

## 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.

<a href="#">Intro 1</a>	A brief introduction to SAR models
<a href="#">Intro 2</a>	The <b>W</b> matrix
<a href="#">Intro 4</a>	Preparing data: Data with shapefiles
<a href="#">Intro 5</a>	Preparing data: Data containing locations (no shapefiles)
<a href="#">Intro 6</a>	Preparing data: Data without shapefiles or locations
<a href="#">Intro 7</a>	Example from start to finish
<a href="#">Intro 8</a>	The Sp estimation commands

## Remarks and examples

Remarks are presented under the following headings:

- Three types of Sp data*
  - Type 1: Data with shapefiles*
  - Type 2: Data without shapefiles but including location information*
  - Type 3: Data without shapefiles or location information*
- Sp can be used with cross-sectional data or panel data*
- ID variables for cross-sectional data*
- ID variables for panel data*

## 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 **W** matrices easy.

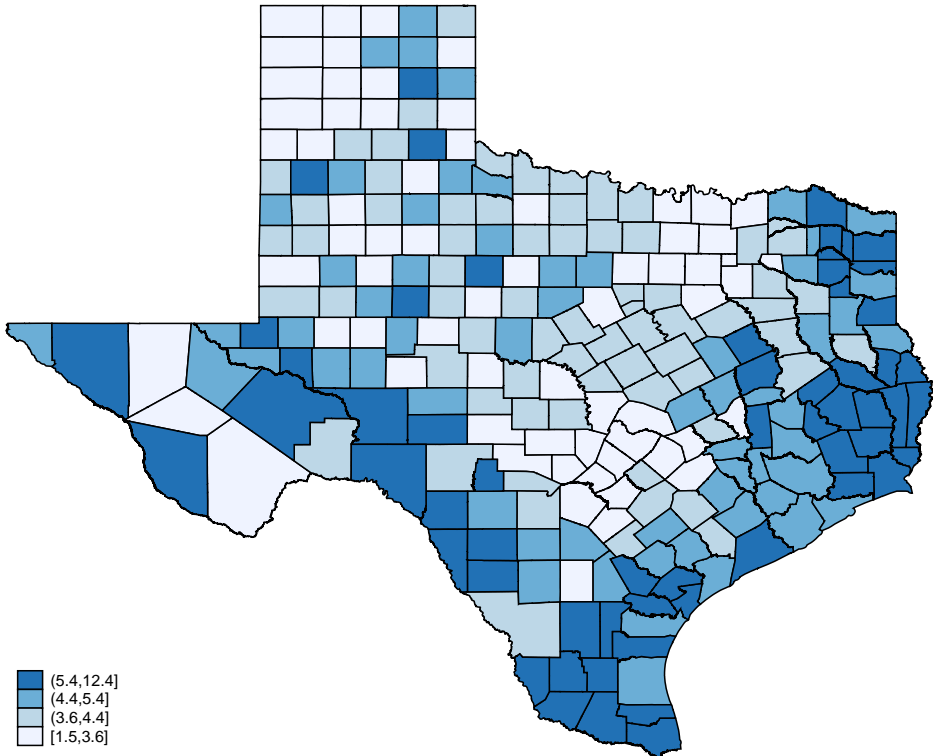
Alternatively, your data could contain the  $(x, y)$  coordinates, and then you do not need shapefiles. However, you still might want them because then you can draw [choropleth](#) maps.

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 **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 **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 **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, ..., but that is not required. Another dataset might contain US 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
<code>area_id</code>	<code>year</code>

`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	ASISD	...		
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

