ASSIGNMENT 6

This final study is a continuation of the analysis in Assignment 5, and will use the results of that analysis. It is assumed that you have constructed the shapefile, **Phil_Tracts_Final.shp**, in ARCMAP, and also the MATLAB workspace **Phil_Housing.mat**. Here you will carry out spatial regressions that take account of the *spatial autocorrelation* found in the OLS residuals when regressing the log of Median Housing Value (**lnMV**) on the log of Median Income (**lnMI**) and Percent NonWhite (**%NW**). First open ARCMAP, and create a blank map document. Now add **Phil_Tracts_Final.shp** to this document and, for convenience, re-label this layer as **Philadelphia Housing**. Save the new document as **Phil_Housing_6** (for Assignment 6).

(a) In this initial phase of the study you will construct a new spatial weight matrix for Philadelphia based on *boundary shares*. This type of weight matrix is more appropriate for areal data such as contiguous census tracts. The procedure for doing so is outlined in the CLASS NOTEBOOK on the class web page. Scroll down to part IV and open section **3.2.3** in **Using Matlab**, called "Making Boundary-Share Weight Matrices using ARCMAP and MATLAB". Here a detailed procedure is outlined using the **Eire** data set illustrated in class.

1. In using this procedure, you can start here at step (1.2) by constructing a second copy of the shapefile, **Phil_Tracts_Final**, say **Phil_Tracts_Final_0**. Now continue the procedure up to Step (1.5).

2. In Step (1.5) you will obtain a file, **Phil_Tracts_Final_Intersect**. (This may take some time to complete.) If you open the Attribute Table as in step (1.6) and look at the bottom of the window, you should see that there are **3478** polyline features, representing shared boundary segments. Working with feature files this size can be slow – so be patient!

3. Finally, the new field, **Arc_Length**, calculated in Step (1.7) gives you the lengths of all shared boundaries. To gain some feeling for how this works, select the **ID** column just to the left of **Arc_Length** (perhaps labeled **ID_1**), that contains the row IDs of one of the tracts sharing the boundary segment in that row. Right click on this column header and click **Sort Ascending** to put these values in ascending order. So if you now scroll down to **ID** = 282, you should see seven rows with 282. Select the first row (**Arc_Length** = 9471.09), find the selected segment on the map, and enlarge that area. You should now see that this census tract (FID = 281) appears to be a "peninsula" defined by two missing tracts (with zero population) connected by the Schuylkill River. Notice in particular that this boundary-share procedure excludes connections between this census tract and all others on the east side of the river. However, if this map had been drawn more crudely (or if the river had been very narrow) so that the river was a *single curve*, then tracts on the east side would now share boundaries with this tract. Since these conventions are often arbitrary, you should be aware that boundary shares can sometimes yield strange results. Hence you should always *look at the map* to be sure that these results are reasonable. Note also that the remaining six rows come in pairs, and refer to the segments shared with three other tracts (FID = 266,270,279). So,

boundaries shared with no other tract appear once (with both **ID** values the same) and those shared with other tracts appear twice. [This distinction is used in the MATLAB program, **shared_bd_lengths.m**, below to identify those boundary segments shared with other tracts.]

4. Now carry out the steps up to (2.1) where you should obtain an EXCEL file, **shared_bd_length.xls**, containing only columns, (**ID**,**ID**,**Shape_Length**). In step (2.2) when you import these results to MATLAB, you should open the workspace **Phil_Housing.mat** created in Assignment 5, and import **shared_bd_length.xls** to this workspace as a numerical matrix, **DAT**.

5. Use **DAT**, and construct **M** as in step (2.2) with the program, **shared_bd_lengths.m**. As a consistency check, now type the command

   » M(282,:) '

   and you should see the output

   (267,1) 1884.6
   (271,1) 627.8
   (280,1) 2032.3

   listing the lengths of boundary segments shared with tract 282 (FID = 281). [The reason for this simple output format is that the matrix **M** is in *sparse* form (as can be seen by both the matrix icon and class label in the workspace). Also note the transpose at the end, which converts rows to columns.]

6. Finally, you should row-normalize this matrix with the command

   » W0 = row_norm(M);

   This will avoid overwriting the four-nearest-neighbors weight matrix, **W**, constructed in Assignment 5. Check the workspace to be sure that **W0** is a 351x351 matrix.

   Note: For *boundary shares* weight matrices such as **W0** this *row normalization* has a clearer interpretation than in most cases. Here one measures the "influence" of tract **j** on tract **i** in terms of the fraction of total shared-boundary for tract **i** that is shared with tract **j**. Notice also that unlike most spatial weights, this measure is fundamentally *asymmetri*c. In particular if total shared boundary of tract **j** is relatively large compared to that of tract **i**, then the "influence" of **i** on **j** may well be smaller than that of **j** on **i**.

7. Since **W0** is also seen to be in sparse form. we can now represent row 282 of **W0** by writing:

» W0(282,:) '

to obtain the output:

(267,1) 0.41468
(271,1) 0.13814
(280,1) 0.44718

This is seen to be a normalization of the lengths above, which now add to one. [It is worth noting here that if a given matrix, **A**, is not sparse, you can make a *sparse copy*, **AA**, by writing

» AA = sparse(A);

This will allow you to examine the rows (or columns) of **A** in the above format by using **AA**.]

8. Finally, as a second illustration that these weights need not always be the most "reasonable" ones, write:

   » W0(351,:) '

   Check this result in ARCMAP and comment on your findings.

(b) Given this boundary-share weight matrix, **W0**, it is now appropriate to recheck for the presence of spatial autocorrelation in the OLS residuals by using **W0**. Rather than regressing on nearest-neighbor residuals, you will here use the more flexible approach based on the *random permutation tests* in the MATLAB program **sac_perm**. This program allows *any* weight matrix to be used, and also reports three different indices (**I**, **rho**, **r**). The first, (Moran's) **I**, is the most commonly used measure of spatial autocorrelation. The second, **rho**, is very similar to the JMP regression of residuals, **res**, on their weighted neighbors, **W0*res**, and gives essentially the same result.

1. In the workspace you should already have the vector of OLS residuals, **res**, constructed in Assignment 5.

2. To perform the random-permutation test with 999 random permutations, write:

   » sac_perm(res,W0,999);

   You will only need the Screen output, and not the Data output.

3. Given this result, comment on the presence of spatial autocorrelation in these OLS residuals. Note in particular how the values of these indices related to the interval of simulated values represented by the MAX and MIN values in the screen output.

(c) The first spatial regression you will perform will use the **spatial errors model** (**SEM**), in the MATLAB program, **sem.m**. To do so, open this program (**» edit sem**) and review its input requirements.

1. To construct the regression data, recall from **Phil_Table** that data for the dependent variable, **lnMV**, is in the 1st column of **Phil_matrix**, and data for the explanatory variables, **lnMI** and **%NW**, is in the 2nd and 3rd columns, respectively. So you can construct the data inputs, **y** and **X**, by writing:

   » y = Phil_matrix(:,1);
   » X = Phil_matrix(:,2:3);

   Before proceeding, display the first few rows of **y** and **X** and compare with JMP to be sure that you have selected the right columns. (If you mistakenly use **MV** rather than **lnMV** your results will be *very* badly behaved.)

2. To make the list of variable names, **vnames**, write:

   » vnames = char(' lnMI ' , ' %NW ')

   By leaving off the semicolon, the vector of names should now be displayed on the screen.

3. Given the size of the weight matrix, **W0**, it is worthwhile calculating the eigenvalues of **W0** before hand, and using these as inputs to the program. To do so, write:

   » eigvals = real(eig(full(W0)));

   Here the function **full** converts the (default) sparse representation of **W0** into a full 351-square matrix, and the function **eig** then computes its eigenvalues. Finally, **real** removes any (small) imaginary components that may have crept in during the root extraction process.

4. To run the **spatial errors model** write: (Note that there are three output matrices in this program. But we are only interested in the first one, which can be obtained by specifying a single output argument.)

   » OUT_sem = sem(y,X,W0,vnames,eigvals);

   Copy-and-paste the screen output to a WORD file for later comparison with the spatial-lag model.

5. Next, noting from the list of **sem.m** outputs that the SEM-residuals and OLS-residuals are in the 3rd and 5th columns of **OUT_sem**, respectively, write:

» res_OLS = OUT_sem(:,5);
» res_SEM = OUT_sem(:,3);

Observe that **res_OLS** should be essentially the same as **res** (from JMP). A simple way to verify this is to write:

» max(abs(res_OLS – res))

This will display on the screen the maximum absolute difference between the components of these two residual vectors. [This will not be zero, but should be very small (around 10^{-6}). If not, check to be sure you have defined **res_OLS** correctly.]

6. Next, use **sac_perm** on the SEM residuals:

   » sac_perm(res_SEM,W0,999);

   A comparison between these significance values and those obtained above for the OLS residuals shows that SEM has effectively removed the positive spatial autocorrelation present in the simple OLS analysis. This is consistent with the significantly positive value of **rho** in the SEM screen display. But there appears to be some "over-correction" here, in the sense that p-values > 0.95 indicate that there is now significant *negative* autocorrelation in these SEM residuals. We shall return to this point below.

(d) Next, you will carry out a second spatial regression using the **spatial lag model (SLM)**, using the MATLAB program, **slm.m**. As before, open this program (**» edit slm**) and review its input requirements.

1. The data inputs, **y** and **X**, are the same for this model. Also, the weight matrix, **W0**, variable names vector, **vnames**, and eigenvalue vector, **eigvals**, are the same.

2. So to run this program, write:

   » OUT_slm = slm(y,X,W0,vnames,eigvals);

   Copy-paste the screen output to the WORD file created above, so that the two sets of outputs can be compared side-by-side.

3. Noting from the list of **slm.m** outputs that the SLM residuals are again in the 3rd column of **OUT_slm**, save these as:

   » res_SLM = OUT_slm(:,3);

4. As before, use **sac_perm** on these SLM residuals:

   » sac_perm(res_SLM,W0,999);

Compare these significance values with those obtained above for the SEM residuals, and comment on your findings. In particular, state whether or not these results appear to be consistent with the test results for the **Common Factor Hypothesis** above.

5. How does the significance test for **rho** (in the AUTOCORRELATION RESULTS") compare with that for **rho** in the SEM model? In particular, how might the smaller estimated value of **rho** in SLM relate to the notion of "over-correction" suggested for SEM?

(e) Given these two sets of regression results, compare the "GOODNESS-OF-FIT" measures for each. Focus in particular on the **Extended R-Squared value**, which is in many ways the most meaningful of these.

(f) You will now compare the residuals for these models **spatially** to see whether the relative **patterns** of spatial residuals agree with the statistical results above. To do so:

1. First combine the three sets of residuals into a common matrix:

   » RES = [res_OLS,res_SEM,res_SLM];

2. Next, to allow this data to be merged with ARCMAP data, add a copy of the appropriate **FID** column by writing:

   » RES(:,4) = [0:352] ' ;

   Remember that **FID** starts with '0' rather than '1'.

3. Now, export this file to your home directory with a command similar to:

   » save 'e:\Home\Residuals.txt' RES -ascii

4. As usual, open this file in EXCEL, choose an appropriate numerical format [as in part 10.(a) of Assignment 5] and add a title row with column labels (**res_OLS**,**res_SLM**,**res_SLM**,**ID**).

5. Save as a **tab-delimited** text file, and rename this file as **Residuals.tab**.

6. Now open your map document, **Phil_Housing.mxd**, in ARCMAP, and add **Residual.tab** to the data frame "Philadelphia Tract Data (1990)".

7. Merge this data with the attribute file for "Median Housing Values" (**Joins and Relates → Join**) using **FID** and the constructed column **ID** as the common attributes for the merge.

8. Save as a new shape file (**Data → Export Data**) named **Resid_Data.shp** and add to the Table of Contents.

9. Rename the layer as "OLS Residuals", and display these as follows:

   (i) First, select **res_OLS** as the variable in the **Symbology** window.

   (ii) Now, open the **Classify** window and set the **Classification Method** = "Standard Deviation". [This is the best way to make all sets of residuals comparable.] Click **OK** and return to the **Symbology** window.

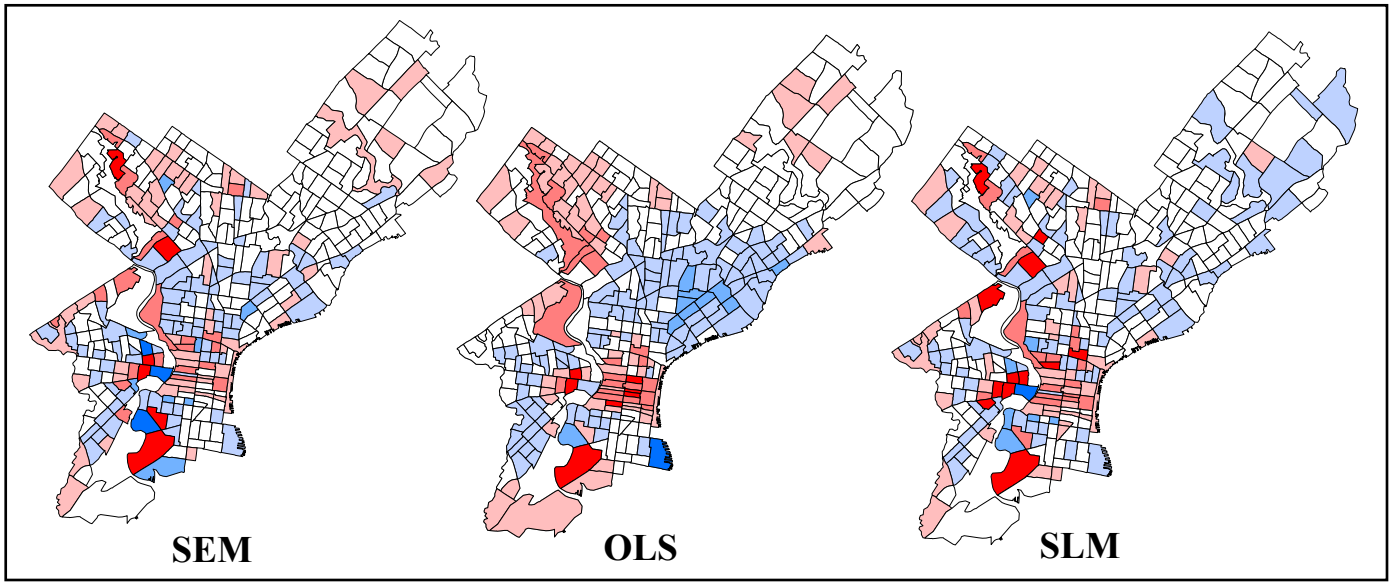
   (iii) To set the colors manually, right click on the **mid range** (-0.50 – 0.50 Std Dev) to bring up the color pallet, and set the color "white" (upper left hand corner). Use increasing shades of "Blue" for the **negative** ranges and "Red" for the **positive** ranges. [A good choice is the first three "Reds" in the second column of the pallet, and the first three "Blues" in the tenth column.] These colors are chosen in order to facilitate visual comparison of the three sets of residuals. However, if you plan to print these in **black-and-white**, then it would be better to use a single gray scale, say running from light to dark as values increase.

10. Next, use **Copy** and **Paste Layer** to add two more copies of the "OLS Residuals" layer.

    (i) Name them as the "SEM Residuals" and "SLM Residuals".

    (ii) Repeat Step 9 for each of these layers, using the respective variables, **res_SEM** and **res_SLM**. (Don't worry if the number of "Classes" differs for each. Classes are assigned automatically according to how many Standard Deviation classes are present in the data distribution.)

11. Using **Edit → Copy Map to Clipboard**, copy all three sets of residuals into WORD, with **OLS Residuals** in the middle, and compare them visually. If you have done all of this correctly, then you should now have a display like the one below:



12. Do these different patterns agree or disagree with your statistical findings? Be explicit.

(g) Finally, consider the **regression results** in all three cases.

1. How do the **beta coefficients** compare between the models? Are there any notable differences in sign or magnitude? If so, how might you account for this?

2. What about differences in the significance levels (**P-values**) of the coefficients? Again, how might you account for this?

3. If you had to pick one of these models as "best", which would you choose and why?

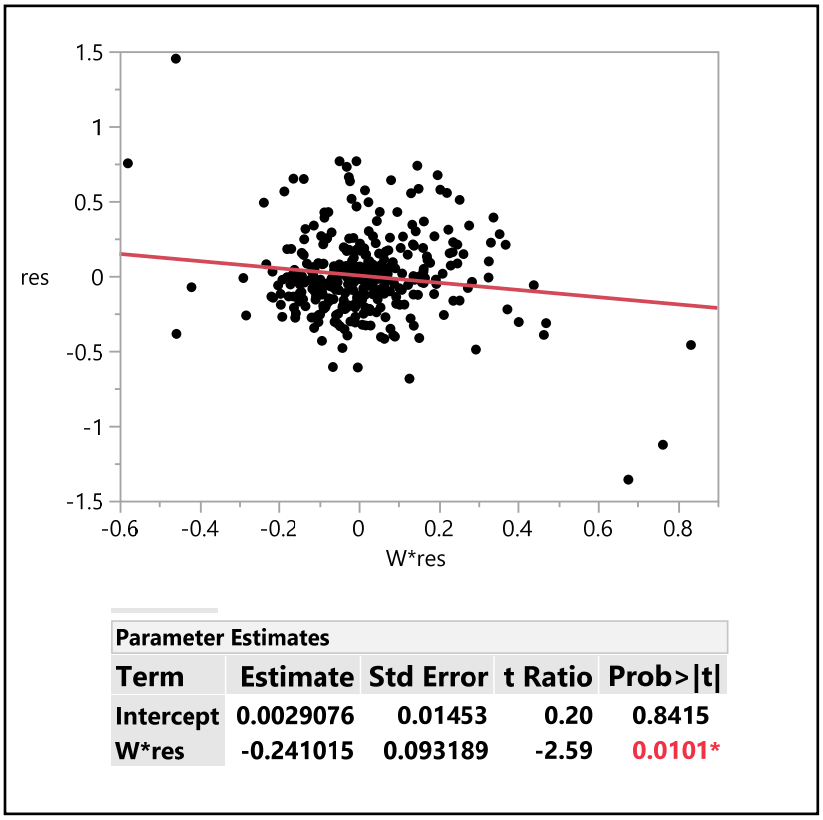
EXTRA CREDIT: A Deeper Analysis of Model Differences

This final section of the assignment is *optional*, and is designed to give you a deeper look into the differences between these model results. As we have seen before in terms of results for alternative choices of weight matrices, different results for models using the *same* weight matrix can often lead to additional insights about the behavior of these models. In the present case, we focus on (i) the apparent "over-correction" of positive spatial autocorrelation by the SEM model that resulted in negative autocorrelation of the final residuals, and (ii) the apparent ability of SLM to avoid this problem.

(i) While the residual maps above show that both SEM and SLM exhibit less spatial autocorrelation than OLS, the differences between SEM and SLM are far less obvious. Moreover, while the **sac_perm test** shows that there is indeed a dramatic difference, the test itself does not tell us *why* this is happening. Here it turns out that the simpler heuristic tool of regressing residuals on neighboring residuals is actually quite informative. To use this tool in the present case, first recall from the motivation for the *rho statistic* in Section 4.1.1 of the NOTEBOOK that the spatial autocorrelation model,

(1) $u = \rho W u + \varepsilon$

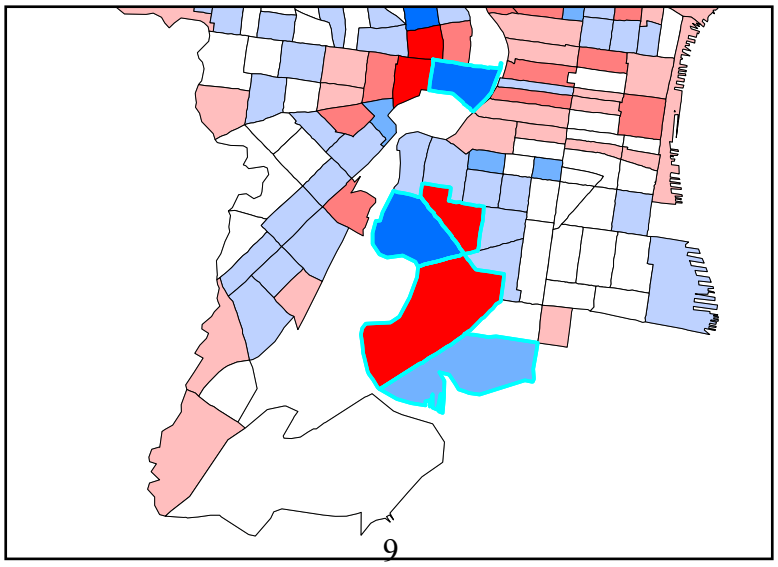
itself suggests that the slope of a regression of estimated residuals, $\hat{u}$, on weighted residual, $W\hat{u}$, should yield a natural estimate of spatial autocorrelation. Such simple regressions are easily plotted, and are often very informative visually. To do so in the present setting, observe from the definition of the SEM output matrix, **OUT_sem**, above that the third and fourth columns contain precisely this data for SEM residuals. Thus, by exporting **OUT_sem** to JMP and keeping only columns 3 and 4, say with labels, **res** and **W*res**, one can perform a "Fit Y by X" regression. For simplicity, you can ignore the no-intercept option, since we have already seen that the plots are virtually identical. [Note also that while one could also construct a regression plot using SLM for comparison, the results are somewhat more difficult to interpret since the autoregressive model in (1) is not applied directly to residuals, but rather to the dependent variable.] If you have carried out this procedure correctly, you should obtain regression results corresponding to those shown below:



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	0.0029076	0.01453	0.20	0.8415
W*res	-0.241015	0.093189	-2.59	0.0101*

As expected, we obtain a significantly negative slope, showing (as we have seen many times before) that the results of this heuristic tool are consistent with the more rigorous results of **sac_perm**. But more importantly in the present case, we see that this negative slope is determined almost entirely by the two outliers in the upper left corner together with the three outliers in the lower right corner. Moreover, by placing the mouse on each of these outliers, you can directly identify the corresponding rows in the data matrix, (324 , 340) in the upper left and (326, 348, 282) in the lower right. Next, recalling that FIDs start with zero, these correspond to FIDs (323, 339, 325, 347, 281) in ARCMAP. If you select these FIDs in ARCMAP with the SEM residual map open, the corresponding selected tracts should appear as shown below:



Notice in particular that four of the five are *adjacent*. Even more important is the "checkerboard" nature of their signs, i.e., all adjacent pairs have residuals of *opposite signs*. What does this tell you about the *negative spatial autocorrelation* exhibited by SEM residuals?

Now take a closer look at these census tracts by using the "Identify" icon. Here we focus on the most prominent middle pair of tracts with FIDs 325 and 339. Notice first from the selected attributes shown in Table 1 below that these tracts represent extreme examples of *racial segregation* in terms of **%NW**. (Given the importance of such segregation, this is already a sufficient reason for not deleting these "outliers".)

	FID 325	**FID 339**
MEDIANVAL	30,300	125,000
PCT_NONWT	0	97.11
OLS_res	-0.77	2.46
SEM_res	-1.23	1.45
SLM_res	-0.58	1.27

Table 1. Selected Attributes of Tracts 325 and 339

Observe next that housing values in the "white" tract (FID) 325 tend to be much lower than in the "black" tract 339. Given the SEM regression results for **%NW**, how does this help to account for the magnitudes and signs of the SEM residual for tracts 325 and 339?

(ii) Turning next to the SLM model, notice first that this same sign pattern is exhibited by the SLM residuals (and indeed by the OLS residuals as well). So, while the residuals for SLM are smaller in magnitude than those for SEM, there is certainly more going on here than can be explained by these numbers alone. A second key factor is suggested by the formal difference between the models themselves. Recall from expressions (6.1.8) and 6.2.6) in Part III of the NOTEBOOK that the SEM model and SLM model (as instances of General Linear Models) have the respective forms:

(2) $Y = X\beta + \varepsilon \ , \ \varepsilon \sim N(0, \sigma^2 V_\rho)$

(3) $Y = X_\rho \beta + \varepsilon \ , \ \varepsilon \sim N(0, \sigma^2 V_\rho)$

where

(4) $X_\rho = (I - \rho W)^{-1} X = X + \rho WX + \rho^2 W^2 X + \cdots$

In these terms, it is clear that the only formal difference between these models is the *spatially-transformed* regression attributes, X_ρ, in SLM. Recall moreover from our interpretation of the "Rippled Pale" in Eire example at the end of Section 7 in Part III of the NOTEBOOK that these

transformed attributes can have profound consequences for the relative fits of these two models. In the present case, it is enough to observe that this transformation tends to "smooth" the relative values of attributes across space by incorporating successively smaller contributions from all other locations. What this implies in the present case is that extreme values of variables such as **%NW** tends to be smoothed toward the mean values across all locations. To see this in the present case, we focus on the attribute, **%NW**. Start by letting **NW_sem** denote the column vector of **%NW** values in the data matrix, **X**, i.e.,

» NW_sem = X(:,2) ;

For illustration we shall use FID 339, which corresponds to row 340 in **X**. As a check, the value of

» NW_sem(340)

should be **97.114**. Next, the mean value

» mean(NW_sem)

of percent non-whites should be **45.59**, so that **NW_sem(340)** is seen to more than twice the mean percent non-white across all tracts. For purposes of comparison, a better value to use is the standardized value,

» sem_dev_340 = (NW_sem(340) - mean(NW_sem))/std(NW_sem)

which in this case should be **1.3048**. In other words, the value **NW_sem(340)** is more than 1.3 standard deviations above the mean.

Next you will construct the spatially-transformed version of this attribute under model SLM. Here we start with the estimated value of **rho** under SLM,

» rho = 0.765;

and construct the 351-square identity matrix,

» I = eye(351);

Then by (4) above, the spatially-transformed attribute matrix for SLM is given by

» X_rho = inv(I - rho*W0)*X;

In these terms, the new vector of spatially-transformed values of **NW_sem** is given by

» NW_slm = X_rho(:,2) ;

In particular, the value

» NW_slm(340)

should be **216.45**. Finally, the corresponding standardized value is then given by

» slm_dev_340 = (NW_slm(340) - mean(NW_slm))/std(NW_slm)

If you have done this calculation correctly, the new deviation, **slm_dev_340**, should be only about 13% as large as **sem_dev_340**, showing that this spatially-transformed value is far less extreme that the original value of percent non-white. More generally, this spatial transformation does indeed tend to reduce the effects of outliers.

Given these findings, what can you conclude about the use of SLM versus SEM in the present analysis of Philadelphia housing values?