5

Spatially Adjusted Regression and Related Spatial Econometrics

LEARNING OBJECTIVES:

- To reformulate linear regression models to account for spatial autocorrelation
- To reformulate binomial/logistic regression models to account for spatial autocorrelation
- To reformulate Poisson regression models to account for spatial autocorrelation
- To differentiate between static geographic distributions and spatial interaction cases

Regression analysis seeks to establish an equation for predicting some response variable, Y, from a set of P covariates, X_1, X_2, …, X_P. One statistical problem is to estimate the coefficients for these covariates in order to construct this prediction equation. Classical statistics attaches a probability distribution to the residuals (i.e., differences between observed and predicted values of Y) of this prediction equation. Spatial statistics modifies this situation by specifying a prediction function that has Y on both sides of the equation. In other words, a value at location i is at least partially a function of the values of Y at nearby locations. This conceptualization captures the essence of spatial autocorrelation.

5.1. Linear regression

Cliff and Ord (1973) furnish much of the seminal work for a linear regression model. Griffith (1993b) details the translation of a range of linear regression model specifications, from ANOVA, through product moment correlation coefficients, to two-group discriminant function analysis. This section features the autoregressive model most commonly employed in spatial statistics, namely the simultaneous autoregressive (SAR) specification,

$$\mathbf{Y} = \rho\mathbf{W}\mathbf{Y} + (\mathbf{I} - \rho\mathbf{W})\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{5.1}$$

which is the spatial statistical counterpart to the standard linear regression model specification of

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{5.2}$$

where $\mathbf{W}$ is the row standardized geographic connectivity matrix (see Section 4.2), $\mathbf{I}$ is an $n \times n$ identity matrix, ρ is the spatial autocorrelation parameter, $\boldsymbol{\beta}$ is a $(P + 1) \times 1$ vector of regression coefficients (including the intercept term), and $\boldsymbol{\varepsilon}$ is an $n \times 1$ vector of iid $N(0, \sigma^2)$ random variables, which may be written in matrix form as the multivariate normal distribution $MVN(\mathbf{0}, \mathbf{I}\sigma^2)$. Positing a row standardized geographic connectivity matrix $\mathbf{W}$ restricts positive spatial autocorrelation values of ρ to be in the interval [0, 1). The presence of non-zero spatial autocorrelation means equation (5.2) has the modified specification

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + (\mathbf{I} - \rho\mathbf{W})^{-1}\boldsymbol{\varepsilon}, \tag{5.3}$$

where the spatial linear operator $(\mathbf{I} - \rho\mathbf{W})^{-1}$ embeds spatial autocorrelation into the error term, and hence the calculated residual. In other words, equation (5.2) becomes

$$\mathbf{Y}, = \mathbf{X}\boldsymbol{\beta} + [\rho\mathbf{W}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) + \boldsymbol{\varepsilon}] = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\xi},$$

with $\boldsymbol{\xi}$ no longer being distributed as $MVN(\mathbf{0}, \mathbf{I}\sigma^2)$. The conventional (i.e., ordinary least squares, or OLS) estimates $\mathbf{b}$ of $\boldsymbol{\beta}$ remain unbiased. But spatial autocorrelation alters their sampling distribution variances (i.e., their standard errors) as well as the regression model R^2 value (see Dutilleul et al., 2008).

Although equation (5.1) is properly specified, its estimation requires employment of a weighting function that achieves two goals: first, it ensures that the probabilities in both the autocorrelated and its corresponding unautocorrelated mathematical space integrate to 1; and second, it restricts the value of ρ to the interval [0, 1). This estimation version of equation (5.1) is (Griffith, 1988)

$$\frac{\mathbf{Y}}{\exp\left(\sum_{i=1}^{n}(1-\rho\lambda_i)\right)} = \frac{\rho\mathbf{WY}}{\exp\left(\sum_{i=1}^{n}(1-\rho\lambda_i)\right)} - \frac{\rho\mathbf{WX\beta}}{\exp\left(\sum_{i=1}^{n}(1-\rho\lambda_i)\right)} + \frac{\mathbf{X\beta}}{\exp\left(\sum_{i=1}^{n}(1-\rho\lambda_i)\right)} + \frac{\boldsymbol{\varepsilon}}{\exp\left(\sum_{i=1}^{n}(1-\rho\lambda_i)\right)},$$

where the λ_i are the n eigenvalues of matrix $\mathbf{W}$. Estimation requires nonlinear techniques because ρ appears in both the numerator and the denominator of the first two terms on the right-hand side of this equation. Furthermore, the derivatives are not straightforward, and their calculation is cumbersome. These technical complications become hidden in software implementations of spatial autoregression estimation procedures.

Consider the 2007 geographic distribution of number of farms utilizing irrigation. The Box–Cox power transformation better aligning it with a bell-shaped curve is $\ln(Y/\text{area} + 0.04)$; normal diagnostic probability increases from < 0.001 to 0.611 (see Section 4.1.4). Results for regressing this response variable on average annual rainfall include the following:

Estimation	$\hat{\rho}$	$\hat{\beta}_0$	$\hat{\beta}_1$	(pseudo-)R^2	Normality probability
OLS	0	–0.2067	–0.0207	0.138	0.210
		(0.4360)	(0.0061)		
SAR	0.5760	–0.4327	–0.0174	0.379	0.213
		(0.5129)[1]	(0.0071)		

These results imply the presence of moderately strong positive spatial autocorrelation, and illustrate the effects on standard errors and the increase in variance accounted for (e.g., R^2).

[1] Standard error results may differ slightly because of differences in the nonlinear optimization algorithm used and/or the type of standard error computed (e.g., asymptotic).

Spatial autocorrelation can affect a pairwise correlation coefficient calculated for two georeferenced variables (see Clifford et al., 1989), again largely in terms of its standard error. The Pearson product moment correlation coefficient for the power-transformed 2007 geographic distribution of number of farms utilizing irrigation and the average annual rainfall is –0.3719. The SAR spatial autocorrelation parameter estimates for these two variables respectively are 0.6469 and 0.8239. In order to adjust for these levels of spatial autocorrelation, the correlation coefficient to calculate is between the variables

$$(\mathbf{I} - 0.6469\ \mathbf{W})\langle \ln(Y/\text{area} + 0.04)\rangle \text{ and } (\mathbf{I} - 0.8239\ \mathbf{W})\ \mathbf{X},$$

where the wide angle brackets denote a vector. Adjusting for the latent spatial autocorrelation in this way reduces the correlation coefficient to –0.2426. In other words, spatial autocorrelation makes the relationship between these two variables look stronger than it actually is in the superpopulation.

In Section 4.1.4, the initial statistical decision is that a difference exists in regional means of the power-transformed 2007 geographic distribution of number of farms utilizing irrigation. Accounting for average annual rainfall reverses this decision. But after adjusting for spatial autocorrelation of 0.6469 in this transformed variable, the ANOVA results change as follows:

Source	df	Sum of squares	Mean square	*F*-ratio	Pr > *F*
Regions	4	2.2023	0.5506	1.35	0.2603
Error	68	27.7215	0.4077		
Corrected total	72	29.9239			

The normality diagnostic statistic probabilities are:

Region	San Juan	Arecibo	Mayaguez	Ponce	Caguas
Probability	0.5666	0.1938	0.2120	0.9804	0.8066

Meanwhile, Levene's test yields:

Source	df	Sum of squares	Mean square	*F*-ratio	Pr > *F*
Regions	4	0.2654	0.0713	0.47	0.7541
Error	68	10.2194	0.1503		

These model diagnostics support the underlying assumptions for model-based inference. Consequently, the initial differences detected in regional means disappear after accounting for spatial autocorrelation. This finding explains why adjusting for average annual rainfall, with its high level of positive spatial autocorrelation, also reverses the statistical decision.

The two-group discriminant function analysis (DFA) model is the final classical linear model treated here (see Tatsuoka, 1988). It also can be formulated as a linear regression specification, for which the response variable is binary (i.e., takes the value 0 or 1). The bivariate regression results are as follows (with standard errors in parentheses):

Estimation	$\hat{\rho}$	$\hat{\beta}_0$	$\hat{\beta}_1$	(pseudo-)R^2
OLS	0	–0.4440	0.0125	0.124
		(0.2795)	(0.0039)	
SAR	0.7426	–1.5004	0.0272	0.494
		(0.3276)	(0.0044)	

The normality assumption no longer is valid with this analysis; the response variable is binary, not continuous. In addition, the linear model specification does not guarantee that the 0–1 response values are contained in the interval [0, 1]. Nevertheless, the coefficients are proportional to discriminant function analysis coefficients in multivariate statistical theory.

5.2. Nonlinear regression

In the preceding section, implementation of spatial autoregressive models requires nonlinear regression techniques. But the error term assumption is still the normal

probability model. Nonlinear regression also involves non-normal probability models, such as those for binomial and Poisson random variables. The generalized linear model (GLM) is the implementation of the latter models.

Eigenvector spatial filtering furnishes a sound methodology for estimating non-normal probability models with georeferenced data containing non-zero spatial autocorrelation. This methodology accounts for spatial autocorrelation in random variables by incorporating heterogeneity into parameters in order to model non-homogeneous populations. It renders a mixture of distributions that can be used to model observed georeferenced data whose various characteristics differ from those that are consistent with a single, simple, underlying distribution with constant parameters across all observations. The aim of this technique is to capture spatial autocorrelation effects with a linear combination of spatial proxy variables – namely, eigenvectors – rather than to identify a global spatial autocorrelation parameter governing average direct pairwise correlations between selected observed values. As such, it utilizes the misspecification interpretation of spatial autocorrelation, which assumes that spatial autocorrelation is induced by missing exogenous variables, which themselves are spatially autocorrelated, and hence relates to heterogeneity.

Eigenvector spatial filtering conceptualizes spatial dependency as a common factor that is a linear combination of synthetic variates summarizing distinct features of the neighbors' geographic configuration structure for a given georeferenced dataset. The synthetic variates may be the eigenvectors of the matrix $(\mathbf{I} - \mathbf{1}\mathbf{1}^{\mathrm{T}}/n)\mathbf{C}(\mathbf{I} - \mathbf{1}\mathbf{1}^{\mathrm{T}}/n)$ discussed in Section 4.2.1, a term appearing in the Moran Coefficient (MC) index of spatial autocorrelation.[2] De Jong et al. (1984) show that the extreme eigenfunctions of this matrix define the most extreme levels possible of spatial autocorrelation for a given surface partitioning, a result in combination with Tiefelsdorf and Boots (1995) and Griffith (1996) that attaches conceptual meaning to the extracted synthetic variates. These variates summarize distinct map pattern features because they are both orthogonal and uncorrelated.

The eigenfunction problem solution is similar to that obtained with principal components analysis in which the covariance matrix is given by $[\mathbf{I} + k(\mathbf{I} - \mathbf{1}\mathbf{1}^{\mathrm{T}}/n) \times \mathbf{C}(\mathbf{I} - \mathbf{1}\mathbf{1}^{\mathrm{T}}/n)]$, for some suitable value of k; sequential, rather than simultaneous, variance extraction is desired in order to preserve interpretation of the extremes. This solution relates to the following decomposition theorem (after Tatsuoka, 1988, p.141):

> the first eigenvector, say $\mathbf{E}_1$, is the set of numerical values that has the largest MC achievable by any set of real numbers for the spatial arrangement defined by matrix $\mathbf{C}$; the second eigenvector is the set of real numbers that has the largest achievable MC by

[2] The Geary ratio counterpart to this matrix also could be used.

any set that is uncorrelated with $\mathbf{E}_1$; the third eigenvector is the third such set of values; and so on through $\mathbf{E}_n$, the set of values that has the largest negative MC achievable by any set that is uncorrelated with the preceding $(n - 1)$ eigenvectors.

The corresponding eigenvalues index these levels of spatial autocorrelation: $\text{MC} = n\mathbf{E}^{\text{T}}\mathbf{CE}/\mathbf{1}^{\text{T}}\mathbf{C1}$. But, in contrast to principal components analysis, rather than using the resulting eigenvectors to construct linear combinations of attribute variables (which would be the n 0–1 binary indicator variables forming matrix $\mathbf{C}$), the eigenvectors themselves (instead of principal components scores) are the desired synthetic variates, each containing n elements, one for each areal unit (i.e., location). Figure 5.1 illustrates global, regional, and local geographic patterns of spatial autocorrelation portrayed by selected eigenvectors.

5.2.1. Binomial/logistic regression

The preceding discriminant function analysis can be recast as a logistic regression problem, which ensures that the predicted values corresponding to the observed 0–1 values are contained in the interval [0, 1]. The Bernoulli (i.e., binomial with number of trials (N_{tr}) equal to 1) probability model underlies

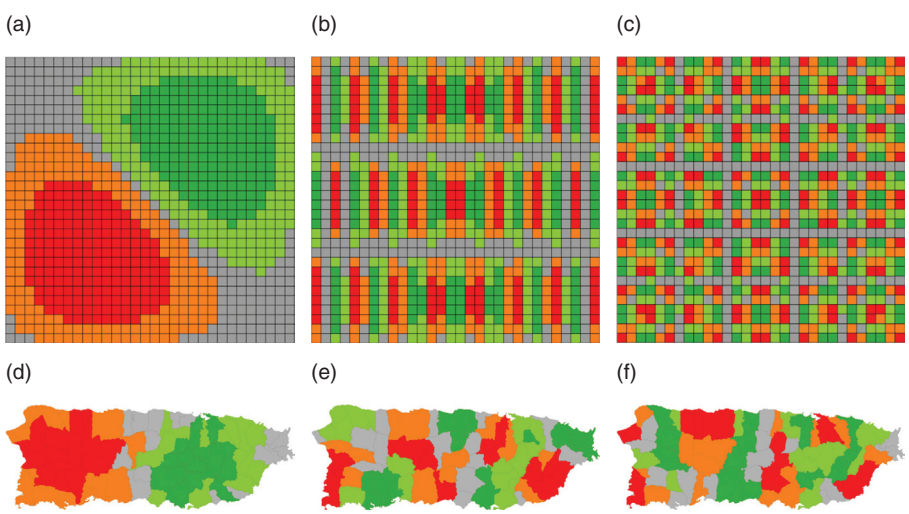

Figure 5.1 Spatial filter map patterns for (a–c) a regular square tessellation (top) and (d–f) the Puerto Rico municipality surface partitioning; quintile eigenvector value classes (which are relative to a factor of –1) range from dark green to dark red. (a,d) Global map pattern. (b,e) Regional map pattern. (c,f) Local map pattern.

this model specification. The spatial filter (SF) conceptualization enables this model to be implemented with a GLM while still accounting for positive spatial autocorrelation.

Figure 5.2a portrays the coastal lowlands/interior highlands classification scheme. This distribution contains weak positive spatial autocorrelation: MC = 0.2374 (standard error 0.075), Geary Ratio (GR) = 0.8120. Average rainfall accounts for roughly 12% of the variation in the binary classification. A stepwise selection procedure adjusting for non-constant variance includes nine eigenvectors in the model to account for spatial autocorrelation. This spatial autocorrelation component accounts for an additional approximately 64% of the variation in the classification scheme. The residuals for the binary predicted values contain only trace amounts of spatial autocorrelation: MC = –0.1244, GR = 1.1879. Figure 5.2b is the geographic distribution of the estimated probability of a municipality being a member of the interior highlands group. Rounding all values between 0 and 0.5 to 0, and all values between 0.5 and 1 to 1, Figure 5.2c portrays the predicted classification scheme. Figures 5.2a and 5.2c are very similar.

The preceding example illustrates a dichotomous classification case. But many georeferenced variables are percentages. For these variables, a binomial model specification is appropriate. Such a specification involves both a lower limit (i.e., 0) as well as an upper limit (i.e., N_{tr}) on counts. The percentage of farms in a municipality utilizing irrigation furnishes one example of this type of variable (Figure 5.3a). This geographic distribution contains weak positive spatial autocorrelation: MC = 0.1533, GR = 0.6665. Because the variable is linked to a binomial probability model, the relationship between its mean and its variance is given as follows: variance = $(1 - p)$ mean. Overdispersion occurs when deviations from this relationship are such that variance > $(1 - p)$ mean. The deviance statistic that indexes this overdispersion has an ideal value of 1. For the Puerto Rico farm irrigation example, average rainfall accounts for roughly 29% of the geographic variance in percentage of farms utilizing irrigation, with an accompanying deviance statistic of 9.67 (i.e., excessive overdispersion). Spatial

Figure 5.2 The coastal lowlands/interior highlands classification scheme. (a) Geographic distribution of the observed classification. (b) Predicted probabilities for the observed classification. (c) Predicted classification.

Figure 5.3 Gray scale darkness is directly proportional to values. (a) Geographic distribution of percentage of farms utilizing irrigation. (b) Predicted geographic distribution of percentage of farms utilizing irrigation. (c) Constructed spatial filter.

autocorrelation (Figure 5.3c) accounts for an additional roughly 32% of geographic variance, reducing the deviance statistic to 4.42 (i.e., a substantial reduction, but still indicating excessive overdispersion). The SF (Figure 5.3c) represents strong positive spatial autocorrelation: MC = 0.8097, GR = 0.2598. Meanwhile, the model residuals contain little spatial autocorrelation: MC = –0.1415, GR = 0.9572. Of note is that the GR values[3] here suggest the presence of some data complications (e.g., messiness, dirtiness, noisiness).

5.2.2. Poisson/negative binomial regression

One difference between Poisson and binomial variables is that the only bound the former have is a lower one of 0. Counts for a Poisson variable are not constrained by an upper bound (i.e., N_{tr}), and can go to infinity. Another difference is that the presence of overdispersion can be conceptualized as a non-constant mean, which when characterized by a gamma probability model converts a Poisson into a negative binomial variable.

5.2.2.1. Geographic distributions

The Box–Cox power transformation for 2007 farm count density (Y) can be recast as a Poisson variable for farm counts coupled with an area offset variable (i.e., a variable whose regression coefficient is set to 1 rather than being estimated). Because the area variable is introduced into an exponential function, it must be done in its natural logarithm form (i.e., $e^{\ln(x)} = x$). A model specification of this type avoids specification error arising from employing a bell-shaped curve with a power-transformed variable, as well as avoiding the need to calculate a back-transformation after completing an analysis (see equation [4.1]).

[3] A heuristic test for well-behaved data is that MC + GR should be very close to 1.

The Box–Cox power transformation renders the variable $(Y - 0.12)^{0.38}$ as approximately normally distributed; the Shapiro–Wilk probability, P(S-W), increases from less than 0.0001 to 0.5688. Regressing the transformed variable on mean annual rainfall yields a set of predicted values together with a mean squared error of 0.2550 (i.e., $\hat{\sigma}^2$). Mean annual rainfall accounts for roughly 13% of the variance in the transformed variable. The back-transformation involves the exponent 1/0.38 = 2.6315789. Equation (4.2) yields

$$C_1 = \prod_{h=1}^{1} \frac{0.5}{h}\left[-\frac{1}{4} + \left(\frac{1}{0.38} - 2h + \frac{3}{2}\right)^2\right] = 2.14681$$

and equation (4.1) yields, for the n values of E(Y),

$$\hat{\mu}_i^{1/0.38} + \sum_{j=1}^{1} (2.14681)\hat{\mu}_i^{2-2j}\left(\sqrt{0.25496}\right)^{2j} + 0.12, \qquad i = 1, 2, \ldots, n,$$

which accounts for roughly 16% of the variance in Y (Figure 5.4a). The range of these back-transformed predicted values is roughly 2 to 7, whereas that for the observed values is 0 to 15. The bivariate regression of Y on these back-transformed predicted values renders an intercept of –0.8048 and a slope of 1.2244.

Employing the Poisson model specification yields a deviance statistic of nearly 94, indicating that the variance and the mean are not equal. Respecifying this Poisson model as a negative binomial model (i.e., a Poisson random variable with a gamma-distributed mean) reduces this deviance statistic value to 1.11 (which mostly affects the calculation of standard errors); accordingly, $\hat{\sigma}^2 = \hat{\mu} + 0.5067\hat{\mu}^2$. Densities computed with the predicted counts account for about 15% of the variance in Y (Figure 5.4b). The range of these predicted values is roughly 2 to 9, an improvement upon the normal approximation results. The bivariate regression of Y on these predicted values renders an intercept of 0.5078 and a slope of 0.8858, both of which are closer to their respective ideal values of 0 and 1 than the normal approximation results. The difference between these GLM and the bell-shaped curve results is attributable to specification error: the paired results are reasonably similar (i.e., the approximation is very good), but have conspicuous differences.

Positive spatial autocorrelation can be detected not only in variable Y (MC = 0.3343, GR = 0.7732), but also in the residuals from both model specifications (normal approximation MC = 0.3847, GR = 0.7104; negative binomial MC = 0.4040, GR = 0.6788). Constructing an SF to account for this

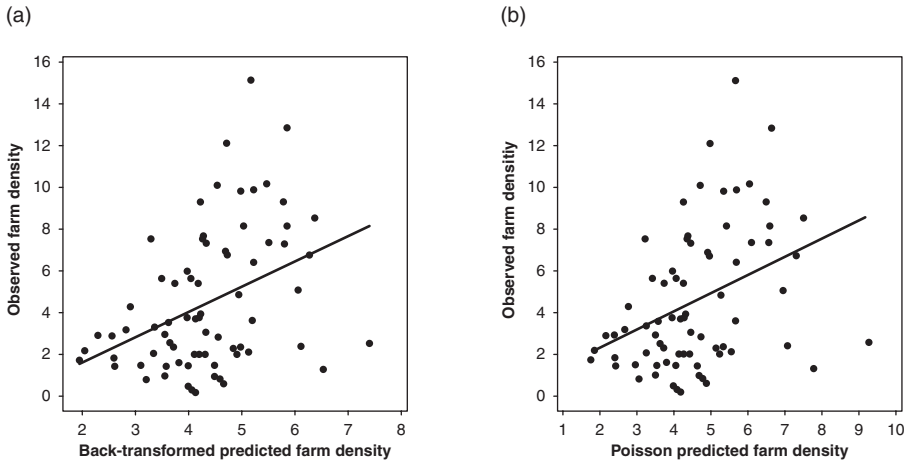

Figure 5.4 Observed versus predicted scatterplots. (a) Normal probability model results. (b) Poisson probability model results.

spatial autocorrelation results in eight vectors being selected for the normal approximation specification, and five of these same eight vectors being selected for the negative binomial specification (using $\alpha = 0.01$ in this second case). Now roughly 73% of the variation is accounted for in the transformed variable, and roughly 72% of the variation in Y (Figure 5.5a) after calculating the back-transformation. The range of these back-transformed predicted values improves to roughly 1 to 14. The bivariate regression of Y on these back-transformed predicted values renders an intercept of 0.1867 and a slope of 0.9635. In contrast, the negative binomial specification yields predicted values that account for roughly 68% of the variation in Y, and produces a 1 to 14 range of predicted values. Its bivariate regression results include an intercept of 0.4136 and a slope of 0.8993. In other words, the normal approximation outperforms the GLM.

Equation (2.7) furnishes the expected value for normally distributed residuals from a linear regression analysis. Here the value is –0.1059 for the normal approximation regression analysis, and –0.0777 for the negative binomial regression analysis. Spatial autocorrelation index values for the back-transformed residuals are MC = –0.1751 and GR = 1.2692. In contrast, spatial autocorrelation index values for the negative binomial residuals are MC = –0.1276 and GR = 1.2484. The GR values suggest possible overcorrection by the SFs for detected spatial autocorrelation.

These results can be extended to ANOVA problems by introducing the appropriate indicator variables into the regression model specifications,

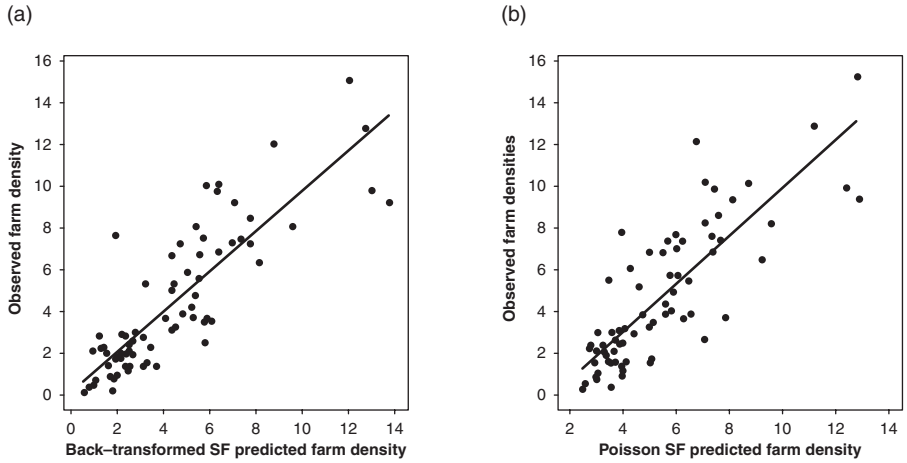

Figure 5.5 Observed versus predicted scatterplots. (a) Spatial filter normal probability model results. (b) Spatial filter Poisson probability model results.

allowing the assumed ANOVA probability model to be non-normal. The transformed variable results for a one-way ANOVA, in which the classification is based upon urban and non-urban municipalities, yields the following results:

unadjusted

for variable Y: P(S-W) = 0.5376 and 0.7834

for variable Y: P(Levene statistic) = 0.2976

ANOVA $F = (-6.18)^2 = 38.19$, $P(F) < 0.0001$

adjusted for spatial autocorrelation[4]

for regression residuals: P(S-W) = 0.5054 and 0.7594

for regression residuals: P(Levene statistic) = 0.3216

ANOVA $F = (-2.94)^2 = 8.64$, $P(F) = 0.0046$

In other words, a difference in farm densities between the urban and non-urban groups is expected to exist in the superpopulation. Meanwhile, the negative binomial

[4] The spatial filter construction with stepwise regression includes one additional eigenvector when the model specification includes the classification variable.

yields a significant regression coefficient for the difference between the two indicator variables ($P(b_{class}) = 0.0005$). The deviance statistic is 1.22, while the individual group deviance statistics are 1.29 and 1.49. Overall, both analyses furnish the same statistical inference, and indicate that this implication is a sound model-based inference.

5.2.2.2. Geographic flows: a journey-to-work example

Because the n^2 geographic flows between locations are counts, they constitute a Poisson random variable. Each flow tends to be positively correlated with the size of its origin and the size of its destination, and negatively correlated with the size of the intervening distance. In other words, as the number of workers at a location increases, the number leaving that origin location to travel to work tends to increase. Similarly, as the number of jobs at a location increases, the number of workers arriving at that destination to work tends to increase. And, as the distance separating an origin and a destination location increases, the number of workers tending to travel from that origin to that destination tends to decrease. The following simple equation furnishes a very good description of this situation (see Section 4.2.2; Griffith, 2011):

$$F_{ij} \approx \kappa A_i O_i B_j D_j e^{-\gamma d_{ij}} e^{SF_{O_i \times Dj}}, \tag{5.4}$$

where

- F_{ij} denotes the flow (e.g., number of workers) between locations i and j;
- κ is a constant of proportionality;
- A_i denotes an origin balancing factor;
- O_i denotes the total amount of flow leaving from origin i (e.g., number of workers residing at an origin);
- B_j denotes a destination balancing factor;
- D_j denotes the total amount of flow arriving at destination j (e.g., the number of jobs available at a destination);
- d_{ij} denotes the distance separating origin i and destination j;
- γ denotes the global distance decay rate.
- SF_{O_i} denotes the origin i spatial filter accounting for spatial autocorrelation in flows, calculated by holding D_j constant in $SF_{O_i \times D_j}$
- SF_{D_j} denotes the destination j spatial filter accounting for spatial autocorrelation in flows, calculated by holding O_i constant in $SF_{O_i \times D_j}$

Selected results from the estimation of equation (5.4) for the Puerto Rico 2000 journey-to-work data (874,832 inter-municipality trips for $73^2 = 5{,}329$ dyads) include the following:

Set values	$\hat{\kappa}$	$\hat{\gamma}$	*Overdispersion*	*pseudo-R^2*
$SF_{O_i} = 0$, $SF_{D_j} = 0$, $A_i = 1$, $B_j = 1$	9.4×10^{-6}	0.1625	14.5227^2	0.8039
$SF_{O_i} = 0$, $SF_{D_j} = 0$	5.6×10^{-6}	0.2286	7.9801^2	0.9825
None	5.1×10^{-6}	0.2084	6.4750^2	0.9892

The spatial filter comprises 85 of 121 candidate eigenvectors (those with an MC of at least 0.25), from a total of 5,329 possible eigenvectors. These results illustrate the failure to estimate an accurate global distance decay parameter value when ignoring spatial autocorrelation in flows. Spatial autocorrelation in flows contributes to excess Poisson variation, too. Adjusting for spatial autocorrelation in flows yields a better alignment of the largest predicted and observed values, which slightly improves the pseudo-R^2 value (Figure 5.6). The following bivariate regression results quantify this improved alignment, which signifies a reduction in model misspecification:

Set values	*Intercept*	*Slope*	*PRESS/ESS*[5]	*Predicted R^2*
$SF_{O_i} = 0$, $SF_{D_j} = 0$, $A_i = 1$, $B_j = 1$	95.36	0.42	3.53	0.3086
$SF_{O_i} = 0$, $SF_{D_j} = 0$	7.09	0.96	1.41	0.9753
None	4.08	0.98	1.21	0.9876

The ideal values here are 0 for the intercept, 1 for the slope, 1 for the PRESS/ESS ratio, and 1 for the predicted R^2.

Figure 5.7 portrays the balancing factors and spatial filters for the Puerto Rico journey-to-work example. The A_i and B_j values display conspicuous geographic patterns (Figures 5.7a,b). The origin balancing factors display an east–west trend from values between 0 and 1 (deflating departure flows), to values greater than 1 (inflating departure flows). The destination balancing factors display the opposite trend. Spatial autocorrelation accounts for roughly 90% of the variation in each of these geographic distributions. Meanwhile, the origin spatial filter (Figure 5.7c) contrasts the San Juan metropolitan region

[5] ESS is error sum of squares, PRESS is predicted error sum of squares.

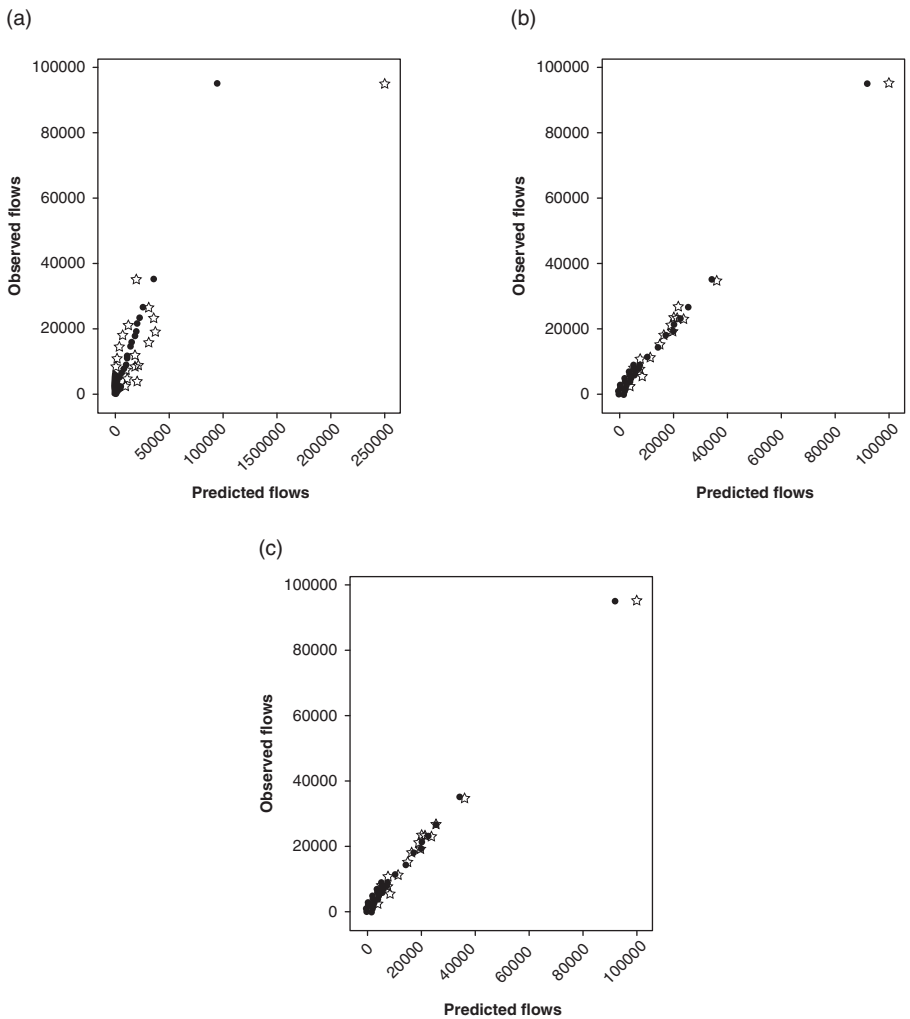

Figure 5.6 Scatterplots of the journey-to-work trips predicted by equation (5.4) and observed. (a) $SF_{O_i} = 0$, $SF_{D_j} = 0$, $A_i = 1$, $B_j = 1$. (b) $SF_{O_i} = 0$, $SF_{D_j} = 0$. (c) All parameters estimated. Solid circle denotes observed flow values; star denotes predicted flow values.

with the remainder of the island. This contrast is consistent with the origin balancing factors map pattern. The destination spatial filter highlights the four urban catchment areas (San Juan–Caguas, Arecibo, Mayaguez, and Ponce; see Figure 4.1).

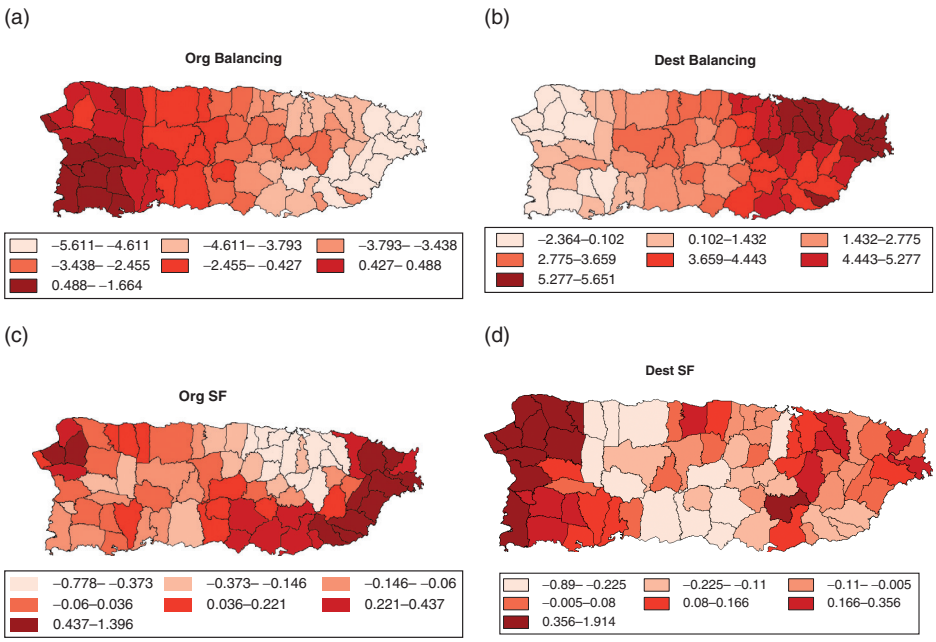

Figure 5.7 Geographic distributions of equation (5.4) terms. (a) Origin balancing factor, A_i. (b) Destination balancing factor, B_j. (c) Origin spatial filter. (d) Destination spatial filter. Darkness of gray scale is directly proportional to value.

5.3. R code for concept implementations

Computer Code 5.1 demonstrates implementations of the spatially adjusted regression models presented in this chapter. These implementations include eigenvector spatial filter specifications for the normal, binomial, Poisson, and negative binomial models. Standard stepwise regression methods in R, such as the *step* and *stepAIC* functions, make selections based upon such measures as the Akaike information criterion (AIC). The stepwise procedure with AIC tends to select more eigenvectors than are chosen by the traditional stepwise procedure based upon statistical significance. All data analyses in this chapter were performed with SAS, including the eigenvector selections. The R code implementations in this section utilize the *stepwise.forward* function (which is defined in the all_functions.R file) for the normal cases, and the *stepAIC* function for the non-normal distribution cases. Hence, when compared with the eigenvectors selected for the normal cases, slightly more eigenvectors tend to be selected with

stepAIC for the non-normal cases. Nevertheless, many eigenvectors are common to these two sets. In order to replicate the data analyses in this chapter, the results from traditional stepwise regression based upon statistical significance are presented as well as R code using the *stepAIC* function. Results obtained with R code need to be manually adjusted to match those obtained with SAS.

The *mapping.seq* function is utilized in order to avoid redundant and lengthy R code lines in Computer Code 5.1 when performing repetitive mapping tasks. This function is also defined in the all_functions.R file.

Computer Code 5.1. Implementing spatially adjusted regression and spatial interaction models

```
# load libraries and data
library(car)
library(spdep)
library(RColorBrewer)
library(classInt)

pr.f <- read.csv(file="PR-farm-data.csv")

# 5.1
ifarm.den07 <- pr.f$irr_farms_07/pr.f$area

y <- log(ifarm.den07 + 0.04)
rain <- pr.f$rain_mean

if.lm <- lm(y ~ rain)
summary(if.lm)
shapiro.test(resid(if.lm))

pr.nb <- read.gal("PuertoRico.GAL")
pr.listw <- nb2listw(pr.nb, style="W")
pr.listb <- nb2listw(pr.nb, style="B")
if.sar <- errorsarlm(y ~ rain, listw =
pr.listw)
summary(if.sar)
if.res <- residuals(if.sar)
shapiro.test(if.res)

cor(y, rain)

y.sar <- errorsarlm(y ~ 1, listw=pr.listw)
y.sar$lambda
x.sar <- errorsarlm(rain ~ 1, listw=
pr.listw)
x.sar$lambda

y.sa <- y - y.sar$lambda *
lag.listw(pr.listw,y)
rain.sa <- rain - x.sar$lambda *
lag.listw(pr.listw,rain)
cor(y.sa, rain.sa)

adm <- factor(pr.f$ADM, levels=1:5, labels=
c("San Juan", "Arecibo", "Mayaguez",
"Ponce", "Caguas"))
lm.if.sa <- lm(y.sa ~ adm)
anova(lm.if.sa)

sw.p <- function(x){
  shapiro.test(x)$p.value}
tapply(resid(lm.if.sa),adm,sw.p)
leveneTest(resid(lm.if.sa), adm,
center=mean)

ci <- pr.f$cl_ih
ci.lm <- lm(ci ~ rain, data=pr.f)
summary(ci.lm)
ci.sar <- errorsarlm(ci ~ rain,
listw=pr.listw)
summary(ci.sar)
```

```
Load car, spdep, RColorBrewer, and
classInt packages.

Read Puerto Rico farm data.

Calculate irrigated farm density in
2007.
Transform the density.
Get mean rainfall.

Run linear regression and summarize the
results.
Conduct a normality test.

Read spatial neighbor information.
Generate listw objects with W and B
styles.
Run a spatial autoregressive model and
summarize the results.

Get residuals and conduct Shapiro-Wilk
normality test.

Calculate correlation between the two
variables.
Estimate SAR spatial autocorrelation
parameters for the two variables.

Adjust for the latent spatial
autocorrelation in the variables.

Calculate correlation between the
adjusted variables.
Create a factor variable.

Conduct ANOVA for the spatial
autocorrelation adjusted farm
densities.
Create a function and conduct Shapiro-
Wilk test for each administrative
region.
Conduct Levene's test.

Get a binary variable.
Run a linear regression and summarize
the result.
Run an SAR model and summarize the
result.
```

`#5.2.1`	
`moran.test(ci, pr.listb)`	`Conduct spatial autocorrelation tests.`
`geary.test(ci, pr.listb)`	
`n <- length(pr.nb)`	`Generate eigenvalues and eigenvectors.`
`M <- diag(n) - matrix(1,n,n)/n`	`from a transformed spatial weight`
`B <- listw2mat(pr.listb)`	`matrix (MBM in the R codes).`
`MBM <- M %*% B %*% M`	
`eig <- eigen(MBM,symmetric=T)`	
`EV <- as.data.frame( eig$vectors[ , eig$values/eig$values[1] > 0.25])`	`Construct a candidate set of eigenvectors.`
`colnames(EV) <- paste("EV", 1:NCOL(EV), sep="")`	`Add column names for the eigenvectors.`
`ci.full <- glm(ci ~ rain + ., data=EV, family=binomial)`	`Conduct spatial filtering with` ***`stepAIC`*** `function.This selects more eigenvectors than a selection procedure based solely on significance.`
`ci.sf <- stepAIC(glm(ci ~ rain, data=EV, family=binomial), scope=list(upper= ci.full), direction="forward")`	
`ci.sf <- glm(ci ~ rain + EV4 + EV2 + EV7 + EV9 + EV6 + EV14 + EV13 + EV18 + EV12, data=EV, family=binomial)`	`Get a spatial filter model in the text which is constructed based on significance.`
`summary(ci.sf)`	`Summarize the result.`
`ci.sf.res <- round(residuals(ci.sf, type="response"))`	`Get the residuals of the spatial filter model.`
`moran.test(ci.sf.res , pr.listb)`	`Conduct spatial autocorrelation tests.`
`geary.test(ci.sf.res , pr.listb)`	
`pr <- readShapePoly("PuertoRico.shp")`	`Read Puerto Rico shapefile.`
`pal.wr <- c("white","red")`	`Set a color list with white and red.`
`cols.wr <- pal.wr[ci+1]`	`Find colors for each polygon.`
`plot(pr, col=cols.wr)`	`Plot the polygons with the colors.`
`leg <- c("coastal", "interior")`	`Set legend texts.`
`legend("bottomright", fill=pal.wr, legend=leg, bty="n")`	`Locate a legend.`
`pal.red <- brewer.pal(5,"Reds")`	`Create a color palette with 5 colors.`
`q5 <- classIntervals(ci.sf$fitted, 5, style="quantile")`	`Classify the fitted values into 5 classes with quantile option.`
`cols.red <-findColours(q5, pal.red)`	`Find colors for the polygons.`
`plot(pr, col=cols.red)`	`Plot the polygons with the colors.`
`brks <- round(q5$brks,3)`	`Get break information.`
`leg <- paste(brks[-6], brks[-1], sep=" - ")`	`Create legend texts.`
`legend("bottomright", fill=pal.red, legend=leg, bty="n")`	`Locate a legend.`
`cols.wr <- pal.wr[round(ci.sf$fitted)+1]`	`Convert the fitted values into a binary variable, then map them similarly.`
`plot(pr, col=cols.wr)`	
`leg <- c("coastal", "interior")`	
`legend("bottomright", fill=pal.wr, legend=leg,bty="n")`	
`# The percent of farms utilizing irrigation`	
`fp <- pr.f$irr_farms_02/pr.f$nofarms_02`	`Get irrigated farm densities in 2002.`
`fp.col <- cbind(pr.f$irr_farms_02, pr.f$nofarms_02-pr.f$irr_farms_02)`	`Create a dependent variable for binomial regression: (# of success, # of fail).`
`fp.base <- glm(fp.col ~ rain, family= quasibinomial)`	`Run a binomial regression.`
`disp <- summary(fp.base)$dispersion`	`Get Pearson-type overdispersion value.`
`fp.full <- glm(fp.col ~ rain + ., data=EV, family=binomial)`	`Conduct stepwise regression with` ***`stepAIC`***`.`

```
fp.sf <- stepAIC(glm(fp.col~rain, data=EV,
family=binomial), scale=disp, scope=
list(upper=fp.full), direction="forward")
fp.sf <- glm(fp.col ~ rain + EV1 + EV13 +
EV4 + EV12 + EV2 + EV15, data=EV,
family=quasibinomial)
summary(fp.sf)

moran.test(fp, pr.listb)
geary.test(fp, pr.listb)
summary(fp.base)$deviance/fp.base$df.residu
al
summary(fp.sf)$deviance/fp.sf$df.residual
summary(fp.sf)$dispersion

# Mapping the percentages
source("all_functions.R")
mapping.seq(pr, fp, 5)

# Mapping the predicted
mapping.seq(pr, fp.sf$fitted, 5)

# Mapping the spatial filter
sfilter <-
as.matrix(EV[,c(1,13,4,12,2,15)]) %*%
as.matrix(fp.sf$coefficients[c(-1,-2)])

moran.test(sfilter, pr.listb)
geary.test(sfilter, pr.listb)

sf.res <- residuals(fp.sf, type="response")
moran.test(sf.res, pr.listb)
geary.test(sf.res, pr.listb)

mapping.seq(pr, sfilter, 5, main="SF")

#5.2.2.1

farm.den07 <- pr.f$nofarms_07/pr.f$area
y.fd <- (farm.den07 - 0.12)^0.38
shapiro.test(y.fd)
lm.fd <- lm(y.fd ~ rain)
lm.fd.s <- summary(lm.fd)
s2 <- round(lm.fd.s$sigma^2,5)
c1 <- round(0.5 * (-0.25+(1/0.38-
2+1.5)^2),5)
pred <- lm.fd$fitted
y.fd.e <- pred^(1/0.38) + c1*s2 + 0.12

lm.bt <- lm(farm.den07 ~ y.fd.e)
summary(lm.bt)

pois.fd <- glm(nofarms_07 ~ rain_mean,
offset=log(area), family=poisson,
data=pr.f)
pois.fd$deviance/pois.fd$df.residual
nb.fd <- glm.nb(nofarms_07 ~ rain_mean +
offset(log(area)), data=pr.f)
nb.fd$deviance/nb.fd$df.residual
1/nb.fd$theta
nb.fit <- nb.fd$fitted/pr.f$area
nb.back <- lm(farm.den07 ~ nb.fit)
summary(nb.back)
```

```
Get a spatial filter model in the text
estimated based on significance.

Summarize the result.

Conduct spatial autocorrelation tests.

Calculate deviance statistics for the
base and spatial filter models.

Get Pearson-type overdispersion value.


Load functions in all_functions.R file.
Map the farm percentage with 5 classes.


Map the predicted values.


Construct the spatial filter.



Conduct spatial autocorrelation test
for the spatial filter.

Get residuals of the spatial filter
model, and conduct spatial
autocorrelation tests.

Map the constructed spatial filter.



Calculate farm densities in 2007 and
transform it.
Conduct Shapiro-Wilk test.
Run a linear regression.
Store the summaries of the regression.
Calculate components for back-
transformation.


Calculate back-transformed predicted
values.
Run linear regression between observed
and predicted values, and summarize the
result.
Run a Poisson regression with offset
values.

Calculate deviance statistic.
Run a negative binomial model.

Calculate deviance statistic.
Get dispersion parameter estimate.
Calculate predicted densities.
Run linear regression and summarize it.
```

`par(mfrow=c(1,2))`
`plot(y.fd.e, farm.den07, pch=20)`
`abline(0,1, col=2)`
`nb.den <- fitted(nb.fd)/pr.f$area`
`plot(nb.den, farm.den07, pch=20)`
`abline(0,1, col=2)`
`par(mfrow=c(1,1))`
	Plot observed versus predicted plots from the normal model and negative binomial model.

`moran.test(farm.den07, pr.listb)`
`geary.test(farm.den07, pr.listb)`
`moran.test(farm.den07-y.fd.e, pr.listb)`
`geary.test(farm.den07-y.fd.e, pr.listb)`
`moran.test(farm.den07-`
`nb.fd$fitted/pr.f$area, pr.listb)`
`geary.test(farm.den07-`
`nb.fd$fitted/pr.f$area, pr.listb)`
	Conduct spatial autocorrelation tests for the three sets of values: the farm densities in 2007, residuals from the normal model, and residuals from the negative binomial model.

`lm.full <- lm(y.fd ~ rain + ., data=EV)`
`lm.sf <- stepwise.forward(lm.full, lm(y.fd ~ rain, data=EV), 0.1, verbose=F)`
`summary(lm.sf)$r.squared`
`pred.sf <- lm.sf$fitted`
	Conduct stepwise regression with *stepwise.forward*.
	Summarize the result.
	Get predicted values.

`s2.sf <- round(summary(lm.sf)$sigma^2,5)`
`y.e.sf <- pred.sf^(1/0.38) + c1*s2.sf + 0.12`
`lm.sf.bt <- lm(farm.den07 ~ y.e.sf)`
`summary(lm.sf.bt)`
	Conduct back-transformation.
	Examine its model fit.

`plot(y.e.sf, farm.den07, pch=20)` `abline(lm.sf.bt)`	Plot a scatterplot with observed and predicted values.

`lm.sf.res <- farm.den07 - y.e.sf`
`moran(lm.sf.res, pr.listb, n, Szero(pr.listb))`
`geary(lm.sf.res, pr.listb, n, n-1, Szero(pr.listb))`
	Get residuals of the normal model.
	Calculate Moran's *I*.
	Calculate Geary's *C*.

`X <- as.matrix(cbind(rep(1,n), lm.sf$model[,-1]))`
`num <- -n*sum(diag(solve(crossprod(X), crossprod(X,B)%*%X)))`
`den <- lm.sf$df.residual * sum(B)`
`num/den`
	Get independent variables to calculate the expected value of Moran's *I*: numerator
	denominator
	the expected value of Moran's *I*.

`nb.full <- glm.nb(pr.f$nofarms_07 ~ rain + offset(log(pr.f$area)) + ., data=EV)`
`nb.sf <- stepAIC(glm.nb(pr.f$nofarms_07 ~ rain + offset(log(pr.f$area)), data=EV), scope=list(upper=nb.full), direction="forward")`
`nb.sf <- glm.nb(pr.f$nofarms_07 ~ rain + EV12 + EV4 + EV1 + EV2 + EV18 + offset(log(pr.f$area)), data=EV)`
`summary(nb.sf)`
	Run stepwise negative binomial regression with *stepAIC*.
	Get a spatial filter model in the text estimated based on significance.
	Summarize the result.

`glm.sf.bt <- lm(farm.den07 ~ I(nb.sf$fitted/pr.f$area))` `summary(glm.sf.bt)`	Examine the model fit of the negative binomial model.
`plot(nb.sf$fitted/pr.f$area, farm.den07, pch=20)` `abline(glm.sf.bt)`	Create a scatterplot of observed versus predicted values.

```
glm.sf.res <- farm.den07 -
nb.sf$fitted/pr.f$area
moran(glm.sf.res, pr.listb, n,
Szero(pr.listb))
geary(glm.sf.res, pr.listb, n, n-1,
Szero(pr.listb))

X <- as.matrix(cbind(rep(1,n),
nb.sf$model[,c(-1,-8)]))
num <- -n*sum(diag(solve(crossprod(X),
crossprod(X,B)%*%X)))
den <- nb.sf$df.residual * sum(B)
num/den

ur <- factor(pr.f$u_r, levels=0:1,
labels=c("urban", "rural"))
tapply(y.fd, ur, sw.p)
leveneTest(y.fd, ur, center=mean)
anova(lm(y.fd ~ ur))

ur.d <- ifelse(pr.f$u_r==0,-1,1)
lm.full <- lm(y.fd ~ rain+ur.d+., data=EV)
lm.ur <- stepwise.forward(lm.full, lm(y.fd
~ rain + ur.d, data=EV), 0.1, verbose=F)
lm.ur.res <- residuals(lm.ur)
tapply(lm.ur.res, ur, sw.p)
leveneTest(lm.ur.res, ur, center=mean)
summary(lm.ur)$coefficients[3,]

nb.ur <- update(nb.sf, . ~ . + EV10 + EV15
+ ur.d)
summary(nb.ur)$coefficients[10,]
nb.ur$deviance/nb.ur$df.residual


#5.2.2.2

pr.j2w <- read.csv("PR_journey-to-
work_2000.csv")
n <- sqrt(NROW(pr.j2w))

# model1
f.os <- function(x,flow.df,n){
sum(flow.df[flow.df[,"ResID"]==x,"Count"])}
Oi.sum <- sapply(1:n, f.os, flow.df=pr.j2w,
n=n)
Oi.sum <- rep(Oi.sum, each=n)

f.ds <- function(x,flow.df,n){
sum(flow.df[flow.df[,"WorkID"]==x,"Count"])
}
Dj.sum <- sapply(1:n, f.ds, flow.df=pr.j2w,
n=n)
Dj.sum <- rep(Dj.sum, n)
lnOiDj <- log(Oi.sum) + log(Dj.sum)

si.nc <- glm(Count ~ dist, offset=lnOiDj,
data=pr.j2w, family=poisson)
exp(si.nc$coefficients[1])
si.nc$coefficients[2]
si.nc$deviance/si.nc$df.res
lm.nc <- lm(pr.j2w$Count~si.nc$fitted)
```

```
Get the residuals of the negative
binomial model.
Calculate Moran's I.

Calculate Geary's C.


Get independent variables to calculate
the expected value of Moran's I.
numerator

denominator
the expected value of Moran's I.

Create a factor variable for urban &
rural.
Conduct normality tests.
Conduct Levene's test.
Conduct ANOVA.

Create a dummy variable with -1 & 1.
Run stepwise regression for a spatial
filter model.

Get residuals of the spatial filter
model, and then conduct normality and
Levene's tests.
Get statistics for ur.d variable (mean
difference test).
Get a spatial filter model in the text
estimated based on significance.
Get statistics of ur.d variable.
Get deviance statistic.



Load journey-to-work data.

The number of regions (i.e.,
municipalities).

Define a function to calculate sums of
flows from each origin.
Get sums of flows from each origin.

Match the sums to the
origin/destination (OD) list.
Define a function to calculate sums of
flows from each origin.

Get sums of flows from each
destination.
Match the sums to the OD list.
Prepare an offset variable.

Run Poisson regression with only
distance variable.
Constant estimate.
Distance-decay estimate.
Deviance type overdispersion estimate.
Examine the model fit.
```

```
summary(lm.nc)

# model2
b.id <- 63                                  Set a base level for dummy variables.
fid.f <- as.factor(pr.j2w$ResID)            Get a factor variable for origins.
contr.f <- contr.treatment(levels(fid.f),   Create dummy variables for origins with
base=b.id)                                  b.id region as base.
xo <- contr.f[fid.f,]                       Match the dummy variable to the OD list.
colnames(xo) <- paste("R", levels(fid.f)[-  Set column names for the dummy
b.id], sep="")                              variables.
rownames(xo) <- 1:(n^2)                     Set row names.

tid.f <- as.factor(pr.j2w$WorkID)           Similarly create dummy variables for
contr.t <- contr.treatment(levels(tid.f),   destinations.
base=b.id)
xd <- contr.t[tid.f,]
colnames(xd) <- paste("W", levels(tid.f)[-
b.id], sep="")
rownames(xd) <- 1:(n^2)

si.dc <- glm(Count ~ dist + xo + xd,        Run a Poisson regression with the dummy
offset=lnOiDj, data=pr.j2w, family=poisson) variables and distance.
exp(si.dc$coefficients[1])                  Constant estimate.
si.dc$coefficients[2]                       Distance-decay estimate.
si.dc$deviance/si.dc$df.res                 Deviance type overdispersion estimate.
lm.dc <- lm(pr.j2w$Count~si.dc$fitted)      Examine the model fit.
summary(lm.dc)                              Summarize the result.


# model3
attach(pr.j2w)                              Add pr.j2w to a search space.
# eigenvector treatment for flows
evec <- read.table("pr_evecs.txt",          Read eigenvectors from the transformed
header=T)                                   spatial weight matrix.
evec <- evec[,c(-1,-2)]                     Drop two ID columns.
EV <- evec[,1:11]                           Select the first 11 eigenvectors.
EVo <- apply(EV,2, function(x,n)            Match the 11 eigenvectors to the
{rep(x,each=n)}, n=n)                       origins in the OD list.
EVd <- apply(EV,2, function(x,n)            Match the 11 eigenvectors to the
{rep(x,n)}, n=n)                            destinations in the OD list.
EVod <- kronecker(EVo, matrix(1,1,11)) *    Generate 121 eigenvectors by
kronecker(matrix(1,1,11),EVd) * 100         multiplying the matched eigenvectors
                                            for origins and destinations.
colnames(EVod) <- paste("EV",1:121,sep="")  Set column names.
EVod.df <- as.data.frame(EVod)              Convert a matrix to a data frame.

#disp <- si.dc$deviance/si.dc$df.res        Conduct stepwise regression to
#si.full <- glm(Count ~ dist + xo + xd + .,  construct a spatial filter model. Note
data=EVod, family=poisson)                  that this stepAIC function will take a
#si.sf <- stepAIC(glm(Count ~ dist + xo +    while. Also note that the selected
xd, data=EVod, family=poisson), scale=disp, eigenvectors with significance in the
scope=list(upper=si.full),                  text are stored in pr_flow_sel_evec.txt
direction="forward", trace=0)               file.

evs <- scan("pr_flow_sel_evecs.txt")        Read selected eigenvector information.
EVod.sel <- EVod[,evs]                      Get only the selected eigenvectors.

si.sf <- glm(Count ~ dist + xo + xd +       Run a Poisson with distance, dummy
EVod.sel, offset=lnOiDj, family=poisson)    variables, and the selected
                                            eigenvectors.
exp(si.sf$coefficients[1])                  Constant estimate.
si.sf$coefficients[2]                       Distance-decay estimate.
si.sf$deviance/si.sf$df.res                 Deviance type overdispersion estimate.
```

```
lm.sf <- lm(Count~si.sf$fitted)                    Examine the model fit.
summary(lm.sf)                                     Summarize the result.

plot(si.nc$fitted, Count, pch=4)                   Scatterplots of observed versus
points(Count, Count)                               predicted values for the three models.
plot(si.dc$fitted, Count, pch=4)
points(Count, Count)
plot(si.sf$fitted, Count, pch=4)
points(Count, Count)

detach(pr.j2w)                                     Remove pr.j2w from the search space.

ai.v <-substr(names(si.sf$coef),1,3)=="xoR"        Find origin dummy variables.
ai <- si.sf$coef[ai.v]                             Get estimated coefficients of origin
                                                   dummy variables.
bj.v <-substr(names(si.sf$coef),1,3)=="xdW"        Find destination dummy variables.
bj <- si.sf$coef[bj.v]                             Get estimated coefficients for
                                                   destination dummy variables.
insert <- function(v,e,pos){                       Create a function to insert zero for
  return(c(v[1:(pos-1)], e,                        base regions of the dummy variables.
  v[(pos):length(v)]))}
ai <- insert(ai, 0, b.id)                          Put zero for the origin base region.
bj <- insert(bj, 0, b.id)                          Put zero for the destination base
                                                   region.
ev.v <- substr(names(si.sf$ coef),1,8) ==          Get estimated coefficients for
"EVod.sel"                                         eigenvectors.
ev.beta <- si.sf$coef[ev.v]
sf.if <- EVod.sel %*% ev.beta                      Calculate a spatial filter.
sf.df <- pr.j2w[,c("ResID", "WorkID")]             Combine the filter values with origin
sf.df$sfij <- sf.if                                and destination IDs.

f.oi <- function(x,flow.df) { median(              Define a function to get medians of the
flow.df[flow.df[,"ResID"]==x,"sfij"])}             spatial filter vales for each origin.
sf.oi <- sapply(1:n, f.oi, flow.df=sf.df)          Get medians for origins.

f.dj <- function(x,flow.df) {median(               Similarly, get medians for
flow.df[flow.df[,"WorkID"]==x,"sfij"])}            destinations.
sf.dj <- sapply(1:n, f.dj, flow.df=sf.df)

mapping.seq(pr,ai,7,main="Org Balancing")          Map the balancing factors and spatial
mapping.seq(pr,bj,7,main="Dest Balancing")         filters.
mapping.seq(pr,sf.oi,7,main="Org SF")
```