ArcSOM: A new ArcToolbox for Visualization of Self-Organizing Maps

Charles R. Schmidt
Regional Analysis Laboratory
Department of Geography
San Diego State University

Martin Lacayo-Emery
Department of Geography
San Diego State University

December 11th, 2006

1 Introduction

The Self-Organizing Map (SOM) as introduced by Kohonen (2000) is offered as a compelling method for data reduction and visualization. The SOM algorithm uses an artificial neural network to map high dimensional input signal onto a 2D space. SOM_PAK is a package of command line utilities for the creation and visualization of SOMs (Kohonen et al., 1996). The visualizations produced by SOM_PAK are simply encapsulated postscript files which offer no interactive ability. In order to leverage GIS tools to explore and visual the SOM many intermediate steps must be under taken. As shown in figure 2.2.3 Skupin (2006) provides an overview of the steps required[1]. This process is long, cumbersome and enhances the possibility of user error. The goal at the start of this project was to shorten the distance between SOM and GIS in order to create a user friendly environment for the exploration of SOM.

To that end we have created an ArcToolbox with several tasks in mind. The first task was to take a trained SOM, represented in the standard codebook (.cod) file format, and create of shapefile of its topology. The second task was to allow the visualization of the SOM's component planes in ArcMap. The final task was to map data back onto the trained SOM for visualization in ArcMap.

2 The Tools

The tools we created for this project are divided into two groups, those dealing with output from SOM_PAK and compatible 2D Euclidian SOMs and those dealing with new Spherical SOMs. The tools are largely the same and the major differences lie in the drawing of the neuronal lattice.

[1]It should be noted that 'CreateSOMNeuronGeometry.exe' was produced by one of Skupin's students and is not widely available.

2.1 SOM_PAK Tools

SOM_PAK is capable of producing 2D SOMs using either a rectangular or hexagonal topology. The .cod file format introduced by Kohonen et al. (1996) is a simple ASCII file. The first line contains the header which describes the topology type, grid size, dimensionality and other training parameters. The first tool creates a new shapefile containing the polygon representation of the codebook's topology which each dimension listed as a field in the associated dbf.

2.1.1 Visualize

Lacayo developed a process to draw the appropriate topology for a given .cod file. The script first reads the header in order to determine which type of topology to draw. Next a text file is creating with the exact coordinates of each neuron's polygon. This text file is passed on to the geoprocessor in order to create a shapefile, then a slightly reformatted versions of the .cod file can then be joined back to the shapefile.[2]

2.1.2 Map Data

The second tool maps data back onto a trained codebook file. This is done by located an input signal's closest match in n dimensional Euclidean distance. Each observation's id and the centroid of the associated best matching unit are recorded into a new shapefile.

2.1.3 Limitations

- Using C:\tmp instead of a user specified workspace for storing intermediate files.

2.2 Spherical

Schmidt is currently developing a spherical SOM, it use the basic Kohonen algorithm but implements it over a spherical topology. The output is in the form of a standard .cod file and an associated .geo file that describes the point location of each neuron on the sphere in terms of longitude and latitude. The spatial distribution of the points is irregular and the Voronoi diagram is used for visualization.

2.2.1 Visualize

This tool relies heavily on the geoprocessor. The geo file is first brought in as an XY event layer which is passed onto the Voronoi tool via the geoprocessor. The resulting raster is converted back into a polygon shapefile and joined with the codebook data.

The Voronoi tool available only works in 2D Euclidean space, as such the resulting diagram is inaccurate nears the poles and dateline. A spherical Voronoi algorithm is available

[2]For another class project (Geog780: Geovisualization and Geocompuation) Lacayo developed a method in which he manually creates the shapefile, including the associated dbf. This method was used in the final toolbox on the grounds of efficiency and reliability.

1. rename codebook to file.cod
2. put file.cod and CreateSOMNeuronGeometry.exe in same directory
3. run CreateSOMNeuronGeometry.exe for output:
4. convert NeuronPolys.txt to coverage using ArcInfo Generate tool
5. convert NeuronCentroids.txt to coverage using ArcInfo Generate tool
6. join attributes to polygons
7. join attributes to centroids
8. run visual.exe on codebook for x,y index for input data
9. join index with NeuronIDstoSOMPAKxy.txt
10. join index with neuron centroids given in NeuronCentroids.txt
11. join data using joined NeuronIDstoSOMPAKxy.txt

Figure 1: Steps for SOM visualization in GIS

in FORTRAN and attempts were made to incorporate this into the toolbox. The FORTRAN code outputs line segments in XYZ coordinates which are converted to LAT LONG and then split at the dateline for compatibility with ArcGIS. Both ArcMap and ArcGlobe have difficulty displaying the line segments near the poles, however after the split, the ones that cross the dateline are fine. Attempts to create polygons from these line segments was almost entirely unsuccessful, areas near the poles and dateline were completely butchered.

It was discovered that learning enough FORTRAN to modify the Voronoi code so that it would report the polygons was a simpler process then trying to get ArcMap to work with a Sphere. The issue of splitting those polygons at the poles and dateline for ESRI compatibility is a another issue, one that will be assigned to the category of “Future Work”.

2.2.2 Map Data

The same process is used as above, only the handling of topology is different here.

2.2.3 Limitations

- Results and intermediate data are all stored in C:\tmp
- Voronoi Regions inaccurate near poles and dateline.

References

Kohonen, T. (2000). *Self-Organizing Maps*. Springer, 3 edition.

Kohonen, T., Hynninen, J., Kangas, J., and Laaksonen, J. (1996). Som_pak: The self-organizing map program package. Technical Report A31, Helsinki University of Technology, Laboratory of Computer and Information Science, FIN-02150 Espoo, Finland.

Skupin, A. (2006). Som_pak: Breif overview. Lecture Notes.