}_r$, and $\mathbf{M}_s$ are assumed to be known and nonstochastic. See [SP] **Intro 2** and Darmofal (2015, chap. 2) for an introduction to spatial weighting matrices, and see Kelejian and Prucha (2010) for a technical discussion of how normalization affects parameter definition.

The scalar parameters γ_p and λ_r measure the degree to which the dependent variable depends on its neighboring covariate's values and outcomes. See example 1 and LeSage and Pace (2009, sec. 2.7) for discussions of effect estimation.

The innovations $\boldsymbol{\epsilon}$ are assumed to be i.i.d. or independent but heteroskedastically distributed, where the heteroskedasticity is of unknown form. The errors $\mathbf{u}$ are spatially autoregressive.

The GS2SLS estimator produces consistent estimates in both cases when the `heteroskedastic` option is specified. For the first-order SARAR model, see Kelejian and Prucha (1998, 1999, 2010), Arraiz et al. (2010), and Drukker, Egger, and Prucha (2013) for formal results and discussions; for the higher-order SARAR(R, S) model, see Badinger and Egger (2011) for formal results. The ML estimator is consistent in the i.i.d. case for the SARAR(1, 1) model but generally not consistent in the heteroskedastic case. See Lee (2004) for some results for the ML estimator; see Arraiz et al. (2010) for evidence that the ML estimator does not produce consistent estimates in the heteroskedastic case.

The GS2SLS estimator can fit the SARAR(R, S) model, whereas the ML estimator can only fit the SARAR$(1, 1)$ model.

GS2SLS estimator

In this section, we give a detailed description of the computations performed by `spregress, gs2sls`. For the SARAR$(1, 1)$ model, `spregress, gs2sls` implements the estimator described in Kelejian and Prucha (2010), Arraiz et al. (2010), and Drukker, Egger, and Prucha (2013); for the SARAR(R, S) model, `spregress, gs2sls` implements the estimator described in Badinger and Egger (2011). We will describe the GS2SLS estimator for the SARAR(R, S) model, which generalizes the first-order SARAR model.

Let's first rewrite (1) in a compact form:

$$\begin{aligned} \mathbf{y} &= \mathbf{X}\beta + \overline{\mathbf{X}}\gamma + \overline{\mathbf{Y}}\lambda + \mathbf{u} = \mathbf{Z}\delta + \mathbf{u} \\ \mathbf{u} &= \overline{\mathbf{U}}\rho + \boldsymbol{\epsilon} \end{aligned} \tag{2}$$

where

$\mathbf{X} = [\mathbf{x}_k]_{k=1,\ldots,K}$ is an $n \times K$ matrix of exogenous covariates;

$\overline{\mathbf{X}} = [\mathbf{W}_p\,\mathbf{x}_p]_{p=1,\ldots,P}$ is an $n \times P$ matrix of spatial lags for the exogenous covariates;

$\overline{\mathbf{Y}} = [\mathbf{W}_r\,\mathbf{y}]_{r=1,\ldots,R}$ is an $n \times R$ matrix of spatial lags for the dependent variables;

$\overline{\mathbf{U}} = [\mathbf{M}_s\mathbf{u}]_{s=1,\ldots,S}$ is an $n \times S$ matrix of spatial lags for the error term;

$\mathbf{Z} = \left[\mathbf{X}, \overline{\mathbf{X}}, \overline{\mathbf{Y}}\right]$ is an $n \times (K + P + R)$ matrix;

β, γ, λ, and ρ denote the $K \times 1$, $P \times 1$, $R \times 1$, and $S \times 1$ vectors of coefficients corresponding to $\mathbf{X}$, $\overline{\mathbf{X}}$, $\overline{\mathbf{Y}}$, and $\overline{\mathbf{U}}$, respectively; and

$\delta = (\beta', \gamma', \lambda')'$ is a $(K + P + R) \times 1$ vector of coefficients for $\mathbf{Z}$.

In the following, we review the two-stage least-squares (2SLS), generalized spatial two-stage least-squares (GS2SLS), and GMM estimation approaches as discussed in Badinger and Egger (2011).

2SLS estimator of δ

In the first step, we apply 2SLS to (2) using an instrument matrix $\mathbf{H}_1$ to estimate δ. The 2SLS estimator of δ—say, $\widetilde{\delta}$—is defined as

$$\widetilde{\delta} = \left(\widetilde{\mathbf{Z}}'\mathbf{Z}\right)^{-1}\widetilde{\mathbf{Z}}'\mathbf{y}$$

where $\widetilde{\mathbf{Z}} = \mathbf{P}_{\mathbf{H}_1}\mathbf{Z}$ and $\mathbf{P}_{\mathbf{H}_1} = \mathbf{H}_1\left(\mathbf{H}_1'\mathbf{H}_1\right)^{-1}\mathbf{H}_1'$. The 2SLS estimator $\widetilde{\delta}$ depends on the instrument matrix $\mathbf{H}_1$. Let $\mathbf{X}_\mathrm{f}$ denote all the exogenous regressors; that is, $\mathbf{X}_\mathrm{f} = \left[\mathbf{X}, \overline{\mathbf{X}}\right]$ in our case. The instrument matrix $\mathbf{H}_1$ contains the linearly independent columns in

$$\mathbf{H}_1 = \left[\mathbf{X}_\mathrm{f}, \mathbf{W}^1\mathbf{X}_\mathrm{f}, \ldots, \mathbf{W}^q\mathbf{X}_\mathrm{f}\right]$$

where $\mathbf{W}^1 \equiv \{W_r\}_{r=1,\ldots,R}$ denotes all the spatial weighting matrices applied to the dependent variable, and $\mathbf{W}^q \equiv \left\{\mathbf{W}_{j_1}\mathbf{W}_{j_2}\ldots\mathbf{W}_{j_q}\right\}_{j_1,j_2,\ldots,j_q=1,\ldots,R}$ denotes the product of q matrices from W^1 in any possible permutation order.

The `impower(#)` option specifies q, the number of the power in $\mathbf{W}^q$. The default is `impower(2)`. Increasing q may improve the precision of the estimation of δ.

We now illustrate the construction of $\mathbf{H}_1$ with an example. Suppose we use two spatial weighting matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ to generate the spatial lags for the dependent variable. So $\mathbf{W}^1 = (\mathbf{W}_1, \mathbf{W}_2)$. If we have $q = 2$, then $\mathbf{W}^2 = (\mathbf{W}_1\mathbf{W}_1, \mathbf{W}_1\mathbf{W}_2, \mathbf{W}_2\mathbf{W}_1, \mathbf{W}_2\mathbf{W}_2)$. Plug $\mathbf{W}^1$ and $\mathbf{W}^2$ into the definition of $\mathbf{H}_1$, and the instrument matrix $\mathbf{H}_1$ in this special case contains the linear independent columns in the following matrix:

$$\mathbf{H}_1 = [\mathbf{X}_\mathrm{f}, \mathbf{W}_1\mathbf{X}_\mathrm{f}, \mathbf{W}_2\mathbf{X}_\mathrm{f}, \mathbf{W}_1\mathbf{W}_1\mathbf{X}_\mathrm{f}, \mathbf{W}_1\mathbf{W}_2\mathbf{X}_\mathrm{f}, \mathbf{W}_2\mathbf{W}_1\mathbf{X}_\mathrm{f}, \mathbf{W}_2\mathbf{W}_2\mathbf{X}_\mathrm{f}]$$

GMM estimator of ρ based on 2SLS residuals

The initial GMM estimates of ρ solve the sample equivalent of the population moment conditions

$$\begin{aligned}(1/N)\, E(\boldsymbol{\epsilon}'\mathbf{A}_s\boldsymbol{\epsilon}) &= 0\\ (1/N)\, E(\boldsymbol{\epsilon}'\mathbf{B}_s\boldsymbol{\epsilon}) &= 0 \qquad \text{for} \quad s \in \{1, \ldots, S\}\end{aligned}$$

where $\mathbf{A}_s = \mathbf{M}_s$ and $\mathbf{B}_s = \mathbf{M}_s'\mathbf{M}_s - \mathrm{diag}(\mathbf{M}_s'\mathbf{M}_s)$. See the estimator derived in Badinger and Egger (2011) and Prucha, Drukker, and Egger (2016) for details.

GS2SLS estimator of δ

The GS2SLS estimator of δ is based on the spatially Cochrane–Orcutt-transformed model.

$$\mathbf{y}_{nt} = \mathbf{Z}_*(\rho)\, \delta + \boldsymbol{\epsilon} \tag{3}$$

where $\mathbf{y}_{nt} = (\mathbf{I}_n - \sum_{s=1}^{S} \rho_s\mathbf{M}_s)\mathbf{y}$, $\mathbf{Z}_*(\rho) = (\mathbf{I}_n - \sum_{s=1}^{S} \rho_s\mathbf{M}_s)\mathbf{Z}$, and $\mathbf{I}_n$ is an $n \times n$ identity matrix.

Now, we apply the 2SLS estimator to (3) by using an instrument matrix $\mathbf{H}_2$ and replacing ρ with $\widetilde{\rho}$. The GS2SLS estimator of δ—say, $\widehat{\delta}$—is defined as

$$\widehat{\delta} = \left\{\widehat{\mathbf{Z}_*(\widetilde{\rho})}'\mathbf{Z}_*(\widetilde{\rho})\right\}^{-1} \widehat{\mathbf{Z}_*(\widetilde{\rho})}'\mathbf{y}_*(\widetilde{\rho})$$

where

$\mathbf{y}_*(\widetilde{\rho}) = (\mathbf{I}_n - \sum_{s=1}^{S} \widetilde{\rho}_s\mathbf{M}_s)\mathbf{y}$,

$\mathbf{Z}_*(\widetilde{\rho}) = (\mathbf{I}_n - \sum_{s=1}^{S} \widetilde{\rho}_s\mathbf{M}_s)\mathbf{Z}$,

$\widehat{\mathbf{Z}_*(\widetilde{\rho})} = \mathbf{P}_{\mathbf{H_2}}\mathbf{Z}_*(\widetilde{\rho})$, and

$\mathbf{P}_{\mathbf{H_2}} = \mathbf{H}_2\,(\mathbf{H}_2'\mathbf{H}_2)^{-1}\,\mathbf{H}_2'$.

The instrument matrix $\mathbf{H}_2$ contains the linearly independent columns in

$$\mathbf{H}_2 = [\mathbf{H}_1, \mathbf{M_1}\mathbf{H}_1, \ldots, \mathbf{M_S}\mathbf{H}_1]$$

Efficient GMM estimator of ρ based on GS2SLS residuals

The form of the efficient GMM weighting matrix is given in Badinger and Egger (2011) and Prucha, Drukker, and Egger (2016). The matrix has one form in the default homoskedastic case and another in the heteroskedastic case. The form of the matrix causes the estimates of spatially autoregressive error correlations and the standard errors to differ when the `heteroskedastic` option is specified.

ML estimator

We implement a quasi–maximum likelihood (QML) estimator for the first-order SARAR model. We can write SARAR(1, 1) [see (1)] as

$$\begin{aligned}\mathbf{y} &= \mathbf{X}\beta + \overline{\mathbf{X}}\gamma + \lambda\mathbf{W}\mathbf{y} + \mathbf{u} = \mathbf{X}_{\mathrm{f}}\zeta + \lambda\mathbf{W}\mathbf{y} + \mathbf{u}\\ \mathbf{u} &= \rho\mathbf{M}\mathbf{u} + \boldsymbol{\epsilon}\end{aligned} \tag{4}$$

where

$\mathbf{X}_{\mathrm{f}} = [\mathbf{X}, \overline{\mathbf{X}}]$ is an $n \times L$ matrix containing exogenous covariates and spatial lags for the exogenous variables, with $L = K + P$;

$\zeta = (\beta', \gamma')'$ is an $L \times 1$ vector of coefficients;

$\mathbf{W}$ and $\mathbf{M}$ are $n \times n$ spatial weighting matrices with 0 diagonal elements; and

λ and ρ are scalar parameters.

Log-likelihood function

We give the log-likelihood function assuming that $\boldsymbol{\epsilon} \sim N(0, \sigma^2\mathbf{I}_n)$. Lee (2004) gives formal results on the consistency and asymptotic normality of the QML estimator when the innovations are i.i.d. but not necessarily normally distributed. Violations of the assumption that the innovations are i.i.d. can cause the QML estimator to produce inconsistent results.

The reduced form of (4) is

$$\mathbf{y} = (\mathbf{I}_n - \lambda\mathbf{W})^{-1}\mathbf{X}_{\mathrm{f}}\,\zeta + (\mathbf{I}_n - \lambda\mathbf{W})^{-1}(\mathbf{I}_n - \rho\mathbf{M})^{-1}\boldsymbol{\epsilon}$$

The unconcentrated log-likelihood function is

$$\begin{aligned}\ln L(\mathbf{y}\,|\,\zeta, \lambda, \rho, \sigma^2) = &-\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma^2) + \ln||\mathbf{I}_n - \lambda\mathbf{W}|| + \ln||\mathbf{I}_n - \rho\mathbf{M}||\\ &+ \frac{1}{2\sigma^2}\left\{(\mathbf{I}_n - \lambda\mathbf{W})\mathbf{y} - \mathbf{X}_{\mathrm{f}}\,\zeta\right\}'(\mathbf{I}_n - \rho\mathbf{M})'(\mathbf{I}_n - \rho\mathbf{M})\left\{(\mathbf{I}_n - \lambda\mathbf{W})\mathbf{y} - \mathbf{X}_{\mathrm{f}}\,\zeta\right\}\end{aligned} \tag{5}$$

We can concentrate the log-likelihood function by taking first-order derivatives with respect to ζ and σ^2 in (5) and setting them to 0, yielding the maximizers

$$\begin{aligned}\widehat{\zeta}(\lambda, \rho) &= \left\{\mathbf{X}_{\mathrm{f}}'(\mathbf{I}_n - \rho\mathbf{M})'(\mathbf{I}_n - \rho\mathbf{M})\mathbf{X}_{\mathrm{f}}\right\}^{-1}\mathbf{X}_{\mathrm{f}}'(\mathbf{I}_n - \rho\mathbf{M})'(\mathbf{I}_n - \rho\mathbf{M})(\mathbf{I}_n - \lambda\mathbf{W})\mathbf{y}\\ \widehat{\sigma^2}(\lambda, \rho) &= \frac{1}{n}\left\{(\mathbf{I}_n - \lambda\mathbf{W})\mathbf{y} - \mathbf{X}_{\mathrm{f}}\,\widehat{\zeta}(\lambda, \rho)\right\}'(\mathbf{I}_n - \rho\mathbf{M})'(\mathbf{I}_n - \rho\mathbf{M})\\ &\quad\times\left\{(\mathbf{I}_n - \lambda\mathbf{W})\mathbf{y} - \mathbf{X}_{\mathrm{f}}\,\widehat{\zeta}(\lambda, \rho)\right\}\end{aligned}$$

Substituting $\widehat{\zeta}(\lambda,\rho)$ and $\widehat{\sigma^2}(\lambda,\rho)$ into the log-likelihood function in (5), we have the concentrated log-likelihood function

$$\ln L_c(\mathbf{y}\,|\,\lambda,\rho) = -\frac{n}{2}\{\ln(2\pi)+1\} - \frac{n}{2}\ln\{\sigma^2(\lambda,\rho)\} + \ln||\mathbf{I}_n - \lambda\mathbf{W}|| + \ln||\mathbf{I}_n - \rho\mathbf{M}||$$

The QML estimates for $\widehat{\lambda}$ and $\widehat{\rho}$ can be computed by maximizing the concentrated log likelihood. Then, we can calculate the QML estimates for ζ and σ^2 as $\widehat{\zeta}(\widehat{\lambda},\widehat{\rho})$ and $\widehat{\sigma^2}(\widehat{\lambda},\widehat{\rho})$.

`spregress, ml` uses a grid search to find reasonable initial values for λ and ρ.

The formula for the robust VCE is given in Lee (2004).

Pseudo-R^2

The pseudo-R^2 is calculated as $\{\text{corr}(y,\widehat{y})\}^2$, where $\widehat{y}$ is the reduced-form prediction of y.

References

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

Badinger, H., and P. H. Egger. 2011. Estimation of higher-order spatial autoregressive cross-section models with heteroscedastic disturbances. *Papers in Regional Science* 90: 213–235.

Britt, C. L. 1994. Crime and unemployment among youths in the United States, 1958–1990: A time series analysis. *American Journal of Economics and Sociology* 53: 99–109.

Cliff, A. D., and J. K. Ord. 1973. *Spatial Autocorrelation*. London: Pion.

——. 1981. *Spatial Processes: Models and Applications*. London: Pion.

Cressie, N. 1993. *Statistics for Spatial Data*. Rev. ed. New York: Wiley.

Darmofal, D. 2015. *Spatial Analysis for the Social Sciences*. New York: Cambridge University Press.

Drukker, D. M., P. H. Egger, and I. R. Prucha. 2013. On two-step estimation of a spatial autoregressive model with autoregressive disturbances and endogenous regressors. *Econometric Reviews* 32: 686–733.

Drukker, D. M., I. R. Prucha, and R. Raciborski. 2013. Maximum likelihood and generalized spatial two-stage least-squares estimators for a spatial-autoregressive model with spatial-autoregressive disturbances. *Stata Journal* 13: 221–241.

Gini, C. 1909. Concentration and dependency ratios (in Italian). English translation in *Rivista di Politica Economica 1997* 87: 769–789.

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

——. 1999. A generalized moments estimator for the autoregressive parameter in a spatial model. *International Economic Review* 40: 509–533.

——. 2010. Specification and estimation of spatial autoregressive models with autoregressive and heteroskedastic disturbances. *Journal of Econometrics* 157: 53–67.

Lee, L.-F. 2004. Asymptotic distributions of quasi-maximum likelihood estimators for spatial autoregressive models. *Econometrica* 72: 1899–1925.

Lee, L.-F., X. Liu, and X. Lin. 2010. Specification and estimation of social interaction models with network structures. *Econometrics Journal* 13: 145–176.

LeSage, J., and R. K. Pace. 2009. *Introduction to Spatial Econometrics*. Boca Raton, FL: Chapman & Hall/CRC.

Messner, S. F., L. Anselin, D. F. Hawkins, G. Deane, S. E. Tolnay, and R. D. Baller. 2000. An Atlas of the Spatial Patterning of County-Level Homicide, 1960–1990. Pittsburgh: National Consortium on Violence Research.

Prucha, I. R., D. M. Drukker, and P. H. Egger. 2016. Simultaneous equations models with higher-order spatial or social network interactions. Working paper, Department of Economics, University of Maryland. http://econweb.umd.edu/~prucha/papers/WP_IRP_PHE_DMD_2016.pdf.

Waller, L. A., and C. A. Gotway. 2004. *Applied Spatial Statistics for Public Health Data*. Hoboken, NJ: Wiley.

Whittle, P. 1954. On stationary processes in the plane. *Biometrika* 434–449.

Also see

[SP] **spregress postestimation** — Postestimation tools for spregress

[SP] **estat moran** — Moran's test of residual correlation with nearby residuals

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spivregress** — Spatial autoregressive models with endogenous covariates

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spxtregress** — Spatial autoregressive models for panel data

[R] **regress** — Linear regression

[U] **20 Estimation and postestimation commands**

Title

spregress postestimation — Postestimation tools for spregress

Postestimation commands

The following postestimation command is of special interest after `spregress`:

Command	Description
`estat impact`	direct, indirect, and total impacts

The following postestimation commands are also available:

Command	Description
`contrast`	contrasts and ANOVA-style joint tests of estimates
* `estat ic`	Akaike's and Schwarz's Bayesian information criteria (AIC and BIC)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estimates`	cataloging estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of coefficients
* `lrtest`	likelihood-ratio test
`margins`	marginal means, predictive margins, marginal effects, and average marginal effects
`marginsplot`	graph the results from margins (profile plots, interaction plots, etc.)
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
`predict`	predictions, residuals, influence statistics, and other diagnostic measures
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of estimates
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

* `estat ic` and `lrtest` are not appropriate with `gs2sls` estimation results.

predict

Description for predict

`predict` creates a new variable containing predictions such as the reduced-form mean, the direct mean, the indirect mean, the limited-information mean, the full-information mean, the naïve-form prediction, the linear prediction, the residuals, or the uncorrelated residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] *newvar* [*if*] [*in*] [`,` *statistic*]

statistic	Description
Main	
`rform`	reduced-form mean; the default
`direct`	direct mean
`indirect`	indirect mean
`limited`	limited-information mean
`full`	full-information mean
`naive`	naïve-form prediction
`xb`	linear prediction
`residuals`	residuals
`ucresiduals`	uncorrelated residuals

These statistics are only available in a subset of the estimation sample.

Options for predict

Main

`rform`, the default, calculates the reduced-form mean. It is the predicted mean of the dependent variable conditional on the independent variables and any spatial lags of the independent variables. See *Methods and formulas*.

`direct` calculates the direct mean. It is a unit's predicted contribution to its own reduced-form mean. The direct and indirect means sum to the reduced-form mean.

`indirect` calculates the indirect mean. It is the predicted sum of the other units' contributions to a unit's reduced-form mean.

`limited` calculates the limited-information mean. It is the predicted mean of the dependent variable conditional on the independent variables, any spatial lags of the independent variables, and any spatial lags of the dependent variable. `limited` is not available when the `heteroskedastic` option is used with `spregress, gs2sls`.

`full` calculates the full-information mean. It is the predicted mean of the dependent variable conditional on the independent variables, any spatial lags of the independent variables, and the other units' values of the dependent variable. `full` is not available when the `heteroskedastic` option is used with `spregress, gs2sls`.

`naive` calculates the naïve-form prediction. It is the predicted linear combination of the independent variables, any spatial lags of the independent variables, and any spatial lags of the dependent variable. It is not a consistent estimator of an expectation. See *Methods and formulas*.

`xb` calculates the predicted linear combination of the independent variables.

`residuals` calculates the residuals, including any autoregressive error term.

`ucresiduals` calculates the uncorrelated residuals, which are estimates of the uncorrelated error term.

margins

Description for margins

`margins` estimates margins of response for the reduced-form mean, direct mean, indirect mean, and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

`margins` [*marginlist*] [`,` *options*]

`margins` [*marginlist*] `,` `predict(`*statistic* ...`)` [`predict(`*statistic* ...`)` ...] [*options*]

statistic	Description
`rform`	reduced-form mean; the default
`direct`	direct mean
`indirect`	indirect mean
`xb`	linear prediction
`limited`	not allowed with `margins`
`full`	not allowed with `margins`
`naive`	not allowed with `margins`
`residuals`	not allowed with `margins`
`ucresiduals`	not allowed with `margins`

Statistics not allowed with `margins` are functions of stochastic quantities other than `e(b)`.

For the full syntax, see [R] **margins**.

Remarks for margins

The computations that `margins` must do to calculate standard errors can sometimes be time consuming. Time will depend on the complexity of the spatial model and the number of spatial units in the data. You may want to fit your model with a subsample of your data, run `margins`, and extrapolate to estimate the time required to run `margins` on the full sample. See [P] **timer** and [P] **rmsg**.

estat impact

Description for estat impact

`estat impact` estimates the mean of the direct, indirect, and total impacts of independent variables on the reduced-form mean of the dependent variable.

Syntax for estat impact

`estat impact` [*varlist*] [*if*] [*in*] [`, nolog vce(`*vcetype*`)`]

varlist is a list of independent variables, including factor variables, taken from the fitted model. By default, all independent variables from the fitted model are used.

Options for estat impact

Main

`nolog` suppresses the calculation progress log that shows the percentage completed. By default, the log is displayed.

VCE

`vce(`*vcetype*`)` specifies how the standard errors of the impacts are calculated.

`vce(delta)`, the default, is the delta method and treats the independent variables as fixed.

`vce(unconditional)` specifies that standard errors account for sampling variance in the independent variables. This option is not available when `if` or `in` is specified with `estat impact`.

Remarks for estat impact

`estat impact` is essential for interpreting the output of `spregress`. See [SP] **Intro 7** and example 1 of [SP] **spregress** for explanations and examples.

Stored results for estat impact

`estat impact` stores the following in `r()`:

Scalars

`r(N)`	number of observations

Macros

`r(vce)`	*vcetype* specified in `vce()`
`r(xvars)`	names of independent variables

Matrices

`r(b_direct)`	vector of estimated direct impacts
`r(Jacobian_direct)`	Jacobian matrix for direct impacts
`r(V_direct)`	estimated variance–covariance matrix of direct impacts
`r(b_indirect)`	vector of estimated indirect impacts
`r(Jacobian_indirect)`	Jacobian matrix for indirect impacts
`r(V_indirect)`	estimated variance–covariance matrix of indirect impacts
`r(b_total)`	vector of estimated total impacts
`r(Jacobian_total)`	Jacobian matrix for total impacts
`r(V_total)`	estimated variance–covariance matrix of total impacts

Methods and formulas

Methods and formulas are presented under the following headings:

Predictions

To motivate the predictions, consider the vector form of a spatial autoregressive model

$$\mathbf{y} = \lambda \mathbf{W}\mathbf{y} + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{1}$$

where

$\mathbf{y}$ is the vector containing each unit's dependent-variable observation,

$\mathbf{W}\mathbf{y}$ is a spatial lag of $\mathbf{y}$,

$\mathbf{X}$ is the matrix of independent-variable observations,

$\boldsymbol{\epsilon}$ is a vector of errors, and

λ and $\boldsymbol{\beta}$ are the coefficients.

Any spatial lags of the independent variables are assumed to be in $\mathbf{X}$. Spatial lags of the error do not affect the reduced-form, direct, or indirect means, so they are not included in (1) for simplicity.

Reduced-form mean

Equation (1) represents the spatial autoregressive model as a system of equations. The solution

$$\mathbf{y} = (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}) \tag{2}$$

implies that the mean of $\mathbf{y}$ given the independent variables and the spatial weighting matrix is

$$E(\mathbf{y} \mid \mathbf{X}, \mathbf{W}) = (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}\boldsymbol{\beta}) \tag{3}$$

This is known as the reduced-form mean because the solution in (2) is known as the reduced form of the model. The predicted reduced-form mean substitutes estimates of λ and $\boldsymbol{\beta}$ into (3).

Direct and indirect means

To define the direct mean and the indirect mean, let

$$\mathbf{S} = (\mathbf{I} - \lambda\mathbf{W})^{-1}$$

and let $\mathbf{S}_d$ be a matrix with diagonal elements of $\mathbf{S}$ on its diagonal and with off-diagonal elements set to 0.

The direct means are

$$\mathbf{S}_d\mathbf{X}\boldsymbol{\beta}$$

which capture the contributions of each unit's independent variables on its own reduced-form mean. Substituting estimates of λ and $\boldsymbol{\beta}$ produces the predictions.

The indirect means capture the contributions of the other units' independent variables on a unit's reduced-form prediction, and they are

$$\left\{(\mathbf{I} - \lambda\mathbf{W})^{-1} - \mathbf{S}_d\right\}\mathbf{X}\boldsymbol{\beta}$$

Limited-information mean

Instead of solving for the reduced form, the limited-information mean conditions on the spatial lag of y for observation i, which we denote by $(\mathbf{W}\,\mathbf{y})_i$, which yields

$$E\{y_i \mid \mathbf{X}, \mathbf{W}, (\mathbf{W}\,\mathbf{y})_i\} = \mathbf{x}_i\boldsymbol{\beta} + \lambda(\mathbf{W}\,\mathbf{y})_i + u_i \tag{4}$$

where u_i is the predictable part of the error term given $(\mathbf{W}\,\mathbf{y})_i$. See Kelejian and Prucha (2007) and Drukker, Prucha, and Raciborski (2013).

Full-information mean

The full-information mean conditions on the dependent-variable values of all the other units instead of conditioning on the spatial lag of the dependent variable, as does the limited-information mean. The additional information produces a better prediction of the error term when a spatial lag of the errors is in the model. See Kelejian and Prucha (2007).

Naïve-form predictor

The naïve-form predictor sets u_i to 0 in (4). It is not consistent for $E\{y_i \mid \mathbf{X}, \mathbf{W}, (\mathbf{W}\,\mathbf{y})_i\}$ because it ignores u_i.

Linear predictor

The linear predictor is $\mathbf{X}\beta$.

Residuals

The residuals are u_i from (4).

Uncorrelated residuals

The uncorrelated residuals are

$$\widehat{\boldsymbol{\epsilon}} = (\mathbf{I} - \widehat{\rho}\mathbf{M})^{-1}\,\mathbf{u}$$

where $\mathbf{u}$ is the vector of u_i's, $\mathbf{M}$ is the spatial weighting matrix for the autoregressive error term, and $\widehat{\rho}$ is the estimated correlation of $\mathbf{u}$.

Impacts

The total impact of an independent variable $\mathbf{x}$ is the average of the marginal effects it has on the reduced-form mean,

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})}{\partial x_j}$$

where $E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})$ is the ith element of the vector $E(\mathbf{y} \mid \mathbf{X}, \mathbf{W})$, whose formula is given in (2), and x_j is the jth unit's value for $\mathbf{x}$.

The direct impact of an independent variable $\mathbf{x}$ is the average of the direct, or own, marginal effects:

$$\frac{1}{n}\sum_{i=1}^{n}\frac{\partial E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})}{\partial x_i}$$

The indirect impact of an independent variable $\mathbf{x}$ is the average of the indirect, or spillover, marginal effects:

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\frac{\partial E(\mathbf{y}_i \mid \mathbf{X}, \mathbf{W})}{\partial x_j}$$

LeSage and Pace (2009, 36–37) call the average direct impact the "average total direct impact", and they call the average indirect impact the "average total indirect impact".

`estat impact` with the default `vce(delta)` uses the delta method to calculate the estimated variance of the impacts. This variance is conditional on the values of the independent variables in the model.

`estat impact` with `vce(unconditional)` uses the generalized method of moments estimation strategy to estimate the unconditional variance of the impacts. It accounts for sampling variance of the independent variables in the model.

References

Drukker, D. M., I. R. Prucha, and R. Raciborski. 2013. Maximum likelihood and generalized spatial two-stage least-squares estimators for a spatial-autoregressive model with spatial-autoregressive disturbances. *Stata Journal* 13: 221–241.

Kelejian, H. H., and I. R. Prucha. 2007. The relative efficiencies of various predictors in spatial econometric models containing spatial lags. *Regional Science and Urban Economics* 37: 363–374.

LeSage, J., and R. K. Pace. 2009. *Introduction to Spatial Econometrics*. Boca Raton, FL: Chapman & Hall/CRC.

Liu, D. 2017. How to create animated graphics to illustrate spatial spillover effects. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2018/03/06/how-to-create-animated-graphics-to-illustrate-spatial-spillover-effects/.

Also see

[SP] **spregress** — Spatial autoregressive models

[U] **20 Estimation and postestimation commands**

Title

spset — Declare data to be Sp spatial data

Description

Data must be `spset` before you can use the other Sp commands. The `spset` command serves three purposes:

1. It reports whether the data are `spset` and if so, how.
2. It sets the spatial data for the first time.
3. It modifies how the data are `spset` at any time.

Data that are `spset` are called Sp data.

Quick start

Query whether or how data are `spset`

```
spset
```

In cross-sectional data, specify geographic unit identifier `id`

```
spset id
```

Add coordinates stored in variables `x` and `y` to previously `spset` data

```
spset, modify coord(x y)
```

In panel data, specify geographic unit identifier `id` and time within area identifier `tvar`

```
xtset id tvar
spset id
```

Menu

Statistics > Spatial autoregressive models

Syntax

Display the current setting

```
spset
```

Set data with shapefiles

```
spshape2dta ...
```
(see [SP] **spshape2dta**)

Set data without shapefiles

```
spset idvar [ , options ]
```

Modify how data are set with shapefiles

```
spset [idvar], modify [shpmodoptions]
```

Modify how data are set without shapefiles

```
spset, modify [modoptions]
```

Clear the setting

```
spset, clear
```

idvar is an existing, numeric variable that uniquely identifies the geographic units, meaning the observations in cross-sectional data and the panels in panel data.

shapefile refers to a Stata-format shapefile, specified with or without the `.dta` suffix. Such files usually have names of the form *name*`_shp.dta`.

options	Description
`coord(`*xvar yvar*`)`	designate *xvar* and *yvar* as the location coordinates
`coordsys(`*coordsys*`)`	specify how coordinates are interpreted

shpmodoptions	Description
`coordsys(`*coordsys*`)`	change how coordinates are interpreted
`noshpfile`	break link with shapefile
`replace`	replace current geographic identifier with *idvar*

modoptions	Description
`coord(`*xvar yvar*`)`	replace location coordinates with *xvar* and *yvar*
`coordsys(`*coordsys*`)`	change how coordinates are interpreted
`nocoord`	clear coordinate settings
`shpfile(`*shapefile*`)`	establish link to shapefile

Options

`coord(`*xvar yvar*`)` and `nocoord` specify coordinates. `coord()` specifies the variables recording the x and y coordinates or the longitude and latitude. `nocoord` specifies that previously set coordinates be forgotten.

`coord(`*xvar yvar*`)` creates or replaces the contents of Sp variables `_CX` and `_CY`.

`coord()` and `nocoord` are allowed only if the data are not linked to a shapefile. If you want to use different coordinates than the shapefile provides, break the connection to the shapefile by typing

```
. spset, modify noshpfile
```

and then use `spset, modify coord(`*xvar yvar*`)`. You can later use `spset, modify shpfile(`*shapefile*`)` to reestablish the link. Relinking to the shapefiles reestablishes the original coordinates stored in `_CX` and `_CY`.

`coordsys(`*coordsys*`)` specifies how to interpret coordinates. You may specify `coordsys()` regardless of whether you are linked to a shapefile. `coordsys()` syntax is

`coordsys(planar)`	(default)
`coordsys(latlong)`	(kilometers implied)
`coordsys(latlong, kilometers)`	
`coordsys(latlong, miles)`	

`coordsys(latlong)` specifies latitude and longitude coordinates. `kilometers` and `miles` specify the units in which distances should be calculated. Distances for `planar` coordinates are always in the units of the planar coordinates.

`modify` specifies that existing `spset` settings are to be modified. Omitting `modify` means that the data are being `spset` for the first time.

You can modify Sp settings as often as you wish.

`clear` clears all Sp settings. It drops the variables `_ID`, `_CX`, and `_CY` that `spset` previously created.

`replace` replaces the current geographic identifier with *idvar*.

`noshpfile` breaks the link to the Stata-format shapefile, the file that usually has *shapefile*`_shp.dta`. Data that were linked to a shapefile will be just as if they had never been linked to it. Before breaking the link, you should make a note of the shapefile's name:

```
. spset            (make a note of the shapefile's name)
. spset, modify noshpfile
```

The shapefile might have been named *shapefile*`_shp.dta`. You will need the name later should you wish to reestablish the link.

`shpfile(`*shapefile*`)` and `drop` are for linking or relinking to a shapefile. To reestablish the link to the shapefile that was just unlinked above, you would type

```
. spset, modify shpfile(shapefile_shp)
```

The shapefile will be relinked, and the coordinates stored in `_CX` and `_CY` will be restored.

`shpfile()` will refuse to link the shapefile if the data in memory contain observations for `_ID` values not found in the shapefile. In this case, specify `shpfile()` and `drop` if you are willing to drop the extra observations from the data in memory.

Remarks and examples

Remarks are presented under the following headings:

Determining whether and how data are spset

`spset` without arguments queries the Sp setting. Data starts out not being `spset`:

```
. spset
data not spset
r(459);
```

After the data have been `spset`, the output might be

```
. spset
  Sp dataset
                data:  cross sectional
   spatial-unit ID:  _ID
       coordinates:  _CY, _CX (latitude-and-longitude, miles)
  linked shapefile:  shapefile_shp.dta
```

These data are as described in [SP] **Intro 4**. They are linked to a Stata-format shapefile.

Or, the output might be

```
. spset
  Sp dataset
                data:  cross sectional
   spatial-unit ID:  _ID (equal to fips)
       coordinates:  _CY, _CX (latitude-and-longitude, miles)
  linked shapefile:  none
```

These data are as described in [SP] **Intro 5**. The data contain coordinates but are not linked to a shapefile.

Or, the output might be

```
. spset
  Sp dataset
                data:  cross sectional
   spatial-unit ID:  _ID (equal to fips)
       coordinates:  none
  linked shapefile:  none
```

These data are as described in [SP] **Intro 6**. They do not contain coordinates nor are they linked to a shapefile.

All the examples above are for cross-sectional data. If the data were panel data, the output might be

```
. spset
  Sp dataset
                  data:  panel
     spatial-unit ID:  _ID
             time id:  time (see xtset)
          coordinates:  _CY, _CX (latitude-and-longitude, miles)
    linked shapefile:  shapefile_shp.dta
```

Setting data for the first time

There are two kinds of data as far as Sp is concerned: cross-sectional and panel. In brief, cross-sectional data contain one observation per spatial unit, such as one observation per county. Panel data contain multiple observations, such as one observation per county per calendar year. The kinds of data are described in more detail in [SP] **Intro 3**.

We are about to explain the various `spset` cases one at a time. We will discuss cross-sectional and panel data together. In all the examples, we will assume that you want to `spset` *analysis*`.dta`. This example dataset has the following characteristics:

1. It is cross-sectional or panel.
2. It contains data on U.S. counties. Variable `fips` contains the standard federal information processing standard (FIPS) code identifying U.S. counties.

If the data are cross-sectional, then `fips` uniquely identifies the observations.

If the data are panel, then variable `time` will be assumed to contain the second-level identifier. `fips` and `time` uniquely identify the observations. The time variable need not be named `time`, nor is the second-level identifier required to be time. See [SP] **Intro 3**.

`spset` adds one or three variables to your data.

1. `_ID`, which identifies the geographical areas.
2. `_CX` and `_CY`, which record the coordinates of the areas. Variables `_CX` and `_CY` are added only if the coordinates are known.

`spset` also adds information stored in Stata characteristics.

3. *coordsys*, the system in which coordinates are recorded and whether distances should be measured in kilometers or miles.
4. *shpfile*, the name of the Stata-format shapefile to which the data are linked, if they are linked.

The variables and characteristics that `spset` adds to your data should be viewed as `spset`'s property. Do not modify or drop them. Use `spset, modify` to change settings.

Setting data with a standard-format shapefile

Shapefiles contain maps for each of the spatial units, which we will imagine are counties of the United States. You obtain shapefiles over the web.

You use [SP] **spshape2dta** to translate standard-format `*.zip` shapefiles to Stata-format `*_shp.dta` files. How you do that is explained in [SP] **Intro 4**.

`spshape2dta` performs the initial Sp setting of the data for you. That initial setting will be

```
. spset
    data:  cross sectional
 spatial-unit id:  _ID
     coordinates:  _CY, _CX (planar)
linked shapefile:  shapefile_shp.dta
```

Note that spshape2dta derived the centroid coordinates for each of the spatial units (counties) and spset them.

You can modify settings. One important setting specifies how the coordinates are recorded. They are either planar, which is another word for rectangular, or they are degrees latitude and longitude. By default, Sp assumes coordinates are planar. Sp provides two coordinate-system settings:

```
. spset, modify coordsys(planar)
. spset, modify coordsys(latlong)
```

It is important that you modify coordsys() to be latlong if that is what the data record, because the formulas for calculating distances differ; see [SP] **spdistance**. Sp datasets record the coordinate values in variables _CX and _CY.

coordsys(latlong) has an extra setting that may be important to you:

```
. spset, modify coordsys(latlong, kilometers)
. spset, modify coordsys(latlong, miles)
```

By default, coordsys(latlong) calculates distances in kilometers.

Setting data with a Stata-format shapefile

All shapefiles start out as standard-format shapefiles and are translated into Stata-format _shp.dta files. It is possible that you have a Stata-format _shp.dta file from a previous analysis that is appropriate for this analysis. In that case, you can just link to it.

Let's assume that we want to spset *analysis*.dta, which you may recall is county data and contains variable fips (and time if it is panel data).

Let's assume that you also have Stata-format shapefile *shapefile*_shp.dta from a previous analysis. The _shp.dta file is indexed on FIPS codes.

To spset the data and link them to *shapefile*_shp.dta, type

Cross-sectional data:

```
. use analysis
. spset fips
. spset, modify shpfile(shapefile_shp)
```

Panel data:

```
. use analysis
. xtset fips time
. spset fips
. spset, modify shpfile(shapefile_shp)
```

The above will work as long as *analysis*.dta does not contain counties that do not appear in *shapefile*_shp.dta; see shpfile() and drop under *Options* above.

Notice that spset expects xtset to handle panel-data details. With panel data, you are required to xtset the data first. After the spset, if you typed xtset without arguments, you would discover that the spset modified the xtset setting. Data that were xtset on fips and time will now be xtset on _ID and time. When you typed spset fips, spset created the variable _ID equal to fips, and then it changed the xtset setting to match its own. spset does not drop the variable fips; it just makes its own copy of it.

Actually, what we typed for the panel-data case may not be sufficient. We should have typed

Panel data:

```
. use analysis
. xtset fips time
. spbalance, balance                // <-- new
. spset fips
. spset, modify shpfile(shapefile_shp)
```

`spset` requires that panel data be strongly balanced. `spbalance, balance` will make panel data strongly balanced if they are not already. We omitted it because `spset` will verify that the data are strongly balanced and, if they are not, will issue an error. If `spset` complains, we can type `spbalance, balance` and then type the `spset` command again. See [SP] **spbalance**.

Whether data are cross-sectional or panel, you may need to modify the `coordsys()` setting. Use

All data:

```
. spset, modify coordsys(latlong, kilometers)
. spset, modify coordsys(latlong, miles)
```

It is important that the coordinate system be set correctly; see [SP] **spdistance**.

Setting data without a shapefile but with coordinates

Assume that *analysis*`.dta` is the same county dataset used previously. In addition to `fips` and perhaps `time`, the data also contain variables `x` and `y` recording the coordinates of each county.

To `spset` the data without a shapefile, type

Cross-sectional data:

```
. use analysis
. spset fips, coord(x y)
```

Panel data:

```
. use analysis
. xtset fips time
. spset fips, coord(x y)
```

If `x` contains longitude and `y` contains latitude, also type

All data:

```
. spset, modify coordsys(latlong, kilometers)
. spset, modify coordsys(latlong, miles)
```

Setting data without a shapefile

Assume that *analysis*`.dta` no longer contains variables `x` and `y`. We have no shapefile and no coordinates. At this point, the data are probably not even geographically based. So rather than `fips`, we will assume the spatial units are uniquely identified by `node`. If the data are panel data, we assume observations are identified by `node` and `time`.

To `spset` the data, type

Cross-sectional data:

```
. use analysis
. spset node
```

Panel data:

```
. use analysis
. xtset node time
. spset fips
```

Modifying settings

You use `spset, modify` to modify settings of data that are already `spset`. You may modify whether the data contain coordinates, whether the coordinates are planar or latitude and longitude, the ID-variable codes used to identify the spatial units, and whether the data are linked to a shapefile.

Modifying coordinates

The coordinates for each of the spatial units in your data are stored in variables `_CX` and `_CY` if Sp knows them.

Sp knows the coordinates if you are linked to a shapefile. It knows them because Sp itself calculated the centroids of the spatial units from the information in the shapefile and stored the results in `_CX` and `_CY`.

Sp also knows the coordinates if you are not linked to a shapefile but specified the coordinates when you originally `spset` the data. In that case, it copied the coordinates you supplied into `_CX` and `_CY`.

If you are linked to a shapefile, you may not modify the coordinates Sp has stored—nor would you want to modify them.

If you are not linked to a shapefile, you can add or replace coordinates by typing

```
spset, modify coord(xvar yvar)
```

If you want to delete the coordinates, type

```
spset, modify nocoord
```

Modifying how coordinates are interpreted

The coordinates stored in `_CX` and `_CY` are interpreted as planar or as degrees latitude and longitude. The interpretation determines how distances are calculated; see [SP] **spdistance**. You can change the interpretation by typing

```
spset, modify coordsys(planar)
spset, modify coordsys(latlong)
```

In the case of the `latlong` setting, you can specify the units to be used for distances, too:

```
spset, modify coordsys(latlong, kilometers)
spset, modify coordsys(latlong, miles)
```

When you set or reset `coordsys(latlong)`, kilometers are assumed.

Modifying the ID variable

Variable _ID identifies the spatial units in your data. Each unit has a different code. The codes could be 1, 2, and 3 or any set of numbers.

If you started with a standard-format shapefile, Sp used 1, 2, 3, and so on for _ID when you used spshape2dta to translate the file to Stata format. Perhaps you subsequently modified the coding stored in _ID. We did in [SP] **Intro 4** when we showed you how to prepare data using a shapefile. We modified _ID to contain FIPS codes.

You can modify the codes stored in _ID at any time. The commands are as follows:

spset *newidvar*, modify	if you are not linked to a shapefile
spset *newidvar*, modify replace	if you are linked to a shapefile

Avoid doing this. These commands exist so that you can modify _ID at the outset when you are preparing your data. At that stage, you have no investment in the codes that are being used.

Later, you have an investment. The codes were used to identify the rows and columns of spatial weighting matrices you created. If you change the codes, any weighting matrices you have saved will become unusable.

If you are linked to a shapefile and change the codes, Sp will reindex both your data and its linked shapefile. If other datasets are linked to the same shapefile, their links to it will no longer work.

In [SP] **Intro 4**, [SP] **Intro 5**, and [SP] **Intro 6**, we modified codes before you became invested in the coding system used.

Sometimes, you really do have to change codes later. Let's imagine the unimaginable situation where the U.S. Census Bureau changes from FIPS in favor of NEWFIPS. Even then, we would ask you whether you really need to migrate to NEWFIPS, but for this example we will assume that you do. We will assume that you have a migration dataset containing two variables, fips and newfips. Variable newfips is never missing, but some fips values might be. Start by taking the migration dataset and dropping any observations for which fips is missing:

```
. use migration, clear
. drop if missing(fips)
. save mymigration
```

Now, merge with your analysis file:

```
. use project, clear
. merge 1:1 fips using mymigration
. assert _merge!=1              // no master unmatched
. keep if _merge==3             // keep the matches
. drop _merge
```

You can now change Sp's _ID variable. If project.dta is not linked to a shapefile, type

```
. spset newfips, modify
. save, replace
```

If it is linked to a shapefile, type

```
. spset newfips, modify replace
. save, replace
```

File project.dta now uses NEWFIPS. There is no solution that will allow the use of old spatial weighting matrices indexed on FIPS. You will be using the NEWFIPS codes.

If your data were linked to a shapefile and you have other datasets linked to the shapefile you just reindexed, you need to do the following with each dataset:

```
. use dataset, clear
. spset, modify noshpfile
. merge 1:1 fips using mymigration
. assert _merge!=1              // no master unmatched
. keep if _merge==3             // keep the matches
. drop _merge
. spset newfips, modify
. spset, modify shpfile(shapefile_shp.dta)
. save, replace
```

Modifying whether the data are linked to a shapefile

The commands

```
spset, modify shpfile(shapefile)
spset, modify noshpfile
```

make and break links to shapefiles. When you establish a connection, variable _ID must use the same codes as the Stata-format shapefile *shapefile*.

We used these commands in the example in the previous section.

Converting cross-sectional data to panel data and vice versa

Cross-sectional data can become panel data and vice versa. A cross-sectional dataset could become panel because of a merge. A panel dataset could become cross-sectional because of a drop.

Here is a case of cross-sectional data becoming panel data:

```
. use analysis              // cross-sectional data
. spset
  Sp dataset
                  data:  cross sectional
     spatial-unit id:  _ID
         coordinates:  _CY, _CX (latitude and longitude, miles)
    linked shapefile:  shapefile_shp.dta
. merge 1:m fips using paneldata
  (output omitted)
```

Note that the data were `spset` before the `merge`, and after the `merge`, the data are panel data, but they are not yet `xtset`. If you typed `spset` without arguments right now, it would complain about repeated _ID values. To fix the problem, `xtset` the data:

```
. xtset fips time
```

Now, `spset` will report

```
. spset
  Sp dataset
                  data:  panel
     spatial-unit ID:  _ID
             time id:  time (see xtset)
         coordinates:  _CY, _CX (latitude-and-longitude, miles)
    linked shapefile:  shapefile_shp.dta
```

Now, let's convert these panel data back to cross-sectional data:

```
. keep if time==1
```

Here is how you tell Sp that the data are no longer panel data:

```
. xtset, clear
```

Now, spset will report

```
. spset
  Sp dataset
                 data:  cross sectional
    spatial-unit ID:  _ID
        coordinates:  _CY, _CX (latitude-and-longitude, miles)
   linked shapefile:  shapefile_shp.dta
```

Stored results

spset stores the following in r():

Macros

r(sp_ver)	1
r(sp_id)	_ID
r(sp_id_var)	*varname* or empty
r(sp_shp_dta_path)	path to _shp.dta file
r(sp_shp_dta)	*shapefile*_shp.dta
r(sp_cx)	_CX or empty
r(sp_cy)	_CY or empty
r(sp_coord_sys)	planar or latlong
r(sp_coord_sys_dunit)	kilometers or miles if r(sp_coord_sys) = latlong

Also see

[SP] **Intro 3** — Preparing data for analysis

[SP] **Intro 4** — Preparing data: Data with shapefiles

[SP] **Intro 5** — Preparing data: Data containing locations (no shapefiles)

[SP] **Intro 6** — Preparing data: Data without shapefiles or locations

[SP] **Intro 7** — Example from start to finish

[SP] **spbalance** — Make panel data strongly balanced

[SP] **spdistance** — Calculator for distance between places

[SP] **spshape2dta** — Translate shapefile to Stata format

[XT] **xtset** — Declare data to be panel data

Title

spshape2dta — Translate shapefile to Stata format

Description

`spshape2dta` *name* reads files *name*`.shp` and *name*`.dbf` and creates Sp dataset *name*`.dta` and translated shapefile *name*`_shp.dta`. The translated shapefile will be linked to the Sp dataset *name*`.dta`.

Quick start

Create `myfile.dta` and `myfile_shp.dta` from `myfile.shp` and `myfile.dbf`

```
spshape2dta myfile
```

Create `newfile.dta` and `newfile_shp.dta` from `oldfile.shp` and `oldfile.dbf`

```
spshape2dta oldfile, saving(newfile)
```

Menu

Statistics > Spatial autoregressive models

Syntax

`spshape2dta` *name* [, *options*]

options	Description
`clear`	clear existing data from memory
`replace`	if *name*`.dta` or *name*`_shp.dta` exists, replace them
`saving(`*name2*`)`	create new files named *name2*`.dta` and *name2*`_shp.dta` instead of *name*`.dta` and *name*`_shp.dta`

`spshape2dta` translates files *name*`.shp` and *name*`.dbf`. They must be in the current directory.

`spshape2dta` creates files *name*`.dta` and *name*`_shp.dta`. They will be created in the current directory. The data in memory, if any, remain unchanged.

Options

`clear` specifies to clear any data in memory.

`replace` specifies that if the new files being created already exist on disk, they can be replaced.

`saving(`*name2*`)` specifies that rather than the new files being named *name*`.dta` and *name*`_shp.dta`, they be named *name2*`.dta` and *name2*`_shp.dta`.

Remarks and examples

`spshape2dta` is the first step in preparing data to be used with shapefiles. See [SP] **Intro 4** for step-by-step instructions.

`spshape2dta` creates two files:

name`.dta`
name`_shp.dta`

name`.dta` is an ordinary Stata dataset. The dataset will have N observations, one for each spatial unit. The dataset will be `spset`.

```
. use name
. spset
  Sp dataset
                  data:  cross sectional
     spatial-unit ID:  _ID
         coordinates:  _CY, _CX (latitude-and-longitude, miles)
   linked shapefile:  name_shp.dta
```

name`.dta` will contain the variables

`_ID`	values $1, 2, \ldots, N$. This variable links observations in the data to observations in the Stata-format shapefile, *name*`_shp.dta`.
`_CX`, `_CY`	contain the centroids for the places (spatial units)

name`.dta` will include the other variables defined in *name*`.dbf`. Usually, there will be five or ten. What they contain varies but can usually be determined from their names and by looking at their values.

name`.dta` will be linked to *name*`_shp.dta`, which is called the Stata-format shapefile. It contains the map. It too is an ordinary Stata dataset, but you ignore it. Sp will use *name*`_shp.dta` behind the scenes when you construct contiguity spatial weighting matrices using `spmatrix create contiguity` or when you graph choropleth maps using `grmap`.

Also see

[SP] **Intro 3** — Preparing data for analysis

[SP] **Intro 4** — Preparing data: Data with shapefiles

Title

spxtregress — Spatial autoregressive models for panel data

Description

`spxtregress` fits spatial autoregressive (SAR) models, also known as simultaneous autoregressive models, for panel data. The commands `spxtregress, fe` and `spxtregress, re` are extensions of `xtreg, fe` and `xtreg, re` for spatial data; see [XT] **xtreg**.

If you have not read [SP] **Intro 1**–[SP] **Intro 8**, you should do so before using `spxtregress`.

To use `spxtregress`, your data must be Sp data and `xtset`. See [SP] **Intro 3** for instructions on how to prepare your data.

To specify spatial lags, you will need to have one or more spatial weighting matrices. See [SP] **Intro 2** and [SP] **spmatrix** for an explanation of the types of weighting matrices and how to create them.

Quick start

SAR fixed-effects model of `y` on `x1` and `x2` with a spatial lag of `y` specified by the spatial weighting matrix `W`

```
spxtregress y x1 x2, fe dvarlag(W)
```

Add a spatially lagged error term also specified by `W`

```
spxtregress y x1 x2, fe dvarlag(W) errorlag(W)
```

Add spatial lags of covariates `x1` and `x2`

```
spxtregress y x1 x2, fe dvarlag(W) errorlag(W) ivarlag(W: x1 x2)
```

Add an additional spatial lag of the covariates specified by the matrix `M`

```
spxtregress y x1 x2, fe dvarlag(W) errorlag(W) ivarlag(W: x1 x2)    ///
     ivarlag(M: x1 x2)
```

SAR random-effects model

```
spxtregress y x1 x2, re dvarlag(W) errorlag(W) ivarlag(W: x1 x2)    ///
     ivarlag(M: x1 x2)
```

An `re` model with panel effects that follow the same spatial process as the errors using `sarpanel`

```
spxtregress y x1 x2, re sarpanel dvarlag(W) errorlag(W)             ///
     ivarlag(W: x1 x2) ivarlag(M: x1 x2)
```

Menu

Statistics > Spatial autoregressive models

Syntax

Fixed-effects maximum likelihood

```
spxtregress depvar [indepvars] [if] [in], fe [fe_options]
```

Random-effects maximum likelihood

```
spxtregress depvar [indepvars] [if] [in], re [re_options]
```

fe_options	Description
Model	
* `fe`	use fixed-effects estimator
`dvarlag(`*spmatname*`)`	spatially lagged dependent variable
`errorlag(`*spmatname*`)`	spatially lagged errors
`ivarlag(`*spmatname* `:` *varlist*`)`	spatially lagged independent variables; repeatable
`force`	allow estimation when estimation sample is a subset of the sample used to create the spatial weighting matrix
`gridsearch(#)`	resolution of the initial-value search grid; seldom used
Reporting	
`level(#)`	set confidence level; default is `level(95)`
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process; seldom used
`coeflegend`	display legend instead of statistics

re_options	Description
Model	
* `re`	use random-effects estimator
`dvarlag(`*spmatname*`)`	spatially lagged dependent variable
`errorlag(`*spmatname*`)`	spatially lagged errors
`ivarlag(`*spmatname* `:` *varlist*`)`	spatially lagged independent variables; repeatable
`sarpanel`	alternative formulation of the estimator in which the panel effects follow the same spatial process as the errors
`noconstant`	suppress constant term
`force`	allow estimation when estimation sample is a subset of the sample used to create the spatial weighting matrix
Reporting	
`level(#)`	set confidence level; default is `level(95)`
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process; seldom used
`coeflegend`	display legend instead of statistics

* You must specify either `fe` or `re`.

indepvars and *varlist* specified in `ivarlag()` may contain factor variables; see **[U] 11.4.3 Factor variables**.

`coeflegend` does not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options for spxtregress, fe

Model

`fe` requests the fixed-effects regression estimator.

`dvarlag(`*spmatname*`)` specifies a spatial weighting matrix that defines a spatial lag of the dependent variable. Only one `dvarlag()` option may be specified. By default, no spatial lags of the dependent variable are included.

`errorlag(`*spmatname*`)` specifies a spatial weighting matrix that defines a spatially lagged error. Only one `errorlag()` option may be specified. By default, no spatially lagged errors are included.

`ivarlag(`*spmatname* `:` *varlist*`)` specifies a spatial weighting matrix and a list of independent variables that define spatial lags of the variables. This option is repeatable to allow spatial lags created from different matrices. By default, no spatial lags of the independent variables are included.

`force` requests that estimation be done when the estimation sample is a proper subset of the sample used to create the spatial weighting matrices. The default is to refuse to fit the model. Weighting matrices potentially connect all the spatial units. When the estimation sample is a subset of this space, the spatial connections differ and spillover effects can be altered. In addition, the normalization of the weighting matrix differs from what it would have been had the matrix been normalized over the estimation sample. The better alternative to `force` is first to understand the spatial space of the estimation sample and, if it is sensible, then create new weighting matrices for it. See [SP] **spmatrix** and *Missing values, dropped observations, and the W matrix* in [SP] **Intro 2**.

gridsearch(#) specifies the resolution of the initial-value search grid. The default is gridsearch(0.1). You may specify any number between 0.001 and 0.1 inclusive.

Reporting

level(#); see [R] **Estimation options**.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(*style*), cformat(%*fmt*), pformat(%*fmt*), sformat(%*fmt*), and nolstretch; see [R] **Estimation options**.

Maximization

maximize_options: difficult, technique(*algorithm_spec*), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), and nonrtolerance; see [R] **Maximize**.

The following option is available with spxtregress, fe but is not shown in the dialog box:

coeflegend; see [R] **Estimation options**.

Options for spxtregress, re

Model

re requests the generalized least-squares random-effects estimator.

dvarlag(*spmatname*) specifies a spatial weighting matrix that defines a spatial lag of the dependent variable. Only one dvarlag() option may be specified. By default, no spatial lags of the dependent variable are included.

errorlag(*spmatname*) specifies a spatial weighting matrix that defines a spatially lagged error. Only one errorlag() option may be specified. By default, no spatially lagged errors are included.

ivarlag(*spmatname* : *varlist*) specifies a spatial weighting matrix and a list of independent variables that define spatial lags of the variables. This option is repeatable to allow spatial lags created from different matrices. By default, no spatial lags of the independent variables are included.

sarpanel requests an alternative formulation of the estimator in which the panel effects follow the same spatial process as the errors. By default, the panel effects are included in the estimation equation as an additive term, just as they are in the standard nonspatial random-effects model. When sarpanel and errorlag(*spmatname*) are specified, the panel effects also have a spatial autoregressive form based on *spmatname*. If errorlag() is not specified with sarpanel, the estimator is identical to the estimator when sarpanel is not specified. The sarpanel estimator was originally developed by Kapoor, Kelejian, and Prucha (2007); see *Methods and formulas*.

noconstant; see [R] **Estimation options**.

force requests that estimation be done when the estimation sample is a proper subset of the sample used to create the spatial weighting matrices. The default is to refuse to fit the model. This is the same force option described for use with spxtregress, fe.

Reporting

`level(#)`; see [R] **Estimation options**.

display_options: `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(#)`, `fvwrapon(`*style*`)`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

Maximization

maximize_options: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, `[no]log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(#)`, `ltolerance(#)`, `nrtolerance(#)`, and `nonrtolerance`; see [R] **Maximize**.

The following option is available with `spxtregress, re` but is not shown in the dialog box:

`coeflegend`; see [R] **Estimation options**.

Remarks and examples

See [SP] **Intro** for an overview of SAR models.

Datasets for Sp panel models contain observations on geographical areas or other units with multiple observations on each unit. See [SP] **Intro 3** for an explanation of how to work with Sp panel data. The data must be `xtset` and must be strongly balanced. There must be a within-panel identifier, a variable indicating time or the equivalent, and the values of this identifier must be the same for every panel. The command `spbalance` will strongly balance datasets that are not strongly balanced. See [SP] **Intro 3**, [SP] **Intro 7**, and [SP] **spbalance**.

Remarks and examples are presented under the following headings:

Sp panel models

Both the fixed-effects and the random-effects models for spatial panel data can be written as

$$\begin{aligned} \mathbf{y}_{nt} &= \lambda\mathbf{W}\mathbf{y}_{nt} + \mathbf{X}_{nt}\beta + \mathbf{c}_n + \mathbf{u}_{nt} \\ \mathbf{u}_{nt} &= \rho\,\mathbf{M}\,\mathbf{u}_{nt} + \mathbf{v}_{nt} \qquad\qquad t = 1, 2, \ldots, T \end{aligned} \tag{1}$$

where $\mathbf{y}_{nt} = (y_{1t}, y_{2t}, \ldots, y_{nt})'$ is an $n \times 1$ vector of observations for the dependent variable for time period t with n number of panels; $\mathbf{X}_{nt}$ is a matrix of time-varying regressors; $\mathbf{c}_n$ is a vector of panel-level effects; $\mathbf{u}_{nt}$ is the spatially lagged error; $\mathbf{v}_{nt}$ is a vector of disturbances and is independent and identically distributed (i.i.d.) across panels and time with variance σ^2; and $\mathbf{W}$ and $\mathbf{M}$ are spatial weighting matrices.

The fixed-effects model

For fixed effects, `spxtregress, fe` implements the quasi–maximum likelihood (QML) estimator in Lee and Yu (2010a) to fit the model. A transformation is used to eliminate the fixed effects from the equations, yielding

$$\widetilde{\mathbf{y}}_{nt} = \lambda\mathbf{W}\widetilde{\mathbf{y}}_{nt} + \widetilde{\mathbf{X}}_{nt}\beta + \widetilde{\mathbf{u}}_{nt}$$
$$\widetilde{\mathbf{u}}_{nt} = \rho\mathbf{M}\widetilde{\mathbf{u}}_{nt} + \widetilde{\mathbf{v}}_{nt} \qquad t = 1, 2, \ldots, T-1$$

Panel effects, which are effects that are constant within panels, are conditioned out of the likelihood. Only covariates that vary within panels can be fit with this estimator.

The random-effects model

For random effects, `spxtregress, re` assumes that $\mathbf{c}_n$ in (1) is normal i.i.d. across panels with mean 0 and variance $\sigma^2_{\mathbf{c}}$. The output of `spxtregress, re` displays estimates of $\sigma_{\mathbf{c}}$, labeled as `/sigma_u`, and σ, labeled as `/sigma_e`, which is consistent with how `xtreg, re` labels the output.

The random-effects model with autoregressive panel effects

The `sarpanel` option for random-effects models fits a slightly different set of equations from (1):

$$\mathbf{y}_{nt} = \lambda\mathbf{W}\mathbf{y}_{nt} + \mathbf{X}_{nt}\beta + \mathbf{u}_{nt}$$
$$\mathbf{u}_{nt} = \rho\,\mathbf{M}\,\mathbf{u}_{nt} + \mathbf{c}_n + \mathbf{v}_{nt}, \qquad t = 1, 2, \ldots, T$$

In this variant due to Kapoor, Kelejian, and Prucha (2007), the panel-level effects $\mathbf{c}_n$ are considered a disturbance in the error equation. Because $\mathbf{c}_n$ enters the equation as an additive term next to $\mathbf{v}_{nt}$, the panel-level effects $\mathbf{c}_n$ have the same autoregressive form as the time-level errors $\mathbf{v}_{nt}$.

Differences among models

All three of the models—`fe`, `re`, and `re sarpanel`—are fit using maximum likelihood (ML) estimation. The differences are 1) `fe` removes the panel-level effects from the estimation and no distributional assumptions are made about them; 2) `re` models the panel-level effects as normal i.i.d.; and 3) `re sarpanel` assumes a normal distribution for panel-level effects but with the same autoregressive form as the time-level errors. The `fe` model allows the panel-level effects to be correlated with the observed covariates, whereas the `re` models require that the panel-level effects are independent of the observed covariates. See *Methods and formulas* for details. Also see *Choosing weighting matrices and their normalization* in [SP] **spregress**; the discussion there applies to these three estimation models.

Examples

Example 1: spxtregress, re

We have data on the homicide rate in counties in southern states of the U.S. for the years 1960, 1970, 1980, and 1990. `homicide_1960_1990.dta` contains `hrate`, the county-level homicide rate per year per 100,000 persons for each of the four years. It also contains `ln_population`, the logarithm of the county population; `ln_pdensity`, the logarithm of the population density; and `gini`, the Gini coefficient for the county, a measure of income inequality where larger values represent more inequality (Gini 1909). The data are an extract of the data originally used by Messner et al. (2000); see Britt (1994) for a literature review of the topic. The 1990 data are used in the examples in [SP] **spregress**.

We used `spshape2dta` to convert shapefiles into Stata `.dta` files, and then we merged the data file by county ID with our homicide-rate data. See [SP] **Intro 4**, [SP] **Intro 7**, [SP] **spshape2dta**, and [SP] **spset**.

Because the analysis dataset and the Stata-formatted shapefile must be in our working directory to `spset` the data, we first save both `homicide_1960_1990.dta` and `homicide_1960_1990_shp.dta` to our working directory by using the `copy` command. We then load the data and type `spset` to see the Sp settings.

```
. copy https://www.stata-press.com/data/r16/homicide_1960_1990.dta .
. copy https://www.stata-press.com/data/r16/homicide_1960_1990_shp.dta .
. use homicide_1960_1990
(S.Messner et al.(2000), U.S southern county homicide rate in 1960-1990)
. spset
  Sp dataset homicide_1960_1990.dta
                data:  cross sectional
   spatial-unit id:  _ID
       coordinates:  _CX, _CY (planar)
  linked shapefile:  homicide_1960_1990_shp.dta
variable _ID does not uniquely identify the observations
   Do these data need to be xtset?
r(459);
```

We get an error! The data have not been `xtset`, and `spxtregress` requires it. Our data consist of 1,412 counties, and for each county we have data for four years. Our data look like this:

```
. list _ID year in 1/8, sepby(_ID)
```

	_ID	year
1.	876	1960
2.	876	1970
3.	876	1980
4.	876	1990
5.	921	1960
6.	921	1970
7.	921	1980
8.	921	1990

We type

```
. xtset _ID year
       panel variable:  _ID (strongly balanced)
        time variable:  year, 1960 to 1990, but with gaps
                delta:  1 unit
```

`xtset` reports that our data are strongly balanced. Each county has data for the same four years. `spxtregress` requires the data to be strongly balanced. Missing values in our variables could cause the estimation sample to be unbalanced. The Sp panel estimators will complain, and we will have to make the data strongly balanced for the nonmissing values of the variables in our model. If you get a message that your data are not strongly balanced, see [SP] **spbalance**.

After having `xtset` our data, we type `spset` to check our Sp settings.

```
. spset
  Sp dataset homicide_1960_1990.dta
                data:  panel
    spatial-unit id:  _ID
           time id:  year (see xtset)
       coordinates:  _CX, _CY (planar)
     linked shapefile:  homicide_1960_1990_shp.dta
```

We first run a nonspatial random-effects model by using `xtreg, re` and include dummies for the years by using the `i.year` factor-variable notation.

```
. xtreg hrate ln_population ln_pdensity gini i.year, re
Random-effects GLS regression                   Number of obs     =      5,648
Group variable: _ID                             Number of groups  =      1,412

R-sq:                                           Obs per group:
     within  = 0.0478                                         min =          4
     between = 0.1666                                         avg =        4.0
     overall = 0.0905                                         max =          4

                                                Wald chi2(6)      =     414.32
corr(u_i, X)   = 0 (assumed)                    Prob > chi2       =     0.0000

------------------------------------------------------------------------------
       hrate |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
ln_populat~n |   .4394103   .1830599     2.40   0.016     .0806194    .7982012
 ln_pdensity |   .3220698   .1591778     2.02   0.043     .0100872    .6340525
        gini |   34.43792   2.905163    11.85   0.000      28.7439    40.13193
             |
        year |
       1970  |   1.411074   .2579218     5.47   0.000     .9055562    1.916591
       1980  |   1.347822   .2499977     5.39   0.000     .8578352    1.837808
       1990  |   .3668468   .2648395     1.39   0.166    -.1522291    .8859228
             |
       _cons |  -10.07267   1.800932    -5.59   0.000    -13.60243   -6.542908
-------------+----------------------------------------------------------------
     sigma_u |  3.5995346
     sigma_e |   5.646151
         rho |  .28898083   (fraction of variance due to u_i)
------------------------------------------------------------------------------
```

We emphasize that you can ignore the spatial aspect of the data and use any of Stata's estimation commands even though the data are spatial. Doing that is often a good idea. It provides a baseline against which you can compare subsequent spatial results.

We are now going to estimate a spatial random-effects model. To do that, we need a spatial weighting matrix. We will create one that puts the same positive weight on contiguous counties and a 0 weight on all other counties—a matrix known as a contiguity matrix. We will use the default

spectral normalization for this example. See [SP] **spmatrix create**. When we create the matrix, we must restrict `spmatrix create` to one observation per panel. That is easy to do using an `if` statement:

```
. spmatrix create contiguity W if year == 1990
```

Do not misinterpret the purpose of `if year == 1990`. The matrix created will be appropriate for creating spatial lags for any year, because our map does not change. If two counties share a border in 1990, they share it in the other years too.

We can now fit our model. We include a spatial lag of the dependent variable and a spatially autoregressive error term.

```
. spxtregress hrate ln_population ln_pdensity gini i.year, re dvarlag(W)
> errorlag(W)
  (5648 observations)
  (5648 observations used)
  (data contain 1412 panels (places) )
  (weighting matrix defines 1412 places)

Fitting starting values:

Iteration 0:   log likelihood = -13299.332
Iteration 1:   log likelihood = -13298.431
Iteration 2:   log likelihood =  -13298.43
Iteration 3:   log likelihood =  -13298.43

Optimizing concentrated log likelihood:

initial:       log likelihood = -18826.009
improve:       log likelihood = -18826.009
rescale:       log likelihood = -18826.009
rescale eq:    log likelihood = -18500.374
Iteration 0:   log likelihood = -18500.374  (not concave)
Iteration 1:   log likelihood = -18473.617  (not concave)
Iteration 2:   log likelihood = -18465.333
Iteration 3:   log likelihood = -18434.609
Iteration 4:   log likelihood = -18356.316
Iteration 5:   log likelihood = -18354.863
Iteration 6:   log likelihood =  -18354.84
Iteration 7:   log likelihood =  -18354.84

Optimizing unconcentrated log likelihood:

Iteration 0:   log likelihood =  -18354.84
Iteration 1:   log likelihood =  -18354.84  (backed up)
```

```
Random-effects spatial regression               Number of obs     =      5,648
Group variable: _ID                             Number of groups  =      1,412
                                                Obs per group     =          4
                                                Wald chi2(7)      =    1421.80
                                                Prob > chi2       =     0.0000
Log likelihood = -1.835e+04                     Pseudo R2         =     0.0911
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	-.2988716	.1622148	-1.84	0.065	-.6168068	.0190637
ln_pdensity	.7893219	.1380612	5.72	0.000	.518727	1.059917
gini	22.77053	2.604624	8.74	0.000	17.66556	27.8755
year						
1970	.3977166	.1906034	2.09	0.037	.0241408	.7712924
1980	.4033441	.1825721	2.21	0.027	.0455094	.7611789
1990	-.1284627	.1946898	-0.66	0.509	-.5100478	.2531224
_cons	-4.182034	1.607561	-2.60	0.009	-7.332796	-1.031272
W						
hrate	.5740163	.0249799	22.98	0.000	.5250565	.622976
e.hrate	-.4626342	.0508732	-9.09	0.000	-.5623438	-.3629245
/sigma_u	3.087658	.1046893			2.88914	3.299816
/sigma_e	5.40831	.0661566			5.280188	5.539542

```
Wald test of spatial terms:          chi2(2) = 713.88     Prob > chi2 = 0.0000
```

`spxtregress, re` first fits an `spxtregress, fe` model to get starting values. Then, it optimizes the concentrated log likelihood and then optimizes the unconcentrated log likelihood. The final log likelihood of the concentrated will always be equal to the optimized log likelihood of the unconcentrated. The unconcentrated starts at the right point, takes a step to check that it is the right point, backs up to this point, and declares convergence as it should.

We can compare estimates of `/sigma_u`, the standard deviation of the panel effects, and `/sigma_e`, the standard deviation of the errors, with those fit by `xtreg, re`. They are similar. We cannot, however, directly compare the coefficient estimates with those of `xtreg, re`. When a spatial lag of the dependent variable is included in the model, covariates have both direct and indirect effects, as explained in example 1 of [SP] **spregress**. To obtain the direct, indirect, and total effects of the covariates, we must use `estat impact`.

Here are the averages of the effects of gini:

```
. estat impact gini
progress   :100%
Average impacts                                 Number of obs     =      5,648
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	24.1144	2.715901	8.88	0.000	18.79133	29.43747
indirect						
gini	22.73746	2.787574	8.16	0.000	17.27391	28.201
total						
gini	46.85185	5.126096	9.14	0.000	36.80489	56.89882

The percentages at the top of the output indicate progress in the estimation process. For large datasets, calculating standard errors of the effects can be time consuming, so **estat impact** reports its progress as it does the computations.

gini has significant average direct and average indirect effects on **hrate**, with both being positive. An increase in inequality is associated with an increase in the homicide rate.

We used a contiguity weighting matrix W for the spatial lags. Alternatively, we can use a weighting matrix based on the inverse distance between counties. We create this matrix, using again the default spectral normalization:

```
. spmatrix create idistance M if year == 1990
. spmatrix dir
```

Weighting matrix name	N x N	Type	Normalization
M	1412 x 1412	idistance	spectral
W	1412 x 1412	contiguity	spectral

We would like to know if the effects of `gini` differ over time, so we include an interaction of `gini` and `year` in our model, and we use the weighting matrix `M` that we just created.

```
. spxtregress hrate ln_population ln_pdensity c.gini##i.year, re
> dvarlag(M) errorlag(M)
  (5648 observations)
  (5648 observations used)
  (data contain 1412 panels (places) )
  (weighting matrix defines 1412 places)
```

(*output omitted*)

```
Random-effects spatial regression               Number of obs     =      5,648
Group variable: _ID                             Number of groups  =      1,412
                                                Obs per group     =          4
                                                Wald chi2(10)     =     710.10
                                                Prob > chi2       =     0.0000
Log likelihood = -1.827e+04                     Pseudo R2         =     0.1150
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	.7908003	.1764818	4.48	0.000	.4449023	1.136698
ln_pdensity	-.1223671	.166526	-0.73	0.462	-.448752	.2040178
gini	17.82039	4.278775	4.16	0.000	9.434144	26.20663
year						
1970	-2.456656	2.303069	-1.07	0.286	-6.970587	2.057275
1980	-9.470622	2.501527	-3.79	0.000	-14.37353	-4.567718
1990	-22.81817	2.528685	-9.02	0.000	-27.7743	-17.86204
year#c.gini						
1970	6.664314	6.130443	1.09	0.277	-5.351133	18.67976
1980	24.86122	6.715026	3.70	0.000	11.70001	38.02243
1990	57.40946	6.691086	8.58	0.000	44.29517	70.52374
_cons	-11.17804	2.061044	-5.42	0.000	-15.21762	-7.138471
M						
hrate	.694492	.0496075	14.00	0.000	.5972631	.7917209
e.hrate	1.950078	.0513563	37.97	0.000	1.849422	2.050735
/sigma_u	2.696022	.1147302			2.480277	2.930533
/sigma_e	5.645628	.0618616			5.525674	5.768186

```
Wald test of spatial terms:          chi2(2) = 1711.10    Prob > chi2 = 0.0000
```

Using the `contrast` command, we test the significance of the `gini` and `year` interaction:

```
. contrasts c.gini#year
Contrasts of marginal linear predictions
Margins      : asbalanced
```

	df	chi2	P>chi2
hrate			
year#c.gini	3	81.59	0.0000

The interaction is significant. We can explore the effect of `gini` by `year` using `estat impact` with an `if` statement.

```
. estat impact gini if year == 1960
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	17.85376	4.285821	4.17	0.000	9.453709	26.25382
indirect						
gini	37.06435	11.60646	3.19	0.001	14.31612	59.81259
total						
gini	54.91812	14.85782	3.70	0.000	25.79732	84.03891

```
. estat impact gini if year == 1970
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	24.53056	5.033537	4.87	0.000	14.66501	34.39611
indirect						
gini	50.92536	15.21235	3.35	0.001	21.10971	80.741
total						
gini	75.45591	18.8175	4.01	0.000	38.57429	112.3375

```
. estat impact gini if year == 1980
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	42.76155	5.683654	7.52	0.000	31.62179	53.9013
indirect						
gini	88.77282	23.09515	3.84	0.000	43.50716	134.0385
total						
gini	131.5344	26.20928	5.02	0.000	80.16512	182.9036

```
. estat impact gini if year == 1990
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	75.37074	5.628577	13.39	0.000	64.33893	86.40255
indirect						
gini	156.4694	37.24055	4.20	0.000	83.47925	229.4595
total						
gini	231.8401	39.0186	5.94	0.000	155.3651	308.3152

The `if year == ···` statement used with `estat impact` allows us to estimate the average effects for each year. The direct, indirect, and total effects of `gini` trend upward.

Until now, we used the default form of the random-effects estimator. Let's run the command again, specifying the `sarpanel` option to use the alternative form of the estimator, where the panel-level effects have the same autoregressive form as the time-level errors.

```
. spxtregress hrate ln_population ln_pdensity c.gini##i.year, re sarpanel
> dvarlag(M) errorlag(M)
  (5648 observations)
  (5648 observations used)
  (data contain 1412 panels (places) )
  (weighting matrix defines 1412 places)
  (output omitted)
Random-effects spatial regression               Number of obs     =      5,648
Group variable: _ID                             Number of groups  =      1,412
                                                Obs per group     =          4
                                                Wald chi2(10)     =    1136.49
                                                Prob > chi2       =     0.0000
Log likelihood = -1.824e+04                     Pseudo R2         =     0.1177
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	.4366742	.1752502	2.49	0.013	.0931901	.7801583
ln_pdensity	.1896	.1641334	1.16	0.248	-.1320955	.5112956
gini	18.92328	4.42621	4.28	0.000	10.24807	27.59849
year						
1970	-.9590229	2.362015	-0.41	0.685	-5.588488	3.670442
1980	-8.19778	2.554504	-3.21	0.001	-13.20452	-3.191045
1990	-22.4189	2.610152	-8.59	0.000	-27.53471	-17.3031
year#c.gini						
1970	5.865776	6.255297	0.94	0.348	-6.39438	18.12593
1980	24.20335	6.834194	3.54	0.000	10.80858	37.59812
1990	58.38273	6.881893	8.48	0.000	44.89447	71.87099
_cons	-6.535916	2.257841	-2.89	0.004	-10.9612	-2.110629
M						
hrate	.3317434	.0967132	3.43	0.001	.142189	.5212978
e.hrate	2.860571	.0558304	51.24	0.000	2.751145	2.969996
/sigma_u	2.686156	.1123355			2.474764	2.915605
/sigma_e	5.609948	.0612095			5.491253	5.731208

```
Wald test of spatial terms:          chi2(2) = 2685.83    Prob > chi2 = 0.0000
```

The `re` and `re sarpanel` estimators give appreciably different estimates for the coefficient of the spatial lag of `hrate` and for the autoregressive error term. Estimates of other terms are similar. It appears that some of the spatial-lag effect of `hrate` is being accounted for by the autoregressive form of the panel effects in the `sarpanel` model.

◁

Example 2: spxtregress, fe

The random-effects estimator assumes that the panel-level effects are uncorrelated with the covariates in the model. We can relax that assumption using the fixed-effects estimator.

We will fit fixed-effects models for the same data we used in example 1. Here's a nonspatial model fit with `xtreg, fe`.

```
. xtreg hrate ln_population ln_pdensity gini, fe
Fixed-effects (within) regression               Number of obs      =      5,648
Group variable: _ID                             Number of groups   =      1,412

R-sq:                                           Obs per group:
     within  = 0.0356                                         min =          4
     between = 0.0084                                         avg =        4.0
     overall = 0.0131                                         max =          4

                                                F(3,4233)          =      52.04
corr(u_i, Xb)  = -0.2819                        Prob > F           =     0.0000

------------------------------------------------------------------------------
       hrate |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
ln_populat~n |   -2.16467   1.702073    -1.27   0.204    -5.501627    1.172286
 ln_pdensity |   1.007573   1.659751     0.61   0.544    -2.246409    4.261555
        gini |   35.12694   2.816652    12.47   0.000     29.60483    40.64906
       _cons |   13.90421   10.91007     1.27   0.203    -7.485242    35.29366
-------------+----------------------------------------------------------------
     sigma_u |  5.2469262
     sigma_e |  5.7428609
         rho |  .45496484   (fraction of variance due to u_i)
------------------------------------------------------------------------------
F test that all u_i=0: F(1411, 4233) = 2.61                     Prob > F = 0.0000
```

We now use `spxtregress, fe` and include a spatial lag of the dependent variable `hrate`.

```
. spxtregress hrate ln_population ln_pdensity gini, fe dvarlag(M)
  (5648 observations)
  (5648 observations used)
  (data contain 1412 panels (places) )
  (weighting matrix defines 1412 places)

Performing grid search ... finished

Optimizing concentrated log likelihood:

Iteration 0:   log likelihood =  -13321.27
Iteration 1:   log likelihood =  -13321.27  (backed up)
Iteration 2:   log likelihood = -13321.269

Optimizing unconcentrated log likelihood:

Iteration 0:   log likelihood = -13321.269
Iteration 1:   log likelihood = -13321.269  (backed up)
```

```
Fixed-effects spatial regression                Number of obs      =      5,648
Group variable: _ID                             Number of groups   =      1,412
                                                Obs per group      =          4
                                                Wald chi2(4)       =     548.39
                                                Prob > chi2        =     0.0000
Log likelihood = -1.332e+04                     Pseudo R2          =     0.0146
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	-1.852636	1.662249	-1.11	0.265	-5.110586	1.405313
ln_pdensity	-.0352675	1.621715	-0.02	0.983	-3.21377	3.143235
gini	11.58058	3.001197	3.86	0.000	5.698348	17.46282
M						
hrate	.8982519	.0457977	19.61	0.000	.80849	.9880138
/sigma_e	5.608237	.0609629			5.490016	5.729004

```
Wald test of spatial terms:          chi2(1) = 384.69     Prob > chi2 = 0.0000
```

`spxtregress, fe` does not give an estimate of `/sigma_u` because the spatial fixed-effects estimator does not give consistent estimates for the levels of the panel fixed effects nor for their standard deviation. See *Methods and formulas*.

We cannot fit a fixed-effects model with all of the terms we included in example 1. The `i.year` dummies are constant within panel and the fixed-effects estimator is already conditional on constant effects for each panel and constant effects for each time. Models can include only variables that vary across both panels and time.

We cannot fit a time effect because time does not vary across panels, but we can fit a time-variable interaction because it varies across time and panels. This will model the effects of a variable over time.

In example 1, we found that gini was an important regressor and that the effect of gini differed across time. We will use Stata's factor-variable notation and add to the model c.gini#i.year, which is gini interacted by year without main effects.

```
. spxtregress hrate ln_population ln_pdensity c.gini#i.year, fe
> dvarlag(M) errorlag(M)
  (5648 observations)
  (5648 observations used)
  (data contain 1412 panels (places) )
  (weighting matrix defines 1412 places)
```

(*output omitted*)

```
Fixed-effects spatial regression                Number of obs     =      5,648
Group variable: _ID                             Number of groups  =      1,412
                                                Obs per group     =          4
                                                Wald chi2(7)      =     128.16
                                                Prob > chi2       =     0.0000
Log likelihood = -1.330e+04                     Pseudo R2         =     0.0001
```

hrate	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
hrate						
ln_populat~n	-2.169113	1.70931	-1.27	0.204	-5.519298	1.181073
ln_pdensity	-.7395584	1.638919	-0.45	0.652	-3.95178	2.472663
year#c.gini						
1960	4.637191	4.648658	1.00	0.319	-4.474012	13.74839
1970	11.15786	4.234693	2.63	0.008	2.858016	19.45771
1980	11.92355	4.158854	2.87	0.004	3.77235	20.07476
1990	11.13694	3.975612	2.80	0.005	3.344885	18.929
M						
hrate	.1251126	.2552473	0.49	0.624	-.3751629	.625388
e.hrate	1.604259	.1898228	8.45	0.000	1.232213	1.976305
/sigma_e	5.582721	.0606909			5.465027	5.702949

```
Wald test of spatial terms:          chi2(2) = 116.83     Prob > chi2 = 0.0000
```

We look at the effects:

```
. estat impact
progress   : 33%  67% 100%
Average impacts                                 Number of obs     =      5,648
```

	Delta-Method dy/dx	Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
ln_populat~n	-2.169186	1.709375	-1.27	0.204	-5.5195	1.181127
ln_pdensity	-.7395835	1.638973	-0.45	0.652	-3.951911	2.472744
gini	9.714218	4.112071	2.36	0.018	1.654707	17.77373
indirect						
ln_populat~n	-.2894662	.7155598	-0.40	0.686	-1.691938	1.113005
ln_pdensity	-.0986934	.3143279	-0.31	0.754	-.7147649	.517378
gini	1.29631	3.022576	0.43	0.668	-4.62783	7.22045
total						
ln_populat~n	-2.458653	2.065714	-1.19	0.234	-6.507378	1.590073
ln_pdensity	-.838277	1.867989	-0.45	0.654	-4.499469	2.822915
gini	11.01053	5.357526	2.06	0.040	.5099701	21.51109

The output shows the effects of `gini` across all the years. `estat impact` is smart enough to know that there are not `year` effects in the fixed-effects model. When it looks at the term `c.gini#i.year`, it only gives the effects for `gini`. If `year` were replaced by a variable that varied within time, `estat impact` would show the effects for that variable, too.

If we want to see how the effects of `gini` change across the years, we can use `if` with `estat impact` as we did in example 1.

```
. estat impact gini if year == 1960
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	Delta-Method dy/dx	Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	4.637349	4.648981	1.00	0.319	-4.474486	13.74918
indirect						
gini	.6188292	1.70156	0.36	0.716	-2.716167	3.953826
total						
gini	5.256178	5.794721	0.91	0.364	-6.101266	16.61362

```
. estat impact gini if year == 1970
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	11.15824	4.234355	2.64	0.008	2.859059	19.45743
indirect						
gini	1.489007	3.335444	0.45	0.655	-5.048344	8.026358
total						
gini	12.64725	5.00173	2.53	0.011	2.844039	22.45046

```
. estat impact gini if year == 1980
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	11.92396	4.158654	2.87	0.004	3.773148	20.07477
indirect						
gini	1.591188	3.62961	0.44	0.661	-5.522717	8.705093
total						
gini	13.51515	5.380726	2.51	0.012	2.96912	24.06118

```
. estat impact gini if year == 1990
progress   :100%
Average impacts                                 Number of obs     =      1,412
```

	dy/dx	Delta-Method Std. Err.	z	P>|z|	[95% Conf.	Interval]
direct						
gini	11.13732	3.975637	2.80	0.005	3.345217	18.92943
indirect						
gini	1.486215	3.459169	0.43	0.667	-5.293632	8.266063
total						
gini	12.62354	5.485123	2.30	0.021	1.872894	23.37418

There is no evidence of a trend in the average total effect of `gini` from the `fe` model.

◁

Stored results

spxtregress, fe and spxtregress, re store the following in e():

Scalars

e(N)	number of observations
e(N_g)	number of groups (panels)
e(g)	group size
e(k)	number of parameters
e(df_m)	model degrees of freedom
e(df_c)	degrees of freedom for test of spatial terms
e(ll)	log likelihood
e(iterations)	number of maximum log-likelihood estimation iterations
e(rank)	rank of e(V)
e(r2_p)	pseudo-R^2
e(chi2)	χ^2
e(chi2_c)	χ^2 for test of spatial terms
e(p)	p-value for model test
e(p_c)	p-value for test of spatial terms
e(converged)	1 if converged, 0 otherwise

Macros

e(cmd)	spxtregress
e(cmdline)	command as typed
e(depvar)	name of dependent variable
e(indeps)	names of independent variables
e(idvar)	name of ID variable
e(model)	fe, re, or re sarpanel
e(title)	title in estimation output
e(constant)	hasconstant or noconstant (re only)
e(dlmat)	name of spatial weighting matrix applied to *depvar*
e(elmat)	name of spatial weighting matrix applied to errors
e(chi2type)	Wald; type of model χ^2 test
e(vce)	oim
e(ml_method)	type of ml method
e(technique)	maximization technique
e(properties)	b V
e(estat_cmd)	program used to implement estat
e(predict)	program used to implement predict
e(marginsok)	predictions allowed by margins
e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved

Matrices

e(b)	coefficient vector
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(Hessian)	Hessian matrix
e(V)	variance–covariance matrix of the estimators

Functions

e(sample)	marks estimation sample

Methods and formulas

spxtregress, fe estimates the parameters of the SAR model with spatially autoregressive errors and fixed effects using the QML estimator derived by Lee and Yu (2010a).

spxtregress, re estimates the parameters of two different SAR models with spatially autoregressive errors and random effects. In the default model, the random effects enter the equation for the dependent variable linearly. This model and the ML estimator for its parameters were derived by Lee and Yu (2010b). When the sarpanel option is specified, the random effects are subject to the

same spatial autoregressive process as the idiosyncratic errors. This model and the ML estimator of its parameters were derived by Lee and Yu (2010b), which builds on the original formulation by Kapoor, Kelejian, and Prucha (2007). All of these papers build on theoretical work in Kelejian and Prucha (2001) and Lee (2004). We use the estimator derived by Baltagi and Liu (2011) to get initial values.

Methods and formulas are presented under the following headings:

Fixed-effects estimators

The Lee and Yu (2010a) SAR model for panel data with fixed effects is

$$\begin{aligned}\mathbf{y}_{nt} &= \lambda\mathbf{W}\mathbf{y}_{nt} + \mathbf{X}_{nt}\beta + \mathbf{c}_n + \mathbf{u}_{nt}\\ \mathbf{u}_{nt} &= \rho\,\mathbf{M}\,\mathbf{u}_{nt} + \mathbf{v}_{nt} \qquad\qquad t = 1, 2, \ldots, T\end{aligned} \tag{2}$$

where

$\mathbf{y}_{nt} = (y_{1t}, y_{2t}, \ldots, y_{nt})'$ is an $n \times 1$ vector of observations on the dependent variable for time period t;

$\mathbf{X}_{nt}$ is an $n \times k$ matrix of nonstochastic time-varying regressors for time period t. $\mathbf{X}_{nt}$ may also contain spatial lag of exogenous covariates;

$\mathbf{c}_n$ is an $n \times 1$ vector of individual effects;

$\mathbf{u}_{nt}$ is an $n \times 1$ vector of spatially lagged error;

$\mathbf{v}_{nt} = (v_{1t}, v_{2t}, \ldots, v_{nt})'$ is an $n \times 1$ vector of innovations, and v_{it} is i.i.d. across i and t with variance σ^2; and

$\mathbf{W}$ and $\mathbf{M}$ are $n \times n$ spatial weighting matrices.

`spxtregress, fe` estimates the parameters in this model by using the QML estimator derived by Lee and Yu (2010a). Lee and Yu (2010a) uses an orthogonal transformation to remove the fixed effects $\mathbf{c}_n$ without inducing dependence in the transformed errors. The transform $\mathbf{F}_{T,T-1}$ is part of $[\mathbf{F}_{T,T-1}, 1/\sqrt{T}\,\mathbf{l}_T]$, which is the orthonormal eigenvector matrix of $(\mathbf{I}_T - 1/T\,\mathbf{l}_T\mathbf{l}_T')$, where $\mathbf{I}_T$ is the $T \times T$ identity matrix and $\mathbf{l}_T$ is a $T \times 1$ vector of 1s. Kuersteiner and Prucha (2015) discuss this class of transforms.

For any $n \times T$ matrix $[\mathbf{z}_{n1}, \mathbf{z}_{n2}, \ldots, \mathbf{z}_{nT}]$, the transformed $n \times (T-1)$ matrix is defined as

$$[\widetilde{\mathbf{z}}_{n1}, \widetilde{\mathbf{z}}_{n2}, \ldots, \widetilde{\mathbf{z}}_{n,T-1}] = [\mathbf{z}_{n1}, \mathbf{z}_{n2}, \ldots, \mathbf{z}_{nT}]\mathbf{F}_{T,T-1}$$

Thus, the transformed model for (2) is

$$\begin{aligned}\widetilde{\mathbf{y}}_{nt} &= \lambda\mathbf{W}\widetilde{\mathbf{y}}_{nt} + \widetilde{\mathbf{X}}_{nt}\beta + \widetilde{\mathbf{u}}_{nt}\\ \widetilde{\mathbf{u}}_{nt} &= \rho\mathbf{M}\widetilde{\mathbf{u}}_{nt} + \widetilde{\mathbf{v}}_{nt} \qquad t = 1, 2, \ldots, T-1\end{aligned}$$

The transformed innovations $\widetilde{\mathbf{v}}_{nt}$ are uncorrelated for all i and t.

The log-likelihood function for the transformed model is

$$\ln L_{n,T}(\theta) = -\frac{n(T-1)}{2}\ln(2\pi\sigma^2) + (T-1)[\ln|\mathbf{S}_n(\lambda)| + \ln|\mathbf{R}_n(\rho)|] - \frac{1}{2\sigma^2}\sum_{t=1}^{T-1}\widetilde{\mathbf{v}}_{nt}'(\theta)\widetilde{\mathbf{v}}_{nt}(\theta)$$

where $\mathbf{S}_n(\lambda) = \mathbf{I}_n - \lambda\mathbf{W}$, $\mathbf{R}_n(\rho) = \mathbf{I}_n - \rho\,\mathbf{M}$, and $\theta = (\beta', \lambda, \rho, \sigma^2)'$.

Random-effects estimators

`spxtregress, re` fits two different random-effects SAR models for panel data. In the default model, the random effects enter the equation for $\mathbf{y}_{nt}$ linearly.

$$\begin{aligned}\mathbf{y}_{nt} &= \lambda\mathbf{W}\mathbf{y}_{nt} + \mathbf{Z}_{nt}\beta + \mathbf{c}_n + \mathbf{u}_{nt}\\ \mathbf{u}_{nt} &= \rho\mathbf{M}\,\mathbf{u}_{nt} + \mathbf{v}_{nt} \qquad\qquad t = 1, 2, \ldots, T\end{aligned} \tag{3}$$

where

$\mathbf{Z}_{nt}$ may contain time-variant and -invariant regressors;

$\mathbf{c}_n$ is random effects with mean 0 and variance $\sigma^2_{\mathbf{c}}$; and

all the other terms are defined as in (2).

When the `sarpanel` option is specified, `xtspregress, re` fits a model in which the random effects $\mathbf{c}_n$ are subject to the same spatial autoregressive process as the errors.

$$\begin{aligned}\mathbf{y}_{nt} &= \lambda\mathbf{W}\mathbf{y}_{nt} + \mathbf{Z}_{nt}\beta + \mathbf{u}_{nt}\\ \mathbf{u}_{nt} &= \rho\mathbf{M}\,\mathbf{u}_{nt} + \mathbf{c}_n + \mathbf{v}_{nt} \qquad\qquad t = 1, 2, \ldots, T\end{aligned} \tag{4}$$

When the $\mathbf{c}_n$ are treated as fixed effects and transformed out of the model, the default model in (3) is equivalent to the `sarpanel` model in (4). When treating the $\mathbf{c}_n$ as random effects, these two models are different.

For (3) or (4), we can stack all the time periods and write the equations as an $nT \times 1$ vector form

$$\mathbf{y}_{nT} = \lambda(\mathbf{I}_T \otimes \mathbf{W})\mathbf{y}_{nT} + \mathbf{Z}_{nT}\beta + \xi_{nT} \tag{5}$$

where

$\mathbf{y}_{nT} = (\mathbf{y}'_{n1}, \mathbf{y}'_{n2}, \ldots, \mathbf{y}'_{nt})'$ is an $nT \times 1$ vector of observations of the dependent variable for $i = 1, \ldots, n$ and $t = 1, \ldots, T$;

$\mathbf{v}_{nT} = (\mathbf{v}'_{n1}, \mathbf{v}'_{n2}, \ldots, \mathbf{v}'_{nt})'$ is an $nT \times 1$ vector of innovations;

$\mathbf{Z}_{nT} = \{\mathbf{Z}'_{n1}, \mathbf{Z}'_{n2}, \ldots, \mathbf{Z}_*(\rho)'\}'$ is an $nT \times k$ matrix of k regressors for $i = 1, \ldots, n$ and $t = 1, \ldots, T$; and

ξ_{nT} is the overall disturbance $nT \times 1$ vector.

For (3), the overall disturbance vector ξ_{nT} is

$$\xi_{nT} = \mathbf{l}_T \otimes \mathbf{c}_n + \{\mathbf{I}_T \otimes \mathbf{R}_n(\rho)^{-1}\}\mathbf{v}_{nT}$$

where $\mathbf{R}_n(\rho) = \mathbf{I}_n - \rho\mathbf{M}$. Its variance matrix is

$$\Omega_{nT}(\theta) = \sigma^2_{\mathbf{c}}\,(\mathbf{l}_T\mathbf{l}'_T \otimes \mathbf{I}_T) + \sigma^2\{\mathbf{I}_T \otimes \mathbf{R}_n(\rho)^{-1}\mathbf{R}'_n(\rho)^{-1}\}$$

For (4), the overall disturbance vector ξ_{nT} is

$$\xi_{nT} = \mathbf{l}_T \otimes \mathbf{R}_n(\rho)^{-1}\mathbf{c}_n + \{\mathbf{I}_T \otimes \mathbf{R}_n(\rho)^{-1}\}\mathbf{v}_{nT}$$

Its variance matrix is

$$\Omega_{nT}(\theta) = \sigma^2_{\mathbf{c}}\{\mathbf{l}_T\mathbf{l}'_T \otimes \mathbf{R}_n(\rho)^{-1}\mathbf{R}'_n(\rho)^{-1}\} + \sigma^2\{\mathbf{I}_T \otimes \mathbf{R}_n(\rho)^{-1}\mathbf{R}'_n(\rho)^{-1}\}$$

The log-likelihood function for (5) is

$$\ln L_{nT}(\theta) = -\frac{nT}{2}\ln(2\pi) - \frac{1}{2}\ln|\Omega_{nT}(\theta)| + T\ln|\mathbf{S}_n(\lambda)| - \frac{1}{2}\xi'_{nT}(\theta)\Omega_{nT}(\theta)^{-1}\xi_{nT}(\theta)$$

where $\mathbf{S}_n(\lambda) = \mathbf{I}_n - \lambda\mathbf{W}$, and $\theta = (\beta', \lambda, \rho, \sigma^2_{\mathbf{c}}, \sigma^2)'$.

References

Baltagi, B. H., and L. Liu. 2011. Instrumental variable estimation of a spatial autoregressive panel model with random effects. *Economics Letters* 111: 135–137.

Britt, C. L. 1994. Crime and unemployment among youths in the United States, 1958–1990: A time series analysis. *American Journal of Economics and Sociology* 53: 99–109.

Gini, C. 1909. Concentration and dependency ratios (in Italian). English translation in *Rivista di Politica Economica 1997* 87: 769–789.

Kapoor, M., H. H. Kelejian, and I. R. Prucha. 2007. Panel data models with spatially correlated error components. *Journal of Econometrics* 140: 97–130.

Kelejian, H. H., and I. R. Prucha. 2001. On the asymptotic distribution of the Moran I test statistic with applications. *Journal of Econometrics* 104: 219–257.

Kuersteiner, G. M., and I. R. Prucha. 2015. Dynamic spatial panel models: Networks, common shocks, and sequential exogeneity. Working paper, Department of Economics, University of Maryland. http://econweb.umd.edu/~prucha/Papers/WP_GMK_IRP_2015.pdf.

Lee, L.-F. 2004. Asymptotic distributions of quasi-maximum likelihood estimators for spatial autoregressive models. *Econometrica* 72: 1899–1925.

Lee, L.-F., and J. Yu. 2010a. Estimation of spatial autoregressive panel data models with fixed effects. *Journal of Econometrics* 154: 165–185.

——. 2010b. Some recent developments in spatial panel data models. *Regional Science and Urban Economics* 40: 255–271.

Messner, S. F., L. Anselin, D. F. Hawkins, G. Deane, S. E. Tolnay, and R. D. Baller. 2000. An Atlas of the Spatial Patterning of County-Level Homicide, 1960–1990. Pittsburgh: National Consortium on Violence Research.

Also see

[SP] **spxtregress postestimation** — Postestimation tools for spxtregress

[SP] **estat moran** — Moran's test of residual correlation with nearby residuals

[SP] **Intro** — Introduction to spatial data and SAR models

[SP] **spbalance** — Make panel data strongly balanced

[SP] **spivregress** — Spatial autoregressive models with endogenous covariates

[SP] **spmatrix** — Categorical guide to the spmatrix command

[SP] **spregress** — Spatial autoregressive models

[XT] **xtreg** — Fixed-, between-, and random-effects and population-averaged linear models

[U] 20 Estimation and postestimation commands

Title

spxtregress postestimation — Postestimation tools for spxtregress

Postestimation commands

The following postestimation command is of special interest after `spxtregress`:

Command	Description
`estat impact`	direct, indirect, and total impacts

The following postestimation commands are also available:

Command	Description
`contrast`	contrasts and ANOVA-style joint tests of estimates
`estat ic`	Akaike's and Schwarz's Bayesian information criteria (AIC and BIC)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estimates`	cataloging estimation results
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of coefficients
`lrtest`	likelihood-ratio test
`margins`	marginal means, predictive margins, marginal effects, and average marginal effects
`marginsplot`	graph the results from margins (profile plots, interaction plots, etc.)
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
`predict`	predictions, residuals, influence statistics, and other diagnostic measures
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of estimates
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

predict

Description for predict

`predict` creates a new variable containing predictions such as the reduced-form mean, the direct mean, the indirect mean, or the linear prediction.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] *newvar* [*if*] [*in*] [`,` *statistic*]

statistic	Description
Main	
`rform`	reduced-form mean; the default
`direct`	direct mean
`indirect`	indirect mean
`xb`	linear prediction

These statistics are only available in a subset of the estimation sample.

Options for predict

Main

`rform`, the default, calculates the reduced-form mean. It is the predicted mean of the dependent variable conditional on the independent variables and any spatial lags of the independent variables. See *Methods and formulas*.

`direct` calculates the direct mean. It is a unit's predicted contribution to its own reduced-form mean. The direct and indirect means sum to the reduced-form mean.

`indirect` calculates the indirect mean. It is the predicted sum of the other units' contributions to a unit's reduced-form mean.

`xb` calculates the predicted linear combination of the independent variables.

margins

Description for margins

`margins` estimates margins of response for reduced-form mean, direct mean, indirect mean, and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

`margins` [*marginlist*] [`,` *options*]

`margins` [*marginlist*] `,` `predict(`*statistic* ...`)` [`predict(`*statistic* ...`)` ...] [*options*]

statistic	Description
`rform`	reduced-form mean; the default
`direct`	direct mean
`indirect`	indirect mean
`xb`	linear prediction

For the full syntax, see [R] **margins**.

Remarks for margins

The computations that `margins` must do to calculate standard errors can sometimes be time consuming. Time will depend on the complexity of the spatial model and the number of spatial units in the data. You may want to fit your model with a subsample of your data, run `margins`, and extrapolate to estimate the time required to run `margins` on the full sample. See [P] **timer** and [P] **rmsg**.

estat impact

Description for estat impact

`estat impact` estimates the mean of the direct, indirect, and total impacts of independent variables on the reduced-form mean of the dependent variable.

Syntax for estat impact

```
estat impact [varlist] [if] [in] [, nolog]
```

varlist is a list of independent variables, including factor variables, taken from the fitted model. By default, all independent variables from the fitted model are used.

Options for estat impact

Main

`nolog` suppresses the calculation progress log that shows the percentage completed. By default, the log is displayed.

Remarks for estat impact

`estat impact` is essential for interpreting the output of `spxtregress`. See [SP] **Intro 7**, example 1 of [SP] **spregress**, and examples 1 and 2 of [SP] **spxtregress** for explanations and examples.

Stored results for estat impact

`estat impact` stores the following in `r()`:

Scalars	
`r(N)`	number of observations
Macros	
`r(xvars)`	names of independent variables
Matrices	
`r(b_direct)`	vector of estimated direct impacts
`r(Jacobian_direct)`	Jacobian matrix for direct impacts
`r(V_direct)`	estimated variance–covariance matrix of direct impacts
`r(b_indirect)`	vector of estimated indirect impacts
`r(Jacobian_indirect)`	Jacobian matrix for indirect impacts
`r(V_indirect)`	estimated variance–covariance matrix of indirect impacts
`r(b_total)`	vector of estimated total impacts
`r(Jacobian_total)`	Jacobian matrix for total impacts
`r(V_total)`	estimated variance–covariance matrix of total impacts

Methods and formulas

Methods and formulas are presented under the following headings:

Predictions

To motivate the predictions, consider the vector form of a spatial panel autoregressive model

$$\mathbf{y}_{nt} = \lambda\mathbf{W}\mathbf{y}_{nt} + \mathbf{X}_{nt}\boldsymbol{\beta} + \mathbf{c}_n + \boldsymbol{\epsilon}_{nt} \qquad t = 1, 2, \ldots, T \tag{1}$$

where

$\mathbf{y}_{nt}$ is the $n \times 1$ vector containing each unit's dependent-variable observations for time period t,

$\mathbf{W}\mathbf{y}_{nt}$ is a spatial lag of $\mathbf{y}_{nt}$,

$\mathbf{X}_{nt}$ is the matrix of independent-variable observations for time period t,

$\mathbf{c}_n$ are individual effects, which can be either fixed effects or random effects,

$\boldsymbol{\epsilon}_{nt}$ are the vector errors, and

λ and $\boldsymbol{\beta}$ are the coefficients.

Any spatial lags of the independent variables are assumed to be in $\mathbf{X}_{nt}$. Spatial lags of the error do not affect the reduced-form, direct, or indirect means, so they are not included in (1) for simplicity.

Reduced-form mean

Equation (1) represents the spatial autoregressive model as a system of equations. The solution to this system is

$$\mathbf{y}_{nt} = (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}_{nt}\boldsymbol{\beta} + \mathbf{c}_n + \boldsymbol{\epsilon}_{nt}) \tag{2}$$

To simplify later notation, we define $\widetilde{\mathbf{y}}_{nt}$ as $\mathbf{y}_{nt}$ minus the spatial spillover of the individual effects $\mathbf{c}_n$.

$$\begin{aligned}\widetilde{\mathbf{y}}_{nt} &= \mathbf{y}_{nt} - (\mathbf{I} - \lambda\mathbf{W})^{-1}\mathbf{c}_n \\ &= (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}_{nt}\boldsymbol{\beta} + \boldsymbol{\epsilon}_{nt})\end{aligned} \tag{3}$$

For the random-effects model, the individual effects $\mathbf{c}_n$ are treated as part of random errors. Thus, (2) implies that the mean of $\mathbf{y}_{nt}$ conditional on the independent variables and their spatial lags is

$$E(\mathbf{y}_{nt} \,|\, \mathbf{X}_{nt}, \mathbf{W}) = (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}_{nt}\boldsymbol{\beta}) \tag{4}$$

This is known as the reduced-form mean because the solution in (2) is known as the reduced form of the model. The predicted reduced-form mean substitutes estimates of λ and $\boldsymbol{\beta}$ into (4).

For the fixed-effects model, the individual effects $\mathbf{c}_n$ are treated as fixed effects, and they cannot be consistently estimated. The reduced-form prediction after `spxtregress, fe` is the conditional mean of $\widetilde{\mathbf{y}}_{nt}$ given the independent variables and their spatial lags:

$$E(\widetilde{\mathbf{y}}_{nt} \,|\, \mathbf{X}_{nt}, \mathbf{W}) = (\mathbf{I} - \lambda\mathbf{W})^{-1}(\mathbf{X}_{nt}\boldsymbol{\beta}) \tag{5}$$

Direct and indirect means

To define the direct mean and the indirect mean, let

$$\mathbf{S}_n = (\mathbf{I} - \lambda\mathbf{W})^{-1}$$

and let $\mathbf{S}_d$ be a matrix with diagonal elements of $\mathbf{S}_n$ on its diagonal and with all off-diagonal elements set to 0.

The direct means are

$$\mathbf{S}_d\mathbf{X}_{nt}\boldsymbol{\beta}$$

which capture the contributions of each unit's independent variables on its own reduced-form mean. Substituting estimates of λ and $\boldsymbol{\beta}$ produces the predictions.

The indirect means capture the contributions of the other units' independent variables on a unit's reduced-form mean. They are

$$\left\{(\mathbf{I} - \lambda\mathbf{W})^{-1} - \mathbf{S}_d\right\}\mathbf{X}_{nt}\boldsymbol{\beta}$$

Linear predictor

The linear predictor is $\mathbf{X}_{nt}\boldsymbol{\beta}$.

Impacts in random-effects models

The total impact of an independent variable $\mathbf{x}$ is the average of the marginal effects it has on the reduced-form mean of $\mathbf{y}_{nt}$,

$$\frac{1}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial E(\mathbf{y}_{it}|\mathbf{X}_{nt},\mathbf{W})}{\partial x_{jt}}$$

where $E(\mathbf{y}_{it}\,|\,\mathbf{X}_{nt},\mathbf{W})$ is the ith element of the vector $E(\mathbf{y}_{nt}\,|\,\mathbf{X}_{nt},\mathbf{W})$, whose formula is given in (3), and x_{jt} is the jth unit's value for $\mathbf{x}$ at time t.

The direct impact of an independent variable $\mathbf{x}$ is the average of the direct, or own, marginal effects:

$$\frac{1}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\partial E(\mathbf{y}_{it}\,|\,\mathbf{X}_{nt},\mathbf{W})}{\partial x_{it}}$$

The indirect impact of an independent variable $\mathbf{x}$ is the average of the indirect, or spillover, marginal effects:

$$\frac{1}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\frac{\partial E(\mathbf{y}_{it}\,|\,\mathbf{X}_{nt},\mathbf{W})}{\partial x_{jt}}$$

LeSage and Pace (2009, 36–37) call the average direct impact the "average total direct impact" and they call the average indirect impact the "average total indirect impact".

Impacts in fixed-effects models

The total impact of an independent variable $\mathbf{x}$ is the average of the marginal effects it has on the reduced-form mean of $\widetilde{\mathbf{y}}_{nt}$,

$$\frac{1}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial E(\widetilde{\mathbf{y}}_{it}\,|\,\mathbf{X}_{nt},\mathbf{W})}{\partial x_{jt}}$$

where $E(\widetilde{\mathbf{y}}_{it}\,|\,\mathbf{X}_{nt},\mathbf{W})$ is the ith element of the vector $E(\widetilde{\mathbf{y}}_{nt}\,|\,\mathbf{X}_{nt},\mathbf{W})$, whose formula is given in (5), and x_{jt} is the jth unit's value for $\mathbf{x}$ at time t.

The direct impact of an independent variable $\mathbf{x}$ is the average of the direct, or own, marginal effects:

$$\frac{1}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\partial E(\widetilde{\mathbf{y}}_{it}\,|\,\mathbf{X}_{nt},\mathbf{W})}{\partial x_{it}}$$

The indirect impact of an independent variable $\mathbf{x}$ is the average of the indirect, or spillover, marginal effects:

$$\frac{1}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\frac{\partial E(\widetilde{\mathbf{y}}_{it}\,|\,\mathbf{X}_{nt},\mathbf{W})}{\partial x_{jt}}$$

Reference

LeSage, J., and R. K. Pace. 2009. *Introduction to Spatial Econometrics*. Boca Raton, FL: Chapman & Hall/CRC.

Also see

[SP] **spxtregress** — Spatial autoregressive models for panel data

[U] 20 Estimation and postestimation commands

Glossary

adjacent. Two areas are said to be adjacent if they share a border. Also see *contiguity matrix*.

AR(1). See *autoregressive errors*.

areal data. Areal data is a term for data on areas. SAR models are appropriate for areal and lattice data.

areas. Areas is an informal term for geographic units.

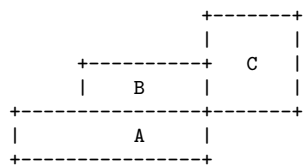
attributes. Attributes is the name given to the variables included in standard-format shapefiles.

autoregressive errors. Spatially autoregressive errors account for spatially lagged correlation of the residuals. ρ is the correlation parameter. It is not a correlation coefficient, but it shares certain properties with correlation coefficients. It is bounded by -1 and 1, and 0 has the same meaning, namely, no correlation.

autoregressive models. Spatially autoregressive models include a spatially lagged dependent variable or spatially autoregressive errors. See [SP] **Intro 1**.

balanced and strongly balanced. Panel data are balanced if each panel contains the same number of observations. They are strongly balanced if they record data for the same times (subcategory).

border and vertex. Consider the following map:



A and B share a border because there is a line segment separating them. For the same reasons, B and C share a border.

A and C share a vertex. They have only a single point in common.

How should you treat vertex-only adjacency? This issue arises when constructing a contiguity matrix. It is up to you whether a vertex in common is sufficient to label the areas as contiguous. Vertex-only adjacency occurs frequently when the shapes of the geographic units are rectangular.

choropleth map. A choropleth map is a map in which shading or coloring is used to indicate values of a variable within areas.

contiguity matrix and ex post contiguity matrix. A contiguity matrix is a symmetric matrix containing 0s and 1s before normalization, with 1s indicating that areas are adjacent.

`spmatrix create contiguity` creates contiguity matrices and other matrices that would not be considered contiguity matrices by the above definition. It can create first-order neighbor matrices containing 0s and 1s. That is a contiguity matrix. It can create first- and second-order neighbor matrices containing 0s and 1s. That is not a contiguity matrix strictly speaking. And it can create other matrices where second-order neighbors are recorded as 0.5 or any other value of your choosing.

And finally, even if the matrix started out as a contiguity matrix strictly speaking, after normalization the two values that it contains are 0 and c.

As a result, commands like `spmatrix summarize` use a different definition for contiguity matrix.

An ex post contiguity matrix is any matrix in which all values are either 0 or c, a positive constant. It is meaningful to count neighbors in such cases. Thus, the matrix `W2` created by typing

```
. spmatrix create contiguity W2, second
```

is an ex post contiguity matrix, and the matrix `W` created by typing

```
. spmatrix create contiguity W, first second(0.5)
```

is not.

coordinate system. A coordinate system is the encoding used by numbers used to designate locations. Latitude and longitude are a coordinate system. As far as Sp is concerned, the only other coordinate system is planar. Planar coordinates are also known as rectangular or Cartesian coordinates. In theory, standard-format shapefiles provide planar coordinates. In practice, they sometimes use latitude and longitude, but standards for encoding the system used are still developing. See [SP] **spdistance** for a more complete description, and see [SP] **Intro 4** for how you can determine whether coordinates are planar or latitude and longitude.

covariate. See *explanatory variable*.

cross-sectional data. Cross-sectional data contain one observation per spatial unit. Also see *panel data*.

.dbf files. See *shapefiles*.

dependent variable. See *outcome variable*.

distance matrix. A distance matrix is a spatial weighting matrix based on some function of distance. Usually that function is $1/\text{distance}$, and the matrix is then called an inverse-distance spatial weighting matrix.

explanatory variable. An explanatory variable is a variable that appears on the right-hand side of the equation used to "explain" the values of the outcome variable.

FIPS codes. FIPS stands for federal information processing standard. FIPS codes are used for designating areas of the United States. At the most detailed level is the five-digit FIPS county codes, which range from 01001 for Autauga County in Alabama to 78030 for St. Thomas Island in the Virgin Islands. The FIPS county code includes counties, U.S. possessions, and freely associated areas.

The first two digits of the five-digit code are FIPS state codes. The two-digit code covers states, U.S. possessions, and freely associated areas.

The five-digit code appears in some datasets as the two-digit state code plus a three-digit county code. The full five-digit code is formed by joining the two-digit and three-digit codes.

geographic units. Geographic units is the generic term for places or areas such as zip-code areas, census blocks, cities, counties, countries, and the like. The units do not need to be based on geography. They could be network nodes, for instance. In this manual, we also use the words places and areas for the geographic units. Also see *spatial units*.

GIS data. GIS is an acronym for geographic information system. Some of the information in shapefiles is from such systems.

ID, _ID variable. An ID variable is a variable that uniquely identifies the observations. Sp's `_ID` variable is an example of an ID variable that uniquely identifies the geographic units. Sp's `_ID` variable is a numeric variable that uniquely identifies the observations in cross-sectional data and uniquely identifies the panels in panel data.

idistance spatial weighting matrix. An idistance spatial weighting matrix is Sp jargon for an inverse-distance spatial weighting matrix.

i.i.d. I.i.d. stands for independent and identically distributed. A variable is i.i.d. when each observation of the variable has the same probability distribution as all the other observations and all are independent of one another.

imported spatial weighting matrix. An imported spatial weighting matrix is a spatial weighting matrix created with the `spmatrix import` command.

instrumental variables. Instrumental variables are variables related to the covariates (explanatory variables) and unrelated to the errors (residuals).

inverse-distance spatial weighting matrix. An inverse-distance spatial weighting matrix is a matrix in which the elements $W_{i,j}$ before normalization contain the reciprocal of the distance between places j and i. The term is also used for inverse-distance matrices in which places farther apart than a specified distance are set to 0.

lags. See *spatial lags*.

latitude and longitude. See *coordinate system*.

lattice data. Lattice data are a kind of area data. In lattice data, all places are vertices appearing on a grid. SAR models are appropriate for lattice data and areal data.

neighbors, first- and second-order. First-order neighbors share borders. Second-order neighbors are neighbors of neighbors.

normalized spatial weighting matrix. A normalized spatial weighting matrix is a spatial weighting matrix multiplied by a constant to improve numerical accuracy and to make nonexplosive autoregressive parameters bounded by -1 and 1. See *Choosing weighting matrices and their normalization* in [SP] **spregress** for details about normalization.

outcome variable (dependent variable). The outcome variable of a model is the variable appearing on the left-hand side of the equation. It is the variable being "explained" or predicted.

panel data. Panel data contain data on geographic units at various times. Each observation contains data on a geographic unit at a particular time, and thus the data contain multiple observations per geographic unit. Also see *cross-sectional data*.

places. Places is an informal term for geographic units.

planar coordinates. See *coordinate system*.

proximity matrix. Proximity matrix is another word for distance matrix.

SAR. SAR stands for spatial autoregressive or simultaneous autoregressive, which themselves mean the same thing but are used by researchers in different fields. See *autoregressive models* and *autoregressive errors*.

shapefiles. Shapefiles are files defining maps and more that you find on the web. A shapefile might be *name*`.zip`. *name*`.zip` contains *name*`.shp`, *name*`.dbf`, and files with other suffixes.

In this manual, shapefiles are also the shapefiles as described above translated into Stata format. They are Stata datasets named *name*`_shp.dta`.

To distinguish the two meanings, we refer to standard-format and Stata-format shapefiles.

Sp. Sp stands for spatial and refers to the SAR system described in this manual.

Sp data. Sp data are data that have been `spset`, whether directly or indirectly. You can type `spset` without arguments to determine whether your data are `spset`.

spatial lags. Spatial lags are the spatial analogy of time-series lags. In time series, the lag of x_t is x_{t-1}. In spatial analysis, the lag of x_i—x in place i—is a weighted sum of x in nearby places given by $\mathbf{Wx}$. See [SP] **Intro 1**.

spatial units. Spatial units is the term we use for the units measuring distance when the coordinates are planar. For instance, New York and Boston might be recorded in planar units as being at $(\texttt{\_CX}, \texttt{\_CY}) = (1.3, 7.836)$ and $(1.447, 7.118)$. In that case, the distance between them is 0.0284 spatial units. Because they are about 190 miles apart, evidently a spatial unit is 6,690 miles. Also see [SP] **spdistance**.

spatial weighting matrix. A spatial weighting matrix is square matrix $\mathbf{W}$. $\mathbf{Wx}$ plays the same role in spatial analysis that L.$\mathbf{x}$ plays in time-series analysis. One can think of $\mathbf{W}$'s elements as recording the potential spillover for place j to i.

Spatial weighting matrices have zero on the diagonal and nonzero or zero values elsewhere. A contiguity spatial weighting matrix would have 0s and 1s. $W_{i,j} = W_{j,i}$ would equal 1 when i and j were neighbors.

The scale in which the elements of spatial weighting matrices are recorded is irrelevant. See [SP] **Intro 2**.

spatially autoregressive errors. See *autoregressive errors*.

spillover effects. Spillover effects and potential spillover effects are the informal words we use to describe the elements of a spatial weighting matrix. $W_{i,j}$ records the (potential) spillover from place j to i. See [SP] **Intro 2**.

standard-format shapefile. See *shapefiles*.

Stata-format shapefile. See *shapefiles*.

strongly balanced. See *balanced and strongly balanced*.

time variable. The time variable is the variable in panel data that identifies the second level of the panel. The variable is not required to measure time, but it usually does.

user-defined matrix. A user-defined matrix is a spatial weighting matrix created by typing

```
spmatrix userdefined
spmatrix fromdata
spmatrix spfrommata
```

vertex. See *border and vertex*.

Subject and author index

See the combined subject index and the combined author index in the *Glossary and Index*.