Spatial Econometrics

James P. LeSage
Department of Economics
University of Toledo
CIRCULATED FOR REVIEW

December, 1998

Preface

This text provides an introduction to spatial econometrics as well as a set of MATLAB functions that implement a host of spatial econometric estimation methods. The intended audience is faculty and students involved in modeling spatial data sets using spatial econometric methods. The MATLAB functions described in this book have been used in my own research as well as teaching both undergraduate and graduate econometrics courses.

Toolboxes are the name given by the MathWorks to related sets of MATLAB functions aimed at solving a particular class of problems. Toolboxes of functions useful in signal processing, optimization, statistics, finance and a host of other areas are available from the MathWorks as add-ons to the standard MATLAB software distribution. I use the term *Econometrics Toolbox* to refer to my collection of function libraries described in a manual entitled *Applied Econometrics using MATLAB* available at http://www.econ.utoledo.edu.

The MATLAB spatial econometrics functions used to apply the spatial econometric models discussed in this text rely on many of the functions in the *Econometrics Toolbox*. The spatial econometric functions constitute a "library" within the broader set of econometric functions. To use the spatial econometrics functions library you need to install the entire set of *Econometrics Toolbox* functions in MATLAB. The spatial econometrics functions library is part of the *Econometrics Toolbox* and will be installed and available for use as well as the econometrics functions.

Researchers currently using Gauss, RATS, TSP, or SAS for econometric programming might find switching to MATLAB advantageous. MATLAB software has always had excellent numerical algorithms, and has recently been extended to include: sparse matrix algorithms and very good graphical capabilities. MATLAB software is available on a wide variety of computing platforms including mainframe, Intel, Apple, and Linux or Unix workstations. A Student Version of MATLAB is available for less than $100. This version is limited in the size of problems it can solve, but many of the ex-

amples in this text rely on a small data sample with 49 observations that can be used with the Student Version of MATLAB.

The collection of around 450 functions and demonstration programs are organized into libraries, with approximately 30 spatial econometrics library functions described in this text. For those interested in other econometric functions or in adding programs to the spatial econometrics library, see the manual for the *Econometrics Toolbox*. The 350 page manual provides many details regarding programming techniques used to construct the functions and examples of adding new functions to the *Econometrics Toolbox*. This text does not focus on programming methods. The emphasis here is on applying the existing spatial econometric estimation functions to modeling spatial data sets.

A consistent design was implemented that provides documentation, example programs, and functions to produce printed as well as graphical presentation of estimation results for all of the econometric functions. This was accomplished using the "structure variables" introduced in MATLAB Version 5. Information from econometric estimation is encapsulated into a single variable that contains "fields" for individual parameters and statistics related to the econometric results. A thoughtful design by the MathWorks allows these structure variables to contain scalar, vector, matrix, string, and even multi-dimensional matrices as fields. This allows the econometric functions to return a single structure that contains all estimation results. These structures can be passed to other functions that can intelligently decipher the information and provide a printed or graphical presentation of the results.

The *Econometrics Toolbox* along with the spatial econometrics library functions should allow faculty to use MATLAB in undergraduate and graduate level courses with absolutely no programming on the part of students or faculty.

In addition to providing a set of spatial econometric estimation routines and documentation, the book has another goal, applied modeling strategies and data analysis. Given the ability to easily implement a host of alternative models and produce estimates rapidly, attention naturally turns to which models and estimates work best to summary a spatial data sample. Much of the discussion in this text is on these issues.

This text is provided in Adobe PDF format for online use. It attempts to draw on the unique aspects of a computer presentation platform. The ability to present program code, data sets and applied examples in an online fashion is a relatively recent phenomena, so issues of how to best accomplish a useful online presentation are numerous. For the online text the following

features were included in the PDF document.

1. A detailed set of "bookmarks" that allow the reader to jump to any section or subsection in the text.

2. A detailed set of "bookmarks" that allow the reader to jump to an examples or figures in the text.

3. A set of "bookmarks" that allow the reader to view the spatial datasets and documentation for the datasets using a Web browser.

4. A set of "bookmarks" that allow the reader to view all of the example programs using a Web browser.

All of the examples in the text as well as the datasets are available offline as well on my Web site: http://www.econ.utoledo.edu under the MATLAB gallery icon.

Finally, there are obviously omissions, bugs and perhaps programming errors in the *Econometrics Toolbox* and the spatial econometrics library functions. This would likely be the case with any such endeavor. I would be grateful if users would notify me when they encounter problems. It would also be helpful if users who produce generally useful functions that extend the toolbox would submit them for inclusion. Much of the econometric code I encounter on the internet is simply too specific to a single research problem to be generally useful in other applications. If econometric researchers are serious about their newly proposed estimation methods, they should take the time to craft a generally useful MATLAB function that others could use in applied research. Inclusion in the spatial econometrics function library would have the added benefit of introducing new research methods to faculty and their students.

The latest version of the *Econometrics Toolbox* functions can be found on the Internet at: http://www.econ.utoledo.edu under the MATLAB gallery icon. Instructions for installing these functions are in an Appendix to this text along with a listing of the functions in the library and a brief description of each.

Contents

List of Examples

List of Figures

Chapter 1

Introduction

This chapter provides an overview of the nature of spatial econometrics. An applied model-based approach is taken where various spatial econometric methods are introduced in the context of spatial data sets and associated models based on the data. The remaining chapters of the text are organized along the lines of alternative spatial econometric estimation procedures. Each chapter illustrates applications of a different econometric estimation method and provides references to the literature regarding these methods.

Section 1.1 sets forth the nature of spatial econometrics and discusses differences with traditional econometrics. We will see that spatial econometrics is characterized by: 1) spatial dependence between sample data observations at various points in the Cartesian plane, and 2) spatial heterogeneity that arises from relationships or model parameters that vary with our sample data as we move over the Cartesian plane.

The nature of spatially dependent or spatially correlated data is taken up in Section 1.2 and spatial heterogeneity is discussed in Section 1.3. Section 1.4 takes up the subject of how we formally incorporate the locational information from spatial data in econometric models. In addition to the theoretical discussion of incorporating locational information in econometric models, Section 1.4 provides a preview of alternative spatial econometric estimation methods that will be covered in Chapters 2 through 4.

Finally, Section 1.5 describes software design issues related to a spatial econometric function library based on MATLAB software from the MathWorks Inc. Functions are described throughout the text that implement the spatial econometric estimation methods discussed. These functions provide a consistent user-interface in terms of documentation and related functions

that provided printed as well as graphical presentation of the estimation results. Section 1.5 introduces the spatial econometrics function library which is part of a broader collection of econometric estimation functions available in my public domain *Econometrics Toolbox*.

1.1 Spatial econometrics

Applied work in regional science relies heavily on sample data that is collected with reference to location measured as points in space. The subject of how we incorporate the locational aspect of sample data is deferred until Section 1.4. What distinguishes spatial econometrics from traditional econometrics? Two problems arise when sample data has a locational component: 1) spatial dependence exists between the observations and 2) spatial heterogeneity occurs in the relationships we are modeling.

Traditional econometrics has largely ignored these two issues that violate the traditional Gauss-Markov assumptions used in regression modeling. With regard to spatial dependence between the observations, recall that Gauss-Markov assumes the explanatory variables are fixed in repeated sampling. Spatial dependence violates this assumption, a point that will be made clear in the next section. This gives rise to the need for alternative estimation approaches. Similarly, spatial heterogeneity violates the Gauss-Markov assumption that a single linear relationship exists across the sample data observations. If the relationship varies as we move across the spatial data sample, alternative estimation procedures are needed to successfully model this type of variation and draw appropriate inferences.

The subject of this text is alternative estimation approaches that can be used when dealing with spatial data samples. A subject that is seldom discussed in traditional econometrics textbooks. For example, no discussion of issues and models related to spatial data samples can be found in Amemiya (1985), Chow (1983), Dhrymes (1978), Fomby et al. (1984), Green (1997), Intrilligator (1978), Kelijian and Oates (1989), Kmenta (1986), Maddala (1977), Pindyck and Rubinfeld (1981), Schmidt (1976), and Vinod and Ullah (1981).

Anselin (1988) provides a complete treatment of many facets of spatial econometrics which this text draws upon. In addition to introducing ideas set forth in Anselin (1988), this presentation includes some more recent approaches based on Bayesian methods applied to spatial econometric models. In terms of focus, the materials presented here are more applied than Anselin (1988), providing program functions and illustrations of hands-

on approaches to implementing the estimation methods described. Another departure from Anselin (1988) is in the use of sparse matrix algorithms available in the MATLAB software to implement spatial econometric estimation procedures. These implementation details represent previously unpublished material that describes a set of programs (freely available) for solving large-scale spatial econometric problems involving thousands of observations in a few minutes on a modest desktop computer. Students as well as researchers can use these programs with absolutely no programming to implement some of the latest estimation procedures on large-scale spatial data sets.

Of course another distinction of the presentation here is the interactive aspect of a Web-based format that allows a hands-on approach that provides links to code, sample data and examples.

1.2 Spatial dependence

Spatial dependence in a collection of sample data observations refers to the fact that one observation associated with a location which we might label i depends on other observations at locations $j \neq i$. Formally, we might state:

$$y_i = f(y_j), i = 1, \ldots, n \quad j \neq i \tag{1.1}$$

Note that we allow the dependence to be among several observations, as the index i can take on any value from $i = 1, \ldots, n$. Why would we expect sample data observed at one point in space to be dependent on values observed at other locations? There are two reasons commonly given. First, data collection of observations associated with spatial units such as zip-codes, counties, states, census tracts and so on, might reflect measurement error. This would occur if the administrative boundaries for collecting information do not accurately reflect the nature of the underlying process generating the sample data. As an example, consider the case of unemployment rates and labor force measures. Because laborers are mobile and can cross county or state lines to find employment in neighboring areas, labor force or unemployment rates measured on the basis of where people live could exhibit spatial dependence.

Consider the following illustration that demonstrates how measurement problems could produce a set of spatial unemployment rate observations that appear to be related to location. Recall that the labor force is measured as persons actively seeking employment and the unemployment rate represents the percentage of those actively seeking employment who are not employed.

Suppose that labor force is measured by place of residence and employment is measured by place of work.

Consider six regions with differing levels of labor force measured by place of residence and employment measured by place of work shown in Figure 1.1. Assume the true unemployment rate is 25% in all six regions, exhibiting a uniform distribution across space. We also assume that regions (3) and (4) are at the center with (2) and (5) being neighboring regions and (1) and (6) farthest from the center. In other words, the regions are assumed to occupy a map configuration identical to that shown in Figure 1.1.

(1)	(2)	(3)	(4)	(5)	(6)	
1,000	2,000	3,000	3,000	2,000	1,000	Labor force by residence
0	1,000	3,000	3,000	1,000	0	Employment by place
100%	50%	0%	0%	50%	100%	Measured rate of Unemployment
25%	25%	25%	25%	25%	25%	Acutal rate of Unemployment

Figure 1.1: Measurement error and spatial dependence

If all employment were in the four central regions (2,3,4,5) as shown in Figure 1.1, the measured unemployment rates would take on the pattern shown. One might infer from the measured unemployment rates that a distinctive spatial pattern of unemployment exists, and that distance from the central regions (3) and (4) is an important determinant of unemployment. Yet, as indicated, the true pattern of unemployment exhibits a uniform dis-

tribution across the spatial dimension of the data sample.

A second and perhaps more important reason we would expect spatial dependence is that the spatial dimension of economic activity may truly be an important aspect of a modeling problem. Regional science is based on the premise that location and distance are important forces at work in human geography and market activity. All of these notions have been formalized in regional science theory that relies on notions of spatial interaction and diffusion effects, hierarchies of place and spatial spillovers.

As a concrete example of this type of spatial dependence, we took a sample of 35,000 homes that sold within the last 5 years in Lucas county, Ohio. Using the latitude-longitude coordinates, the distance of each house was calculated from the center of the city of Toledo (which is in Lucas county). The 35,000 homes were then sorted by distance from the center and grouped into seven batches of 5,000 homes. Distributions were constructed for a variable representing the year each house was built for all seven groups. We would expect to see newer homes farther away from the city and older homes near the center.

Figure 1.2 shows the seven house age distributions where it is clear that the newest homes are in the group of 5,000 homes farthest from the center of the city. There are some very old muncipalities surrounding the city of Toledo that contain older historic homes and these appear in some of the distributions. We also see a distinct pattern where all house construction was adversely affected by the great depression in the 1930's and the war in the 1940's.

The fact that these distributions are very different as we move over space away from the center of the city, with similarity within the seven samples of 5,000 homes grouped by distance needs to be taken into account when we attempt to model a spatial sample such as this. We also need to be aware of the common influence on all of the distributions exerted by the depression and the war.

Of course, this pattern of older homes in the central city and newer homes in developments farther from the center has numerous regional science explanations. Nonetheless, this fact leads to the distinctive spatial pattern that would be indicative of spatial dependence in this variable vector.

Spatial dependence arising from underlying regional interactions often observed in regional science data samples suggests the need to quantify and model the nature of the unspecified functional dependence f, set forth in (1.1). Before turning attention to this task, the next section discusses the other underlying condition leading to a need for spatial econometrics — spatial heterogeneity.

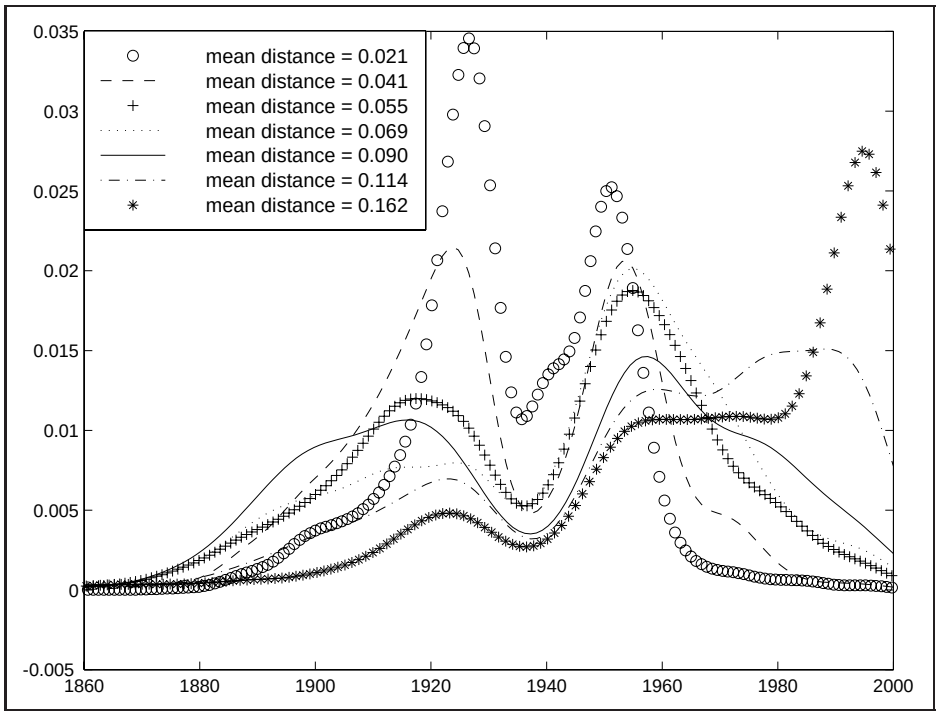

Figure 1.2: Distributions of house age versus distance

1.3 Spatial heterogeneity

The term spatial heterogeneity refers to variation in relationships over space. In the most general case consider that we might expect a different relationship to hold for every point in space. Formally, we write a linear relationship depicting this as:

$$y_i = X_i\beta_i + \varepsilon_i \tag{1.2}$$

Where i indexes observations collected at $i = 1, \ldots, n$ points in space, X_i represents a (1 x k) vector of explanatory variables with an associated set of parameters β_i, y_i is the dependent variable at observation (or location) i and ε_i denotes a stochastic disturbance in the linear relationship.

Given a sample of n data observations, we could not hope to estimate a set of n parameter vectors β_i because there are degrees of freedom problems. We simply do not have enough sample data information with which to produce estimates for every point in space. To proceed with analysis of

the relationship, we must provide a specification for variation over space. This specification must be parsimonious, that is, only a handful of parameters can be used in the specification. A large amount of spatial econometric research centers on alternative parsimonious specifications for modeling variation over space. Questions arise regarding: 1) how sensitive the inferences are to a particular specification regarding spatial variation?, 2) is the specification consistent with the sample data information?, 3) how do competing specifications perform and what inferences do they provide?, and a host of other issues that will be explored in this text.

One can also view the specification task as one of placing restrictions on the nature of variation in the relationship over space. For example, suppose we classified our spatial observations into urban and rural regions. We could then restrict our analysis to two relationships, one homogeneous across all urban observational units and another for the rural units. This raises a number of questions: 1) are two relations consistent with the data, or is there evidence to suggest more than two?, 2) is there a trade-off between efficiency in the estimates and the number of restrictions we use?, 3) are the estimates biased if the restrictions are inconsistent with the sample data information?, and other issues we will explore.

1.4 Quantifying location in our models

A first task we must undertake before we can ask questions about spatial dependence and heterogeneity is quantification of the locational aspects of our sample data. Given that we can always map a set of spatial data observations, we have two sources of information on which we can draw.

The location in Cartesian space represented by latitude and longitude is one source of information. This information would also allow us to calculate distances from any point in space, or the distance of observations located at distinct points in space to observations at other locations. Spatial dependence should conform to the fundamental theorem of regional science, distance matters. Observations that are near each other should reflect a greater degree of spatial dependence than those more distant from each other. In other words, the strength of spatial dependence between observations should decline with the distance between observations.

The second source of locational information is contiguity, reflecting the relative position in space of one regional unit of observation to other such units. Measures of contiguity rely on a knowledge of the size and shape of the observational units depicted on a map. From this, we can determine which

units are neighbors (have borders that touch) or represent observational units in reasonable proximity to each other. Regarding spatial dependence, neighboring units should exhibit a higher degree of spatial dependence than units located far apart.

I note in passing that these two types of information are not necessarily different. Given the latitude-longitude coordinates of an observation, we could construct a contiguity structure by defining a "neighboring observation" as one that lies within a certain distance. Consider also, given the centroid coordinates of a set of observations associated with contiguous map regions, we can calculate distances between the regions (or observations).

We will illustrate how both types of locational information can be used in spatial econometric modeling. We first take up the issue of quantifying spatial contiguity, which is used in the models presented in Chapter 2.

1.4.1 Quantifying spatial contiguity

Figure 1.2 shows a hypothetical example of five regions as they would appear on a map. We wish to construct a 5 by 5 binary matrix W containing 25 elements taking values of 0 or 1 that captures the notion of "connectiveness" between the five entities depicted in the map configuration. We record in each row of the matrix W a set of contiguity relations associated with one of the five regions. For example the matrix element in row 1, column 2 would record the presence (represented by a 1) or absence (denoted by 0) of a contiguity relationship between regions 1 and 2. As another example, the row 3, column 4 element would reflect the presence or absence of contiguity between regions 3 and 4. Of course, a matrix constructed in such fashion must be symmetric — if regions 3 and 4 are contiguous, so are regions 4 and 3.

It turns out there are an embarrassingly large number of ways to accomplish our task. Below, we enumerate some of the alternative ways we might define a binary matrix W that represent alternative definitions of the "contiguity" relationships between the five entities in Figure 1.3. For the enumeration below, start with a matrix filled with zeros, then consider the following alternative ways to define the presence of a contiguity relationship.

> Linear contiguity: Define $W_{ij} = 1$ for entities that share a common edge to the immediate right or left of the region of interest. For row 1, where we record the relations associated with region 1, we would have all $W_{1j} = 0, j = 1, \ldots, 5$. On the other hand, for row 5, where we record relationships involving region 5, we would have $W_{53} = 1$ and

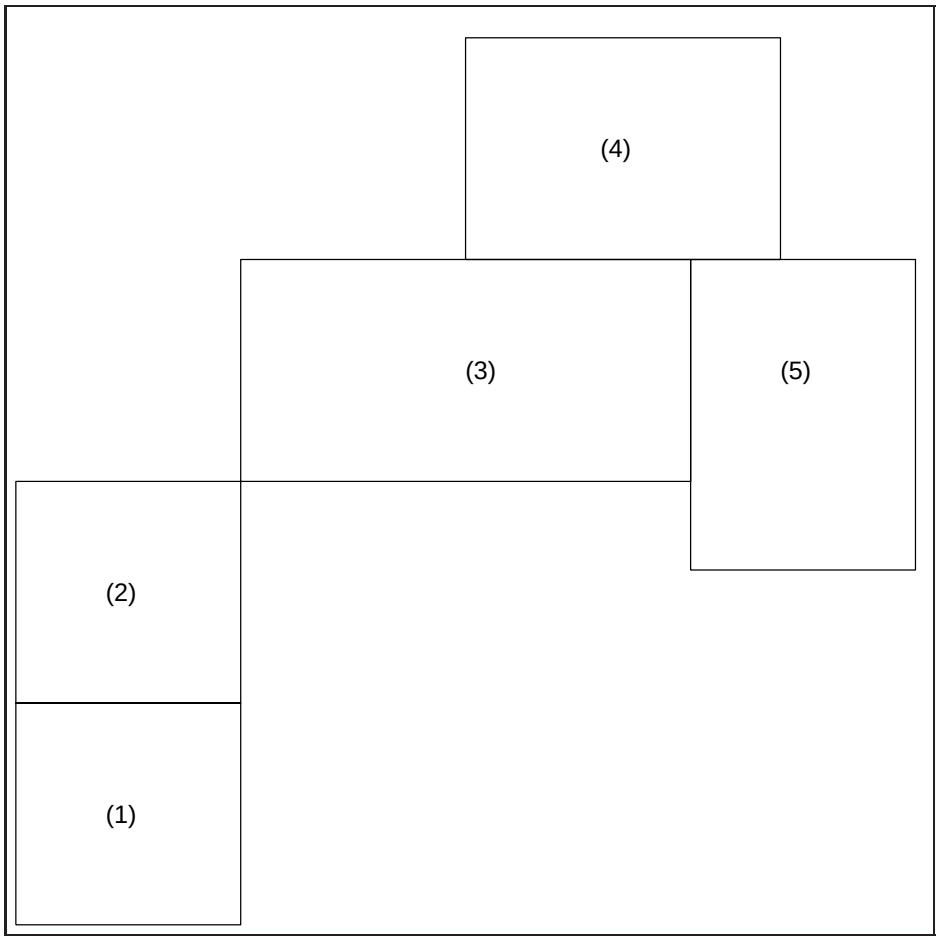

Figure 1.3: An illustration of contiguity

all other row-elements equal to zero.

Rook contiguity: Define $W_{ij} = 1$ for regions that share a common side with the region of interest. For row 1, reflecting region 1's relations we would have $W_{12} = 1$ with all other row elements equal to zero. As another example, row 3 would record $W_{34} = 1, W_{35} = 1$ and all other row elements equal to zero.

Bishop contiguity: Define $W_{ij} = 1$ for entities that share a common vertex with the region of interest. For region 2 we would have $W_{23} = 1$ and all other row elements equal to zero.

Double linear contiguity: For two entities to the immediate right or left of the region of interest, define $W_{ij} = 1$. This definition would produce the same results as linear contiguity for the regions in Figure 1.3.

Double rook contiguity: For two entities to the right, left, north and south of the region of interest define $W_{ij} = 1$. This would result in the same matrix W as rook contiguity for the regions shown in Figure 1.3.

Queen contiguity: For entities that share a common side or vertex with the region of interest define $W_{ij} = 1$. For region 3 we would have: $W_{32} = 1, W_{34} = 1, W_{35} = 1$ and all other row elements zero.

Believe it or not, there are even more ways one could proceed. For a good discussion of these issues, see Appendix 1 of Kelejian and Robinson (1995). Note also that the double linear and double rook definitions are sometimes referred to as "second order" contiguity, whereas the other definitions are termed "first order". More elaborate definitions sometimes rely on the distance of shared borders. This might impact whether we considered regions (4) and (5) in Figure 1.3 as contiguous or not. They have a common border, but it is very short. Note that in the case of a vertex, the rook definition rules out a contiguity relation, whereas the bishop and queen definitions would record a relationship.

The guiding principle in selecting a definition should be the nature of the problem being modeled, and perhaps additional non-sample information that is available. For example, suppose that a major highway connecting regions (2) and (3) existed and we knew that region (2) was a "bedroom community" for persons who work in region (3). Given this non-sample information, we would not want to rely on the rook definition that would rule out a contiguity relationship, as there is quite reasonably a large amount of spatial interaction between these two regions.

We will use the rook definition to define a first-order contiguity matrix for the five regions in Figure 1.3 as a concrete illustration. This is a definition that is often used in applied work. Perhaps the motivation for this is that we simply need to locate all regions on the map that have common borders with some positive length.

The matrix W reflecting first-order rook's contiguity relations for the five regions in Figure 1.3 is:

$$W = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \tag{1.3}$$

Note that the matrix W is symmetric as indicated above, and by convention the matrix always has zeros on the main diagonal. A transformation often used in applied work is to convert the matrix W to have row-sums of unity. This is referred to as a "standardized first-order" contiguity matrix, which we denote as C:

$$C = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1/2 & 1/2 & 0 \end{pmatrix} \tag{1.4}$$

The motivation for the standardization can be seen by considering what happens if we use matrix multiplication of C and a vector of observations on some variable associated with the five regions which we label y. This matrix product $y^\star = Cy$ represents a new variable equal to the mean of observations from contiguous regions:

$$\begin{aligned} \begin{pmatrix} y_1^\star \\ y_2^\star \\ y_3^\star \\ y_4^\star \\ y_5^\star \end{pmatrix} &= \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0.5 & 0.5 & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} \\ \begin{pmatrix} y_1^\star \\ y_2^\star \\ y_3^\star \\ y_4^\star \\ y_5^\star \end{pmatrix} &= \begin{pmatrix} y_2 \\ y_1 \\ 1/2y_4 + 1/2y_5 \\ 1/2y_3 + 1/2y_5 \\ 1/2y_3 + 1/2y_4 \end{pmatrix} \end{aligned} \tag{1.5}$$

This is one way of quantifying the notion that $y_i = f(y_j), j \neq i$, expressed in (1.1). Consider now a linear relationship that uses the variable $y^\star$ we constructed in (1.5) as an explanatory variable in a linear regression relationship to explain variation in y across the spatial sample of observations.

$$y = \rho Cy + \varepsilon \tag{1.6}$$

Where ρ represents a regression parameter to be estimated and ε denotes the stochastic disturbance in the relationship. The parameter ρ would reflect the spatial dependence inherent in our sample data, measuring the average influence of neighboring or contiguous observations on observations in the vector y. If we posit spatial dependence between the individual observations in the data sample y, some part of the total variation in y across the spatial sample would be explained by each observation's dependence on its neighbors. The parameter ρ would reflect this in the typical sense of regression. In addition, we could calculate the proportion of the total variation in y that is explained by spatial dependence. This would be represented by $\hat{\rho}Cy$, where $\hat{\rho}$ represents the estimated value of ρ.

We will examine spatial econometric models that rely on this type of formulation in great detail in Chapter 2, where we set forth maximum likelihood estimation procedures for a taxonomy of these models known as spatial autoregressive models.

One point to note is that traditional explanatory variables of the type encountered in regression can be added to the model in (1.6). We can represent these with the traditional matrix notation: $X\beta$, allowing us to modify (1.6) to take the form shown in (1.7).

$$y = \rho Cy + X\beta + \varepsilon \tag{1.7}$$

As an illustration, consider the following example which is intended to serve as a preview of material covered in the next two chapters. We provide a set of regression estimates based on maximum likelihood procedures for a spatial data set consisting of 49 neighborhoods in Columbus, Ohio set forth in Anselin (1988). The data set consists of observations on three variables: *neighborhood crime incidents, household income*, and *house values* for all 49 neighborhoods. The model uses the *income* and *house values* to explain variation in *neighborhood crime incidents*. That is, $y =$ *neighborhood crime*, $X =$(*a constant, household income, house values*). The estimates are shown below, printed in the usual regression format with associated statistics for precision of the estimates, fit of the model and an estimate of the disturbance variance, $\hat{\sigma}^2_{\varepsilon}$.

```
Spatial autoregressive Model Estimates
Dependent Variable =      Crime
R-squared      =    0.6518
Rbar-squared   =    0.6366
```

```
sigma^2        =   95.5032
log-likelihood =       -165.41269
Nobs, Nvars    =     49,     3
***************************************************************
Variable         Coefficient      t-statistic    t-probability
constant           45.056251         6.231261         0.000000
income             -1.030641        -3.373768         0.001534
house value        -0.265970        -3.004945         0.004331
rho                 0.431381         3.625340         0.000732
```

For this example, we can calculate the proportion of total variation explained by spatial dependence with a comparison of the fit measured by $\bar{R}^2$ from this model to the fit of a least-squares model that excludes the spatial dependence variable Cy. The least-squares regression for comparison is shown below:

```
Ordinary Least-squares Estimates
Dependent Variable =      Crime
R-squared      =    0.5521
Rbar-squared   =    0.5327
sigma^2        =  130.8386
Durbin-Watson  =    1.1934
Nobs, Nvars    =     49,     3
***************************************************************
Variable         Coefficient      t-statistic    t-probability
constant           68.609759        14.484270         0.000000
income             -1.596072        -4.776038         0.000019
house value        -0.274079        -2.655006         0.010858
```

We see that around 10 percent of the variation in the crime incidents is explained by spatial dependence, because the $\bar{R}^2$ is roughly 0.63 in the model that takes spatial dependence into account and 0.53 in the least-squares model that ignores this aspect of the spatial data sample. Note also that the $t-$statistic on the parameter for the spatial dependence variable Cy is 3.62, indicating that this explanatory variable has a coefficient estimate that is significantly different from zero. We will pursue more examples in Chapters 2 and 3, with this example provided as a concrete demonstration of some of the ideas we have discussed.

1.4.2 Quantifying spatial position

Associating location in space with observations is essential to modeling relationships that exhibit spatial heterogeneity. Recall this means there is variation in the relationship being modeled over space. We illustrate two

approaches to using location that allow locally linear regressions to be fit over sub-regions of space. These form the basis for models we will discuss in Chapter 4.

Casetti (1972, 1992) introduced our first approach that involves a method he labels "spatial expansion". The model is shown in (1.8), where y denotes an nx1 dependent variable vector associated with spatial observations and X is an $nxnk$ matrix consisting of terms x_i representing kx1 explanatory variable vectors, as shown in (1.9). The locational information is recorded in the matrix Z which has elements $Z_{xi}, Z_{yi}, i = 1, \ldots, n$, that represent latitude and longitude coordinates of each observation as shown in (1.9).

The model posits that the parameters vary as a function of the latitude and longitude coordinates. The only parameters that need be estimated are the parameters in β_0 that we denote, β_x, β_y. These represent a set of $2k$ parameters. Recall our discussion about spatial heterogeneity and the need to utilize a parsimonious specification for variation over space. This represents one approach to this type of specification.

We note that the parameter vector β in (1.8) represents an nkx1 matrix in this model that contains parameter estimates for all k explanatory variables at every observation. The parameter vector β_0 contains the $2k$ parameters to be estimated.

$$\begin{aligned} y &= X\beta + \varepsilon \\ \beta &= ZJ\beta_0 \end{aligned} \tag{1.8}$$

Where:

$$\begin{aligned} y &= \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad X = \begin{pmatrix} x_1' & 0 & \ldots & 0 \\ 0 & x_2' & & \\ \vdots & & \ddots & \\ 0 & & & x_n' \end{pmatrix} \quad \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix} \\ Z &= \begin{pmatrix} Z_{x1} \otimes I_k & Z_{y1} \otimes I_k & 0 & \ldots \\ 0 & \ddots & \ddots & \\ \vdots & & Z_{xn} \otimes I_k & Z_{yn} \otimes I_k \end{pmatrix} J = \begin{pmatrix} I_k & 0 \\ 0 & I_k \\ \vdots & \\ 0 & I_k \end{pmatrix} \\ \beta_0 &= \begin{pmatrix} \beta_x \\ \beta_y \end{pmatrix} \end{aligned} \tag{1.9}$$

This model can be estimated using least-squares to produce estimates of the $2k$ parameters β_x, β_y. Given these estimates, the remaining estimates for individual points in space can be derived using the second equation in (1.8). This process is referred to as the "expansion process". To see this, substitute the second equation in (1.8) into the first, producing:

$$y = XZJ\beta_0 + \varepsilon \tag{1.10}$$

Here it is clear that X, Z and J represent available information or data observations and only β_0 represent parameters in the model that need be estimated.

The model would capture spatial heterogeneity by allowing variation in the underlying relationship such that clusters of nearby or neighboring observations measured by latitude-longitude coordinates take on similar parameter values. As the location varies, the regression relationship changes to accommodate a locally linear fit through clusters of observations in close proximity to one another.

To provide a preview of this model which is discussed in Chapter 4, Figure 1.4 presents the spatially expanded parameter estimates for the Columbus neighborhood crime data set. The observations have been ordered by distance from the central city, so as we move to the far right along the x-axis, we are examining parameter values associated with more suburban locations. The estimates are based on a slightly altered version of the spatial expansion model that relies on distance from a central city neighborhood rather than the latitude-longitude location of observations defined in (1.9). (We will discuss this variant on the spatial expansion model more fully in Chapter 4.)

We see from the figure that neighborhood crime is negatively influenced by both household income and housing values. As we move away from the central city neighborhoods, this negative influence increases.

Another approach to modeling variation over space is based on locally weighted regressions to produce estimates for every point in space by using a sub-sample of data information from nearby observations. McMillen (1996) and Brundson, Fotheringham and Charlton (1996) introduce this type of approach. It has been labeled "geographically weighted regression", (GWR) by Brundson, Fotheringham and Charlton (1996). Let y denote an nx1 vector of dependent variable observations collected at n points in space, X an nxk matrix of explanatory variables, and ε an nx1 vector of normally distributed, constant variance disturbances. Letting W_i represent an nxn diagonal matrix containing distance-based weights for observation i that

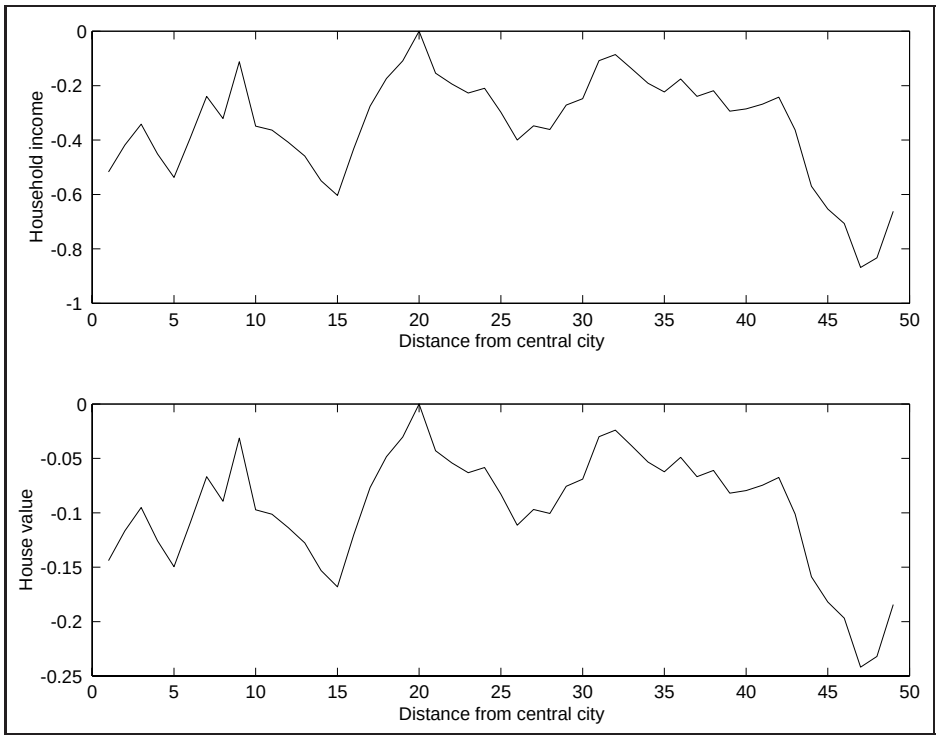

Figure 1.4: Distance expansion estimates

reflects the distance between observation i and all other observations, we can write the GWR model as:

$$W_i y = W_i X \beta_i + \varepsilon_i \tag{1.11}$$

The subscript i on β_i indicates that this kx1 parameter vector is associated with observation i. The GWR model produces n such vectors of parameter estimates, one for each observation. These estimates are produced using:

$$\hat{\beta}_i = (X' W_i^2 X)^{-1} (X' W_i^2 y) \tag{1.12}$$

One confusing aspect of this notation is that $W_i y$ denotes an n-vector of distance-weighted observations used to produce estimates for observation i. The notation is confusing because we usually rely on subscripts to index scalar magnitudes representing individual elements of a vector. Note

also, that W_iX represents a distance-weighted data matrix, not a single observation and ε_i represents a n-vector. The precise nature of the distance weighting is taken up in Chapter 4.

As an applied illustration, we provide a graphical presentation in Figure 1.5 of the estimates produced by the GWR method sorted by distance from the central city, making these comparable to those already presented for the Casetti distance expansion model. The two sets of estimates are quite different, raising the question of which approach provides a better set of inferences regarding the relationship between the neighborhood crime rates and the explanatory variables in the model. This topic will be taken up when we discuss alternative modeling approaches to deal with spatial dependence and heterogeneity.

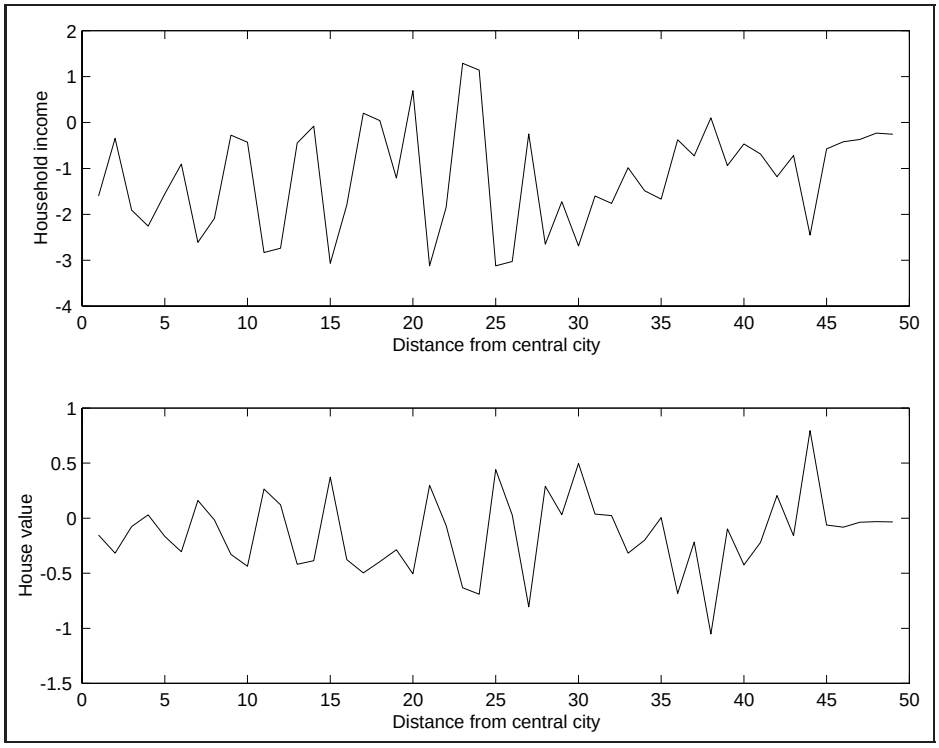

Figure 1.5: GWR estimates

It may have occurred to the reader that a homogeneous model fit to a spatial data sample that exhibits heterogeneity will produce residuals that exhibit spatial dependence. The residuals or errors made by the homo-

geneous model should reflect unexplained variation attributable to heterogeneity in the underlying relationship over space. Spatial clustering of the residuals would occur with positive and negative residuals appearing in distinct regions and patterns on the map. This of course was our motivation and illustration of spatial dependence as illustrated in Figure 1.2. You might infer correctly that spatial heterogeneity and dependence are often related in the context of modeling. An inappropriate model that fails to capture spatial heterogeneity will result in residuals that exhibit spatial dependence. This is another topic we discuss in the following chapters of this text.

1.5 The MATLAB spatial econometrics library

As indicated in the preface, all of the spatial econometric methods discussed in this text have been implemented using the MATLAB software from MathWorks Inc. Toolboxes are the name given by the MathWorks to related sets of MATLAB functions aimed at solving a particular class of problems. Toolboxes of functions useful in signal processing, optimization, statistics, finance and a host of other areas are available from the MathWorks as add-ons to the standard MATLAB distribution. We will reserve the term *Econometrics Toolbox* to refer to my larger collection of econometric functions available in the public domain at www.econ.utoledo.edu. The spatial econometrics library represents a smaller part of this larger collection of software functions for econometric analysis. I have used the term library to denote subsets of functions aimed at various categories of estimation methods. The *Econometrics Toolbox* contains libraries for econometric regression analysis, time-series and vector autoregressive modeling, optimization functions to solve general maximum likelihood estimation problems, Bayesian Gibbs sampling diagnostics, error correction testing and estimation methods, simultaneous equation models and a collection of utility functions that I designate as the *utility function library*. Taken together, these constitute the *Econometrics Toolbox* that is described in a 350 page manual available at the Web site listed above.

The spatial econometrics library functions rely on some of the utility functions and are implemented using a general design that provides a common user-interface for the entire toolbox of econometric estimation functions. In Chapter 2 we will use MATLAB functions to carry out spatial econometric estimation methods. Here, we discuss the general design that is used to implement all of the spatial econometric estimation functions. Having some feel for the way in which these functions work and communi-

cate with other functions in the *Econometric Toolbox* should allow you to more effectively use these functions to solve spatial econometric estimation problems.

The entire *Econometrics Toolbox* has been included in the internet-based materials provided here, as well as an online HTML interface to examine the functions available along with their documentation. All functions have accompanying demonstration files that illustrate the typical use of the functions with sample data. These demonstration files can be viewed using the online HTML interface. We have also provided demonstration files for all of the estimation functions in the spatial econometrics library that can be viewed online along with their documentation. Examples are provided in this text and the program files along with the datasets have been included in the Web-based module.

In designing a spatial econometric library of functions, we need to think about organizing our functions to present a consistent user-interface that packages all of our MATLAB functions in a unified way. The advent of 'structures' in MATLAB version 5 allows us to create a host of alternative spatial econometric functions that all return 'results structures'.

A structure in MATLAB allows the programmer to create a variable containing what MATLAB calls 'fields' that can be accessed by referencing the structure name plus a period and the field name. For example, suppose we have a MATLAB function to perform ordinary least-squares estimation named **ols** that returns a structure. The user can call the function with input arguments (a dependent variable vector y and explanatory variables matrix x) and provide a variable name for the structure that the **ols** function will return using:

```
result = ols(y,x);
```

The structure variable 'result' returned by our **ols** function might have fields named 'rsqr', 'tstat', 'beta', etc. These fields might contain the R-squared statistic, $t-$statistics and the least-squares estimates $\hat{\beta}$. One virtue of using the structure to return regression results is that the user can access individual fields of interest as follows:

```
bhat = result.beta;
disp('The R-squared is:');
result.rsqr
disp('The 2nd t-statistic is:');
result.tstat(2,1)
```

There is nothing sacred about the name 'result' used for the returned structure in the above example, we could have used:

```
bill_clinton = ols(y,x);
result2      = ols(y,x);
restricted   = ols(y,x);
unrestricted = ols(y,x);
```

That is, the name of the structure to which the **ols** function returns its information is assigned by the user when calling the function.

To examine the nature of the structure in the variable 'result', we can simply type the structure name without a semi-colon and MATLAB will present information about the structure variable as follows:

```
result =
     meth: 'ols'
        y: [100x1 double]
     nobs: 100.00
     nvar: 3.00
     beta: [  3x1 double]
     yhat: [100x1 double]
    resid: [100x1 double]
     sige: 1.01
    tstat: [  3x1 double]
     rsqr: 0.74
     rbar: 0.73
       dw: 1.89
```

Each field of the structure is indicated, and for scalar components the value of the field is displayed. In the example above, 'nobs', 'nvar', 'sige', 'rsqr', 'rbar', and 'dw' are scalar fields, so there values are displayed. Matrix or vector fields are not displayed, but the size and type of the matrix or vector field is indicated. Scalar string arguments are displayed as illustrated by the 'meth' field which contains the string 'ols' indicating the regression method that was used to produce the structure. The contents of vector or matrix strings would not be displayed, just their size and type. Matrix and vector fields of the structure can be displayed or accessed using the MATLAB conventions of typing the matrix or vector name without a semi-colon. For example,

```
result.resid
result.y
```

would display the residual vector and the dependent variable vector y in the MATLAB command window.

Another virtue of using 'structures' to return results from our regression functions is that we can pass these structures to another related function

that would print or plot the regression results. These related functions can query the structure they receive and intelligently decipher the 'meth' field to determine what type of regression results are being printed or plotted. For example, we could have a function **prt** that prints regression results and another **plt** that plots actual versus fitted and/or residuals. Both these functions take a regression structure as input arguments. Example 1.1 provides a concrete illustration of these ideas.

```
% ----- Example 1.1 Demonstrate regression using the ols() function
load y.data;
load x.data;
result = ols(y,x);
prt(result);
plt(result);
```

The example assumes the existence of functions **ols, prt, plt** and data matrices y, x in files 'y.data' and 'x.data'. Given these, we carry out a regression, print results and plot the actual versus predicted as well as residuals with the MATLAB code shown in example 1.1. We will discuss the **prt** and **plt** functions in Section 1.5.2.

1.5.1 Estimation functions

Now to put these ideas into practice, consider implementing an **ols** function. The function code would be stored in a file 'ols.m' whose first line is:

```
function results=ols(y,x)
```

The keyword 'function' instructs MATLAB that the code in the file 'ols.m' represents a callable MATLAB function.

The help portion of the MATLAB 'ols' function is presented below and follows immediately after the first line as shown. All lines containing the MATLAB comment symbol '%' will be displayed in the MATLAB command window when the user types 'help ols'.

```
function results=ols(y,x)
% PURPOSE: least-squares regression
%---------------------------------------------------
% USAGE: results = ols(y,x)
% where: y = dependent variable vector (nobs x 1)
%        x = independent variables matrix (nobs x nvar)
%---------------------------------------------------
% RETURNS: a structure
```

```
%        results.meth  = 'ols'
%        results.beta  = bhat
%        results.tstat = t-stats
%        results.yhat  = yhat
%        results.resid = residuals
%        results.sige  = e'*e/(n-k)
%        results.rsqr  = rsquared
%        results.rbar  = rbar-squared
%        results.dw    = Durbin-Watson Statistic
%        results.nobs  = nobs
%        results.nvar  = nvars
%        results.y     = y data vector
% --------------------------------------------------
% SEE ALSO: prt(results), plt(results)
%--------------------------------------------------
```

All functions in the spatial econometrics library present a unified documentation format for the MATLAB 'help' command by adhering to the convention of sections entitled, 'PURPOSE', 'USAGE', 'RETURNS', 'SEE ALSO', and perhaps a 'REFERENCES' section, delineated by dashed lines.

The 'USAGE' section describes how the function is used, with each input argument enumerated along with any default values. A 'RETURNS' section portrays the structure that is returned by the function and each of its fields. To keep the help information uncluttered, we assume some knowledge on the part of the user. For example, we assume the user realizes that the '.residuals' field would be an (nobs x 1) vector and the '.beta' field would consist of an (nvar x 1) vector.

The 'SEE ALSO' section points the user to related routines that may be useful. In the case of our **ols** function, the user might what to rely on the printing or plotting routines **prt** and **plt**, so these are indicated. The 'REFERENCES' section would be used to provide a literature reference (for the case of our more exotic spatial estimation procedures) where the user could read about the details of the estimation methodology.

As an illustration of the consistency in documentation, consider the function **sar** that provides estimates for the spatial autoregressive model that we presented in Section 1.4.1. The documentation for this function is shown below:

```
 PURPOSE: computes spatial autoregressive model estimates
          y = p*W*y + X*b + e, using sparse matrix algorithms
---------------------------------------------------
 USAGE: results = sar(y,x,W,rmin,rmax,convg,maxit)
 where:  y = dependent variable vector
         x = explanatory variables matrix
```

```
         W = standardized contiguity matrix
      rmin = (optional) minimum value of rho to use in search
      rmax = (optional) maximum value of rho to use in search
     convg = (optional) convergence criterion (default = 1e-8)
     maxit = (optional) maximum # of iterations (default = 500)
-------------------------------------------------
 RETURNS: a structure
        results.meth  = 'sar'
        results.beta  = bhat
        results.rho   = rho
        results.tstat = asymp t-stat (last entry is rho)
        results.yhat  = yhat
        results.resid = residuals
        results.sige  = sige = (y-p*W*y-x*b)'*(y-p*W*y-x*b)/n
        results.rsqr  = rsquared
        results.rbar  = rbar-squared
        results.lik   = -log likelihood
        results.nobs  = # of observations
        results.nvar  = # of explanatory variables in x
        results.y     = y data vector
        results.iter   = # of iterations taken
        results.romax  = 1/max eigenvalue of W (or rmax if input)
        results.romin  = 1/min eigenvalue of W (or rmin if input)
 -------------------------------------------------
 SEE ALSO: prt(results), sac, sem, far
-------------------------------------------------
REFERENCES: Anselin (1988), pages 180-182.
-------------------------------------------------
```

The actual execution code to produce least-squares or spatial autoregressive parameter estimates would follow the documentation in the file discussed above. We do not discuss programming of the spatial econometric functions in the text, but you can of course examine all of the functions to see how they work. The manual for the *Econometrics Toolbox* provides a great deal of discussion of programming in MATLAB and examples of how to add new functions to the toolbox or change existing functions in the toolbox.

1.5.2 Using the results structure

To illustrate the use of the 'results' structure returned by our **ols** function, consider the associated function **plt_reg** which plots actual versus predicted values along with the residuals. The results structure contains everything needed by the **plt_reg** function to carry out its task. Earlier, we referred to functions **plt** and **prt** rather than **plt_reg**, but **prt** and **plt** are "wrapper"

functions that call the functions **prt_reg** and **plt_reg** where the real work of printing and plotting regression results is carried out. The motivation for taking this approach is that separate smaller functions can be devised to print and plot results from all of the spatial econometric procedures, facilitating development. The wrapper functions eliminate the need for the user to learn the names of different printing and plotting functions associated with each group of spatial econometric procedures — all results structures can be printed and plotted by simply invoking the **prt** and **plt** functions.

Documentation for the **plt** function which plots results from all spatial econometrics functions as well as the *Econometrics Toolbox* is shown below. This function is a the wrapper function that calls an appropriate plotting function, **plt_spat** based on the econometric method identified in the results structure 'meth' field argument.

```
 PURPOSE: Plots results structures returned by most functions
          by calling the appropriate plotting function
-------------------------------------------------
 USAGE: plt(results,vnames)
 Where: results = a structure returned by an econometric function
        vnames  = an optional vector of variable names
        e.g. vnames = vnames = strvcat('y','const','x1','x2');
 -------------------------------------------------
 NOTES: this is simply a wrapper function that calls another function
 -------------------------------------------------
 RETURNS: nothing, just plots the results
 -------------------------------------------------
 SEE ALSO: prt()
-------------------------------------------------
```

A decision was made not to place the 'pause' command in the **plt** function, but rather let the user place this statement in the calling program or function. An implication of this is that the user controls viewing regression plots in 'for loops' or in the case of multiple invocations of the **plt** function. For example, only the second 'plot' will be shown in the following code.

```
result1 = sar(y,x1,W);
plt(result1);
result2 = sar(y,x2,W);
plt(result2);
```

If the user wishes to see the plots associated with the first spatial autoregression, the code would need to be modified as follows:

```
result1 = sar(y,x1,W);
```

```
plt(result1);
pause;
result2 = sar(y,x2,W);
plt(result2);
```

The 'pause' statement would force a plot of the results from the first spatial autoregression and wait for the user to strike any key before proceeding with the second regression and accompanying plot of these results.

A more detailed example of using the results structure is the **prt** function which produces printed output from all of the functions in the spatial econometrics library. The printout of estimation results is similar to that provided by most statistical packages.

The **prt** function allows the user an option of providing a vector of fixed width variable name strings that will be used when printing the regression coefficients. These can be created using the MATLAB **strvcat** function that produces a vertical concatenated list of strings with fixed width equal to the longest string in the list. We can also print results to an indicated file rather than the MATLAB command window. Three alternative invocations of the **prt** function illustrating these options for usage are shown below:

```
vnames = strvcat('crime','const','income','house value');
res = sar(y,x,W);
prt(res);                    % print with generic variable names
prt(res,vnames);             % print with user-supplied variable names
fid = fopen('sar.out','wr'); % open a file for printing
prt(res,vnames,fid);         % print results to file 'sar.out'
```

The first use of **prt** produces a printout of results to the MATLAB command window that uses 'generic' variable names:

```
Spatial autoregressive Model Estimates
R-squared      =    0.6518
Rbar-squared   =    0.6366
sigma^2        =   95.5033
Nobs, Nvars    =     49,     3
log-likelihood =       -165.41269
# of iterations =     17
min and max rho =   -1.5362,   1.0000
*************************************************************
Variable        Coefficient      t-statistic    t-probability
variable 1        45.056480         6.231281         0.000000
variable 2        -1.030647        -3.373784         0.001513
variable 3        -0.265970        -3.004944         0.004290
rho                0.431377         3.625292         0.000720
```

The second use of **prt** uses the user-supplied variable names. The MATLAB function **strvcat** carries out a vertical concatenation of strings and pads the shorter strings in the 'vnames' vector to have a fixed width based on the longer strings. A fixed width string containing the variable names is required by the **prt** function. Note that we could have used:

```
vnames = ['crime      ',
          'const      ',
          'income     ',
          'house value'];
```

but, this takes up more space and is slightly less convenient as we have to provide the padding of strings ourselves. Using the 'vnames' input in the **prt** function would result in the following printed to the MATLAB command window.

```
Spatial autoregressive Model Estimates
Dependent Variable =      crime
R-squared       =    0.6518
Rbar-squared    =    0.6366
sigma^2         =   95.5033
Nobs, Nvars     =     49,     3
log-likelihood  =       -165.41269
# of iterations =     12
min and max rho =   -1.5362,   1.0000
***************************************************************
Variable         Coefficient      t-statistic    t-probability
const              45.056481         6.231281         0.000000
income             -1.030647        -3.373784         0.001513
house value        -0.265970        -3.004944         0.004290
rho                 0.431377         3.625292         0.000720
```

The third case specifies an output file opened with the command:

```
fid = fopen('sar.out','wr');
```

The file 'sar.out' would contain output identical to that from the second use of **prt**. It is the user's responsibility to close the file that was opened using the MATLAB command:

```
fclose(fid);
```

In the following chapters that present various spatial estimation methods we will provide the documentation but not the details concerning implementation of the estimation procedures in MATLAB. A function has been

devised and incorporated in the spatial econometrics library for each of the estimation procedures that we discuss and illustrate. These functions carry out estimation and provide printed as well as graphical presentation of the results using the design framework that was set forth in this section.

1.6 Chapter Summary

This chapter introduced two main features of spatial data sets, spatial dependence and spatial heterogeneity. Spatial dependence refers to the fact that sample data observations exhibit correlation with reference to points or location in space. We often observe spatial clustering of sample data observations with respect to map regions. An intuitive motivation for this type of result is the existence of spatial hierarchical relationships, spatial spillovers and other types of spatial interactivity studied in regional science.

Spatial heterogeneity refers to the fact that underlying relationships we wish to study may vary systematically over space. This creates problems for regression and other econometric methods that do not accommodate spatial variation in the relationships being modeled. A host of methods have arisen in spatial econometrics that allow the estimated relationship to vary systematically over space.

A large part of the chapter was devoted to introducing how locational information regarding sample data observations is formally incorporated in spatial econometric models. After introducing the concept of a spatial contiguity matrix, we provided a preview of the spatial autoregressive model that relies on the contiguity concept. Chapters 2 and 3 cover this spatial econometric method in detail.

In addition to spatial contiguity, other spatial econometric methods rely on the latitude-longitude information available for spatial data samples to allow variation in the relationship being studied over space. Two approaches to this were introduced, the spatial expansion model and geographically weighted regression, which are the subject of Chapter 4.

A preview that provided an applied illustration of the methods that will be detailed in Chapters 2 through 4 was provided using a spatial data set from Anselin (1988) on crime incidents in 49 neighborhoods in Columbus, Ohio.

Finally, a software design for implementing the spatial econometric estimation methods discussed in this text was set forth. Our estimation methods will be implemented using MATLAB software from the MathWorks Inc. A design based on MATLAB structure variables was set forth. This approach

to developing a set of spatial econometric estimation functions can provide a consistent user-interface for the function documentation and help information as well as encapsulation of the estimation results in a MATLAB structure variable. This construct can be accessed by related functions to provide printed and graphical presentation of the estimation results.

1.7 References

Amemiya, T. 1985. *Advanced Econometrics*, (Cambridge, MA: Harvard University Press).

Anselin, L. 1988. *Spatial Econometrics: Methods and Models*, (Dorddrecht: Kluwer Academic Publishers).

Brundson, C. A. S. Fotheringham, and M. Charlton. 1996. "Geographically weighted regression: a method for exploring spatial nonstationarity," *Geographical Analysis*, Vol. 28, pp. 281-298.

Casetti, E., 1972. "Generating Models by the Expansion Method: Applications to Geographic Research", *Geographical Analysis*, Vol. 4, pp. 81-91.

Casetti, E. 1992. "Bayesian Regression and the Expansion Method", *Geographical Analysis*, Vol. 24, pp. 58-74.

Chow, G. 1983. *Econometrics*, (New York: McGraw-Hill);

Dhrymes, P. 1981. *Distributed Lags: Problems of Estimation and Formulation*, (Amsterdam: North-Holland).

Fomby, T., R. Hill, and S. Johnson. 1984. *Advanced Econometric Methods*, (New York: Springer).

Intrilligator, M. 1978. *Econometric Models, Techniques, and Applications*, (Englewood Cliffs: Prentice-Hall).

Green, W. H. 1997. *Econometric Analysis, third edition*, (Upper Saddle River, N.J: Prentice Hall).

Kelejian, H. and W. Oates. 1989. *Introduction to Econometrics: Principles and Applications*, (New York: Harper and Row).

Kelejian, H. H. and D. P. Robinson. 1995. "Spatial Correlation: A suggested alternative to the autoregressive model", in *New Directions in Spatial Econometrics*, L. Anselin and R.J.G.M. Florax (eds.). (Berlin: Springer).

Kmenta, J. 1971. *Elements of Econometrics*, (New York: Macmillan).

Maddala, G.S. 1977. *Econometrics*, (New York: McGraw-Hill).

McMillen, D.P. 1996. "One hundred fifty years of land values in Chicago: a nonparameteric approach," *Journal of Urban Economics*, Vol. 40, pp. 100-124.

Pindyck, R. and D. Rubinfeld. 1981. *Econometric Models and Economic Forecasts*, (New York: McGraw-Hill).

Schmidt, P. 1976. *Econometrics*, (New York: Marcel Dekker).

Vinod, H. and A. Ullah. 1981. *Recent Advances in Regression Methods*, (New York: Marcel Dekker).

Chapter 2

Spatial autoregressive models

This chapter discusses in detail the spatial autoregressive models introduced in Chapter 1. A class of spatial autoregressive models have been introduced to model cross-sectional spatial data samples taking the form shown in (2.1) (Anselin, 1988).

$$\begin{aligned} y &= \rho W_1 y + X\beta + u \\ u &= \lambda W_2 u + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.1}$$

Where y contains an nx1 vector of cross-sectional dependent variables and X represents an nxk matrix of explanatory variables. W_1 and W_2 are known nxn spatial weight matrices, usually containing first-order contiguity relations or functions of distance. As explained in Section 1.4.1, a first-order contiguity matrix has zeros on the main diagonal, rows that contain zeros in positions associated with non-contiguous observational units and ones in positions reflecting neighboring units that are (first-order) contiguous based on one of the contiguity definitions.

From the general model in (2.1) we can derive special models by imposing restrictions. For example, setting $X = 0$ and $W_2 = 0$ produces a first-order spatial autoregressive model shown in (2.2).

$$\begin{aligned} y &= \rho W_1 y + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.2}$$

This model attempts to explain variation in y as a linear combination of contiguous or neighboring units with no other explanatory variables. The

model is termed a first order spatial autoregression because its represents a spatial analogy to the first order autoregressive model from time series analysis, $y_t = \rho y_{t-1} + \varepsilon_t$, where total reliance is on the past period observations to explain variation in y_t.

Setting $W_2 = 0$ produces a mixed regressive-spatial autoregressive model shown in (2.3). This model is analogous to the lagged dependent variable model in time series. Here we have additional explanatory variables in the matrix X that serve to explain variation in y over the spatial sample of observations.

$$\begin{aligned} y &= \rho W_1 y + X\beta + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.3}$$

Letting $W_1 = 0$ results in a regression model with spatial autocorrelation in the disturbances as shown in (2.4).

$$\begin{aligned} y &= X\beta + u \\ u &= \lambda W_2 u + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.4}$$

This chapter is organized into sections that discuss and illustrate each of these special cases of the spatial autoregressive model as well as the most general model form in (2.1). Section 2.1 deals with the first-order spatial autoregressive model presented in (2.2). The mixed regressive-spatial autoregressive model is taken up in Section 2.2. Section 2.3 takes up the regression model containing spatial autocorrelation in the disturbances and illustrates various tests for spatial dependence using regression residuals. The most general model is the focus of Section 2.4. Applied illustrations of all the models are provided using a variety of spatial data sets. Spatial econometrics library functions that utilize MATLAB sparse matrix algorithms allow us to estimate models with over 3,000 observations in around 100 seconds on an inexpensive desktop computer.

2.1 The first-order spatial AR model

This model is seldom used in applied work, but it serves to motivate some of the ideas that we draw on in later sections of the chapter. The model which we label FAR, takes the form:

$$\begin{aligned} y &= \rho W y + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.5}$$

Where the spatial contiguity matrix W has been standardized to have row sums of unity and the variable vector y is expressed in deviations from the means form to eliminate the constant term in the model.

To illustrate the problem with least-squares estimation of spatial autoregressive models, consider applying least-squares to the model in (2.5) which would produce an estimate for the single parameter ρ in the model:

$$\hat{\rho} = (y'W'Wy)^{-1}y'W'y \tag{2.6}$$

Can we show that this estimate is unbiased? If not, is it consistent? Taking the same approach as in least-squares, we substitute the expression for y from the model statement and attempt to show that $E(\hat{\rho}) = \rho$ to prove unbiasedness.

$$\begin{aligned} E(\hat{\rho}) &= (y'W'Wy)^{-1}y'W'(\rho Wy + \varepsilon) \\ &= \rho + (y'W'Wy)^{-1}y'W'\varepsilon \end{aligned} \tag{2.7}$$

Note that the least-squares estimate is biased, since we cannot show that $E(\hat{\rho}) = \rho$. The usual argument that the explanatory variables matrix X in least-squares is fixed in repeated sampling allows one to pass the expectation operator over terms like $(y'W'Wy)^{-1}y'W'$ and argue that $E(\varepsilon) = 0$, eliminating the bias term. Here however, because of spatial dependence, we cannot make the case that Wy is fixed in repeated sampling. This also rules out making a case for consistency of the least-squares estimate of ρ, because the probability limit (plim) for the term $y'W'\varepsilon$ is not zero. In fact, Anselin (1988) establishes that:

$$\text{plim}N^{-1}(y'W'\varepsilon) = \text{plim}N^{-1}\varepsilon'W(I - \rho W)^{-1}\varepsilon \tag{2.8}$$

This is equal to zero only in the trivial case where ρ equals zero and we have no spatial dependence in the data sample.

Given that least-squares will produce biased and inconsistent estimates of the spatial autoregressive parameter ρ in this model, how do we proceed to estimate ρ? The maximum likelihood estimator for ρ requires that we find a value of ρ that maximizes the likelihood function shown in (2.9).

$$L(y|\rho,\sigma^2) = \frac{1}{2\pi\sigma^{2(n/2)}}|I_n - \rho W| \exp\{-\frac{1}{2\sigma^2}(y-\rho Wy)'(y-\rho Wy)\} \quad (2.9)$$

In order to simplify the maximization problem, we obtain a concentrated log likelihood function based on eliminating the parameter σ^2 for the variance of the disturbances. This is accomplished by substituting $\hat{\sigma}^2 = (1/n)(y-\rho Wy)'(y-\rho Wy)$ in the likelihood (2.9) and taking logs which yields:

$$\text{Ln}(L) \propto -\frac{n}{2}\ln(y-\rho Wy)'(y-\rho Wy) + \ln|I_n - \rho W| \quad (2.10)$$

This expression can be maximized with respect to ρ using a simplex univariate optimization routine. The estimate for the parameter σ^2 can be obtained using the value of ρ that maximizes the log-likelihood function (say, $\tilde{\rho}$) in: $\hat{\sigma}^2 = (1/n)(y-\tilde{\rho}Wy)'(y-\tilde{\rho}Wy)$. In the next section, we discuss a sparse matrix algorithm approach to evaluating this likelihood function that allows us to solve problems involving thousands of observations quickly with small amounts of computer memory.

Two implementation details arise with this approach to solving for maximum likelihood estimates. First, there is a constraint that we need to impose on the parameter ρ. This parameter can take on feasible values in the range (Anselin and Florax, 1994):

$$1/\lambda_{min} < \rho < 1/\lambda_{max}$$

where λ_{min} represents the minimum eigenvalue of the standardized spatial contiguity matrix W and λ_{max} denotes the largest eigenvalue of this matrix. This suggests that we need to constrain our optimization procedure search over values of ρ within this range.

The second implementation issue is that the numerical hessian matrix that would result from a gradient-based optimization procedure and provide estimates of dispersion for the parameters is not available with simplex optimization. We can overcome this problem in two ways. For problems involving a small number of observations, we can use our knowledge of the theoretical information matrix to produce estimates of dispersion. An asymptotic variance matrix based on the Fisher information matrix shown below for the parameters $\theta = (\rho, \sigma^2)$ can be used to provide measures of dispersion for the estimates of ρ and σ^2. (see Anselin, 1980 page 50):

$$[I(\theta)]^{-1} = -E[\frac{\partial^2 L}{\partial\theta\partial\theta'}]^{-1} \quad (2.11)$$

This approach is computationally impossible when dealing with large scale problems involving thousands of observations. In these cases we can evaluate the numerical hessian matrix using the maximum likelihood estimates of ρ and σ^2 as well as our sparse matrix function to compute the likelihood. We will demonstrate results from using both of these approaches in the next section.

2.1.1 The far() function

Building on the software design set forth in section 1.5 for our spatial econometrics function library, we have implemented a function **far** to produce maximum likelihood estimates for the first-order spatial autoregressive model. We rely on on the sparse matrix functionality of MATLAB so large-scale problems can be solved using a minimum of time and computer RAM memory. We demonstrate this function in action on a data set involving 3,107 U.S. counties.

Estimating the FAR model requires that we find eigenvalues for the large n by n matrix W, as well as the determinant of the related n by n matrix $(I_n - \rho W)$. In addition, matrix multiplications involving W and $(I_n - \rho W)$ are required to compute the information matrix used to produce estimates of dispersion.

We constructed a function **far** that can produce estimates for the first-order spatial autoregressive model in a case involving 3,107 observations in 95 seconds on a moderately fast inexpensive desktop computer. The MATLAB algorithms for dealing with sparse matrices make it ideally suited for spatial modeling because spatial weight matrices are almost always sparse.

Another issue we need to address is computing measures of dispersion for the estimates ρ and σ^2 in large estimation problems. As already noted, we cannot rely on the information matrix approach because this involves matrix operations on very large matrices. An approach that we take to produce measures of dispersion is to numerically evaluate the hessian matrix using the maximum likelihood estimates of ρ and σ^2. The approach basically produces a numerical approximation to the expression in (2.11). A key to using this approach is the ability to evaluate the log likelihood function using the sparse algorithms to handle large matrices.

It should be noted that Pace and Barry (1997) when confronted with the task of providing measures of dispersion for spatial autoregressive estimates based on sparse algorithms suggest using likelihood ratio tests to determine the significance of the parameters. The approach taken here may suffer from some numerical inaccuracy relative to measures of dispersion based on the

theoretical information matrix, but has the advantage that users will be presented with traditional $t-$statistics on which they can base inferences.

We will have more to say about how our approach to solving large spatial autoregressive estimation problems using sparse matrix algorithms in MATLAB compares to one proposed by Pace and Barry (1997), when we apply the function **far** to a large data set in the next section.

Documentation for the function **far** is presented below. This function was written to perform on both large and small problems. If the problem is small (involving less than 500 observations), the function **far** computes measures of dispersion using the theoretical information matrix. If more observations are involved, the function determines these measures by computing a numerical hessian matrix. (Users may need to decrease the number of observations to less than 500 if they have computers without a large amount of RAM memory.)

```
 PURPOSE: computes 1st-order spatial autoregressive estimates
          y = p*W*y + e, using sparse matrix algorithms
---------------------------------------------------
 USAGE: results = far(y,W,rmin,rmax,convg,maxit)
 where:  y = dependent variable vector
         W = standardized contiguity matrix
      rmin = (optional) minimum value of rho to use in search
      rmax = (optional) maximum value of rho to use in search
     convg = (optional) convergence criterion (default = 1e-8)
     maxit = (optional) maximum # of iterations (default = 500)
---------------------------------------------------
 RETURNS: a structure
        results.meth  = 'far'
        results.rho   = rho
        results.tstat = asymptotic t-stat
        results.yhat  = yhat
        results.resid = residuals
        results.sige  = sige = (y-p*W*y)'*(y-p*W*y)/n
        results.rsqr  = rsquared
        results.lik   = -log likelihood
        results.nobs  = nobs
        results.nvar  = nvar = 1
        results.y     = y data vector
        results.iter   = # of iterations taken
        results.romax  = 1/max eigenvalue of W (or rmax if input)
        results.romin  = 1/min eigenvalue of W (or rmin if input)
 -----------------------------------------------
```

One option we provide allows the user to supply minimum and maximum values of ρ rather than rely on the eigenvalues of W. This might be used if

we wished to constrain the estimation results to a range of say $0 < \rho < 1$. Note also that this would save the time needed to compute the maximum and minimum eigenvalues of the large W matrix.

2.1.2 Applied examples

Given our function **far** that implements maximum likelihood estimation of small and large first-order spatial autoregressive models, we turn attention to illustrating the use of the function with some spatial data sets. In addition to the estimation functions, we have functions **prt** and **plt** that provide printed and graphical presentation of the estimation results.

Example 2.1 provides an illustration of using these functions to estimate a first-order spatial autoregressive model for neighborhood crime from the Anselin (1988) spatial data sample. Note that we convert the variable vector containing crime incidents to deviations from the means form.

```
% ----- Example 2.1 Using the far() function
load wmat.dat;    % standardized 1st-order contiguity matrix
load anselin.dat; % load Anselin (1988) Columbus neighborhood crime data
y = anselin(:,1);
ydev = y - mean(y);
W = wmat;
vnames = strvcat('crime','rho');
res = far(ydev,W);  % do 1st-order spatial autoregression
prt(res,vnames);    % print the output
plt(res,vnames);    % plot actual vs predicted and residuals
```

This example produced the following printed output with the graphical output presented in Figure 2.1. From the output we would infer that a distinct spatial dependence among the crime incidents for the sample of 49 neighborhoods exists since the parameter estimate for ρ has a $t-$statistic of 4.259. We would interpret this statistic in the typical regression fashion to indicate that the estimated ρ lies 4.2 standard deviations away from zero. We also see that this model explains nearly 44% of the variation in crime incidents in deviations from the means form.

```
First-order spatial autoregressive model Estimates
Dependent Variable =       crime
R-squared       =    0.4390
sigma^2         =  153.8452
Nobs, Nvars     =     49,     1
log-likelihood  =       -373.44669
# of iterations =     17
min and max rho =   -1.5362,   1.0000
```

```
************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.669775         4.259172         0.000095
```

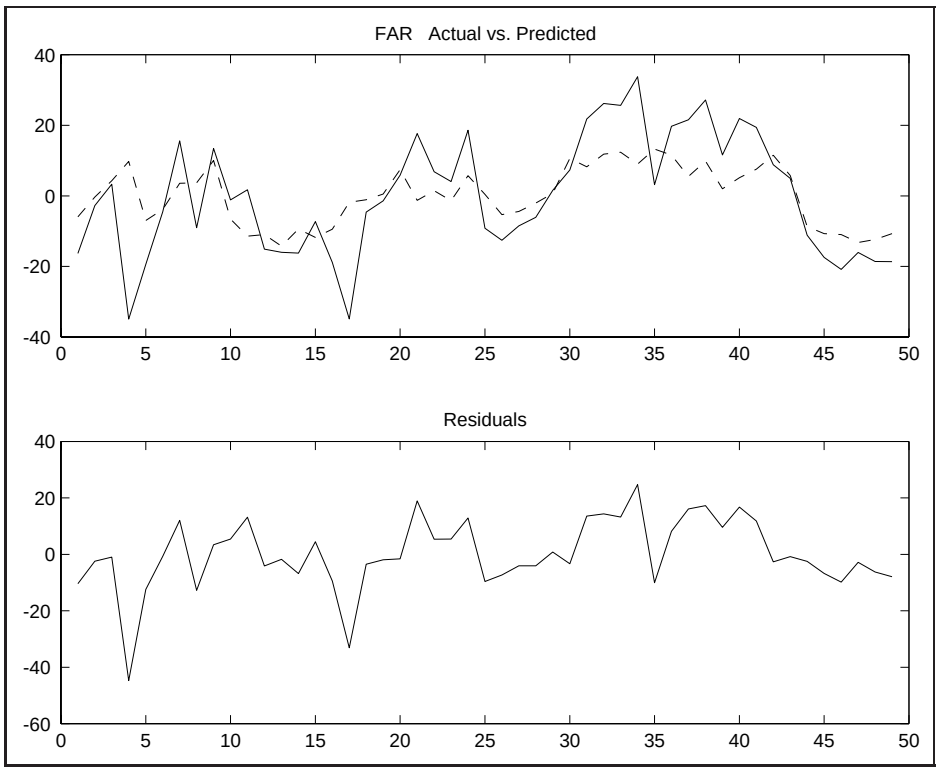

Figure 2.1: Spatial autoregressive fit and residuals

Another more challenging example involves a large sample of 3,107 observations representing counties in the continental U.S. from Pace and Barry (1997). They examine presidential election results for this large sample of observations covering the U.S. presidential election of 1980 between Carter and Reagan. The variable we wish to explain using the first-order spatial autoregressive model is the proportion of total possible votes cast for both candidates. Only persons 18 years and older are eligible to vote, so the proportion is based on those voting for both candidates divided by the population over 18 years of age.

Pace and Barry (1997) suggest an alternative approach to that implemented here in the function **far**. They propose overcoming the difficulty we

face in evaluating the determinant $(I - \rho W)$ by computing this determinant once over a grid of values for the parameter ρ ranging from $1/\lambda_{min}$ to $1/\lambda_{max}$ prior to estimation. They suggest a grid based on 0.01 increments for ρ over the feasible range. Given these pre-determined values for the determinant $(I - \rho W)$, they point out that one can quickly evaluate the log-likelihood function for all values of ρ in the grid and determine the optimal value of ρ as that which maximizes the likelihood function value over this grid. Note that their proposed approach would involve evaluating the determinant around 200 times if the feasible range of ρ was -1 to 1. In many cases the range is even greater than this and would require even more evaluations of the determinant. In contrast, our function **far** reports that only 17 iterations requiring log likelihood function evaluations involving the determinant were needed to solve for the estimates in the case of the Columbus neighborhood crime data set.

In addition, consider that one might need to construct a finer grid around the approximate maximum likelihood value of ρ determined from the initial grid search, whereas our use of the MATLAB simplex algorithm produces an estimate that is accurate to a number of decimal digits.

After some discussion of the computational savings associated with the use of sparse matrices, we illustrate the use of our function **far** and compare it to the approach suggested by Pace and Barry.

A first point to note regarding sparsity is that large problems such as this will inevitably involve a sparse spatial contiguity weighting matrix. This becomes obvious when you consider the contiguity structure of our sample of 3,107 U.S. counties. At most, individual counties exhibited only 8 first-order (rook definition) contiguity relations, so the remaining 2,999 entries in this row are zero. The average number of contiguity relationships is 4, so a great many of the elements in the matrix W are zero, which is the definition of a sparse matrix.

To understand how sparse matrix algorithms conserve on storage space and computer memory, consider that we need only record the non-zero elements of a sparse matrix for storage. Since these represent a small fraction of the total 3107x3107 = 9,653,449 elements in the weight matrix, we save a tremendous amount of computer memory. In fact for our example of the 3,107 U.S. counties, only 12,429 non-zero elements were found in the first-order spatial contiguity matrix, representing a very small fraction (far less than 1 percent) of the total elements.

MATLAB provides a function **sparse** that can be used to construct a large sparse matrix by simply indicating the row and column positions of non-zero elements and the value of the matrix element for these non-zero

row and column elements. Continuing with our example, we can store the first-order contiguity matrix in a single data file containing 12,429 rows with 3 columns that take the form:

```
row column value
```

This represents a considerable savings in computational space when compared to storing a matrix containing 9,653,449 elements. A handy utility function in MATLAB is **spy** which allows one to produce a specially formatted graph showing the sparsity structure associated with sparse matrices. We demonstrate by executing **spy(W)** on our weight matrix W from the Pace and Barry data set, which produced the graph shown in Figure 2.2. As we can see from the figure, most of the non-zero elements reside near the diagonal.

As an example of storing a sparse first-order contiguity matrix, consider example 2.2 below that reads data from the file 'ford.dat' in sparse format and uses the function **sparse** to construct a working spatial contiguity matrix W. The example also produces a graphical display of the sparsity structure using the MATLAB function **spy**.

```
% ----- Example 2.2 Using sparse matrix functions
load ford.dat; % 1st order contiguity matrix
               % stored in sparse matrix form
ii = ford(:,1);
jj = ford(:,2);
ss = ford(:,3);
clear ford;                    % clear out the matrix to save RAM memory
W = sparse(ii,jj,ss,3107,3107);
clear ii; clear jj; clear ss; % clear out these vectors to save memory
spy(W);
```

To compare our function **far** with the approach proposed by Pace and Barry, we implemented their approach and provide timing results. We take a more efficient approach to the grid search over values of the parameter ρ than suggested by Pace and Barry. Rather than search over a large number of values for ρ, we based our search on a large increment of 0.1 for an initial grid of values covering ρ from $1/\lambda_{min}$ to $1/\lambda_{max}$. Given the determinant of $(I - \rho W)$ calculated using sparse matrix algorithms in MATLAB, we evaluated the negative log likelihood function values for this grid of ρ values to find the value that minimizes the likelihood function. We then make a second pass based on a tighter grid with increments of 0.01 around the optimal ρ value found in the first pass. A third and final pass is based on an

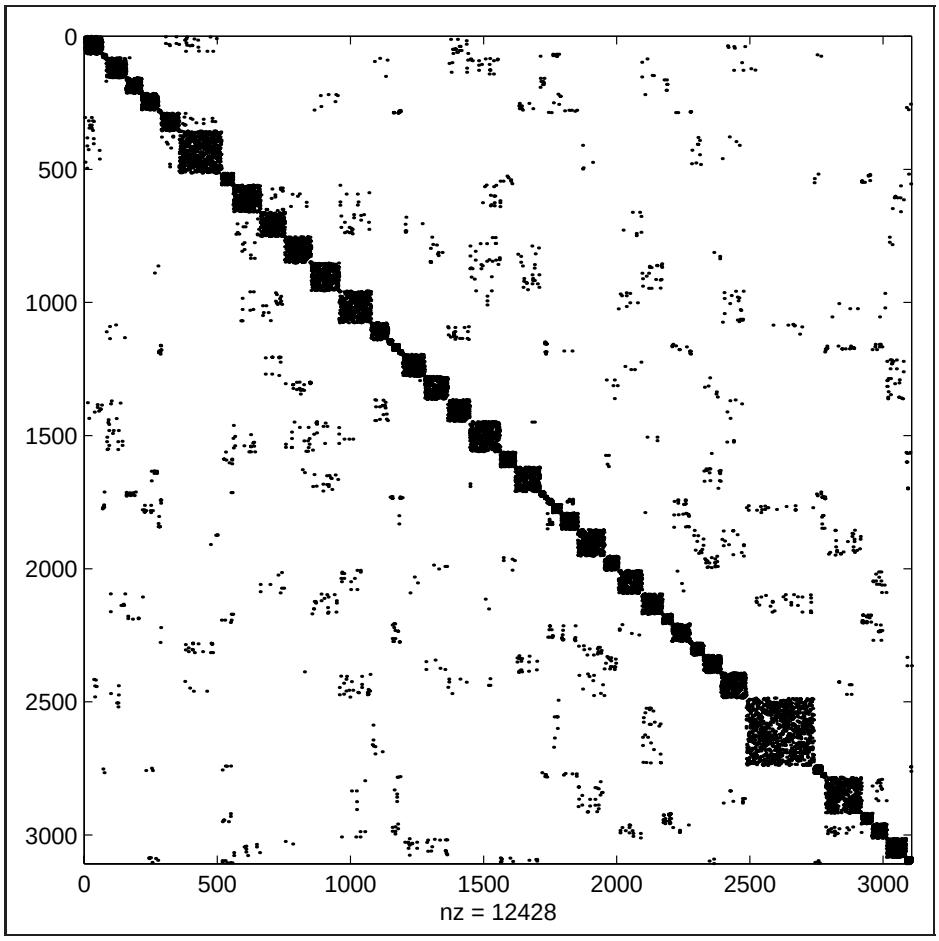

Figure 2.2: Sparsity structure of W from Pace and Barry

even finer grid with increments of 0.001 around the optimal estimate from the second pass.

Note that we used a MATLAB sparse function **eigs** to solve for the eigenvalues of the contiguity matrix W, which requires 60 seconds to solve this part of the problem as shown in the output below. The time necessary to perform each pass over the grid of 21 values for ρ was around 10 seconds. With a total of 3 passes to produce an estimate of ρ accurate to 3 decimal digits, we have a total elapsed time of 1 minute and 30 seconds to solve for the maximum likelihood estimate of ρ. This is certainly a reasonable amount of computational time for such a large problem on a reasonably in-

expensive desktop computing platform. Of course, there is still the problem of producing measures of dispersion for the estimates that Pace and Barry address by suggesting the use of likelihood ratio statistics.

```
elapsed_time = 59.8226 % computing min,max eigenvalues
elapsed_time = 10.5280 % carrying out 1st 21-point grid over rho
elapsed_time = 10.3791 % carrying out 2nd 21-point grid over rho
elapsed_time = 10.3747 % carrying out 3rd 21-point grid over rho
estimate of rho =   0.7220
estimate of sigma =   0.0054
```

How does our approach compare to that of Pace and Barry? Example 2.3 shows a program to estimate the same FAR model using our **far** function.

```
% ----- Example 2.3 Using the far() function
%       with very large data set from Pace and Barry

load elect.dat;             % load data on votes
y = elect(:,7)./elect(:,8); % proportion of voters casting votes
ydev = y - mean(y);         % deviations from the means form
clear y;     % conserve on RAM memory
clear elect; % conserve on RAM memory
load ford.dat; % 1st order contiguity matrix stored in sparse matrix form
ii = ford(:,1); jj = ford(:,2); ss = ford(:,3);
n = 3107;
clear ford; % clear ford matrix to save RAM memory
W = sparse(ii,jj,ss,n,n);
clear ii; clear jj; clear ss; % conserve on RAM memory
tic; res = far(ydev,W); toc;
prt(res);
```

In terms of time needed to solve the problem, our use of the simplex optimization algorithm takes only 10.6 seconds to produce a more accurate estimate than that based on the grid approach of Pace and Barry. Their approach which we modified took 30 seconds to solve for a ρ value accurate to 3 decimal digits. Note also in contrast to Pace and Barry, we compute a conventional measure of dispersion using the numerical hessian estimates which takes only 1.76 seconds. The total time required to compute not only the estimates and measures of dispersion for ρ and σ, but the $R-$squared statistics and log likelihood function was around 100 seconds.

```
elapsed_time = 59.8226 % computing min,max eigenvalues
elapsed_time = 10.6622 % time required for simplex solution of rho
elapsed_time =  1.7681 % time required for hessian evaluation
elapsed_time =  1.7743 % time required for likelihood evaluation
total time   = 74.01   % comparable time to Pace and Barry
```

```
First-order spatial autoregressive model Estimates
R-squared       =    0.5375
sigma^2         =    0.0054
Nobs, Nvars     =   3107,     1
log-likelihood  =        1727.9824
# of iterations =     13
min and max rho =   -1.0710,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.721474        59.495159         0.000000
```

Many of the ideas developed in this section regarding the use of MATLAB sparse matrix algorithms will apply equally to the estimation procedures we develop in the next three sections for the other members of the spatial autoregressive model family.

2.2 The mixed autoregressive-regressive model

This model extends the first-order spatial autoregressive model to include a matrix X of explanatory variables such as those used in traditional regression models. Anselin (1988) provides a maximum likelihood method for estimating the parameters of this model that he labels a 'mixed regressive - spatial autoregressive model'. We will refer to this model as the spatial autoregressive model (SAR). The SAR model takes the form:

$$\begin{aligned} y &= \rho W y + X\beta + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.12}$$

Where y contains an nx1 vector of dependent variables, X represents the usual nxk data matrix containing explanatory variables and W is a known spatial weight matrix, usually a first-order contiguity matrix. The parameter ρ is a coefficient on the spatially lagged dependent variable, Wy, and the parameters β reflect the influence of the explanatory variables on variation in the dependent variable y. The model is termed a mixed regressive - spatial autoregressive model because it combines the standard regression model with a spatially lagged dependent variable, reminiscent of the lagged dependent variable model from time-series analysis.

Maximum likelihood estimation of this model is based on a concentrated likelihood function as was the case with the FAR model. A few regressions

are carried out along with a univariate parameter optimization of the concentrated likelihood function over values of the autoregressive parameter ρ. The steps are enumerated in Anselin (1988) as:

1. perform OLS for the model: $y = X\beta_0 + \varepsilon_0$
2. perform OLS for the model $Wy = X\beta_L + \varepsilon_L$
3. compute residuals $e_0 = y - X\hat{\beta}_0$ and $e_L = Wy - X\hat{\beta}_L$
4. given e_0 and e_L find ρ that maximizes the concentrated likelihood function: $L_C = C - (n/2)\ln(1/n)(e_0 - \rho e_L)'(e_0 - \rho e_L) + \ln|I - \rho W|$
5. given $\hat{\rho}$ that maximizes L_C, compute $\hat{\beta} = (\hat{\beta}_0 - \rho\hat{\beta}_L)$ and $\hat{\sigma}^2_\varepsilon = (1/n)(e_0 - \rho e_L)'(e_0 - \rho e_L)$

Again we face the problem that using a univariate simplex optimization algorithm to find a maximum likelihood estimate of ρ based on the concentrated log likelihood function leaves us with no estimates of the dispersion associated with the parameters. We can overcome this using the theoretical information matrix for small problems and the numerical hessian approach introduced for the FAR model in the case of large problems.

Since this model is quite similar to the FAR model which we already presented, we will turn immediately to describing the function.

2.2.1 The sar() function

The function **sar** is fairly similar to our **far** function, with the documentation presented below.

```
 PURPOSE: computes spatial autoregressive model estimates
          y = p*W*y + X*b + e, using sparse matrix algorithms
-----------------------------------------------------
 USAGE: results = sar(y,x,W,rmin,rmax,convg,maxit)
 where:  y = dependent variable vector
         x = explanatory variables matrix
         W = standardized contiguity matrix
      rmin = (optional) minimum value of rho to use in search
      rmax = (optional) maximum value of rho to use in search
     convg = (optional) convergence criterion (default = 1e-8)
     maxit = (optional) maximum # of iterations (default = 500)
-----------------------------------------------------
 RETURNS: a structure
```

```
        results.meth  = 'sar'
        results.beta  = bhat
        results.rho   = rho
        results.tstat = asymp t-stat (last entry is rho)
        results.yhat  = yhat
        results.resid = residuals
        results.sige  = sige = (y-p*W*y-x*b)'*(y-p*W*y-x*b)/n
        results.rsqr  = rsquared
        results.rbar  = rbar-squared
        results.lik   = -log likelihood
        results.nobs  = # of observations
        results.nvar  = # of explanatory variables in x
        results.y     = y data vector
        results.iter   = # of iterations taken
        results.romax  = 1/max eigenvalue of W (or rmax if input)
        results.romin  = 1/min eigenvalue of W (or rmin if input)
 -------------------------------------------------
```

As in the case of **far**, we allow the user to provide minimum and maximum values of ρ to use in the search. This may save time in cases where we wish to restrict our estimate of ρ to the positive range.

The other point to note is that this function also uses the numerical hessian approach to compute measures of dispersion for large problems involving more than 500 observations.

2.2.2 Applied examples

As an illustration of using the **sar** function, consider the program in example 2.4, where we estimate a model to explain variation in votes casts on a per capita basis in the 3,107 counties. The explanatory variables in the model were: the proportion of population with high school level education or higher, the proportion of the population that are homeowners and the income per capita. Note that the population deflater used to convert the variables to per capita terms was the population 18 years or older in the county.

```
% ----- Example 2.4 Using the sar() function with a very large data set
load elect.dat;             % load data on votes in 3,107 counties
y =  (elect(:,7)./elect(:,8));    % convert to per capita variables
x1 = log(elect(:,9)./elect(:,8)); % education
x2 = log(elect(:,10)./elect(:,8));% homeownership
x3 = log(elect(:,11)./elect(:,8));% income
n = length(y); x = [ones(n,1) x1 x2 x3];
clear x1; clear x2; clear x3;
clear elect;                % conserve on RAM memory
```

```
load ford.dat; % 1st order contiguity matrix stored in sparse matrix form
ii = ford(:,1); jj = ford(:,2); ss = ford(:,3);
n = 3107;
clear ford; % clear ford matrix to save RAM memory
W = sparse(ii,jj,ss,n,n);
clear ii; clear jj; clear ss; % conserve on RAM memory
vnames = strvcat('voters','const','educ','homeowners','income');
to = clock;
res = sar(y,x,W);
etime(clock,to)
prt(res,vnames);
```

We use the MATLAB **clock** function as well as **etime** to determine the overall execution time needed to solve this problem, which was 130 seconds. The estimation results are presented below:

```
Spatial autoregressive Model Estimates
Dependent Variable =       voters
R-squared      =    0.6419
Rbar-squared   =    0.6416
sigma^2        =    0.0042
Nobs, Nvars    =   3107,     4
log-likelihood =        5053.5179
# of iterations =     13
min and max rho =   -1.0710,   1.0000
***************************************************************
Variable        Coefficient      t-statistic    t-probability
const              0.753169        25.963031         0.000000
educ               0.148553        17.341543         0.000000
homeowners         0.208960        26.142340         0.000000
income            -0.085462        -9.413244         0.000000
rho                0.563764        39.797104         0.000000
```

We see from the results that all of the explanatory variables exhibit a significant effect on the variable we wished to explain. The results also indicate that the dependent variable y exhibits strong spatial dependence even after taking the effect of these variables into account as the estimate of ρ on the spatial lagged variable is large and significant.

As an illustration of the bias associated with least-squares estimation of spatial autoregressive models, we present an example based on a spatial sample of 88 observations for counties in the state of Ohio. A sample of average housing values for each of 88 counties in Ohio will be related to population per square mile, the housing density and unemployment rates in each county. This regression relationship can be written as:

$$HOUSE_i = \alpha + \beta POP_i + \gamma HDENSITY_i + \delta UNEMPLOY_i + \varepsilon_i \quad (2.13)$$

The motivation for the regression relationship is that population and household density as well as unemployment rates work to determine the house values in each county. Consider that the advent of suburban sprawl and the notion of urban rent gradients suggests that housing values in contiguous counties should be related. The least-squares relationship in (2.13) ignores the spatial contiguity information whereas the SAR model would allow for this type of variation in the model.

The first task is to construct a spatial contiguity matrix for use with our spatial autoregressive model. This could be accomplished by examining a map of the 88 counties and recording neighboring tracts for every observation, a very tedious task. An alternative is to use the latitude and longitude coordinates to construct a contiguity matrix. We rely on a function **xy2cont** that carries out this task. This function is part of Pace and Barry's Spatial Statistics Toolbox for MATLAB, but has been modified to fit the documentation conventions of the spatial econometrics library. The function documentation is shown below:

```
PURPOSE: uses x,y coord to produce spatial contiguity weight matrices
         with delaunay routine from MATLAB version 5.2
---------------------------------------------------
USAGE: [w1 w2 w3] = xy2cont(xcoord,ycoord)
where:     xcoord = x-direction coordinate vector (nobs x 1)
           ycoord = y-direction coordinate vector (nobs x 1)
---------------------------------------------------
RETURNS: w1 = W*W*S, a row-stochastic spatial weight matrix
         w2 = W*S*W, a symmetric spatial weight matrix (max(eig)=1)
         w3 = diagonal matrix with i,i equal to 1/sqrt(sum of ith row)
---------------------------------------------------
References: Kelley Pace, Spatial Statistics Toolbox 1.0
---------------------------------------------------
```

This function essentially uses triangles connecting the x-y coordinates in space to deduce contiguous entities. As an example of using the function, consider constructing a spatial contiguity matrix for the Columbus neighborhood crime data set where we know both the first-order contiguity structure taken from a map of the neighborhoods as well as the x-y coordinates. Here is a program to generate the first-order contiguity matrix from the latitude and longitude coordinates and produce a graphical comparison of the two contiguity structures shown in Figure 2.3.

```
% ----- Example 2.5 Using the xy2cont() function
load anselin.data;  % Columbus neighborhood crime
xc = anselin(:,5);  % longitude coordinate
yc = anselin(:,4);  % latitude coordinate
```

```
load Wmat.data;     % load standardized contiguity matrix
% create contiguity matrix from x-y coordinates
[W1 W2 W3] = xy2cont(xc,yc);
% graphically compare the two
spy(W2,'ok'); hold on; spy(Wmat,'+k');
legend('generated','actual');
```

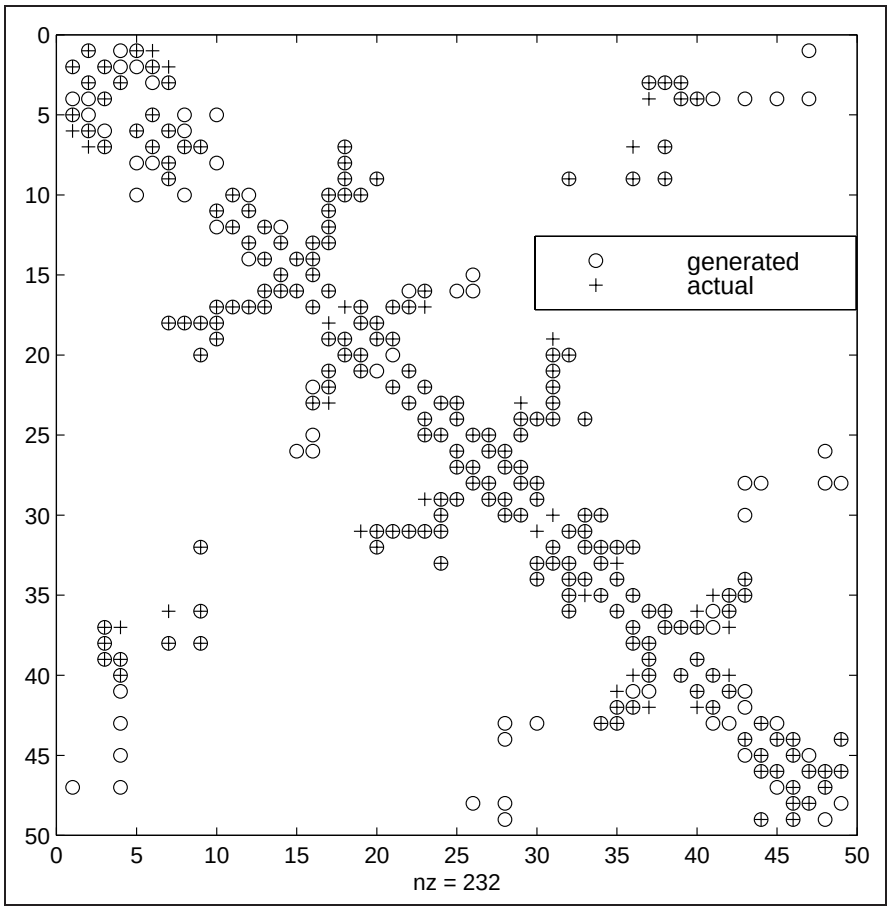

Figure 2.3: Generated contiguity structure results

Example 2.6 reads in the data from two files containing a database for the 88 Ohio counties as well as data vectors containing the latitude and longitude information needed to construct a contiguity matrix. We rely on a log transformation of the dependent variable house values to provide better scaling for the data. Note the use of the MATLAB construct: 'ohio2(:,5)./ohio1(:,2)',

which divides every element in the column vector 'ohio(:,5)' containing total households in each county by every element in the column vector 'ohio1(:,2)', which contains the population for every county. This produces the number of households per capita for each county as an explanatory variable measuring household density.

```
% ----- Example 2.6 Least-squares bias
%       demonstrated with Ohio county data base
load ohio1.dat; % 88 counties (observations)
% 10 columns
% col1  area in square miles
% col2  total population
% col3  population per square mile
% col4  black population
% col5  blacks as a percentage of population
% col6  number of hospitals
% col7  total crimes
% col8  crime rate per capita
% col9  population that are high school graduates
% col10 population that are college graduates
load ohio2.dat; % 88 counties
% 10 columns
% col1  income per capita
% col2  average family income
% col3  families in poverty
% col4  percent of families in poverty
% col5  total number of households
% col6  average housing value
% col7  unemployment rate
% col8  total manufacturing employment
% col9  manufacturing employment as a percent of total
% col10 total employment
load ohio.xy; % latitude-longitude coordinates of county centroids
[junk W junk2] = xy2cont(ohio(:,1),ohio(:,2)); % make W-matrix
y = log(ohio2(:,6)); n = length(y);
x = [ ones(n,1) ohio1(:,3) ohio2(:,5)./ohio1(:,2) ohio2(:,7) ];
vnames = strvcat('hvalue','constant','popsqm','housedensity','urate');
res = ols(y,x);   prt(res,vnames);
res = sar(y,x,W); prt(res,vnames);
```

The results from these two regressions are shown below. The first point to note is that the spatial autocorrelation coefficient estimate for the SAR model is statistically significant, indicating the presence of spatial autocorrelation in the regression relationship. Least-squares ignores this type of variation producing estimates that lead us to conclude that all three explanatory variables are significant in explaining housing values across the 88 county sample. In contrast, the SAR model leads us to conclude that the

population density (popsqm) is not statistically significant at conventional levels. Keep in mind that the OLS estimates are biased and inconsistent, so the inference of significance from OLS we would draw is likely to be incorrect.

```
Ordinary Least-squares Estimates
Dependent Variable =     hvalue
R-squared      =    0.6292
Rbar-squared   =    0.6160
sigma^2        =    0.0219
Durbin-Watson  =    2.0992
Nobs, Nvars    =     88,     4
***************************************************************
Variable          Coefficient      t-statistic    t-probability
constant            11.996858        71.173358         0.000000
popsqm               0.000110         2.983046         0.003735
housedensity        -1.597930        -3.344910         0.001232
urate               -0.067693        -7.525022         0.000000

Spatial autoregressive Model Estimates
Dependent Variable =     hvalue
R-squared       =    0.7298
Rbar-squared    =    0.7201
sigma^2         =    0.0153
Nobs, Nvars     =     88,     4
log-likelihood  =        87.284225
# of iterations =     13
min and max rho =   -2.0158,   1.0000
***************************************************************
Variable          Coefficient      t-statistic    t-probability
constant             6.300144        35.621170         0.000000
popsqm               0.000037         1.196689         0.234794
housedensity        -1.251435        -3.140028         0.002332
urate               -0.055474        -7.387845         0.000000
rho                  0.504131        53.749348         0.000000
```

A second point is that taking the spatial variation into account improves the fit of the model, raising the R-squared statistic for the SAR model. Finally, the magnitudes of the OLS parameter estimates indicate that house values are more sensitive to the household density and the unemployment rate variables than the SAR model. For example, the OLS estimates imply that a one percentage point increase in the unemployment rate leads to a decrease of 6.8 percent in house values whereas the SAR model places this at 5.5 percent. Similarly, the OLS estimates for household density is considerably larger in magnitude than that from the SAR model.

The point of this illustration is that ignoring information regarding the spatial configuration of the data observations will produce different inferences that may lead to an inappropriate model specification. Anselin and Griffith (1988) also provide examples and show that traditional specification tests are plagued by the presence of spatial autocorrelation, so that we should not rely on these tests in the presence of significant spatial autocorrelation.

2.3 The spatial errors model

Here we turn attention to the spatial errors model shown in (2.14), where the disturbances exhibit spatial dependence. Anselin (1988) provides a maximum likelihood method for this model which we label SEM here.

$$\begin{aligned} y &= X\beta + u \\ u &= \lambda W u + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.14}$$

y contains an nx1 vector of dependent variables and X represents the usual nxk data matrix containing explanatory variables. W is a known spatial weight matrix and the parameter λ is a coefficient on the spatially correlated errors analogous to the serial correlation problem in time series models. The parameters β reflect the influence of the explanatory variables on variation in the dependent variable y.

We introduce a number of statistical tests that can be used to detect the presence of spatial autocorrelation in the residuals from a least-squares model. Use of these tests will be illustrated in the next section.

The first test for spatial dependence in the disturbances of a regression model is called Moran's $I-$statistic. Given that this test indicates spatial correlation in the least-squares residuals, the SEM model would be an appropriate way to proceed.

Moran's $I-$statistic takes two forms depending on whether the spatial weight matrix W is standardized or not.

1. W not standardized
$$I = (n/s)[e'We]/e'e \tag{2.15}$$

2. W standardized
$$I = e'We/e'e \tag{2.16}$$

Where e represent regression residuals. Cliff and Ord (1972, 1973, 1981) show that the asymptotic distribution for Moran's I based on least-squares residuals corresponds to a standard normal distribution after adjusting the $I-$statistic by subtracting the mean and dividing by the standard deviation of the statistic. The adjustment takes two forms depending on whether W is standardized or not. (Anselin, 1988, page 102).

1. W not standardized, let $M = (I - X(X'X)^{-1}X')$ and tr denotes the trace operator.

$$\begin{aligned} E(I) &= (n/s)\mathrm{tr}(MW)/(n-k) \\ V(i) &= (n/s)^2[\mathrm{tr}(MWMW') + \mathrm{tr}(MW)^2 + (\mathrm{tr}(MW))^2]/d - E(I)^2 \\ d &= (n-k)(n-k+2) \\ Z_I &= [I - E(I)]/V(I)^{1/2} \end{aligned} \tag{2.17}$$

2. W standardized

$$\begin{aligned} E(I) &= \mathrm{tr}(MW)/(n-k) \\ V(i) &= [\mathrm{tr}(MWMW') + \mathrm{tr}(MW)^2 + (\mathrm{tr}(MW))^2]/d - E(I)^2 \\ d &= (n-k)(n-k+2) \\ Z_I &= [I - E(I)]/V(I)^{1/2} \end{aligned} \tag{2.18}$$

We implement this test in the MATLAB function **moran**, which takes a regression model and spatial weight matrix W as input and returns a structure variable containing the results from a Moran test for spatial correlation in the residuals. The **prt** function can be used to provide a nicely formatted print out of the test results. The help documentation for the function is shown below.

```
 PURPOSE: computes Moran's I-statistic for spatial correlation
          in the residuals of a regression model
---------------------------------------------------
 USAGE: result = moran(y,x,W)
 where: y = dependent variable vector
        x = independent variables matrix
        W = contiguity matrix (standardized or unstandardized)
---------------------------------------------------
 RETURNS: a  structure variable
        result.morani = e'*W*e/e'*e (I-statistic)
        result.istat  = [i - E(i)]/std(i), standardized version
        result.imean  = E(i),   expectation
```

```
        result.ivar   = var(i), variance
        result.prob   = std normal marginal probability
        result.nobs   = # of observations
        result.nvar   = # of variables in x-matrix
-------------------------------------------------
NOTE: istat > 1.96, => small prob,
                    => reject HO: of no spatial correlation
-------------------------------------------------
See also: prt(), lmerrors, walds, lratios
-------------------------------------------------
```

A number of other asymptotic approaches exist for testing whether spatial correlation is present in the residuals from a least-squares regression model. Some of these are the likelihood ratio test, the Wald test and a lagrange multiplier test, all of which are based on maximum likelihood estimation of the SEM model.

The likelihood ratio test is based on the difference between the log likelihood from the SEM model and the log likelihood from a least-squares regression. This quantity represents a statistic that is distributed $\chi^2(1)$. A function **lratios** carries out this test and returns a results structure which can be passed to the **prt** function for presentation of the results. Documentation for the function is:

```
 PURPOSE: computes likelihood ratio test for spatial
          correlation in the errors of a regression model
-------------------------------------------------
 USAGE: result = lratios(y,x,W)
    or: result = lratios(y,x,W,sem_result);
 where:   y = dependent variable vector
          x = independent variables matrix
          W = contiguity matrix (standardized or unstandardized)
 sem_result = a results structure from sem()
-------------------------------------------------
 RETURNS: a  structure variable
        result.meth   = 'lratios'
        result.lratio = likelihood ratio statistic
        result.chi1   = 6.635
        result.prob   = marginal probability
        result.nobs   = # of observations
        result.nvar   = # of variables in x-matrix
-------------------------------------------------
NOTES: lratio > 6.635,  => small prob,
                        => reject HO: of no spatial correlation
       calling the function with a results structure from sem()
       can save time for large models that have already been estimated
-------------------------------------------------
```

Note that we allow the user to supply a 'results' structure variable from the **sem** estimation function, which would save the time needed to estimate the SEM model if the model has already been estimated. This could represent a considerable savings for large problems.

Another approach is based on a Wald test for residual spatial autocorrelation. This test statistic (shown in (2.19)) is distributed $\chi^2(1)$. (Anselin, 1988, page 104).

$$\begin{aligned} W &= \lambda^2[t_2 + t_3 - (1/n)(t_1^2)] \sim \chi^2(1) \\ t_1 &= \mathrm{tr}(W. * B^{-1}) \\ t_2 &= \mathrm{tr}(WB^{-1})^2 \\ t_3 &= \mathrm{tr}(WB^{-1})'(WB^{-1}) \end{aligned} \tag{2.19}$$

Where $B = (I_n - \lambda W)$, with the maximum likelihood estimate of λ used, and $.*$ denotes element-by-element matrix multiplication.

We have implemented a MATLAB function **walds**, that carries out this test. The function documentation is shown below:

```
 PURPOSE: Wald statistic for spatial autocorrelation in
          the residuals of a regression model
-------------------------------------------------
 USAGE: result = walds(y,x,W)
 where: y = dependent variable vector
        x = independent variables matrix
        W = contiguity matrix (standardized)
-------------------------------------------------
 RETURNS: a structure variable
        result.meth = 'walds'
        result.wald = Wald statistic
        result.prob = marginal probability
        result.chi1 = 6.635
        result.nobs = # of observations
        result.nvar = # of variables
-------------------------------------------------
NOTE: wald > 6.635,  => small prob,
                     => reject H0: of no spatial correlation
-------------------------------------------------
See also:  lmerror, lratios, moran
-------------------------------------------------
```

A fourth approach is the Lagrange Multiplier (LM) test which is based on the least-squares residuals and calculations involving the spatial weight matrix W. The LM statistic takes the form: (Anselin, 1988, page 104).

$$\begin{aligned} LM &= (1/T)[(e'We)/\sigma^2]^2 \sim \chi^2(1) \\ T &= \mathrm{tr}(W + W').*W \end{aligned} \tag{2.20}$$

Where e denote least-squares residuals and again we use $.*$ to denote element by element matrix multiplication.

This test is implemented in a MATLAB function **lmerrors** with the documentation for the function shown below.

```
 PURPOSE: LM error statistic for spatial correlation in
          the residuals of a regression model
---------------------------------------------------
 USAGE: result = lmerror(y,x,W)
 where: y = dependent variable vector
        x = independent variables matrix
        W = contiguity matrix (standardized)
---------------------------------------------------
 RETURNS: a structure variable
        result.meth = 'lmerror'
        result.lm   = LM statistic
        result.prob = marginal probability
        result.chi1 = 6.635
        result.nobs = # of observations
        result.nvar = # of variables
---------------------------------------------------
NOTE: lm > 6.635,  => small prob,
                   => reject HO: of no spatial correlation
---------------------------------------------------
See also:  walds, lratios, moran
---------------------------------------------------
```

Finally, a test based on the residuals from the SAR model can be used to examine whether inclusion of the spatial lag term eliminates spatial dependence in the residuals of the model. This test differs from the four tests outlined above in that we allow for the presence of the spatial lagged variable Cy in the model. The test for spatial dependence is conditional on having a ρ parameter not equal to zero in the model, rather than relying on least-squares residuals as in the case of the other four tests.

One could view this test as based on the following model:

$$\begin{aligned} y &= \rho Cy + X\beta + u \\ u &= \lambda Wu + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 I_n) \end{aligned} \tag{2.21}$$

Where the focus of the test is on whether the parameter $\lambda = 0$. This test statistic is also a Lagrange Multiplier statistic based on: (Anselin, 1988, page 106).

$$\begin{aligned}(e'We/\sigma^2)[T_{22} &- (T_{21})^2\text{var}(\rho)]^{-1} \sim \chi^2(1) \\ T_{22} &= \text{tr}(W. * W + W'W) \\ T_{21} &= \text{tr}(W. * CA^{-1} + W'CA^{-1})\end{aligned} \tag{2.22}$$

Where W is the spatial weight matrix shown in (2.21), $A = (I_n - \rho C)$ and var(ρ) is the maximum likelihood estimate of the variance of the parameter ρ in the model.

We have implemented this test in a MATLAB function **lmsar** with the documentation for the function shown below.

```
 PURPOSE: LM statistic for spatial correlation in the
          residuals of a spatial autoregressive model
-------------------------------------------------
 USAGE: result = lmsar(y,x,W1,W2)
 where: y = dependent variable vector
        x = independent variables matrix
       W1 = contiguity matrix for rho
       W2 = contiguity matrix for lambda
-------------------------------------------------
 RETURNS: a structure variable
        result.meth = 'lmsar'
        result.lm   = LM statistic
        result.prob = marginal probability
        result.chi1 = 6.635
        result.nobs = # of observations
        result.nvar = # of variables
-------------------------------------------------
NOTE: lm > 6.635,  => small prob,
                   => reject H0: of no spatial correlation
-------------------------------------------------
See also:  walds, lratios, moran, lmerrors
-------------------------------------------------
```

It should be noted that a host of other methods to test for spatial dependence in various modeling situations have been proposed. In addition, the small sample properties of many alternative tests have been compared in Anselin and Florax (1994) and Anselin and Rey (1991). One point to consider is that many of the matrix computations required for these tests cannot be carried out with very large data samples. We discuss this issue and suggest alternative approaches in the applied examples of Section 2.3.2.

2.3.1 The sem() function

To implement estimation of the spatial error model (SEM) we can draw on the sparse matrix approach we used for the FAR and SAR models. One approach to estimating this model is based on an iterative approach that: 1) constructs least-squares estimates and associated residuals, 2) finds a value of λ that maximizes the log likelihood conditional on the least-squares β values, 3) updates the least-squares values of β using the value of λ determined in step 2). This process is continued until convergence in the residuals.

Next, we present documentation for the function **sem** that carries out the iterative estimation process. This is quite similar in approach to the functions **far** and **sar** already described.

```
 PURPOSE: computes spatial error model estimates
          y = XB + u,  u = L*W*u + e, using sparse algorithms
---------------------------------------------------
 USAGE: results = sem(y,x,W,lmin,lmax,convg,maxit)
 where: y = dependent variable vector
        x = independent variables matrix
        W = contiguity matrix (standardized)
     lmin = (optional) minimum value of lambda to use in search
     lmax = (optional) maximum value of lambda to use in search
    convg = (optional) convergence criterion (default = 1e-8)
    maxit = (optional) maximum # of iterations (default = 500)
---------------------------------------------------
 RETURNS: a structure
        results.meth  = 'sem'
        results.beta  = bhat
        results.lam   = L (lambda)
        results.tstat = asymp t-stats (last entry is lam)
        results.yhat  = yhat
        results.resid = residuals
        results.sige  = sige = e'(I-L*W)'*(I-L*W)*e/n
        results.rsqr = rsquared
        results.rbar = rbar-squared
        results.lik  = log likelihood
        results.nobs = nobs
        results.nvar = nvars (includes lam)
        results.y    = y data vector
        results.iter   = # of iterations taken
        results.lmax  = 1/max eigenvalue of W (or lmax if input)
        results.lmin  = 1/min eigenvalue of W (or lmin if input)
 -------------------------------------------------
```

It should be noted that an alternative approach to estimating this model would be to directly maximize the log likelihood function for this model

using a general optimization algorithm. It might produce an improvement in speed, depending on how many likelihood function evaluations are needed when solving large problems. We provide an option for doing this in a function **semo** that relies on a MATLAB optimization function **maxlik** that is part of my *Econometrics Toolbox* software.

2.3.2 Applied examples

We provide examples of using the functions **moran, lmerror, walds** and **lratios** that test for spatial correlation in the least-squares residuals as well as **lmsar** to test for spatial correlation in the residuals of an SAR model. These examples are based on the Anselin neighborhood crime data set. It should be noted that computation of the Moran $I-$statistic, the LM error statistic, and the Wald test require matrix multiplications involving the large spatial weight matrices C and W. This is not true of the likelihood ratio statistic implemented in the function **lratios**. This test only requires that we compare the likelihood from a least-squares model to that from a spatial error model. As we can produce SEM estimates using our sparse matrix algorithms, this test can be implemented for large models.

Example 2.7 shows a program that carries out all of the test for spatial correlation as well as estimating an SEM model.

```
% ----- Example 2.7 Testing for spatial correlation
load wmat.dat;    % standardized 1st-order contiguity matrix
load anselin.dat; % load Anselin (1988) Columbus neighborhood crime data
y = anselin(:,1); nobs = length(y);
x = [ones(nobs,1) anselin(:,2:3)];
W = wmat;
vnames = strvcat('crime','const','income','house value');
res1 = moran(y,x,W);
prt(res1);
res2 = lmerror(y,x,W);
prt(res2);
res3 = lratios(y,x,W);
prt(res3);
res4 = walds(y,x,W);
prt(res4);
res5 = lmsar(y,x,W,W);
prt(res5);
res = sem(y,x,W);% do 1st-order spatial autoregression
prt(res,vnames); % print the output
```

Note that we have provided code in the **prt** function to provide a nicely formatted print out of the test results from our spatial correlation testing

functions. From the results printed below, we see that the least-squares residuals exhibit spatial correlation. We infer this from the small marginal probabilities that indicate significance at the 99% level of confidence. With regard to the LM error test for spatial correlation in the residuals of the SAR model implemented in the function **lmsar**,we see from the marginal probability of 0.565 that here we can reject any spatial dependence in the residuals from this model.

```
Moran I-test for spatial correlation in residuals
Moran I                    0.23610178
Moran I-statistic          2.95890622
Marginal Probability       0.00500909
mean                      -0.03329718
standard deviation         0.09104680

LM error tests for spatial correlation in residuals
LM value                   5.74566426
Marginal Probability       0.01652940
chi(1) .01 value          17.61100000

LR tests for spatial correlation in residuals
LR value                   8.01911539
Marginal Probability       0.00462862
chi-squared(1) value       6.63500000

Wald test for spatial correlation in residuals
Wald value                14.72873758
Marginal Probability       0.00012414
chi(1) .01 value           6.63500000

LM error tests for spatial correlation in SAR model residuals
LM value                   0.33002340
Marginal Probability       0.56564531
chi(1) .01 value           6.63500000

Spatial error Model Estimates
Dependent Variable =      crime
R-squared       =    0.6515
Rbar-squared    =    0.6364
sigma^2         =   95.5675
log-likelihood  =       -166.40057
Nobs, Nvars     =     49,     3
# iterations    =     12
min and max lam =   -1.5362,   1.0000
***************************************************************
Variable         Coefficient      t-statistic    t-probability
const              59.878750        11.157027         0.000000
income             -0.940247        -2.845229         0.006605
```

```
house value        -0.302236        -3.340320         0.001667
lambda              0.562233         4.351068         0.000075
```

As an example of estimating an SEM model on a large data set, we use the Pace and Barry data set with the same model used to demonstrate the SAR estimation procedure.

```
% ----- Example 2.8 Using the sem() function with a very large data set
load elect.dat;             % load data on votes in 3,107 counties
y =  (elect(:,7)./elect(:,8));    % convert to per capita variables
x1 = log(elect(:,9)./elect(:,8)); % education
x2 = log(elect(:,10)./elect(:,8));% homeownership
x3 = log(elect(:,11)./elect(:,8));% income
n = length(y); x = [ones(n,1) x1 x2 x3];
clear x1; clear x2; clear x3;
clear elect;                % conserve on RAM memory
load ford.dat; % 1st order contiguity matrix stored in sparse matrix form
ii = ford(:,1); jj = ford(:,2); ss = ford(:,3);
n = 3107;
clear ford; % clear ford matrix to save RAM memory
W = sparse(ii,jj,ss,n,n);
clear ii; clear jj; clear ss; % conserve on RAM memory
vnames = strvcat('voters','const','educ','homeowners','income');
to = clock;
res = sem(y,x,W);
etime(clock,to)
prt(res,vnames);
```

We computed estimates using both the iterative procedure implemented in the function **sem** and the optimization procedure implemented in the function **semo**. The time required for the optimization procedure was 338 seconds, which compared to 311 seconds for the iterative procedure. The optimization approach required only 6 function evaluations whereas the iterative procedure required 10 function evaluations. Needless to say, almost all of the time is spent in the log likelihood function evaluations. We present the estimates from both approaches to demonstrate that they produce estimates that are identical to 3 decimal places. Both of these functions are part of the spatial econometrics library, as it may be the case that the optimization approach would produce estimates in less time than the iterative approach in other applications. This would likely be the case if very good initial estimates were available as starting values.

```
% estimates from iterative approach using sem() function
Spatial error Model Estimates
Dependent Variable =       voters
```

```
R-squared       =    0.6570
Rbar-squared    =    0.6566
sigma^2         =    0.0040
log-likelihood  =        5063.9904
Nobs, Nvars     =   3107,     4
# iterations    =     10
min and max lam =   -1.0710,   1.0000
***************************************************************
Variable        Coefficient      t-statistic    t-probability
const              1.216656        35.780274         0.000000
educ               0.192118        14.564908         0.000000
homeowners         0.250041        29.083714         0.000000
income            -0.117625        -9.560005         0.000000
lambda             0.659193        43.121763         0.000000
% estimates from optimization approach using semo() function
Spatial error Model Estimates
Dependent Variable =       voters
R-squared       =    0.6570
Rbar-squared    =    0.6566
sigma^2         =    0.0040
log-likelihood  =       5063.9904
Nobs, Nvars     =   3107,     4
# iterations    =      6
min and max lam =   -1.0710,   1.0000
***************************************************************
Variable        Coefficient      t-statistic    t-probability
const              1.216673        35.780840         0.000000
educ               0.192125        14.565561         0.000000
homeowners         0.250036        29.083079         0.000000
income            -0.117633        -9.560617         0.000000
lambda             0.659176        43.119452         0.000000
```

The estimates from this model indicate that after taking into account the influence of the explanatory variables, we still have spatial correlation in the residuals of the model that can be modeled successfully with the SEM model. As a confirmation of this, consider that the LR test implemented with the function **lratios** produced the results shown below:

```
LR tests for spatial correlation in residuals
LR value                1163.01773404
Marginal Probability       0.00000000
chi-squared(1) value       6.63500000
```

Recall that this is a test of spatial autocorrelation in the residuals from a least-squares model, and the test results provide a strong indication of spatial dependence in the least-squares residuals. Note also that this is the only test that can be implemented successfully with large data sets.

A reasonable alternative would be to simply estimate a FAR model using the least-squares residuals to test for the presence of spatial dependence in the errors. We illustrate this approach in Section 2.5.

2.4 The general spatial model

A general version of the spatial model includes both the spatial lagged term as well as a spatially correlated error structure as shown in (2.23).

$$\begin{aligned} y &= \rho W_1 y + X\beta + u \\ u &= \lambda W_2 u + \varepsilon \\ \varepsilon &\sim N(0, \sigma_\varepsilon^2 I_n) \end{aligned} \tag{2.23}$$

One point to note about this model is that W_1 can equal W_2, but there may be identification problems in this case. The log likelihood for this model can be maximized using our general optimization algorithm on a concentrated version of the likelihood function. The parameters β and σ^2 are concentrated out of the likelihood function, leaving the parameters ρ and λ. This eliminates the ability to use the univariate simplex optimization algorithm **fmin** that we used with the other spatial autoregressive models.

We can still produce a sparse matrix algorithm for the log likelihood function and proceed in a similar fashion to that used for the other spatial autoregressive models. One difference is that we cannot easily impose restrictions on the parameters ρ and λ to force them to lie within the ranges defined by the maximum and minimum eigenvalues from their associated weight matrices W_1 and W_2.

When might one rely on this model? If there were evidence that spatial dependence existed in the error structure from a spatial autoregressive (SAR) model, the SAC model is an appropriate approach to modeling this type of dependence in the errors. Recall, we can use the LM-test implemented in the function **lmsars** to see if spatial dependence exists in the residuals of an SAR model.

Another place where one might rely on this model is a case where a second-order spatial contiguity matrix was used for W_2 that corresponds to a first-order contiguity matrix W_1. This type of model would express the belief that the disturbance structure involved higher-order spatial dependence, perhaps due to second-round effects of a spatial phenomena being modeled.

A third example of using matrices W_1 and W_2 might be where W_1 represented a first-order contiguity matrix and W_2 was constructed as a diagonal matrix measuring the distance from the central city. This type of configuration of the spatial weight matrices would indicate a belief that contiguity alone does not suffice to capture the spatial effects at work. The distance from the central city might also represent an important factor in the phenomena we are modeling. This raises the identification issue, should we use the distance weighting matrix in place of W_1 and the first-order contiguity matrix for W_2, or rely on the opposite configuration? Of course, comparing likelihood function values along with the statistical significance of the parameters ρ and λ from models estimated using both configurations might point to a clear answer.

The log likelihood function for this model is:

$$\begin{aligned} L &= C - (n/2) * ln(\sigma^2) + ln(|A|) + ln(|B|) - (1/2\sigma^2)(e'B'Be) \\ e &= (Ay - X\beta) \\ A &= (I_n - \rho W_1) \\ B &= (I_n - \lambda W_2) \end{aligned} \tag{2.24}$$

We concentrate the function using the following expressions for β and σ^2:

$$\begin{aligned} \beta &= (X'A'AX)^{-1}(X'A'ABy) \\ e &= By - x\beta \\ \sigma^2 &= (e'e)/n \end{aligned} \tag{2.25}$$

Given the expressions in (2.25), we can evaluate the log likelihood given values of ρ and λ. The values of the other parameters β and σ^2 can be calculated as a function of the ρ, λ parameters and the sample data in y, X.

2.4.1 The sac() function

Documentation for the MATLAB function **sac** that carries out the nonlinear optimization of the log likelihood function for this model is shown below. There are a number of things to note about this function. First, we provide optimization options for the user in the form of a structure variable 'info'. These options allow the user to control some aspects of the **maxlik**

optimization algorithm and to print intermediate results while optimization is proceeding.

This is the first example of a function that uses the MATLAB structure variable as an input argument. This allows us to provide a large number of input arguments using a single structure variable. Note that you can name the structure variable used to input the options anything — it is the fieldnames that the function **sac** parses to find the options.

```
 PURPOSE: computes general Spatial Model
 model: y = p*W1*y + X*b + u,  u = lam*W2*u + e
---------------------------------------------------
 USAGE: results = sac(y,x,W1,W2)
 where: y  = dependent variable vector
        x  = independent variables matrix
        W1 = spatial weight matrix (standardized)
        W2 = spatial weight matrix
     info        = a structure variable with optimization options
     info.parm   = (optional) 2x1 starting values for rho, lambda
     info.convg  = (optional) convergence criterion (default = 1e-7)
     info.maxit  = (optional) maximum # of iterations (default = 500)
     info.method = 'bfgs', 'dfp' (default bfgs)
     info.pflag  = flag for printing of intermediate results
---------------------------------------------------
 RETURNS: a structure
        results.meth  = 'sac'
        results.beta  = bhat
        results.rho   = p (rho)
        results.lam   = L (lambda)
        results.tstat = asymptotic t-stats (last 2 are rho,lam)
        results.yhat  = yhat
        results.resid = residuals
        results.sige  = sige = e'(I-L*W)'*(I-L*W)*e/n
        results.rsqr  = rsquared
        results.rbar  = rbar-squared
        results.lik   = likelihood function value
        results.nobs  = nobs
        results.nvar  = nvars
        results.y     = y data vector
        results.iter  = # of iterations taken
 ---------------------------------------------------
```

We take the same approach to optimization failure as we did with the **sem** function. A message is printed to warn the user that optimization failed, but we let the function continue to process and return a results structure consisting of failed parameter estimates. This decision was made to allow the user to examine the failed estimates and attempt estimation based

on alternative optimization options. For example, the user might elect to attempt a Davidson-Fletcher-Powell ('dfp') algorithm in place of the default Broyden-Fletcher-Goldfarb-Smith ('bfgs') routine or supply starting values for the parameters ρ and λ.

With regard to optimization algorithm failures, it should be noted that the *Econometrics Toolbox* contains alternative optimization functions that can be used in place of **maxlik**. Any of these functions could be substituted for **maxlik** in the function **sac**. Chapter 10 in the *Econometrics Toolbox* provides examples of using these functions as well as their documentation.

The next section turns to illustrating the use of the estimation functions we have constructed for the general spatial autoregressive model.

2.4.2 Applied examples

Our first example illustrates the use of the general spatial model with the Anselin Columbus neighborhood crime data set. We construct a matrix W_2 for use in the model based on $W_2 = W'W$. To provide an illustration of what this matrix inner product involving the first-order contiguity matrix W represents in terms of spatial structure, we use the **spy** function to produce comparative graphs of the two contiguity matrices that are shown in Figure 2.4. As we can see from the graphs, the inner product allows for a more wide-ranging set of spatial influences reflecting secondary influences not captured by the first-order contiguity matrix. The sparse matrix literature refers to this phenomena where matrix products produce less sparsity as "fill-in". The motivation for this terminology should be clear from the figure. For another approach to producing "spatial lag" matrices, see the function **slag** which represents a more appropriate way to create higher-order spatial contiguity matrices.

Our example illustrates the point discussed earlier regarding model specification with respect to the use of W and W_2 by producing estimates for two models based on alternative configurations of these two spatial weight matrices. We also produce estimates based on a model that uses the W matrix for both the autoregressive lag and the autoregressive error terms.

```
% ----- Example 2.9 Using the sac function
load Wmat.dat;    % standardized 1st-order contiguity matrix
load anselin.dat; % load Anselin (1988) Columbus neighborhood crime data
y = anselin(:,1); nobs = length(y);
x = [ones(nobs,1) anselin(:,2:3)];
W = Wmat;
vnames = strvcat('crime','const','income','house value');
W2 = W'*W;
```

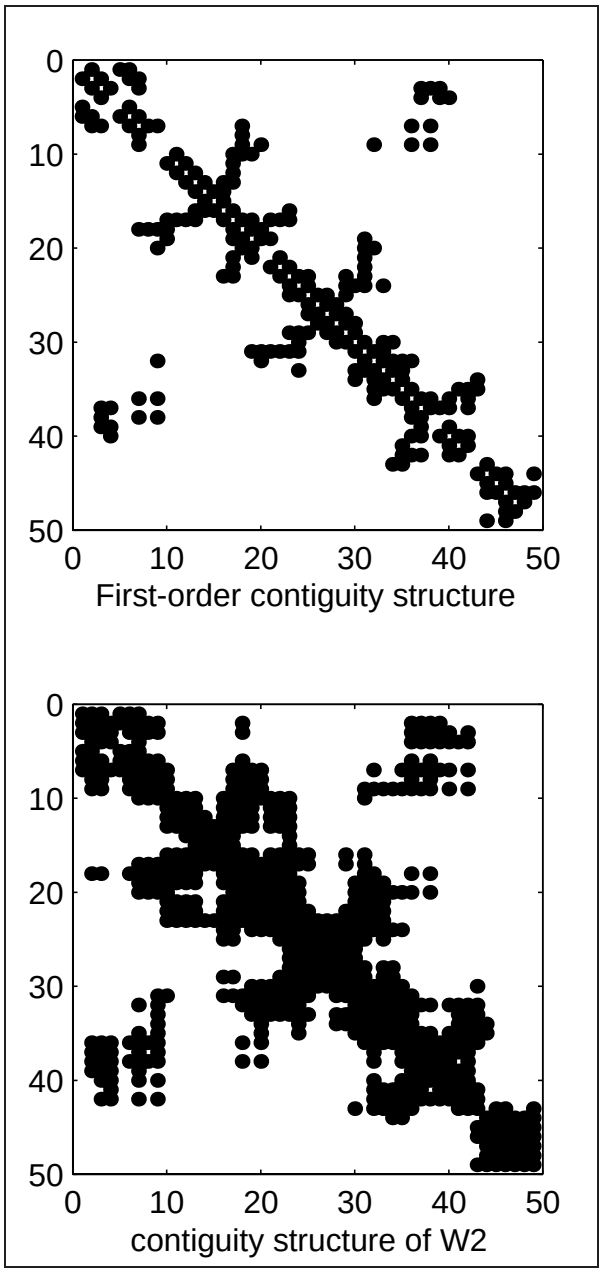

Figure 2.4: Spatial contiguity structures

```
subplot(2,1,1), spy(W);
xlabel('First-order contiguity structure');
subplot(2,1,2), spy(W2);
xlabel('contiguity structure of W2');
res1 = sac(y,x,W2,W);% general spatial model W2,W
prt(res1,vnames);    % print the output
res2 = sac(y,x,W,W2);% general spatial model W,W2
prt(res2,vnames);    % print the output
res3 = sac(y,x,W,W); % general spatial model W,W
prt(res3,vnames);    % print the output
```

The estimation results are shown below for all three versions of the model. The first model produced the worst fit indicated by the lower R^2 value, but a slightly higher log likelihood function value. In addition, we see that both the coefficients on ρ and λ are statistically significant, as indicated by the $t-$statistics and associated marginal probabilities. The negative spatial lag parameter is problematical as this type of influence is difficult to explain intuitively. Negative spatial lags would indicate that higher crime rates in neighboring regions exerted a negative influence on crime on average across the entire Columbus neighborhood sample. When considered from an overall and limiting case viewpoint, this seems a logical impossibility.

```
General Spatial Model Estimates
Dependent Variable =      crime
R-squared      =    0.5591
Rbar-squared   =    0.5400
sigma^2        =  120.9027
log-likelihood =       -164.88367
Nobs, Nvars    =     49,     3
# iterations   =      8
***************************************************************
Variable         Coefficient      t-statistic    t-probability
const              37.261470         5.472898         0.000002
income             -0.958005        -2.573394         0.013360
house value        -0.219046        -2.125609         0.038937
rho                -0.713070        -2.070543         0.044041
lambda              0.571698        18.759773         0.000000

General Spatial Model Estimates
Dependent Variable =      crime
R-squared      =    0.6528
Rbar-squared   =    0.6377
sigma^2        =   95.2216
log-likelihood =       -166.12487
Nobs, Nvars    =     49,     3
# iterations   =      6
***************************************************************
```

```
Variable         Coefficient     t-statistic    t-probability
const              55.343773        4.628119         0.000030
income             -0.672762       -2.163164         0.035760
house value        -0.320741       -3.552861         0.000894
rho                 0.512647        3.747903         0.000497
lambda              0.064988        0.829892         0.410886

General Spatial Model Estimates
Dependent Variable =      crime
R-squared      =    0.6512
Rbar-squared   =    0.6360
sigma^2        =   95.6698
log-likelihood =       -165.25612
Nobs, Nvars    =     49,     3
# iterations   =      7
***************************************************************
Variable         Coefficient     t-statistic    t-probability
const              47.275680        4.138342         0.000147
income             -0.977532       -2.971857         0.004696
house value        -0.284369       -3.137873         0.002968
rho                 0.167197        0.476045         0.636296
lambda              0.368187        1.348037         0.184248
```

The second model produced the typical positive coefficient ρ on the spatial lag term along with an insignificant estimate for λ. As noted above, it also produced a better fit to the sample data. Confirmation of these results from the second model are provided by the test for spatial dependence in the disturbances of the SAR model that we carried out in example 2.7, which are repeated below. From these results, we see that there is no evidence of spatial dependence in the disturbances of the SAR model. This explains why the parameter λ in the general spatial version of this model is not significantly different from zero.

The third model that relies on the same W matrix for both the spatial lag and error correlation terms produced estimates for both ρ and λ that were insignificant. Since we have statistical evidence of spatial correlation in the residuals from least-squares as well as the statistical significance of the parameter ρ in the SAR model, we would reject this model as inferior to the SAR model results.

By way of summary, I would reject the specification associated with the first model since it produces counter-intuitive results. An important point to note regarding any non-linear optimization problem such as that involved in the SAC model is that the estimates may not reflect global optimum. A few different solutions of the optimization problem based on alternative starting values is usually undertaken to confirm that the estimates do indeed

represent global solutions to the problem. The function **sac** allows the user to input alternative starting values, making this relatively easy to do.

Given that we accept the second model as best, because the parameter λ is not statistically significant, we would conclude that an SAR model would suffice for this sample data.

```
LM error tests for spatial correlation in SAR model residuals
LM value                  0.33002340
Marginal Probability      0.56564531
chi(1) .01 value          6.63500000
```

A final example uses the large Pace and Barry data set to illustrate the **sac** function in operation on large problems. Example 2.10 turns on the printing flag so we can observe intermediate results from the optimization algorithm as it proceeds.

```
% ----- Example 2.10 Using the sac() function on a large data set
load elect.dat;             % load data on votes in 3,107 counties
y =  (elect(:,7)./elect(:,8));    % convert to per capita variables
x1 = log(elect(:,9)./elect(:,8)); % education
x2 = log(elect(:,10)./elect(:,8));% homeownership
x3 = log(elect(:,11)./elect(:,8));% income
n = length(y); x = [ones(n,1) x1 x2 x3];
clear x1; clear x2; clear x3;
clear elect;                % conserve on RAM memory
load ford.dat; % 1st order contiguity matrix stored in sparse matrix form
ii = ford(:,1); jj = ford(:,2); ss = ford(:,3);
n = 3107;
clear ford; % clear ford matrix to save RAM memory
W = sparse(ii,jj,ss,n,n);
clear ii; clear jj; clear ss; % conserve on RAM memory
vnames = strvcat('voters','const','educ','homeowners','income');
to = clock; info.pflag = 1;
res = sac(y,x,W,W'*W,info);
etime(clock,to)
prt(res,vnames);
```

The results including intermediate results that were printed are shown below. It took 602 seconds to solve this problem involving only 6 iterations by the **maxlik** function. This function tends to be faster than the alternative optimization algorithms available in the *Econometrics Toolbox*.

```
==== Iteration ==== 2
log-likelihood   bconvergence   fconvergence
    -5052.2920         1.1386         0.4438
Parameter    Estimates   Gradient
```

```
Parameter 1     0.6329  -479.9615
Parameter 2     0.0636 -4927.5720

==== Iteration ==== 3
log-likelihood   bconvergence   fconvergence
    -5068.0928         0.7587         0.0031

Parameter    Estimates   Gradient
Parameter 1     0.6644   -18.8294
Parameter 2     0.0185  -793.3583
==== Iteration ==== 4
log-likelihood   bconvergence   fconvergence
    -5068.2786         0.2197         0.0000
Parameter    Estimates   Gradient
Parameter 1     0.6383   -54.3318
Parameter 2     0.0219    44.6226

==== Iteration ==== 5
log-likelihood   bconvergence   fconvergence
    -5068.5622         0.0405         0.0001
Parameter    Estimates   Gradient
Parameter 1     0.6497    47.7892
Parameter 2     0.0223    27.5841

==== Iteration ==== 6
log-likelihood   bconvergence   fconvergence
    -5068.5627         0.0107         0.0000
Parameter    Estimates   Gradient
Parameter 1     0.6500     0.6407
Parameter 2     0.0221    -3.6765

General Spatial Model Estimates
Dependent Variable =       voters
R-squared      =    0.6598
Rbar-squared   =    0.6594
sigma^2        =    0.0040
log-likelihood =        5068.5627
Nobs, Nvars    =   3107,     4
# iterations   =      6
***************************************************************
Variable        Coefficient     t-statistic    t-probability
const              1.193076        34.732852         0.000000
educ               0.180548        13.639337         0.000000
homeowners         0.267665        30.881364         0.000000
income            -0.107153        -8.596555         0.000000
rho                0.649976        39.916262         0.000000
lambda             0.022120         3.011273         0.002623
```

From the estimation results we see some evidence that a general spatial

model might be appropriate for this modeling problem. The parameters ρ and λ are both statistically significant, but the estimated value of the parameter λ is close to zero. This is consistent with the idea that any second order effects are small. Nonetheless, the fit of this model is slightly better than the SAR and SEM models (see example 2.4 and example 2.8) as indicated by a slightly higher adjusted $R-$squared value and the likelihood function value is also slightly higher.

2.5 An applied exercise

In a well-known paper, Harrison and Rubinfeld (1978) used a housing data set for the Boston SMSA with 506 observations (one observation per census tract) containing 14 variables. They were interested in the housing demand and clean air issues. We will use this spatial data set to illustrate specification and testing for the spatial autoregressive models presented in this chapter. Our dependent variable will be median housing prices for each of the 506 census tracts. The explanatory variables in the dataset are shown below:

```
The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedonic
prices and the demand for clean air', J. Environ. Economics & Management,
vol.5, 81-102, 1978.

Variables in order:
CRIM     per capita crime rate by town
ZN       proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS    proportion of non-retail business acres per town
CHAS     Charles River dummy  (= 1 if tract bounds river; 0 otherwise)
NOX      nitric oxides concentration (parts per 10 million)
RM       average number of rooms per dwelling
AGE      proportion of owner-occupied units built prior to 1940
DIS      weighted distances to five Boston employment centres
RAD      index of accessibility to radial highways
TAX      full-value property-tax rate per $10,000
PTRATIO  pupil-teacher ratio by town
B        1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
LSTAT    percent lower status of the population
MEDV     Median value of owner-occupied homes in $1000's
```

Belsley, Kuh, and Welch (1980) used the data to examine the effects of robust estimation and published the observations in an appendix on pages 244-261. It should be noted that they published data that included various transformations, so the data in their appendix does not match our data

which is in the raw untransformed format. Pace (1993), Gilley and Pace (1996), and Pace and Gilley (1997) have used this data set with spatial econometric models and longitude-latitude coordinates for the census tracts have been added to the dataset.

Our regression model will simply relate the median house values to all of the explanatory variables, simplifying our specification task. We will focus on alternative spatial specifications and models.

The next task involves some scaling and standardization issues surrounding the data set. Belsley, Kuh and Welsch (1980) used this data set to illustrate numerically ill-conditioned data that contained outliers and influential observations. Poor scaling will adversely impact our numerical hessian approach to determining the variance-covariance structure of the spatial autoregressive parameter estimates. Intuitively, the **hessian** function attempts to compute a numerical derivative by perturbing each parameter in turn and examining the impact on the likelihood function. If the parameters vary widely in terms of magnitudes because the data is poorly scaled, this task will be more difficult and we may calculate negative variances.

Example 2.11 demonstrates the nature of these scaling problems, carrying out a least-squares regression. We see that the coefficient estimates varying widely in magnitude, so we scale the variables in the model using a function **studentize** from the *Econometric Toolbox* that subtracts the means and divides by the standard deviations. Another least-squares regression is then carried out to illustrate the impact of scaling on the model coefficients.

```
% ----- Example 2.11 Least-squares on the Boston dataset
load boston.raw; % Harrison-Rubinfeld data
[n k] = size(boston);y = boston(:,k);  % median house values
x = [ones(n,1) boston(:,1:k-1)];       % other variables
vnames = strvcat('hprice','constant','crime','zoning','industry', ...
         'charlesr','noxsq','rooms2','houseage','distance', ...
         'access','taxrate','pupil/teacher','blackpop','lowclass');
res = ols(y,x); prt(res,vnames);
ys = studentize(y); xs = studentize(x(:,2:k));
res2 = ols(ys,xs);
vnames2 = strvcat('hprice','crime','zoning','industry','charlesr', ...
         'noxsq','rooms2','houseage','distance','access','taxrate', ...
         'pupil/teacher','blackpop','lowclass');
prt(res2,vnames2);
% sort actual and predicted by housing values from low to high
yhat = res2.yhat; [ysort yi] = sort(ys); yhats = yhat(yi,1);
tt=1:n; % plot actual vs. predicted
plot(tt,ysort,'ok',tt,yhats,'+k');
ylabel('housing values');
```

```
xlabel('census tract observations');
```

The results indicate that the coefficient estimates based on the unscaled data vary widely in magnitude from 0.000692 to 36.459, whereas the scaled variables produce coefficients ranging from 0.0021 to -0.407.

```
Ordinary Least-squares Estimates (non-scaled variables)
Dependent Variable =    hprice
R-squared      =    0.7406
Rbar-squared   =    0.7338
sigma^2        =   22.5179
Durbin-Watson  =    1.2354
Nobs, Nvars    =    506,    14
***************************************************************
Variable           Coefficient      t-statistic    t-probability
constant             36.459488        7.144074         0.000000
crime                -0.108011       -3.286517         0.001087
zoning                0.046420        3.381576         0.000778
industry              0.020559        0.334310         0.738288
charlesr              2.686734        3.118381         0.001925
noxsq               -17.766611       -4.651257         0.000004
rooms2                3.809865        9.116140         0.000000
houseage              0.000692        0.052402         0.958229
distance             -1.475567       -7.398004         0.000000
access                0.306049        4.612900         0.000005
taxrate              -0.012335       -3.280009         0.001112
pupil/teacher        -0.952747       -7.282511         0.000000
blackpop              0.009312        3.466793         0.000573
lowclass             -0.524758      -10.347146         0.000000

Ordinary Least-squares Estimates (scaled variables)
Dependent Variable =    hprice
R-squared      =    0.7406
Rbar-squared   =    0.7343
sigma^2        =    0.2657
Durbin-Watson  =    1.2354
Nobs, Nvars    =    506,    13
***************************************************************
Variable           Coefficient      t-statistic    t-probability
crime                -0.101017       -3.289855         0.001074
zoning                0.117715        3.385011         0.000769
industry              0.015335        0.334650         0.738032
charlesr              0.074199        3.121548         0.001905
noxsq                -0.223848       -4.655982         0.000004
rooms2                0.291056        9.125400         0.000000
houseage              0.002119        0.052456         0.958187
distance             -0.337836       -7.405518         0.000000
access                0.289749        4.617585         0.000005
```

```
taxrate          -0.226032      -3.283341       0.001099
pupil/teacher    -0.224271      -7.289908       0.000000
blackpop          0.092432       3.470314       0.000565
lowclass         -0.407447     -10.357656       0.000000
```

The program in example 2.11 also produces a plot of the actual versus predicted values from the model sorted by housing values from low to high. From this plot (shown in Figure 2.5), we see large predicted errors for the highest housing values. This suggests a log transformation on the dependent variable y in the model would be appropriate. The figure also illustrates that housing values above $50,000 have been censored to a value of $50,000, a subject we take up in Chapter 5.

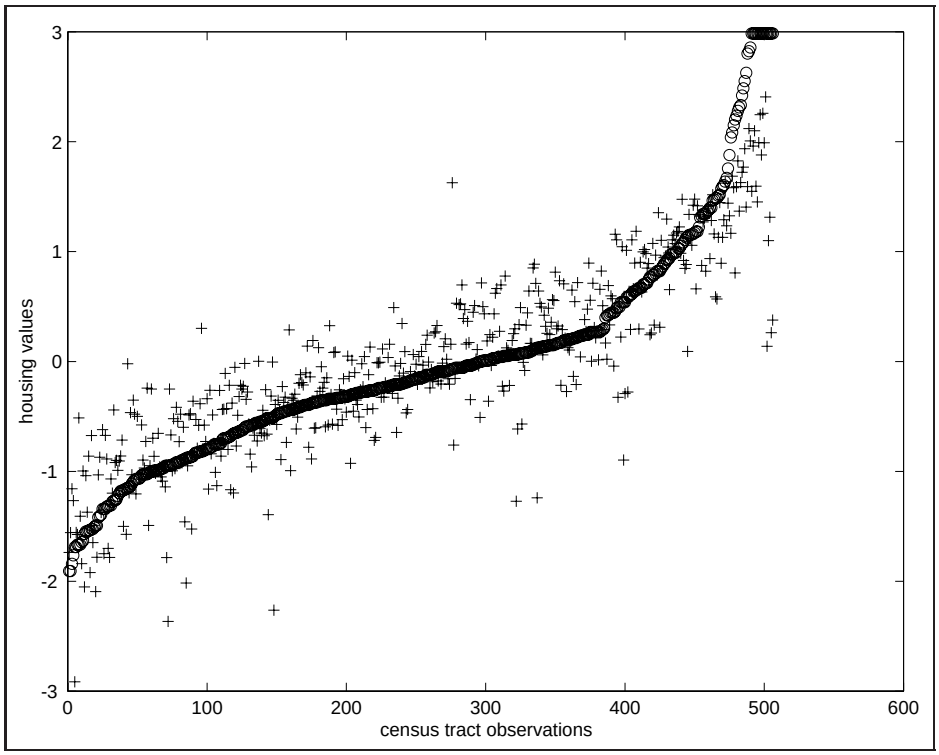

Figure 2.5: Actual vs. Predicted housing values

We adopt a model based on the scaled data and a log transformation for the dependent variable and carry out least-squares estimation again in example 2.12. As a test for spatial autocorrelation in the least-squares residuals, we employ a first-order spatial autoregressive model on the residuals.

We also carry out a Moran's I test for spatial autocorrelation, which may or may not work depending on how much RAM memory you have in your computer. Note that we can directly use the **prt** function without first returning the results from **moran** to a results structure. We rely on our function **xy2cont** to generate a spatial contiguity matrix needed by **far** and **moran** to test for spatial autocorrelation.

```
% ----- Example 2.12 Testing for spatial correlation
load boston.raw; % Harrison-Rubinfeld data
load latitude.data; load longitude.data;
[W1 W W3] = xy2cont(latitude,longitude); % create W-matrix
[n k] = size(boston);y = boston(:,k);     % median house values
x = boston(:,1:k-1);                      % other variables
vnames = strvcat('hprice','crime','zoning','industry','charlesr', ...
         'noxsq','rooms2','houseage','distance','access','taxrate', ...
         'pupil/teacher','blackpop','lowclass');
ys = studentize(log(y)); xs = studentize(x);
res = ols(ys,xs); prt(res,vnames);
resid = res.resid; % recover residuals
rmin = 0; rmax = 1;
res2 = far(resid,W,rmin,rmax); prt(res2);
prt(moran(ys,xs,W));
```

The results shown below indicate that we have strong evidence of spatial autocorrelation in the residuals from the least-squares model. Our FAR model produced a spatial correlation coefficient estimate of 0.647 with a large $t-$statistic and this indication of spatial autocorrelation in the residuals is confirmed by the Moran test results.

```
Ordinary Least-squares Estimates
Dependent Variable =    hprice
R-squared      =    0.7896
Rbar-squared   =    0.7845
sigma^2        =    0.2155
Durbin-Watson  =    1.0926
Nobs, Nvars    =    506,    13
***************************************************************
Variable          Coefficient     t-statistic    t-probability
crime               -0.216146       -7.816230         0.000000
zoning               0.066897        2.136015         0.033170
industry             0.041401        1.003186         0.316263
charlesr             0.062690        2.928450         0.003564
noxsq               -0.220667       -5.096402         0.000000
rooms2               0.156134        5.435516         0.000000
houseage             0.014503        0.398725         0.690269
distance            -0.252873       -6.154893         0.000000
access               0.303919        5.377976         0.000000
```

```
taxrate              -0.258015       -4.161602         0.000037
pupil/teacher        -0.202702       -7.316010         0.000000
blackpop              0.092369        3.850718         0.000133
lowclass             -0.507256      -14.318127         0.000000

(Test for spatial autocorrelation using FAR model)
First-order spatial autoregressive model Estimates
R-squared       =    0.3085
sigma^2         =    0.1452
Nobs, Nvars     =    506,     1
log-likelihood  =       -911.39591
# of iterations =      9
min and max rho =    0.0000,   1.0000
*************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.647624        14.377912         0.000000

Moran I-test for spatial correlation in residuals
Moran I                    0.35833634
Moran I-statistic         14.70009315
Marginal Probability       0.00000000
mean                      -0.01311412
standard deviation         0.02526858
```

Example 2.13 carries out a series of alternative spatial autoregressive models in an attempt to specify the most appropriate model.

```
% ----- Example 2.13 Spatial autoregressive model estimation
load boston.raw; % Harrison-Rubinfeld data
load latitude.data; load longitude.data;
[W1 W W3] = xy2cont(latitude,longitude); % create W-matrix
[n k] = size(boston);y = boston(:,k);     % median house values
x = boston(:,1:k-1);                      % other variables
vnames = strvcat('hprice','crime','zoning','industry','charlesr', ...
         'noxsq','rooms2','houseage','distance','access','taxrate', ...
         'pupil/teacher','blackpop','lowclass');
ys = studentize(log(y)); xs = studentize(x);
rmin = 0; rmax = 1;
tic; res1 = sar(ys,xs,W,rmin,rmax); prt(res1,vnames); toc;
tic; res2 = sem(ys,xs,W,rmin,rmax); prt(res2,vnames); toc;
tic; res3 = sac(ys,xs,W,W);         prt(res3,vnames); toc;
```

The results from example 2.13 are presented below. We see that all three models produced estimates indicating significant spatial autocorrelation. For example, the SAR model produced a coefficient estimate for ρ equal to 0.4508 with a large $t-$statistic, and the SEM model produced an estimate for λ of 0.7576 that was also significant. The SAC model produced estimates of ρ and λ that were both significant at the 99% level.

Which model is best? The log-likelihood function values are much higher for the SEM and SAC models, so this would be evidence against the SAR model. A further test of the SAR model would be to use the function **lmsar** that tests for spatial autocorrelation in the residuals of the SAR model. If we find evidence of residual spatial autocorrelation, it suggests that the SAC model might be most appropriate. Note that the SAC model exhibits the best log-likelihood function value.

```
Spatial autoregressive Model Estimates
Dependent Variable =    hprice
R-squared       =    0.8421
Rbar-squared    =    0.8383
sigma^2         =    0.1576
Nobs, Nvars     =    506,    13
log-likelihood  =       -85.099051
# of iterations =      9
min and max rho =    0.0000,   1.0000
***************************************************************
Variable           Coefficient      t-statistic    t-probability
crime                -0.165349        -6.888522         0.000000
zoning                0.080662         3.009110         0.002754
industry              0.044302         1.255260         0.209979
charlesr              0.017156         0.918665         0.358720
noxsq                -0.129635        -3.433659         0.000646
rooms2                0.160858         6.547560         0.000000
houseage              0.018530         0.595675         0.551666
distance             -0.215249        -6.103520         0.000000
access                0.272237         5.625288         0.000000
taxrate              -0.221229        -4.165999         0.000037
pupil/teacher        -0.102405        -4.088484         0.000051
blackpop              0.077511         3.772044         0.000182
lowclass             -0.337633       -10.149809         0.000000
rho                   0.450871        12.348363         0.000000

Spatial error Model Estimates
Dependent Variable =    hprice
R-squared       =    0.8708
Rbar-squared    =    0.8676
sigma^2         =    0.1290
log-likelihood  =       -58.604971
Nobs, Nvars     =    506,    13
# iterations    =     10
min and max lam =    0.0000,   1.0000
***************************************************************
Variable           Coefficient      t-statistic    t-probability
crime                -0.186710        -8.439402         0.000000
zoning                0.056418         1.820113         0.069348
industry             -0.000172        -0.003579         0.997146
```

```
charlesr             -0.014515       -0.678562        0.497734
noxsq                -0.220228       -3.683553        0.000255
rooms2                0.198585        8.325187        0.000000
houseage             -0.065056       -1.744224        0.081743
distance             -0.224595       -3.421361        0.000675
access                0.352244        5.448380        0.000000
taxrate              -0.257567       -4.527055        0.000008
pupil/teacher        -0.122363       -3.839952        0.000139
blackpop              0.129036        4.802657        0.000002
lowclass             -0.380295      -10.625978        0.000000
lambda                0.757669       19.133467        0.000000

General Spatial Model Estimates
Dependent Variable =    hprice
R-squared      =    0.8662
Rbar-squared   =    0.8630
sigma^2        =    0.1335
log-likelihood =       -55.200525
Nobs, Nvars    =    506,    13
# iterations   =      7
***************************************************************
Variable           Coefficient     t-statistic    t-probability
crime                -0.198184       -8.766862        0.000000
zoning                0.086579        2.824768        0.004923
industry              0.026961        0.585884        0.558222
charlesr             -0.004154       -0.194727        0.845687
noxsq                -0.184557       -3.322769        0.000958
rooms2                0.208631        8.573808        0.000000
houseage             -0.049980       -1.337513        0.181672
distance             -0.283474       -5.147088        0.000000
access                0.335479        5.502331        0.000000
taxrate              -0.257478       -4.533481        0.000007
pupil/teacher        -0.120775       -3.974717        0.000081
blackpop              0.126116        4.768082        0.000002
lowclass             -0.374514      -10.707764        0.000000
rho                   0.625963        9.519920        0.000000
lambda                0.188257        3.059010        0.002342
```

The results from the **lmsar** test shown below indicate the presence of spatial autocorrelation in the residuals of the SAR model, suggesting that the SAC model would be appropriate.

```
LM error tests for spatial correlation in SAR model residuals
LM value                 60.37309581
Marginal Probability      0.00000000
chi(1) .01 value          6.63500000
```

We would conclude that the SAC model is most appropriate here. Regarding inferences, an interesting point is that the Charles River location

dummy variable was statistically significant in the least-squares version of the model, but not in any of the three spatial autoregressive models. Intuitively, taking explicit account of the spatial nature of the data eliminates the need for this locational dummy variable. Other differences in the inferences that would be made from least-squares versus the SAC model center on the magnitudes of 'pupil/teacher' ratio and 'lower class' population variables. The least-squares estimates for these two variables are roughly twice the magnitude of those from the SAC model. Since the other two spatial autoregressive models produce similar estimates for these two variables, we would infer that the least-squares estimates for these coefficients are exhibiting upward bias.

To a lesser extent, we would draw a different inference regarding the magnitude of impact for the 'rooms2' variable from the two spatial autoregressive models that we think most appropriate (SEM and SAC) than least-squares. The SEM and SAC models produce estimates around 0.2 compared to a value of 0.156 from least-squares. Interestingly, the SAR model estimate for this variable is 0.160, close to that from least-squares.

We used this applied data set to explore the accuracy of the numerical hessian versus information matrix approach to determining the variance-covariance matrix for the estimates. Since this is a reasonably large dataset with a large number of explanatory variables, it should provide a good indication of how well the numerical hessian approach does. The $t-$statistics from the SAR, SEM and SAC models estimated with both the information matrix approach and the numerical hessian are presented below.

11.2 sec.	46.1 sec.	79.7 sec.	58.8 sec.	170.1 sec.	114.6 sec.
SAR(info)	SAR(hess)	SEM(info)	SEM(hess)	SAC(info)	SAC(hess)
-6.763498	-6.888522	-8.474183	-8.439402	-8.718561	-8.766862
3.007078	3.009110	1.820262	1.820113	3.266408	2.824768
1.255180	1.255260	-0.003584	-0.003579	0.750766	0.585884
0.923109	0.918665	-0.690453	-0.678562	-0.227615	-0.194727
-3.397977	-3.433659	-3.687834	-3.683553	-4.598758	-3.322769
6.521229	6.547560	8.345316	8.325187	8.941594	8.573808
0.593747	0.595675	-1.756358	-1.744224	-1.597997	-1.337513
-6.037894	-6.103520	-3.435146	-3.421361	-7.690499	-5.147088
5.577032	5.625288	5.478342	5.448380	6.906236	5.502331
-4.147236	-4.165999	-4.527097	-4.527055	-4.985237	-4.533481
-4.025972	-4.088484	-3.886484	-3.839952	-4.765931	-3.974717
3.744277	3.772044	4.808276	4.802657	5.971414	4.768082
-9.900943	-10.149809	-10.954441	-10.625978	-11.650367	-10.707764
10.351829	12.348363	20.890074	19.133467	24.724209	9.519920
				3.747132	3.059010

The numerical approach appears to work very well, producing identical inferences for all coefficients. The SAC model produced the greatest divergence between the $t-$statistics and unfortunately a large discrepancy exists for the spatial parameters ρ and λ in this model. Nonetheless, we would draw the same inferences regarding the significance of these parameters from both approaches to computing measures of dispersion. Pace and Barry (1998) express the belief that the information matrix approach (whether computed numerically or analytically) may not work well for spatial autoregressive models because of the asymmetry of the profile likelihood for these models that arises from the restricted range of the parameter ρ. Additionally, they argue that the necessary "smoothness" needed to compute well-behaved second derivatives may not always occur as Ripley (1988) documents for a particular case. Given this, we should perhaps qualify our evaluation of success regarding the variance-covariance calculations. On the other hand, for this particular data set and model, the variance-covariance structure for the spatial autoregressive models is remarkably similar to that from the least-squares model as indicated by the similar magnitudes for the $t-$statistics. Further, for the single case of the Charles River dummy variable where the least-squares and spatial $t-$statistics differed, we have a plausible explanation for this difference.

The times (in seconds) required by both information matrix and numerical hessian approaches to estimating the variance-covariance structure of the parameters are reported, and we see that the information matrix approach was quite a bit faster for the SAR model but slower for the other two models. One point to consider regarding the comparative execution times is that the spatial contiguity weight matrix is not exceptionally sparse for this problem. Of the (506x506)=256,036 elements, there are 3,006 non-zero entries which is 1.17 percent of the elements. This would bias upward the times reported for the numerical hessian approach since the sparse algorithms can't work to their fullest.

By way of concluding this applied illustration, we might estimate an SAC model based on a second order spatial contiguity matrix in place of the first-order matrix used in example 2.13. We can also produce FAR model estimates using the residuals from the SAC model in example 2.13 as well as the new model based on a second-order spatial weight matrix to check for second order spatial effects in the residuals of our model. Example 2.14 implements these final checks.

```
% ----- Example 2.14 Final model checking
load boston.raw; % Harrison-Rubinfeld data
load latitude.data; load longitude.data;
```

```
[W1 W W3] = xy2cont(latitude,longitude); % create W-matrix
[n k] = size(boston);y = boston(:,k);     % median house values
x = boston(:,1:k-1);                      % other variables
vnames = strvcat('hprice','crime','zoning','industry','charlesr', ...
         'noxsq','rooms2','houseage','distance','access','taxrate', ...
         'pupil/teacher','blackpop','lowclass');
ys = studentize(log(y)); xs = studentize(x);
rmin = 0; rmax = 1;
W2 = W*W;
res = sac(ys,xs,W,W2); prt(res,vnames);
res2 = sac(ys,xs,W,W);
resid = res2.resid; % recover SAC residuals
prt(far(resid,W2,rmin,rmax));
```

The results shown below indicate that the SAC model using a second-order spatial weight matrix produces a slightly lower likelihood function value than the SAC model in example 2.13, but estimates that are reasonably similar in all other regards.

```
General Spatial Model Estimates
Dependent Variable =    hprice
R-squared      =    0.8766
Rbar-squared   =    0.8736
sigma^2        =    0.1231
log-likelihood =        -56.71359
Nobs, Nvars    =    506,    13
# iterations   =      7
***************************************************************
Variable          Coefficient      t-statistic    t-probability
crime               -0.195223        -8.939992         0.000000
zoning               0.085436         2.863017         0.004375
industry             0.018200         0.401430         0.688278
charlesr            -0.008227        -0.399172         0.689939
noxsq               -0.190621        -3.405129         0.000715
rooms2               0.204352         8.727857         0.000000
houseage            -0.056388        -1.551836         0.121343
distance            -0.287894        -5.007506         0.000001
access               0.332325         5.486948         0.000000
taxrate             -0.245156        -4.439223         0.000011
pupil/teacher       -0.109542        -3.609570         0.000338
blackpop             0.127546         4.896422         0.000001
lowclass            -0.363506       -10.368125         0.000000
rho                  0.684424        12.793448         0.000000
lambda               0.208597         2.343469         0.019502

First-order spatial autoregressive model Estimates
R-squared       =    0.0670
sigma^2         =    0.1245
```

```
Nobs, Nvars     =    506,     1
log-likelihood  =        -827.5289
# of iterations =      9
min and max rho =    0.0000,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.188278         1.420560         0.156062
```

The residuals from the SAC model in example 2.13 show no significant second-order spatial autocorrelation, since the FAR model estimate is not statistically significant.

To further explore this model as an exercise, you might consider replacing the W2 weight matrix in example 2.14 with a matrix based on distances. See if this variant of the SAC model is successful. Another exercise would be to estimate a model using: sac(ys,xs,W2,W), and compare it to the model in example 2.14.

2.6 Chapter Summary

We have seen that spatial autoregressive models can be estimated using univariate and bivariate optimization algorithms to solve for estimates by maximizing the likelihood function. The sparse matrix routines in MATLAB allow us to write functions that evaluate the log likelihood function for large models rapidly and with a minimum of computer RAM memory. This approach was used to construct a library of estimation functions that were illustrated on a problem involving all 3,107 counties in the continental U.S. on an inexpensive desktop computer.

In addition to providing functions that estimate these models, the use of a general software design allowed us to provide both printed and graphical presentation of the estimation results.

Another place where we produced functions that can be used in spatial econometric analysis was in the area of testing for spatial dependence in the residuals from least-squares models and SAR models. Functions were devised to implement Moran's $I-$statistic as well as likelihood ratio and Lagrange multiplier tests for spatial autocorrelation in the residuals from least-squares and SAR models. These tests are a bit more hampered by large-scale data sets, but alternative approaches based on using the FAR model on residuals or likelihood ratio tests can be used.

2.7 References

Anselin, L. 1980. *Estimation Methods for Spatial Autoregressive Structures*, (New York: Regional Science Dissertation and Monograph Series 8).

Anselin, L. 1988. *Spatial Econometrics: Methods and Models*, (Dorddrecht: Kluwer Academic Publishers).

Anselin, L. and D.A. Griffith. 1988. "Do spatial effects really matter in regression analysis? *Papers of the Regional Science Association*, 65, pp. 11-34.

Anselin, L. and R.J.G. Florax. 1994. "Small Sample Properties of Tests for Spatial Dependence in Regression Models: Some Further Results", Research paper 9414, *Regional Research Institute*, West Virginia University, Morgantown, West Virginia.

Anselin, L. and S. Rey. 1991. "Properties of tests for spatial dependence in linear regression models", *Geographical Analysis*, Volume 23, pages 112-31.

Belsley, D. A., E. Kuh, and R. E. Welsch. 1980. *Regression Diagnostics: Identifying Influential Data and Source of Collinearity*, (John Wiley: New York).

Cliff, A. and J. Ord, 1972. "Testing for spatial autocorrelation among regression residuals", *Geographical Analysis*, Vol. 4, pp. 267-84.

Cliff, A. and J. Ord, 1973. *Spatial Autocorrelation*, (London: Pion)

Cliff, A. and J. Ord, 1981. *Spatial Processes, Models and Applications*, (London: Pion)

Gilley, O.W., and R. Kelley Pace. 1996. "On the Harrison and Rubinfeld Data," *Journal of Environmental Economics and Management*, Vol. 31 pp. 403-405.

Pace, R. K. and R. Barry. 1997. "Quick Computation of Spatial Autoregressive Estimators", forthcoming in *Geographical Analysis*.

Pace, R. Kelley. 1993. "Nonparametric Methods with Applications to Hedonic Models," *Journal of Real Estate Finance and Economics* Vol. 7, pp. 185-204.

Pace, R. Kelley, and O.W. Gilley. 1997. "Using the Spatial Configuration of the Data to Improve Estimation," *Journal of the Real Estate Finance and Economics* Vol. 14 pp. 333-340.

Pace, R. Kelley, and R. Barry. 1998. "Simulating mixed regressive spatially autoregressive estimators," *Computational Statistics*, Vol. 13 pp. 397-418.

Ripley, Brian D. 1988. *Statistical Inference for Spatial Processes*, (Cambridge University Press: Cambridge, U.K.).

Chapter 3

Bayesian Spatial autoregressive models

This chapter discusses spatial autoregressive models from a Bayesian perspective. It is well-known that Bayesian regression methods implemented with diffuse prior information can replicate maximum likelihood estimation results. We demonstrate this type of application, but focus on some extensions that are available with the Bayesian approach. The maximum likelihood estimation methods set forth in the previous chapter are based on the presumption that the underlying disturbance process involved in generating the model is normally distributed.

Further, many of the formal tests for spatial dependence and heterogeneity including those introduced in the previous chapter rely on characteristics of quadratic forms for normal variates in order to derive the asymptotic distribution of the test statistics.

There is a history of Bayesian literature that deals with heteroscedastic and leptokurtic disturbances, treating these two phenomena in a similar fashion. Geweke (1993) points out that a non-Bayesian regression methodology introduced by Lange, Little and Taylor (1989), which assume an independent Student-t distribution for the regression disturbances, is identical to a Bayesian heteroscedastic linear regression model he introduces. Lange, Little and Taylor (1989) show that their regression model provides robust results in a wide range of applied data settings.

Geweke (1993) argues the same for his method and makes a connection to the Bayesian treatment of symmetric leptokurtic disturbance distributions through the use of scale mixtures of normal distributions. We adopt the approach of Geweke (1993) in order to extend the spatial autoregressive

models introduced in Chapter 2.

The extended version of the model is:

$$\begin{aligned} y &= \rho W_1 y + X\beta + u \\ u &= \lambda W_2 u + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 V) \\ V &= \text{diag}(v_1, v_2, \ldots, v_n) \end{aligned} \tag{3.1}$$

Where the change made to the basic model is in the assumption regarding the disturbances ε. We assume that they exhibit non-constant variance, taking on different values for every observation. The magnitudes $v_i, i = 1, \ldots, n$ represent parameters to be estimated. This assumption of inherent spatial heterogeneity seems more appropriate than the traditional Gauss-Markov assumption that the variance of the disturbance terms is constant over space.

The first section of this chapter introduces a Bayesian heteroscedastic regression model and the topic of Gibbs sampling estimation without complications introduced by the spatial autoregressive model. The next section applies these ideas to the simple FAR model and implements a Gibbs sampling estimation procedure for this model. Following sections deal with the other spatial autoregressive models that we introduced in the previous chapter.

3.1 The Bayesian regression model

We consider the case of a heteroscedastic linear regression model with an informative prior that can be written as in (3.2).

$$\begin{aligned} y &= X\beta + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 V) \\ V &= \text{diag}(v_1, v_2, \ldots, v_n) \\ \beta &\sim N(c, T) \\ \sigma &\sim (1/\sigma) \\ r/v_i &\sim \text{ID } \chi^2(r)/r \\ r &\sim \Gamma(m, k) \end{aligned} \tag{3.2}$$

Where y is an nx1 vector of dependent variables and X represents the nxk matrix of explanatory variables. We assume that ε is an nx1 vector of normally distributed random variates with non-constant variance. We place a normal prior on the parameters β and a diffuse prior on σ. The relative variance terms $(v_1, v_2, \ldots, v_n)$, are assumed fixed but unknown parameters that need to be estimated. The thought of estimating n parameters, $v_1, v_2, \ldots, v_n$, in addition to the k+1 parameters, β, σ using n data observations seems problematical. Bayesian methods don't encounter the same degrees of freedom constraints, because we can rely on an informative prior for these parameters. This prior distribution for the v_i terms will take the form of an independent $\chi^2(r)/r$ distribution. Recall that the χ^2 distribution is a single parameter distribution, where we have represented this parameter as r. This allows us to estimate the additional n v_i parameters in the model by adding the single parameter r to our estimation procedure.

This type of prior has been used by Lindley (1971) for cell variances in an analysis of variance problem, and Geweke (1993) in modeling heteroscedasticity and outliers. The specifics regarding the prior assigned to the v_i terms can be motivated by considering that the prior mean equals unity and the variance of the prior is $2/r$. This implies that as r becomes very large, the terms v_i will all approach unity, resulting in $V = I_n$, the traditional Gauss-Markov assumption. We will see that the role of $V \neq I_n$ is to robustify against outliers and observations containing large variances by downweighting these observations. Large r values are associated with a prior belief that outliers and non-constant variances do not exist.

Now consider the posterior distribution from which we would derive our estimates. Following the usual Bayesian methodology, we would combine the likelihood function for our simple model with the prior distributions for β, σ and V to arrive at the posterior. There is little use in doing this as we produce a complicated function that is not amenable to analysis. As an alternative, consider the conditional distributions for the parameters β, σ and V. These distributions are those that would arise from assuming each of the other parameters were known. For example, the conditional distribution for β assuming that we knew σ and V would looks as follows:

$$\begin{aligned} \beta|(\sigma, V) &\sim N[H(X'V^{-1}y + \sigma^2 R'T^{-1}c) \ , \ \sigma^2 H]. \\ H &= (X'V^{-1}X + R'T^{-1}R)^{-1} \end{aligned} \tag{3.3}$$

Note that this is quite analogous to a generalized least-squares (GLS) version of the Theil and Goldberger (1961) estimation formulas, known as the

"mixed estimator". Consider also that this would be fast and easy to compute.

Next consider the conditional distribution for the parameter σ assuming that we knew the parameters β and V in the problem. This distribution would be:

$$[\sum_{i=1}^{n}(e_i^2/v_i)/\sigma^2]|(\beta, V) \sim \chi^2(n) \tag{3.4}$$

Where we let $e_i = y_i - x_i'\beta$. This result parallels the simple regression case where we know that the residuals are χ^2 distributed. A difference from the standard case is that we adjust the e_i using the relative variance terms v_i.

Finally, the conditional distribution for the parameters V represent a χ^2 distribution with $r + 1$ degrees of freedom (see Geweke, 1993):

$$[(\sigma^{-2}e_i^2 + r)/v_i]|(\beta, \sigma) \sim \chi^2(r + 1) \tag{3.5}$$

The Gibbs sampler provides a way to sample from a multivariate posterior probability density of the type we encounter in this estimation problem based only on the densities of subsets of vectors conditional on all others. In other words, we can use the conditional distributions set forth above to produce estimates for our model, despite the fact that the posterior distribution was not tractable.

The Gibbs sampling approach that we will use throughout this chapter to estimate Bayesian variants of the spatial autoregressive models is based on this simple idea. We specify the conditional distributions for all of the parameters in the model and proceed to carry out random draws from these distributions until we collect a large sample of parameter draws. Gelfand and Smith (1990) demonstrate that Gibbs sampling from the sequence of complete conditional distributions for all parameters in the model produces a set of draws that converge in the limit to the true (joint) posterior distribution of the parameters. That is, despite the use of conditional distributions in our sampling scheme, a large sample of the draws can be used to produce valid posterior inferences about the mean and moments of the multivariate posterior parameter distribution.

The method is most easily described by developing and implementing a Gibbs sampler for our heteroscedastic Bayesian regression model. Given the three conditional posterior densities in (3.3) through (3.5), we can formulate a Gibbs sampler for this model using the following steps:

1. Begin with arbitrary values for the parameters β^0, σ^0 and v_i^0 which we designate with the superscript 0.

2. Compute the mean and variance of β using (3.3) conditional on the initial values σ^0 and v_i^0.

3. Use the computed mean and variance of β to draw a multivariate normal random vector, which we label β^1.

4. Calculate expression (3.4) using β^1 determined in step 3 and use this value along with a random $\chi^2(n)$ draw to determine σ^1.

5. Using β^1 and σ^1, calculate expression (3.5) and use the value along with an $n-$vector of random $\chi^2(r+1)$ draws to determine $v_i, i = 1, \ldots, n$.

These steps constitute a single pass of the Gibbs sampler. We wish to make a large number of passes to build up a sample $(\beta^j, \sigma^j, v_i^j)$ of j values from which we can approximate the posterior distributions for our parameters.

To illustrate this approach in practice, consider example 3.1 which shows the MATLAB code for carry out estimation using the Gibbs sampler set forth above. In this example, we generate a regression model data set that contains a heteroscedastic set of disturbances based on a time trend variable. Only the last 50 observations in the generated data sample contain non-constant variances. This allows us to see if the estimated v_i parameters detect this pattern of non-constant variance over the last half of the sample.

The generated data set used values of unity for β_0, the intercept term, and the two slope parameters, β_1 and β_2. The prior means for the β parameters were set to the true values of unity with prior variances of unity, reflecting a fair amount of uncertainty. The following MATLAB program implements the Gibbs sampler for this model. Note how easy it is to implement the mathematical equations in MATLAB code.

```
% ----- Example 3.1 Heteroscedastic Gibbs sampler
n=100; k=3; % set number of observations and variables
x = randn(n,k); b = ones(k,1);  % generate data set
tt = ones(n,1); tt(51:100,1) = [1:50]';
y = x*b + randn(n,1).*sqrt(tt); % heteroscedastic disturbances
ndraw = 1100; nomit = 100;      % set the number of draws
bsave = zeros(ndraw,k);         % allocate storage for results
ssave = zeros(ndraw,1);
vsave = zeros(ndraw,n);
c = [1.0 1.0 1.0]';             % prior b means
R = eye(k); T = eye(k);         % prior b variance
Q = chol(inv(T)); q = Q*c;
b0 = x\y;                       % use ols starting values
```

```
sige = (y-x*b0)'*(y-x*b0)/(n-k);
V = ones(n,1); in = ones(n,1);  % initial value for V
rval = 4;                       % initial value for rval
qpq = Q'*Q; qpv = Q'*q;         % calculate Q'Q, Q'q only once
tic;                            % start timing
for i=1:ndraw;                  % Start the sampling
  ys = y.*sqrt(V); xs = matmul(x,sqrt(V));
  xpxi = inv(xs'*xs + sige*qpq);
  b = xpxi*(xs'*ys + sige*qpv); % update b
  b = norm_rnd(sige*xpxi) + b;  % draw MV normal mean(b), var(b)
  bsave(i,:) = b';              % save b draws
  e = ys - xs*b; ssr = e'*e;    % update sige
  chi = chis_rnd(1,n);          % do chisquared(n) draw
  sige = ssr/chi; ssave(i,1) = sige; % save sige draws
  chiv = chis_rnd(n,rval+1);    % update vi
  vi = ((e.*e./sige) + in*rval)./chiv;
  V = in./vi; vsave(i,:) = vi'; % save the draw
end;                            % End the sampling
toc;                            % stop timing
bhat = mean(bsave(nomit+1:ndraw,:));  % calculate means and std deviations
bstd = std(bsave(nomit+1:ndraw,:));   tstat = bhat./bstd;
smean = mean(ssave(nomit+1:ndraw,1));
vmean = mean(vsave(nomit+1:ndraw,:));
tout = tdis_prb(tstat',n); % compute t-stat significance levels
% set up for printing results
in.cnames = strvcat('Coefficient','t-statistic','t-probability');
in.rnames = strvcat('Variable','variable 1','variable 2','variable 3');
in.fmt = '%16.6f'; tmp = [bhat' tstat' tout];
fprintf(1,'Gibbs estimates \n'); % print results
mprint(tmp,in);
fprintf(1,'Sigma estimate = %16.8f \n',smean);
result = theil(y,x,c,R,T);       % compare to Theil-Goldberger estimates
prt(result);  plot(vmean);       % plot vi-estimates
title('mean of vi-estimates');
```

We rely on MATLAB functions **norm_rnd** and **chis_rnd** to provide the multivariate normal and chi-squared random draws. These functions are part of the *Econometrics Toolbox* and are discussed in Chapter 8 of the manual. Note also, we omit the first 100 draws at start-up to allow the Gibbs sampler to achieve a steady state before we begin sampling for the parameter distributions.

The results are shown below, where we find that it took only 11.5 seconds to carry out the 1100 draws and produce a sample of 1000 draws on which we can base our posterior inferences regarding the parameters β and σ. For comparison purposes, we produced estimates using the **theil** function from the *Econometrics Toolbox* that implements mixed estimation. These

estimates are similar, but the $t-$statistics are lower because they suffer from the heteroscedasticity. Our Gibbs sampled estimates take this into account increasing the precision of the estimates.

```
elapsed_time =
   11.5534
Gibbs estimates
Variable        Coefficient      t-statistic    t-probability
variable 1         1.071951         2.784286         0.006417
variable 2         1.238357         3.319429         0.001259
variable 3         1.254292         3.770474         0.000276
Sigma estimate =      10.58238737

Theil-Goldberger Regression Estimates
R-squared      =    0.2316
Rbar-squared   =    0.2158
sigma^2        =   14.3266
Durbin-Watson  =    1.7493
Nobs, Nvars    =    100,     3
***************************************************************
Variable         Prior Mean    Std Deviation
variable 1         1.000000         1.000000
variable 2         1.000000         1.000000
variable 3         1.000000         1.000000
***************************************************************
      Posterior Estimates
Variable        Coefficient      t-statistic    t-probability
variable 1         1.056404         0.696458         0.487808
variable 2         1.325063         0.944376         0.347324
variable 3         1.293844         0.928027         0.355697
```

Figure 3.1 shows the mean of the 1,000 draws for the parameters v_i plotted for the 100 observation sample. Recall that the last 50 observations contained a time-trend generated pattern of non-constant variance. This pattern was detected quite accurately by the estimated v_i terms.

One point that should be noted about Gibbs sampling estimation is that convergence of the sampler needs to be diagnosed by the user. The *Econometrics Toolbox* provides a set of convergence diagnostic functions along with illustrations of their use in Chapter 5 of the manual. Fortunately, for simple regression models (and spatial autoregressive models) convergence of the sampler is usually a certainty, and convergence occurs quite rapidly. A simple approach to testing for convergence is to run the sampler once to carry out a small number of draws, say 300 to 500, and a second time to carry out a larger number of draws, say 1000 to 2000. If the means and variances for the posterior estimates are similar from both runs, convergence seems assured.

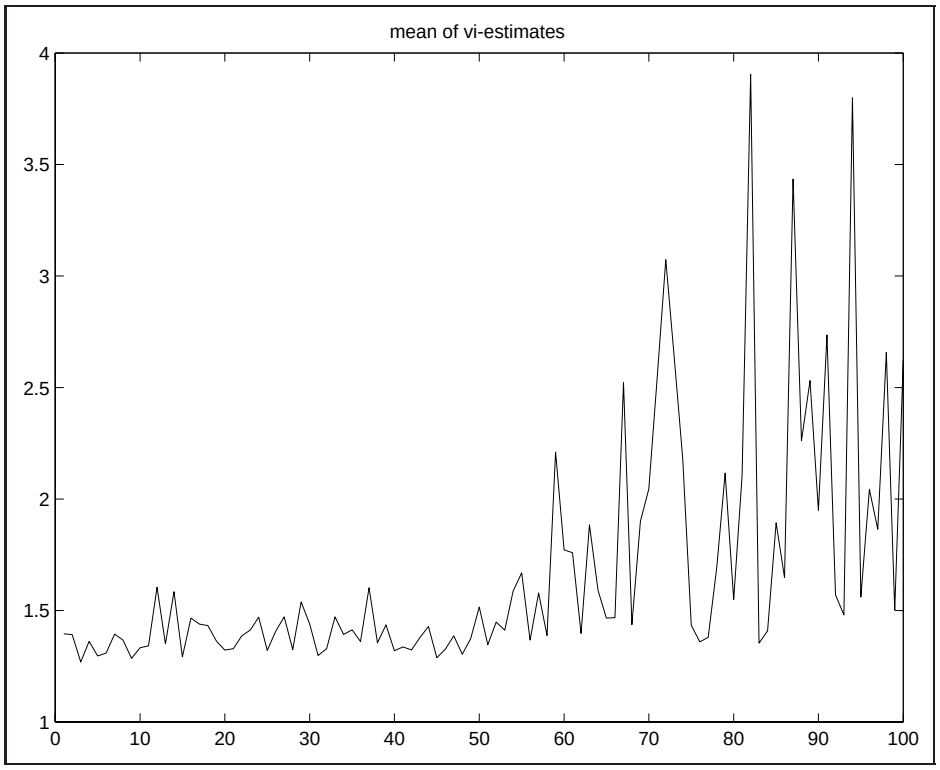

Figure 3.1: V_i estimates from the Gibbs sampler

3.2 The Bayesian FAR model

In this section we turn attention to a Gibbs sampling approach for the FAR model that can accommodate heteroscedastic disturbances and outliers. Note that the presence of a few spatial outliers due to enclave effects or other aberrations in the spatial sample will produce a violation of normality in small samples. The distribution of disturbances will take on a fat-tailed or leptokurtic shape. This is precisely the type of problem that the heteroscedastic modeling approach of Geweke (1993) based on Gibbs sampling estimation was designed to address.

The Bayesian extension of the FAR model takes the form:

$$\begin{aligned} y &= \rho W y + \varepsilon \\ \varepsilon &\sim N(0, \sigma^2 V) \end{aligned} \tag{3.6}$$

$$\begin{aligned} V &= \text{diag}(v_1, v_2, \ldots, v_n) \\ \rho &\sim N(c, T) \\ r/v_i &\sim \text{ID } \chi^2(r)/r \\ r &\sim \Gamma(m, k) \\ \sigma &\sim \Gamma(\nu_0, d_0) \end{aligned}$$

Where as in Chapter 2, the spatial contiguity matrix W has been standardized to have row sums of unity and the variable vector y is expressed in deviations from the means to eliminate the constant term in the model. We allow for an informative prior on the spatial autoregressive parameter ρ, the heteroscedastic control parameter r and the disturbance variance σ. This is the most general Bayesian model, but practitioners would probably implement diffuse priors for σ and ρ.

A diffuse prior for ρ would be implemented by setting the prior mean c to zero and using a large prior variance for T, say 1e+12. To implement a diffuse prior for σ we would set $\nu_0 = 0, d_0 = 0$. The prior for r is based on a $\Gamma(m, k)$ distribution which has a mean equal to m/k and a variance equal to m/k^2. Recall our discussion of the role of the prior hyperparameter r in allowing the v_i estimates to deviate from their prior means of unity. Small values for r around 2 to 7 allow for non-constant variance and are associated with a prior belief that outliers or non-constant variance exist. Large values such as $r = 20$ or $r = 50$ would produce v_i estimates that are all close to unity, forcing the model to take on a homoscedastic character and produce estimates equivalent to those from the maximum likelihood FAR model discussed in Chapter 2. This would make little sense — if we wished to produce maximum likelihood estimates, it would be much quicker to use the **far** function from Chapter 2. In the heteroscedastic regression model demonstrated in example 3.1, we set $r = 4$ to allow ample opportunity for the v_i parameters to deviate from unity. Note that in example 3.1, the v_i estimates for the first 50 observations were all close to unity despite our prior setting of $r = 4$. We will provide examples that suggests an optimal strategy for setting r is to use small values in the range from 2 to 7. If the sample data exhibits homoscedastic disturbances that are free from outliers, the v_i estimates will reflect this fact. On the other hand, if there is evidence of heterogeneity in the errors, these settings for the hyperparameter r will allow the v_i estimates to deviate substantially from unity. Estimates for the v_i parameters that deviate from unity are needed to produce an adjustment in the estimated ρ and σ that take non-constant variance into account or robustify our estimates in the presence of outliers.

Econometric estimation problems amenable to Gibbs sampling can take one of two forms. The simplest case is where all of the conditional distributions are from well-known distributions allowing us to sample random deviates using standard computational algorithms. This was the case with our heteroscedastic Bayesian regression model.

A second more complicated case that one sometimes encounters in Gibbs sampling is where one or more of the conditional distributions can be expressed mathematically, but they take an unknown form. It is still possible to implement a Gibbs sampler for these models using a host of alternative methods that are available to produce draws from distributions taking non-standard forms.

One of the more commonly used ways to deal with this situation is known as the 'Metropolis algorithm'. It turns out that the FAR model falls into this latter category requiring us to rely on what is known as a Metropolis-within-Gibbs sampler. To see how this problem arises, consider the conditional distributions for the FAR model parameters where we rely on diffuse priors, $\pi(\rho)$ and $\pi(\sigma)$ for the parameters (ρ, σ) shown in (3.7).

$$\begin{aligned} \pi(\rho) &\propto \text{constant} \\ \pi(\sigma) &\propto (1/\sigma), \quad 0 < \sigma < +\infty \end{aligned} \tag{3.7}$$

These priors can be combined with the likelihood for this model producing a joint posterior distribution for the parameters, $p(\rho, \sigma|y)$.

$$p(\rho, \sigma|y) \propto |I_n - \rho W|\sigma^{-(n+1)}\exp\{-\frac{1}{2\sigma^2}(y - \rho Wy)'(y - \rho Wy)\} \tag{3.8}$$

If we treat ρ as known, the kernel for the conditional posterior (that part of the distribution that ignores inessential constants) for σ given ρ takes the form:

$$p(\sigma|\rho, y) \propto \sigma^{-(n+1)}\exp\{-\frac{1}{2\sigma^2}\varepsilon'\varepsilon\} \tag{3.9}$$

where $\varepsilon = y - \rho Wy$. It is important to note that by conditioning on ρ (treating it as known) we can subsume the determinant, $|I_n - \rho W|$, as part of the constant of proportionality, leaving us with one of the standard distributional forms. From (3.9) we conclude that $\sigma^2 \sim \chi^2(n)$.

Unfortunately, the conditional distribution of ρ given σ takes the following non-standard form:

$$p(\rho|\sigma, y) \propto \sigma^{-n/2}|I_n - \rho W|\{(y - \rho Wy)'(y - \rho Wy)\}^{-n/2} \tag{3.10}$$

To sample from (3.10) we can rely on a method called 'Metropolis sampling', within the Gibbs sampling sequence, hence it is often labeled 'Metropolis-within-Gibbs'.

Metroplis sampling is described here for the case of a symmetric normal candidate generating density. This should work well for the conditional distribution of ρ because, as Figure 3.2 shows, a conditional distribution of ρ is similar to a normal distribution with the same mean value. The figure also shows a $t-$distribution with 3 degrees of freedom, which would also work well in this application.

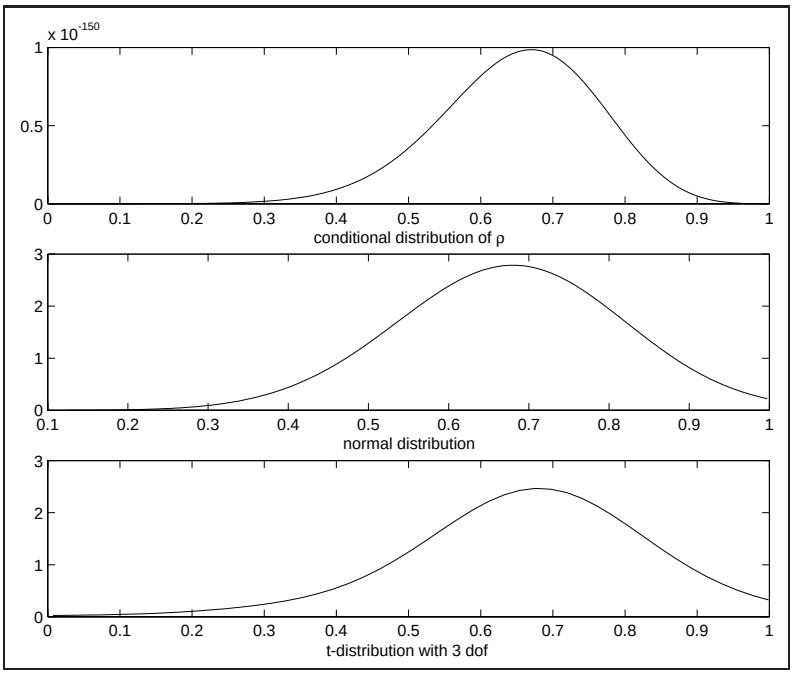

Figure 3.2: Conditional distribution of ρ

To describe Metropolis sampling in general, suppose we are interested in sampling from a density $f()$ and x_0 denotes the current draw from f. Let the candidate value be generated by $y = x_0 + cZ$, where Z is a draw from a standard normal distribution and c is a known constant. (If we wished

to rely on a $t-$distribution, we could simply replace Z with a random draw from the $t-$distribution.)

An acceptance probability is computed using: $p = min\{1, f(y)/f(x_0)\}$. We then draw a uniform random deviate we label U, and if $U < p$, the next draw from f is given by $x_1 = y$. If on the other hand, $U \geq p$, the draw is taken to be the current value, $x_1 = x_0$.

A MATLAB program to implement this approach for the case of the homoscedastic first-order spatial autoregressive (FAR) model is shown in example 3.2. We use this simple case as an introduction before turning to the heteroscedastic case. Note also that we do not rely on the sparse matrix algorithms which we will turn attention to after this introductory example.

An implementation issue is that we need to impose the restriction:

$$1/\lambda_{min} < \rho < 1/\lambda_{max}$$

where λ_{min} and λ_{max} are the minimum and maximum eigenvalues of the standardized spatial weight matrix W. We impose this restriction using an approach that has been labeled 'rejection sampling'. Restrictions such as this, as well as non-linear restrictions, can be imposed on the parameters during Gibbs sampling by simply rejecting values that do not meet the restrictions (see Gelfand, Hills, Racine-Poon and Smith, 1990).

```
% ----- Example 3.2 Metropolis within Gibbs sampling FAR model
n=49; ndraw = 1100; nomit = 100; nadj = ndraw-nomit;
% generate data based on a given W-matrix
load wmat.dat; W = wmat; IN = eye(n); in = ones(n,1); weig = eig(W);
lmin = 1/min(weig); lmax = 1/max(weig); % bounds on rho
rho = 0.7;    % true value of rho
y = inv(IN-rho*W)*randn(n,1); ydev = y - mean(y); Wy = W*ydev;
              % set starting values
rho = 0.5;    % starting value for the sampler
sige = 10.0;  % starting value for the sampler
c = 0.5;      % for the Metropolis step (adjusted during sampling)
rsave = zeros(nadj,1); % storage for results
ssave = zeros(nadj,1);  rtmp = zeros(nomit,1);
iter = 1; cnt = 0;
while (iter <= ndraw);                % start sampling;
e = ydev - rho*Wy; ssr = (e'*e);      % update sige;
chi = chis_rnd(1,n); sige = (ssr/chi);
% metropolis step to get rho update
rhox = c_rho(rho,sige,ydev,W);        % c_rho evaluates conditional
rho2 = rho + c*randn(1); accept = 0;
 while accept == 0;                   % rejection bounds on rho
  if ((rho2 > lmin) & (rho2 < lmax)); accept = 1;  end;
 rho2 = rho + c*randn(1); cnt = cnt+1;
```

```
 end;                                  % end of rejection for rho
rhoy = c_rho(rho2,sige,ydev,W);        % c_rho evaluates conditional
ru = unif_rnd(1,0,1); ratio = rhoy/rhox; p = min(1,ratio);
if (ru < p)
 rho = rho2;
 end;
 rtmp(iter,1) = rho;
  if (iter >= nomit);
    if iter == nomit          % update c based on initial draws
    c = 2*std(rtmp(1:nomit,1));
    end;
   ssave(iter-nomit+1,1) = sige; rsave(iter-nomit+1,1) = rho;
  end; % end of if iter > nomit
iter = iter+1;
end; % end of sampling loop
% print-out results
fprintf(1,'hit rate =  %6.4f \n',ndraw/cnt);
fprintf(1,'mean and std of rho %6.3f %6.3f \n',mean(rsave),std(rsave));
fprintf(1,'mean and std of sig %6.3f %6.3f \n',mean(ssave),std(ssave));
% maximum likelihood estimation for comparison
res = far(ydev,W);
prt(res);
```

Rejection sampling is implemented in the example with the following code fragment that examines the candidate draws in 'rho2' to see if they are in the feasible range. If 'rho2' is not in the feasible range, another candidate value 'rho2' is drawn and we increment a counter variable 'cnt' to keep track of how many candidate values are found outside the feasible range. The 'while loop' continues to draw new candidate values and examine whether they are in the feasible range until we find a candidate value within the limits. Finding this value terminates the 'while loop'. This approach ensures than any values of ρ that are ultimately accepted as draws will meet the constraints.

```
% metropolis step to get rho update
rho2 = rho + c*randn(1); accept = 0;
 while accept == 0;              % rejection bounds on rho
  if ((rho2 > lmin) & (rho2 < lmax)); accept = 1;  end;
 rho2 = rho + c*randn(1); cnt = cnt+1;
 end;                            % end of rejection for rho
```

Another point to note about the example is that we adjust the parameter 'c' used to produce the random normal candidate values. This is done by using the initial nomit=100 values of 'rho' to compute a new value for 'c' based on two standard deviations of the initial draws. The following code

fragment carries this out, where the initial 'rho' draws have been stored in a vector 'rtmp'.

```
if iter == nomit          % update c based on initial draws
 c = 2*std(rtmp(1:nomit,1));
end;
```

Consider also, that we delay collecting our sample of draws for the parameters ρ and σ until we have executed 'nomit' burn-in draws, which is 100 in this case. This allows the sampler to settle into a steady state, which might be required if poor values of ρ and σ were used to initialize the sampler. In theory, any arbitrary values can be used, but a choice of good values will speed up convergence of the sampler. A plot of the first 100 values drawn from this example is shown in Figure 3.3. We used $\rho = -0.5$ and $\sigma^2 = 100$ as starting values despite our knowledge that the true values were $\rho = 0.7$ and $\sigma^2 = 1$. The plots of the first 100 values indicate that even if we start with very poor values, far from the true values used to generate the data, only a few iterations are required to reach a steady state. This is usually true for regression-based Gibbs samplers.

The function **c_rho** evaluates the conditional distribution for ρ given σ^2 at any value of ρ. Of course, we could use sparse matrix algorithms in this function to handle large data sample problems, which is the way we approach this task when constructing our function **far_g** for the spatial econometrics library.

```
function yout = c_rho(rho,sige,y,W)
% evaluates conditional distribution of rho
% given sige for the spatial autoregressive model
n = length(y);
IN = eye(n); B = IN - rho*W;
detval = log(det(B));
epe = y*B'*B*y;
yout = (n/2)*log(sige) + (n/2)*log(epe) - detval;
```

Finally, we present results from executing the code shown in example 3.2, where both Gibbs estimates based on the mean of the 1,000 draws for ρ and σ as well as the standard deviations are shown. For contrast, we present maximum likelihood estimates, which for the case of the homoscedastic Gibbs sampler implemented here with a diffuse prior on ρ and σ should produce similar estimates.

```
Gibbs sampling estimates
hit rate =  0.3561
```

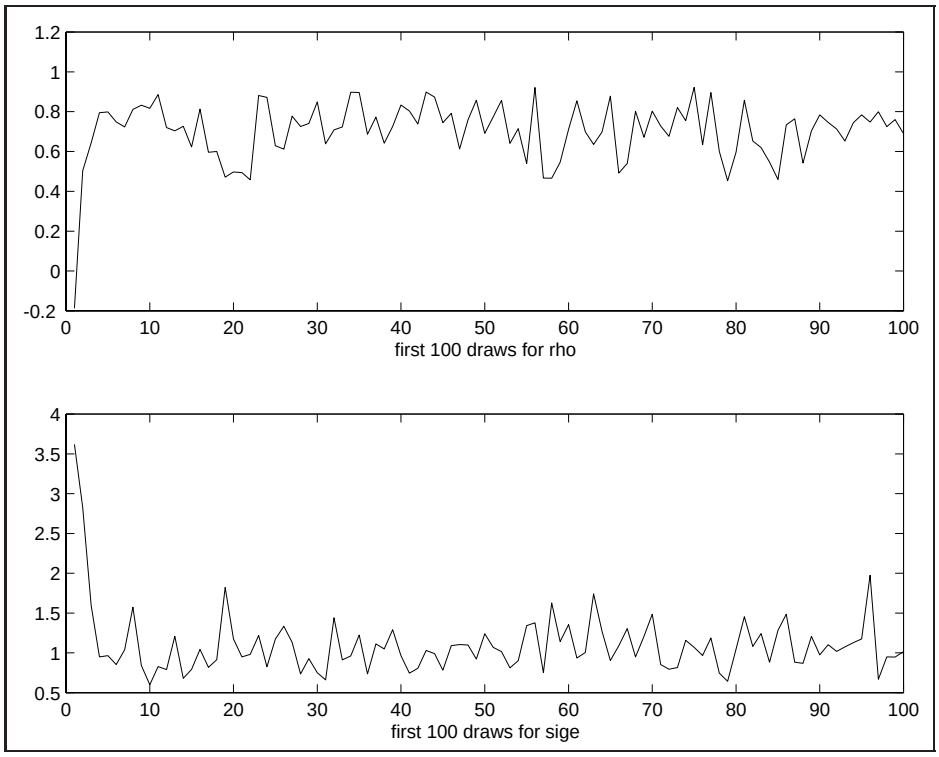

Figure 3.3: First 100 Gibbs draws for ρ and σ

```
mean and std of rho  0.649  0.128
mean and std of sig  1.348  0.312

First-order spatial autoregressive model Estimates
R-squared       =    0.4067
sigma^2         =    1.2575
Nobs, Nvars     =     49,     1
log-likelihood  =       -137.94653
# of iterations =     13
min and max rho =   -1.5362,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.672436         4.298918         0.000084
```

The time needed to generate 1,100 draws was around 10 seconds, which represents 100 draws per second. We will see that similar speed can be achieved even for large data samples.

From the results we see that the mean and standard deviations from

the Gibbs sampler produce estimates very close to the maximum likelihood estimates, and to the true values used to generate the model data. The mean estimate for $\rho = 0.649$ divided by the standard deviation of 0.128 implies a $t-$statistic of 5.070, which is very close to the maximum likelihood $t-$statistic. The estimate of $\sigma^2 = 1.348$ based on the mean from 1,000 draws is also very close to the true value of unity used to generate the model.

The reader should keep in mind that we do not advocate using the Gibbs sampler in place of maximum likelihood estimation. That is, we don't really wish to implement a homoscedastic version of the FAR model Gibbs sampler that relies on diffuse priors. We turn attention to the more general heteroscedastic case that allows for either diffuse or informative priors in the next section.

3.2.1 The far_g() function

We discuss some of the implementation details concerned with constructing a MATLAB function **far_g** to produce estimates for the Bayesian FAR model. This function will rely on a sparse matrix algorithm approach to handle problems involving large data samples. It will also allow for diffuse or informative priors and handle the case of heterogeneity in the disturbance variance.

The first thing we need to consider is that to produce a large number of draws, say 1,000, we would need to evaluate the conditional distribution of ρ 2,000 times. (Note that we called this function twice in example 3.2). Each evaluation would require that we compute the determinant of the matrix $(I_n - \rho W)$, which we have already seen is a non-trivial task for large data samples. To avoid this, we rely on the Pace and Barry (1997) approach discussed in the previous chapter. Recall that they suggested evaluating this determinant over a grid of values in the feasible range of ρ once at the outset. Given that we have carried out this evaluation and stored the determinant values along with associated ρ values, we can simply "look-up" the appropriate determinant in our function that evaluates the conditional distribution. That is, the call to the conditional distribution function will provide a value of ρ for which we need to evaluate the conditional distribution. If we already know the determinant for a grid of all feasible ρ values, we can simply look up the determinant value closest to the ρ value and use it during evaluation of the conditional distribution. This saves us the time involved in computing the determinant twice for each draw of ρ.

Since we need to carry out a large number of draws, this approach works better than computing determinants for every draw. Note that in the case

of maximum likelihood estimation from Chapter 2, the opposite was true. There we only needed 10 to 20 evaluations of the likelihood function, making the initial grid calculation approach of Pace and Barry much slower.

Now we turn attention to the function **far_g** that implements the Gibbs sampler for the FAR model. The documentation for the function is shown below. You can of course examine the code in the function, but it essentially carries out the approach set forth in example 3.2, with modifications to allow for the relative variance parameters v_i and informative priors for ρ, σ and r in this extended version of the model.

```
 PURPOSE: Gibbs sampling estimates of the 1st-order Spatial
          model: y = rho*W*y + e,    e = N(0,sige*V),
          V = diag(v1,v2,...vn), r/vi = ID chi(r)/r, r = Gamma(m,k)
          rho = N(c,T),  sige = gamma(nu,d0)
-------------------------------------------------------------
 USAGE: result =  far_g(y,W,ndraw,nomit,prior,start)
 where: y = nobs x 1 independent variable vector
        W = nobs x nobs 1st-order contiguity matrix (standardized)
       ndraw = # of draws
       nomit = # of initial draws omitted for burn-in
       prior = a structure variable for prior information input
       prior.rho,  prior mean for rho,  c above, default = 0 (diffuse)
       prior.rcov, prior rho  variance, T above, default = 1e+12 (diffuse)
       prior.nu,   informative Gamma(nu,d0) prior on sige
       prior.d0    default: nu=0,d0=0 (diffuse prior)
       prior.rval, r prior hyperparameter, default=4
       prior.m,    informative Gamma(m,k) prior on r
       prior.k,    default: not used
       prior.rmin, (optional) min value of rho to use in sampling
       prior.rmax, (optional) max value of rho to use in sampling
       start = (optional) (2x1) vector of rho, sige starting values
                  (defaults, rho = 0.5, sige = 1.0)
-------------------------------------------------------------
 RETURNS: a structure:
          results.meth   = 'far_g'
          results.pdraw  = rho draws (ndraw-nomit x 1)
          results.sdraw  = sige draws (ndraw-nomit x 1)
          results.vmean  = mean of vi draws (1 x nobs)
          results.yhat   = predicted values of y
          results.rdraw  = r-value draws (ndraw-nomit x 1)
          results.pmean  = rho prior mean    (if prior input)
          results.pstd   = rho prior std dev (if prior input)
          results.nu     = prior nu-value for sige (if prior input)
          results.d0     = prior d0-value for sige (if prior input)
          results.r      = value of hyperparameter r (if input)
          results.m      = m prior parameter (if input)
          results.k      = k prior parameter (if input)
```

```
          results.nobs   = # of observations
          results.ndraw  = # of draws
          results.nomit  = # of initial draws omitted
          results.y      = actual observations
          results.yhat   = predicted values for y
          results.time   = time taken for sampling
          results.accept = acceptance rate
          results.pflag  = 1 for prior, 0 for no prior
          results.rmax   = 1/max eigenvalue of W (or rmax if input)
          results.rmin   = 1/min eigenvalue of W (or rmin if input)
------------------------------------------------------------
 NOTE: use either improper prior.rval
       or informative Gamma prior.m, prior.k, not both of them
------------------------------------------------------------
```

3.2.2 Applied examples

As the documentation makes clear, there are a number of user options to facilitate different models. Example 3.3 illustrates using the function with various input options. We generate a FAR model vector y based on the standardized W weight matrix from the Columbus neighborhood crime data set. The program then produces maximum likelihood estimates for comparison to our Gibbs sampled estimates. The first set of Gibbs estimates are produced with a homoscedastic prior based on $r = 30$ and diffuse priors for ρ and σ. Diffuse priors for ρ and σ are the defaults used by **far_g**, and the default for r equals 4. My experience indicates this represents a good rule-of-thumb value.

After producing the first estimates, we add two outliers to the data set at observations 10 and 39. We then compare maximum likelihood estimates to the Gibbs sampled estimates based on a heteroscedastic prior with $r = 4$.

```
% ----- Example 3.3 Using the far_g function
load wmat.dat; % standardized 1st-order spatial weight matrix
W = wmat;      % from the Columbus neighborhood data set
[n junk] = size(W); IN = eye(n);
rho = 0.75;    % true value of rho
y = inv(IN-rho*W)*randn(n,1)*5; % generate data
ydev = y - mean(y);
vnames = strvcat('y-simulated','y-spatial lag');
rmin = 0; rmax = 1;
resml = far(ydev,W,rmin,rmax); % do maximum likelihood for comparison
prt(resml,vnames);
ndraw = 1100; nomit = 100;
prior.rval = 30; % homoscedastic prior diffuse rho,sigma (the default)
prior.rmin = 0; prior.rmax = 1;
result = far_g(ydev,W,ndraw,nomit,prior); % call Gibbs sampling function
```

```
prt(result,vnames);

% add outliers to the generated data
ydev(20,1) =  ydev(20,1)*10; ydev(39,1) = ydev(39,1)*10;
prior.rval = 4; % heteroscedastic model, diffuse rho,sigma (the default)
resml2 = far(ydev,W); % do maximum likelihood for comparison
prt(resml2,vnames);
result2 = far_g(ydev,W,ndraw,nomit,prior); % call Gibbs sampling function
prt(result2,vnames);
% plot the mean of the vi-draws, which represent vi-estimates
plot(result2.vmean);
```

The program produced the following output. Note that our printing function computes means and standard deviations using the draws returned in the results structure of **far_g**. We also compute $t-$statistics and evaluate the marginal probabilities. This allows us to provide printed output in the form of a traditional regression model.

```
% homoscedastic models
First-order spatial autoregressive model Estimates
Dependent Variable =    y-simulated
R-squared       =    0.5908
sigma^2         =   24.9531
Nobs, Nvars     =     49,     1
log-likelihood  =       -285.86289
# of iterations =      9
min and max rho =    0.0000,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.771980         6.319186         0.000000

Gibbs sampling First-order spatial autoregressive model
Dependent Variable =    y-simulated
R-squared       =    0.5805
sigma^2         =   25.2766
r-value         =     30
Nobs, Nvars     =     49,     1
ndraws,nomit    =   1100,   100
acceptance rate =    0.8886
time in secs    =   13.7984
min and max rho =    0.0000,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.739867         8.211548         0.000000

% outlier models
First-order spatial autoregressive model Estimates
Dependent Variable =    y-simulated
```

```
R-squared       =    0.0999
sigma^2         =  267.3453
Nobs, Nvars     =     49,     1
log-likelihood  =       -398.06025
# of iterations =     14
min and max rho =   -1.5362,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.368404         1.591690         0.118019

Gibbs sampling First-order spatial autoregressive model
Dependent Variable =    y-simulated
R-squared       =    0.1190
sigma^2         =  107.8131
r-value         =      3
Nobs, Nvars     =     49,     1
ndraws,nomit    =   1100,   100
acceptance rate =    0.8568
time in secs    =    8.2693
min and max rho =    0.0000,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.503992         3.190913         0.002501
```

The first two sets of output illustrate the point we made regarding Bayesian analysis implemented with a diffuse prior. The results from the Gibbs sampling approach are very close to those from maximum likelihood estimation. Note that the printed output shows the time required to carry out 1,100 draws along with the acceptance rate. It took only 13.7 seconds to produce 1,100 draws.

After introducing two outliers, we see that the maximum likelihood estimates produce a poor fit to the data as well as an inflated estimate of σ^2. The coefficient estimate for ρ is also affected adversely, deviating from the true value of 0.75, and the precision of the estimate is degraded. In contrast, the Gibbs sampled estimate of ρ was closer to the true value and exhibits greater precision as indicated by the larger $t-$statistic. The estimate for σ^2 is much smaller than that from maximum likelihood. Robust estimates will generally exhibit a smaller R^2 statistic as the estimates downweight outliers rather than try to fit these data observations.

We also produced a plot of the mean of the v_i draws which serve as an estimate of these relative variance terms. This graph is shown in Figure 3.4, where we see that the two outliers were identified.

Example 3.4 illustrates the use of the **far_g** function on the large Pace and Barry data set. We set an r value of 4 which will capture heterogeneity

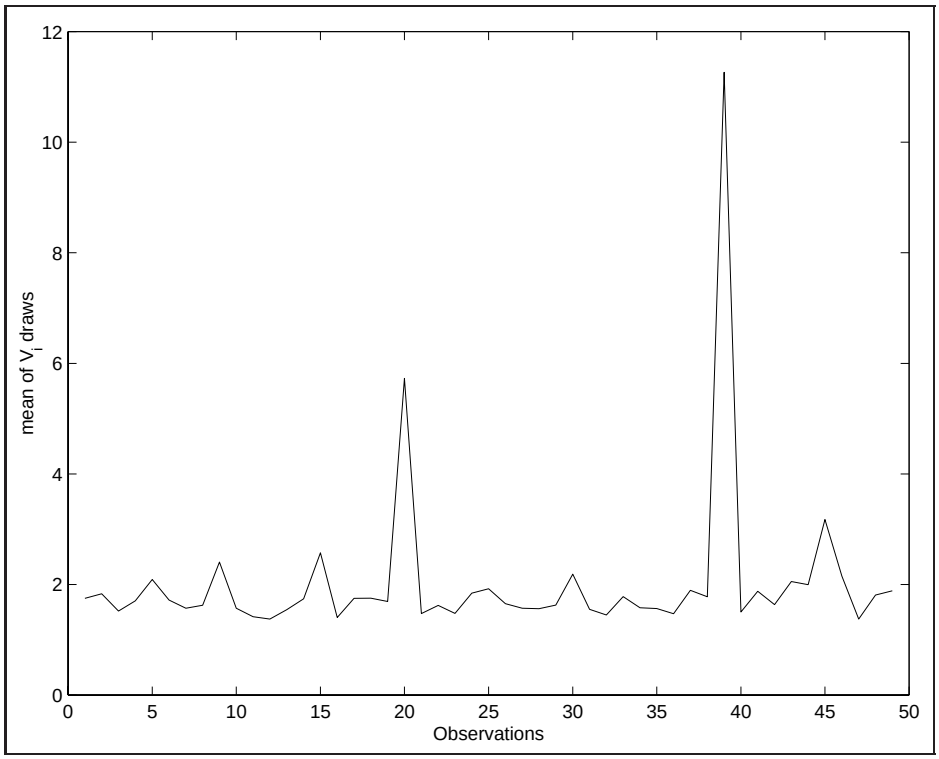

Figure 3.4: Mean of the v_i draws

if it exists. We rely on the input options to set a minimum and maximum value of ρ between 0 and 1 over which to search, to speed up computation. This avoids computation of the eigenvalues for the large matrix W which would provide the range over which to search. If you find an estimate for ρ near zero, this restriction to the (0,1) interval is unwise.

```
% ----- Example 3.4 Using far_g with a large data set
load elect.dat;             % load data on votes in 3,107 counties
y =  (elect(:,7)./elect(:,8));    % convert to per capita variables
ydev = y - mean(y);
clear elect;                % conserve on RAM memory
load ford.dat; % 1st order contiguity matrix stored in sparse matrix form
ii = ford(:,1); jj = ford(:,2); ss = ford(:,3);
n = 3107;
clear ford; % clear ford matrix to save RAM memory
W = sparse(ii,jj,ss,n,n);
clear ii; clear jj; clear ss; % conserve on RAM memory
prior.rval = 4; prior.rmin = 0; prior.rmax = 1;
```

```
ndraw = 1100; nomit = 100;
res = far_g(ydev,W,ndraw,nomit,prior);
prt(res);
plot(res.vmean);
xlabel('Observations');
ylabel('V_i estimates');
pause;
pltdens(res.pdraw,0.1,0,1);
```

We present maximum likelihood results for comparison with the Gibbs sampling results. If there is no substantial heterogeneity in the disturbance, the two sets of estimates should be similar, as we saw from example 3.3.

```
% Maximum likelihood results
First-order spatial autoregressive model Estimates
R-squared       =    0.5375
sigma^2         =    0.0054
Nobs, Nvars     =   3107,     1
log-likelihood  =        3506.3203
# of iterations =     13
min and max rho =   -1.0710,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.721474        59.567710         0.000000
% Gibbs sampling estimates
Gibbs sampling First-order spatial autoregressive model
R-squared       =    0.5337
sigma^2         =    0.0052
r-value         =      4
Nobs, Nvars     =   3107,     1
ndraws,nomit    =   1100,   100
acceptance rate =    0.7131
time in secs    =  262.4728
min and max rho =    0.0000,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
rho              0.706526        47.180554         0.000000
```

From the results we see that the maximum likelihood and Bayesian robust estimates are very similar, suggesting a lack of heterogeneity. We can further explore this issue by examining a plot of the mean v_i draws, which serve as estimates for these parameters in the model. Provided we use a small value of r, the presence of heterogeneity and outliers will be indicated by large v_i estimates that deviate substantially from unity. Figure 3.5 shows a plot of the mean of the v_i draws, confirming that a handful of large v_i values exist. Close inspection reveals that only 58 v_i values greater than 3 exist

in a sample of 3107 observations. Apparently this amount of heterogeneity does not affect the estimates for this model.

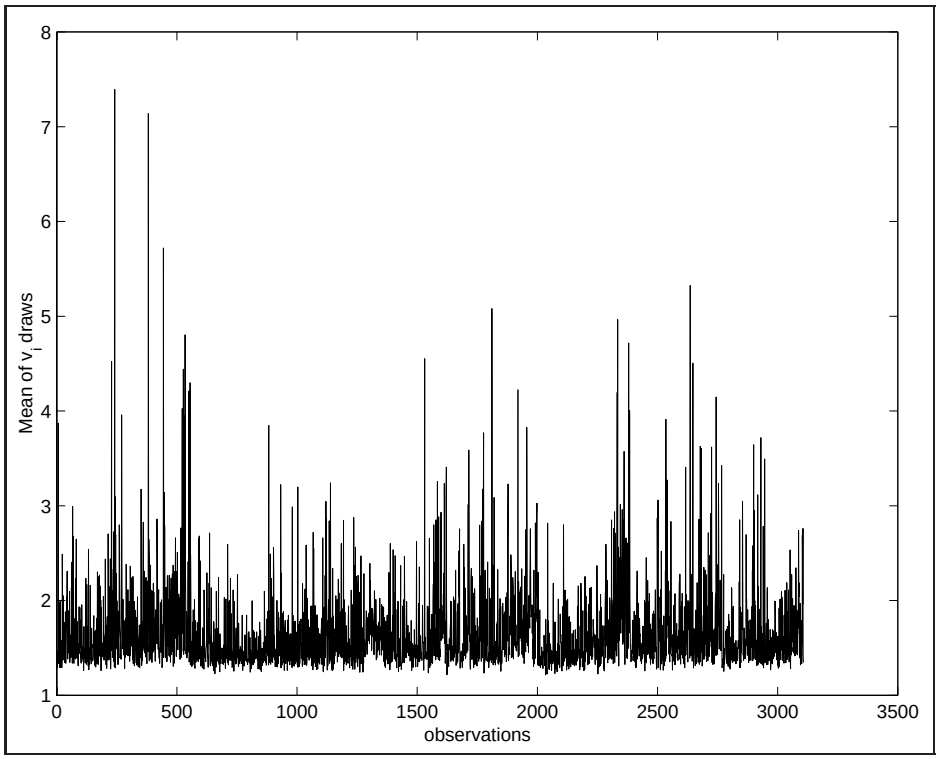

Figure 3.5: Mean of the v_i draws for Pace and Barry data

An advantage of Gibbs sampling is that valid estimates of dispersion for the parameters as well as the entire posterior distribution associated with the estimated parameters are available. Recall that this presents a problem for large data sets estimated using maximum likelihood methods, which we solved using a numerical hessian calculation. In the presence of outliers or non-constant variance the numerical hessian approach may not be valid because normality in the disturbance generating process might be violated. In the case of Gibbs sampling, the law of large numbers suggests that we can compute valid means and measures of dispersion from the sample of draws. As an illustration, we use a function **pltdens** from the *Econometrics Toolbox* to produce a non-parametric density estimate of the posterior distribution for ρ. Figure 3.6 shows the posterior density that is plotted using the command:

```
pltdens(res.pdraw,0.1,0,1);
```

An optional argument to the function allows a kernel smoothing parameter to be input as indicated in the documentation for the function **pltdens** shown below. The default kernel bandwidth produces fairly non-smooth densities that tend to overfit the data, so we supply our own value.

```
 PURPOSE: Draw a nonparametric density estimate.
-------------------------------------------------
 USAGE: [h f y] = pltdens(x,h,p,kernel)
        or pltdens(x) which uses gaussian kernel default
 where:
        x is a vector
        h is the kernel bandwidth
          default=1.06 * std(x) * n^(-1/5); Silverman page 45
        p is 1 if the density is 0 for negative values
        k is the kernel type:
          =1 Gaussian (default)
          =2 Epanechnikov
          =3 Biweight
          =4 Triangular
   A jittered plot of the
   observations is shown below the density.
-------------------------------------------------
 RETURNS:
        h = the interval used
        f = the density
        y = the domain of support
        plot(y,f) will produce a plot of the density
 -------------------------------------------------
```

The disadvantage of the Gibbs sampling estimation approach is the time required. This is reported in the printed output which indicates that it took 262 seconds to produce 1,100 draws. This is relatively competitive with the maximum likelihood estimation method that took around 100 seconds to produce estimates.

3.3 Other spatial autoregressive models

It should perhaps be clear that implementation of Gibbs samplers for the other spatial autoregressive models is quite straightforward. We need simply to determine the complete sequence of conditional distributions for the parameters in the model and code a loop to carry out the draws. LeSage (1997) sets forth an alternative approach to the Metropolis within Gibbs

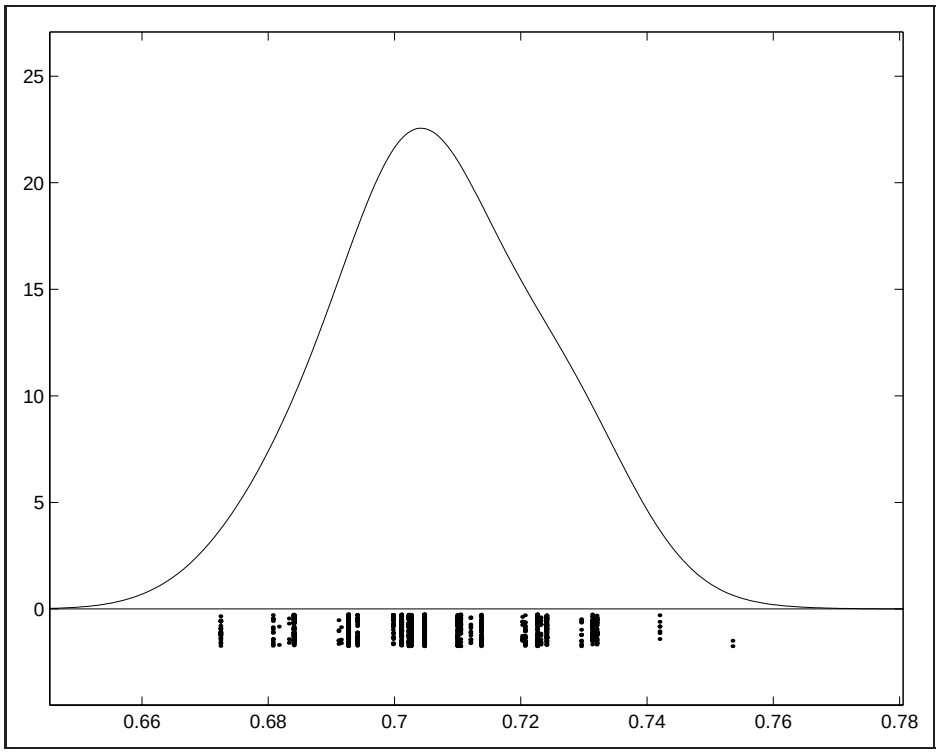

Figure 3.6: Posterior distribution for ρ

sampling in that article. There are many ways to generate samples from an unknown conditional distribution and the "ratio of uniforms" approach set forth in LeSage (1997) is another approach. My experience has convinced me that the Metropolis approach set forth here is superior as it requires far less time.

All of the spatial autoregressive models will have in common the need to produce a Metropolis-within Gibbs estimate for ρ based on a conditional distribution involving the determinant $(I_n - \rho W)$. In the case of the SAC model, we need two determinants, one for $(I_n - \rho W_1)$ and another for $(I_n - \lambda W_2)$. Of course we will carry this out initially over a grid of values and store the results. These will be passed to the functions that perform the conditional distribution calculations.

There are functions **sar_g**, **sem_g** and **sac_g** that implement Gibbs sampling estimation for the Bayesian variants of the spatial autoregressive models. The documentation for **sem_g** (which is similar to that for the other

models) is shown below:

```
 PURPOSE: Gibbs sampling estimates of the heteroscedastic
          spatial error model:
          y = XB + u,  u = lam*W + e
          e is N(0,sige*V)
          V = diag(v1,v2,...vn), r/vi = ID chi(r)/r, r = Gamma(m,k)
          B = N(c,T),  sige = gamma(nu,d0), lam = diffuse prior
-------------------------------------------------
 USAGE: results = sem_g(y,x,W,ndraw,nomit,prior,start)
 where: y = dependent variable vector (nobs x 1)
        x = independent variables matrix (nobs x nvar)
        W = 1st order contiguity matrix (standardized, row-sums = 1)
    prior = a structure for:  B = N(c,T),  sige = gamma(nu,d0)
            prior.beta, prior means for beta,   c above (default 0)
            prior.bcov, prior beta covariance , T above (default 1e+12)
            prior.rval, r prior hyperparameter, default=4
            prior.m,    informative Gamma(m,k) prior on r
            prior.k,    (default: not used)
            prior.nu,   a prior parameter for sige
            prior.d0,   (default: diffuse prior for sige)
            prior.lmin, (optional) min value of lambda to use in sampling
            prior.lmax, (optional) max value of lambda to use in sampling
    ndraw = # of draws
    nomit = # of initial draws omitted for burn-in
    start = (optional) structure containing starting values:
            defaults: beta=ones(k,1),sige=1,rho=0.5, V=ones(n,1)
            start.b   = beta starting values (nvar x 1)
            start.lam = lam starting value   (scalar)
            start.sig = sige starting value  (scalar)
            start.V   = V starting values (n x 1)
-------------------------------------------------
 RETURNS:  a structure:
          results.meth  = 'sem_g'
          results.bdraw = bhat draws (ndraw-nomit x nvar)
          results.pdraw = lam draws  (ndraw-nomit x 1)
          results.sdraw = sige draws (ndraw-nomit x 1)
          results.vmean = mean of vi draws (1 x nobs)
          results.rdraw = r draws (ndraw-nomit x 1) (if m,k input)
          results.bmean = b prior means, prior.beta from input
          results.bstd  = b prior std deviations sqrt(diag(prior.bcov))
          results.r     = value of hyperparameter r (if input)
          results.nobs  = # of observations
          results.nvar  = # of variables in x-matrix
          results.ndraw = # of draws
          results.nomit = # of initial draws omitted
          results.y     = actual observations (nobs x 1)
          results.yhat  = predicted values
          results.nu    = nu prior parameter
```

```
results.d0    = d0 prior parameter
results.time  = time taken for sampling
results.accept= acceptance rate
results.lmax = 1/max eigenvalue of W (or lmax if input)
results.lmin = 1/min eigenvalue of W (or lmin if input)
```

As the other functions are quite similar, we leave it to the reader to examine the documentation and demonstration files for these functions. One point to note regarding use of these functions is that two options exist for specifying a value for the hyperparameter r. The default is to rely on an improper prior based on $r = 4$. The other option allows a proper Gamma(m,k) prior to be assigned for r.

The first option has the virtue that convergence will be quicker and less draws are required to produce estimates. The drawback is that the estimates are conditional on the single value of r set for the hyperparameter. The second approach produces draws from a Gamma(m,k) distribution for r on each pass through the sampler. This produces estimates that average over alternative r values, in essence integrating over this parameter, resulting in unconditional estimates.

My experience is that estimates produced with a Gamma(8,2) prior having a mean of $r = 4$ and variance of 2 are quite similar to those based on an improper prior with $r = 4$. Use of the Gamma prior tends to require a larger number of draws based on convergence diagnostics routines implemented by the function **coda** from the *Econometrics Toolbox* described in Chapter 5 of the manual.

3.4 Applied examples

We turn attention to some applications involving the use of these models. First we present example 3.5 that generates SEM models for a set of λ parameters ranging from 0.1 to 0.9, based on the spatial weight matrix from the Columbus neighborhood crime data set. Both maximum likelihood and Gibbs estimates are produced by the program and a table is printed out to compare the estimation results. The hyperparameter r was set to 30 in this example, which should produce estimates similar to the maximum likelihood results.

During the loop over alternative data sets, we recover the estimates and other information we are interested in from the results structures. These are stored in a matrix that we print using the **mprint** function to add column and row labels.

```
% ----- Example 3.5 Using the sem_g function
load wmat.dat;    % standardized 1st-order contiguity matrix
load anselin.dat; % load Anselin (1988) Columbus neighborhood crime data
y = anselin(:,1);  n = length(y);
x = [ones(n,1) anselin(:,2:3)];
W = wmat; IN = eye(n);
vnames = strvcat('crime','const','income','house value');
tt = ones(n,1); tt(25:n,1) = [1:25]';
rvec = 0.1:.1:.9; b = ones(3,1);
nr = length(rvec); results = zeros(nr,6);
ndraw = 1100; nomit = 100; prior.rval = 30; bsave = zeros(nr,6);
for i=1:nr, rho = rvec(i);
u = (inv(IN-rho*W))*randn(n,1);
y =  x*b  + u;
% do maximum likelihood for comparison
resml = sem(y,x,W); prt(resml);
results(i,1) = resml.lam;
results(i,2) = resml.tstat(4,1);
bsave(i,1:3) = resml.beta';
% call Gibbs sampling function
result = sem_g(y,x,W,ndraw,nomit,prior); prt(result);
results(i,3) = mean(result.pdraw);
results(i,4) = results(i,3)/std(result.pdraw);
results(i,5) = result.time;
results(i,6) = result.accept;
bsave(i,4:6) = mean(result.bdraw);
end;
in.rnames = strvcat('True lam','0.1','0.2','0.3', ...
                    '0.4','0.5','0.6','0.7','0.8','0.9');
in.cnames = strvcat('ML lam','lam t','Gibbs lam','lam t', ...
                    'time','accept');
mprint(results,in);
in2.cnames = strvcat('b1 ML','b2 ML','b3 ML',...
                     'b1 Gibbs','b2 Gibbs','b3 Gibbs');
mprint(bsave,in2);
```

From the results we see that 1100 draws took around 13 seconds. The acceptance rate falls slightly for values of λ near 0.9, which we would expect. Since this is close to the upper limit of unity, we will see an increase in rejections of candidate values for λ that lie outside the feasible range.

The estimates are reasonably similar to the maximum likelihood results — even for the relatively small number of draws used. In addition to presenting estimates for λ, we also provide estimates for the parameters β in the problem.

```
% sem model demonstration
True lam   ML lam     lam t Gibbs lam     lam t      time    accept
```

```
0.1       -0.6357   -3.1334   -0.5049   -2.5119   12.9968    0.4475
0.2        0.0598    0.3057    0.0881    0.4296   12.8967    0.4898
0.3        0.3195    1.8975    0.3108    1.9403   13.0212    0.4855
0.4        0.2691    1.5397    0.2091    1.1509   12.9347    0.4827
0.5        0.5399    4.0460    0.5141    3.6204   13.1345    0.4770
0.6        0.7914   10.2477    0.7466    7.4634   13.3044    0.4616
0.7        0.5471    4.1417    0.5303    3.8507   13.2014    0.4827
0.8        0.7457    8.3707    0.7093    6.7513   13.5251    0.4609
0.9        0.8829   17.5062    0.8539   14.6300   13.7529    0.4349

      b1 ML      b2 ML      b3 ML   b1 Gibbs   b2 Gibbs   b3 Gibbs
     1.0570     1.0085     0.9988     1.0645     1.0112     0.9980
     1.2913     1.0100     1.0010     1.2684     1.0121     1.0005
     0.7910     1.0298     0.9948     0.7876     1.0310     0.9936
     1.5863     0.9343     1.0141     1.5941     0.9283     1.0157
     1.2081     0.9966     1.0065     1.2250     0.9980     1.0058
     0.3893     1.0005     1.0155     0.3529     1.0005     1.0171
     1.2487     0.9584     1.0021     1.3191     0.9544     1.0029
     1.8094     1.0319     0.9920     1.8366     1.0348     0.9918
    -0.9454     1.0164     0.9925    -0.9783     1.0158     0.9923
```

As another example, we use a generated data set where we insert 2 outliers. The program generates a vector y based on the Columbus neighborhood crime data set and then adjusts two of the generated y values for observations 10 and 40 to create outliers.

Example 3.6 produces maximum likelihood and two sets of Bayesian SAR model estimates. One Bayesian model uses a homoscedastic prior and the other sets $r = 4$, creating a a heteroscedastic prior. This is to illustrate that the differences in the parameter estimates are due to robustification, not the Gibbs sampling approach to estimation. A point to consider is that maximum likelihood estimates of precision based on the information matrix rely on normality, which is violated by the existence of outliers. These create a disturbance distribution that contains 'fatter tails' than the normal distribution, not unlike the $t-$distribution. In fact, this is the motivation for Geweke's approach to robustifying against outliers. Gibbs estimates based on a heteroscedastic prior don't rely on normality. If you find a difference between the estimates of precision from maximum likelihood estimates and Bayesian Gibbs estimates, it is a good indication that outliers may exist.

```
% ----- Example 3.6 An outlier example
load anselin.dat; load wmat.dat;
load anselin.dat; % load Anselin (1988) Columbus neighborhood crime data
x = [anselin(:,2:3)]; [n k] = size(x); x = [ones(n,1) x];
W = wmat; IN = eye(n);
rho = 0.5;        % true value of rho
```

```
b = ones(k+1,1);  % true value of beta
Winv = inv(IN-rho*W);
y = Winv*x*b + Winv*randn(n,1);
vnames = strvcat('y-simulated','constant','income','house value');
% insert outliers
y(10,1) = y(10,1)*2; y(40,1) = y(40,1)*2;
% do maximum likelihood for comparison
resml = sar(y,x,W);  prt(resml,vnames);
ndraw = 1100; nomit = 100;
prior.rval = 100; % homoscedastic model,
resg = sar_g(y,x,W,ndraw,nomit,prior);
prt(resg,vnames);
prior.rval = 4; % heteroscedastic model,
resg2 = sar_g(y,x,W,ndraw,nomit,prior);
prt(resg2,vnames);
% plot the vi-estimates
plot(resg2.vmean);
xlabel('Observations');
ylabel('mean of V_i draws');
```

The maximum likelihood SAR estimates along with the two sets of Bayesian model estimates are shown below. We see that the homoscedastic Gibbs estimates are similar to the maximum likelihood estimates, demonstrating that the Gibbs sampling estimation procedure is not responsible for the difference in estimates we see between maximum likelihood and the heteroscedastic Bayesian model. For the case of the heteroscedastic prior we see much better estimates for both β and ρ. Note that the $R-$squared statistic is lower for the robust estimates which will be the case because robustification requires that we not attempt to 'fit' the outlying observations. This will generally lead to a worse fit for models that produce robust estimates.

```
Spatial autoregressive Model Estimates
Dependent Variable =      y-simulated
R-squared      =    0.6779
Rbar-squared   =    0.6639
sigma^2        =  680.8004
Nobs, Nvars    =     49,     3
log-likelihood =       -212.59468
# of iterations =     14
min and max rho =   -1.5362,   1.0000
***************************************************************
Variable         Coefficient     t-statistic    t-probability
constant           19.062669        1.219474         0.228880
income             -0.279572       -0.364364         0.717256
house value         1.966962        8.214406         0.000000
```

```
rho                 0.200210         1.595514         0.117446

Gibbs sampling spatial autoregressive model
Dependent Variable =      y-simulated
R-squared       =    0.6770
sigma^2         = 1077.9188
r-value         =    100
Nobs, Nvars     =     49,     3
ndraws,nomit    =   1100,   100
acceptance rate =    0.9982
time in secs    =   28.9612
min and max rho =   -1.5362,   1.0000
*************************************************************
Variable          Prior Mean    Std Deviation
constant            0.000000   1000000.000000
income              0.000000   1000000.000000
house value         0.000000   1000000.000000
*************************************************************
      Posterior Estimates
Variable         Coefficient      t-statistic    t-probability
constant           18.496993         1.079346         0.286060
income             -0.123720        -0.142982         0.886929
house value         1.853332        10.940066         0.000000
rho                 0.219656         1.589213         0.118863

Gibbs sampling spatial autoregressive model
Dependent Variable =      y-simulated
R-squared       =    0.6050
sigma^2         = 1374.8421
r-value         =      3
Nobs, Nvars     =     49,     3
ndraws,nomit    =   1100,   100
acceptance rate =    0.9735
time in secs    =   17.2292
min and max rho =   -1.5362,   1.0000
*************************************************************
Variable          Prior Mean    Std Deviation
constant            0.000000   1000000.000000
income              0.000000   1000000.000000
house value         0.000000   1000000.000000
*************************************************************
      Posterior Estimates
Variable         Coefficient      t-statistic    t-probability
constant           13.673728         0.778067         0.440513
income              0.988163         0.846761         0.401512
house value         1.133790         3.976679         0.000245
rho                 0.388923         2.844965         0.006610
```

One point that may be of practical importance is that using large values

for the hyperparameter r slows down the Gibbs sampling process. This is because the chi-squared random draws take longer for large r values. This shows up in this example where the 1100 draws for the homoscedastic prior based on $r = 100$ took close to 29 seconds and those for the model based on $r = 4$ took only 17.2 seconds. This suggests that a good operational strategy would be not to rely on values of r greater than 30 or 40. These values may produce v_i estimates that deviate from unity somewhat, but should in most cases replicate the maximum likelihood estimates **when** there are no outliers.

A better strategy is to always rely on a small r value between 2 and 8, in which case a divergence between the maximum likelihood estimates and those from the Bayesian model reflect the existence of non-constant variance or outliers.

3.5 An applied exercise

We applied the series of spatial autoregressive models from Section 2.5 as well as corresponding Bayesian spatial autoregressive models to the Boston data set. Recall that Belsley, Kuh and Welsch (1980) used this data set to illustrate the impact of outliers and influential observations on least-squares estimation results. Here we have an opportunity to see the how the Bayesian spatial autoregressive models deal with the outliers.

Example 3.7 shows the program code needed to implement both maximum likelihood and Bayesian models for this data set.

```
% ----- Example 3.7 Robust Boston model estimation
load boston.raw; % Harrison-Rubinfeld data
load latitude.data; load longitude.data;
[W1 W W3] = xy2cont(latitude,longitude); % create W-matrix
[n k] = size(boston);y = boston(:,k);     % median house values
x = boston(:,1:k-1);                      % other variables
vnames = strvcat('hprice','crime','zoning','industry','charlesr', ...
         'noxsq','rooms2','houseage','distance','access','taxrate', ...
         'pupil/teacher','blackpop','lowclass');
ys = studentize(log(y)); xs = studentize(x);
rmin = 0; rmax = 1;
tic; res1 = sar(ys,xs,W,rmin,rmax); prt(res1,vnames); toc;
prior.rmin = 0; prior.rmax = 1;
prior.rval = 4;
ndraw = 1100; nomit=100;
tic; resg1 = sar_g(ys,xs,W,ndraw,nomit,prior);
prt(resg1,vnames); toc;
tic; res2 = sem(ys,xs,W,rmin,rmax); prt(res2,vnames); toc;
```

```
tic; resg2 = sem_g(ys,xs,W,ndraw,nomit,prior);
prt(resg2,vnames); toc;
tic; res3 = sac(ys,xs,W,W);         prt(res3,vnames); toc;
tic; resg3 = sac_g(ys,xs,W,W,ndraw,nomit,prior);
prt(resg3,vnames); toc;
```

An interesting aspect is the timing results which were produced using the MATLAB 'tic' and 'toc' commands. Maximum likelihood estimation of the SAR model took 44 seconds while Gibbs sampling using 1100 draws and omitting the first 100 took 124 seconds. For the SEM model the corresponding times were 59 and 164 seconds and for the SAC model 114 and 265 seconds. These times seem quite reasonable for this moderately sized problem.

The results are shown below. (We eliminated the printed output showing the prior means and standard deviations because all of the Bayesian models were implemented with diffuse priors for the parameters β in the model.) A prior value of $r = 4$ was used to produce robustification against outliers and non-constant variance.

What do we learn from this exercise? First, the parameters ρ and λ for the SAR and SEM models from maximum likelihood and Bayesian models are in agreement regarding both the magnitude and significance. For the SAC model, the Bayesian estimate for ρ is in agreement with the maximum likelihood estimate, but that for λ is not. The Bayesian estimate is 0.107 versus 0.188 for maximum likelihood. The maximum likelihood estimate is significant whereas the Bayesian estimate is not. This would impact on our decision regarding which model represents the best specification.

Most of the β estimates are remarkably similar with one notable exception, that for the 'noxsq' pollution variable. In all three Bayesian models this estimate is smaller than the maximum likelihood estimate and insignificant at the 95% level. All maximum likelihood estimates indicate significance. This would represent an important policy difference in the inference made regarding the impact of air pollution on housing values.

```
Spatial autoregressive Model Estimates  (elapsed_time = 44.2218)
Dependent Variable =    hprice
R-squared      =    0.8421
Rbar-squared   =    0.8383
sigma^2        =    0.1576
Nobs, Nvars    =    506,    13
log-likelihood =       -85.099051
# of iterations =      9
min and max rho =    0.0000,   1.0000
*************************************************************
```

```
Variable          Coefficient      t-statistic    t-probability
crime               -0.165349        -6.888522         0.000000
zoning               0.080662         3.009110         0.002754
industry             0.044302         1.255260         0.209979
charlesr             0.017156         0.918665         0.358720
noxsq               -0.129635        -3.433659         0.000646
rooms2               0.160858         6.547560         0.000000
houseage             0.018530         0.595675         0.551666
distance            -0.215249        -6.103520         0.000000
access               0.272237         5.625288         0.000000
taxrate             -0.221229        -4.165999         0.000037
pupil/teacher       -0.102405        -4.088484         0.000051
blackpop             0.077511         3.772044         0.000182
lowclass            -0.337633       -10.149809         0.000000
rho                  0.450871        12.348363         0.000000

Gibbs sampling spatial autoregressive model (elapsed_time = 126.4168)
Dependent Variable =    hprice
R-squared       =    0.8338
sigma^2         =    0.1812
r-value         =      4
Nobs, Nvars     =    506,    13
ndraws,nomit    =   1100,   100
acceptance rate =    0.9910
time in secs    =  110.2646
min and max rho =    0.0000,   1.0000
***************************************************************
      Posterior Estimates
Variable          Coefficient      t-statistic    t-probability
crime               -0.127092        -3.308035         0.001008
zoning               0.057234         1.467007         0.143012
industry             0.045240         0.950932         0.342105
charlesr             0.006076         0.249110         0.803379
noxsq               -0.071410        -1.512866         0.130954
rooms2               0.257551         5.794703         0.000000
houseage            -0.031992        -0.748441         0.454551
distance            -0.171671        -5.806800         0.000000
access               0.173901         2.600740         0.009582
taxrate             -0.202977        -5.364128         0.000000
pupil/teacher       -0.086710        -3.081886         0.002172
blackpop             0.094987         3.658802         0.000281
lowclass            -0.257394        -5.944543         0.000000
rho                  0.479126         5.697191         0.000000

Spatial error Model Estimates   (elapsed_time = 59.0996)
Dependent Variable =    hprice
R-squared       =    0.8708
Rbar-squared    =    0.8676
sigma^2         =    0.1290
```

```
log-likelihood  =       -58.604971
Nobs, Nvars     =    506,    13
# iterations    =     10
min and max lam =    0.0000,   1.0000
***************************************************************
Variable           Coefficient      t-statistic    t-probability
crime                -0.186710        -8.439402         0.000000
zoning                0.056418         1.820113         0.069348
industry             -0.000172        -0.003579         0.997146
charlesr             -0.014515        -0.678562         0.497734
noxsq                -0.220228        -3.683553         0.000255
rooms2                0.198585         8.325187         0.000000
houseage             -0.065056        -1.744224         0.081743
distance             -0.224595        -3.421361         0.000675
access                0.352244         5.448380         0.000000
taxrate              -0.257567        -4.527055         0.000008
pupil/teacher        -0.122363        -3.839952         0.000139
blackpop              0.129036         4.802657         0.000002
lowclass             -0.380295       -10.625978         0.000000
lambda                0.757669        19.133467         0.000000

Gibbs sampling spatial error model (elapsed_time = 164.8779)
Dependent Variable =    hprice
R-squared          =    0.7313
sigma^2            =    0.1442
r-value            =      4
Nobs, Nvars        =    506,    13
ndraws,nomit       =   1100,   100
acceptance rate    =    0.4715
time in secs       =  116.2418
min and max lambda =   -1.9826,   1.0000
***************************************************************
      Posterior Estimates
Variable           Coefficient      t-statistic    t-probability
crime                -0.165360        -3.967705         0.000083
zoning                0.048894         1.226830         0.220472
industry             -0.002985        -0.051465         0.958976
charlesr             -0.014862        -0.538184         0.590693
noxsq                -0.145616        -1.879196         0.060807
rooms2                0.339991         7.962844         0.000000
houseage             -0.130692        -2.765320         0.005900
distance             -0.175513        -2.398220         0.016846
access                0.276588         3.121642         0.001904
taxrate              -0.234511        -4.791976         0.000002
pupil/teacher        -0.085891        -2.899236         0.003908
blackpop              0.144119         4.773623         0.000002
lowclass             -0.241751        -5.931583         0.000000
lambda                0.788149        19.750640         0.000000
```

```
General Spatial Model Estimates  (elapsed_time = 114.5267)
Dependent Variable =    hprice
R-squared      =    0.8662
Rbar-squared   =    0.8630
sigma^2        =    0.1335
log-likelihood =       -55.200525
Nobs, Nvars    =    506,    13
# iterations   =      7
***************************************************************
Variable           Coefficient      t-statistic    t-probability
crime                -0.198184        -8.766862         0.000000
zoning                0.086579         2.824768         0.004923
industry              0.026961         0.585884         0.558222
charlesr             -0.004154        -0.194727         0.845687
noxsq                -0.184557        -3.322769         0.000958
rooms2                0.208631         8.573808         0.000000
houseage             -0.049980        -1.337513         0.181672
distance             -0.283474        -5.147088         0.000000
access                0.335479         5.502331         0.000000
taxrate              -0.257478        -4.533481         0.000007
pupil/teacher        -0.120775        -3.974717         0.000081
blackpop              0.126116         4.768082         0.000002
lowclass             -0.374514       -10.707764         0.000000
rho                   0.625963         9.519920         0.000000
lambda                0.188257         3.059010         0.002342

Gibbs sampling general spatial model (elapsed_time = 270.8657)
Dependent Variable =    hprice
R-squared          =    0.7836
sigma^2            =    0.1487
r-value            =      4
Nobs, Nvars        =    506,    13
ndraws,nomit       =   1100,   100
accept rho rate    =    0.8054
accept lam rate    =    0.9985
time in secs       =  205.6773
min and max rho    =    0.0000,   1.0000
min and max lambda =   -1.9826,   1.0000
***************************************************************
      Posterior Estimates
Variable           Coefficient      t-statistic    t-probability
crime                -0.161673        -3.905439         0.000107
zoning                0.047727         1.274708         0.203013
industry              0.024687         0.433283         0.664999
charlesr              0.008255         0.319987         0.749114
noxsq                -0.121782        -1.814226         0.070251
rooms2                0.333823         7.332684         0.000000
houseage             -0.098357        -2.080260         0.038018
distance             -0.193060        -3.798247         0.000164
```

access	0.227007	2.592018	0.009825
taxrate	-0.231393	-4.713901	0.000003
pupil/teacher	-0.110537	-3.852098	0.000133
blackpop	0.137065	4.555835	0.000007
lowclass	-0.293952	-7.201166	0.000000
rho	0.689346	7.938580	0.000000
lambda	0.107479	1.188044	0.235388

One other variable where we would draw a different inference is the 'houseage' variable in both the SEM and SAC models. The Bayesian estimates indicate significance for this variable whereas maximum likelihood estimates do not.

Another interesting difference between the Bayesian models and maximum likelihood is the lower R^2 statistics for the Bayesian versus corresponding maximum likelihood estimates. The Bayesian SEM and SAC models both show a dramatic reduction in fit indicative that a substantial amount of robustification is taking place. The Bayesian SAR model shows only a modest reduction in fit when compared to the maximum likelihood estimates. Recall that we rejected the SAR model in Chapter 2 for a number of reasons.

How do we decide between the maximum likelihood and Bayesian estimates? One issue we should explore is that of outliers and non-constant variance. A plot of the v_i estimates based on the mean of the draws from the Bayesian SAC model is shown in Figure 3.7. All three Bayesian models produced very similar estimates for the v_i terms as shown for the SAC model in Figure 3.7.

Given these estimates for the v_i terms, it would be hard to maintain the hypothesis of a constant variance normally distributed disturbance process for this model and data set.

Choosing between the Bayesian SEM and SAC models is not necessary as a similar set of inferences regarding the significance of the 'noxsq' air pollution variable and 'houseage' are produced by both models. Note that these are different inferences than one would draw from maximum likelihood estimates for either the SEM or SAC models.

If we wished to pursue this point we might examine the posterior distribution of λ from the SAC model, which is shown in Figure 3.8.

Suppose we felt comfortable imposing the restriction that λ must be positive? We can do this with the Bayesian SAC model, whereas this is not possible (given our optimization procedures) for the maximum likelihood estimation procedure. This can be accomplished using an input option to the **sac_g** function as shown in example 3.8. Note that we increased the

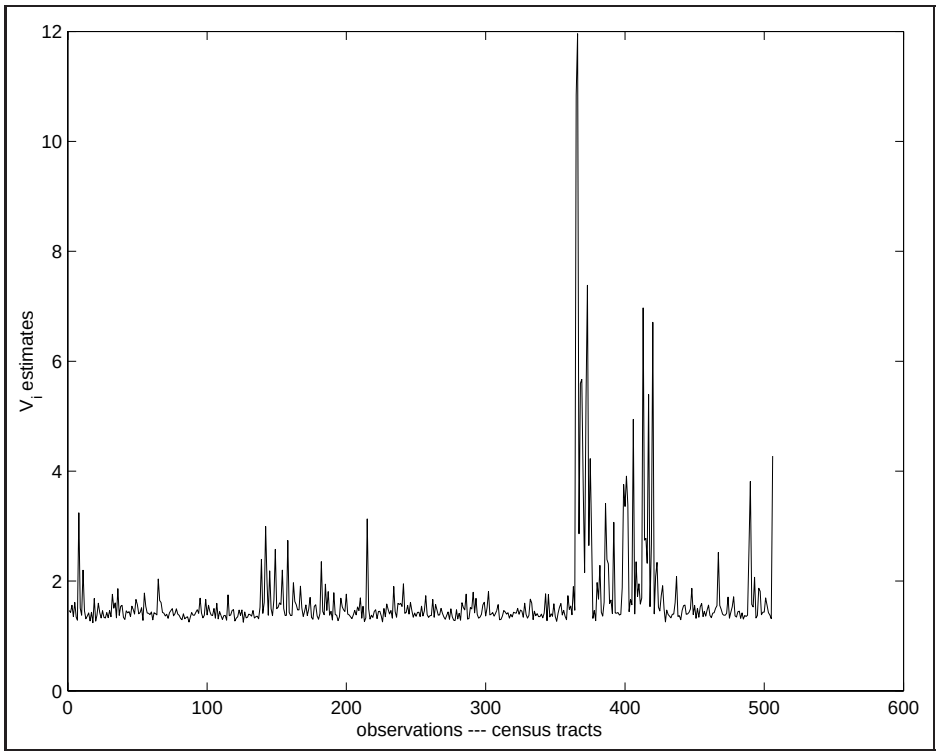

Figure 3.7: V_i estimates for the Boston data

number of draws to 2100 in example 3.8 to increase the precision regarding this model's estimates.

```
% ----- Example 3.8 Imposing restrictions
load boston.raw; % Harrison-Rubinfeld data
load latitude.data; load longitude.data;
[W1 W W3] = xy2cont(latitude,longitude); % create W-matrix
[n k] = size(boston);y = boston(:,k);     % median house values
x = boston(:,1:k-1);                      % other variables
vnames = strvcat('hprice','crime','zoning','industry','charlesr', ...
         'noxsq','rooms2','houseage','distance','access','taxrate', ...
         'pupil/teacher','blackpop','lowclass');
ys = studentize(log(y)); xs = studentize(x);
prior.rmin = 0; prior.rmax = 1;
prior.lmin = 0; prior.lmax = 1;
prior.rval = 4; ndraw = 2100; nomit=100;
resg3 = sac_g(ys,xs,W,W,ndraw,nomit,prior);
prt(resg3,vnames);
```

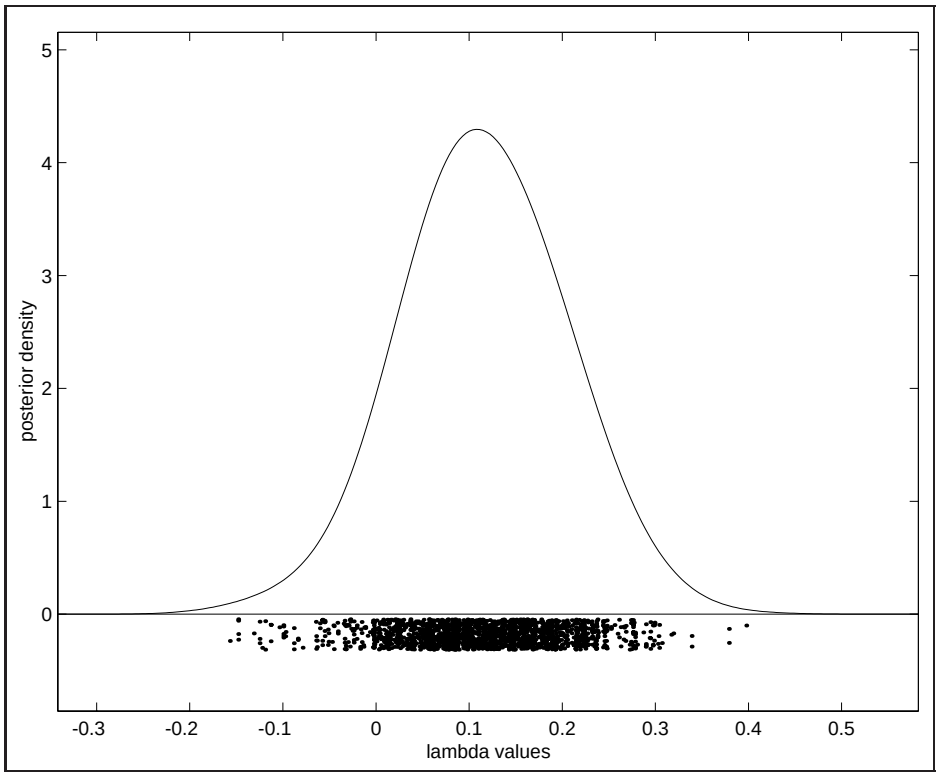

Figure 3.8: Posterior distribution of λ

Carrying out estimation of this version of the model, we found the following results. The estimate for the parameter λ is now perhaps different from zero as indicated by a $t-$statistic that is significant at the 0.10 level.

```
Gibbs sampling general spatial model
Dependent Variable =    hprice
R-squared          =    0.7891
sigma^2            =    0.1491
r-value            =      4
Nobs, Nvars        =    506,    13
ndraws,nomit       =   2100,   100
accept rho rate    =    0.9464
accept lam rate    =    0.6243
time in secs       =  403.2714
min and max rho    =   -1.9826,   1.0000
min and max lambda =    0.0000,   1.0000
*************************************************************
```

```
      Posterior Estimates
Variable          Coefficient      t-statistic    t-probability
crime               -0.164682       -4.096886         0.000049
zoning               0.048255        1.252200         0.211091
industry             0.018722        0.334302         0.738294
charlesr             0.009160        0.360944         0.718296
noxsq               -0.123344       -1.915048         0.056065
rooms2               0.328860        7.352177         0.000000
houseage            -0.096837       -2.045083         0.041377
distance            -0.192698       -3.829066         0.000145
access               0.232155        2.771251         0.005795
taxrate             -0.233451       -5.075241         0.000001
pupil/teacher       -0.109920       -3.730565         0.000213
blackpop             0.136174        4.462714         0.000010
lowclass            -0.297580       -7.136230         0.000000
rho                  0.671345        7.891604         0.000000
lambda               0.134279        1.700944         0.089584
```

A graphical examination of the posterior density for this parameter shown in Figure 3.9 might make us a bit concerned about the nature of the restriction we imposed on λ.

A virtue of Gibbs sampling is that we can compute the probability of $\lambda < 0$ by simply counting the Gibbs draws for this parameter that are less than zero in the unrestricted model. This can be done easily as shown in example 3.9. To improve the accuracy of our calculation, we increased the number of draws to 2100 in this program.

```
% ----- Example 3.9 The probability of negative lambda
load boston.raw; % Harrison-Rubinfeld data
load latitude.data; load longitude.data;
[W1 W W3] = xy2cont(latitude,longitude); % create W-matrix
[n k] = size(boston);y = boston(:,k);     % median house values
x = boston(:,1:k-1);                      % other variables
vnames = strvcat('hprice','crime','zoning','industry','charlesr', ...
         'noxsq','rooms2','houseage','distance','access','taxrate', ...
         'pupil/teacher','blackpop','lowclass');
ys = studentize(log(y)); xs = studentize(x);
prior.rval = 4; ndraw = 2100; nomit=100;
resg3 = sac_g(ys,xs,W,W,ndraw,nomit,prior);
prt(resg3,vnames);
% find the number of lambda draws < 0
nlam = find(resg3.ldraw < 0); numl = length(nlam);
fprintf(1,'The # of negative lambda values is: %5d \n',length(nlam));
fprintf(1,'Probability lambda < 0 is: %6.2f \n',numl/(ndraw-nomit));
```

The results indicate a probability of 6%, which should not make us nervous about imposing the restriction on λ.

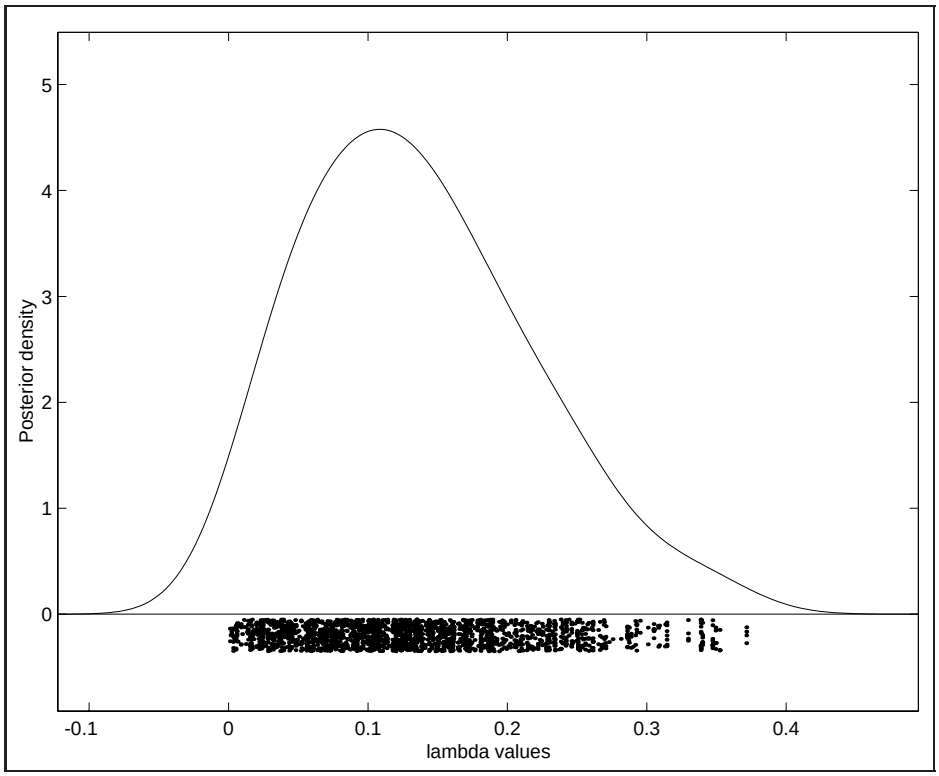

Figure 3.9: Truncated posterior distribution of λ

```
The # of negative lambda values is:   126
Probability lambda < 0 is:   0.06
```

Summarizing, I would use the Gibbs SAC model based on the restriction that $\lambda > 0$. With the exception of λ, this would produce the same inferences regarding all parameters as the model without this restriction. It would also produce the same inferences regarding the β parameters as the SEM model, which is comforting.

3.6 Chapter Summary

We have seen that spatial autoregressive models can be extended to allow for Bayesian prior information as well as non-constant variances over space. These models require a Gibbs sampling estimation approach, making them take more time than the maximum likelihood estimation methods. The time

is quite reasonable, even for large sample problems because we rely on the MATLAB sparse matrix algorithms and a grid-based approach to compute determinants that we use in the sampling process. In fact, as the sample size gets larger, the time difference between maximum likelihood and Gibbs sampling methods diminishes.

An advantage of these models is that they can serve as a check on the assumption of homogeneity that is inherent in the maximum likelihood models. It should be noted that some work has been done on accommodating non-constant variance in the case of maximum likelihood methods (see Anselin, 1988). Unfortunately, these approaches require that the investigator add a specification for the changing variance over space. This adds to the specification problems facing the practitioner, whereas the Bayesian approach set forth here requires no such specification. Outliers and non-constant variance are automatically detected during estimation and the estimates are adjusted for these problems.

We saw in an application from Section 3.5 an example where the existence of outliers produced different inferences from maximum likelihood and Bayesian robust estimates. These differences would have important policy implications for the conclusions one were to draw regarding the impact of air quality on housing values. These Bayesian models would produce different inferences than maximum likelihood regarding two of the explanatory variables 'noxsq' and 'houseage' and for the SAC model there is a difference regarding the significance of the parameter λ.

3.7 References

Anselin, L. 1988. *Spatial Econometrics: Methods and Models*, (Dorddrecht: Kluwer Academic Publishers).

Casella, G. and E.I. George. 1992. "Explaining the Gibbs Sampler", *American Statistician*, Vol. 46, pp. 167-174.

Gelfand, Alan E., and A.F.M Smith. 1990. "Sampling-Based Approaches to Calculating Marginal Densities", *Journal of the American Statistical Association*, Vol. 85, pp. 398-409.

Gelfand, Alan E., Susan E. Hills, Amy Racine-Poon and Adrian F.M. Smith. 1990. "Illustration of Bayesian Inference in Normal Data Models Using Gibbs Sampling", *Journal of the American Statistical Association*, Vol. 85, pp. 972-985.

Gelman, Andrew, John B. Carlin, Hal S. Stern, and Donald B. Rubin. 1995. *Bayesian Data Analysis*, (London: Chapman & Hall).

Geweke, John. 1993. "Bayesian Treatment of the Independent Student t Linear Model", *Journal of Applied Econometrics*, Vol. 8, s19-s40.

Gilks, W.R., S. Richardson and D.J. Spiegelhalter. 1996. *Markov Chain Monte Carlo in Practice*, (London: Chapman & Hall).

Lange, K.L., R.J.A. Little, and J.M.G. Taylor (1989). "Robust Statistical Modeling Using the t Distribution," *Journal of the American Statistical Association*, 84, pp. 881-896.

LeSage, James P. 1997. "Bayesian Estimation of Spatial Autoregressive Models", *International Regional Science Review*, 1997 Vol. 20, number 1&2, pp. 113-129. Also available at www.econ.utoledo.edu.

Lindley, David V. 1971. "The estimation of many parameters," in *Foundations of Statistical Science*, V.P. Godambe and D.A. Sprout (eds.) (Toronto: Holt, Rinehart, and Winston).

Theil, Henri and Arthur S. Goldberger. 1961. "On Pure and Mixed Statistical Estimation in Economics," *International Economic Review*, Vol. 2, pp. 65-78.

Zellner, Arnold. 1984. *Basic Issues in Econometrics*, (Chicago: University of Chicago Press), Chapter 3.

Chapter 4

Locally linear spatial models

This chapter discusses in detail a set of estimation methods that attempt to accommodate spatial heterogeneity by allowing the parameters of the model to vary with the spatial location of the sample data. The first section deals with spatial and distance expansion models introduced by Casetti (1972,1992). A more recent variant of this model presented in Casetti (1982) and Casetti and Can (1998) called a DARP model is the subject of Section 4.2.

Non-parametric locally linear regression models introduced in McMillen (1996), McMillen and McDonald (1997) and Brunsdon, Fotheringham and Charlton (1997) (sometimes labeled geographically weighted regression) represent another way to deal with spatial heterogeneity. These models are covered in Section 4.3. Finally, a Bayesian approach to geographically weighted regressions is presented in Section 4.4.

4.1 Spatial expansion

The first model of this type was introduced by Casetti (1972) and labeled a spatial expansion model. The model is shown in (4.1), where y denotes an nx1 dependent variable vector associated with spatial observations and X is an $nxnk$ matrix consisting of terms x_i representing kx1 explanatory variable vectors, as shown in (4.2). The locational information is recorded in the matrix Z which has elements $Z_{xi}, Z_{yi}, i = 1, \ldots, n$, that represent latitude and longitude coordinates of each observation as shown in (4.2).

The model posits that the parameters vary as a function of the latitude and longitude coordinates. The only parameters that need be estimated are the parameters in β_0 that we denote, β_x, β_y. These represent a set of

$2k$ parameters. Recall our discussion about spatial heterogeneity and the need to utilize a parsimonious specification for variation over space. This represents one approach to this type of specification.

We note that the parameter vector β in (4.1) represents an nkx1 matrix in this model that contains parameter estimates for all k explanatory variables at every observation. The parameter vector β_0 contains the $2k$ parameters to be estimated.

$$\begin{aligned} y &= X\beta + \varepsilon \\ \beta &= ZJ\beta_0 \end{aligned} \tag{4.1}$$

Where:

$$\begin{aligned} y &= \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad X = \begin{pmatrix} x_1' & 0 & \dots & 0 \\ 0 & x_2' & & \\ \vdots & & \ddots & \\ 0 & & & x_n' \end{pmatrix} \quad \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix} \\ Z &= \begin{pmatrix} Z_{x1} \otimes I_k & Z_{y1} \otimes I_k & 0 & \dots \\ 0 & \ddots & \ddots & \\ \vdots & & Z_{xn} \otimes I_k & Z_{yn} \otimes I_k \end{pmatrix} J = \begin{pmatrix} I_k & 0 \\ 0 & I_k \\ \vdots & \\ 0 & I_k \end{pmatrix} \\ \beta_0 &= \begin{pmatrix} \beta_x \\ \beta_y \end{pmatrix} \end{aligned} \tag{4.2}$$

This model can be estimated using least-squares to produce estimates of the $2k$ parameters β_x, β_y. Given these estimates, the remaining estimates for individual points in space can be derived using the second equation in (4.1). This process is referred to as the "expansion process". To see this, substitute the second equation in (4.1) into the first, producing:

$$y = XZJ\beta_0 + \varepsilon \tag{4.3}$$

Here it is clear that X, Z and J represent available information or data observations and only β_0 represent parameters in the model that need be estimated.

The model would capture spatial heterogeneity by allowing variation in the underlying relationship such that clusters of nearby or neighboring

observations measured by latitude-longitude coordinates take on similar parameter values. As the location varies, the regression relationship changes to accommodate a locally linear fit through clusters of observations in close proximity to one another.

Another way to implement this model is to rely on a vector of distances rather than the latitude-longitude coordinates. This implementation defines the distance from a central observation,

$$d_i = \sqrt{(Z_{xi} - Z_{xc})^2 + (Z_{yi} - Z_{yc})^2} \tag{4.4}$$

Where Z_{xc}, Z_{yc} denote the latitude-longitude coordinates of the centrally located observation and Z_{xi}, Z_{yi} denote the latitude-longitude coordinates for observations $i = 1, \ldots, n$ in the data sample.

This approach allows one to ascribe different weights to observations based on their distance from the central place origin. The formulation discussed above would result in a distance vector that increased with distance from the central observation. This would be suitable if one were modeling a phenomena reflecting a "hollowing out" of the central city or a decay of influence with distance from the central point.

The distance expansion model can be written as:

$$\begin{aligned} y &= X\beta + \varepsilon \\ \beta &= DJ\beta_0 \end{aligned} \tag{4.5}$$

Where $D = \text{diag}(d_1, d_2, \ldots, d_n)$, represents the distance of each observation from the central place and β_0 represents a kx1 vector of parameters for the central place. The matrix J in (4.5) is an nxk matrix, $J = (I_k, I_k, \ldots, I_k)'$.

4.1.1 Implementing spatial expansion

Estimating this model is relatively straightforward as we can rely on least-squares. One issue is that there are a number of alternative expansion specifications. For example, one approach would be to construct a model that includes the base k explanatory variables in the matrix X estimated with fixed parameters, plus an additional $2k$ expansion variables based on the latitude-longitude expansion. Another approach would be to include the base k variables in the matrix X and only $2(k-1)$ variables in expansion form by excluding the constant term from the expansion process. Yet another approach would be to rely on a simple expansion of all variables as was illustrated in (4.1).

The second approach was taken in implementing the MATLAB function **casetti** that carries out spatial expansion estimation. This choice was made because it seems unwise to include the constant term in the expansion as one can overfit the sample data when the intercept is allowed to very over space. A motivation for not relying on a simple expansion of all variables is that we would like our model to partition the influence of explanatory variables into fixed plus spatial effects. A simple expansion assigns all influence to spatial effects and also falls prey to the overfitting problem by allowing the intercept term to vary.

The expansion implemented by our function **casetti** can be written as:

$$y = \alpha + X\beta + XZ_x\beta_x + XZ_y\beta_y + \varepsilon \tag{4.6}$$

The function allows the user to specify an option for distance expansion based on a particular point in the spatial data sample or the latitude-longitude expansion. In the case of the distance expansion, the k explanatory variables in the matrix X are used as non-expansion variables estimated with fixed parameters and the $k-1$ variables excluding the constant are included as distance-expanded variables. This version of the model can be written as:

$$y = \alpha + X\beta + XD\beta_0 + \varepsilon \tag{4.7}$$

For the case of distance expansion, a distance vector is calculated as: $d_i = \sqrt{(Z_{xi} - Z_{xc})^2 + (Z_{yi} - Z_{yc})^2}$, where Z_{xc}, Z_{yc} denote the latitude-longitude coordinates of the centrally located observation and Z_{xi}, Z_{yi} denote the coordinates for observation i in the data sample. The distance of the central point is zero of course.

An optional input is provided to carry out isotropic normalization of the x-y coordinates which essentially puts the coordinates in deviations from the means form then standardizes by dividing by the square root of the sum of the variances in the x-y directions. That is:

$$\begin{aligned} x^\star &= (x - \bar{x})/\sqrt{(\sigma_x^2 + \sigma_y^2)} \\ y^\star &= (y - \bar{y})/\sqrt{(\sigma_x^2 + \sigma_y^2)} \end{aligned} \tag{4.8}$$

This normalization is carried out by a function **normxy** in the spatial econometrics library.

This normalization should make the center points xc, yc close to zero and produces a situation where the coefficients for the "base model" represent a

central observation. The distance-expanded estimates provide information about variation in the model parameters with reference to the central point.

The documentation for **casetti** is:

```
 PURPOSE: computes Casetti's spatial expansion regression
-------------------------------------------------
 USAGE: results = casetti(y,x,xc,yc,option)
 where:       y = dependent variable vector
              x = independent variables matrix
             xc = latitude (or longitude) coordinate
             yc = longitude (or latitude) coordinate
        option  = a structure variable containing options
        option.exp  = 0 for x-y expansion (default)
                    = 1 for distance from ctr expansion
        option.ctr  = central point observation # for distance expansion
        option.norm = 1 for isotropic x-y normalization (default=0)
-------------------------------------------------
 RETURNS:
        results.meth   = 'casetti'
        results.b0     = bhat (underlying b0x, b0y)
        results.t0     = t-stats (associated with b0x, b0y)
        results.beta   = spatially expanded estimates (nobs x nvar)
        results.yhat   = yhat
        results.resid  = residuals
        results.sige   = e'*e/(n-k)
        results.rsqr   = rsquared
        results.rbar   = rbar-squared
        results.nobs   = nobs
        results.nvar   = # of variables in x
        results.y      = y data vector
        results.xc     = xc
        results.yc     = yc
        results.ctr    = ctr (if input)
        results.dist   = distance vector (if ctr used)
        results.exp    = exp input option
        results.norm   = norm input option
 -------------------------------------------------
 NOTE: assumes x(:,1) contains a constant term
 -------------------------------------------------
```

Given the exclusion of the constant term from the spatial expansion formulation, we need to impose that the user place the constant term vector in the first column of the explanatory variables matrix X used as an input argument to the function.

Of course, we have an associated function to print the results structure and another to provide graphical presentation of the estimation results. Printing these estimation results is a bit challenging because of the large

number of parameter estimates that we produce using this method. Graphical presentation may provide a clearer picture of the variation in coefficients over space. A call to **plt** using the 'results' structure variable will produce plots of the coefficients in both the x- and y-directions for the case of the latitude-longitude expansion, where we sort the x-direction from left to right and the y-direction from left to right. This provides a visual picture of how the coefficients vary over space. If the x-coordinates are largest for the east and smallest for the west, the plot will show coefficient variation from west to east as in map space. Similarly, if the y-coordinates are smallest for the south and largest in the north, the plot will present coefficient variation from south to north. (Note that if you enter Western hemisphere latitude-longitude coordinates, the x-direction plots will be from east to west, but the y-direction plots will be south to north.)

For the case of distance expansion estimates, the plots present coefficients sorted by distance from the central point, provided by the user in the structure field 'option.ctr'. The central observation (smallest distance) will be on the left of the graph and the largest distance on the right.

Another point to note regarding the graphical presentation of the estimates relates to the fact that we present the coefficients in terms of the individual variables total impact on the dependent variable y. It was felt that users would usually be concerned with the total impact of a particular variable on the dependent variable as well as the decomposition of impacts into spatial and non-spatial effects. The printed output provides the coefficient estimates for the base model as well as the expansion coefficients that can be used to analyze the marginal effects from the spatial and non-spatial decomposition. To provide another view of the impact of the explanatory variables in the model on the dependent variable, the graphical presentation plots the coefficient estimates in a form representing their total impact on the dependent variable. That is we graph:

$$\begin{aligned} \gamma_{xi} &= \beta_i + Z_x\beta_{xi} \\ \gamma_{yi} &= \beta_i + Z_y\beta_{yi} \\ \gamma_{di} &= \beta_i + D\beta_{0i} \end{aligned} \tag{4.9}$$

Where γ_x, γ_y are plotted for the x-y expansion and γ_d is graphed for the distance expansion. This should provide a feel for the total impact of variable i on the dependent variable since it takes into account the non-spatial impact attributed to β_i, as well as the spatially varying impacts in the x-y direction or with respect to distance.

An illustration in the next section will pursue this point in more detail.

4.1.2 Applied examples

Example 4.1 provides an illustration of using the function **casetti** based on the Columbus neighborhood crime data set. Both types of expansion models are estimated by changing the structure variable 'option' field '.exp'. For the case of distance expansion, we rely on a central observation number 20 which lies near the center of the spatial sample of neighborhoods. One point to note is that the x-coordinate in Anselin's data set represents the south-north direction and the y-coordinate reflects the west-east direction.

```
% ----- example 4.1 Using the casetti() function
% load Anselin (1988) Columbus neighborhood crime data
load anselin.dat; y = anselin(:,1); n = length(y);
x = [ones(n,1) anselin(:,2:3)];
% Anselin (1988) x-y coordinates
xc0 = anselin(:,4); yc0 = anselin(:,5);
vnames = strvcat('crime','const','income','hse value');
% do Casetti regression using x-y expansion (default)
res1 = casetti(y,x,xc0,yc0);
prt(res1,vnames); % print the output
plt(res1,vnames); % graph the output
pause;
% do Casetti regression using distance expansion
option.exp = 1; option.ctr = 20; % Obs # of a central neighborhood
res2 = casetti(y,x,xc0,yc0,option);
prt(res2,vnames); % print the output
plt(res2,vnames); % graph the output
```

The default option is to implement an x-y expansion, which produces the result structure variable 'res1'. The next case relies on a structure variable 'option' to select distance expansion. The printed output is shown below. Both the base estimates as well as the expansion estimates are presented in the printed output. If you are working with a large model containing numerous observations, you can rely on the printing option that places the output in a file. Recall from Section 1.5, we need simply open an output file and input the 'file-id' as an option to the **prt** function.

Another point to note regarding the printed output is that in the case of a large number of explanatory variables, the printed estimates will 'wrap'. A set of estimates that take up 80 columns will be printed for all observations, and remaining estimates will be printed below for all observations. This 'wrapping' will continue until all of the parameter estimates are printed.

```
Casetti X-Y Spatial Expansion Estimates
Dependent Variable =        crime
R-squared     =    0.6330
Rbar-squared  =    0.5806
sige          =  117.4233
Nobs, Nvars   =     49,     3
***************************************************************
Base x-y estimates
Variable         Coefficient      t-statistic    t-probability
const              69.496160        15.105146         0.000000
income             -4.085918        -1.951941         0.057048
hse value           0.403956         0.517966         0.606965
x-income           -0.046062        -1.349658         0.183731
x-hse value         0.026732         2.027587         0.048419
y-income            0.121440         2.213107         0.031891
y-hse value        -0.048606        -2.341896         0.023571
***************************************************************
Expansion estimates
 Obs#    x-income x-hse value    y-income y-hse value
    1     -1.6407      0.9522      5.1466     -2.0599
    2     -1.6813      0.9757      4.9208     -1.9695
    3     -1.6909      0.9813      4.7009     -1.8815
    4     -1.5366      0.8918      4.6645     -1.8670
    5     -1.7872      1.0372      5.3519     -2.1421
    6     -1.8342      1.0645      5.0009     -2.0016
    7     -1.8429      1.0695      4.6147     -1.8470
    8     -2.0152      1.1695      4.7702     -1.9092
    9     -1.8245      1.0588      4.2395     -1.6968
   10     -2.1930      1.2727      4.4228     -1.7702
   11     -2.2377      1.2986      4.1848     -1.6750
   12     -2.2851      1.3262      3.9650     -1.5870
   13     -2.3082      1.3395      3.6323     -1.4538
   14     -2.3602      1.3697      3.3760     -1.3512
   15     -2.3441      1.3604      3.0651     -1.2268
   16     -2.2312      1.2949      3.3918     -1.3576
   17     -2.1525      1.2492      3.8752     -1.5510
   18     -2.0009      1.1612      4.3621     -1.7459
   19     -1.9977      1.1594      4.0634     -1.6264
   20     -1.8945      1.0995      4.0245     -1.6108
   21     -2.0244      1.1749      3.8387     -1.5364
   22     -2.0313      1.1789      3.6918     -1.4776
   23     -2.0129      1.1682      3.5436     -1.4183
   24     -1.8904      1.0971      3.4950     -1.3989
   25     -1.9913      1.1556      3.3165     -1.3274
   26     -1.9655      1.1406      3.0311     -1.2132
   27     -1.8982      1.1016      3.1453     -1.2589
   28     -1.8112      1.0511      3.1392     -1.2565
   29     -1.8927      1.0984      3.3384     -1.3362
   30     -1.7651      1.0244      3.4999     -1.4008
```

31	-1.9028	1.1043	3.7525	-1.5019
32	-1.8130	1.0522	3.9929	-1.5982
33	-1.8296	1.0618	3.7209	-1.4893
34	-1.7637	1.0236	3.6857	-1.4752
35	-1.6859	0.9784	3.8970	-1.5598
36	-1.7319	1.0051	4.1387	-1.6565
37	-1.7103	0.9926	4.3864	-1.7556
38	-1.7434	1.0118	4.4083	-1.7644
39	-1.6559	0.9610	4.4204	-1.7693
40	-1.6453	0.9549	4.3233	-1.7304
41	-1.6472	0.9559	4.2091	-1.6847
42	-1.6651	0.9664	4.1192	-1.6487
43	-1.5698	0.9110	3.6942	-1.4786
44	-1.3966	0.8105	3.4319	-1.3736
45	-1.2870	0.7469	3.6250	-1.4509
46	-1.2561	0.7290	3.4258	-1.3712
47	-1.1170	0.6482	3.2412	-1.2973
48	-1.1732	0.6809	3.1222	-1.2497
49	-1.3367	0.7758	3.2279	-1.2919

Casetti Distance Spatial Expansion Estimates
Dependent Variable = crime
R-squared = 0.6307
Rbar-squared = 0.5878
sige = 112.7770
Nobs, Nvars = 49, 3
central obs = 20

****

Base centroid estimates

Variable	Coefficient	t-statistic	t-probability
const	62.349645	12.794160	0.000000
income	-0.855052	-1.048703	0.299794
hse value	-0.138951	-0.520305	0.605346
d-income	-0.048056	-0.613545	0.542538
d-hse value	-0.013384	-0.473999	0.637743

****

Expansion estimates

Obs#	income	hse value
1	-0.5170	-0.1440
2	-0.4187	-0.1166
3	-0.3417	-0.0952
4	-0.4512	-0.1257
5	-0.5371	-0.1496
6	-0.3915	-0.1090
7	-0.2397	-0.0668
8	-0.3208	-0.0893
9	-0.1121	-0.0312
10	-0.3490	-0.0972
11	-0.3636	-0.1013

```
12     -0.4082     -0.1137
13     -0.4586     -0.1277
14     -0.5495     -0.1530
15     -0.6034     -0.1681
16     -0.4314     -0.1201
17     -0.2755     -0.0767
18     -0.1737     -0.0484
19     -0.1087     -0.0303
20     -0.0000     -0.0000
21     -0.1542     -0.0429
22     -0.1942     -0.0541
23     -0.2269     -0.0632
24     -0.2096     -0.0584
25     -0.2978     -0.0829
26     -0.4000     -0.1114
27     -0.3479     -0.0969
28     -0.3610     -0.1005
29     -0.2715     -0.0756
30     -0.2477     -0.0690
31     -0.1080     -0.0301
32     -0.0860     -0.0239
33     -0.1379     -0.0384
34     -0.1913     -0.0533
35     -0.2235     -0.0622
36     -0.1755     -0.0489
37     -0.2397     -0.0668
38     -0.2189     -0.0610
39     -0.2941     -0.0819
40     -0.2856     -0.0795
41     -0.2682     -0.0747
42     -0.2422     -0.0675
43     -0.3631     -0.1011
44     -0.5700     -0.1587
45     -0.6533     -0.1820
46     -0.7069     -0.1969
47     -0.8684     -0.2419
48     -0.8330     -0.2320
49     -0.6619     -0.1843
```

We turn attention to interpreting the output from this example. For this purpose we compare the 'base' spatial expansion estimates to those from least-squares which are presented below. The addition of the four x-y expansion variables increased the fit of the model slightly as indicated by the higher adjusted R^2 statistic. We see that the intercept estimate is relatively unaffected by the inclusion of expansion variables, but the coefficients on income and house value take on very different values. The significance of the income variable falls as indicated by the lower $t-$statistic, and the house

value variable becomes insignificant.

Three of the four x-y expansion variables are significant at the 0.05 level, providing evidence that the influence of these variables on neighborhood crime varies over space. Keep in mind that depending on the amount of inherent variation in the x-y coordinates, we may introduce a substantial amount of collinearity into the model when we add expansion variables. These are likely highly correlated with the base variables in the model, and this may account for the lack of significance of the house value variable in the 'base model'. A strict interpretation of these 'base' estimates would be that income expansion in the x-direction (south-north) is not significant, whereas it is in the y-direction (west-east).

```
Ordinary Least-squares Estimates
Dependent Variable =      crime
R-squared      =    0.5521
Rbar-squared   =    0.5327
sigma^2        =  130.8386
Durbin-Watson  =    1.1934
Nobs, Nvars    =     49,     3
***************************************************************
Variable         Coefficient      t-statistic    t-probability
const              68.609759        14.484270         0.000000
income             -1.596072        -4.776038         0.000019
house value        -0.274079        -2.655006         0.010858

Casetti X-Y Spatial Expansion Estimates
Dependent Variable =        crime
R-squared     =    0.6330
Rbar-squared  =    0.5806
sige          =  117.4233
Nobs, Nvars   =     49,     3
***************************************************************
Base x-y estimates
Variable         Coefficient      t-statistic    t-probability
const              69.496160        15.105146         0.000000
income             -4.085918        -1.951941         0.057048
hse value           0.403956         0.517966         0.606965
x-income           -0.046062        -1.349658         0.183731
x-hse value         0.026732         2.027587         0.048419
y-income            0.121440         2.213107         0.031891
y-hse value        -0.048606        -2.341896         0.023571

Casetti Distance Spatial Expansion Estimates
Dependent Variable =        crime
R-squared     =    0.6307
Rbar-squared  =    0.5878
sige          =  112.7770
```

```
Nobs, Nvars   =     49,     3
central obs   =     20
************************************************************
Base centroid estimates
Variable         Coefficient      t-statistic    t-probability
const              62.349645        12.794160         0.000000
income             -0.855052        -1.048703         0.299794
hse value          -0.138951        -0.520305         0.605346
d-income           -0.048056        -0.613545         0.542538
d-hse value        -0.013384        -0.473999         0.637743
```

In order to interpret the x-y expansion estimates, we need to keep in mind that the x-direction reflects the south-north direction with larger values of xc indicating northward movement. Similarly, larger values for yc reflect west-east movement. Using these interpretations, the base model estimates indicate that income exerts an insignificant negative influence on crime as we move from south to north. Considering the y-direction representing west-east movement, we find that income exerts a positive influence as we move in the easterly direction. One problem with interpreting the expansion estimates is that the base model coefficient is -4.085, indicating that income exerts a negative influence on crime. It is difficult to assess the total impact on neighborhood crime from both the base model coefficient representing non-spatial impact plus the small 0.121 value for the expanded coefficient reflecting the impact of spatial variation.

If we simply plotted the expansion coefficients, they would suggest that income in the y-direction has a positive influence on crime, a counterintuitive result. This is shown in the plot of the expanded coefficients sorted by the south-north and east-west directions shown in Figure 4.1. We are viewing a positive coefficient on the y-income variable in the graph.

Figure 4.2 shows a graph of the total impact of income on the dependent variable crime that takes into account both the base model non-spatial impact plus the spatial impact indicated by the expansion coefficient. Here we see that the total impact of income on crime is negative, except for neighborhoods in the extreme east at the right of the graph in the lower left-hand corner of Figure 4.2. The coefficient graphs produced using the **plt** function on the results structure from **casetti** are identical to those shown in Figure 4.2. You can of course recover the spatial expansion estimates from the results structure returned by **casetti**, sort the estimates in the x-y directions and produce your own plots of just the expansion estimates if this is of interest. As an example, the following code would produce this type of graph, where we are assuming the existence of a structure 'result' returned by **casetti**.

```
[xcs xci] = sort(result.xc);
[ycs yci] = sort(result.yc);
beta = result.beta;
[nobs nvar] = size(beta);
nvar = nvar/2;
betax =  beta(xci,1:nvar); % sort estimates
betay =  beta(yci,nvar+1:2*nvar);
tt=1:nobs;
for j=1:nvar
plot(tt,betax(:,j)); pause;
end;
for j=1:nvar
plot(tt,betay(:,j)); pause;
end;
```

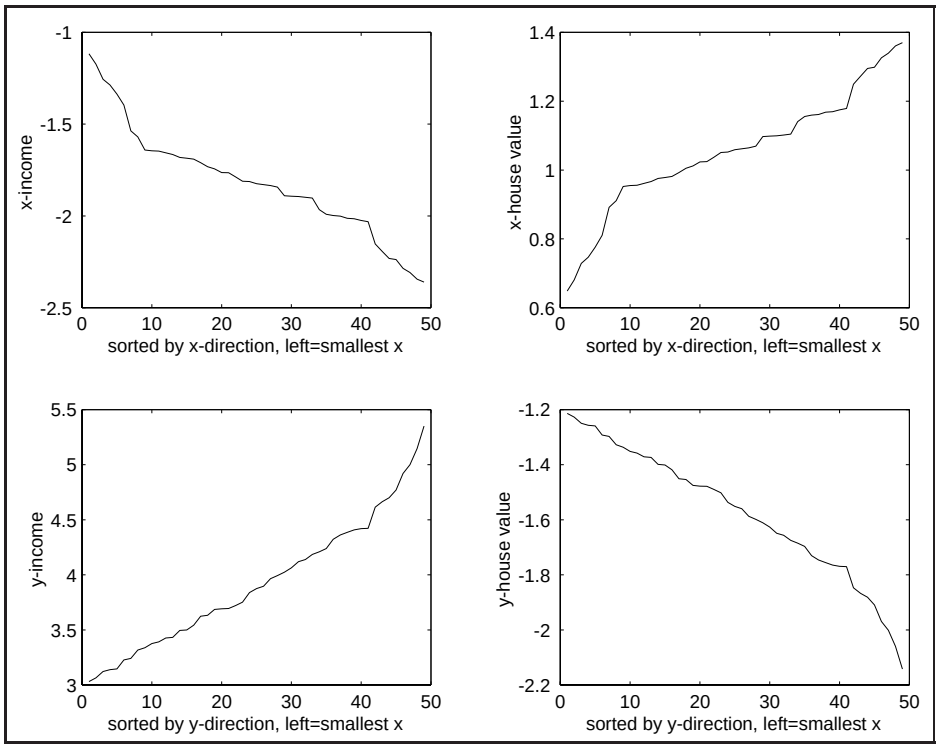

Figure 4.1: Spatial x-y expansion estimates

The distance expansion method produced a similar fit to the data as indicated by the adjusted R^2 statistic. This is true despite the fact that only the constant term is statistically significant at conventional levels.

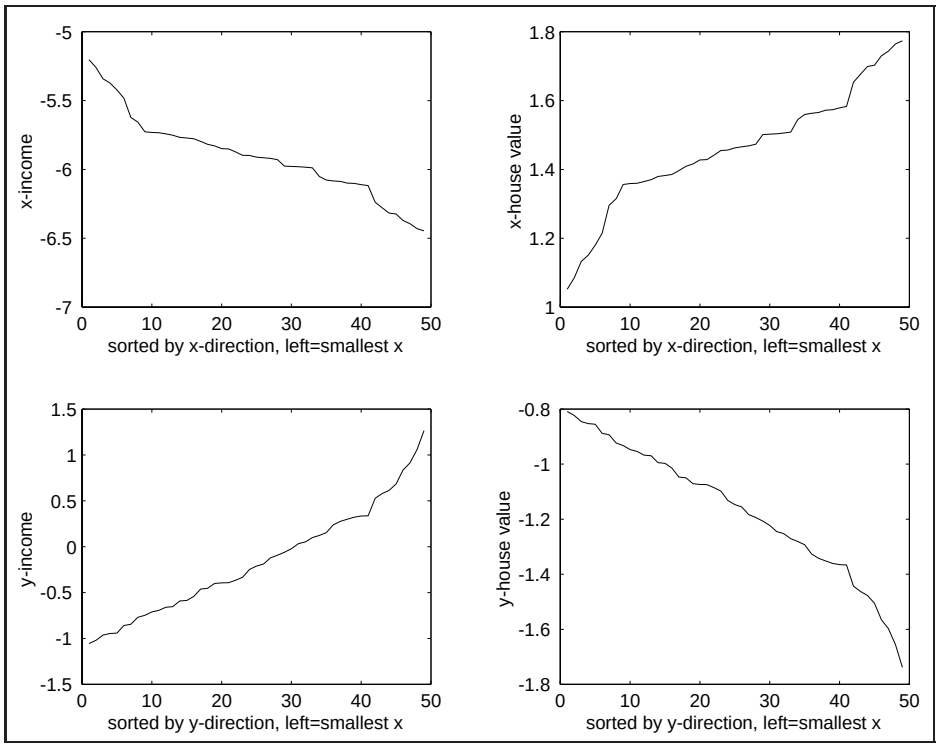

Figure 4.2: Spatial x-y total impact estimates

These estimates take on a value of zero for the central observation since the distance is zero at that point. This makes them somewhat easier to interpret than the estimates for the case of x-y coordinates. We know that the expansion coefficients take on values of zero at the central point, producing estimates based on the non-spatial base coefficients for this point. As we move outward from the center, the expansion estimates take over and adjust the constant coefficients to account for variation over space. The printed distance expansion estimates reported for the base model reflect values near the distance-weighted average of all points in space. Given this, we would interpret the coefficients for the model at the central point to be (62.349, -0.855, -0.138) for the intercept, income and house value variables respectively. In Figure 4.3 we see the total impact of income and house values on neighborhood crime as we move away from the central point. Both income and house values have a negative effect on neighborhood crime as we move away from the central city. Note that this is quite different from

the pattern shown for the x-y expansion. Anselin (1988) in analyzing the x-y model shows that heteroscedastic disturbances produce problems that plague inferences for the model. Adjusting for the heteroscedastic disturbances dramatically alters the inferences. We turn attention to this issue when we discuss the DARP version of this model in Section 4.2.

The plots of the coefficient estimates provide an important source of information about the nature of coefficient variation over space, but you should keep in mind that they do not indicate levels of significance, simply point estimates.

Our **plt** wrapper function works to call the appropriate function **plt_cas** that provides individual graphs of each coefficient in the model as well as a two-part graph showing actual versus predicted and residuals. Figure 4.4 shows the actual versus predicted and residuals from the distance expansion model. This plot is produced by the **plt** function when given a results structure from the **casetti** function.

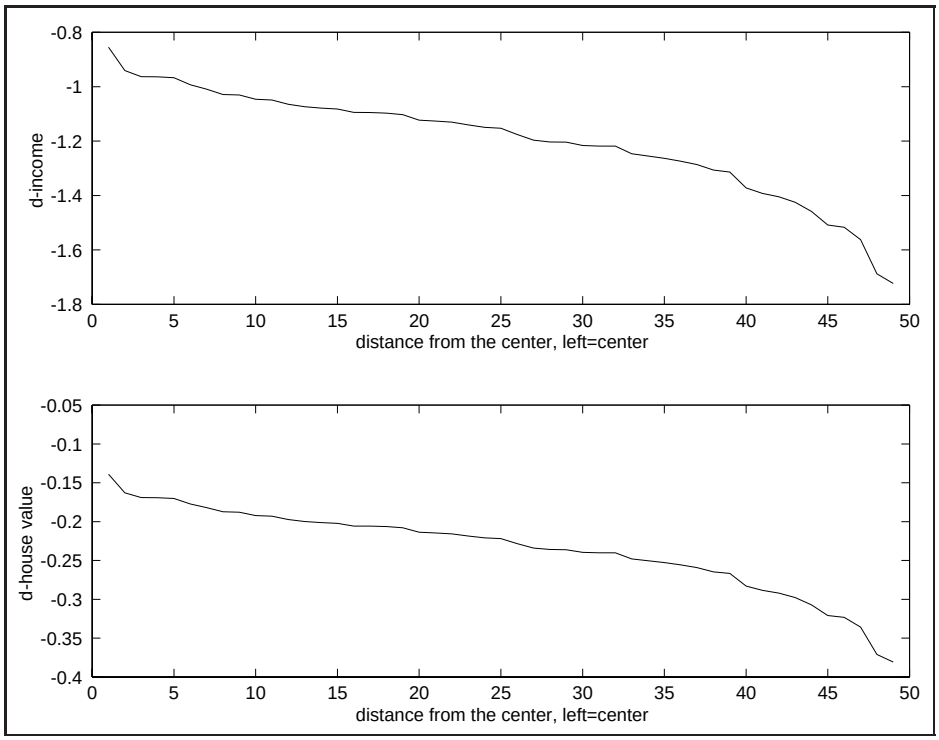

Figure 4.3: Distance expansion estimates

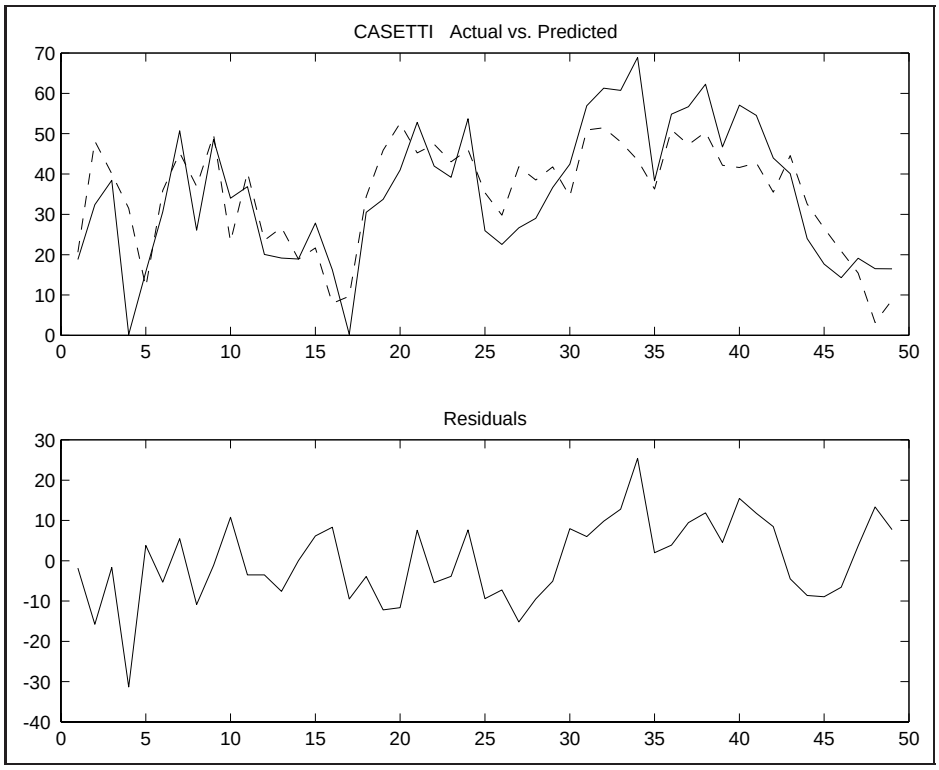

Figure 4.4: Actual versus Predicted and residuals

4.2 DARP models

A problem with the spatial expansion model is that heteroscedasticity is inherent in the way the model is constructed. To see this, consider the slightly altered version of the distance expansion model shown in (4.10), where we have added a stochastic term u to reflect some error in the expansion relationship.

$$\begin{aligned} y &= X\beta + e \\ \beta &= DJ\beta_0 + u \end{aligned} \tag{4.10}$$

Now consider substituting the second equation from (4.10) into the first, producing:

$$y = XDJ\beta_0 + Xu + e \tag{4.11}$$

It should be clear that the new composite disturbance term $Xu+e$ will reflect heteroscedasticity unless the expansion relationship is exact and $u = 0$.

Casetti (1982) and Casetti and Can (1998) propose a model they label DARP, an acronym for Drift Analysis of Regression Parameters, that aims at solving this problem. This model case be viewed as an extended expansion model taking the form:

$$\begin{aligned} y &= X\beta + e \\ \beta &= f(Z, \rho) + u \end{aligned} \tag{4.12}$$

Where $f(Z, \rho)$ represents the expansion relationship based on a function f, variables Z and parameters ρ. Estimation of this model attempts to take into account that the expanded model will have a heteroscedastic error as shown in (4.11).

To keep our discussion concrete, we will rely on $f(Z, \rho) = DJ\beta_0$, the distance expansion relationship in discussing this model. In order to take account of the spatial heteroscedasticity an explicit model of the composite disturbance term: $\varepsilon = Xu + e$, is incorporated during estimation. This disturbance is assumed to have a variance structure that can be represented in alternative ways shown below. We will rely on these alternative scalar and matrix representations in the mathematical development and explanation of the model.

$$\begin{aligned} E(\varepsilon\varepsilon') &= \Phi = \sigma^2\Psi \\ \Psi &= \exp(\text{diag}(\gamma d_1, \gamma d_2, \ldots, \gamma d_n)) \\ \Phi &= \text{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2) \\ \sigma_i^2 &= \exp(\gamma_0 + \gamma_i d_i) \end{aligned} \tag{4.13}$$

Where d_i denotes the squared distance between the ith observation and the central point, and $\sigma^2, \gamma_0, \gamma_1$ are parameters to be estimated. Of course, a more general statement of the model would be that $\sigma_i^2 = g(h_i, \gamma)$, indicating that any functional form g involving some known variable h_i and associated parameters γ could be employed to specify the non-constant variance over space.

Note that the alternative specifications in (4.13) imply: $\sigma^2 = \exp(\gamma_0)$, the constant scalar component of the variance structure, and $\exp(\gamma_1 d_i)$ reflects the non-constant component which is modeled as a function of distance from the central point.

An intuitive motivation for this type of variance structure is based on considering the nature of the composite disturbance: $Xu + e$. The constant scalar σ^2 reflects the constant component e, while the role of the scalar parameter γ_1 associated with distance is to measure the impact of Xu, the non-constant variance component on average. Somewhat loosely, consider a linear regression involving the residuals on a constant term plus a vector of distances from the central place. The constant term estimate would reflect $\sigma^2 = \exp(\gamma_0)$, while the distance coefficient is intended to capture the influence of the non-constant Xu component: $\Psi = \exp(\gamma_1 d)$.

If $\gamma_1 = 0$, we have that $\Psi = \sigma^2 I_n$, a constant scalar value across all observations in space. This would be indicative of a situation where u, the error made in the expansion specification is small. This homoscedastic case would be indicative that a simple deterministic spatial expansion specification for spatial coefficient variation is performing well.

On the other hand, if $\gamma_1 > 0$, we find that moving away from the central point produces a positive increase in the variance. This is interpreted as evidence that 'parameter drift' is present in the relationship being modeled. The motivation is that increasing variance would be indicative of larger errors (u) in the expansion relationship as we move away from the central point.

Note that one can assume a value for γ_1 rather than estimate this parameter. If you impose a positive value, you are assuming a DARP model that will generate locally linear estimates since movement in the parameters will increase with movement away from the central point. This is because allowing increasing variance in the stochastic component of the expansion relation brings about more rapid change or adjustment in the parameters. Another way to view this is that the change in parameters need not adhere as strictly to the deterministic expansion specification. We will argue in Section 4.4 that a Bayesian approach to this type of specification is more intuitively appealing.

Negative values for γ_1 suggest that the errors made by the deterministic expansion specification are smaller as we move away from the central point. This indicates that the expansion relation works well for points farther from the center, but not as well for the central area observations. Casetti and Can (1998) interpret this as suggesting "differential performance" of the base model with movement over space, and they label this phenomena as 'performance drift'. The intuition here is most likely based on the fact that the expansion relationship is of a locally linear nature. Given this, better performance with distance is indicative of a need to change the deterministic expansion relationship to improve performance. Again, I will argue that a

Bayesian model represents a more intuitively appealing way to deal with these issues in Section 4.4.

Estimation of the parameters of the model require either feasible generalized least squares (FGLS) or maximum likelihood (ML) methods. Feasible generalized least squares obtains a consistent estimate of the unknown parameter γ_1 and then proceeds to estimate the remaining parameters in the model conditional on this estimate.

As an example, consider using least-squares to estimate the expansion model and associated residuals, $\hat{e} = y - XDJ\beta_0$. We could then carry out a regression of the form:

$$log(\hat{e}^2) = \gamma_0 + \gamma_1 d + \nu \tag{4.14}$$

Casetti and Can (1998) argue that the estimate $\hat{\gamma}_1$ from this procedure would be consistent. Given this estimate and our knowledge of the distances in the vector d, we construct and estimate of Ψ that we label $\hat{\Psi}$. Using $\hat{\Psi}$, generalized least-squares can produce:

$$\begin{aligned} \hat{\beta}_{FGLS} &= (X'\hat{\Psi}^1 X)^{-1} X'\hat{\Psi}^{-1} y \\ \hat{\sigma}^2_{FGLS} &= (y - X\hat{\beta}_{FGLS})\hat{\Psi}^{-1}(y - X\hat{\beta}_{FGLS})/(n-k) \end{aligned}$$

Of course, the usual GLS variance-covariance matrix for the estimates applies:

$$var - cov(\hat{\beta}_{FGLS}) = \hat{\sigma}^2 (X'\hat{\Psi}^{-1} X)^{-1} \tag{4.15}$$

Casetti and Can (1998) also suggest using a statistic: $\hat{\gamma}_1^2 / 4.9348 \sum d_i$ which is chi-squared distributed with one degree of freedom to test the null hypothesis that $\gamma_1 = 0$.

Maximum likelihood estimation involves using optimization routines to solve for a minimum of the negative of the log-likelihood function. We have already seen how to solve optimization problems in the context of spatial autoregressive models in Chapter 2. We will take the same approach for this model. The log-likelihood is:

$$L(\beta, \gamma_1 | .) = c - (1/2) ln|\sigma^2 \Psi| - (1/2)(y - X\beta)' \Psi^{-1} (y - X\beta) \tag{4.16}$$

As in Chapter 2 we can construct a MATLAB function to evaluate the negative of this log-likelihood function and rely on our function **maxlik**. The asymptotic variance-covariance matrix for the estimates β is equal to

that for the FGLS estimates shown in (4.15). The asymptotic variance-covariance matrix for the parameters (σ^2, γ_1) is given by:

$$\begin{aligned} var - cov(\sigma^2, \gamma_1) &= 2(D'D)^{-1} \\ D &= (\iota, d) \end{aligned} \tag{4.17}$$

In the case of maximum likelihood estimates, a Wald statistic based on $\hat{\gamma}_1^2/2\sum d_i$ that has a chi-squared distribution with one degree of freedom can be used to test the null hypothesis that $\gamma_1 = 0$. Note that the maximum likelihood estimate of γ_1 is more efficient than the FGLS estimate. This can be seen by comparing the ML estimate's asymptotic variance of $2\sum d_i$, to that for the FGLS which equals $4.9348\sum d_i$. Bear in mind, tests regarding the parameter γ_1 are quite often the focus of this methodology as it provides exploratory evidence regarding 'performance drift' versus 'parameter drift', so increased precision regarding this parameter may be important.

The function **darp** implements this estimation procedure using the **maxlik** function from the *Econometrics Toolbox* to find estimates via maximum likelihood. The documentation for **darp** is:

```
 PURPOSE: computes Casetti's DARP model
---------------------------------------------------
 USAGE: results = darp(y,x,xc,yc,option)
 where:       y = dependent variable vector
              x = independent variables matrix
             xc = latitude (or longitude) coordinate
             yc = longitude (or latitude) coordinate
        option  = a structure variable containing options
        option.exp  = 0 for x-y expansion (default)
                    = 1 for distance from ctr expansion
        option.ctr  = central point observation # for distance expansion
        option.iter = # of iterations for maximum likelihood routine
        option.norm = 1 for isotropic x-y normalization (default=0)
---------------------------------------------------
 RETURNS:
        results.meth   = 'darp'
        results.b0     = bhat (underlying b0x, b0y)
        results.t0     = t-stats (associated with b0x, b0y)
        results.beta   = spatially expanded estimates (nobs x nvar)
        results.yhat   = yhat
        results.resid  = residuals
        results.sige   = e'*e/(n-k)
        results.rsqr   = rsquared
        results.rbar   = rbar-squared
        results.nobs   = nobs
```

```
        results.nvar   = # of variables in x
        results.y      = y data vector
        results.xc     = xc
        results.yc     = yc
        results.ctr    = ctr (if input)
        results.dist   = distance vector
        results.exp    = exp input option
        results.norm   = norm input option
        results.iter   = # of maximum likelihood iterations
 -------------------------------------------------
 NOTE: assumes x(:,1) contains a constant term
 -------------------------------------------------
```

Because we are relying on maximum likelihood estimation which may not converge, we provide FGLS estimates as output in the event of failure. A message is printed to the MATLAB command window indicating that this has occurred. We rely on the FGLS estimates to provide starting values for the **maxlik** routine, which should speed up the optimization process.

The DARP model can be invoked with either the x-y or distance expansion as in the case of the spatial expansion model. Specifically, for x-y expansion the variance specification is based on:

$$log(\hat{e}^2) = \gamma_0 + \gamma_1 xc + \gamma_2 yc + \nu \tag{4.18}$$

This generalizes the distance expansion approach presented previously.

Of course, we have an accompanying **prt** and **plt** function to provide printed and graphical presentation of the estimation results.

Example 4.2 shows how to use the function **darp** for both x-y and distance expansion using the Columbus neighborhood crime data set.

```
% ----- example 4.2 Using the darp() function
% load Anselin (1988) Columbus neighborhood crime data
load anselin.dat; y = anselin(:,1); n = length(y);
x = [ones(n,1) anselin(:,2:3)];
xc = anselin(:,4); yc = anselin(:,5); % Anselin  x-y coordinates
vnames = strvcat('crime','const','income','hse value');
% do Casetti darp using x-y expansion
res1 = darp(y,x,xc,yc);
prt(res1,vnames); % print the output
plt(res1,vnames); % plot the output
pause;
% do Casetti darp using distance expansion from observation #20
option.exp = 1; option.ctr = 20;
res2 = darp(y,x,xc,yc,option);
prt(res2,vnames); % print the output
plt(res2,vnames); % plot the output
```

The printed results are shown below, where we report not only the estimates for β_0 and the expansion estimates, but estimates for the parameters γ as well. A chi-squared statistic to test the null hypothesis that $\gamma_1 = 0$ is provided as well as a marginal probability level. For the case of the x-y expansion, we see that γ_1 parameter is negative and significant by virtue of the large chi-squared statistic and associated marginal probability level of 0.0121. The inference we would draw is that performance drift occurs in the south-north direction. For the γ_2 parameter, we find a positive value that is not significantly different from zero because of the marginal probability level of 0.8974. This indicates that the simple deterministic expansion relationship is working well in the west-east direction. Note that these results conform to those found with the spatial expansion model, where we indicated that parameter variation in the west-east direction was significant, but not for the south-north.

```
DARP X-Y Spatial Expansion Estimates
Dependent Variable =        crime
R-squared       =    0.6180
Rbar-squared    =    0.5634
sige            =  122.2255
gamma1,gamma2   =   -0.0807,   0.0046
gamma1, prob    =    6.2924,   0.0121
gamma2, prob    =    0.0166,   0.8974
# of iterations =     16
log-likelihood  =        -181.3901
Nobs, Nvars   =     49,     3
***************************************************************
Base x-y estimates
Variable         Coefficient      t-statistic    t-probability
const              66.783527         6.024676         0.000000
income             -2.639184        -0.399136         0.691640
hse value           0.249214         0.095822         0.924078
x-income           -0.048337        -0.537889         0.593247
x-hse value         0.021506         0.640820         0.524819
y-income            0.084877         0.564810         0.574947
y-hse value        -0.037460        -0.619817         0.538436
***************************************************************
Expansion estimates
 Obs#    x-income x-hse value    y-income y-hse value
    1     -1.3454      0.6595      4.0747     -1.6828
    2     -1.3786      0.6758      3.8959     -1.6089
    3     -1.3865      0.6796      3.7218     -1.5370
    4     -1.2600      0.6176      3.6930     -1.5251
    5     -1.4655      0.7183      4.2372     -1.7499
    6     -1.5040      0.7372      3.9593     -1.6351
    7     -1.5112      0.7407      3.6536     -1.5088
```

```
   8     -1.6524      0.8100      3.7766     -1.5597
   9     -1.4961      0.7333      3.3565     -1.3862
  10     -1.7982      0.8814      3.5017     -1.4461
  11     -1.8349      0.8994      3.3132     -1.3683
  12     -1.8738      0.9185      3.1392     -1.2964
  13     -1.8926      0.9277      2.8757     -1.1876
  14     -1.9353      0.9487      2.6729     -1.1038
  15     -1.9221      0.9422      2.4267     -1.0022
  16     -1.8296      0.8968      2.6854     -1.1090
  17     -1.7650      0.8652      3.0680     -1.2670
  18     -1.6407      0.8042      3.4536     -1.4263
  19     -1.6381      0.8029      3.2171     -1.3286
  20     -1.5535      0.7615      3.1863     -1.3159
  21     -1.6600      0.8137      3.0392     -1.2551
  22     -1.6657      0.8165      2.9229     -1.2071
  23     -1.6505      0.8091      2.8056     -1.1586
  24     -1.5501      0.7598      2.7671     -1.1428
  25     -1.6328      0.8004      2.6258     -1.0844
  26     -1.6116      0.7900      2.3998     -0.9911
  27     -1.5565      0.7630      2.4902     -1.0284
  28     -1.4851      0.7280      2.4854     -1.0264
  29     -1.5520      0.7607      2.6431     -1.0915
  30     -1.4473      0.7095      2.7709     -1.1443
  31     -1.5603      0.7648      2.9709     -1.2269
  32     -1.4866      0.7287      3.1613     -1.3055
  33     -1.5002      0.7354      2.9459     -1.2166
  34     -1.4462      0.7089      2.9181     -1.2051
  35     -1.3824      0.6776      3.0853     -1.2742
  36     -1.4201      0.6961      3.2767     -1.3532
  37     -1.4024      0.6874      3.4728     -1.4342
  38     -1.4296      0.7008      3.4901     -1.4413
  39     -1.3578      0.6656      3.4997     -1.4453
  40     -1.3491      0.6613      3.4228     -1.4136
  41     -1.3507      0.6621      3.3324     -1.3762
  42     -1.3654      0.6693      3.2613     -1.3468
  43     -1.2872      0.6310      2.9248     -1.2079
  44     -1.1452      0.5613      2.7171     -1.1221
  45     -1.0553      0.5173      2.8700     -1.1852
  46     -1.0300      0.5049      2.7123     -1.1201
  47     -0.9159      0.4490      2.5662     -1.0598
  48     -0.9620      0.4715      2.4719     -1.0209
  49     -1.0961      0.5373      2.5556     -1.0554

DARP Distance Expansion Estimates
Dependent Variable =        crime
R-squared     =    0.6083
Rbar-squared  =    0.5628
sige          =  119.6188
gamma         =   -0.0053
```

```
gamma, prob   =    0.0467,   0.8289
# of iterations =     10
log-likelihood  =       -138.64471
Nobs, Nvars   =     49,     3
central obs   =     20
************************************************************
Base centroid estimates
Variable         Coefficient      t-statistic    t-probability
const              62.323508         5.926825         0.000000
income             -0.889528        -0.925670         0.359448
hse value          -0.312813        -1.015500         0.315179
d-income           -0.004348        -0.790909         0.433056
d-hse value         0.000659         0.349488         0.728318
************************************************************
Expansion estimates
 Obs#      income   hse value
    1     -0.4032      0.0055
    2     -0.2644      0.0036
    3     -0.1761      0.0024
    4     -0.3070      0.0042
    5     -0.4350      0.0060
    6     -0.2311      0.0032
    7     -0.0866      0.0012
    8     -0.1552      0.0021
    9     -0.0190      0.0003
   10     -0.1837      0.0025
   11     -0.1994      0.0027
   12     -0.2513      0.0034
   13     -0.3172      0.0043
   14     -0.4554      0.0062
   15     -0.5492      0.0075
   16     -0.2807      0.0038
   17     -0.1145      0.0016
   18     -0.0455      0.0006
   19     -0.0178      0.0002
   20     -0.0000      0.0000
   21     -0.0359      0.0005
   22     -0.0569      0.0008
   23     -0.0776      0.0011
   24     -0.0662      0.0009
   25     -0.1338      0.0018
   26     -0.2413      0.0033
   27     -0.1826      0.0025
   28     -0.1965      0.0027
   29     -0.1112      0.0015
   30     -0.0925      0.0013
   31     -0.0176      0.0002
   32     -0.0111      0.0002
   33     -0.0287      0.0004
```

```
34     -0.0552      0.0008
35     -0.0753      0.0010
36     -0.0465      0.0006
37     -0.0867      0.0012
38     -0.0723      0.0010
39     -0.1305      0.0018
40     -0.1230      0.0017
41     -0.1085      0.0015
42     -0.0885      0.0012
43     -0.1989      0.0027
44     -0.4900      0.0067
45     -0.6437      0.0088
46     -0.7538      0.0103
47     -1.1374      0.0156
48     -1.0465      0.0143
49     -0.6607      0.0090
```

For the case of the distance expansion we find a single γ parameter that is negative but not significant. This would be interpreted to mean that the deterministic expansion relationship is performing adequately over space.

A comparison of the base model estimates from the x-y **darp** model versus those from **casetti** show relatively similar coefficient estimates as we would expect. In the case of the x-y expansion all of the signs are the same for spatial expansion and darp models. The distance expansion version of the model exhibits a sign change for the coefficient on income, which goes from positive in the expansion model to negative in the darp model. Correcting for the heteroscedastic character of the estimation problem produces dramatic changes in the statistical significance found for the base model estimates. They all become insignificant, a finding consistent with results reported by Anselin (1988) based on a jacknife approach to correcting for heteroscedasticity in this model. (Anselin finds that the income coefficient is still marginally significant after the correction.)

One approach to using this model is to expand around every point in space and examine the parameters γ for evidence indicating where the model is suffering from performance or parameter drift. Example 4.3 shows how this might be accomplished using a 'for loop' over all observations. For this purpose, we wish only to recover the estimated values for the parameter γ along with the marginal probability level.

```
% ----- example 4.3 Using darp() over space
% load Anselin (1988) Columbus neighborhood crime data
load anselin.dat; y = anselin(:,1); n = length(y);
x = [ones(n,1) anselin(:,2:3)];
xc = anselin(:,4); yc = anselin(:,5); % Anselin  x-y coordinates
```

```
vnames = strvcat('crime','const','income','hse value');
% do Casetti darp using distance expansion from all
% observations in the data sample
option.exp = 1;
output = zeros(n,2);
tic;
for i=1:n % loop over all observations
 option.ctr = i;
res = darp(y,x,xc,yc,option);
output(i,1) = res.gamma(1);
output(i,2) = res.cprob(1);
end;
toc;
in.cnames = strvcat('gamma estimate','marginal probability');
in.rflag = 1;
mprint(output,in)
```

We use the MATLAB 'tic' and 'toc' commands to time the operation of producing these maximum likelihood estimates across the entire sample. The results are shown below, where we find that it took around 70 seconds to solve the maximum likelihood estimation problem, calculate expansion estimates and produce all of the ancillary statistics 49 times, once for each observation in the data set.

```
elapsed_time = 69.2673
 Obs#       gamma estimate marginal probability
    1              -0.2198               0.0714
    2              -0.2718               0.0494
    3              -0.3449               0.0255
    4              -0.4091               0.0033
    5              -0.2223               0.0532
    6              -0.3040               0.0266
    7              -0.4154               0.0126
    8              -0.2071               0.1477
    9              -0.5773               0.0030
   10               0.1788               0.1843
   11               0.1896               0.1526
   12               0.1765               0.1621
   13               0.1544               0.1999
   14               0.1334               0.2214
   15               0.1147               0.2708
   16               0.1429               0.2615
   17               0.1924               0.2023
   18              -0.1720               0.3112
   19               0.1589               0.3825
   20              -0.3471               0.0810
   21               0.2020               0.2546
   22               0.1862               0.2809
```

23	0.1645	0.3334
24	0.0904	0.6219
25	0.1341	0.4026
26	0.1142	0.4264
27	0.1150	0.4618
28	0.0925	0.5584
29	0.1070	0.5329
30	-0.2765	0.1349
31	0.0453	0.8168
32	-0.6580	0.0012
33	-0.3293	0.0987
34	-0.5949	0.0024
35	-0.8133	0.0000
36	-0.5931	0.0023
37	-0.4853	0.0066
38	-0.4523	0.0121
39	-0.5355	0.0016
40	-0.6050	0.0005
41	-0.6804	0.0001
42	-0.7257	0.0001
43	-0.7701	0.0000
44	-0.5150	0.0001
45	-0.3997	0.0005
46	-0.3923	0.0003
47	-0.3214	0.0004
48	-0.3586	0.0001
49	-0.4668	0.0000

From the estimated values of γ and the associated marginal probabilities, we infer that the model suffers from performance drift over the initial 9 observations and sample observations from 32 to 49. We draw this inference from the negative γ estimates that are statistically significant for these observations. (Note that observation #8 is only marginally significant.) Over the middle range of the sample, from observations 10 to 31 we find that the deterministic distance expansion relationship works well. This inference arises from the fact that none of the estimated γ parameters are significantly different from zero.

In Section 4.4 we provide evidence that observations 2 and 4 represent outliers that impact on estimates for neighboring observations 1 to 9. We also show that this is true of observation 34, which influences observations 31 to 44. This suggests that the DARP model is working correctly to spot places where the model encounters problems.

4.3 Non-parametric locally linear models

These models represent an attempt to draw on the flexibility and tractability of non-parametric estimators. Note that the use of the spatial expansion and DARP methods in the previous section did not involve matrix manipulations or inversion of large sparse matrices. The models presented in this section share that advantage over the spatial autoregressive models.

Locally linear regression methods introduced in McMillen (1996,1997) and labeled geographically weighted regressions (GWR) in Brunsdon, Fotheringham and Charlton (1996) (BFC hereafter) are discussed in this section. The main contribution of the GWR methodology is the use of distance-weighted sub-samples of the data to produce locally linear regression estimates for every point in space. Each set of parameter estimates is based on a distance-weighted sub-sample of "neighboring observations", which has a great deal of intuitive appeal in spatial econometrics. While this approach has a definite appeal, it also presents some problems that are discussed in Section 4.4, where a Bayesian approach is used to overcome the problems.

The distance-based weights used by BFC for data at observation i take the form of a vector W_i which can be determined based on a vector of distances d_i between observation i and all other observations in the sample. This distance vector along with a distance decay parameter are used to construct a weighting function that places relatively more weight on sample observations from neighboring observations in the spatial data sample.

A host of alternative approach have been suggested for constructing the weight function. One approach suggested by BFC is:

$$W_i^2 = \exp(-d_i/\theta) \tag{4.19}$$

The parameter θ is a decay parameter that BFC label "bandwidth". Changing the bandwidth results in a different exponential decay profile, which in turn produces estimates that vary more or less rapidly over space.

Another weighting scheme is the tri-cube function proposed by McMillen (1998):

$$W_i^2 = (1 - (\theta/d_i)^3)^3 \tag{4.20}$$

Still another approach is to rely on a Gaussian function ϕ:

$$W_i^2 = \phi(d_i/\sigma\theta) \tag{4.21}$$

Where ϕ denotes the standard normal density and σ represents the standard deviation of the distance vector d_i.

The distance vector is calculated for observation i as:

$$d_i = \sqrt{(Z_{xi} - Z_{xj})^2 + (Z_{yi} - Z_{yj})^2} \tag{4.22}$$

Where Z_{xj}, Z_{yj} denote the latitude-longitude coordinates of the observations $j = 1, \ldots, n$.

Note that the notation used here may be confusing as we usually rely on subscripted variables to denote scalar elements of a vector. In the notation used here, the subscripted variable d_i represents a vector of distances between observation i and all other sample data observations. Similarly, the W_i is a vector of distance-based weights associated with observation i.

BFC use a single value of θ, the bandwidth parameter for all observations determined using a cross-validation procedure that is often used in locally linear regression methods. A score function taking the form:

$$\sum_{i=1}^{n} [y_i - \hat{y}_{\neq i}(\theta)]^2 \tag{4.23}$$

is used, where $\hat{y}_{\neq i}(\theta)$ denotes the fitted value of y_i with the observations for point i omitted from the calibration process. A value of θ that minimizes this score function is used as the distance-weighting bandwidth to produce GWR estimates.

The non-parametric GWR model relies on a sequence of locally linear regressions to produce estimates for every point in space by using a subsample of data information from nearby observations. Let y denote an nx1 vector of dependent variable observations collected at n points in space, X an nxk matrix of explanatory variables, and ε an nx1 vector of normally distributed, constant variance disturbances. Letting W_i represent an nxn diagonal matrix containing distance-based weights for observation i that reflect the distance between observation i and all other observations, we can write the GWR model as:

$$W_i^{1/2} y = W_i^{1/2} X \beta_i + \varepsilon_i \tag{4.24}$$

The subscript i on β_i indicates that this kx1 parameter vector is associated with observation i. The GWR model produces n such vectors of parameter estimates, one for each observation. These estimates are produced using:

$$\hat{\beta}_i = (X' W_i X)^{-1} (X' W_i y) \tag{4.25}$$

Keep in mind the notation, $W_i y$ denotes an n-vector of distance-weighted observations used to produce estimates for observation i. Note also, that $W_i X$ represents a distance-weighted data matrix, not a single observation and ε_i represents an n-vector.

Note that these GWR estimates for β_i are conditional on the parameter θ. That is, changing θ, the distance decay parameter will produce a different set of GWR estimates.

The best way to understand this approach to dealing with spatial heterogeneity is to apply the method, a subject to which we turn in the next section.

4.3.1 Implementing GWR

We have an optimization problem to solve regarding minimizing the score function to find the cross-validation bandwidth parameter θ. We first construct a MATLAB function to compute the scores associated with different bandwidths. This univariate function of the scalar bandwidth parameter can then be minimized using the simplex algorithm **fmin**.

Given the optimal bandwidth, estimation of the GWR parameters β and associated statistics can proceed via generalized least-squares. A function **gwr** whose documentation is shown below implements the estimation procedure.

```
 PURPOSE: compute geographically weighted regression
---------------------------------------------------
 USAGE: results = gwr(y,x,east,north,info)
 where:   y = dependent variable vector
          x = explanatory variable matrix
       east = x-coordinates in space
      north = y-coordinates in space
       info = a structure variable with fields:
       info.bwidth = scalar bandwidth to use or zero
                     for cross-validation estimation (default)
       info.dtype  = 'gaussian'    for Gaussian weighting (default)
                   = 'exponential' for exponential weighting
  NOTE: res = gwr(y,x,east,north) does CV estimation of bandwidth
 ---------------------------------------------------
 RETURNS: a results structure
        results.meth  = 'gwr'
        results.beta  = bhat matrix    (nobs x nvar)
        results.tstat = t-stats matrix (nobs x nvar)
        results.yhat  = yhat
        results.resid = residuals
        results.sige  = e'e/(n-dof) (nobs x 1)
```

```
        results.nobs  = nobs
        results.nvar  = nvars
        results.bwidth  = bandwidth
        results.dtype   = input string for Gaussian, exponential weights
        results.iter    = # of simplex iterations for cv
        results.north = north (y-coordinates)
        results.east  = east  (x-coordinates)
        results.y     = y data vector
-------------------------------------------------
 NOTES: uses auxiliary function scoref for cross-validation
 -------------------------------------------------
```

The following program illustrates using the **gwr** function on the Anselin (1988) neighborhood crime data set to produce estimates based on both Gaussian and exponential weighting functions. Figure 4.5 shows a graph of these two sets of estimates, indicating that they are not very different.

```
% ----- example 4.4 Using the gwr() function
% load the Anselin data set
load anselin.dat; y = anselin(:,1); nobs = length(y);
x = [ones(nobs,1) anselin(:,2:3)];  tt=1:nobs;
north = anselin(:,4); east = anselin(:,5);
info.dtype = 'gaussian';    % Gaussian weighting function
res1 = gwr(y,x,east,north,info);
info.dtype = 'exponential'; % Exponential weighting function
res2 = gwr(y,x,east,north,info);
subplot(3,1,1), plot(tt,res1.beta(:,1),tt,res2.beta(:,1),'--');
legend('Gaussian','Exponential'); ylabel('Constant term');
subplot(3,1,2), plot(tt,res1.beta(:,2),tt,res2.beta(:,2),'--');
legend('Gaussian','Exponential'); ylabel('Household income');
subplot(3,1,3), plot(tt,res1.beta(:,3),tt,res2.beta(:,3),'--');
legend('Gaussian','Exponential'); ylabel('House value');
```

The printed output for these models is voluminous as illustrated below, where we only print estimates associated with two observations.

```
Geometrically weighted regression estimates
Dependent Variable =        crime
R-squared      =    0.9418
Rbar-squared   =    0.9393
Bandwidth      =    0.6518
Decay type     =     gaussian
# iterations   =     17
Nobs, Nvars    =     49,     3
***************************************
Obs =    1, x-coordinate= 42.3800, y-coordinate= 35.6200, sige=  1.1144
Variable       Coefficient      t-statistic    t-probability
```

```
const            51.198618        16.121809         0.000000
income           -0.461074        -2.938009         0.005024
hse value        -0.434240        -6.463775         0.000000

Obs =    2, x-coordinate= 40.5200, y-coordinate= 36.5000, sige=  2.7690
Variable       Coefficient      t-statistic    t-probability
const            63.563830        15.583144         0.000000
income           -0.369869        -1.551568         0.127201
hse value        -0.683562        -7.288304         0.000000
```

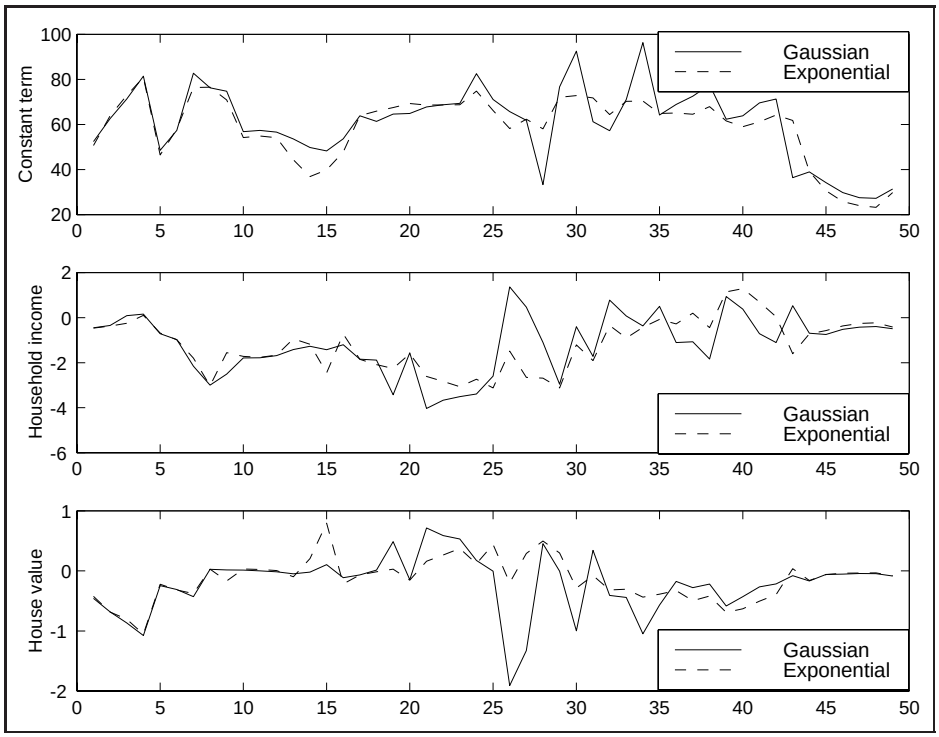

Figure 4.5: Comparison of GWR distance weighting schemes

4.4 A Bayesian Approach to GWR

A Bayesian treatment of locally linear geographically weighted regressions is set forth in this section. While the use of locally linear regression seems appealing, it is plagued by some problems. A Bayesian treatment can resolve

these problems and has some advantages over the non-parametric approach discussed in Section 4.3.

One problem with the non-parametric approach is that valid inferences cannot be drawn for the GWR regression parameters. To see this, consider that the locally linear estimates use the same sample data observations (with different weights) to produce a sequence of estimates for all points in space. Given a lack of independence between estimates for each location, conventional measures of dispersion for the estimates will likely be incorrect. These (invalid) conventional measures are what we report in the results structure from **gwr**, as this is the approach taken by Brunsdon, Fotheringham and Charlton (1996).

Another problem is that non-constant variance over space, aberrant observations due to spatial enclave effects, or shifts in regime can exert undue influence on locally linear estimates. Consider that all nearby observations in a sub-sequence of the series of locally linear estimates may be "contaminated" by an outlier at a single point in space. The Bayesian approach introduced here solves this problem by robustifying against aberrant observations. Aberrant observations are automatically detected and downweighted to lessen their influence on the estimates. The non-parametric implementation of the GWR model assumed no outliers.

A third problem is that the non-parametric estimates may suffer from "weak data" problems because they are based on a distance weighted subsample of observations. The effective number of observations used to produce estimates for some points in space may be very small. This problem can be solved with the Bayesian approach by incorporating subjective prior information during estimation. Use of subjective prior information is a well-known approach for overcoming "weak data" problems.

In addition to overcoming these problems, the Bayesian formulation introduced here specifies the relationship that is used to smooth parameters over space. This allows us to subsume the non-parametric GWR method as part of a much broader class of spatial econometric models. For example, the Bayesian GWR can be implemented with parameter smoothing relationships that result in: 1) a locally linear variant of the spatial expansion methods discussed in section 4.1, 2) a parameter smoothing relation appropriate for monocentric city models where parameters vary systematically with distance from the center of the city, 3) a parameter smoothing scheme based on contiguity relationships, and 4) a parameter smoothing scheme based on distance decay.

The Bayesian approach, which we label BGWR is best described using matrix expressions shown in (4.26) and (4.27). First, note that (4.26) is the

same as the non-parametric GWR relationship, but the addition of (4.27) provides an explicit statement of the parameter smoothing that takes place across space. The parameter smoothing involves a locally linear combination of neighboring areas, where neighbors are defined in terms of the distance weighting function that decays over space.

$$W_i^{1/2} y = W_i^{1/2} X \beta_i + \varepsilon_i \tag{4.26}$$

$$\beta_i = \begin{pmatrix} w_{i1} \otimes I_k & \dots & w_{in} \otimes I_k \end{pmatrix} \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_n \end{pmatrix} + u_i \tag{4.27}$$

The terms w_{ij} represent a normalized distance-based weight so the row-vector $(w_{i1}, \dots, w_{in})$ sums to unity, and we set $w_{ii} = 0$. That is, $w_{ij} = \exp(-d_{ij}/\theta)/\sum_{j=1}^{n} \exp(-d_{ij}/\theta)$.

To complete our model specification, we add distributions for the terms ε_i and u_i:

$$\varepsilon_i \sim N[0, \sigma^2 V_i], \qquad V_i = \text{diag}(v_1, v_2, \dots, v_n) \tag{4.28}$$

$$u_i \sim N[0, \delta^2 \sigma^2 (X' W_i X)^{-1})] \tag{4.29}$$

The $V_i = \text{diag}(v_1, v_2, \dots, v_n)$, represent our n variance scaling parameters from Chapter 3. These allow for non-constant variance as we move across space. One point to keep in mind is that here we have n^2 terms to estimate, reflecting n V_i vectors, one vector for each of the n observations. We will use the same assumption as in Chapter 3 regarding the V_i parameters. All n^2 parameters are assumed to be i.i.d. $\chi^2(r)$ distributed, where r is our hyperparameter that controls the amount of dispersion in the V_i estimates across observations. As in Chapter 3, we introduce a single hyperparameter r to the estimation problem and receive in return n^2 parameter estimates. Consider that as r becomes very large, the prior imposes homoscedasticity on the BGWR model and the disturbance variance becomes $\sigma^2 I_n$ for all observations i.

The distribution for u_i in the parameter smoothing relationship is normally distributed with mean zero and a variance based on Zellner's (1971) $g-$prior. This prior variance is proportional to the parameter variance-covariance matrix, $\sigma^2(X'W_iX)^{-1})$ with δ^2 acting as the scale factor. The use of this prior specification allows individual parameters β_i to vary by different amounts depending on their magnitude.

The parameter δ^2 acts as a scale factor to impose tight or loose adherence to the parameter smoothing specification. Consider a case where δ is very small, then the smoothing restriction would force β_i to look like a distance-weighted linear combination of other β_i from neighboring observations. On the other hand, as $\delta \rightarrow \infty$ (and $V_i = I_n$) we produce the non-parametric GWR estimates. To see this, we rewrite the BGWR model in a more compact form:

$$\begin{aligned} \tilde{y}_i &= \tilde{X}_i\beta_i + \varepsilon_i \qquad (4.30) \\ \beta_i &= J_i\gamma + u_i \qquad (4.31) \end{aligned}$$

Where the definitions of the matrix expressions are:

$$\begin{aligned} \tilde{y}_i &= W_i^{1/2}y \\ \tilde{X}_i &= W_i^{1/2}X \\ J_i &= \begin{pmatrix} w_{i1} \otimes I_k & \dots & w_{in} \otimes I_k \end{pmatrix} \\ \gamma &= \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_n \end{pmatrix} \end{aligned}$$

As indicated earlier, the notation is somewhat confusing in that $\tilde{y}_i$ denotes an $n-$vector, not a scalar magnitude. Similarly, ε_i is an $n-$vector and $\tilde{X}_i$ is an n by k matrix. Note that (4.30) can be written in the form of a Theil and Goldberger (1961) estimation problem as shown in (4.32).

$$\begin{pmatrix} \tilde{y}_i \\ J_i\gamma \end{pmatrix} = \begin{pmatrix} \tilde{X}_i \\ -I_k \end{pmatrix} \beta_i + \begin{pmatrix} \varepsilon_i \\ u_i \end{pmatrix} \qquad (4.32)$$

Assuming $V_i = I_n$, the estimates β_i take the form:

$$\begin{aligned} \hat{\beta}_i &= R(\tilde{X}_i'\tilde{y}_i + \tilde{X}_i'\tilde{X}_iJ_i\gamma/\delta^2) \\ R &= (\tilde{X}_i'\tilde{X}_i + \tilde{X}_i'\tilde{X}_i/\delta^2)^{-1} \end{aligned}$$

As δ approaches ∞, the terms associated with the Theil and Goldberger "stochastic restriction", $\tilde{X}_i'\tilde{X}_iJ_i\gamma/\delta^2$ and $\tilde{X}_i'\tilde{X}_i/\delta^2$ become zero, and we have the GWR estimates:

$$\hat{\beta}_i = (\tilde{X}_i'\tilde{X}_i)^{-1}(\tilde{X}_i'\tilde{y}_i) \tag{4.33}$$

In practice, we can use a diffuse prior for δ which allows the amount of parameter smoothing to be estimated from sample data information, rather than by subjective prior information.

Details concerning estimation of the parameters in the BGWR model are taken up in the next section. Before turning to these issues, we consider some alternative spatial parameter smoothing relationships that might be used in lieu of (4.27) in the BGWR model.

One alternative smoothing specification would be the "monocentric city smoothing" set forth in (4.34). This relation assumes that the data observations have been ordered by distance from the center of the spatial sample.

$$\begin{aligned} \beta_i &= \beta_{i-1} + u_i \\ u_i &\sim N[0, \delta^2\sigma^2(X'W_iX)^{-1}] \end{aligned} \tag{4.34}$$

Given that the observations are ordered by distance from the center, the smoothing relation indicates that β_i should be similar to the coefficient β_{i-1} from a neighboring concentric ring. Note that we rely on the same GWR distance-weighted data sub-samples, created by transforming the data using: W_iy, W_iX. This means that the estimates still have a "locally linear" interpretation as in the GWR. We rely on the same distributional assumption for the term u_i from the BGWR which allows us to estimate the parameters from this model by making minor changes to the approach used for the BGWR.

Another alternative is a "spatial expansion smoothing" based on the ideas introduced by Casetti (1972). This is shown in (4.35), where Z_{xi}, Z_{yi} denote latitude-longitude coordinates associated with observation i.

$$\begin{aligned} \beta_i &= \begin{pmatrix} Z_{xi} \otimes I_k & Z_{yi} \otimes I_k \end{pmatrix} \begin{pmatrix} \beta_x \\ \beta_y \end{pmatrix} + u_i \\ u_i &\sim N[0, \delta^2\sigma^2(X'W_iX)^{-1})] \end{aligned} \tag{4.35}$$

This parameter smoothing relation creates a locally linear combination based on the latitude-longitude coordinates of each observation. As in the case of the monocentric city specification, we retain the same assumptions regarding the stochastic term u_i, making this model simple to estimate with minor changes to the BGWR methodology.

Finally, we could adopt a "contiguity smoothing" relationship based on a first-order spatial contiguity matrix as shown in (4.36). The terms c_{ij} represent the ith row of a row-standardized first-order contiguity matrix. This creates a parameter smoothing relationship that averages over the parameters from observations that neighbor observation i.

$$\begin{aligned} \beta_i &= \left(\begin{array}{cccc} c_{i1} \otimes I_k & \ldots & c_{in} \otimes I_k \end{array} \right) \left(\begin{array}{c} \beta_1 \\ \vdots \\ \beta_n \end{array} \right) + u_i \qquad (4.36) \\ u_i &\sim N[0, \delta^2 (X' W_i^2 X)^{-1})] \end{aligned}$$

Alternative approaches to specifying geographically weighted regression models suggest that researchers need to think about which type of spatial parameter smoothing relationship is most appropriate for their application. Additionally, where the nature of the problem does not clearly favor one approach over another, statistical tests of alternative models based on different smoothing relations might be carried out. Posterior odds ratios can be constructed that will shed light on which smoothing relationship is most consistent with the sample data. We illustrate model specification issues in an applied example in Section 4.5.

4.4.1 Estimation of the BGWR model

We use Gibbs sampling to estimate the BGWR model. This approach is particularly attractive in this application because the conditional densities all represent know distributions that are easy to obtain. In Chapter 3 we saw an example of Gibbs sampling where the conditional distribution for the spatial autoregressive parameters were from an unknown distribution and we had to rely on the more complicated case of Metropolis within-Gibbs sampling.

To implement the Gibbs sampler we need to derive and draw samples from the conditional posterior distributions for each group of parameters, β_i, σ, δ, and V_i in the model. Let $P(\beta_i | \sigma, \delta, V_i, \gamma)$ denote the conditional density of β_i, where γ represents the values of other β_j for observations $j \neq i$. Using similar notation for the the other conditional densities, the Gibbs sampling process can be viewed as follows:

1. start with arbitrary values for the parameters $\beta_i^0, \sigma^0, \delta^0, V_i^0, \gamma^0$
2. for each observation $i = 1, \ldots, n$,

(a) sample a value, β_i^1 from $P(\beta_i|\sigma^0, \delta^0, V_i^0, \gamma^0)$
(b) sample a value, V_i^1 from $P(V_i|\beta_i^1, \sigma^0, \delta^0, \gamma^0)$

3. use the sampled values $\beta_i^1, i = 1, \ldots, n$ from each of the n draws above to update γ^0 to γ^1.
4. sample a value, σ^1 from $P(\sigma|\delta^0, V_i^1, \gamma^1)$
5. sample a value, δ^1 from $P(\delta|\sigma^1, V_i^1, \gamma^1)$
6. go to step 1 using $\beta_i^1, \sigma^1, \delta^1, V_i^1, \gamma^1$ in place of the arbitrary starting values.

The sequence of draws outlined above represents a single pass through the sampler, and we make a large number of passes to collect a large sample of parameter values from which we construct our posterior distributions.

We rely on the compact statement of the BGWR model in (4.30) to facilitate presentation of the conditional distributions that we rely on during the sampling.

The conditional posterior distribution of β_i given σ, δ, γ and V_i is a multivariate normal shown in (4.37).

$$p(\beta_i| \ldots) \propto N(\hat{\beta}_i, \sigma^2 R) \tag{4.37}$$

Where:

$$\begin{aligned} \hat{\beta}_i &= R(\tilde{X}_i' V_i^{-1} \tilde{y}_i + \tilde{X}_i' \tilde{X}_i J_i \gamma / \delta^2) \\ R &= (\tilde{X}_i' V_i^{-1} \tilde{X}_i + \tilde{X}_i' \tilde{X}_i / \delta^2)^{-1} \end{aligned} \tag{4.38}$$

This result follows from the assumed variance-covariance structures for ε_i, u_i and the Theil-Goldberger (1961) representation shown in (4.32).

The conditional posterior distribution for σ is a $\chi^2(m = n^2)$ distribution shown in (4.39).

$$\begin{aligned} p(\sigma| \ldots) &\propto \sigma^{-(m+1)} \exp\{-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (\varepsilon_i' V_i^{-1} \varepsilon_i)\} \\ \varepsilon_i &= \tilde{y}_i - \tilde{X}_i \beta_i \end{aligned} \tag{4.39}$$

The sum in (4.39) extends over the subscript i to indicate that the $n-$ vector of the squared residuals (deflated by the n individual V_i terms) from each sub-sample of n observations are summed, and then these n sums are summed as well.

The conditional posterior distribution for V_i is shown in (4.40), which indicates that we draw an n-vector based on a $\chi^2(r+1)$ distribution. Note that the individual elements of the matrix V_i act on the spatial weighting scheme because the estimates involve terms like: $\tilde{X}_i' V_i^{-1} \tilde{X}_i = X' W_i V_i^{-1} W_i X$. The terms $W_i = \sqrt{\exp(-d_i/\theta)}$ from the weighting scheme will be adjusted by the V_i estimates, which are large for aberrant observations or outliers. In the event of an outlier, observation i will receive less weight when the spatial distance-based weight is divided by a large V_i value.

$$p\{[(e_i^2/\sigma^2) + r]/V_i \mid \ldots\} \propto \chi^2(r+1) \tag{4.40}$$

Finally, the conditional distribution for δ is a $\chi^2(nk)$ distribution based on (4.41).

$$p(\delta| \ldots) \propto \delta^{-nk} \exp\{-\sum_{i=1}^{n} (\beta_i - J_i\gamma)'(\tilde{X}_i'\tilde{X}_i)^{-1}(\beta_i - J_i\gamma)/2\delta^2\sigma^2\} \tag{4.41}$$

Now consider the modifications needed to the conditional distributions to implement the alternative spatial smoothing relationships set forth in Section 4.4.1. Since we maintained the assumptions regarding the disturbance terms ε_i, u_i, we need only alter the conditional distributions for β_i and δ. First, consider the case of the monocentric city smoothing relationship. The conditional distribution for β_i is multivariate normal with mean $\hat{\beta}_i$ and variance-covariance $\sigma^2 R$ as shown in (4.42).

$$\begin{aligned} \hat{\beta}_i &= R(\tilde{X}_i' V_i^{-1} \tilde{y}_i + \tilde{X}_i'\tilde{X}_i \beta_{i-1}/\delta^2) \\ R &= (\tilde{X}_i' V_i^{-1} \tilde{X}_i + \tilde{X}_i'\tilde{X}_i/\delta^2)^{-1} \end{aligned} \tag{4.42}$$

The conditional distribution for δ is a $\chi^2(nk)$ based on the expression in (4.43).

$$p(\delta| \ldots) \propto \delta^{-nk} \exp\{-\sum_{i=1}^{n} (\beta_i - \beta_{i-1})'(\tilde{X}_i'\tilde{X}_i)^{-1}(\beta_i - \beta_{i-1})/\delta^2\sigma^2\} \tag{4.43}$$

For the case of the spatial expansion and contiguity smoothing relationships, we can maintain the conditional expressions for β_i and δ from the case

of the BGWR, and simply modify the definition of J, to be consistent with these smoothing relations. One additional change needs to be made for the case of the spatial expansion smoothing relationship, we need to add a conditional distribution for the parameters β_x, β_y in the model. This distribution is a multivariate normal with mean $\hat{\beta} = (\hat{\beta}_x \hat{\beta}_y)'$ and variance-covariance matrix $\sigma^2(J_i' \tilde{X}_i' Q^{-1} \tilde{X}_i J_i)^{-1}$ as defined in (4.44).

$$\begin{aligned} \hat{\beta} &= (J_i' \tilde{X}_i' Q^{-1} \tilde{X}_i J_i)^{-1} (J_i' \tilde{X}_i' Q^{-1} \tilde{y}_i) \\ Q &= (V_i + \tilde{X}_i (\tilde{X}_i' \tilde{X}_i)^{-1} \tilde{X}_i' / \delta^2) \end{aligned} \tag{4.44}$$

4.4.2 Informative priors

Implementing the BGWR model with diffuse priors on δ may lead to large values that essentially eliminate the parameter smoothing relationship from the model. The BGWR estimates will then collapse on the GWR estimates (in the case of a large value for the hyperparameter r that leads to $V_i = I_n$). In cases where the sample data is weak or objective prior information suggests spatial parameter smoothing should follow a particular specification, we can use an informative prior for the parameter δ. A Gamma(a, b) prior distribution which has a mean of a/b and variance of a/b^2 seems appropriate. Given this prior, we could eliminate the conditional density for δ and replace it with a random draw from the Gamma(a, b) distribution.

In order to devise an appropriate prior setting for δ, consider that the GWR variance-covariance matrix is: $\sigma^2(\tilde{X}'\tilde{X})^{-1}$, so setting values for $\delta > 1$ would represent a relatively loose imposition of the parameter smoothing relationship. Values of $\delta < 1$ would impose the parameter smoothing prior more tightly.

A similar approach can be taken for the hyperparameter r. Using a Gamma prior distribution with $a = 8, b = 2$ that indicates small values of r around 4, should provide a fair amount of robustification if there is spatial heterogeneity. In the absence of heterogeneity, the resulting V_i estimates will be near unity so the BGWR distance weights will be similar to those from GWR, even with a small value of r.

Additionally, a $\chi^2(c, d)$ natural conjugate prior for the parameter σ could be used in place of the diffuse prior set forth here. This would affect the conditional distribution used during Gibbs sampling in only a minor way.

Some other alternatives offer additional flexibility when implementing the BGWR model. For example, one can restrict specific parameters to exhibit no variation over the spatial sample observations. This might be

useful if we wish to restrict the constant term to be constant over space. Or, it may be that the constant term is the only parameter that would be allowed to vary over space.

These alternatives can be implemented by adjusting the prior variances in the parameter smoothing relationship:

$$var - cov(\beta_i) = \delta^2\sigma^2(\tilde{X}_i'\tilde{X}_i)^{-1} \tag{4.45}$$

For example, assuming the constant term is in the first column of the matrix $\tilde{X}_i$, setting the first row and column elements of $(\tilde{X}_i'\tilde{X}_i)^{-1}$ to zero would restrict the intercept term to remain constant over all observations.

4.4.3 Implementation details

We have devised a function **bgwr** to carry out Gibbs sampling estimation of the Bayesian GWR model. The documentation for the function is shown below, where a great many user-supplied options are available. These options are input using a structure variable named 'prior' with which alternative types of parameter smoothing relationships can be indicated. Note that only three of the four parameter smoothing relationships discussed in Section 4.4 are implemented. The Casetti spatial expansion parameter smoothing relationship is not yet implemented. Another point to note is that you can implement a contiguity smoothing relationship by either specifying a spatial weight matrix or relying on the function to calculate this matrix based on the x-y coordinates using the function **xy2cont** discussed in Chapter 2.

```
 PURPOSE: compute Bayesian geographically weighted regression
          model: y = Xb(i) + e,      e = N(0,sige*V),
                 b(i) = f[b(j)] + u, u = delta*sige*inv(x'x)
          V = diag(v1,v2,...vn),     r/vi = ID chi(r)/r,
          delta = gamma(s,t),        r = Gamma(m,k)
          f[b(j)] = b(i-1) for concentric city prior
          f[b(j)] = W(i) b for contiguity prior
          f[b(j)] = [exp(-d/c)/sum(exp(-d/c)] b for distance prior
                c = GWR c.v. determined bandwidth
-------------------------------------------------
 USAGE: results = bgwr(y,x,xcoord,ycoord,ndraw,nomit,prior)
 where: y = dependent variable vector
        x = explanatory variable matrix
        xcoord = x-coordinates in space
        ycoord = y-coordinates in space
        prior = a structure variable with fields:
        prior.rval, improper r value, default=4
        prior.m,    informative Gamma(m,k) prior on r
```

```
        prior.k,    (default: not used)
        prior.dval, improper delta value (default=diffuse)
        prior.s,    informative Gamma(s,t) prior on delta
        prior.t,    (default: not used)
        prior.ptype, 'concentric' for concentric city smoothing
                     'distance'   for distance based smoothing (default)
                     'contiguity' for contiguity smoothing
                     'casetti'    for casetti smoothing (not implemented)
        prior.ctr, observation # of central point (for concentric prior)
        prior.W,   (optional) prior weight matrix (for contiguity prior)
        ndraw = # of draws
        nomit = # of initial draws omitted for burn-in
 ---------------------------------------------------
 RETURNS: a results structure
   results.meth    = 'bgwr'
   results.bdraw   = beta draws (ndraw-nomitxnobsxnvar) (3-d matrix)
   results.sdraw   = sige draws (ndraw-nomit x 1)
   results.vmean   = mean of vi draws (1 x nobs)
   results.rdraw   = r-value draws (ndraw-nomit x 1)
   results.ddraw   = delta draws (if diffuse prior used)
   results.r       = value of hyperparameter r (if input)
   results.d       = value of hyperparameter delta (if input)
   results.m       = m prior parameter (if input)
   results.k       = k prior parameter (if input)
   results.s       = s prior parameter (if input)
   results.t       = t prior parameter (if input)
   results.nobs    = nobs
   results.nvar    = nvars
   results.ptype   = input string for parameter smoothing relation
   results.xcoord  = x-coordinates
   results.ycoord  = y-coordinates
   results.ctr     = central point observation # (if concentric prior)
   results.dist    = distance vector (if ptype = 0)
   results.y       = y data vector
   results.logpost = (nobs x 1) vector of approximate log posterior
   results.time    = time taken for sampling
---------------------------------------------------
 NOTE: use either improper prior.rval
       or informative Gamma prior.m, prior.k, not both of them
       uses exponential distance weighting function exp(-d/bwdith)
       where d=distance and bwidth = gwr c.v. determined bandwidth
```

The user also has control over options for assigning a prior to the hyperparameter r that robustifies with respect to outliers and accommodates non-constant variance. Either an improper prior value can be set (as a rule-of-thumb I recommend $r = 4$), or a proper prior based on a Gamma(m,k) distribution can be used. Here, one would try to rely on a prior in the range of 4 to 10, because larger values produce estimates that are not robust to

heteroscedasticity or outliers. As an example, $m = 8, k = 2$ would implement a prior with the mean of $r = 4$ and the variance of $r = 2$, since the mean of the Gamma distribution is m/k, and the variance is (m/k^2).

The hyperparameter δ can be handled in three ways: 1) we can simply assign an improper prior value using say, 'prior.dval=20' as an input option, 2) we can input nothing about this parameter producing a default implementation based on a diffuse prior where δ will be estimated, and 3) we can assign a Gamma(s,t) prior as in the case of the hyperparameter r. Implementation with a diffuse prior for δ and a large value for the hyperparameter r will most likely reproduce the non-parameter GWR estimates, and this approach to producing those estimates requires more computing time. It is possible (but not likely) that a model and the sample data are very consistent with the parameter smoothing relationship. If this occurs, a diffuse prior for δ will produce a relatively small value as the posterior estimate. In the most likely cases encountered in practice, small deviations of the parameters from the smoothing relationship will lead to very large estimates for δ, producing BGWR parameter estimates that come very close to those from the non-parametric GWR model.

The value of the Bayesian approach outlined here lies in the ability to robustifying against outliers, so a default value of $r = 4$ has been implemented if the user enters no information regarding the hyperparameter r.

Consider how the following alternative implementations of the various prior settings could be used to shed light on the nature of parameter variation over space. We can compare the results from a GWR model to a BGWR model implemented with $r = 4$ and either a diffuse prior for δ or an improper prior based on large δ value. This comparison should show the impact of robustification on the estimates, and a plot of the V_i estimates can be used to detect outliers. Another model based on $r = 4$ along with an informative prior for $\delta \leq 1$ that places some weight on the parameter smoothing restrictions can be used to see how this alters the estimates when compared to the robust BGWR estimates. A dramatic difference between the robust BGWR estimates and those based on the informative prior for $\delta \leq 1$ indicates that the parameter smoothing relation is inconsistent with the sample data.

It may be necessary to experiment with alternative values of δ because the scale is unclear in any given problem. One way to deal with the scale issue is to calibrate δ based on the diffuse estimate. As an example, if $\delta^2 = 10$, then a value of $\delta < 10$ will impose the parameter smoothing restriction more tightly. To see this, consider that the GWR variance-covariance matrix is $\sigma^2(\tilde{X}'\tilde{X})^{-1}$, so using $\delta < 1$ moves in the direction of tightening the pa-

rameter smoothing prior. We will provide an example of this in the next section.

The function **bgwr** returns the entire sequence of draws made for the parameters in the problem, allowing one to: check for convergence of the Gibbs sampler, plot posterior density functions using a function **pltdens** from the *Econometrics Toolbox* or compute statistics from the posterior distributions of the parameters. Most users will rely on the **prt** function that calls a related function to produce printed output very similar to the results printed for the **gwr** function. If you do want to access the β estimates, note that they are stored in a MATLAB 3-dimensional matrix structure. We illustrate how to access these in Example 4.5 of the next section.

4.5 An applied exercise

The program in example 4.5 shows how to use the **bgwr** function to produce estimates for the Anselin neighborhood crime data set. We begin by using an improper prior for $\delta = 1000000$ and setting $r = 30$ to demonstrate that the BGWR model can replicate the estimates from the GWR model. The program plots these estimates for comparison. We produce estimates for all three parameter smoothing priors, but given the large $\delta = 1000000$, these relationships should not effectively enter the model. This will result in all three sets of estimates identical to those from the GWR. We did this to illustrate that the BGWR can replicate the GWR estimates with appropriate settings for the hyperparameters.

```
% ----- example 4.5 Using the bgwr() function
load anselin.data;  % load the Anselin data set
y = anselin(:,1); nobs = length(y); x = [ones(nobs,1) anselin(:,2:3)];
east = anselin(:,4); north = anselin(:,5); tt=1:nobs;
ndraw = 250; nomit = 50;
prior.ptype = 'contiguity'; prior.rval = 30; prior.dval = 1000000;
tic; r1 = bgwr(y,x,east,north,ndraw,nomit,prior); toc;
prior.ptype = 'concentric'; prior.ctr = 20;
tic; r2 = bgwr(y,x,east,north,ndraw,nomit,prior); toc;
dist = res2.dist; [dists di] = sort(dist); % recover distance vector
prior.ptype = 'distance';
tic; r3 = bgwr(y,x,east,north,ndraw,nomit,prior); toc;
vnames = strvcat('crime','constant','income','hvalue');
% compare gwr estimates with posterior means
info2.dtype = 'exponential';
result = gwr(y,x,east,north,info2);
bgwr = result.beta(di,:);
b1 = r1.bdraw(:,di,1); b2 = r1.bdraw(:,di,2); b3 = r1.bdraw(:,di,3);
```

```
b1m = mean(b1); b2m = mean(b2); b3m = mean(b3);
c1 = r2.bdraw(:,:,1); c2 = r2.bdraw(:,:,2); c3 = r2.bdraw(:,:,3);
c1m = mean(c1); c2m = mean(c2); c3m = mean(c3);
d1 = r3.bdraw(:,di,1); d2 = r3.bdraw(:,di,2); d3 = r3.bdraw(:,di,3);
d1m = mean(d1); d2m = mean(d2); d3m = mean(d3);
% plot mean of vi draws (sorted by distance from #20)
plot(tt,r1.vmean(1,di),'-b',tt,r2.vmean,'--r',tt,r3.vmean(1,di),'-.k');
title('vi means'); legend('contiguity','concentric','distance');
pause;
% plot beta estimates (sorted by distance from #20)
subplot(3,1,1),
plot(tt,bgwr(:,1),'-k',tt,b1m,'--k',tt,c1m,'-.k',tt,d1m,':k');
legend('gwr','contiguity','concentric','distance');
xlabel('b1 parameter');
subplot(3,1,2),
plot(tt,bgwr(:,2),'-k',tt,b2m,'--k',tt,c2m,'-.k',tt,d2m,':k');
xlabel('b2 parameter');
subplot(3,1,3),
plot(tt,bgwr(:,3),'-k',tt,b3m,'--k',tt,c3m,'-.k',tt,d3m,':k');
xlabel('b3 parameter');
```

As we can see from the graph of the GWR and BGWR estimates shown in Figure 4.6, the three sets of BGWR estimates are nearly identical to the GWR. Keep in mind that we don't recommend using the BGWR model to replicate GWR estimates, as this problem involving 250 draws took 193 seconds for the contiguity prior, 185 for the concentric city prior and 190 seconds for the distance prior.

Example 4.6 produces estimates based on the contiguity smoothing relationship with a value of $r = 4$ for this hyperparameter to indicate a prior belief in heteroscedasticity or outliers. We keep the parameter smoothing relationship from entering the model by setting an improper prior based on $\delta = 1000000$, so there is no need to run more than a single parameter smoothing model. Estimates based on any of the three parameter smoothing relationships would produce the same results because the large value for δ keeps this relation from entering the model. The focus here is on the impact of outliers in the sample data and how BGWR robust estimates compare to the GWR estimates based on the assumption of homoscedasticity. We specify 250 draws with the first 50 to be discarded for "burn-in" of the Gibbs sampler. Figure 4.7 shows a graph of the mean of the 200 draws which represent the posterior parameter estimates, compared to the non-parametric estimates.

```
% ----- example 4.6 Producing robust BGWR estimates
% load the Anselin data set
```

```
load anselin.data;
y = anselin(:,1); nobs = length(y);
x = [ones(nobs,1) anselin(:,2:3)];
east = anselin(:,4); north = anselin(:,5);
ndraw = 250; nomit = 50;
prior.ptype = 'contiguity'; prior.rval = 4; prior.dval = 1000000;
% use diffuse prior for distance decay smoothing of parameters
result = bgwr(y,x,east,north,ndraw,nomit,prior);
vnames = strvcat('crime','constant','income','hvalue');
info.dtype = 'exponential';
result2 = gwr(y,x,east,north,info);
% compare gwr and bgwr estimates
b1 = result.bdraw(:,:,1); b1mean = mean(b1);
b2 = result.bdraw(:,:,2); b2mean = mean(b2);
b3 = result.bdraw(:,:,3); b3mean = mean(b3);
betagwr = result2.beta;
tt=1:nobs;
subplot(3,1,1),
plot(tt,betagwr(:,1),'-k',tt,b1mean,'--k');
```

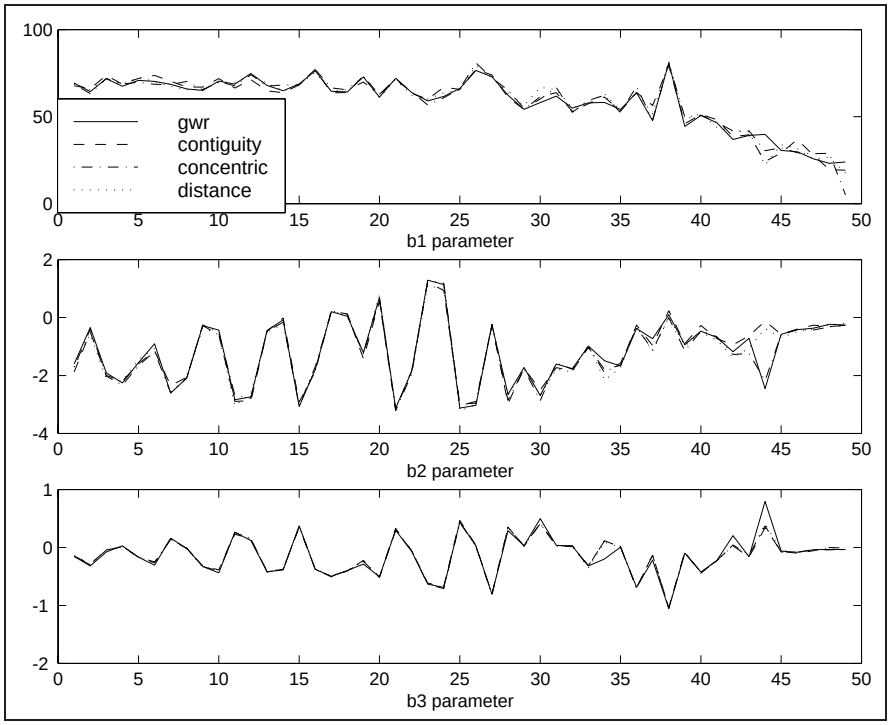

Figure 4.6: GWR and BGWR diffuse prior estimates

```
legend('gwr','bgwr');
xlabel('b1 parameter');
subplot(3,1,2),
plot(tt,betagwr(:,2),'-k',tt,b2mean,'--k');
xlabel('b2 parameter');
subplot(3,1,3),
plot(tt,betagwr(:,3),'-k',tt,b3mean,'--k');
xlabel('b3 parameter');
pause;
plot(result.vmean);
xlabel('Observations');
ylabel('V_{i} estimates');
```

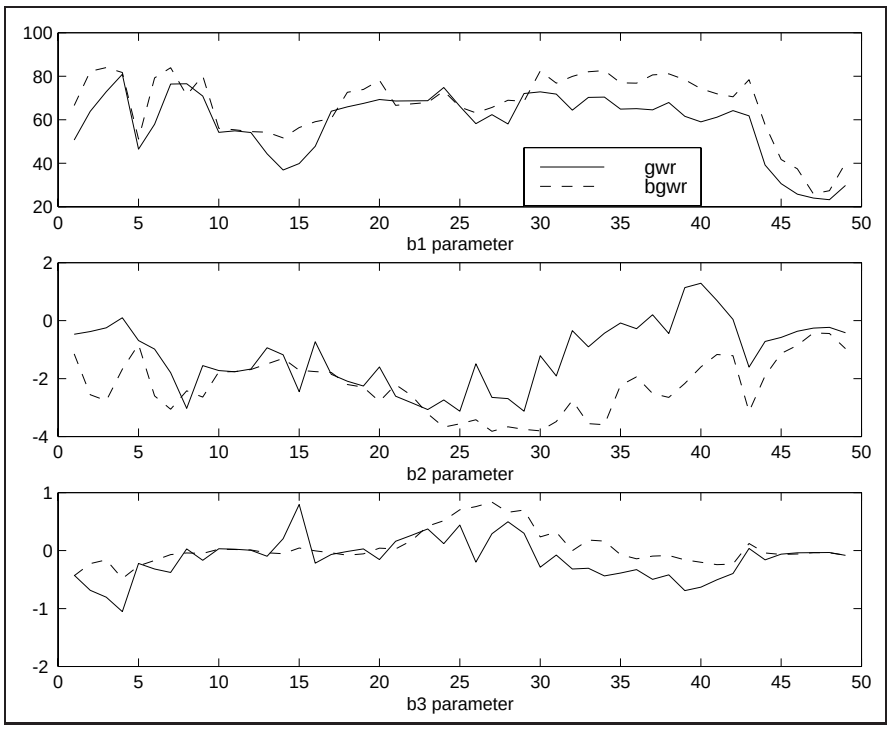

Figure 4.7: GWR and Robust BGWR estimates

We see a departure of the two sets of estimates around the first 10 observations and around observations 30 to 45. To understand why, we need to consider the V_i terms that represent the only difference between the BGWR and GWR models given that we used a large δ value. Figure 4.9 shows the mean of the draws for the V_i parameters. Large estimates for the

V_i terms at observations 2, 4 and 34 indicate aberrant observations. This accounts for the difference in trajectory taken by the non-parametric GWR estimates and the Bayesian estimates that robustify against these aberrant observations.

To illustrate how robustification takes place, Figure 4.8 shows the weighting terms W_i from the GWR model plotted alongside the weights adjusted by the V_i terms, $W_i^{1/2}V_i^{-1}W_i^{1/2}$ from the BGWR model. A sequence of six observations from 30 to 35 are plotted, with a symbol 'o' placed at observation #34 on the BGWR weights to help distinguish this observation in the figure.

Beginning with observation #30, the aberrant observation #34 is downweighted when estimates are produced for observations #30 to #35. This downweighting of the distance-based weight for observation #34 occurs during estimation of β_i for observations #30 through #35, all of which are near #34 in terms of the GWR distance measure. This alternative weighting produces the divergence between the GWR and BGWR estimates that we observe in Figure 4.7 starting around observation #30.

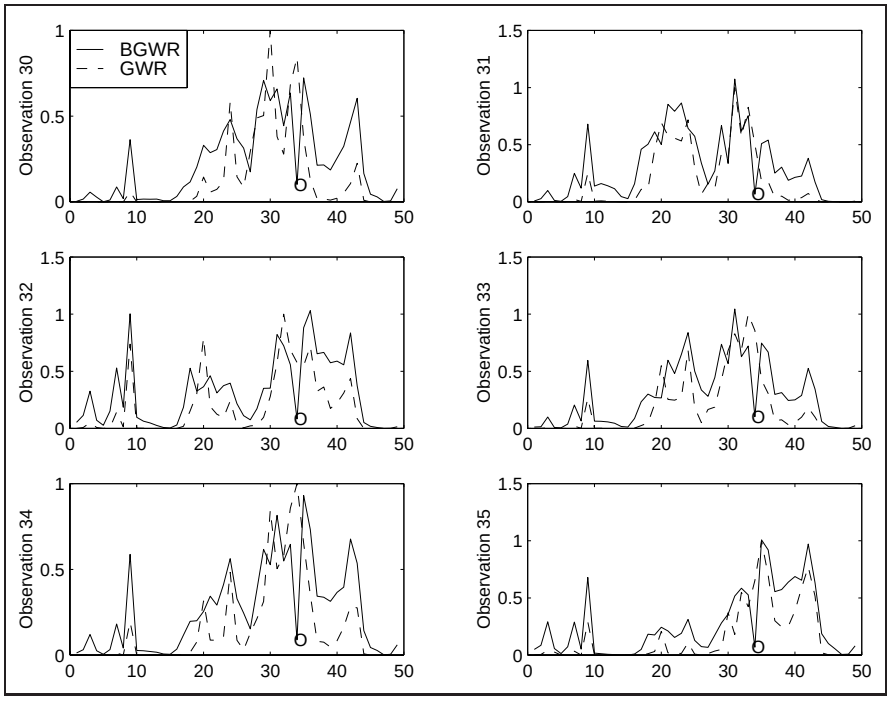

Figure 4.8: GWR and BGWR Distance-based weights adjusted by V_i

Ultimately, the role of the parameters V_i in the model and the prior assigned to these parameters reflects our prior knowledge that distance alone may not be reliable as the basis for spatial relationships between variables. If distance-based weights are used in the presence of aberrant observations, inferences will be contaminated for whole neighborhoods and regions in our analysis. Incorporating this prior knowledge turns out to be relatively simple in the Bayesian framework, and it appears to effectively robustify estimates against the presence of spatial outliers.

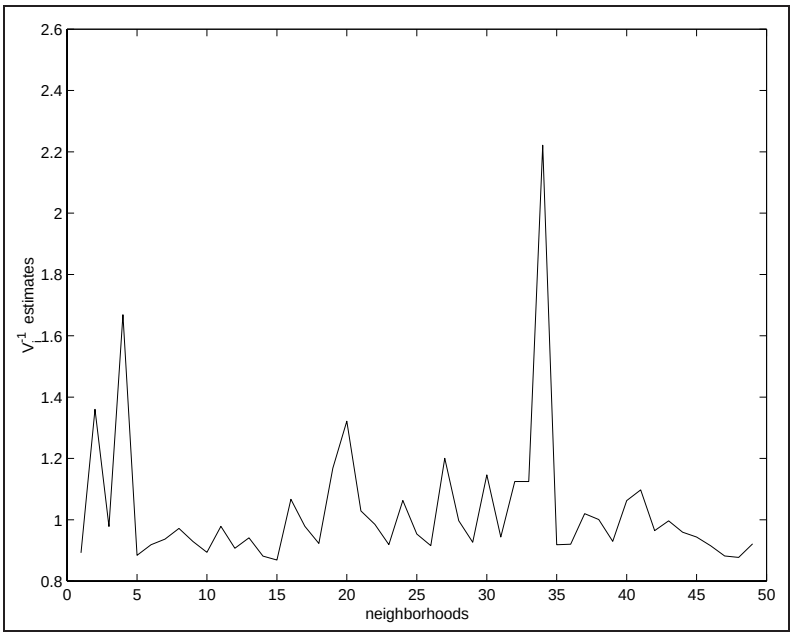

Figure 4.9: Average V_i estimates over all draws and observations

The function **bgwr** has associated **prt** and **plt** methods to produced printed and graphical presentation of the results. Some of the printed output is shown below. Note that the time needed to carry out 550 draws was 289 seconds, making this estimation approach quite competitive to DARP or GWR. The **plt** function produces the same output as for the GWR model.

```
Gibbs sampling geographically weighted regression model
Dependent Variable =         crime
R-squared          =    0.5650
sigma^2            =    0.7296
Nobs, Nvars        =     49,     3
ndraws,nomit       =    550,    50
```

```
r-value          =    4.0000
delta-value      =   25.0776
gam(m,k) d-prior =     50,     2
time in secs     =  289.2755
prior type       =   distance
************************************************************
Obs =    1, x-coordinate= 35.6200, y-coordinate= 42.3800
Variable      Coefficient      t-statistic    t-probability
constant        74.130145        15.143596         0.000000
income          -2.208382        -9.378908         0.000000
hvalue          -0.197050        -5.166565         0.000004

Obs =    2, x-coordinate= 36.5000, y-coordinate= 40.5200
Variable      Coefficient      t-statistic    t-probability
constant        82.308344        20.600005         0.000000
income          -2.559334       -11.983369         0.000000
hvalue          -0.208478        -4.682454         0.000023
```

The next task was to implement the BGWR model with a diffuse prior on the δ parameter. The results indicated that the mean of the draws for δ was: around 32 for the contiguity prior, 43 for the concentric prior and 33 for the distance prior. The BGWR estimates were almost identical to those from the improper prior $\delta = 1000000$, so we do not present these graphically.

Given these estimates for δ with a diffuse prior, we can impose the parameter restrictions by setting smaller values for δ, say in the range of 1 to 10. This should produce differing estimates that rely on the alternative parameter smoothing relationships. We used $\delta = 1$ to impose the parameter smoothing relationships fairly tightly. The resulting parameter estimates are shown in Figure 4.10.

Here we see some departure between the estimates based on alternative smoothing relationships. This raises the question of which smoothing relationship is most consistent with the data.

We can compute posterior probabilities for each of the three models based on alternative parameter smoothing relationships using an approximation from Leamer (1983). Since this is a generally useful way of comparing alternative Bayesian models, the **bgwr()** function returns a vector of the approximate log posterior for each observation. The nature of this approximation as well as the computation is beyond the scope of our discussion here. The program in example 4.7 demonstrates how to use the vector of log posterior magnitudes to compute posterior probabilities for each model. We set the parameter $\delta = 0.5$, to impose the three alternative parameter smoothing relationships even tighter than the hyperparameter value of $\delta = 1$ used to generate the estimates in Figure 4.10. A successive tightening of the

parameter smoothing relationships will show which relationship is most consistent with the sample data and which relationship is rejected by the sample data. A fourth model based on $\delta = 1000$ is also estimated to test whether the sample data rejects all three parameter smoothing relationships.

```
% ----- example 4.7 Posterior probabilities for models
% load the Anselin data set
load anselin.data; y = anselin(:,1); nobs = length(y);
x = [ones(nobs,1) anselin(:,2:3)]; [junk nvar] = size(x);
east = anselin(:,4); north = anselin(:,5);
ndraw = 550; nomit = 50; % estimate all three models
prior.ptype = 'contiguity';
prior.rval = 4; prior.dval = 0.5;
res1 = bgwr(y,x,east,north,ndraw,nomit,prior);
prior2.ptype = 'concentric';
prior2.ctr = 20; prior2.rval = 4; prior2.dval = 0.5;
res2 = bgwr(y,x,east,north,ndraw,nomit,prior2);
prior3.ptype = 'distance';
prior3.rval = 4; prior3.dval = 0.5;
```

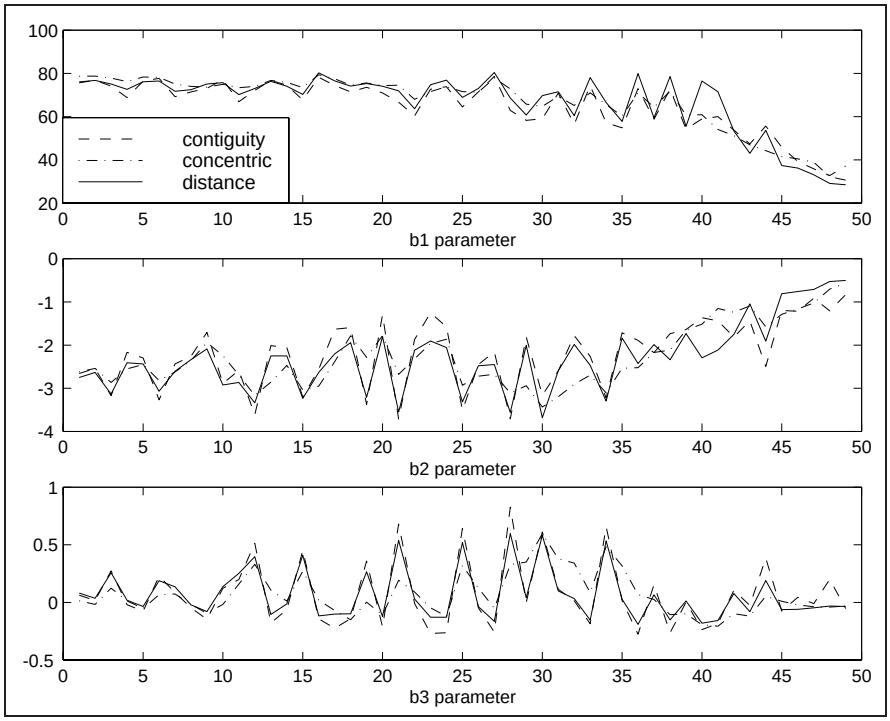

Figure 4.10: Alternative smoothing BGWR estimates

```
res3 = bgwr(y,x,east,north,ndraw,nomit,prior3);
prior4.ptype = 'distance';
prior4.rval = 4; prior4.dval = 1000;
res4 = bgwr(y,x,east,north,ndraw,nomit,prior4);
% compute posterior model probabilities
nmodels = 4;
pp = zeros(nobs,nmodels); lpost = zeros(nobs,nmodels);
lpost(:,1) = res1.logpost;    lpost(:,2) = res2.logpost;
lpost(:,3) = res3.logpost;    lpost(:,4) = res4.logpost;
psum = sum(lpost');
for j=1:nmodels
        pp(:,j) = lpost(:,j)./psum';
end;
% compute posterior means for beta
bb = zeros(nobs,nvar*nmodels);
b1 = res1.bdraw(:,:,1);  bb(:,1) = mean(b1)';
b2 = res1.bdraw(:,:,2);  bb(:,2) = mean(b2)';
b3 = res1.bdraw(:,:,3);  bb(:,3) = mean(b3)';
c1 = res2.bdraw(:,:,1);  bb(:,4) = mean(c1)';
c2 = res2.bdraw(:,:,2);  bb(:,5) = mean(c2)';
c3 = res2.bdraw(:,:,3);  bb(:,6) = mean(c3)';
d1 = res3.bdraw(:,:,1);  bb(:,7) = mean(d1)';
d2 = res3.bdraw(:,:,2);  bb(:,8) = mean(d2)';
d3 = res3.bdraw(:,:,3);  bb(:,9) = mean(d3)';
e1 = res4.bdraw(:,:,1);  bb(:,10) = mean(e1)';
e2 = res4.bdraw(:,:,2);  bb(:,11) = mean(e2)';
e3 = res4.bdraw(:,:,3);  bb(:,12) = mean(e3)';
tt=1:nobs;
plot(tt,pp(:,1),'ok',tt,pp(:,2),'*k',tt,pp(:,3),'+k', tt,pp(:,4),'-');
legend('contiguity','concentric','distance','diffuse');
xlabel('observations'); ylabel('probabilities');
pause;
subplot(3,1,1),
plot(tt,bb(:,1),'-k',tt,bb(:,4),'--k',tt,bb(:,7),'-.',tt,bb(:,10),'+');
legend('contiguity','concentric','distance','diffuse');
xlabel('b1 parameter');
subplot(3,1,2),
plot(tt,bb(:,2),'-k',tt,bb(:,5),'--k',tt,bb(:,8),'-.',tt,bb(:,11),'+');
xlabel('b2 parameter');
subplot(3,1,3),
plot(tt,bb(:,3),'-k',tt,bb(:,6),'--k',tt,bb(:,9),'-.',tt,bb(:,12),'+');
xlabel('b3 parameter');
% produce a Bayesian model averaging set of estimates
bavg = zeros(nobs,nvar); cnt = 1;
for j=1:nmodels
bavg = bavg + matmul(pp(:,j),bb(:,cnt:cnt+nvar-1)); cnt = cnt+nvar;
end;
ttp = tt';
b1out = [ttp bavg(:,1) bb(:,1) bb(:,4) bb(:,7) bb(:,10)];
```

```
in.fmt = strvcat('%4d','%8.2f','%8.2f','%8.2f','%8.2f','%8.2f');
in.cnames = strvcat('Obs','avg','contiguity','concentric',...
'distance','diffuse');
fprintf(1,'constant term parameter \n');
mprint(b1out,in);
b2out = [ttp bavg(:,2) bb(:,2) bb(:,5) bb(:,8) bb(:,11)];
fprintf(1,'household income parameter \n');
mprint(b2out,in);
b3out = [ttp bavg(:,3) bb(:,3) bb(:,6) bb(:,9) bb(:,12)];
in.fmt = strvcat('%4d','%8.3f','%8.3f','%8.3f','%8.3f','%8.3f');
fprintf(1,'house value parameter \n');
mprint(b3out,in);
```

The graphical display of the posterior probabilities produced by the program in example 4.7 are shown in Figure 4.11 and the parameter estimates are shown in Figure 4.12. We see some divergence between the parameter estimates produced by the alternative spatial smoothing priors and the model with no smoothing prior whose estimates are graphed using "+" symbols, but not a dramatic amount. Given the similarity of the parameter estimates, we would expect relatively uniform posterior probabilities for the four models.

In Figure 4.11 the posterior probabilities for the model with no parameter smoothing is graphed as a line to make comparison with the three models that impose parameter smoothing easy. We see certain sample observations where the parameter smoothing relationships produce lower model probabilities than the model without parameter smoothing. For most observations however, the parameter smoothing relationships are relatively consistent with the sample data, producing posterior probabilities above the model with no smoothing relationship. As we would expect, none of the models dominates.

A Bayesian solution to the problem of model specification and choice is to produce a "mixed" or averaged model that relies on the posterior probabilities as weights. We would simply multiply the four sets of coefficient estimates by the four probability vectors to produce a Bayesian model averaging solution to the problem of which estimates are best.

A tabular presentation of this type of result is shown below. The averaged results have the virtue that a single set of estimates are available from which to draw inferences and one can feel comfortable that the inferences are valid for a wide variety of model specifications.

```
constant term parameter
       Obs    average contiguity concentric   distance    diffuse
         1      65.92      62.72      63.53      77.64      57.16
```

2	73.88	70.93	69.09	79.02	76.78
3	78.40	75.45	76.31	79.41	82.78
4	72.94	68.92	69.34	78.35	75.52
5	62.61	64.05	59.64	74.42	49.72
6	73.14	72.44	69.78	78.08	72.32
7	78.72	75.96	79.39	78.53	81.11
8	72.28	70.77	72.37	73.27	72.71
9	75.88	74.12	77.12	75.90	76.40
10	63.02	60.14	67.73	65.27	55.47
11	61.74	57.78	65.78	64.69	55.75
12	59.85	56.30	63.28	63.03	55.18
13	57.39	55.26	61.57	60.29	52.13
14	54.34	53.01	55.23	58.39	51.30
15	52.68	53.36	48.64	57.87	51.69
16	61.30	58.86	64.67	62.91	58.67
17	65.12	61.18	70.64	66.82	60.95
18	71.63	70.19	73.53	72.92	69.68
19	71.78	66.98	76.34	72.72	70.83
20	75.95	72.70	78.37	75.42	77.34
21	71.09	67.98	75.85	71.74	68.54

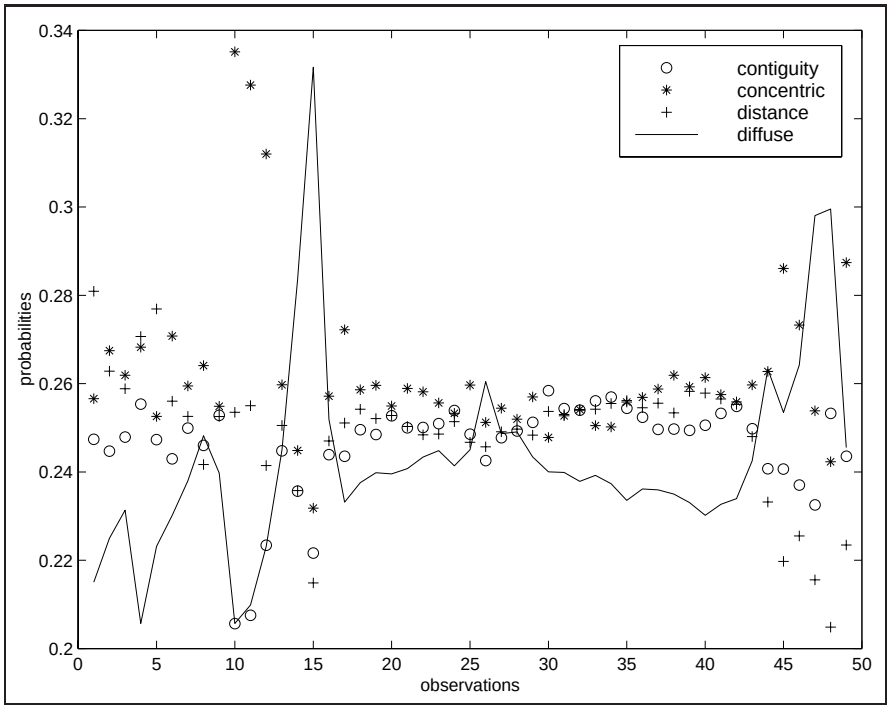

Figure 4.11: Posterior model probabilities

22	69.67	65.65	73.99	71.23	67.62
23	70.48	67.08	74.32	71.02	69.39
24	72.66	70.60	74.26	73.61	72.16
25	68.20	64.11	72.25	69.78	66.48
26	63.00	55.68	65.25	68.75	62.21
27	68.06	61.08	73.99	71.04	65.97
28	66.06	58.06	67.35	72.25	66.56
29	70.67	66.11	74.73	72.47	69.26
30	74.97	71.80	76.25	75.23	76.81
31	74.99	72.23	77.88	74.64	75.23
32	76.53	75.21	78.22	75.86	76.86
33	76.54	75.11	77.90	75.72	77.53
34	75.99	75.03	75.24	75.90	77.92
35	74.54	73.76	76.34	75.27	72.63
36	73.08	72.79	73.09	75.27	71.00
37	75.10	72.84	78.38	76.24	72.66
38	75.99	74.56	76.65	76.88	75.82
39	73.69	72.20	74.50	76.46	71.30
40	71.52	70.39	72.53	75.59	67.05

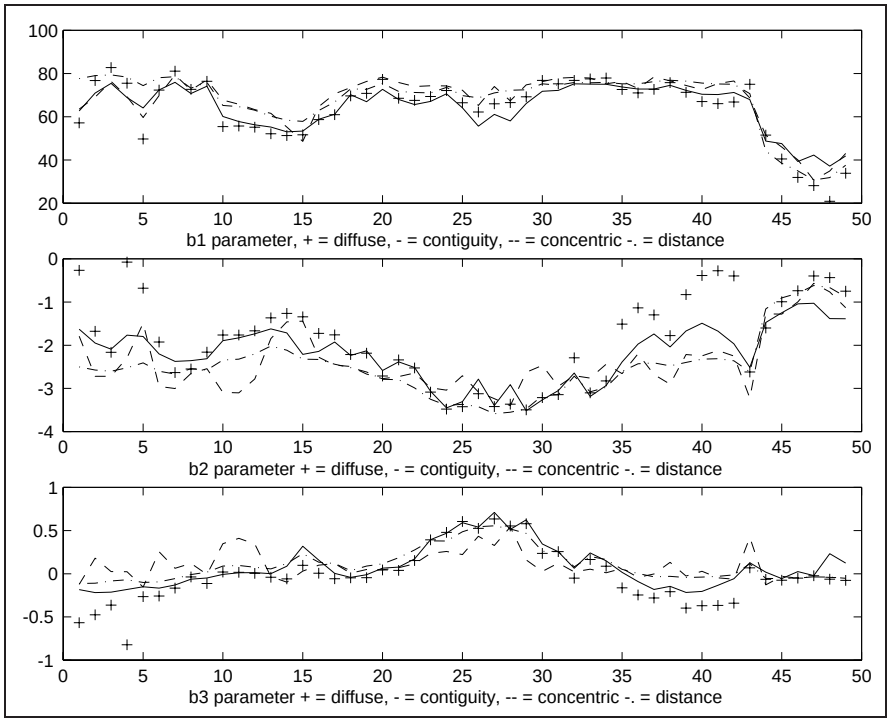

Figure 4.12: Smoothed parameter estimates

41	71.88	70.31	75.41	75.16	66.08
42	72.53	71.24	76.56	75.02	66.81
43	70.53	67.93	68.68	70.68	75.03
44	49.13	48.73	51.45	44.19	51.55
45	43.33	47.61	45.99	38.48	40.45
46	36.60	39.29	40.08	34.98	31.96
47	32.64	42.26	30.70	30.76	28.13
48	30.66	37.17	34.90	31.87	20.91
49	39.28	41.89	43.01	37.57	33.87

household income parameter

Obs	average	contiguity	concentric	distance	diffuse
1	-1.62	-1.63	-1.79	-2.50	-0.27
2	-2.26	-1.95	-2.72	-2.57	-1.68
3	-2.40	-2.09	-2.72	-2.60	-2.16
4	-1.76	-1.77	-2.29	-2.52	-0.07
5	-1.64	-1.80	-1.49	-2.41	-0.68
6	-2.45	-2.20	-2.96	-2.60	-1.92
7	-2.67	-2.38	-3.00	-2.67	-2.63
8	-2.53	-2.36	-2.65	-2.54	-2.55
9	-2.40	-2.31	-2.54	-2.57	-2.16
10	-2.38	-1.89	-3.09	-2.34	-1.76
11	-2.36	-1.83	-3.10	-2.32	-1.76
12	-2.16	-1.74	-2.78	-2.20	-1.67
13	-1.72	-1.62	-1.85	-2.02	-1.37
14	-1.62	-1.71	-1.46	-2.12	-1.26
15	-1.77	-2.21	-1.44	-2.32	-1.34
16	-2.12	-2.14	-2.28	-2.33	-1.73
17	-2.16	-1.93	-2.46	-2.44	-1.76
18	-2.37	-2.24	-2.49	-2.52	-2.22
19	-2.41	-2.13	-2.67	-2.62	-2.19
20	-2.72	-2.59	-2.78	-2.79	-2.71
21	-2.57	-2.38	-2.73	-2.80	-2.34
22	-2.67	-2.52	-2.64	-2.99	-2.53
23	-3.10	-3.06	-2.99	-3.25	-3.09
24	-3.34	-3.45	-3.04	-3.39	-3.48
25	-3.19	-3.31	-2.71	-3.37	-3.42
26	-3.10	-2.79	-3.08	-3.42	-3.12
27	-3.41	-3.40	-3.24	-3.59	-3.42
28	-3.31	-2.91	-3.40	-3.55	-3.36
29	-3.27	-3.52	-2.63	-3.48	-3.50
30	-3.03	-3.26	-2.47	-3.17	-3.21
31	-3.07	-3.05	-2.95	-3.15	-3.14
32	-2.59	-2.64	-2.70	-2.72	-2.29
33	-3.03	-3.19	-2.76	-3.08	-3.10
34	-2.80	-2.93	-2.44	-2.98	-2.83
35	-2.31	-2.39	-2.66	-2.61	-1.51
36	-1.95	-1.97	-2.20	-2.43	-1.14
37	-2.05	-1.74	-2.68	-2.41	-1.30

38	-2.31	-2.04	-2.90	-2.48	-1.78
39	-1.80	-1.67	-2.22	-2.40	-0.83
40	-1.65	-1.49	-2.25	-2.32	-0.39
41	-1.63	-1.67	-2.12	-2.31	-0.28
42	-1.77	-1.97	-2.25	-2.36	-0.40
43	-2.75	-2.51	-3.22	-2.61	-2.62
44	-1.38	-1.47	-1.27	-1.16	-1.60
45	-1.12	-1.25	-1.28	-0.90	-0.99
46	-0.90	-1.04	-1.00	-0.80	-0.74
47	-0.63	-1.03	-0.57	-0.62	-0.40
48	-0.80	-1.38	-0.76	-0.67	-0.44
49	-1.04	-1.39	-1.13	-0.88	-0.75

house value parameter

Obs	average	contiguity	concentric	distance	diffuse
1	-0.230	-0.183	-0.124	-0.113	-0.566
2	-0.141	-0.218	0.181	-0.110	-0.475
3	-0.152	-0.213	0.026	-0.084	-0.364
4	-0.229	-0.183	0.023	-0.071	-0.823
5	-0.163	-0.150	-0.154	-0.101	-0.265
6	-0.055	-0.167	0.255	-0.094	-0.258
7	-0.070	-0.130	0.063	-0.056	-0.166
8	0.005	-0.063	0.120	-0.010	-0.037
9	-0.037	-0.048	-0.008	0.018	-0.113
10	0.142	-0.008	0.348	0.089	0.020
11	0.164	0.010	0.409	0.096	0.017
12	0.129	0.013	0.345	0.075	0.004
13	-0.005	0.001	-0.033	0.056	-0.041
14	0.008	0.088	-0.097	0.118	-0.059
15	0.160	0.318	0.030	0.232	0.097
16	0.098	0.157	0.099	0.134	0.006
17	0.049	0.004	0.118	0.118	-0.057
18	-0.011	-0.042	0.007	0.036	-0.048
19	0.021	-0.012	0.049	0.089	-0.046
20	0.068	0.065	0.045	0.115	0.044
21	0.093	0.077	0.070	0.186	0.039
22	0.178	0.172	0.115	0.275	0.153
23	0.350	0.395	0.236	0.380	0.393
24	0.395	0.468	0.257	0.379	0.479
25	0.472	0.591	0.222	0.485	0.604
26	0.510	0.539	0.434	0.544	0.526
27	0.555	0.711	0.327	0.555	0.635
28	0.532	0.514	0.541	0.519	0.554
29	0.455	0.624	0.160	0.468	0.578
30	0.217	0.344	0.026	0.259	0.235
31	0.223	0.255	0.124	0.258	0.256
32	0.030	0.062	0.021	0.080	-0.051
33	0.165	0.241	0.052	0.200	0.164
34	0.104	0.149	0.011	0.165	0.086

35	-0.005	0.017	0.061	0.051	-0.164
36	-0.091	-0.090	-0.043	0.002	-0.246
37	-0.122	-0.183	-0.003	-0.035	-0.280
38	-0.059	-0.146	0.131	-0.030	-0.209
39	-0.165	-0.216	-0.027	-0.044	-0.399
40	-0.139	-0.205	0.029	-0.040	-0.370
41	-0.139	-0.133	-0.048	-0.030	-0.367
42	-0.114	-0.057	-0.063	-0.013	-0.341
43	0.186	0.127	0.420	0.116	0.068
44	-0.059	0.018	-0.128	-0.055	-0.065
45	-0.054	-0.057	-0.031	-0.056	-0.076
46	-0.032	0.025	-0.052	-0.045	-0.052
47	-0.029	-0.022	-0.036	-0.037	-0.023
48	0.022	0.231	-0.042	-0.039	-0.062
49	-0.014	0.124	-0.047	-0.052	-0.076

4.6 Chapter Summary

We have seen that locally linear regression models can be estimated using distance weighted sub-samples of the observations to produce different estimates for every point in space. This approach can deal with spatial heterogeneity and provide some feel for parameter variation over space in the relationships being explored.

Some problems arise in using spatial expansion models because they tend to produce heteroscedastic disturbances by construction. This problem is overcome to some extent with the DARP model approach.

The non-parametric locally linear regression models produce problems with respect to inferences about the parameters as they vary over space. We saw how a Bayesian approach can provide valid posterior inferences overcome these problems.

The Bayesian GWR model also solves some problems with the non-parametric implementation of the GWR regarding non-constant variance over space or outliers. Given the locally linear nature of the GWR estimates, aberrant observations tend to contaminate whole subsequences of the estimates. The BGWR model robustifies against these observations by automatically detecting and downweighting their influence on the estimates. A further advantage of this approach is that a diagnostic plot can be used to identity observations associated with regions of non-constant variance or spatial outliers.

Finally, an advantage of the Bayesian approach is that is subsumes the spatial expansion, DARP and GWR models as special cases and provides a more flexible implementation by explicitly including a relationship that

describes parameter smoothing over space to the model. Diffuse implementation of the parameter smoothing specification leads to the non-parametric GWR model. In addition to replicating the GWR estimates, the Bayesian model presented here can produce estimates based on parameter smoothing specifications that rely on: distance decay relationships, contiguity relationships, monocentric distance from a central point, or the latitude-longitude locations proposed by Casetti (1972).

4.7 References

Brunsdon, C., A. S. Fotheringham, and M.E. Charlton. (1996) "Geographically weighted regression: A method for exploring spatial nonstationarity", *Geographical Analysis*, Volume 28, pp. 281-298.

Casetti, E. (1972) "Generating Models by the Expansion Method: Applications to Geographic Research", *Geographical Analysis*, December, Volume 4, pp. 81-91.

Casetti, E. (1992) "Bayesian Regression and the Expansion Method", *Geographical Analysis*, January, Volume 24, pp. 58-74.

Casetti, E. (1982) "Drift Analysis of Regression Parameters: An Application to the Investigation of Fertility Development Relations", *Modeling and Simulation 13, Part 3:*, pp. 961-66.

Casetti, E. and A. Can (1998) "The Econometric estimation and testing of DARP models." Paper presented at the RSAI meetings, Sante Fe, New Mexico.

Leamer, Edward. 1983. "Model Choice and Specification Analysis", in *Handbook of Econometrics, Volume 1, Chapter 5*, Zvi Griliches and Michael Intriligator (eds.) (Amsterdam: North-Holland).

LeSage, James P. (1997) "Bayesian Estimation of Spatial Autoregressive Models", *International Regional Science Review*, 1997 Volume 20, number 1&2, pp. 113-129

McMillen, Daniel P. (1996) "One Hundred Fifty Years of Land Values in Chicago: A Nonparameteric Approach," *Journal of Urban Economics*, Vol. 40, pp. 100-124.

McMillen, Daniel P. and John F. McDonald. (1997) "A Nonparametric Analysis of Employment Density in a Polycentric City," *Journal of Regional Science*, Vol. 37, pp. 591-612.

Theil, Henri and Arthur S. Goldberger. (1961) "On Pure and Mixed Statistical Estimation in Economics," *International Economic Review*, Vol. 2, pp. 65-78.

Zellner, Arnold. (1971) *An Introduction to Bayesian Inference in Econometrics*. (New York: John Wiley & Sons.)

Chapter 5

Limited dependent variable models

These models arise when the dependent variable y in our spatial autoregressive model takes values 0, 1, 2, ... representing counts of some event or a coding system for qualitative outcomes. For example, $y = 0$ might represent a coding scheme indicating a lack of highways in our sample of geographical regions, and $y = 1$ denotes the presence of a highway. As another example where the values taken on by y represent counts, we might have $y = 0, 1, 2, \ldots$ denoting the number of foreign direct investment projects in a given county where our sample of observations represent counties for a particular state.

Spatial autoregressive modeling of these data would be interpreted in the framework of a probability model that attempts to describe the $Prob$(event i occurs) $= F(X$: parameters). If the outcomes represent two possibilities, $y = 0, 1$, the model is said to be binary, whereas models with more than two outcomes are referred to as multinomial or polychotomous.

Traditional spatial autoregressive models could be used to carry out a spatial autoregression using the binary response variable $y = 0, 1$, but two problems arise. First, the errors are by construction heteroscedastic. This is because the actual $y = 0, 1$ minus the value $\rho Wy + X\beta = -\rho Wy - X\beta$ or $\iota - \rho Wy - X\beta$. Note also that the heteroscedastic errors are a function of the parameter vector β and ρ. The second problem with using spatial autoregressive models in this setting is that the predicted values can take on values outside the (0,1) interval, which is problematic given a probability model framework. In this setting we would like to see:

$$lim_{\rho Wy+X\beta\rightarrow+\infty} Prob(y=1) = 1 \tag{5.1}$$

$$lim_{\rho Wy+X\beta\rightarrow-\infty} Prob(y=1) = 0 \tag{5.2}$$

Two distributions that have been traditionally used to produce this type of outcome in the case of regression models (that ensures predicted values between zero and one) are the logisitic and normal distributions resulting in the logit model shown in (5.3) and probit model shown in (5.4), where Φ denotes the cumulative normal probability function.

$$Prob(y=1) = e^{X\beta}/(1+e^{X\beta}) \tag{5.3}$$

$$Prob(y=1) = \Phi(X\beta) \tag{5.4}$$

The logistic distribution is similar to the normal except in the tails where it is fatter resembling a Student $t-$distribution. Green (1997) and others indicate that the logistic distribution resembles a $t-$distribution with seven degrees of freedom.

5.1 Introduction

McMillen (1992) proposed methods for estimating SAR and SEM probit models containing spatial heteroscedasticity that rely on the EM algorithm. Aside from McMillen (1992), very little work has appeared regarding spatial autoregressive models that contain binary or polychotomous dependent variables.

McMillen (1995) investigates the impact of heteroscedasticity that is often present in spatial models on probit estimation for non-spatial autoregressive models. McMillen and McDonald (1998) propose a non-parametric locally linear probit method for GWR models of the type discussed in Chapter 4.

Bayesian estimation of logit/probit and tobit variants of spatial autoregressive models that exhibit heteroscedasticity is developed in this chapter. The approach taken draws on work by Chib (1992) and Albert and Chib (1993) as well as the Bayesian estimation of spatial autoregressive models set forth in Chapter 3.

Accounting for heteroscedasticity in logit/probit and tobit models is important because estimates based on the assumption of homoscedasticity in

the presence of heteroscedastic disturbances are inconsistent. The proposed Bayesian estimation methodology overcomes several drawbacks associated with McMillen's (1992) EM approach to estimating these models in the presence of heteroscedastic disturbances.

EM estimation methods rely on an iterative sequencing between the E-step that involves estimation and the M-step that solves a conditional maximization problem. The maximization problem is conditional on parameters determined in the E-step, making it easier to solve than the entire problem involving all of the parameters in the problem.

The approach proposed here extends work of Chib (1992) for the tobit model and Albert and Chib (1993) for the probit model to the case of spatial autoregressive and spatial error models. The basic idea exhibits a similarity to the EM algorithm proposed by McMillen (1992), where the censored or latent unobserved observations on the dependent variable y in the model are replaced by estimated values. Given estimates of the missing y values, the EM algorithm proceeds to estimate the other parameters in the model using methods applied to non-truncated data samples. In other words, conditional on the estimated y-values, the estimation problem is reduced to a non-censored estimation problem which can be solved using maximum likelihood methods.

There are some drawbacks to McMillen's EM estimator that we will overcome using the Bayesian approach set forth in this chapter. One drawback to McMillen's EM estimator is that the information matrix approach to determining measures of precision for the parameter estimates cannot be used. The likelihood function for the heteroscedastic probit model contains a number of integrals equal to the number of observations, so evaluating the likelihood function for these models is impossible. McMillen (1992) overcomes this problem using a non-linear weighted least-squares interpretation of the probit estimator conditional on the spatial lag parameters ρ in the SAR model and λ in the SEM model. This rules out estimates of dispersion for these important parameters. The use of a covariance matrix conditional on the spatial lag parameters produces biased, but consistent confidence intervals that may be too small.

Another problem with McMillen's approach is the need to specify a functional form for the non-constant variance over space. That is, one must specify a model for the noise vector ε such that, $[\text{var}(\varepsilon_i)]^{1/2} = g(Z_i)\gamma$, where g is a continuous, twice differentiable function and Z_i is a vector of explanatory variables for $\text{var}(\varepsilon_i)$. This approach was illustrated by McMillen (1992) for a simple 2-variable model where both of the variables in squared form were used to form the Z_i vector, i.e., $g_i = exp(\gamma_1 X_{1i}^2 + \gamma_2 X_{2i}^2)$. In larger

models a practitioner would need to devote considerable effort to testing and specifying the functional form and variables involved in the model for var(ε_i). Assuming success in finding a few candidate specifications, there is still the problem of inferences that may vary across alternative specifications for the non-constant variance.

We rely on Gibbs sampling to estimate the spatial logit/probit and tobit models. During sampling, we introduce a conditional distribution for the censored or latent observations conditional on all other parameters in the model. This distribution is used to produce a random draw for each censored value of y_i in the case of tobit and for all y_i in the probit model. The conditional distribution for the latent variables takes the form of a normal distribution centered on the predicted value truncated at the right by 0 in the case of tobit, and truncated by 0 from the left and right in the case of probit, for $y_i = 1$ and $y_i = 0$ respectively.

An important difference between the EM approach and the sampling-based approach set forth here is the proof outlined by Gelfand and Smith (1990) that Gibbs sampling from the sequence of complete conditional distributions for all parameters in the model produces a set of draws that converge in the limit to the true (joint) posterior distribution of the parameters. Because of this, we overcome the bias inherent in the EM algorithm's use of conditional distributions. Valid measures of dispersion for all parameters in the model can be constructed from the large sample of parameter draws produced by the Gibbs sampler. Further, if one is interested in linear or non-linear functions of the parameters, these can be constructed using the underlying parameter draws in the linear or non-linear function and then computing the mean over all draws. A valid measure of dispersion for this linear or non-linear combination of the parameters can be based on the distribution of the functional combination of parameters.

The Gibbs sampling approach to estimating the spatial autoregressive models presented in Chapter 3 can be adapted to produce probit and tobit estimates by adding a single conditional distribution for the censored or latent observations. Intuitively, once we have a sample for the unobserved latent dependent variables, the problem reduces to the Bayesian heteroscedastic spatial autoregressive models presented in Chapter 3. The conditional distributions for all other parameters in spatial autoregressive model presented in Chapter 3 remain valid.

Another important advantage of the method proposed here is that heteroscedasticity and outliers can be easily accommodated with the use of the methods outlined in Chapter 3. For the case of the probit model, an interesting interpretation can be given to the family of $t-$distributions that

arise from the methods in Chapter 3 to deal with spatial heterogeneity and spatial outliers. Recall that models involving binary data can rely on any continuous probability function as the probability rule linking fitted probabilities with the binary observations. Probit models arise from a normal probability rule and logit models from a logistic probability rule. When one introduces the latent variables z_i in the probit model to reflect unobserved values based on the binary dependent variables y_i, we have an underlying conditional regression involving z and the usual spatial regression model variables X, W, where X represents the explanatory variables and W denotes the row-standardized spatial weight matrix. The heteroscedastic spatial autoregressive model introduced in Chapter 3 can be viewed in the case of binary dependent variables as a probability rule based on a family of $t-$distributions that represent a mixture of the underlying normal distribution used in the probit regression. (It is well known that the normal distribution can be modeled as a mixture of $t-$distributions, see Albert and Chib, 1993).

The most popular choice of probability rule to relate fitted probabilities with binary data is the logit function corresponding to a logistic distribution for the cumulative density function. Albert and Chib (1993) show that the quantiles of the logistic distribution correspond to a t-distribution around 7 or 8 degrees of freedom. We also know that the normal probability density is similar to a $t-$distribution when the degrees of freedom are large. This allows us to view both the probit and logit models as special cases of the family of models introduced here using a chi-squared prior based on a hyperparameter specifying alternative degrees of freedom to model spatial heterogeneity and outliers.

By using alternative values for the prior hyperparameter that we labeled r in Chapter 3, one can test the sensitivity of the fitted probabilities to alternative distributional choices for the regression model. For example, if we rely on a value of r near 7 or 8, the estimates resulting from the Bayesian version of the heteroscedastic probit model correspond to those one would achieve using a logit model. On the other hand, using a large degrees of freedom parameter, say $r = 50$ would lead to estimates that produce fitted probabilities based on the probit model choice of a normal probability rule. The implication of this is that the heteroscedastic spatial probit model we introduce here represents a more general model than either probit or logit. The generality derives from the family of $t-$distributions associated with alternative values of the hyperparameter r in the model.

A final advantage of the method described here is that estimates of the non-constant variance for each point in space are provided and the prac-

titioner need not specify a functional form for the non-constant variance. Spatial outliers or aberrant observations as well as patterns of spatial heterogeneity will be identified in the normal course of estimation. This represents a considerable improvement over the approach described by McMillen (1992), where a separate model for the non-constant variance needs to be specified.

5.2 The Gibbs sampler

For spatial autoregressive models with uncensored y observations where the error process is homoscedastic and outliers are absent, the computational intensity of the Gibbs sampler is a decided disadvantage over maximum likelihood methods. As demonstrated in Chapter 3, the estimates produced by the Bayesian model estimated with Gibbs sampling are equivalent to those from maximum likelihood in these cases. For the case of probit and tobit models, the Gibbs sampler might be very competitive to the EM algorithm presented in McMillen (1992) because numerous probit or tobit maximum likelihood problems need to be solved to implement the EM method.

Before turning attention to the heteroscedastic version of the Bayesian spatial autoregressive logit/probit and tobit models, consider the case of a homoscedastic spatial autoregressive tobit model where the observed y variable is censored. One can view this model in terms of a latent but unobservable variable z such that values of $z_i < 0$, produce an observed variable $y_i = 0$. Similarly, spatial autoregressive probit models can be associated with a latent variable $z_i < 0$ that produces an observed variable $y_i = 0$ and $z_i \geq 0$ resulting in $y_i = 1$. In both these models, the posterior distribution of z conditional on all other parameters takes the form of truncated normal distribution (see Chib, 1992 and Albert and Chib, 1993).

For the spatial tobit model, the conditional distribution of z_i given all other parameters is a truncated normal distribution constructed by truncating a $N[\tilde{y}_i, \sigma_{ti}^2]$ distribution from the right by zero. Where the predicted value for z_i is denoted by $\tilde{y}_i$ which represents the ith row of $\tilde{y} = B^{-1}X\beta$ for the SAR model and the ith row of $\tilde{y} = X\beta$ for the SEM model. The variance of the prediction is $\sigma_{ti}^2 = \sigma_\varepsilon^2 \sum_j \omega_{ij}^2$, where ω_{ij} denotes the ijth element of $(I_n - \rho W)^{-1}\varepsilon$ for both the SAR and SEM models. The pdf of the latent variables z_i is then:

$$f(z_i|\rho,\beta,\sigma) = \begin{cases} [1-\Phi(\tilde{y}_i/\sigma_{ti})]^{-1}\exp[-(z_i-\tilde{y}_i)^2/2\sigma_{it}], & \text{if } z_i \leq 0 \\ 0 & \text{if } z_i > 0 \end{cases} \tag{5.5}$$

Similarly, for the case of probit, the conditional distribution of z_i given all other parameters is:

$$f(z_i|\rho,\beta,\sigma) \sim \begin{cases} N(\tilde{y}_i,\sigma_{pi}^2), & \text{truncated at the left by 0 if } y_i = 1 \\ N(\tilde{y}_i,\sigma_{pi}^2), & \text{truncated at the right by 0 if } y_i = 0 \end{cases} \tag{5.6}$$

Where $\sigma_{pi}^2 = \sum_j \omega_{ij}^2$, because the probit model is unable to identify both β and σ_ε^2, leading us to scale our problem so σ_ε^2 equals unity. The predicted value $\tilde{y}_i$ takes the same form for the SAR and SEM models as described above for the case of tobit.

The tobit expression (5.5) indicates that we rely on the actual observed y values for non-censored observations and use the sampled latent variables for the unobserved values of y. For the case of the probit model, we replace values of $y_i = 1$ with the sampled normals truncated at the left by 0 and values of $y_i = 0$ with sampled normals truncated at the right by 0.

Given these sampled continuous variables from the conditional distribution of z_i, we can implement the remaining steps of the Gibbs sampler described in Chapter 3 to determine draws from the conditional distributions for ρ, β (and σ in the case of tobit) using the sampled z_i values in place of the censored variables y_i.

5.3 Heteroscedastic models

The models described in this section can be expressed in the form shown in (5.5), where we relax the usual assumption of homogeneity for the disturbances used in SAR, SEM and SAC modeling. Given the discussion in Section 5.2, we can initially assume the existence of non-censored observations on the dependent variable y because these can be replaced with sampled values z as motivated in the previous section.

$$\begin{aligned} y &= \rho W_1 y + X\beta + u \\ u &= \lambda W_2 u + \varepsilon \\ \varepsilon &\sim \mathrm{N}(0,\sigma^2 V), \qquad V = \mathrm{diag}(v_1, v_2, \ldots, v_n) \end{aligned} \tag{5.7}$$

Where $v_i, i = 1, \ldots, n$ represent a set of relative variance parameters to be estimated. We restrict the spatial lag parameters to the interval $1/\mu_{min} < \rho, \lambda < 1/\mu_{max}$.

It should be clear that we need only add one additional step to the Gibbs sampler developed in Chapter 3 for Bayesian heteroscedastic spatial autoregressive models. The additional step will provide truncated draws for the censored or limited dependent variables.

The above reasoning suggest the following Gibbs sampler. Begin with arbitrary values for the parameters $\sigma^0, \beta^0, \rho^0$ and v_i^0, which we designate with the superscript 0. (Keep in mind that we do not sample the conditional distribution for σ in the case of the probit model where this parameter is set to unity.)

1. Calculate $p(\sigma|\rho^0, \beta^0, v_i^0)$, which we use along with a random $\chi^2(n)$ draw to determine σ^1.

2. Calculate $p(\beta|\rho^0, \sigma^1 v_i^0)$ using σ^1 from the previous step. Given the means and variance-covariance structure for β, we carry out a multivariate random draw based on this mean and variance to determine β^1.

3. Calculate $p(v_i|\rho^0, \sigma^1\beta^1)$, which is based on an $n-$vector of random $\chi^2(r+1)$ draws to determine $v_i^1, i = 1, \ldots, n$.

4. Use metropolis within Gibbs sampling to determine ρ^1 as explained in Chapter 3, using the the values σ^1, β^1 and $v_i^1, i = 1, \ldots, n$ determined in the previous steps.

5. Sample the censored y_i observations from a truncated normal centered on the predictive mean and variance determined using $\rho^1, \sigma^1, \beta^1, v_i^1$ (as described in Section 5.2) for the probit and tobit models.

In the above development of the Gibbs sampler we assumed the hyperparameter r that determines the extent to which the disturbances take on a leptokurtic character was known. It is unlikely in practice that investigators would have knowledge regarding this parameter, so an issue that confronts us when attempting to implement the heteroscedastic model is setting the hyperparameter r. As already discussed in Chapter 3, I suggest using a small value near $r = 4$, which produces estimates close to those based on a logit probability rule. If you wish to examine the sensitivity of your inferences to use of a logit versus probit probability rule, you can produce estimates based on a larger value of $r = 30$ for comparison.

5.4 Implementing these models

We have functions **sarp_g** and **sart_g** that carry out Gibbs sampling estimation of the probit and tobit spatial autoregressive models. The documentation for **sarp_g** is:

```
 PURPOSE: Gibbs sampling spatial autoregressive Probit model
          y = p*Wy + Xb + e, e is N(0,sige*V)
          y is a 0,1 vector
          V = diag(v1,v2,...vn), r/vi = ID chi(r)/r, r = Gamma(m,k)
          B = N(c,T),  sige = gamma(nu,d0), p = diffuse prior
-------------------------------------------------
 USAGE: results = sarp_g(y,x,W,ndraw,nomit,prior,start)
 where: y = dependent variable vector (nobs x 1)
        x = independent variables matrix (nobs x nvar)
        W = 1st order contiguity matrix (standardized, row-sums = 1)
    ndraw = # of draws
    nomit = # of initial draws omitted for burn-in
    prior = a structure for:  B = N(c,T),  sige = gamma(nu,d0)
            prior.beta, prior means for beta,   c above (default 0)
            prior.bcov, prior beta covariance , T above (default 1e+12)
            prior.rval, r prior hyperparameter, default=4
            prior.m,    informative Gamma(m,k) prior on r
            prior.k,    (default: not used)
            prior.nu,   a prior parameter for sige
            prior.d0,   (default: diffuse prior for sige)
            prior.rmin = (optional) min rho used in sampling
            prior.rmax = (optional) max rho used in sampling
    start = (optional) structure containing starting values:
            defaults: beta=1,sige=1,rho=0.5, V= ones(n,1)
            start.b   = beta starting values (nvar x 1)
            start.p   = rho starting value   (scalar)
            start.sig = sige starting value  (scalar)
            start.V   = V starting values (n x 1)
-------------------------------------------------
 RETURNS:  a structure:
          results.meth  = 'sarp_g'
          results.bdraw = bhat draws (ndraw-nomit x nvar)
          results.sdraw = sige draws (ndraw-nomit x 1)
          results.vmean = mean of vi draws (1 x nobs)
          results.ymean = mean of y draws (1 x nobs)
          results.rdraw = r draws (ndraw-nomit x 1) (if m,k input)
          results.pdraw = p draws    (ndraw-nomit x 1)
          results.pmean = b prior means, prior.beta from input
          results.pstd  = b prior std deviations sqrt(diag(T))
          results.r     = value of hyperparameter r (if input)
          results.r2mf  = McFadden R-squared
          results.rsqr  = Estrella R-squared
```

```
results.nobs  = # of observations
results.nvar  = # of variables in x-matrix
results.zip   = # of zero y-values
results.ndraw = # of draws
results.nomit = # of initial draws omitted
results.y     = actual observations (nobs x 1)
results.yhat  = predicted values
results.nu    = nu prior parameter
results.d0    = d0 prior parameter
results.time  = time taken for sampling
results.accept= acceptance rate
results.rmax = 1/max eigenvalue of W (or rmax if input)
results.rmin = 1/min eigenvalue of W (or rmin if input)
```

This function documentation and use is very similar to the **sar_g** function from Chapter 3. One difference is in the measures of fit calculated. These are $R-$squared measures that are traditionally used for limited dependent variable models.

Following an example provided by McMillen (1992) for his EM algorithm approach to estimating SAR and SEM probit models, we employ the data set from Anselin (1988) on crime in Columbus, Ohio. McMillen censored the dependent variable on crime such that $y_i = 1$ for values of crime greater than 40 and $y_i = 0$ for values of crime less than or equal to 40. The explanatory variables in the model are neighborhood housing values and neighborhood income. Example 5.1 demonstrates how to implement a spatial probit model Gibbs sampler using the **probit_g** function.

```
% ----- Example 5.1 SAR Probit Model
load anselin.data;
y = anselin(:,1); [n junk] = size(y);
x = [ones(n,1) anselin(:,2:3)];
vnames = strvcat('crime','constant','income','hvalue');
load Wmat.data; W = Wmat;
yc = zeros(n,1);
% now convert the data to 0,1 values
for i=1:n
if y(i,1) > 40.0
yc(i,1) = 1;
end;
end;
ndraw = 1100; nomit = 100;
prior.rval = 4; prior.rmin = 0; prior.rmax = 1;
result = sarp_g(yc,x,W,ndraw,nomit,prior);
prt(result,vnames);
plt(result,vnames);
```

The printed results are shown below and the graphical results provided by the **plt** function are shown in Figure 5.1. For comparison we also present the results from ignoring the limited dependent variable nature of the y variable in this model and using the **sar** function to produce maximum likelihood estimates.

```
Gibbs sampling spatial autoregressive Probit model
Dependent Variable =         crime
McFadden R^2    =    0.4122
Estrella R^2    =    0.5082
sigma^2         =    2.4771
r-value            =      4
Nobs, Nvars     =     49,     3
# 0, 1 y-values =     30,    19
ndraws,nomit    =   1100,   100
acceptance rate =    0.7836
time in secs    =   69.0622
min and max rho =    0.0000,   1.0000
*************************************************************
Variable       Prior Mean    Std Deviation
constant         0.000000   1000000.000000
income           0.000000   1000000.000000
hvalue           0.000000   1000000.000000
*************************************************************
      Posterior Estimates
Variable      Coefficient      t-statistic    t-probability
constant         3.616236        2.408302         0.020092
income          -0.197988       -1.698013         0.096261
hvalue          -0.036941       -1.428174         0.159996
rho              0.322851        2.200875         0.032803

Spatial autoregressive Model Estimates
R-squared       =    0.5216
Rbar-squared    =    0.5008
sigma^2         =    0.1136
Nobs, Nvars     =     49,     3
log-likelihood  =       -1.1890467
# of iterations =     13
min and max rho =   -1.5362,   1.0000
*************************************************************
Variable        Coefficient      t-statistic    t-probability
variable 1         0.679810         2.930130         0.005259
variable 2        -0.019912        -1.779421         0.081778
variable 3        -0.005525        -1.804313         0.077732
rho                0.539201         2.193862         0.033336

Gibbs sampling spatial autoregressive Probit model
Dependent Variable =         crime
```

```
McFadden R^2    =    0.3706
Estrella R^2    =    0.4611
sigma^2         =    2.2429
r-value            =     40
Nobs, Nvars     =     49,     3
# 0, 1 y-values =     30,    19
ndraws,nomit    =   1100,   100
acceptance rate =    0.9616
time in secs    =   70.3423
min and max rho =    0.0000,   1.0000
*************************************************************
Variable       Prior Mean    Std Deviation
constant         0.000000   1000000.000000
income           0.000000   1000000.000000
hvalue           0.000000   1000000.000000
*************************************************************
      Posterior Estimates
Variable      Coefficient      t-statistic    t-probability
constant         4.976554         2.408581         0.020078
income          -0.259836        -2.047436         0.046351
hvalue          -0.053615        -1.983686         0.053278
rho              0.411042         3.285293         0.001954
```

There is a remarkable difference between the maximum likelihood estimates that ignore the limited dependent nature of the variable y which we would expect. To explore the difference between 'logit' and 'probit' estimates, we produced a set of Bayesian estimates based on a hyperparameter value of $r = 40$, which would correspond to the Probit model. Green (1997) states that the issue of which distributional form should be used on applied econometric problems is unresolved. He further indicates that inferences from either logit or probit models are often the same. This does not appear to be the case for the data set in example 5.1, where we see a difference in both the magnitude and significance of the parameters from the models based on $r = 4$ and $r = 40$.

A final point with respect to interpreting the estimates is that the marginal impacts of the variables on the fitted probabilities is usually the inference aim of these models. The estimates would need to be converted according to the probability rule implied by the alternative underlying $t-$distributions associated with the r value employed. These marginal impacts would be very similar (as in the case of non-spatial maximum likelihood logit versus probit marginal impacts) despite the apparent differences in the coefficients. For the purpose of computing marginal impacts, one needs to evaluate the posterior density of p_k, which we denote $\hat{\pi}(p_k)$ for k ranging over all sample observations. (It is conventional to assess marginal impacts across all obser-

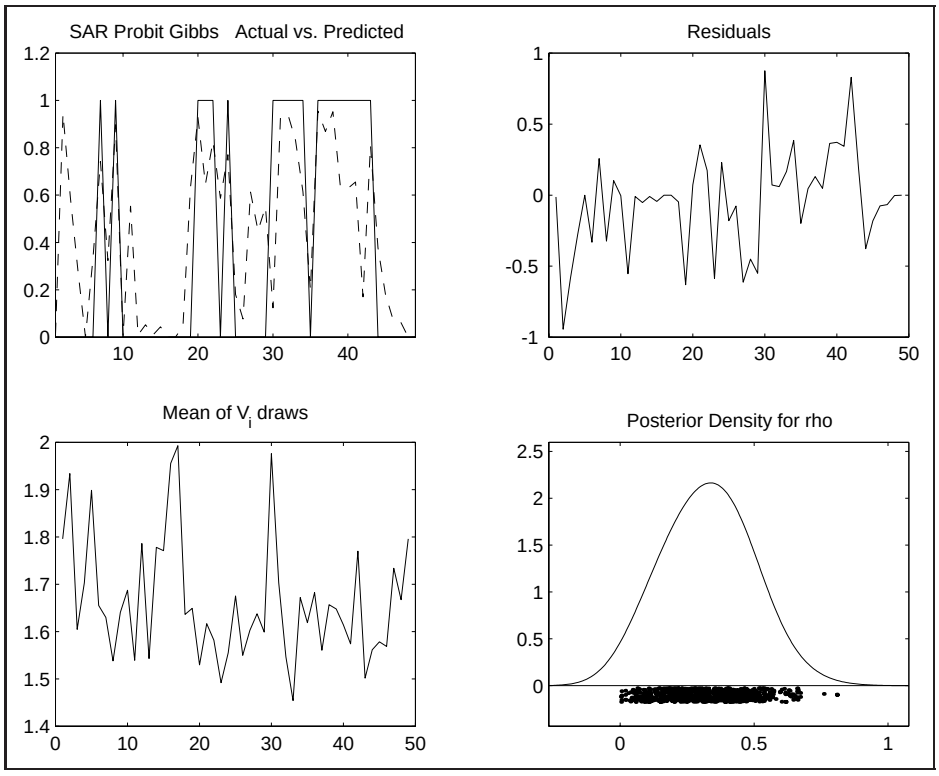

Figure 5.1: Results of plt() function

vations in the model and then average these.) For the heteroscedastic model p_k will take the form: $\Phi(\tilde{y}_k) = \Phi(\rho W_k v_k^{(-1/2)} y + v_k^{(-1/2)} x_k' \beta)$. To find the posterior density estimate of $\hat{\pi}(p_k)$ we would employ the draws for ρ^i, v_k^i, β^i in normal densities, denoted $\mathrm{N}(\mu, \sigma)$ as indicated in (5.8)

$$\begin{aligned} \hat{\pi}(p_k) &= (1/m) \sum_{i=1}^{m} \mathrm{N}[\tilde{y}_k^i, (1/v_k^i) x_k' (X' V_i^{-1} X)^{-1} x_k] / \mathrm{N}[0,1] \qquad (5.8) \\ \tilde{y}_k^i &= \rho^i W_k (1/\sqrt{v_k^i}) y + (1/\sqrt{v_k^i}) x_k' \beta^i \\ V_i^{-1} &= diag(1/v_j^i), j = 1, \ldots, n \end{aligned}$$

Table 5.1 shows a comparison of McMillen's EM algorithm estimates and those from Gibbs sampling. The Gibbs estimates are based on 1,100 draws with the first 100 discarded for startup or burn-in. Gibbs SAR and SEM

estimates are reported for both $r = 4$ and $r = 40$. Results for two values of r were reported because the inferences are different from these two models. It should be noted that there is a tendency of the consistent EM estimates of dispersion to overstate the precision, producing generally larger $t-$statistics for the EM versus Gibbs estimates.

Table 5.1: EM versus Gibbs estimates

	EM SAR	Gibbs SAR $r = 4$	Gibbs SAR $r = 40$	EM SEM	Gibbs SEM $r = 4$	Gibbs $r = 40$
CONSTANT	2.587	3.758	4.976	2.227	3.925	2.710
$t-$value	2.912	2.150	2.408	3.115	1.936	2.168
INCOME	-0.128	-0.213	-0.259	-0.123	-0.214	-0.143
$t-$value	-2.137	-1.583	-2.047	-2.422	-1.636	-1.719
HOUSING	-0.029	-0.037	-0.053	-0.025	-0.048	-0.032
$t-$value	-1.617	-1.416	-1.983	-1.586	-1.439	-1.791
ρ	0.429	0.325	0.411	0.279	0.311	0.315
$t-$value	—	2.175	3.285	—	1.766	1.796

Another reason why the EM and Gibbs estimates may be different is that McMillen's approach requires that a model for the non-constant variance be specified. The specification used by McMillen was: $v_i = 0.0007\text{INCOME}^2 + 0.0004\text{HOUSING}^2$. This is quite different from the approach taken in the Bayesian model that relies on v_i estimates from Gibbs sampling.

There are functions for carrying out Bayesian Probit and Tobit versions of all spatial autoregressive models. SAR models are implemented by **sarp_g** and **sart_g**, SEM models by **semp_g** and **semt_g**, SAC models by **sacp_g** and **sact_g**.

Tobit models can involve either left or right truncation. That is censored observations can be y values that fall below a limiting value or censoring can take place above a limit value. The functions **sart_g, sact_g** and **semt_g** allow the user to specify the type of censoring and a limit value. This defaults to the typical case of left censoring at zero. The documentation for the function **sart_g** is:

```
 PURPOSE: Gibbs sampling spatial autoregressive Tobit model
          y = p*Wy + Xb + e, e is N(0,sige*V)
          y is a censored vector (assumed to be at zero)
          V = diag(v1,v2,...vn), r/vi = ID chi(r)/r, r = Gamma(m,k)
          B = N(c,T),  sige = gamma(nu,d0), p = diffuse prior
---------------------------------------------------
 USAGE: results = sart_g(y,x,W,prior,ndraw,nomit,start)
 where: y = dependent variable vector (nobs x 1)
```

```
        x = independent variables matrix (nobs x nvar)
        W = 1st order contiguity matrix (standardized, row-sums = 1)
    ndraw = # of draws
    nomit = # of initial draws omitted for burn-in
    prior = a structure for:  B = N(c,T),  sige = gamma(nu,d0)
            prior.beta, prior means for beta,   c above (default 0)
            prior.bcov, prior beta covariance , T above (default 1e+12)
            prior.rval, r prior hyperparameter, default=4
            prior.m,    informative Gamma(m,k) prior on r
            prior.k,    (default: not used)
            prior.nu,    a prior parameter for sige
            prior.d0,    (default: diffuse prior for sige)
            prior.trunc = 'left' or 'right' censoring (default = left)
            prior.limit = value for censoring (default = 0)
    start = (optional) structure containing starting values:
            defaults: beta=1,sige=1,rho=0.5, V= ones(n,1)
            start.b   = beta starting values (nvar x 1)
            start.p   = rho starting value   (scalar)
            start.sig = sige starting value  (scalar)
            start.V   = V starting values (n x 1)
---------------------------------------------------
 NOTE:  1st column of x-matrix must contain iota vector (constant term)
---------------------------------------------------
 RETURNS:  a structure:
          results.meth  = 'sart_g'
          results.bdraw = bhat draws (ndraw-nomit x nvar)
          results.sdraw = sige draws (ndraw-nomit x 1)
          results.vmean = mean of vi draws (1 x nobs)
          results.rdraw = sige draws (ndraw-nomit x 1)
          results.pdraw = p draws    (ndraw-nomit x 1)
          results.ymean = mean of y draws (1 x nobs)
          results.pmean = b prior means, prior.beta from input
          results.pstd  = b prior std deviations sqrt(diag(T))
          results.r     = value of hyperparameter r (if input)
          results.nobs  = # of observations
          results.nvar  = # of variables in x-matrix
          results.nobsc = # of censored y-values
          results.ndraw = # of draws
          results.nomit = # of initial draws omitted
          results.y     = actual observations (nobs x 1)
          results.yhat  = predicted values
          results.nu    = nu prior parameter
          results.d0    = d0 prior parameter
          results.time  = time taken for sampling
          results.accept= acceptance rate
          results.rmax = 1/max eigenvalue of W (or rmax if input)
          results.rmin = 1/min eigenvalue of W (or rmin if input)
```

To illustrate the case of Tobit model estimation, example 5.2 shows an example where we generate an SAR model based on the Anselin neighborhood crime spatial contiguity matrix. We censor observations that are less than zero. Estimates based on the uncensored data using the function **sar_g** are compared to those from using **sar_g** and **sart_g** on the censored data. We would expect that ignoring censoring should produce poor estimates from application of **sar_g** on the censored data whereas use of **sart_g** would produce better estimates that are closer to those from application of **sar_g** to the uncensored data.

A vector y is generated based on the neighborhood crime independent variables standardized. This standardization produces a relatively even number of negative and positive values for y. Negative values are then censored.

```
% ----- Example 5.2 SAR Tobit Model
load anselin.dat;
xt = [anselin(:,2:3)]; n = length(xt);
% center and scale the data so our y-values
% are evenly distributed around zero, the censoring point
x = [ones(n,1) studentize(xt)];
[n k] = size(x);
load wmat.dat;        W = wmat;
sige = 5.0;           evec = randn(n,1)*sqrt(sige);
rho = 0.75;           beta = ones(k,1);
B = eye(n) - rho*W;   BI = inv(B);
y = BI*x*beta + BI*evec;
yc = y;
% now censor neighborhoods with crime < 1
for i=1:n
if y(i,1) < 1
yc(i,1) = 1;
end;
end;
Vnames = strvcat('crime','constant','income','hvalue');
ndraw = 600; nomit = 100;
prior.rval = 30;
res1 = sar(y,x,W);
res2 = sar_g(yc,x,W,ndraw,nomit,prior);
prior.limit = 1;
prior.trunc = 'left';
res3 = sart_g(yc,x,W,ndraw,nomit,prior);
prt(res1,Vnames);
prt(res2,Vnames);
prt(res3,Vnames);
```

The printed results for an SAR model based on the uncensored data as well as a set of estimates that ignore the sample censoring and the Tobit version of the SAR model are presented below.

```
Spatial autoregressive Model Estimates
Dependent Variable =         crime
R-squared       =    0.7344
Rbar-squared    =    0.7229
sigma^2         =    5.6564
Nobs, Nvars     =     49,     3
log-likelihood  =       -99.041858
# of iterations =     13
min and max rho =   -1.5362,   1.0000
***************************************************************
Variable      Coefficient      t-statistic    t-probability
constant         0.617226         1.328646         0.190519
income           1.064967         2.271643         0.027831
hvalue           0.897449         2.254232         0.028988
rho              0.724180         5.068285         0.000007

Gibbs sampling spatial autoregressive model
Dependent Variable =         crime
R-squared       =    0.7013
sigma^2         =    3.4570
r-value         =     30
Nobs, Nvars     =     49,     3
ndraws,nomit    =    600,   100
acceptance rate =    0.9662
time in secs    =   14.4129
min and max rho =   -1.5362,   1.0000
***************************************************************
Variable       Prior Mean    Std Deviation
constant         0.000000   1000000.000000
income           0.000000   1000000.000000
hvalue           0.000000   1000000.000000
***************************************************************
      Posterior Estimates
Variable      Coefficient      t-statistic    t-probability
constant         1.412658         3.037593         0.003922
income           1.253225         3.479454         0.001111
hvalue           0.364549         1.267454         0.211372
rho              0.599010         6.097006         0.000000

Gibbs sampling spatial autoregressive Tobit model
Dependent Variable =         crime
R-squared          =    0.6977
sigma^2            =    6.9121
r-value            =     30
Nobs, Nvars        =     49,     3
```

```
# censored values  =     16
ndraws,nomit       =    600,   100
acceptance rate    =    0.9120
time in secs       =   19.7692
min and max rho    =   -1.5362,   1.0000
*************************************************************
Variable       Prior Mean    Std Deviation
constant         0.000000   1000000.000000
income           0.000000   1000000.000000
hvalue           0.000000   1000000.000000
*************************************************************
      Posterior Estimates
Variable      Coefficient      t-statistic    t-probability
constant         0.936753         1.847863         0.071058
income           1.479955         2.844135         0.006624
hvalue           0.544580         1.320079         0.193340
rho              0.629394         6.446799         0.000000
```

A key to understanding how the Gibbs sampler works on these problems is the generation of predicted values for the censored observations. These values may also be useful for purposes of inference regarding the censored observations. The tobit spatial autoregressive functions return a structure variable field 'results.ymean' that represents the mean of the sampled values for the censored observations as well as the actual values of the uncensored observations. Figure 5.2 shows a plot of this data vector against the actual y variables. Ordinarily, we wouldn't know the values of the censored y values, but in this case because we generated the data set and then censored the observations we have this information.

5.5 An applied example

The Harrison and Rubinfeld Boston data set used in Chapter 3 to illustrate Gibbs sampling Bayesian spatial autoregressive models contains censored values. Median house values greater than \$50,000 were set equal to 50,000 for 16 of the 506 sample observations (see Gilley and Pace, 1995). This provides an opportunity to see if using tobit estimation to take the sample truncation into account produces different parameter estimates.

Example 5.3 reads the Boston data set and sorts by median housing values. Note that we must also sort the explanatory variables using the index vector 'yind' returned by the sort for the y values as well as the latitude and longitude vectors. After carrying out Gibbs sampling estimation for the SAR, SEM and SAC models, we add to the prior structure variable a field for right-truncation and we supply the limit value which is the log of 50,000

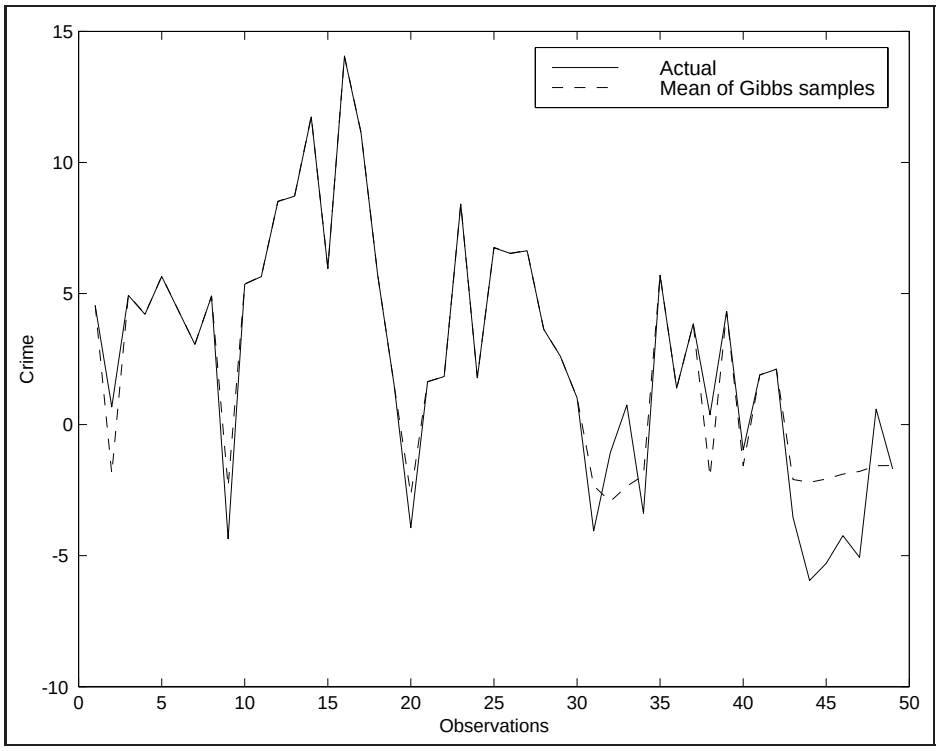

Figure 5.2: Actual vs. simulated censored y-values

standardized. By virtue of our sorting the y vector, this transformed limit value must equal the last 16 observations, so we use the last to define the limit value.

```
% ----- Example 5.3 Right-censored Tobit for the Boston data
load boston.raw; % Harrison-Rubinfeld data
load latitude.data; load longitude.data;
[n k] = size(boston);y = boston(:,k);     % median house values
% sort by median house values
[ys yind] = sort(y); xs = boston(yind,1:k-1);
lats = latitude(yin,1); lons = longitude(yin,1);
[W1 W W3] = xy2cont(lats,lons); % create W-matrix
vnames = strvcat('hprice','constant','crime','zoning','industry', ...
'charlesr','noxsq','rooms2','houseage','distance','access','taxrate', ...
'pupil/teacher','blackpop','lowclass');
y = studentize(log(ys)); x = [ones(n,1) studentize(xs)];
% define censoring limit
limit = y(506,1); % median values >=50,000 are censored to 50
```

```
ndraw = 1100; nomit = 100;
prior.rval = 4;
prior.rmin = 0; prior.rmax = 1;
prior.lmin = 0; prior.lmax = 1;
% ignore censoring
res1 = sar_g(y,x,W,ndraw,nomit,prior);
prt(res1,vnames);
res2 = sem_g(y,x,W,ndraw,nomit,prior);
prt(res2,vnames);
res3 = sac_g(y,x,W,W,ndraw,nomit,prior);
prt(res3,vnames);
% use Tobit for censoring
prior.trunc = 'right';
prior.limit = limit;
res4 = sart_g(y,x,W,ndraw,nomit,prior);
prt(res4,vnames);
res5 = semt_g(y,x,W,ndraw,nomit,prior);
prt(res5,vnames);
res6 = sact_g(y,x,W,W,ndraw,nomit,prior);
prt(res6,vnames);
```

Intuitively, we might not expect a large difference in the parameter estimates for this case where only 16 of the 506 sample observations are censored. The results are presented below, where the information that is typically printed regarding prior means and standard deviations has been eliminated because we used a diffuse prior. The order of the results has been ordered to present both SAR and the tobit SAR, then the SEM and tobit SEM and finally SAC and tobit SAC estimates.

```
Gibbs sampling spatial autoregressive model
Dependent Variable =    hprice
R-squared       =    0.8243
sigma^2         =    0.1921
r-value         =      4
Nobs, Nvars     =    506,    14
ndraws,nomit    =   1100,   100
acceptance rate =    0.8662
time in secs    =  129.8548
min and max rho =    0.0000,   1.0000
***************************************************************
      Posterior Estimates
Variable          Coefficient      t-statistic    t-probability
constant            -0.025169       -1.126015         0.260708
crime               -0.152583       -3.509177         0.000491
zoning               0.050736        1.304237         0.192763
industry             0.045046        0.915162         0.360555
charlesr             0.020157        0.787959         0.431100
```

```
noxsq               -0.089610       -1.803644        0.071899
rooms2               0.267168        5.731838        0.000000
houseage            -0.036438       -0.878105        0.380315
distance            -0.178140       -5.673916        0.000000
access               0.188203        2.713291        0.006895
taxrate             -0.212748       -5.525728        0.000000
pupil/teacher       -0.117601       -3.980184        0.000079
blackpop             0.107424        3.873596        0.000122
lowclass            -0.313225       -7.068474        0.000000
rho                  0.314435        3.212364        0.001403

Gibbs sampling spatial autoregressive Tobit model
Dependent Variable =    hprice
R-squared          =    0.8225
sigma^2            =    0.1595
r-value            =      4
Nobs, Nvars        =    506,    14
# censored values  =     16
ndraws,nomit       =   1100,   100
acceptance rate    =    0.9206
time in secs       =  158.1523
min and max rho    =    0.0000,   1.0000
*************************************************************
      Posterior Estimates
Variable           Coefficient      t-statistic    t-probability
constant             -0.034993       -1.763936        0.078363
crime                -0.159346       -3.631451        0.000311
zoning               0.051875        1.399918        0.162168
industry             0.025951        0.561893        0.574445
charlesr             0.010791        0.446021        0.655778
noxsq                -0.084991       -1.800372        0.072414
rooms2               0.240183        5.251582        0.000000
houseage             -0.048693       -1.265013        0.206465
distance             -0.176466       -5.657902        0.000000
access               0.193611        3.125944        0.001877
taxrate              -0.219662       -6.201483        0.000000
pupil/teacher        -0.115413       -4.128876        0.000043
blackpop             0.107766        4.064385        0.000056
lowclass             -0.301859       -7.543143        0.000000
rho                  0.296382        2.937506        0.003464

Gibbs sampling spatial error model
Dependent Variable =    hprice
R-squared          =    0.7304
sigma^2            =    0.1445
r-value            =      4
Nobs, Nvars        =    506,    14
ndraws,nomit       =   1100,   100
acceptance rate    =    0.4870
```

```
time in secs       =  114.7631
min and max lambda =    0.0000,   1.0000
************************************************************
      Posterior Estimates
Variable           Coefficient      t-statistic    t-probability
constant             -0.039799        -0.420595         0.674234
crime                -0.165742        -4.010247         0.000070
zoning                0.049197         1.177168         0.239698
industry             -0.005192        -0.087059         0.930660
charlesr             -0.015074        -0.579339         0.562625
noxsq                -0.147566        -1.925225         0.054777
rooms2                0.338988         8.416259         0.000000
houseage             -0.127886        -2.885708         0.004077
distance             -0.175342        -2.494614         0.012936
access                0.276185         3.014877         0.002704
taxrate              -0.237284        -4.804194         0.000002
pupil/teacher        -0.084627        -2.720154         0.006756
blackpop              0.144584         4.584164         0.000006
lowclass             -0.243651        -5.941416         0.000000
lambda                0.786425        19.084355         0.000000

Gibbs sampling spatial error Tobit model
Dependent Variable =    hprice
R-squared          =    0.7313
sigma^2            =    0.1493
r-value            =      4
Nobs, Nvars        =    506,    14
# censored values  =     16
ndraws,nomit       =   1100,   100
acceptance rate    =    0.4808
time in secs       =  142.1482
min and max lambda =    0.0000,   1.0000
************************************************************
      Posterior Estimates
Variable           Coefficient      t-statistic    t-probability
constant             -0.039910        -0.544675         0.586224
crime                -0.155688        -4.035574         0.000063
zoning                0.049964         1.139609         0.255004
industry             -0.013470        -0.220422         0.825634
charlesr             -0.019834        -0.649249         0.516481
noxsq                -0.108960        -1.401207         0.161783
rooms2                0.277678         5.450260         0.000000
houseage             -0.116271        -2.463411         0.014103
distance             -0.161768        -2.319974         0.020751
access                0.255551         2.665245         0.007946
taxrate              -0.241885        -4.658895         0.000004
pupil/teacher        -0.088207        -2.694077         0.007300
blackpop              0.134053         4.262707         0.000024
lowclass             -0.259032        -5.544375         0.000000
```

```
lambda               0.728140        12.943930         0.000000

Gibbs sampling general spatial model
Dependent Variable =    hprice
R-squared          =    0.8601
sigma^2            =    0.1468
r-value            =      4
Nobs, Nvars        =    506,    14
ndraws,nomit       =   1100,   100
accept rho rate    =    0.9857
accept lam rate    =    0.5040
time in secs       =  214.1628
min and max rho    =    0.0000,   1.0000
min and max lambda =    0.0000,   1.0000
************************************************************
      Posterior Estimates
Variable           Coefficient      t-statistic    t-probability
constant             -0.026721        -0.556941         0.577821
crime                -0.167938        -4.414590         0.000012
zoning                0.047543         1.179076         0.238938
industry              0.016986         0.299507         0.764680
charlesr              0.009328         0.343660         0.731249
noxsq                -0.128901        -1.944391         0.052418
rooms2                0.330110         7.369387         0.000000
houseage             -0.099118        -2.108214         0.035518
distance             -0.191117        -3.638851         0.000303
access                0.235348         2.793913         0.005411
taxrate              -0.232871        -4.977616         0.000001
pupil/teacher        -0.108696        -3.825547         0.000147
blackpop              0.133964         4.398670         0.000013
lowclass             -0.297188        -7.347890         0.000000
rho                   0.717992        13.037376         0.000000
lambda                0.083388         1.815698         0.070025

Gibbs sampling general spatial Tobit model
Dependent Variable =    hprice
R-squared          =    0.8602
sigma^2            =    0.1333
r-value            =      4
Nobs, Nvars        =    506,    14
# censored values  =     16
ndraws,nomit       =   1100,   100
accept rho rate    =    0.9839
accept lam rate    =    0.7113
time in secs       =  249.0965
min and max rho    =    0.0000,   1.0000
min and max lambda =    0.0000,   1.0000
************************************************************
      Posterior Estimates
```

Variable	Coefficient	t-statistic	t-probability
constant	-0.040444	-0.967258	0.333890
crime	-0.155084	-4.123056	0.000044
zoning	0.045257	1.152237	0.249783
industry	0.005911	0.111250	0.911463
charlesr	-0.003574	-0.144284	0.885335
noxsq	-0.106601	-1.739330	0.082602
rooms2	0.292600	6.640997	0.000000
houseage	-0.104126	-2.252888	0.024706
distance	-0.173827	-3.428468	0.000658
access	0.214266	2.627556	0.008869
taxrate	-0.239525	-5.153612	0.000000
pupil/teacher	-0.110112	-4.067087	0.000055
blackpop	0.131408	4.760042	0.000003
lowclass	-0.283951	-6.968683	0.000000
rho	0.666444	9.336428	0.000000
lambda	0.100070	1.795139	0.073245

Contrary to our expectation that these two sets of estimates would produce identical inferences, an interesting and perhaps substantive conclusion arises. In comparing the estimates that ignore sample censoring to the tobit estimates we find further evidence regarding the 'noxsq' air pollution variable. For all of the tobit models, this variable is less significant than for the non-tobit models. Recall that in Chapter 3 we found that maximum likelihood estimates produced estimates for this variable that were significantly different from zero for all three spatial autoregressive models at the traditional 5% level. After introducing the Bayesian heteroscedastic spatial models we found estimates that were not significantly different from zero at the 5% level, but still significant at the 10% level. After introducing tobit variants of the Bayesian heteroscedastic spatial models we find further movement away from significance for this variable. In the case of the SEM model we find that 'noxsq' has a marginal probability of 0.16. Recall that the SEM and SAC models were judged to be most appropriate for this data set.

It is especially interesting that none of the other variables in the models change in significance or magnitude by very much. This is as we would expect, given the small number (16) of censored observations in a relatively large sample (506).

5.6 Chapter Summary

A Gibbs sampling approach to estimating heteroscedastic spatial autoregressive and spatial error probit and tobit models was presented. With the

exception of McMillen (1992) who set forth an EM algorithm approach to estimating spatial autoregressive models in the presence of heteroscedastic disturbances, no other methods exist for producing estimates under these conditions. It was argued that the Bayesian approach set forth here has several advantages over the EM algorithm approach suggested by McMillen (1992). First, the method produces posterior distributions for all parameters in the model whereas McMillen's approach does not provide estimates of precision for the spatial parameters ρ and λ.. The posteriors allow for inferences regarding the mean and dispersion of all parameters, including the important spatial lag

A second advantage is that the Gibbs sampled measures of dispersion based on the posterior distributions are valid whereas the EM algorithm produces consistent estimates of dispersion that are likely to overstate parameter precision. Some evidence of overstatement was in fact found in the results in Table 5.1.

Perhaps the greatest advantage of the Bayesian approach introduced here is that no model for the non-constant variance need be specified by the investigator. The Gibbs sampling approach produces estimates of the non-constant variance for every observation in space. These estimates can be used to draw inferences regarding the presence of spatial outliers or general patterns of non-constant variance over space.

Another point is that the EM methods introduced in McMillen do not apply to tobit models where the likelihood function takes a more complicated form than the probit model. The Gibbs sampling approach introduced here applies to the tobit model as well as probit and is equally easy to implement. In fact, the Gibbs sampling approach to estimating the heteroscedastic spatial probit model introduced here subsumes a logit version of the model as a special case.

Finally, because the approach introduced here is quite similar to the Gibbs sampling approach for spatial autoregressive and spatial error models presented in Chapter 3, it provides a unified methodology for estimating spatial models that involve continuous or dichotomous dependent variables.

5.7 References

Albert, James H. and Siddhartha Chib (1993), "Bayesian Analysis of Binary and Polychotomous Response Data", *Journal of the American Statistical Association*, Volume 88, number 422, pp. 669-679.

Anselin, Luc (1988). *Spatial Econometrics: Methods and Models*,

(Dordrecht: Kluwer Academic Publishers).

Anselin, Luc and D.A. Griffith (1988). "Do spatial effects really matter in regression analysis? *Papers of the Regional Science Association*, 65, pp. 11-34.

Chib, Siddhartha (1992), "Bayes Inference in the Tobit Censored Regression Model", *Journal of Econometrics*, Volume 51, pp. 79-99.

Gelfand, Alan E., and A.F.M Smith. 1990. "Sampling-Based Approaches to Calculating Marginal Densities", *Journal of the American Statistical Association*, Vol. 85, pp. 398-409.

Geweke, John (1991) "Evaluating the Accuracy of Sampling-Based Approaches to the Calculation of Posterior Moments", *Federal Reserve Bank of Minneapolis, Research Department* Staff Report 148.

Geweke, John, (1992a). "Bayesian Treatment of the Heteroscedastic Normal and Independent Student-t Linear Models", *Federal Reserve Bank of Minneapolis, Institute for Empirical Macroeconomics*, Working paper 495.

Geweke John, (1992b). "Priors for Macroeconomic Time Series and Their Application", *Federal Reserve Bank of Minneapolis, Research Department*, Discussion Paper 64.

Gilley, O.W., and R. Kelley Pace. 1996. "On the Harrison and Rubinfeld Data," *Journal of Environmental Economics and Management*, Vol. 31 pp. 403-405.

McMillen, Daniel P. (1992) "Probit with spatial autocorrelation", *Journal of Regional Science*, Volume 32, number 3, pp. 335-348.

Chapter 6

VAR and Error Correction Models

This chapter describes vector autoregressive (VAR) and error correction (EC) models which have been used to model regional labor markets and other types of time-series representing regional economic activity. The MATLAB functions described here provide a way to implement more appropriate spatial prior information in Bayesian vector autoregressive models than that found in RATS software.

Section 6.1 describes the basic VAR model and our function to estimate and print results for this method. Section 6.2 turns attention to EC models while Section 6.3 discusses Bayesian spatial variants on these models. Finally, we take up forecasting in Section 6.4 and Section 6.5 provides applied examples.

6.1 VAR models

A VAR model is shown in (6.1) that contains n variables. The ε_{it} denote independent disturbances, C_i represent constants and $y_{it}, i = 1, \ldots, n$ denote the n variables in the model at time t. Model parameters $A_{ij}(\ell)$ take the form, $\sum_{k=1}^{m} a_{ijk}\ell^k$, where ℓ is the lag operator defined by $\ell^k y_t = y_{t-k}$, and m is the lag length specified by the modeler.

$$\begin{bmatrix} y_{1t} \\ y_{2t} \\ \vdots \\ y_{nt} \end{bmatrix} = \begin{bmatrix} A_{11}(\ell) & \dots & A_{1n}(\ell) \\ \vdots & \ddots & \vdots \\ A_{n1}(\ell) & \dots & A_{nn}(\ell) \end{bmatrix} \begin{bmatrix} y_{1t} \\ y_{2t} \\ \vdots \\ y_{nt} \end{bmatrix} + \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_n \end{bmatrix} + \begin{bmatrix} \varepsilon_{1t} \\ \varepsilon_{2t} \\ \vdots \\ \varepsilon_{nt} \end{bmatrix} \tag{6.1}$$

The VAR model posits a set of relationships between past lagged values of all variables in the model and the current value of each variable in the model. For example, if the y_{it} represent employment in state i at time t, the VAR model structure allows employment variation in each state to be explained by past employment variation in the state itself, $y_{it-k}, k = 1, \dots, m$ as well as past employment variation in other states, $y_{jt-k}, k = 1, \dots, m,\ j \neq i$. This is attractive since regional or state differences in business cycle activity suggest lead/lag relationships in employment of the type set forth by the VAR model structure.

The model is estimated using ordinary least-squares, so we can draw on our **ols** routine from the *Econometrics Toolbox*. A function **var** produces estimates for the coefficients in the VAR model as well as related regression statistics and Granger-causality test statistics.

The documentation for the **var** function is:

```
 PURPOSE: performs vector autoregressive estimation
---------------------------------------------------
 USAGE:  result = var(y,nlag,x)
 where:    y    = an (nobs x neqs) matrix of y-vectors
           nlag = the lag length
           x    = optional matrix of variables (nobs x nx)
 NOTE:     constant vector automatically included
---------------------------------------------------
 RETURNS a structure
 results.meth = 'var'
 results.nobs = nobs, # of observations
 results.neqs = neqs, # of equations
 results.nlag = nlag, # of lags
 results.nvar = nlag*neqs+nx+1, # of variables per equation
 --- the following are referenced by equation # ---
 results(eq).beta  = bhat for equation eq
 results(eq).tstat = t-statistics
 results(eq).tprob = t-probabilities
 results(eq).resid = residuals
 results(eq).yhat  = predicted values
 results(eq).y     = actual values
 results(eq).sige  = e'e/(n-k)
 results(eq).rsqr  = r-squared
```

```
 results(eq).rbar  = r-squared adjusted
 results(eq).boxq  = Box Q-statistics
 results(eq).ftest = Granger F-tests
 results(eq).fprob = Granger marginal probabilities
-------------------------------------------------
 SEE ALSO: varf, prt_var, pgranger, pftests
-------------------------------------------------
```

This function utilizes a new aspect of MATLAB structure variables, arrays that can store information for each equation in the VAR model. Estimates of the $\hat{\beta}$ parameters for the first equation can be accessed from the results structure using: 'result(1).beta' as can other results that are equation-specific.

In most applications, the user would simply pass the results structure on to the **prt** function that will provide an organized printout of the regression results for each equation. Here is an example of a typical program to estimate a VAR model.

```
% ----- Example 6.1 Using the var() function
dates = cal(1982,1,12);
load test.dat;               % monthly mining employment in 8 states
y = growthr(test(:,1:2),12); % (use only two states) convert to growth-rates
yt = trimr(y,dates.freq,0);  % truncate to account for lags in growth-rates
dates = cal(1983,1,1);       % redefine calendar for truncation
vnames   = strvcat('illinos','indiana');
nlag = 2;
result = var(yt,nlag);       % estimate 2-lag VAR model
prt(result,vnames);          % printout results
```

It would produce the following printout of the estimation results:

```
***** Vector Autoregressive Model *****
Dependent Variable =          illinos
R-squared     =    0.9044
Rbar-squared  =    0.9019
sige          =    3.3767
Q-statistic   =    0.2335
Nobs, Nvars   =    159,     5
***************************************************************
Variable            Coefficient      t-statistic    t-probability
illinos  lag1          1.042540        13.103752         0.000000
illinos  lag2         -0.132170        -1.694320         0.092226
indiana  lag1          0.228763         3.790802         0.000215
indiana  lag2         -0.213669        -3.538905         0.000531
constant              -0.333739        -1.750984         0.081940

 ****** Granger Causality Tests *******
```

```
Variable          F-value      Probability
illinos        363.613553         0.000000
indiana          7.422536         0.000837

Dependent Variable =          indiana
R-squared     =    0.8236
Rbar-squared  =    0.8191
sige          =    6.5582
Q-statistic   =    0.0392
Nobs, Nvars   =    159,     5
***************************************************************
Variable             Coefficient      t-statistic    t-probability
illinos  lag1           0.258853         2.334597         0.020856
illinos  lag2          -0.195181        -1.795376         0.074555
indiana  lag1           0.882544        10.493894         0.000000
indiana  lag2          -0.029384        -0.349217         0.727403
constant               -0.129405        -0.487170         0.626830

 ****** Granger Causality Tests *******
Variable          F-value      Probability
illinos          2.988892         0.053272
indiana        170.063761         0.000000
```

There are two utility functions that help analyze VAR model Granger-causality output. The first is **pgranger**, which prints a matrix of the marginal probabilities associated with the Granger-causality tests in a convenient format for the purpose of inference. The documentation is:

```
 PURPOSE: prints VAR model Granger-causality results
  ------------------------------------------------
 USAGE: pgranger(results,varargin);
        where: results = a structure returned by var(), ecm()
               varargin = a variable input list containing
                vnames  = an optional variable name vector
                cutoff  = probability cutoff used when printing
  usage example 1: pgranger(result,0.05);
        example 2: pgranger(result,vnames);
        example 3: pgranger(result,vnames,0.01);
        example 4: pgranger(result,0.05,vnames);
-----------------------------------------------------
                 e.g. cutoff = 0.05 would only print
                      marginal probabilities < 0.05
-----------------------------------------------------
 NOTES: constant term is added automatically to vnames list
       you need only enter VAR variable names plus deterministic
-----------------------------------------------------
```

As example of using this function, consider our previous program to estimate the VAR model for monthly mining employment in eight states. Rather

than print out the detailed VAR estimation results, we might be interested in drawing inferences regarding Granger-causality from the marginal probabilities. The following program would produce a printout of just these probabilities. It utilizes an option to suppress printing of probabilities greater than 0.1, so that our inferences would be drawn on the basis of a 90% confidence level.

```
% ----- Example 6.2 Using the pgranger() function
dates = cal(1982,1,12);     % monthly data starts in 82,1
load test.dat;              % monthly mining employment in 8 states
y = growthr(test,12);       % convert to growth-rates
yt = trimr(y,dates.freq,0); % truncate
dates = cal(1983,1,1);      % redefine the calendar for truncation
vname  = strvcat('il','in','ky','mi','oh','pa','tn','wv');
nlag = 12;
res = var(yt,nlag);         % estimate 12-lag VAR model
cutoff = 0.1;               % examine Granger-causality at 90% level
pgranger(res,vname,cutoff); % print Granger probabilities
```

We use the 'NaN' symbol to replace marginal probabilities above the cutoff point (0.1 in the example) so that patterns of causality are easier to spot. The results from this program would look as follows:

```
 ****** Granger Causality Probabilities *******
Variable      il      in      ky      mi      oh      pa      tn      wv
il          0.00    0.01    0.01     NaN     NaN    0.04    0.09    0.02
in          0.02    0.00     NaN     NaN     NaN     NaN     NaN     NaN
ky           NaN     NaN    0.00     NaN    0.10     NaN    0.07     NaN
mi           NaN    0.01     NaN    0.00     NaN     NaN     NaN     NaN
oh           NaN    0.05    0.08     NaN    0.00    0.01     NaN    0.01
pa          0.05     NaN     NaN     NaN     NaN    0.00     NaN    0.06
tn          0.02    0.05     NaN     NaN     NaN    0.09    0.00     NaN
wv          0.02    0.05    0.06    0.01     NaN    0.00     NaN    0.03
```

The format of the output is such that the columns reflect the Granger-causal impact of the column-variable on the row-variable. That is, Indiana, Kentucky, Pennsylvania, Tennessee and West Virginia exert a significant Granger-causal impact on Illinois employment whereas Michigan and Ohio do not. Indiana exerts the most impact, affecting Illinois, Michigan, Ohio, Tennessee, and West Virginia.

The second utility is a function **pftest** that prints just the Granger-causality joint F-tests from the VAR model. Use of this function is similar to **pgranger**, we simply call the function with the results structure returned by the **var** function, e.g., **pftest(result,vnames)**, where the 'vnames' argument is an optional string-vector of variable names. This function would

produce the following output for each equation of a VAR model based on all eight states:

```
 ****** Granger Causality Tests *******
Equation  illinois              F-value    F-probability
illinois                       395.4534          0.0000
indiana                          3.3255          0.0386
kentucky                         0.4467          0.6406
michigan                         0.6740          0.5112
ohio                             2.9820          0.0536
pennsylvania                     6.6383          0.0017
tennessee                        0.9823          0.3768
west virginia                    3.0467          0.0504
```

For an example of using Granger causality tests in regional modeling see LeSage and Reed (1990).

Although the illustrations so far have not involved use of deterministic variables in the VAR model, the **var** function is capable of handling these variables. As an example, we could include a set of seasonal dummy variables in the VAR model using:

```
% ----- Example 6.3 VAR with deterministic variables
dates = cal(1982,1,12);         % monthly data starts in 82,1
load test.dat;
y = test;                       % use levels data
[nobs neqs] = size(test);
sdum = sdummy(nobs,dates.freq); % create seasonal dummies
sdum = trimc(sdum,1,0);         % omit 1 column because we have
                                % a constant included by var()
vnames  = strvcat('illinos','indiana','kentucky','michigan','ohio', ...
          'pennsylvania','tennessee','west virginia');
dnames = strvcat('dum1','dum2','dum3','dum4','dum5','dum6','dum7', ...
          'dum8','dum9','dum10','dum11');
vnames  = strvcat(vnames,dnames);
nlag = 12;
result = var(y,nlag,sdum);
prt(result,vnames);
```

A handy option on the **prt** function is the ability to print the VAR model estimation results to an output file. Because these results are quite large, they can be difficult to examine in the MATLAB command window.

In addition to the **prt** function, there is **plt** that produce graphs of the actual versus predicted and residuals for these models.

One final issue associated with specifying a VAR model is the lag length to employ. A commonly used approach to determining the lag length is to

perform statistical tests of models with longer lags versus shorter lag lengths. We view the longer lag models as an unrestricted model versus the restricted shorter lag version of the model, and construct a likelihood ratio statistic to test for the significance of imposing the restrictions. If the restrictions are associated with a statistically significant degradation in model fit, we conclude that the longer lag length model is more appropriate, rejecting the shorter lag model.

Specifically, the chi-squared distributed test statistic which has degrees of freedom equation to the number of restrictions imposed is:

$$LR = (T - c)(\log|\Sigma_r| - \log|\Sigma_u|) \tag{6.2}$$

where T is the number of observations, c is a degrees of freedom correction factor proposed by Sims (1980), and $|\Sigma_r|, |\Sigma_u|$ denote the determinant of the error covariance matrices from the restricted and unrestricted models respectively. The correction factor, c, recommended by Sims was the number of variables in each unrestricted equation of the VAR model.

A function **lrratio** implements a sequence of such tests beginning at a maximum lag (specified by the user) down to a minimum lag (also specified by the user). The function prints results to the MATLAB command window along with marginal probability levels. As an example, consider the following program to determine the 'statistically optimal' lag length to use for our VAR model involving the eight-state sample of monthly employment data for the mining industry.

```
% ----- Example 6.4 Using the lrratio() function
load test.dat;
y = test; % use all eight states
maxlag = 12;
minlag = 3;
% Turn on flag for Sim's correction factor
sims = 1;
disp('LR-ratio results with Sims correction');
lrratio(y,maxlag,minlag,sims);
```

The output from this program is:

```
LR-ratio results with Sims correction
nlag = 12 11, LR statistic =          75.6240, probability = 0.1517
nlag = 11 10, LR statistic =          89.9364, probability = 0.01798
nlag = 10  9, LR statistic =          91.7983, probability = 0.01294
nlag =  9  8, LR statistic =         108.8114, probability = 0.0004052
nlag =  8  7, LR statistic =         125.7240, probability = 6.573e-06
nlag =  7  6, LR statistic =         114.2624, probability = 0.0001146
```

```
nlag =  6  5, LR statistic =          81.3528, probability = 0.07059
nlag =  5  4, LR statistic =         118.5982, probability = 4.007e-05
nlag =  4  3, LR statistic =         127.1812, probability = 4.489e-06
```

There exists an option flag to use the degrees of freedom correction suggested by Sims, whereas the default behavior of **lrratio** is to set $c = 0$. Example 4.4 turns on the correction factor by setting a flag that we named 'sims' equal to 1. The results suggest that the lag length of 11 cannot be viewed as significantly degrading the fit of the model relative to a lag of 12. For the comparison of lags 11 and 10, we find that at the 0.05 level, we might reject lag 10 in favor of lag 11 as the optimal lag length. On the other hand, if we employ a 0.01 level of significance, we would conclude that the optimal lag length is 9, because the likelihood ratio tests reject lag 8 as significantly degrading the fit of the model at the 0.01 level of confidence.

Another function for use with VAR models is **irf** that estimates response functions and provides a graphical presentation of the results. LeSage and Reed (1989a, 1989b) provide examples of using impulse response functions to examining regional wages in an urban hierarchy.

6.2 Error correction models

We provide a cursory introduction to co-integration and error correction models and refer the reader to an excellent layman's introduction by Dickey, Jansen and Thornton (1991) as well as a more technical work by Johansen (1995). LeSage (1990) and Shoesmith (1995) cover co-integration and EC models in the context of forecasting.

Focusing on the practical case of I(1), (integrated of order 1) series, let y_t be a vector of n time-series that are I(1). An I(1) series requires one difference to transform it to a zero mean, purely non-deterministic stationary process. The vector y_t is said to be co-integrated if there exists an n x r matrix α such that:

$$z_t = \alpha' y_t \tag{6.3}$$

Engle and Granger (1987) provide a Representation Theorem stating that if two or more series in y_t are co-integrated, there exists an error correction representation taking the following form:

$$\Delta y_t = A(\ell)\Delta y_t + \gamma z_{t-1} + \varepsilon_t \tag{6.4}$$

where γ is a matrix of coefficients of dimension n x r of rank r, z_{t-1} is of dimension r x 1 based on $r \leq n-1$ equilibrium error relationships, $z_t =$

$\alpha' y_t$ from (6.3), and ε_t is a stationary multivariate disturbance. The error correction (EC) model in (6.4) is simply a VAR model in first-differences with r lagged error correction terms (z_{t-1}) included in each equation of the model. If we have deterministic components in y_t, we add these terms as well as the error correction variables to each equation of the model.

With the case of only two series y_t and x_t in the model, a two-step procedure proposed by Engle and Granger (1987) can be used to determine the co-integrating variable that we add to our VAR model in first-differences to make it an EC model. The first-step involves a regression: $y_t = \theta + \alpha x_t + z_t$ to determine estimates of α and z_t. The second step carries out tests on z_t to determine if it is stationary, I(0). If we find this to be the case, the condition $y_t = \theta + \alpha x_t$ is interpreted as the equilibrium relationship between the two series and the error correction model is estimated as:

$$\begin{aligned} \Delta y_t &= -\gamma_1 z_{t-1} + \text{lagged}(\Delta x_t, \Delta y_t) + c_1 + \varepsilon_{1t} \\ \Delta x_t &= -\gamma_2 z_{t-1} + \text{lagged}(\Delta x_t, \Delta y_t) + c_2 + \varepsilon_{2t} \end{aligned}$$

where: $z_{t-1} = y_{t-1} - \theta - \alpha x_{t-1}$, c_i are constant terms and ε_{it} denote disturbances in the model.

We provide a function **adf**, (augmented Dickey-Fuller) to test time-series for the I(1), I(0) property, and another routine **cadf** (co-integrating augmented Dickey-Fuller) to carry out the tests from step two above on z_t to determine if it is stationary, I(0). These routines as well as the function **johansen** that implements a multivariate extension of the two-step Engle and Granger procedure were designed to mimic a set of Gauss functions by Sam Quilaris named **coint**.

The **adf** function documentation is:

```
 PURPOSE: carry out DF tests on a time series vector
------------------------------------------------
 USAGE: results = adf(x,p,nlag)
 where:      x = a time-series vector
             p = order of time polynomial in the null-hypothesis
                 p = -1, no deterministic part
                 p =  0, for constant term
                 p =  1, for constant plus time-trend
                 p >  1, for higher order polynomial
         nlags = # of lagged changes of x included
------------------------------------------------
 RETURNS: a results structure
         results.meth  = 'adf'
         results.alpha = estimate of the autoregressive parameter
```

```
        results.adf   = ADF t-statistic
        results.crit  = (6 x 1) vector of critical values
                        [1% 5% 10% 90% 95% 99%] quintiles
        results.nlag = nlag
```

This would be used to test a time-series vector for I(1) or I(0) status. Allowance is made for polynomial time trends as well as constant terms in the function and a set of critical values are returned in a structure by the function. A function **prt_coint** (as well as **prt**) can be used to print output from **adf, cadf** and **johansen**, saving users the work of formatting and printing the result structure output.

The function **cadf** is used for the case of two variables, y_t, x_t, where we wish to test whether the condition $y_t = \alpha x_t$ can be interpreted as an equilibrium relationship between the two series. The function documentation is:

```
PURPOSE: compute augmented Dickey-Fuller statistic for residuals
         from a cointegrating regression, allowing for deterministic
         polynomial trends
-------------------------------------------------------------
USAGE: results = cadf(y,x,p,nlag)
where: y = dependent variable time-series vector
       x = explanatory variables matrix
            p = order of time polynomial in the null-hypothesis
                p = -1, no deterministic part
                p =  0, for constant term
                p =  1, for constant plus time-trend
                p >  1, for higher order polynomial
    nlag = # of lagged changes of the residuals to include in regression
-------------------------------------------------------------
RETURNS: results structure
         results.meth  = 'cadf'
         results.alpha = autoregressive parameter estimate
         results.adf   = ADF t-statistic
         results.crit  =  (6 x 1) vector of critical values
                         [1% 5% 10% 90% 95% 99%] quintiles
         results.nvar  = cols(x)
         results.nlag  = nlag
```

As an illustration of using these two functions, consider testing our two monthly time-series on mining employment in Illinois and Indiana for I(1) status and then carrying out a test of whether they exhibit an equilibrating relationship. The program would look as follows:

```
% ----- Example 6.5 Using the adf() and cadf() functions
dates = cal(1982,1,12);
```

```
load test.dat;
y = test(:,1:2); % use only two series
vnames = strvcat('illinois','indiana');
% test Illinois for I(1) status
nlags = 6;
  for i=1:nlags;
   res = adf(y(:,1),0,i);
   prt(res,vnames(1,:));
  end;
% test Indiana for I(1) status
nlags = 6;
  for i=1:nlags;
   res = adf(y(:,2),0,i);
   prt(res,vnames(2,:));
  end;
% test if Illinois and Indiana are co-integrated
  for i=1:nlags;
   res = cadf(y(:,1),y(:,2),0,i);
    prt(res,vnames);
  end;
```

The program sets a lag length of 6, and loops over lags of 1 to 6 to provide some feel for how the augmented Dickey-Fuller tests are affected by the number of lags used. We specify $p = 0$ because the employment time-series do not have zero mean, so we wish to include a constant term. The result structures returned by the **adf** and **cadf** functions are passed on to **prt** for printing. We present the output for only lag 6 to conserve on space, but all lags produced the same inferences. One point to note is that the **adf** and **cadf** functions return a set of 6 critical values for significance levels 1%,5%,10%,90%,95%,99% as indicated in the documentation for these functions. Only three are printed for purposes of clarity, but all are available in the results structure returned by the functions.

```
Augmented DF test for unit root variable:                 illinois
 ADF t-statistic       # of lags   AR(1) estimate
       -0.164599               6         0.998867
   1% Crit Value    5% Crit Value   10% Crit Value
          -3.464           -2.912           -2.588

Augmented DF test for unit root variable:                 indiana
 ADF t-statistic       # of lags   AR(1) estimate
       -0.978913               6         0.987766
   1% Crit Value    5% Crit Value   10% Crit Value
          -3.464           -2.912           -2.588

 Augmented DF test for co-integration variables:         illinois,indiana
```

```
CADF t-statistic        # of lags   AR(1) estimate
     -1.67691570                6        -0.062974
   1% Crit Value    5% Crit Value   10% Crit Value
          -4.025           -3.404           -3.089
```

We see from the **adf** function results that both Illinois and Indiana are I(1) variables. We reject the augmented Dickey-Fuller hypothesis of I(0) because our t-statistics for both Illinois and Indiana are less than (in absolute value terms) the critical value of -2.588 at the 90% level.

From the results of **cadf** we find that Illinois and Indiana mining employment are not co-integrated, again because the t-statistic of -1.67 does not exceed the 90% critical value of -3.08 (in absolute value terms). We would conclude that an EC model is not appropriate for these two time-series.

For most EC models, more than two variables are involved so the Engle and Granger two-step procedure needs to be generalized. Johansen (1988) provides this generalization which takes the form of a likelihood-ratio test. We implement this test in the function **johansen**. The Johansen procedure provides a test statistic for determining r, the number of co-integrating relationships between the n variables in y_t as well as a set of r co-integrating vectors that can be used to construct error correction variables for the EC model.

As a brief motivation for the work carried out by the **johansen** function, we start with a reparameterization of the EC model:

$$\Delta y_t = \Gamma_1 \Delta y_{t-1} + \ldots + \Gamma_{k-1} \Delta y_{t-k+1} - \Psi y_{t-k} + \varepsilon_t \tag{6.5}$$

where $\Psi = (I_n - A_1 - A_2 - \ldots - A_k)$. If the matrix Ψ contains all zeros, (has rank=0), there are no co-integrating relationships between the variables in y_t. If Ψ is of full-rank, then we have n long-run equilibrium relationships, so all variables in the model are co-integrated. For cases where the matrix Ψ has rank $r < n$, we have r co-integrating relationships. The Johansen procedure provides two tests for the number of linearly independent co-integrating relationships among the series in y_t, which we have labeled r in our discussion. Both tests are based on an eigenvalue-eigenvector decomposition of the matrix Ψ, constructed from canonical correlations between Δy_t and y_{t-k} with adjustments for intervening lags, and taking into account that the test is based on estimates that are stochastic. The test statistics are labeled the 'trace statistic' and the 'maximal eigenvalue statistic'.

Given the value of r, (the number of co-integrating relationships), we can use the eigenvectors provided by the **johansen** function along with the levels of y_t lagged one period to form a set of error correction variables for

our EC model. In practice, the function **ecm** does this for you, so you need not worry about the details.

The documentation for **johansen** is shown below. A few points to note, the published literature contains critical values for the trace statistic for VAR models with up to 12 variables in Johansen (1995), and for the maximal eigenvalue statistic, Johansen and Juselius (1988) present critical values for VAR models containing up to 5 variables. To extend the number of variables for which critical values are available, a procedure by MacKinnon (1996) was used to generate critical values for both the trace and maximal eigenvalue statistics for models with up to 12 variables. MacKinnon's method is an approximation, but it produces values close to those in Johansen (1995). The critical values for the trace statistic have been entered in a function **c_sjt** and those for the maximal eigenvalue statistic are in **c_sja**. The function **johansen** calls these two functions to obtain the necessary critical values. In cases where the VAR model has more than 12 variables, zeros are returned as critical values in the structure field 'result.cvt' for the trace statistic and the 'result.cvm' field for the maximal eigenvalue.

Another less serious limitation is that the critical values for these statistics are only available for trend transformations where $-1 \leq p \leq 1$. This should not present a problem in most applications where p will take on values of -1, 0 or 1.

```
PURPOSE: perform Johansen cointegration tests
-----------------------------------------------------
USAGE: result = johansen(x,p,k)
where:      x = input matrix of time-series in levels, (nobs x m)
            p = order of time polynomial in the null-hypothesis
                p = -1, no deterministic part
                p =  0, for constant term
                p =  1, for constant plus time-trend
                p >  1, for higher order polynomial
            k = number of lagged difference terms used when
                computing the estimator
-----------------------------------------------------
RETURNS: a results structure:
         result.eig  = eigenvalues  (m x 1)
         result.evec = eigenvectors (m x m), where first
                       r columns are normalized coint vectors
         result.lr1  = likelihood ratio trace statistic for r=0 to m-1
                       (m x 1) vector
         result.lr2  = maximum eigenvalue statistic for r=0 to m-1
                       (m x 1) vector
         result.cvt  = critical values for trace statistic
                       (m x 3) vector [90% 95% 99%]
```

```
         result.cvm  = critical values for max eigen value statistic
                       (m x 3) vector [90% 95% 99%]
         result.ind  = index of co-integrating variables ordered by
                       size of the eigenvalues from large to small
-------------------------------------------------------
NOTE: c_sja(), c_sjt() provide critical values generated using
      a method of MacKinnon (1994, 1996).
      critical values are available for n<=12 and -1 <= p <= 1,
      zeros are returned for other cases.
```

As an illustration of the **johansen** function, consider the eight-state sample of monthly mining employment. We would test for the number of co-integrating relationships using the following code:

```
% ----- Example 6.6 Using the johansen() function
vnames  = strvcat('illinos','indiana','kentucky','michigan','ohio',  ...
           'pennsylvania','tennessee','west virginia');
y = load('test.dat'); % use all eight states
nlag = 9;
pterm = 0;
result = johansen(y,pterm,nlag);
prt(result,vnames);
```

The **johansen** function is called with the y matrix of time-series variables for the eight states, a value of $p = 0$ indicating we have a constant term in the model, and 9 lags. (We want $p = 0$ because the constant term is necessary where the levels of employment in the states differ.) The lag of 9 was determined to be optimal using the **lrratio** function in the previous section.

The **johansen** function will return results for a sequence of tests against alternative numbers of co-integrating relationships ranging from $r \leq 0$ up to $r \leq m - 1$, where m is the number of variables in the matrix y.

The function **prt** provides a printout of the trace and maximal eigenvalue statistics as well as the critical values returned in the **johansen** results structure.

```
 Johansen MLE estimates
NULL:                  Trace Statistic  Crit 90%   Crit 95%   Crit 99%
r <= 0   illinos               307.689   153.634    159.529    171.090
r <= 1   indiana               205.384   120.367    125.618    135.982
r <= 2   kentucky              129.133    91.109     95.754    104.964
r <= 3   ohio                   83.310    65.820     69.819     77.820
r <= 4   pennsylvania           52.520    44.493     47.855     54.681
r <= 5   tennessee              30.200    27.067     29.796     35.463
r <= 6   west virginia          13.842    13.429     15.494     19.935
```

```
r <= 7   michigan                0.412     2.705      3.841      6.635

NULL:                  Eigen Statistic  Crit 90%   Crit 95%   Crit 99%
r <= 0   illinos               102.305    49.285     52.362     58.663
r <= 1   indiana                76.251    43.295     46.230     52.307
r <= 2   kentucky               45.823    37.279     40.076     45.866
r <= 3   ohio                   30.791    31.238     33.878     39.369
r <= 4   pennsylvania           22.319    25.124     27.586     32.717
r <= 5   tennessee              16.359    18.893     21.131     25.865
r <= 6   west virginia          13.430    12.297     14.264     18.520
r <= 7   michigan                0.412     2.705      3.841      6.635
```

The printout does not present the eigenvalues and eigenvectors, but they are available in the results structure returned by **johansen** as they are needed to form the co-integrating variables for the EC model. The focus of co-integration testing would be the trace and maximal eigenvalue statistics along with the critical values. For this example, we find: (using the 95% level of significance) the trace statistic rejects $r \leq 0$ because the statistic of 307.689 is greater than the critical value of 159.529; it also rejects $r \leq 1$, $r \leq 2$, $r \leq 3$, $r \leq 4$, and $r \leq 5$ because these trace statistics exceed the associated critical values; for $r \leq 6$ we cannot reject H0, so we conclude that $r = 6$. Note that using the 99% level, we would conclude $r = 4$ as the trace statistic of 52.520 associated with $r \leq 4$ does not exceed the 99% critical value of 54.681.

We find a different inference using the maximal eigenvalue statistic. This statistic allows us to reject $r \leq 0$ as well as $r \leq 1$ and $r \leq 2$ at the 95% level. We cannot reject $r \leq 3$, because the maximal eigenvalue statistic of 30.791 does not exceed the critical value of 33.878 associated with the 95% level. This would lead to the inference that $r = 3$, in contrast to $r = 6$ indicated by the trace statistic. Using similar reasoning at the 99% level, we would infer $r = 2$ from the maximal eigenvalue statistics.

After the **johansen** test determines the number of co-integrating relationships, we can use these results along with the eigenvectors returned by the **johansen** function, to form a set of error correction variables. These are constructed using y_{t-1}, (the levels of y lagged one period) multiplied by the r eigenvectors associated with the co-integrating relationships to form r co-integrating variables. This is carried out by the **ecm** function, documented below.

```
 PURPOSE: performs error correction model estimation
-------------------------------------------------
 USAGE: result = ecm(y,nlag,r)
 where:    y    = an (nobs x neqs) matrix of y-vectors in levels
```

```
           nlag = the lag length
           r    = # of cointegrating relations to use
                  (optional: this will be determined using
                  Johansen's trace test at 95%-level if left blank)
 NOTES: constant vector automatically included
         x-matrix of exogenous variables not allowed
         error correction variables are automatically
         constructed using output from Johansen's ML-estimator
-------------------------------------------------
 RETURNS a structure
 results.meth = 'ecm'
 results.nobs = nobs, # of observations
 results.neqs = neqs, # of equations
 results.nlag = nlag, # of lags
 results.nvar = nlag*neqs+nx+1, # of variables per equation
 results.coint= # of co-integrating relations (or r if input)
 results.index= index of co-integrating variables ranked by
                size of eigenvalues large to small
 --- the following are referenced by equation # ---
 results(eq).beta   = bhat for equation eq (includes ec-bhats)
 results(eq).tstat  = t-statistics
 results(eq).tprob  = t-probabilities
 results(eq).resid  = residuals
 results(eq).yhat   = predicted values (levels) (nlag+2:nobs,1)
 results(eq).dyhat  = predicted values (differenced) (nlag+2:nobs,1)
 results(eq).y      = actual y-level values (nobs x 1)
 results(eq).dy     = actual y-differenced values (nlag+2:nobs,1)
 results(eq).sige   = e'e/(n-k)
 results(eq).rsqr   = r-squared
 results(eq).rbar   = r-squared adjusted
 results(eq).ftest  = Granger F-tests
 results(eq).fprob  = Granger marginal probabilities
 -------------------------------------------------
```

The **ecm** function allows two options for implementing an EC model. One option is to specify the number of co-integrating relations to use, and the other is to let the **ecm** function determine this number using the **johansen** function and the trace statistics along with the critical values at the 95% level of significance. The motivation for using the trace statistic is that it seems better suited to the task of sequential hypotheses for our particular decision problem.

An identical approach can be taken to implement a Bayesian variant of the EC model based on the Minnesota prior as well as a more recent Bayesian variant based on a "random-walk averaging prior". Both of these are discussed in the next section.

The **prt** function will produce a printout of the results structure re-

turned by **ecm** showing the autoregressive plus error correction variable coefficients along with Granger-causality test results as well as the trace, maximal eigenvalue statistics, and critical values from the **johansen** procedure. As an example, we show a program to estimate an EC model based on our eight-state sample of monthly mining employment, where we have set the lag-length to 2 to conserve on the amount of printed output.

```
% ----- Example 6.7 Estimating error correction models
y = load('test.dat'); % monthly mining employment for
                      % il,in,ky,mi,oh,pa,tn,wv 1982,1 to 1996,5
vnames =  strvcat('il','in','ky','mi','oh','pa','tn','wv');
nlag = 2;  % number of lags in var-model
% estimate the model, letting ecm determine # of co-integrating vectors
result = ecm(y,nlag);
prt(result,vnames); % print results to the command window
```

The printed output is shown below for a single state indicating the presence of two co-integrating relationships involving the states of Illinois and Indiana. The estimates for the error correction variables are labeled as such in the printout. Granger causality tests are printed, and these would form the basis for valid causality inferences in the case where co-integrating relationships existed among the variables in the VAR model.

```
Dependent Variable =              wv
R-squared     =    0.1975
Rbar-squared  =    0.1018
sige          =  341.6896
Nobs, Nvars   =    170,    19
***************************************************************
Variable              Coefficient      t-statistic    t-probability
il  lag1                 0.141055         0.261353         0.794176
il  lag2                 0.234429         0.445400         0.656669
in  lag1                 1.630666         1.517740         0.131171
in  lag2                -1.647557        -1.455714         0.147548
ky  lag1                 0.378668         1.350430         0.178899
ky  lag2                 0.176312         0.631297         0.528801
mi  lag1                 0.053280         0.142198         0.887113
mi  lag2                 0.273078         0.725186         0.469460
oh  lag1                -0.810631        -1.449055         0.149396
oh  lag2                 0.464429         0.882730         0.378785
pa  lag1                -0.597630        -2.158357         0.032480
pa  lag2                -0.011435        -0.038014         0.969727
tn  lag1                -0.049296        -0.045237         0.963978
tn  lag2                 0.666889         0.618039         0.537480
wv  lag1                -0.004150        -0.033183         0.973572
wv  lag2                -0.112727        -0.921061         0.358488
```

```
ec term il              -2.158992        -1.522859         0.129886
ec term in              -2.311267        -1.630267         0.105129
constant                 8.312788         0.450423         0.653052
 ****** Granger Causality Tests *******
Variable          F-value      Probability
il               0.115699         0.890822
in               2.700028         0.070449
ky               0.725708         0.485662
mi               0.242540         0.784938
oh               1.436085         0.241087
pa               2.042959         0.133213
tn               0.584267         0.558769
wv               1.465858         0.234146
 Johansen MLE estimates
NULL:        Trace Statistic      Crit 90%      Crit 95%      Crit 99%
r <= 0   il          214.390       153.634       159.529       171.090
r <= 1   in          141.482       120.367       125.618       135.982
r <= 2   ky           90.363        91.109        95.754       104.964
r <= 3   oh           61.555        65.820        69.819        77.820
r <= 4   tn           37.103        44.493        47.855        54.681
r <= 5   wv           21.070        27.067        29.796        35.463
r <= 6   pa           10.605        13.429        15.494        19.935
r <= 7   mi            3.192         2.705         3.841         6.635
NULL:        Eigen Statistic      Crit 90%      Crit 95%      Crit 99%
r <= 0   il           72.908        49.285        52.362        58.663
r <= 1   in           51.118        43.295        46.230        52.307
r <= 2   ky           28.808        37.279        40.076        45.866
r <= 3   oh           24.452        31.238        33.878        39.369
r <= 4   tn           16.034        25.124        27.586        32.717
r <= 5   wv           10.465        18.893        21.131        25.865
r <= 6   pa            7.413        12.297        14.264        18.520
r <= 7   mi            3.192         2.705         3.841         6.635
```

The results indicate that given the two lag model, two co-integrating relationships were found leading to the inclusion of two error correction variables in the model. The co-integrating relationships are based on the trace statistics compared to the critical values at the 95% level. From the trace statistics in the printed output we see that, H0: $r \leq 2$ was rejected at the 95% level because the trace statistic of 90.363 is less than the associated critical value of 95.754. Keep in mind that the user has the option of specifying the number of co-integrating relations to be used in the **ecm** function as an optional argument. If you wish to work at the 90% level of significance, we would conclude from the **johansen** results that $r = 4$ co-integrating relationships exist. To estimate an **ecm** model based on $r = 4$ we need simply call the **ecm** function with:

```
% estimate the model, using 4 co-integrating vectors
```

```
result = ecm(y,nlag,4);
```

6.3 Bayesian variants

Despite the attractiveness of drawing on cross-sectional information from related regional economic time series, the VAR model has empirical limitations. For example, a model with eight variables and six lags produces 49 independent variables in each of the eight equations of the model for a total of 392 coefficients to estimate. Large samples of observations involving time series variables that cover many years are needed to estimate the VAR model, and these are not always available. In addition, the independent variables represent lagged values, e.g., $y_{1t-1}, y_{1t-2}, \ldots, y_{1t-6}$, which tend to produce high correlations that lead to degraded precision in the parameter estimates. To overcome these problems, Doan, Litterman and Sims (1984) proposed the use of Bayesian prior information. The Minnesota prior means and variances suggested take the following form:

$$\begin{aligned} \beta_i &\sim N(1, \sigma^2_{\beta_i}) \\ \beta_j &\sim N(0, \sigma^2_{\beta_j}) \end{aligned} \tag{6.6}$$

where β_i denotes the coefficients associated with the lagged dependent variable in each equation of the VAR and β_j represents any other coefficient. The prior means for lagged dependent variables are set to unity in belief that these are important explanatory variables. On the other hand, a prior mean of zero is assigned to all other coefficients in the equation, β_j in (6.6), indicating that these variables are viewed as less important in the model.

The prior variances, $\sigma^2_{\beta_i}$, specify uncertainty about the prior means $\bar{\beta}_i = 1$, and $\sigma^2_{\beta_j}$ indicates uncertainty regarding the means $\bar{\beta}_j = 0$. Because the VAR model contains a large number of parameters, Doan, Litterman and Sims (1984) suggested a formula to generate the standard deviations as a function of a small number of hyperparameters: θ, ϕ and a weighting matrix $w(i, j)$. This approach allows a practitioner to specify individual prior variances for a large number of coefficients in the model using only a few parameters that are labeled hyperparameters. The specification of the standard deviation of the prior imposed on variable j in equation i at lag k is:

$$\sigma_{ijk} = \theta w(i, j) k^{-\phi} \left(\frac{\hat{\sigma}_{uj}}{\hat{\sigma}_{ui}} \right) \tag{6.7}$$

where $\hat{\sigma}_{ui}$ is the estimated standard error from a univariate autoregression involving variable i, so that $(\hat{\sigma}_{uj}/\hat{\sigma}_{ui})$ is a scaling factor that adjusts for varying magnitudes of the variables across equations i and j. Doan, Litterman and Sims (1984) labeled the parameter θ as 'overall tightness', reflecting the standard deviation of the prior on the first lag of the dependent variable. The term $k^{-\phi}$ is a lag decay function with $0 \leq \phi \leq 1$ reflecting the decay rate, a shrinkage of the standard deviation with increasing lag length. This has the effect of imposing the prior means of zero more tightly as the lag length increases, based on the belief that more distant lags represent less important variables in the model. The function $w(i, j)$ specifies the tightness of the prior for variable j in equation i relative to the tightness of the own-lags of variable i in equation i.

The overall tightness and lag decay hyperparameters used in the standard Minnesota prior have values $\theta = 0.1$, $\phi = 1.0$. The weighting matrix used is:

$$\mathrm{W} = \begin{bmatrix} 1 & 0.5 & \dots & 0.5 \\ 0.5 & 1 & & 0.5 \\ \vdots & & \ddots & \vdots \\ 0.5 & 0.5 & \dots & 1 \end{bmatrix} \tag{6.8}$$

This weighting matrix imposes $\bar{\beta}_i = 1$ loosely, because the lagged dependent variable in each equation is felt to be an important variable. The weighting matrix also imposes the prior mean of zero for coefficients on other variables in each equation more tightly since the β_j coefficients are associated with variables considered less important in the model.

A function **bvar** will provide estimates for this model. The function documentation is:

```
 PURPOSE: Performs a Bayesian vector autoregression of order n
-------------------------------------------------
 USAGE:  result = bvar(y,nlag,tight,weight,decay,x)
 where:    y    = an (nobs x neqs) matrix of y-vectors
           nlag = the lag length
          tight = Litterman's tightness hyperparameter
         weight = Litterman's weight (matrix or scalar)
          decay = Litterman's lag decay = lag^(-decay)
           x    = an optional (nobs x nx) matrix of variables
 NOTE:  constant vector automatically included
-------------------------------------------------
 RETURNS: a structure:
 results.meth      = 'bvar'
```

```
results.nobs      = nobs, # of observations
results.neqs      = neqs, # of equations
results.nlag      = nlag, # of lags
results.nvar      = nlag*neqs+1+nx, # of variables per equation
results.tight     = overall tightness hyperparameter
results.weight    = weight scalar or matrix hyperparameter
results.decay     = lag decay hyperparameter
--- the following are referenced by equation # ---
results(eq).beta  = bhat for equation eq
results(eq).tstat = t-statistics
results(eq).tprob = t-probabilities
results(eq).resid = residuals
results(eq).yhat  = predicted values
results(eq).y     = actual values
results(eq).sige  = e'e/(n-k)
results(eq).rsqr  = r-squared
results(eq).rbar  = r-squared adjusted
-------------------------------------------------
SEE ALSO:  bvarf, var, ecm, rvar, plt_var, prt_var
-------------------------------------------------
```

The function **bvar** allows us to input a scalar weight value or a more general matrix. Scalar inputs will be used to form a symmetric prior, where the scalar is used on the off-diagonal elements of the matrix. A matrix will be used in the form submitted to the function.

As an example of using the **bvar** function, consider our case of monthly mining employment for eight states. A program to estimate a BVAR model based on the Minnesota prior is shown below:

```
% ----- Example 6.8 Estimating BVAR models
vnames =  strvcat('il','in','ky','mi','oh','pa','tn','wv');
y = load('test.dat'); % use all eight states
nlag = 2;
tight = 0.1;  % hyperparameter values
weight = 0.5;
decay = 1.0;
result = bvar(y,nlag,tight,weight,decay);
prt(result,vnames);
```

The printout shows the hyperparameter values associated with the prior. It does not provide Granger-causality test results as these are invalid given the Bayesian prior applied to the model. Results for a single equation of the mining employment example are shown below.

```
***** Bayesian Vector Autoregressive Model *****
*****    Minnesota type Prior         *****
```

```
PRIOR hyperparameters
tightness =     0.10
decay     =     1.00
Symmetric weights based on     0.50

Dependent Variable =               il
R-squared     =    0.9942
Rbar-squared  =    0.9936
sige          =   12.8634
Nobs, Nvars   =    171,    17
***************************************************************
Variable             Coefficient      t-statistic    t-probability
il  lag1               1.134855        11.535932         0.000000
il  lag2              -0.161258        -1.677089         0.095363
in  lag1               0.390429         1.880834         0.061705
in  lag2              -0.503872        -2.596937         0.010230
ky  lag1               0.049429         0.898347         0.370271
ky  lag2              -0.026436        -0.515639         0.606776
mi  lag1              -0.037327        -0.497504         0.619476
mi  lag2              -0.026391        -0.377058         0.706601
oh  lag1              -0.159669        -1.673863         0.095996
oh  lag2               0.191425         2.063498         0.040585
pa  lag1               0.179610         3.524719         0.000545
pa  lag2              -0.122678        -2.520538         0.012639
tn  lag1               0.156344         0.773333         0.440399
tn  lag2              -0.288358        -1.437796         0.152330
wv  lag1              -0.046808        -2.072769         0.039703
wv  lag2               0.014753         0.681126         0.496719
constant               9.454700         2.275103         0.024149
```

There exists a number of attempts to alter the fact that the Minnesota prior treats all variables in the VAR model except the lagged dependent variable in an identical fashion. Some of the modifications suggested have focused entirely on alternative specifications for the prior variance. Usually, this involves a different (non-symmetric) weight matrix W and a larger value of 0.2 for the overall tightness hyperparameter θ in place of the value $\theta = 0.1$ used in the Minnesota prior. The larger overall tightness hyperparameter setting allows for more influence from other variables in the model. For example, LeSage and Pan (1995) constructed a weight matrix based on first-order spatial contiguity to emphasize variables from neighboring states in a multi-state agricultural output forecasting model. LeSage and Magura (1991) employed interindustry input-output weights to place more emphasis on related industries in a multi-industry employment forecasting model.

These approaches can be implemented using the **bvar** function by constructing an appropriate weight matrix. For example, the first order conti-

guity structure for the eight states in our mining employment example can be converted to a set of prior weights by placing values of unity on the main diagonal of the weight matrix, and in positions that represent contiguous entities. An example is shown in (6.9), where row 1 of the weight matrix is associated with the time-series for the state of Illinois. We place a value of unity on the main diagonal to indicate that autoregressive values from Illinois are considered important variables. We also place values of one in columns 2 and 3, reflecting the fact that Indiana (variable 2) and Kentucky (variable 3) are states that have borders touching Illinois. For other states that are not neighbors to Illinois, we use a weight of 0.1 to downweight their influence in the BVAR model equation for Illinois. A similar scheme is used to specify weights for the other seven states based on neighbors and non-neighbors.

$$W = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 0.1 & 0.1 & 0.1 \\ 1.0 & 1.0 & 1.0 & 0.1 & 1.0 & 0.1 & 1.0 & 1.0 \\ 0.1 & 1.0 & 0.1 & 1.0 & 1.0 & 0.1 & 0.1 & 0.1 \\ 0.1 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 0.1 & 1.0 \\ 0.1 & 0.1 & 0.1 & 0.1 & 1.0 & 1.0 & 0.1 & 1.0 \\ 0.1 & 0.1 & 1.0 & 0.1 & 0.1 & 0.1 & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 & 0.1 & 1.0 & 1.0 & 0.1 & 1.0 \end{bmatrix} \tag{6.9}$$

The intuition behind this set of weights is that we really don't believe the prior means of zero placed on the coefficients for mining employment in neighboring states. Rather, we believe these variables should exert an important influence. To express our lack of faith in these prior means, we assign a large prior variance to the zero prior means for these states by increasing the weight values. This allows the coefficients for these time-series variables to be determined by placing more emphasis on the sample data and less emphasis on the prior.

This could of course be implemented using **bvar** with a weight matrix specified, e.g.,

```
% ----- Example 6.9 Using bvar() with general weights
vnames =  strvcat('il','in','ky','mi','oh','pa','tn','wv');
dates = cal(1982,1,12);
y = load('test.dat'); % use all eight states
nlag = 2;
tight = 0.1;
decay = 1.0;
```

```
w = [1.0  1.0  1.0  0.1  0.1  0.1  0.1  0.1
     1.0  1.0  1.0  1.0  1.0  0.1  0.1  0.1
     1.0  1.0  1.0  0.1  1.0  0.1  1.0  1.0
     0.1  1.0  0.1  1.0  1.0  0.1  0.1  0.1
     0.1  1.0  1.0  1.0  1.0  1.0  0.1  1.0
     0.1  0.1  0.1  0.1  1.0  1.0  0.1  1.0
     0.1  0.1  1.0  0.1  0.1  0.1  1.0  0.1
     0.1  0.1  1.0  0.1  1.0  1.0  0.1  1.0];

result = bvar(y,nlag,tight,w,decay);
prt(result,vnames);
```

Another more recent approach to altering the equal treatment character of the Minnesota prior is a "random-walk averaging prior" suggested by LeSage and Krivelyova (1997, 1998).

As noted above, previous attempts to alter the fact that the Minnesota prior treats all variables in the VAR model except the first lag of the dependent variable in an identical fashion have focused entirely on alternative specifications for the prior variance. The prior proposed by LeSage and Krivelyova (1998) involves both prior means and variances motivated by the distinction between important and unimportant variables in each equation of the VAR model. To motivate the prior means, consider the weighting matrix for a five variable VAR model shown in (6.10). The weight matrix contains values of unity in positions associated with important variables in each equation of the VAR model and values of zero for unimportant variables. For example, the important variables in the first equation of the VAR model are variables 2 and 3 whereas the important variables in the fifth equation are variables 4 and 5.

Note that if we do not believe that autoregressive influences reflected by lagged values of the dependent variable are important, we have a zero on the main diagonal of the weight matrix. In fact, the weighting matrix shown in (6.10) classifies autoregressive influences as important in only two of the five equations in the VAR system, equations three and five. As an example of a case where autoregressive influences are totally ignored LeSage and Krivelyova (1997) constructed a VAR system based on spatial contiguity that relies entirely on the influence of neighboring states and ignores the autoregressive influence associated with past values of the variables from the states themselves.

$$W = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{6.10}$$

The weight matrix shown in (6.10) is standardized to produce row-sums of unity resulting in the matrix labeled C shown in (6.11).

$$C = \begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0.33 & 0.33 & 0.33 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix} \tag{6.11}$$

Using the row-standardized matrix C, we consider the random-walk with drift that averages over the important variables in each equation i of the VAR model as shown in (6.12).

$$y_{it} = \alpha_i + \sum_{j=1}^{n} C_{ij} y_{jt-1} + u_{it} \tag{6.12}$$

Expanding expression (6.12) we see that multiplying $y_{jt-1}, j = 1, \ldots, 5$ containing 5 variables at time $t-1$ by the row-standardized weight matrix C shown in (6.11) produces a set of explanatory variables for each equation of the VAR system equal to the mean of observations from important variables in each equation at time $t-1$ as shown in (6.13).

$$\begin{bmatrix} y_{1t} \\ y_{2t} \\ y_{3t} \\ y_{4t} \\ y_{5t} \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \end{bmatrix} + \begin{bmatrix} 0.5y_{2t-1} + 0.5y_{3t-1} \\ 0.5y_{1t-1} + 0.5y_{3t-1} \\ 0.33y_{1t-1} + 0.33y_{2t-1} + 0.33y_{3t-1} \\ 0.5y_{3t-1} + 0.5y_{5t-1} \\ 0.5y_{4t-1} + 0.5y_{5t-1} \end{bmatrix} + \begin{bmatrix} u_{1t} \\ u_{2t} \\ u_{3t} \\ u_{4t} \\ u_{5t} \end{bmatrix} \tag{6.13}$$

This suggests a prior mean for the VAR model coefficients on variables associated with the first own-lag of important variables equal to $1/c_i$, where c_i is the number of important variables in each equation i of the model. In the example shown in (6.13), the prior means for the first own-lag of the important variables y_{2t-1} and y_{3t-1} in the y_{1t} equation of the VAR would equal 0.5. The prior means for unimportant variables, y_{1t-1}, y_{4t-1} and y_{5t-1} in this equation would be zero.

This prior is quite different from the Minnesota prior in that it may downweight the lagged dependent variable using a zero prior mean to discount the autoregressive influence of past values of this variable. In contrast, the Minnesota prior emphasizes a random-walk with drift model that relies on prior means centered on a model: $y_{it} = \alpha_i + y_{it-1} + u_{it}$, where the intercept term reflects drift in the random-walk model and is estimated using a diffuse prior. The random-walk averaging prior is centered on a random-walk model that averages over important variables in each equation of the model and allows for drift as well. As in the case of the Minnesota prior, the drift parameters α_i are estimated using a diffuse prior.

Consistent with the Minnesota prior, LeSage and Krivelyova use zero as a prior mean for coefficients on all lags other than first lags. Litterman (1986) motivates reliance on zero prior means for many of the parameters of the VAR model by appealing to ridge regression. Recall, ridge regression can be interpreted as a Bayesian model that specifies prior means of zero for all coefficients, and as we saw in Chapter 3 can be used to overcome collinearity problems in regression models.

One point to note about the random walk averaging approach to specifying prior means is that the time series for the variables in the model need to be scaled or transformed to have similar magnitudes. If this is not the case, it would make little sense to indicate that the value of a time series observation at time t was equal to the average of values from important related variables at time $t-1$. This should present no problem as time series data can always be expressed in percentage change form or annualized growth rates which meets our requirement that the time series have similar magnitudes.

The prior variances LeSage and Krivelyova specify for the parameters in the model differ according to whether the coefficients are associated with variables that are classified as important or unimportant as well as the lag length. Like the Minnesota prior, they impose lag decay to reflect a prior belief that time series observations from the more distant past exert a smaller influence on the current value of the time series we are modeling. Viewing variables in the model as important versus unimportant suggests that the prior variance (uncertainty) specification should reflect the following ideas:

1. Parameters associated with unimportant variables should be assigned a smaller prior variance, so the zero prior means are imposed more 'tightly' or with more certainty.

2. First own-lags of important variables are given a smaller prior variance,

so the prior means force averaging over the first own-lags of important variables.

3. Parameters associated with unimportant variables at lags greater than one will be given a prior variance that becomes smaller as the lag length increases to reflect our belief that influence decays with time.

4. Parameters associated with lags other than first own-lag of important variables will have a larger prior variance, so the prior means of zero are imposed 'loosely'. This is motivated by the fact that we don't really have a great deal of confidence in the zero prior mean specification for longer lags of important variables. We think they should exert some influence, making the prior mean of zero somewhat inappropriate. We still impose lag decay on longer lags of important variables by decreasing our prior variance with increasing lag length. This reflects the idea that influence decays over time for important as well as unimportant variables.

It should be noted that the prior relies on inappropriate zero prior means for the important variables at lags greater than one for two reasons. First, it is difficult to specify a reasonable alternative prior mean for these variables that would have universal applicability in a large number of VAR model applications. The difficulty of assigning meaningful prior means that have universal appeal is most likely the reason that past studies relied on the Minnesota prior means while simply altering the prior variances. A prior mean that averages over previous period values of the important variables has universal appeal and widespread applicability in VAR modeling. The second motivation for relying on inappropriate zero prior means for longer lags of the important variables is that overparameterization and collinearity problems that plague the VAR model are best overcome by relying on a parsimonious representation. Zero prior means for the majority of the large number of coefficients in the VAR model are consistent with this goal of parsimony and have been demonstrated to produce improved forecast accuracy in a wide variety of applications of the Minnesota prior.

A flexible form with which to state prior standard deviations for variable j in equation i at lag length k is shown in (6.14).

$$\begin{array}{lcllll} \pi(a_{ijk}) & = & N(1/c_i, \sigma_c), & j \in C, & k = 1, & i,j = 1,\ldots,n \\ \pi(a_{ijk}) & = & N(0, \tau\sigma_c/k), & j \in C, & k = 2,\ldots,m, & i,j = 1,\ldots,n \\ \pi(a_{ijk}) & = & N(0, \theta\sigma_c/k), & j\neg \in C, & k = 1,\ldots,m, & i,j = 1,\ldots,n \end{array} \tag{6.14}$$

where:

$$0 < \sigma_c < 1 \tag{6.15}$$

$$\tau > 1 \tag{6.16}$$

$$0 < \theta < 1 \tag{6.17}$$

For variables $j = 1, \ldots, m$ in equation i that are important in explaining variation in variable i, $(j \in C)$, the prior mean for lag length $k = 1$ is set to the average of the number of important variables in equation i and to zero for unimportant variables $(j\neg \in C)$. The prior standard deviation is set to σ_c for the first lag, and obeys the restriction set forth in (6.15), reflecting a tight imposition of the prior mean that forces averaging over important variables. To see this, consider that the prior means $1/c_i$ range between zero and unity so typical σ_c values might be in the range of 0.1 to 0.25. We use $\tau\sigma_c/k$ for lags greater than one which imposes a decrease in this variance as the lag length k increases. Equation (6.16) states the restriction necessary to ensure that the prior mean of zero is imposed on the parameters associated with lags greater than one for important variables loosely, relative to a tight imposition of the prior mean of $1/c_i$ on first own-lags of important variables. We use $\theta\sigma_c/k$ for lags on unimportant variables whose prior means are zero, imposing a decrease in the variance as the lag length increases. The restriction in (6.17) would impose the zero means for unimportant variables with more confidence than the zero prior means for important variables.

This mathematical formulation adequately captures all aspects of the intuitive motivation for the prior variance specification enumerated above. A quick way to see this is to examine a graphical depiction of the prior mean and standard deviation for an important versus unimportant variable. An artificial example was constructed for an important variable in Figure 6.1 and an unimportant variable in Figure 4.3. Figure 6.1 shows the prior mean along with five upper and lower limits derived from the prior standard deviations in (6.14). The five standard deviation limits shown in the figure reflect $\pm$ 2 standard deviation limits resulting from alternative settings for the prior hyperparameter τ ranging from 5 to 9 and a value of $\sigma_c = 0.25$. Larger values of τ generated the wider upper and lower limits.

The solid line in Figure 6.1 reflects a prior mean of 0.2 for lag 1 indicating five important variables, and a prior mean of zero for all other lags. The prior standard deviation at lag 1 is relatively tight producing a small band around the averaging prior mean for this lag. This imposes the 'averaging' prior belief with a fair amount of certainty. Beginning at lag 2, the prior standard

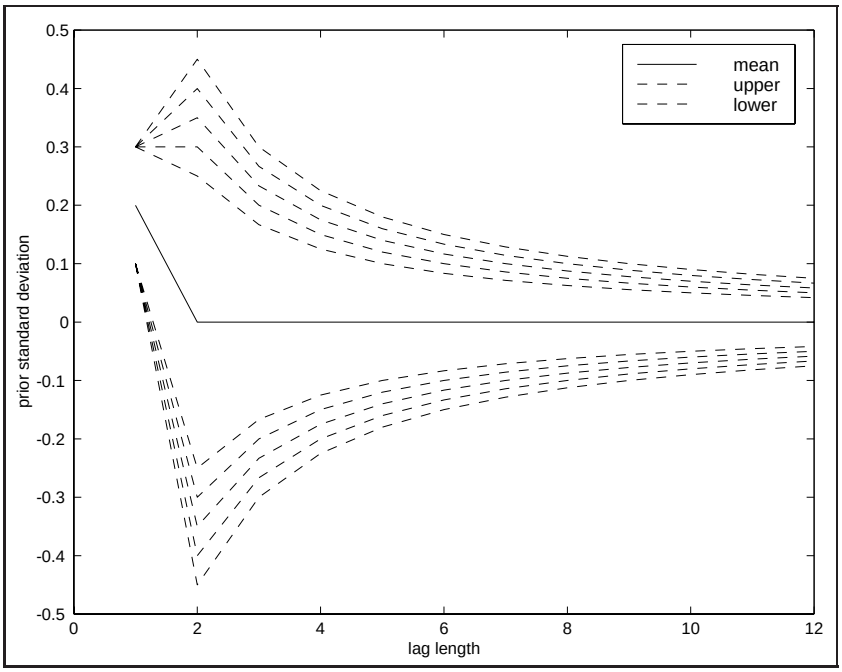

Figure 6.1: Prior means and precision for important variables

deviation is increased to reflect relative uncertainty about the new prior mean of zero for lags greater than unity. Recall, we believe that important variables at lags greater than unity will exert some influence, making the prior mean of zero not quite appropriate. Hence, we implement this prior mean with greater uncertainty.

Figure 6.2 shows an example of the prior means and standard deviations for an unimportant variable based on $\sigma_c = 0.25$ and five values of θ ranging from .35 to .75. Again, the larger θ values produce wider upper and lower limits. The prior for unimportant variables is motivated by the Minnesota prior that also uses zero prior means and rapid decay with increasing lag length.

A function **rvar** implements the random-walk averaging prior and a related function **recm** carries out estimation for an EC model based on this prior. The documentation for the **rvar** function is shown below, where we have eliminated information regarding the results structure variable returned by the function to save space.

```
PURPOSE: Estimates a Bayesian vector autoregressive model
```

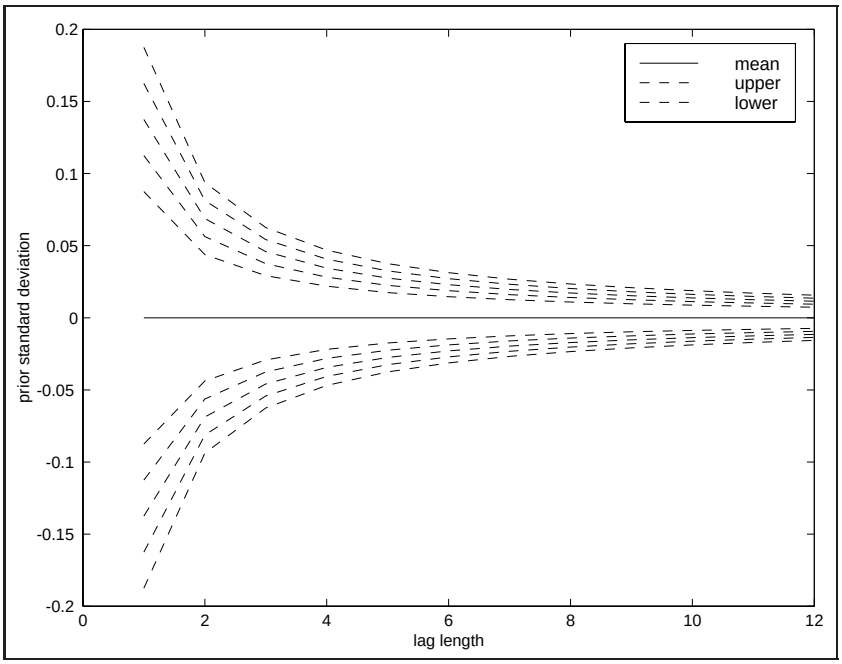

Figure 6.2: Prior means and precision for unimportant variables

```
          using the random-walk averaging prior
-------------------------------------------------
 USAGE:  result = rvar(y,nlag,w,freq,sig,tau,theta,x)
 where:    y    = an (nobs x neqs) matrix of y-vectors (in levels)
           nlag = the lag length
           w    = an (neqs x neqs) matrix containing prior means
                  (rows should sum to unity, see below)
           freq = 1 for annual, 4 for quarterly, 12 for monthly
           sig  = prior variance hyperparameter (see below)
           tau  = prior variance hyperparameter (see below)
          theta = prior variance hyperparameter (see below)
           x    = an (nobs x nx) matrix of deterministic variables
                  (in any form, they are not altered during estimation)
                  (constant term automatically included)
 priors important variables:   N(w(i,j),sig) for 1st own lag
                               N(  0 ,tau*sig/k) for lag k=2,...,nlag
 priors unimportant variables: N(w(i,j),theta*sig/k) for lag 1
                               N(  0 ,theta*sig/k)   for lag k=2,...,nlag
 e.g., if y1, y3, y4 are important variables in eq#1, y2 unimportant
  w(1,1) = 1/3, w(1,3) = 1/3, w(1,4) = 1/3, w(1,2) = 0
 typical values would be: sig = .1-.3, tau = 4-8, theta = .5-1
```

```
-------------------------------------------------
 NOTES: - estimation is carried out in annualized growth terms because
          the prior means rely on common (growth-rate) scaling of variables
          hence the need for a freq argument input.
        - constant term included automatically
-------------------------------------------------
```

Because this model is estimated in growth-rates form, an input argument for the data frequency is required. As an illustration of using both the **rvar** and **recm** functions, consider the following example based on the eight-state mining industry data. We specify a weight matrix for the prior means using first-order contiguity of the states.

```
% ----- Example 6.10 Estimating RECM models
y = load('test.dat'); % a test data set
vnames =  strvcat('il','in','ky','mi','oh','pa','tn','wv');
nlag = 6;  % number of lags in var-model
sig = 0.1;
tau = 6;
theta = 0.5;
freq = 12;   % monthly data
% this is an example of using 1st-order contiguity
% of the states as weights to produce prior means
W=[0      0.5    0.5    0     0     0    0     0
   0.25   0      0.25   0.25  0.25  0    0     0
   0.20   0.20   0      0     0.20  0    0.20  0.20
   0      0.50   0      0     0.50  0    0     0
   0      0.20   0.20   0.20  0     0.20 0.20  0.20
   0      0      0      0     0.50  0    0     0.50
   0      0      1      0     0     0    0     0
   0      0      0.33   0     0.33  0.33 0     0];
% estimate the rvar model
results = rvar(y,nlag,W,freq,sig,tau,theta);
% print results to a file
fid = fopen('rvar.out','wr');
prt(results,vnames,fid);
% estimate the recm model letting the function
% determine the # of co-integrating relationships
results = recm(y,nlag,W,freq,sig,tau,theta);
% print results to a file
fid = fopen('recm.out','wr');
prt(results,vnames,fid);
```

6.4 Forecasting the models

A set of forecasting functions are available that follow the format of the **var, bvar, rvar, ecm, becm, recm** functions named **varf, bvarf, rvarf,**

ecmf, becmf, recmf. These functions all produce forecasts of the time-series levels to simplify accuracy analysis and forecast comparison from the alternative models. They all take time-series levels arguments as inputs and carry out the necessary transformations. As an example, the **varf** documentation is:

```
PURPOSE: estimates a vector autoregression of order n
          and produces f-step-ahead forecasts
-------------------------------------------------------------
 USAGE:yfor = varf(y,nlag,nfor,begf,x,transf)
 where:    y    = an (nobs * neqs) matrix of y-vectors in levels
           nlag = the lag length
           nfor = the forecast horizon
           begf = the beginning date of the forecast
                  (defaults to length(x) + 1)
           x    = an optional vector or matrix of deterministic
                  variables (not affected by data transformation)
         transf = 0, no data transformation
                = 1, 1st differences used to estimate the model
                = freq, seasonal differences used to estimate
                = cal-structure, growth rates used to estimate
                  e.g., cal(1982,1,12) [see cal() function]
-------------------------------------------------------------
 NOTE: constant term included automatically
-------------------------------------------------------------
 RETURNS:
  yfor = an nfor x neqs matrix of level forecasts for each equation
-------------------------------------------------------------
```

Note that you input the variables y in levels form, indicate any of four data transformations that will be used when estimating the VAR model, and the function **varf** will carry out this transformation, produce estimates and forecasts that are converted back to levels form. This greatly simplifies the task of producing and comparing forecasts based on alternative data transformations.

For the case of a growth rates transformation a 'calendar' structure variable is required. These are produced with the **cal** function that is part of the *Econometrics Toolbox* discussed in Chapter 3 of the manual. In brief, using 'cstruct = cal(1982,1,12)' would set up the necessary calendar structure if the data being used were monthly time series that began in 1982.

Of course, if you desire a transformation other than the four provided, such as logs, you can transform the variables y prior to calling the function and specify 'transf=0'. In this case, the function does not provide levels forecasts, but rather forecasts of the logged-levels will be returned. Setting

'transf=0', produces estimates based on the data input and returns forecasts based on this data.

As an example of comparing alternative VAR model forecasts based on two of the four alternative transformations, consider the program in example 6.11.

```
% ----- Example 6.11 Forecasting VAR models
y = load('test.dat'); % a test data set containing
                      % monthly mining employment for
                      % il,in,ky,mi,oh,pa,tn,wv
dates = cal(1982,1,12); % data covers 1982,1 to 1996,5
nfor = 12; % number of forecast periods
nlag = 6;  % number of lags in var-model
begf = ical(1995,1,dates);  % beginning forecast period
endf = ical(1995,12,dates); % ending forecast period
% no data transformation example
fcast1 = varf(y,nlag,nfor,begf);
% seasonal differences data transformation example
freq = 12; % set frequency of the data to monthly
fcast2 = varf(y,nlag,nfor,begf,[],freq);
% compute percentage forecast errors
actual = y(begf:endf,:);
error1 = (actual-fcast1)./actual;
error2 = (actual-fcast2)./actual;
vnames =  strvcat('il','in','ky','mi','oh','pa','tn','wv');
fdates = cal(1995,1,12);
fprintf(1,'VAR model in levels percentage errors \n');
tsprint(error1*100,fdates,vnames,'%7.2f');
fprintf(1,'VAR - seasonally differenced data percentage errors \n');
tsprint(error2*100,fdates,vnames,'%7.2f');
```

In example 6.11 we rely on the calendar structure variables which are inputs to the **tsprint** utility function that prints time series with date labels as shown in the program output below. This utility function is part of the *Econometrics Toolbox* discussed in Chapter 3 of the manual.

```
VAR model in levels percentage errors
 Date       il      in      ky      mi      oh      pa      tn      wv
 Jan95   -3.95   -2.86   -1.15   -6.37   -5.33   -7.83   -0.19   -0.65
 Feb95   -5.63   -2.63   -3.57   -7.77   -7.56   -8.28   -0.99    0.38
 Mar95   -3.62   -1.75   -4.66   -5.49   -5.67   -6.69    2.26    2.30
 Apr95   -3.81   -4.23   -7.11   -4.27   -5.18   -5.41    2.14    0.17
 May95   -4.05   -5.60   -8.14   -0.92   -5.88   -3.93    2.77   -1.11
 Jun95   -4.10   -3.64   -8.87    0.10   -4.65   -4.15    2.90   -2.44
 Jul95   -4.76   -3.76  -10.06    1.99   -1.23   -5.06    3.44   -3.67
 Aug95   -8.69   -3.89   -9.86    4.85   -2.49   -5.41    3.63   -3.59
 Sep95   -8.73   -3.63  -12.24    0.70   -4.33   -6.28    3.38   -4.04
```

```
 Oct95  -11.11   -3.23  -12.10   -7.38   -4.74   -8.34    3.21   -5.57
 Nov95  -11.79   -4.30  -11.53   -8.93   -4.90   -7.27    3.60   -5.69
 Dec95  -12.10   -5.56  -11.12  -13.11   -5.57   -8.78    2.13   -9.38

VAR - seasonally differenced data percentage errors
 Date       il      in      ky      mi      oh      pa      tn      wv
 Jan95   -6.53   -0.52   -3.75    3.41   -1.49   -0.06    3.86    0.05
 Feb95   -4.35    1.75   -6.29    0.35   -3.53   -2.76    4.46    2.56
 Mar95   -1.12    2.61   -6.83    1.53   -2.72    2.24    2.96    3.97
 Apr95   -0.38   -2.36   -7.03   -4.30   -1.28    0.70    5.55    2.73
 May95    0.98   -5.05   -3.90   -4.65   -1.18    2.02    6.49   -0.43
 Jun95   -0.73   -2.55   -2.04   -0.30    2.30    0.81    3.96   -1.44
 Jul95   -1.41   -0.36   -1.69    0.79    4.83   -0.06    7.68   -4.24
 Aug95   -3.36    2.36   -1.78    7.99    4.86   -1.07    8.75   -3.38
 Sep95   -3.19    3.47   -3.26    6.91    2.31   -1.44    8.30   -3.02
 Oct95   -2.74    3.27   -2.88   -2.14    2.92   -0.73    9.00    0.08
 Nov95   -2.47    1.54   -2.63   -5.23    4.33    0.36    9.02    0.64
 Dec95   -1.35    0.48   -3.53   -7.89    4.38    1.33    7.03   -3.92
```

It is also possible to build models that produce forecasts that "feed-in" to another model as deterministic variables. For example, suppose we wished to use national employment in the primary metal industry (SIC 33) as a deterministic variable in our model for primary metal employment in the eight states. The following program shows how to accomplish this.

```
% ----- Example 6.12 Forecasting multiple related models
dates = cal(1982,1,12); % data starts in 1982,1
y=load('sic33.states'); % industry sic33 employment for 8 states
[nobs neqs] = size(y);
load sic33.national;    % industry sic33 national employment
ndates = cal(1947,1,12);% national data starts in 1947,1

begs = ical(1982,1,ndates); % find 1982,1 for national data
ends = ical(1996,5,ndates); % find 1996,5 for national data

x = sic33(begs:ends,1);  % pull out national employment in sic33
                         % for the time-period corresponding to
                         % our 8-state sample
begf = ical(1990,1,dates);  % begin forecasting date
endf = ical(1994,12,dates); % end forecasting date
nfor = 12;                  % forecast 12-months-ahead
nlag = 6;
xerror = zeros(nfor,1);
yerror = zeros(nfor,neqs);
cnt = 0; % counter for the # of forecasts we produce
for i=begf:endf % loop over dates producing forecasts
xactual = x(i:i+nfor-1,1);  % actual national employment
yactual = y(i:i+nfor-1,:);  % actual state employment
```

```
% first forecast national employment in sic33
xfor = varf(x,nlag,nfor,i); % an ar(6) model
xdet = [x(1:i-1,1)          % actual national data up to forecast period
        xfor      ];        % forecasted national data
% do state forecast using national data and forecast as input
yfor = varf(y,nlag,nfor,i,xdet);
% compute forecast percentage errors
xerror = xerror + abs((xactual-xfor)./xactual);
yerror = yerror + abs((yactual-yfor)./yactual);
cnt = cnt+1;
end; % end loop over forecasting experiment dates
% compute mean absolute percentage errors
xmape = xerror*100/cnt; ymape = yerror*100/cnt;
% printout results
in.cnames =  strvcat('national','il','in','ky','mi','oh','pa','tn','wv');
rnames = 'Horizon';
for i=1:12; rnames = strvcat(rnames,[num2str(i),'-step']); end;
in.rnames = rnames;
in.fmt = '%6.2f';
fprintf(1,'national and state MAPE percentage forecast errors \n');
fprintf(1,'based on %d 12-step-ahead forecasts \n',cnt);
mprint([xmape ymape],in);
```

Our model for national employment in SIC33 is simply an autoregressive model with 6 lags, but the same approach would work for a matrix X of deterministic variables used in place of the vector in the example. We can also provide for a number of deterministic variables coming from a variety of models that are input into other models, not unlike traditional structural econometric models. The program produced the following output.

```
national and state MAPE percentage forecast errors
based on 60 12-step-ahead forecasts
Horizon national     il     in     ky     mi     oh     pa     tn     wv
1-step      0.27   0.70   0.78   1.00   1.73   0.78   0.56   0.88   1.08
2-step      0.46   1.02   1.10   1.15   1.95   1.01   0.78   1.06   1.58
3-step      0.68   1.22   1.26   1.39   2.34   1.17   1.00   1.16   1.91
4-step      0.93   1.53   1.45   1.46   2.81   1.39   1.25   1.35   2.02
5-step      1.24   1.84   1.63   1.74   3.27   1.55   1.57   1.53   2.10
6-step      1.55   2.22   1.70   2.05   3.41   1.53   1.81   1.64   2.15
7-step      1.84   2.62   1.59   2.24   3.93   1.68   1.99   1.76   2.49
8-step      2.21   3.00   1.56   2.34   4.45   1.82   2.10   1.89   2.87
9-step      2.55   3.30   1.59   2.58   4.69   1.93   2.33   1.99   3.15
10-step     2.89   3.64   1.74   2.65   5.15   2.08   2.51   2.12   3.39
11-step     3.25   3.98   1.86   2.75   5.75   2.29   2.70   2.27   3.70
12-step     3.60   4.36   1.94   2.86   6.01   2.40   2.94   2.23   3.96
```

Consider that it would be quite easy to assess the contribution of using national employment as a deterministic variable in the model by running

another model that excludes this deterministic variable.

As a final example, consider an experiment where we wish to examine the impact of using different numbers of error correction variables on the forecast accuracy of the EC model. Shoesmith (1995) suggests that one should employ the number of error correction variables associated with the Johansen likelihood ratio statistics, but he provides only limited evidence regarding this contention.

The experiment uses time-series on national monthly employment from 12 manufacturing industries covering the period 1947,1 to 1996,12. Forecasts are carried out over the period from 1970,1 to 1995,12 using the number of error correction terms suggested by the Johansen likelihood ratio trace statistics, as well as models based on +/-1 and +/-2 error correction terms relative to the value suggested by the trace statistic.

We then compare the relative forecast accuracy of these models by examining the ratio of the MAPE forecast error from the models with +/-1 and +/-2 terms to the errors from the model based on r relationships suggested by the trace statistic.

Here is the program code:

```
% ----- Example 6.13 comparison of forecast accuracy as a function of
%                    the # of co-integrating vectors used
load level.mat;             % 20 industries national employment
y = level(:,1:12);          % use only 12 industries
[nobs neqs] = size(y);      dates = cal(1947,1,12);
begf = ical(1970,1,dates);  % beginning forecast date
endf = ical(1995,12,dates); % ending forecast date
nfor = 12;                  % forecast horizon
nlag = 10; cnt = 1;         % nlag based on lrratio() results
for i=begf:endf;
jres = johansen(y,0,nlag);  trstat = jres.lr1; tsignf = jres.cvt;
 r = 0;
 for j=1:neqs; % find r indicated by trace statistic
   if trstat(j,1) > tsignf(j,2), r = j; end;
 end;
% set up r-1,r-2 and r+1,r+2 forecasts in addition to forecasts based on r
if (r >= 3 & r <=10)
 frm2 = ecmf(y,nlag,nfor,i,r-2); frm1 = ecmf(y,nlag,nfor,i,r-1);
 fr   = ecmf(y,nlag,nfor,i,r);   frp1 = ecmf(y,nlag,nfor,i,r+1);
 frp2 = ecmf(y,nlag,nfor,i,r+2); act  = y(i:i+nfor-1,1:12);
 % compute forecast MAPE
 err(cnt).rm2 = abs((act-frm2)./act); err(cnt).rm1 = abs((act-frm1)./act);
 err(cnt).r   = abs((act-fr)./act);   err(cnt).rp1 = abs((act-frp1)./act);
 err(cnt).rp2 = abs((act-frp2)./act); cnt = cnt+1;
else
 fprintf(1,'time %d had %d co-integrating relations \n',i,r);
```

```
end; % end if-else;  end; % end of loop over time
rm2 = zeros(12,12); rm1 = rm2; rm0 = rm2; rp1 = rm2; rp2 = rm2;
for i=1:cnt-1;
rm2 = rm2 + err(i).rm2; rm1 = rm1 + err(i).rm1;
rm0 = rm0 + err(i).r;   rp1 = rp1 + err(i).rp1;
rp2 = rp2 + err(i).rp2;
end;
rm2 = rm2/(cnt-1);      rm1 = rm1/(cnt-1);
rm0 = rm0/(cnt-1);      rp1 = rp1/(cnt-1);
rp2 = rp2/(cnt-1);
rnames = 'Horizon'; cnames = [];
for i=1:12;
rnames = strvcat(rnames,[num2str(i),'-step']);
cnames = strvcat(cnames,['IND',num2str(i)]);
end;
in.rnames = rnames; in.cnames = cnames; in.fmt = '%6.2f';
fprintf(1,'forecast errors relative to error by ecm(r) model \n');
fprintf(1,'r-2 relative to r \n');
mprint(rm2./rm0,in);
fprintf(1,'r-1 relative to r \n');
mprint(rm2./rm0,in);
fprintf(1,'r+1 relative to r \n');
mprint(rp1./rm0,in);
fprintf(1,'r+2 relative to r \n');
mprint(rp2./rm0,in);
```

The program code stores the individual MAPE forecast errors in a structure variable using: **err(cnt).rm2 = abs((actual-frm2)./actual);**, which will have fields for the errors from all five models. These fields are matrices of dimension 12 x 12, containing MAPE errors for each of the 12-step-ahead forecasts for time **cnt** and for each of the 12 industries. We are not really interested in these individual results, but present this as an illustration. As part of the illustration, we show how to access the individual results to compute the average MAPE errors for each horizon and industry. If you wished to access industry number 2's forecast errors based on the model using r co-integrating relations, for the first experimental forecast period you would use: **err(1).rm(:,2)**. The results from our experiment are shown below. These results represent an average over a total of 312 twelve-step-ahead forecasts. Our simple MATLAB program produced a total of 224,640 forecasts, based on 312 twelve-step-ahead forecasts, for 12 industries, times 5 models!

Our experiment indicates that using more than the r co-integrating relationships determined by the Johansen likelihood trace statistic degrades the forecast accuracy. This is clear from the large number of forecast error ratios greater than unity for the two models based on $r + 1$ and $r + 2$ ver-

sus those from the model based on r. On the other hand, using a smaller number of co-integrating relationships than indicated by the Johansen trace statistic seems to improve forecast accuracy. In a large number of industries at many of the twelve forecast horizons, we see comparison ratios less than unity. Further, the forecast errors associated with $r-2$ are superior to those from $r-1$, producing smaller comparison ratios in 9 of the 12 industries.

forecast errors relative to error by ecm(r) model

r-2 relative to r

Horizon	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12
1-step	1.01	0.99	1.00	1.01	1.00	1.00	1.01	0.99	1.00	0.98	0.97	0.99
2-step	0.92	1.01	0.99	0.96	1.03	1.00	1.02	0.99	1.01	1.03	0.99	0.94
3-step	0.89	1.04	1.00	0.94	1.03	1.02	1.01	0.98	0.99	1.03	1.00	0.93
4-step	0.85	1.03	0.99	0.94	1.05	1.03	1.02	1.00	0.97	1.01	1.00	0.91
5-step	0.82	1.03	0.98	0.94	1.03	1.03	1.04	1.00	0.97	0.98	1.02	0.92
6-step	0.81	1.05	0.97	0.94	1.01	1.04	1.04	0.99	0.97	0.96	1.03	0.92
7-step	0.79	1.07	0.96	0.93	0.99	1.03	1.05	0.98	0.97	0.94	1.03	0.92
8-step	0.78	1.04	0.95	0.93	0.98	1.02	1.04	0.96	0.96	0.93	1.03	0.93
9-step	0.76	1.03	0.93	0.92	0.97	1.01	1.02	0.95	0.95	0.91	1.01	0.94
10-step	0.76	1.01	0.92	0.91	0.96	0.99	1.01	0.94	0.94	0.90	0.99	0.94
11-step	0.75	1.00	0.91	0.91	0.95	0.98	1.01	0.95	0.94	0.90	0.99	0.95
12-step	0.74	0.99	0.90	0.91	0.94	0.98	0.99	0.94	0.93	0.89	0.98	0.95

r-1 relative to r

Horizon	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12
1-step	1.01	0.99	1.00	1.01	1.00	1.00	1.01	0.99	1.00	0.98	0.97	0.99
2-step	0.92	1.01	0.99	0.96	1.03	1.00	1.02	0.99	1.01	1.03	0.99	0.94
3-step	0.89	1.04	1.00	0.94	1.03	1.02	1.01	0.98	0.99	1.03	1.00	0.93
4-step	0.85	1.03	0.99	0.94	1.05	1.03	1.02	1.00	0.97	1.01	1.00	0.91
5-step	0.82	1.03	0.98	0.94	1.03	1.03	1.04	1.00	0.97	0.98	1.02	0.92
6-step	0.81	1.05	0.97	0.94	1.01	1.04	1.04	0.99	0.97	0.96	1.03	0.92
7-step	0.79	1.07	0.96	0.93	0.99	1.03	1.05	0.98	0.97	0.94	1.03	0.92
8-step	0.78	1.04	0.95	0.93	0.98	1.02	1.04	0.96	0.96	0.93	1.03	0.93
9-step	0.76	1.03	0.93	0.92	0.97	1.01	1.02	0.95	0.95	0.91	1.01	0.94
10-step	0.76	1.01	0.92	0.91	0.96	0.99	1.01	0.94	0.94	0.90	0.99	0.94
11-step	0.75	1.00	0.91	0.91	0.95	0.98	1.01	0.95	0.94	0.90	0.99	0.95
12-step	0.74	0.99	0.90	0.91	0.94	0.98	0.99	0.94	0.93	0.89	0.98	0.95

r+1 relative to r

Horizon	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12
1-step	1.01	1.00	1.02	1.00	1.00	1.01	1.01	0.99	1.00	1.01	1.02	1.01
2-step	0.99	1.02	1.01	0.99	0.99	1.03	1.00	0.99	0.99	1.05	1.03	1.04
3-step	0.99	1.01	1.01	0.99	1.00	1.04	1.00	0.99	0.98	1.07	1.03	1.04
4-step	0.99	0.99	1.01	0.98	1.01	1.05	1.01	1.01	0.97	1.08	1.04	1.03
5-step	0.98	0.98	1.03	0.99	1.01	1.05	1.01	1.03	0.97	1.08	1.04	1.04
6-step	0.98	0.98	1.03	0.99	1.01	1.06	1.00	1.03	0.97	1.07	1.04	1.04
7-step	0.98	0.98	1.04	1.00	1.01	1.06	1.00	1.04	0.97	1.08	1.04	1.04
8-step	0.98	0.96	1.05	1.00	1.02	1.06	0.99	1.05	0.97	1.06	1.04	1.04
9-step	0.97	0.95	1.05	1.01	1.02	1.07	0.99	1.05	0.96	1.05	1.04	1.04
10-step	0.97	0.96	1.05	1.01	1.02	1.07	0.98	1.05	0.96	1.04	1.04	1.03
11-step	0.97	0.97	1.05	1.01	1.02	1.07	0.98	1.06	0.95	1.05	1.04	1.03
12-step	0.97	0.97	1.05	1.01	1.02	1.07	0.98	1.07	0.95	1.05	1.04	1.03

r+2 relative to r

Horizon	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12
1-step	1.00	1.01	1.02	1.01	0.99	1.01	1.01	0.99	0.99	1.05	1.03	1.01
2-step	1.00	1.05	1.00	0.97	1.00	1.03	1.01	1.00	0.99	1.11	1.03	1.06
3-step	1.00	1.02	1.01	0.96	1.01	1.06	1.02	1.02	0.98	1.13	1.04	1.06
4-step	1.00	0.99	1.01	0.97	1.02	1.07	1.02	1.04	0.97	1.14	1.05	1.05
5-step	1.01	0.97	1.03	0.98	1.04	1.08	1.02	1.07	0.97	1.15	1.05	1.04
6-step	1.01	0.95	1.04	0.99	1.04	1.10	1.02	1.08	0.97	1.15	1.06	1.04
7-step	1.01	0.96	1.06	1.00	1.05	1.10	1.01	1.09	0.96	1.15	1.06	1.03
8-step	1.00	0.93	1.08	0.99	1.05	1.10	1.00	1.10	0.95	1.15	1.07	1.02
9-step	1.01	0.92	1.09	0.99	1.06	1.11	0.99	1.11	0.95	1.14	1.08	1.02
10-step	1.01	0.92	1.09	0.99	1.05	1.11	0.98	1.11	0.94	1.13	1.08	1.01
11-step	1.01	0.93	1.09	0.99	1.05	1.12	0.98	1.13	0.94	1.14	1.09	1.00
12-step	1.00	0.93	1.09	0.99	1.05	1.12	0.97	1.13	0.94	1.15	1.09	0.99

6.5 An applied exercise

To illustrate using vector autoregressive modeling in a regional data example, we create a model that links national and state employment models. Monthly employment time series for ten national 2-digit industry categories is in a Bayesian vector autoregressive model to estimate and forecast employment one-year ahead for the period 1994,1 to 1994,12. We then use the historical national data as well as the 12 months of forecasted values as deterministic variables in a regional employment Bayesian vector autoregressive model that models monthly employment for eight contiguous states in the midwest.

Forecast errors are computed for the 12 month forecast and compared to the errors made by a Bayesian vector autoregressive model that is identical except for the inclusion of the national employment information as deterministic variables. This should provide a test of the forecasting value of national information in the regional model.

Example 6.14 shows the program to carry out the forecasting experiment. The program estimates and forecasts the ten industry national model using no data transformations. That is, the data are used in levels form. For the regional model a first-difference transformation is used. This transformation will be applied to the vector autoregressive variables, but not the deterministic national variables in the model.

```
% ----- Example 6.14 Linking national and regional models
% load 10 sic national manufacturing employment
% time-series covering the period 1947,1 to 1996,12
% from the MATLAB data file level.mat
dates = cal(1947,1,12);
load level.mat;
% the data is in a matrix variable named 'level'
% which we used to create the file (see make_level.m)
% 20  Food and kindred products
% 21  Tobacco manufactures
% 22  Textile, fabrics, yarn and thread mills
% 23  Miscellaneous fabricated textile products
% 24  Lumber and wood products
% 25  Furniture and fixtures
% 26  Paper and allied products
% 27  Printing and publishing
% 28  Chemicals, plastics, and synthetic materials
% 29  Petroleum refining and related industries
% produce a 12-month-ahead national forecast using a bvar model
begf = ical(1994,1,dates);
begs = ical(1982,1,dates);
```

```
nlag = 12;  tight = 0.2; weight = 0.5; decay = 0.1;
nfor = bvarf(level,nlag,12,begf,tight,weight,decay);
national = [level(begs:begf-1,:)
            nfor];
% use the national forecast as a deterministic variable
% in a regional forecasting model
load states.mat; % see make_states.m used to create states.mat
% il,in,ky,mi,oh,pa,tn,wv total state employment
% for 1982,1 to 1996,5
sdates = cal(1982,1,12);
snames = strvcat('il','in','ky','mi','oh','pa','tn','wv');
begf = ical(1994,1,sdates); nlag = 6;
% compute 12-month-ahead state forecast using a bvar model
% in 1st differences with national variables as deterministic
% and national forecasts used for forecasting
sfor = bvarf(states,nlag,12,begf,tight,weight,decay,national,1);
fdates = cal(1994,1,12);
% print forecasts of statewide employment
tsprint(sfor,fdates,1,12,snames,'%8.1f');
% print actual statewide employment
tsprint(states(begf:begf+11,:),fdates,1,12,snames,'%8.1f');
% compute and print percentage forecast errors
ferrors = (states(begf:begf+11,:) - sfor)./states(begf:begf+11,:);
tsprint(ferrors*100,fdates,1,12,snames,'%8.2f');
% compare the above results to a model without national employment
sfor2 = bvarf(states,nlag,12,begf,tight,weight,decay,[],1);
% compute and print percentage forecast errors
ferrors2 = (states(begf:begf+11,:) - sfor2)./states(begf:begf+11,:);
tsprint(ferrors2*100,fdates,1,12,snames,'%8.2f');
```

We can easily compute forecast errors because the vector autoregressive forecasting functions always return forecasted series in levels form. This allows us to simply subtract the forecasted values from the actual employment levels and divide by the actual levels to find a percentage forecast error.

Another point to note about example 6.11 is the use of the function **ical** that returns the observation number associated with January, 1994 the time period we wish to begin forecasting.

The results from the program in example 6.11 are shown below, where we find that use of the national information led to a dramatic improvement in forecast accuracy.

```
forecast values
Date        il       in       ky       mi       oh       pa       tn       wv
Jan94  53615.2  26502.0  15457.8  40274.7  49284.1  51025.3  23508.2  6568.9
Feb94  53860.3  26654.0  15525.7  40440.7  49548.7  51270.1  23676.3  6623.8
Mar94  54153.0  26905.9  15685.2  40752.1  50019.1  51634.6  24005.3  6698.2
Apr94  54486.8  27225.6  15862.7  41068.7  50528.2  52052.0  24206.0  6755.5
```

May94	54934.8	27473.4	15971.8	41517.4	51060.5	52455.3	24407.6	6889.0
Jun94	55118.3	27443.4	15943.5	41464.7	51085.9	52501.8	24439.0	6833.5
Jul94	54915.6	27315.6	15791.5	41033.6	50676.1	52073.2	24293.8	6845.9
Aug94	54727.4	27325.2	15802.0	40901.1	50493.6	51855.2	24358.8	6795.8
Sep94	54935.3	27591.3	15929.6	41188.7	50818.6	52045.4	24568.3	6800.8
Oct94	55128.2	27673.3	15993.3	41439.3	51082.4	52335.2	24683.0	6866.5
Nov94	55331.6	27769.2	16067.1	41480.2	51262.3	52453.1	24794.5	6884.3
Dec94	55508.7	27849.3	16097.4	41508.0	51411.4	52501.9	24873.4	6908.3

acutal values

Date	il	in	ky	mi	oh	pa	tn	wv
Jan94	52732.0	26254.0	15257.0	40075.0	49024.0	50296.0	23250.0	6414.0
Feb94	52999.0	26408.0	15421.0	40247.0	49338.0	50541.0	23469.0	6458.0
Mar94	53669.0	26724.0	15676.0	40607.0	49876.0	51058.0	23798.0	6554.0
Apr94	54264.0	26883.0	15883.0	40918.0	50221.0	51703.0	24009.0	6674.0
May94	54811.0	27198.0	16066.0	41497.0	50930.0	52140.0	24270.0	6894.0
Jun94	55247.0	27141.0	16100.0	41695.0	51229.0	52369.0	24317.0	6783.0
Jul94	54893.0	26938.0	15945.0	41236.0	50578.0	51914.0	24096.0	6807.0
Aug94	55002.0	27120.0	16047.0	41605.0	50742.0	51973.0	24315.0	6797.0
Sep94	55440.0	27717.0	16293.0	42248.0	51366.0	52447.0	24675.0	6842.0
Oct94	55290.0	27563.0	16221.0	42302.0	51700.0	52738.0	24622.0	6856.0
Nov94	55556.0	27737.0	16323.0	42547.0	51937.0	52938.0	25001.0	6983.0
Dec94	55646.0	27844.0	16426.0	42646.0	52180.0	52975.0	24940.0	6888.0

percentage foreast errors with national variables

Date	il	in	ky	mi	oh	pa	tn	wv
Jan94	-1.67	-0.94	-1.32	-0.50	-0.53	-1.45	-1.11	-2.41
Feb94	-1.63	-0.93	-0.68	-0.48	-0.43	-1.44	-0.88	-2.57
Mar94	-0.90	-0.68	-0.06	-0.36	-0.29	-1.13	-0.87	-2.20
Apr94	-0.41	-1.27	0.13	-0.37	-0.61	-0.67	-0.82	-1.22
May94	-0.23	-1.01	0.59	-0.05	-0.26	-0.60	-0.57	0.07
Jun94	0.23	-1.11	0.97	0.55	0.28	-0.25	-0.50	-0.74
Jul94	-0.04	-1.40	0.96	0.49	-0.19	-0.31	-0.82	-0.57
Aug94	0.50	-0.76	1.53	1.69	0.49	0.23	-0.18	0.02
Sep94	0.91	0.45	2.23	2.51	1.07	0.77	0.43	0.60
Oct94	0.29	-0.40	1.40	2.04	1.19	0.76	-0.25	-0.15
Nov94	0.40	-0.12	1.57	2.51	1.30	0.92	0.83	1.41
Dec94	0.25	-0.02	2.00	2.67	1.47	0.89	0.27	-0.29

percentage foreast errors without national variables

Date	il	in	ky	mi	oh	pa	tn	wv
Jan94	-1.47	-0.88	-1.68	-0.53	-0.42	-1.52	-1.12	-2.06
Feb94	-1.15	-0.69	-1.14	-0.38	-0.09	-1.47	-0.69	-1.58
Mar94	-0.11	0.04	-0.26	-0.09	0.45	-0.91	-0.25	-0.74
Apr94	0.79	0.05	0.25	0.40	0.73	-0.03	0.32	0.92
May94	1.37	0.72	0.92	0.99	1.51	0.43	0.97	2.82
Jun94	2.14	0.94	1.49	1.74	2.34	1.11	1.33	2.41
Jul94	1.83	0.69	1.34	1.52	1.81	0.94	1.01	2.69
Aug94	2.17	1.18	1.58	2.46	2.25	1.14	1.54	2.90
Sep94	2.46	2.28	2.13	3.06	2.68	1.50	2.14	3.45
Oct94	1.86	1.46	1.32	2.49	2.77	1.44	1.59	2.66
Nov94	2.00	1.87	1.58	2.88	2.92	1.56	2.77	4.23

```
Dec94     1.87     1.97     1.95     2.88     3.07     1.47     2.26     2.63
```

To continue with this example, we might wish to subject our experiment to a more rigorous test by carrying out a sequence of forecasts that reflect the experience we would gain from running the forecasting model over a period of years. Example 6.15 shows a program that produces forecasts in a loop extending over the period 1990,1 to 1994,12 for a total of five years or 60 12-step-ahead foreasts. The mean absolute percentage errors for the 12-step-ahead foreast horizon are calculated for two regional models, one with the national variables and another without.

```
% ----- example 6.15 Sequential forecasting of regional models
% load 10 sic national manufacturing employment
% time-series covering the period 1947,1 to 1996,12
% from the MATLAB data file national.mat
dates = cal(1947,1,12);
load national.mat;
% the data is in a matrix variable named 'data'
% which we used to create the file (see make_level.m)
% produce a 12-month-ahead national forecast using a bvar model
load states.mat; % see make_states.m used to create states.mat
% il,in,ky,mi,oh,pa,tn,wv total state employment
% for 1982,1 to 1996,5
sdates = cal(1982,1,12);
begin_date = ical(1982,1,dates);
snames = strvcat('il','in','ky','mi','oh','pa','tn','wv');
begf1 = ical(1990,1,dates);
begf2 = ical(1990,1,sdates);
endf = ical(1994,12,sdates);
nlag = 12; nlag2 = 6;
tight = 0.2; weight = 0.5; decay = 0.1;
ferrors1 = zeros(12,8); % storage for errors
ferrors2 = zeros(12,8);
cnt = 0;
for i=begf2:endf; % begin forecasting loop
% national forecast
nfor = bvarf(data,nlag,12,begf1,tight,weight,decay);
national = [data(begin_date:begf1-1,:)
            nfor];
begf1 = begf1+1;
% state forecast using national variables
sfor = bvarf(states,nlag,12,i,tight,weight,decay,national,1);
% compute errors
ferrors1 = ferrors1 + abs((states(i:i+11,:) - sfor)./states(i:i+11,:));
% state forecast without national variables
sfor = bvarf(states,nlag,12,i,tight,weight,decay,national,1);
% compute errors
```

```
ferrors2 = ferrors2 + abs((states(i:i+11,:) - sfor)./states(i:i+11,:));
cnt = cnt+1;
end; % end forecasting loop
in.cnames = snames;
in.rnames = strvcat('Horizon','step1','step2','step3','step4','step5',...
            'step6','step7','step8','step9','step10','step11','step12');
in.fmt = '%6.2f';
mprint((ferrors1/cnt)*100,in);
mprint((ferrors2/cnt)*100,in);
```

A complication in the program results from the fact that the national data sample begins in January, 1947 whereas the state data used in the regional model begins in January, 1982. This required the definitions 'begf1' and 'begf2' and 'begin_date' to pull out appropriate vectors of national variables for use in the regional model.

The mean absolute percentage forecast errors produced by the program are shown below.

```
   forecast MAPE errors with national variables
Horizon     il     in     ky     mi     oh     pa     tn     wv
step1     0.58   0.47   0.57   0.51   0.39   0.40   0.47   1.02
step2     0.75   0.64   0.82   0.70   0.56   0.54   0.64   1.34
step3     0.83   0.83   1.04   0.84   0.71   0.66   0.84   1.49
step4     0.95   0.99   1.20   1.05   0.94   0.81   0.99   1.88
step5     1.05   1.13   1.42   1.21   1.10   1.03   1.24   2.12
step6     1.15   1.35   1.70   1.35   1.26   1.21   1.44   2.34
step7     1.37   1.49   1.82   1.53   1.41   1.34   1.69   2.62
step8     1.46   1.63   1.89   1.66   1.48   1.43   1.82   2.75
step9     1.59   1.77   1.95   1.72   1.55   1.52   1.96   2.97
step10    1.74   1.90   2.11   1.94   1.70   1.55   2.13   3.02
step11    1.89   2.02   2.23   2.05   1.85   1.59   2.20   3.19
step12    2.06   2.07   2.36   2.11   1.97   1.69   2.35   3.48
   forecast MAPE errors with national variables
Horizon     il     in     ky     mi     oh     pa     tn     wv
step1     0.54   0.48   0.55   0.53   0.39   0.40   0.47   1.00
step2     0.68   0.66   0.73   0.76   0.56   0.55   0.59   1.22
step3     0.74   0.86   0.89   0.98   0.72   0.67   0.76   1.36
step4     0.88   1.02   1.04   1.19   0.84   0.70   0.93   1.61
step5     0.86   1.14   1.17   1.49   1.00   0.79   1.10   1.64
step6     0.91   1.20   1.31   1.62   1.09   0.83   1.19   1.63
step7     0.95   1.25   1.35   1.85   1.18   0.89   1.30   1.67
step8     1.04   1.27   1.29   2.03   1.25   0.95   1.44   1.76
step9     1.14   1.35   1.21   2.30   1.30   1.08   1.50   1.71
step10    1.15   1.43   1.29   2.44   1.36   1.21   1.62   1.59
step11    1.18   1.53   1.29   2.63   1.41   1.28   1.74   1.56
step12    1.27   1.59   1.33   2.82   1.48   1.36   1.95   1.65
```

From these results we see the expected pattern where longer horizon forecasts exhibit larger mean absolute percentage errors. The results from our single forecast experiment are not consistent with those from this experiment involving sequential forecast over a period of five years. The inclusion of national variables (and forecast) lead to less accurate forecasting performance for our regional model. The decrease in forecast accuracy is particularly noticable at the longer forecast horizons, which is probably indicative of poor national forecasts.

As an exercise, you might examine the accuracy of the national forecasts and try to improve on that model.

6.6 Chapter Summary

We found that a library of functions can be constructed to produce estimates and forecasts for a host of alternative vector autoregressive and error correction models. An advantage of MATLAB over a specialized program like RATS is that we have more control and flexibility to implement spatial priors. The spatial prior for the **rvar** model cannot be implemented in RATS software as the vector autoregressive function in that program does not allow you to specify prior means for variables other than the first own-lagged variables in the model.

Another advantage is the ability to write auxiliary functions that process the structures returned by our estimation functions and present output in a format that we find helpful. As an example of this, the function **pgranger** produced a formatted table of Granger-causality probabilities making it easy to draw inferences.

Finally, many of the problems encountered in carrying out forecast experiments involve transformation of the data for estimation purposes and reverse transformations needed to compute forecast errors on the basis of the levels of the time-series. Our functions can perform these transformations for the user, making the code necessary to carry out forecast experiments quite simple. In fact, one could write auxiliary functions that compute alternative forecast accuracy measures given matrices of forecast and actual values.

We also demonstrated how the use of structure array variables can facilitate storage of individual forecasts or forecast errors for a large number of time periods, horizons and variables. This would allow a detailed examination of the accuracy and characteristics associated with individual forecast errors for particular variables and time periods. As noted above, auxiliary

functions could be constructed to carry out this type of analysis.

6.7 References

Dickey, David A., Dennis W. Jansen and Daniel L. Thornton. 1991. "A primer on cointegration with an application to money and income," *Federal Reserve Bulletin, Federal Reserve Bank of St. Louis*, March/April, pp. 58-78.

Doan, Thomas, Robert. B. Litterman, and Christopher A. Sims. 1984. "Forecasting and conditional projections using realistic prior distributions," *Econometric Reviews*, Vol. 3, pp. 1-100.

Engle, Robert F. and Clive W.J. Granger. 1987. "Co-integration and error Correction: Representation, Estimation and Testing," *Econoemtrica*, Vol. 55, pp. 251-76.

Johansen, Soren. 1988. "Statistical Analysis of Co-integration vectors," *Journal of Economic Dynamics and Control*, Vol. 12, pp. 231-254.

Johansen, Soren. 1995. *Likelihood-based Inference in Cointegrated Vector autoregressive Models*, Oxford: Oxford University Press.

Johansen, Soren and Katarina Juselius. 1990. 'Maximum likelihood estimation and inference on cointegration - with applications to the demand for money", *Oxford Bulletin of Economics and Statistics*, Vol. 52, pp. 169-210.

LeSage, James P. 1990. "A Comparison of the Forecasting Ability of ECM and VAR Models," *Review of Economics and Statistics*, Vol. 72, pp. 664-671.

LeSage, James P. and Anna Krivelyova. 1997. "A Spatial Prior for Bayesian Vector Autoregressive Models," forthcoming in *Journal of Regional Science*, also available at: www.econ.utoledo.edu,

LeSage, James P. and Michael Magura. 1991. "Using interindustry input-output relations as a Bayesian prior in employment forecasting models", *International Journal of Forecasting*, Vol. 7, pp. 231-238.

LeSage, James P. and J. David Reed. 1989a. "Interregional Wage Transmission in an Urban Hierarchy: Tests Using Vector Autoregressive Models", with J. David Reed, *International Regional Science Review* , Volume 12, No. 3, pp. 305-318.

LeSage, James P. and J. David Reed. 1989b "The Dynamic Relationship Between Export, Local, and Total Area Employment", with J. David Reed, *Regional Science and Urban Economics*, 1989, Volume 19, pp. 615-636.

LeSage, James P. and J. David Reed. 1990. "Testing Criteria for Determining Leading Regions in Wage Transmission Models", with J. David Reed, *Journal of Regional Science*, 1990, Volume 30, no. 1, pp. 37-50.

LeSage, James P. and Zheng Pan. 1995. 'Using Spatial Contiguity as Bayesian Prior Information in Regional Forecasting Models", *International Regional Science Review*, Vol. 18, no. 1, pp. 33-53.

Litterman, Robert B. 1986. "Forecasting with Bayesian Vector Autoregressions — Five Years of Experience," *Journal of Business & Economic Statistics*, Vol. 4, pp. 25-38.

MacKinnon, J.G. 1994 "Approximate Asymptotic Distribution Functions for unit-root and cointegration tests," *Journal of Business & Economic Statistics*, Vol. 12, pp. 167-176.

MacKinnon, J.G. 1996 "Numerical distribution functions for unit-root and cointegration tests," *Journal of Applied Econometrics*, Vol. 11, pp. 601-618.

Shoesmith, Gary L. 1992. "Cointegration, error Correction and Improved Regional VAR Forecasting," *Journal of Forecasting*, Vol. 11, pp. 91-109.

Shoesmith, Gary L. 1995. "Multiple Cointegrating Vectors, error Correction, and Litterman's Model" *International Journal of Forecasting*, Vol. 11, pp. 557-567.

Sims, Christopher A. 1980. "Macroeconomics and Reality," *Econometrica* Vol. 48, pp. 1-48.

Appendix: Toolbox functions

The *Econometric Toolbox* is organized in a set of directories, each containing a different library of functions. When your Internet browser unpacks the compressed file containing the *Econometric Toolbox* the files will be placed in the appropriate directories.

To install the toolbox:

1. create a single subdirectory in the MATLAB toolbox directory:

   ```
   C:\matlab\toolbox\econ
   ```

   Where we have used the name **econ** for the directory.

2. Copy the system of directories to this subdirectory.

3. Use the graphical path tool in MATLAB to add these directories to your path. (On a unix or linux system, you may need to edit your environment variables that set the MATLAB path.) the graphical path tool in MATLAB to add these directories to your path. (On a unix or linux system, you may need to edit your environment variables that set the MATLAB path.)

A listing of the contents file from each subdirectory is presented on the following pages.

The *regression function library* is in a subdirectory **regress**.

```
regression function library

------- regression program functions -----------

ar_g        - Gibbs sampling Bayesian autoregressive model
bma_g       - Gibbs sampling Bayesian model averaging
boxcox      - Box-Cox regression with 1 parameter
boxcox2     - Box-Cox regression with 2 parameters
hmarkov_em  - Hamilton's Markov switching regression
hwhite      - Halbert White's heteroscedastic consistent estimates
lad         - least-absolute deviations regression
lm_test     - LM-test for two regression models
logit       - logit regression
mlogit      - multinomial logit regression
nwest       - Newey-West hetero/serial consistent estimates
ols         - ordinary least-squares
ols_g       - Gibbs sampling Bayesian linear model
olsar1      - Maximum Likelihood for AR(1) errors ols model
olsc        - Cochrane-Orcutt AR(1) errors ols model
olst        - regression with t-distributed errors
probit      - probit regression
probit_g    - Gibbs sampling Bayesian probit model
ridge       - ridge regression
rtrace      - ridge estimates vs parameters (plot)
robust      - iteratively reweighted least-squares
sur         - seemingly unrelated regressions
switch_em   - switching regime regression using EM-algorithm
theil       - Theil-Goldberger mixed estimation
thsls       - three-stage least-squares
tobit       - tobit regression
tobit_g     - Gibbs sampling Bayesian tobit model
tsls        - two-stage least-squares
waldf       - Wald F-test

-------- demonstration programs -----------------

ar_gd       - demonstration of Gibbs sampling ar_g
bma_gd      - demonstrates Bayesian model averaging
box_cox_d   - demonstrates Box-Cox 1-parameter model
boxcox2_d   - demonstrates Box-Cox 2-parmaeter model
demo_all    - demos most regression functions
hmarkov_emd - demos Hamilton's Markov switching regression
hwhite_d    - H. White's hetero consistent estimates demo
lad_d       - demos lad regression
lm_test_d   - demos lm_test
logit_d     - demonstrates logit regression
mlogit_d    - demonstrates multinomial logit
```

```
nwest_d     - demonstrates Newey-West estimates
ols_d       - demonstrates ols regression
ols_d2      - Monte Carlo demo using ols regression
ols_gd      - demo of Gibbs sampling ols_g
olsar1_d    - Max Like AR(1) errors model demo
olsc_d      - Cochrane-Orcutt demo
olst_d      - olst demo
probit_d    - probit regression demo
probit_gd   - demo of Gibbs sampling Bayesian probit model
ridge_d     - ridge regression demo
robust_d    - demonstrates robust regression
sur_d       - demonstrates sur using Grunfeld's data
switch_emd  - demonstrates switching regression
theil_d     - demonstrates theil-goldberger estimation
thsls_d     - three-stage least-squares demo
tobit_d     - tobit regression demo
tobit_gd    - demo of Gibbs sampling Bayesian tobit model
tsls_d      - two-stage least-squares demo
waldf_d     - demo of using wald F-test function

-------- Support functions ------------------------

ar1_like    - used by olsar1   (likelihood)
bmapost     - used by bma_g
box_lik     - used by box_cox  (likelihood)
box_lik2    - used by box_cox2 (likelihood)
boxc_trans  - used by box_cox, box_cox2
chis_prb    - computes chi-squared probabilities
dmult       - used by mlogit
fdis_prb    - computes F-statistic probabilities
find_new    - used by bma_g
grun.dat    - Grunfeld's data used by sur_d
grun.doc    - documents Grunfeld's data set
lo_like     - used by logit    (likelihood)
maxlik      - used by tobit
mcov        - used by hwhite
mderivs     - used by mlogit
mlogit_lik  - used by mlogit
nmlt_rnd    - used by probit_g
nmrt_rnd    - used by probit_g, tobit_g
norm_cdf    - used by probit, pr_like
norm_pdf    - used by prt_reg, probit
olse        - ols returning only residuals (used by sur)
plt         - plots  everything
plt_eqs     - plots equation systems
plt_reg     - plots regressions
pr_like     - used by probit   (likelihood)
prt         - prints everything
prt_eqs     - prints equation systems
```

```
prt_gibbs   - prints Gibbs sampling models
prt_reg     - prints regressions
prt_swm     - prints switching regression results
sample      - used by bma_g
stdn_cdf    - used by norm_cdf
stdn_pdf    - used by norm_pdf
stepsize    - used by logit,probit to determine stepsize
tdis_prb    - computes t-statistic probabilities
to_like     - used by tobit    (likelihood)
```

The utility functions are in a subdirectory **util**.

```
utility function library

 -------- utility functions -----------------------------

accumulate  - accumulates column elements of a matrix
cal         - associates obs # with time-series calendar
ccorr1      - correlation scaling to normal column length
ccorr2      - correlation scaling to unit column length
fturns      - finds turning-points in a time-series
growthr     - converts time-series matrix to growth rates
ical        - associates time-series dates with obs #
indicator   - converts a matrix to indicator variables
invccorr    - inverse for ccorr1, ccorr2
lag         - generates a lagged variable vector or matrix
levels      - generates factor levels variable
lprint      - prints a matrix in LaTeX table-formatted form
lprintf     - enhanced lprint function
mlag        - generates a var-type matrix of lags
mode        - calculates the mode of a distribution
mprint      - prints a matrix
mprint3     - prints coefficient, t-statistics matrices
mth2qtr     - converts monthly to quarterly data
nclag       - generates a matrix of non-contiguous lags
plt         - wrapper function, plots all result structures
prt         - wrapper function, prints all result strucutres
sacf        - sample autocorrelation function estimates
sdiff       - seasonal differencing
sdummy      - generates seasonal dummy variables
shist       - plots spline smoothed histogram
spacf       - sample partial autocorrelation estimates
tally       - computes frequencies of distinct levels
tdiff       - time-series differencing
tsdates     - time-series dates function
tsprint     - print time-series matrix
unsort      - unsorts a sorted vector or matrix
vec         - turns a matrix into a stacked vector
vech        - matrix from lower triangular columns of a matrix
```

```
-------- demonstration programs -------------

cal_d.m     - demonstrates cal function
fturns_d    - demonstrates fturns and plt
ical_d.m    - demonstrates ical function
lprint_d.m  - demonstrates lprint function
lprintf_d.m - demonstrates lprintf function
mprint_d.m  - demonstrates mprint function
mprint3_d.m - demonstrates mprint3 function
sacf_d      - demonstrates sacf
spacf_d     - demonstrates spacf
tsdate_d.m  - demonstrates tsdate function
tsprint_d.m - demonstrates tsprint function
util_d.m    - demonstrated some of the utility functions

-------- functions to mimic analogous Gauss functions -------------

cols        - returns the # of columns in a matrix or vector
cumprodc    - returns cumulative product of each column of a matrix
cumsumc     - returns cumulative sum of each column of a matrix
delif       - select matrix values for which a condition is false
indexcat    - extract indices equal to a scalar or an interval
invpd       - makes a matrix positive-definite, then inverts
matadd      - adds non-conforming matrices, row or col compatible.
matdiv      - divides non-conforming matrices, row or col compatible.
matmul      - multiplies non-conforming matrices, row or col compatible.
matsub      - divides non-conforming matrices, row or col compatible.
prodc       - returns product of each column of a matrix
rows        - returns the # of rows in a matrix or vector
selif       - select matrix values for which a condition is true
seqa        - a sequence of numbers with a beginning and increment
stdc        - std deviations of columns  returned as a column vector
sumc        - returns sum of each column
trimc       - trims columns of a matrix (or vector) like Gauss
trimr       - trims rows of a matrix (or vector) like Gauss
```

A set of graphing functions are in a subdirectory **graphs**.

```
graphing function library

 -------- graphing programs ---------------------------

pairs       - scatter plot (uses histo)
pltdens     - density plots
tsplot      - time-series graphs

-------- demonstration programs -----------------------
```

```
pairs_d     - demonstrates pairwise scatter
pltdens_d   - demonstrates pltdens
tsplot_d    - demonstrates tsplot

------- support functions -----------------------------

histo       - used by pairs
plt_turns   - plots turning points from fturns function
```

A library of routines in the subdirectory **diagn** contain the regression diagnostics functions.

```
regression diagnostics library

-------- diagnostic programs ---------------

bkw              - BKW collinearity diagnostics
bpagan           - Breusch-Pagan heteroscedasticity test
cusums           - Brown,Durbin,Evans cusum squares test
dfbeta           - BKW influential observation diagnostics
diagnose         - compute diagnostic statistics
rdiag            - graphical residuals diagnostics
recresid         - compute recursive residuals
studentize       - standarization transformation

------- demonstration programs -------------

bkw_d            - demonstrates bkw
bpagan_d         - demonstrates bpagan
cusums_d         - demonstrates cusums
dfbeta_d         - demonstrates dfbeta, plt_dfb, plt_dff
diagnose_d       - demonstrates diagnose
rdiag_d          - demonstrates rdiag
recresid_d       - demonstrates recresid

------- support functions ------------------

ols.m            - least-squares regression
plt              - plots everything
plt_cus          - plots cusums test results
plt_dfb          - plots dfbetas
plt_dff          - plots dffits
```

The vector autoregressive library is in a subdirectory **var_bvar**.

```
vector autoregressive function library
```

```
------- VAR/BVAR program functions -----------

 becm_g   - Gibbs sampling BECM estimates
 becmf    - Bayesian ECM model forecasts
 becmf_g  - Gibbs sampling BECM forecasts
 bvar     - BVAR model
 bvar_g   - Gibbs sampling BVAR estimates
 bvarf    - BVAR model forecasts
 bvarf_g  - Gibbs sampling BVAR forecasts
 ecm      - ECM (error correction) model estimates
 ecmf     - ECM model forecasts
 irf      - impulse response functions
 lratio   - likelihood ratio tests for lag length
 recm     - ecm version of rvar
 recm_g   - Gibbs sampling random-walk averaging estimates
 recmf    - random-walk averaging ECM forecasts
 recmf_g  - Gibbs sampling random-walk averaging forecasts
 rvar     - Bayesian random-walk averaging prior model
 rvar_g   - Gibbs sampling RVAR estimates
 rvarf    - Bayesian RVAR model forecasts
 rvarf_g  - Gibbs sampling RVAR forecasts
 var      - VAR model
 varf     - VAR model forecasts

 ------- demonstration programs  -----------

 becm_d    - BECM model demonstration
 becm_g    - Gibbs sampling BECM estimates demo
 becmf_d   - becmf demonstration
 becmf_gd  - Gibbs sampling BECM forecast demo
 bvar_d    - BVAR model demonstration
 bvar_gd   - Gibbs sampling BVAR demonstration
 bvarf_d   - bvarf demonstration
 bvarf_gd  - Gibbs sampling BVAR forecasts demo
 ecm_d     - ECM model demonstration
 ecmf_d    - ecmf demonstration
 irf_d     - impulse response function demo
 irf_d2    - another irf demo
 lrratio_d - demonstrates lrratio
 pftest_d  - demo of pftest function
 recm_d    - RECM model demonstration
 recm_gd   - Gibbs sampling RECM model demo
 recmf_d   - recmf demonstration
 recmf_gd  - Gibbs sampling RECM forecast demo
 rvar_d    - RVAR model demonstration
 rvar_gd   - Gibbs sampling rvar model demo
 rvarf_d   - rvarf demonstration
 rvarf_gd  - Gibbs sampling rvar forecast demo
 var_d     - VAR model demonstration
```

```
varf_d    - varf demonstration

------- support functions  ------------------

johansen  - used by ecm,ecmf,becm,becmf,recm,recmf
lag       - does ordinary lags
lrratio   - likelihood ratio lag length tests
mlag      - does var-type lags
nclag     - does contiguous lags (used by rvar,rvarf,recm,recmf)
ols       - used for VAR estimation
pftest    - prints Granger F-tests
pgranger  - prints Granger causality probabilities
prt       - prints results from all functions
prt_coint - used by prt_var for ecm,becm,recm
prt_var   - prints results of all var/bvar models
prt_varg  - prints results of all Gibbs var/bvar models
rvarb     - used for RVARF forecasts
scstd     - does univariate AR for BVAR
theil_g   - used for Gibbs sampling estimates and forecasts
theilbf   - used for BVAR forecasts
theilbv   - used for BVAR estimation
trimr     - used by VARF,BVARF, johansen (in /util/trimr.m)
vare      - used by lrratio (vare uses /regress/olse.m)
```

The co-integration library functions are in a subdirectory **coint**.

```
co-integration library

------ co-integration testing routines --------

adf        - carries out Augmented Dickey-Fuller unit root tests
cadf       - carries out ADF tests for co-integration
johansen   - carries out Johansen's co-integration tests

------ demonstration programs -----------------

adf_d      - demonstrates adf
cadf_d     - demonstrates cadf
johansen_d - demonstrates johansen

------ support functions ----------------------

c_sja      - returns critical values for SJ maximal eigenvalue test
c_sjt      - returns critical values for SJ trace test
cols       - (like Gauss cols)
detrend    - used by johansen to detrend data series
prt_coint  - prints results from adf,cadf,johansen
ptrend     - used by adf to create time polynomials
rows       - (like Gauss rows)
```

```
rztcrit    - returns critical values for cadf test
tdiff      - time-series differences
trimr      - (like Gauss trimr)
ztcrit     - returns critical values for adf test
```

The Gibbs convergence diagnostic functions are in a subdirectory **gibbs**.

```
Gibbs sampling convergence diagnostics functions

--------- convergence testing functions ---------

apm       - Geweke's chi-squared test
coda      - convergence diagnostics
momentg   - Geweke's NSE, RNE
raftery   - Raftery and Lewis program Gibbsit for convergence

--------- demonstration programs ----------------

apm_d     - demonstrates apm
coda_d    - demonstrates coda
momentg_d - demonstrates momentg
raftery_d - demonstrates raftery

--------- support functions ---------------------

prt_coda  - prints coda, raftery, momentg, apm output  (use prt)
empquant  - These were converted from:
indtest   - Rafferty and Lewis FORTRAN program.
mcest     - These function names follow the FORTRAN subroutines
mctest    -
ppnd      -
thin      -
```

Distribution functions are in the subdirectory **distrib**.

```
Distribution functions library

------- pdf, cdf, inverse functions -----------

beta_cdf  - beta(a,b) cdf
beta_inv  - beta inverse (quantile)
beta_pdf  - beta(a,b) pdf
bino_cdf  - binomial(n,p) cdf
bino_inv  - binomial inverse (quantile)
bino_pdf  - binomial pdf
chis_cdf  - chisquared(a,b) cdf
chis_inv  - chi-inverse (quantile)
chis_pdf  - chisquared(a,b) pdf
```

```
chis_prb  - probability for chi-squared statistics
fdis_cdf  - F(a,b) cdf
fdis_inv  - F inverse (quantile)
fdis_pdf  - F(a,b) pdf
fdis_prb  - probabililty for F-statistics
gamm_cdf  - gamma(a,b) cdf
gamm_inv  - gamma inverse (quantile)
gamm_pdf  - gamma(a,b) pdf
hypg_cdf  - hypergeometric cdf
hypg_inv  - hypergeometric inverse
hypg_pdf  - hypergeometric pdf
logn_cdf  - lognormal(m,v) cdf
logn_inv  - lognormal inverse (quantile)
logn_pdf  - lognormal(m,v) pdf
logt_cdf  - logistic cdf
logt_inv  - logistic inverse (quantile)
logt_pdf  - logistic pdf
norm_cdf  - normal(mean,var) cdf
norm_inv  - normal inverse (quantile)
norm_pdf  - normal(mean,var) pdf
pois_cdf  - poisson cdf
pois_inv  - poisson inverse
pois_pdf  - poisson pdf
stdn_cdf  - std normal cdf
stdn_inv  - std normal inverse
stdn_pdf  - std normal pdf
tdis_cdf  - student t-distribution cdf
tdis_inv  - student t inverse (quantile)
tdis_pdf  - student t-distribution pdf
tdis_prb  - probabililty for t-statistics

------- random samples ----------------------

beta_rnd  - random beta(a,b) draws
bino_rnd  - random binomial draws
chis_rnd  - random chi-squared(n) draws
fdis_rnd  - random F(a,b) draws
gamm_rnd  - random gamma(a,b) draws
hypg_rnd  - random hypergeometric draws
logn_rnd  - random log-normal draws
logt_rnd  - random logistic draws
nmlt_rnd  - left-truncated normal draw
nmrt_rnd  - right-truncated normal draw
norm_crnd - contaminated normal random draws
norm_rnd  - multivariate normal draws
pois_rnd  - poisson random draws
tdis_rnd  - random student t-distribution draws
unif_rnd  - random uniform draws (lr,rt) interval
wish_rnd  - random Wishart draws
```

```
 -------- demonstration and test programs ------

 beta_d    - demo of beta distribution functions
 bino_d    - demo of binomial distribution functions
 chis_d    - demo of chi-squared distribution functions
 fdis_d    - demo of F-distribution functions
 gamm_d    - demo of gamma distribution functions
 hypg_d    - demo of hypergeometric distribution functions
 logn_d    - demo of lognormal distribution functions
 logt_d    - demo of logistic distribution functions
 pois_d    - demo of poisson distribution functions
 stdn_d    - demo of std normal distribution functions
 tdis_d    - demo of student-t distribution functions
 trunc_d   - demo of truncated normal distribution function
 unif_d    - demo of uniform random distribution function

-------- support functions ---------------------

 betacfj   - used by fdis_prb
 betai     - used by fdis_prb
 bincoef   - binomial coefficients
 com_size  - test and converts to common size
 gammalnj  - used by fdis_prb
 is_scalar - test for scalar argument
```

Optimization functions are in the subdirectory **optimize**.

```
Optimization functions library

 --------------- optimization functions -----------------

 dfp_min    - Davidson-Fletcher-Powell
 frpr_min   - Fletcher-Reeves-Polak-Ribiere
 maxlik     - general all-purpose optimization routine
 pow_min    - Powell conjugate gradient
 solvopt    - yet another general purpose optimization routine

 --------------- demonstration programs -----------------

 optim1_d     - dfp, frpr, pow, maxlik demo
 optim2_d     - solvopt demo
 optim3_d     - fmins demo

 --------------- support functions ----------------------

 apprgrdn      - computes gradient for solvopt
 box_like1     - used by optim3_d
 gradt         - computes gradient
```

```
hessian       - evaluates hessian
linmin        - line minimization routine (used by dfp, frpr, pow)
stepsize      - stepsize determination
tol_like1     - used by optim1_d, optim2_d
updateh       - updates hessian
```

A library of spatial econometrics functions are in the subdirectory **spatial**.

```
------- spatial econometrics functions -----------

casetti    - Casetti's spatial expansion model
darp       - Casetti's darp model
far        - 1st order spatial AR model    - y = pWy + e
far_g      - Gibbs sampling Bayesian far model
gwr        - geographically weighted regression
bgwr       - Bayesian geographically weighted regression
lmerror    - LM error statistic for regression model
lmsar      - LM error statistic for sar model
lratios    - Likelihood ratio statistic for regression models
moran      - Moran's I-statistic
sac        - spatial model  - y = p*W1*y + X*b + u, u = c*W2*u + e
sac_g      - Gibbs sampling Bayesian sac model
sacp_g     - Gibbs sampling Bayesian sac probit model
sact_g     - Gibbs sampling Bayesian sac tobit model
sar        - spatial autoregressive model  - y = p*W*y + X*b + e
sar_g      - Gibbs sampling Bayesian sar model
sarp_g     - Gibbs sampling Bayesian sar probit model
sart_g     - Gibbs sampling Bayesian sar tobit model
sem        - spatial error model  - y = X*b +u, u=c*W + e
sem_g      - Gibbs sampling Bayesian spatial error model
semp_g     - Gibbs sampling Bayesian spatial error probit model
semt_g     - Gibbs sampling Bayesian spatial error tobit model
semo       - spatial error model (optimization solution)
sdm        - spatial Durbin model  y = a + X*b1 + W*X*b2 + e
sdm_g      - Gibbs sampling Bayesian sdm model
sdmp_g     - Gibbs sampling Bayesian sdm probit model
sdmt_g     - Gibbs sampling Bayesian sdm tobit model
slag       - creates spatial lags
walds      - Wald test for regression models
xy2cont    - constructs a contiguity matrix from x-y coordinates

------- demonstration programs  -----------
casetti_d  - Casetti model  demo
darp_d     - Casetti darp demo
darp_d2    - darp for all data observations
far_d      - demonstrates far using a small data set
far_d2     - demonstrates far using a large data set
```

```
far_gd     - far Gibbs sampling with small data set
far_gd2    - far Gibbs sampling with large data set
gwr_d      - geographically weighted regression demo
gwr_d2     - GWR demo with Harrison-Rubinfeld Boston data
bgwr_d     - demo of Bayesian GWR
bgwr_d2    - BGWR demo with Harrison-Rubinfeld Boston data
lmerror_d  - lmerror demonstration
lmsar_d    - lmsar demonstration
lratios_d  - likelihood ratio demonstration
moran_d    - moran demonstration
sac_d      - sac model demo
sac_d2     - sac model demonstration large data set
sac_gd     - sac Gibbs sampling demo
sac_gd2    - sac Gibbs demo with large data set
sacp_gd    - sac Gibbs probit demo
sact_gd    - sac Gibbs tobit demo
sact_gd2   - sac tobit right-censoring demo
sar_d      - sar model demonstration
sar_d2     - sar model demonstration large data set
sar_gd     - sar Gibbs sampling demo
sar_gd2    - sar Gibbs demo with large data set
sarp_gd    - sar probit Gibbs sampling demo
sart_gd    - sar tobit model Gibbs sampling demo
sart_gd2   - sar tobit right-censoring demo
sdm_d      - sdm model demonstration
sdm_d2     - sdm model demonstration large data set
sdm_gd     - sdm Gibbs sampling demo
sdm_gd2    - sdm Gibbs demo with large data set
sdmp_g     - sdm Gibbs probit demo
sdmt_g     - sdm Gibbs tobit demo
sem_d      - sem model demonstration
sem_d2     - sem model demonstration large data set
sem_gd     - sem Gibbs sampling demo
sem_gd2    - sem Gibbs demo with large data set
semo_d     - semo function demonstration
semo_d2    - semo demo with large data set
semp_gd    - sem Gibbs probit demo
semt_gd    - sem Gibbs tobit demo
semt_gd2   - sem tobit right-censoring demo
slag_d     - demo of slag function
walds_d    - Wald test demonstration
xy2cont_d  - xy2cont demo

------- support functions  -----------
anselin.dat- Anselin (1988) Columbus crime data
boston.dat - Harrison-Rubinfeld Boston data set
latit.dat  - latittude for HR data
longi.dat  - longitude for HR data
c_far      - used by far_g
```

```
c_sem      - used by sem_g
c_sar      - used by sar_g
c_sdm      - used by sdm_g
c_sac      - used by sac_g
darp_lik1  - used by darp
darp_lik2  - used by darp
elect.dat  - Pace and Barry 3,107 obs data set
ford.dat   - Pace and Barry 1st order contiguity matrix
f_far      - far model likelihood (concentrated)
f_sac      - sac model likelihood (concentrated)
f_sar      - sar model likelihood (concentrated)
f_sem      - sem model likelihood (concentrated)
f_sdm      - sdm model likelihood (concentrated)
f2_far     - far model likelihood
f2_sac     - sac model likelihood
f2_sar     - sar model likelihood
f2_sem     - sem model likelihood
f3_sem     - semo model likelihood
f2_sdm     - sdm model likelihood
normxy     - isotropic normalization of x-y coordinates
prt_gwr    - prints gwr_reg results structure
prt_spat   - prints results from spatial models
scoref     - used by gwr
wmat.dat   - Anselin (1988) 1st order contiguity matrix
```