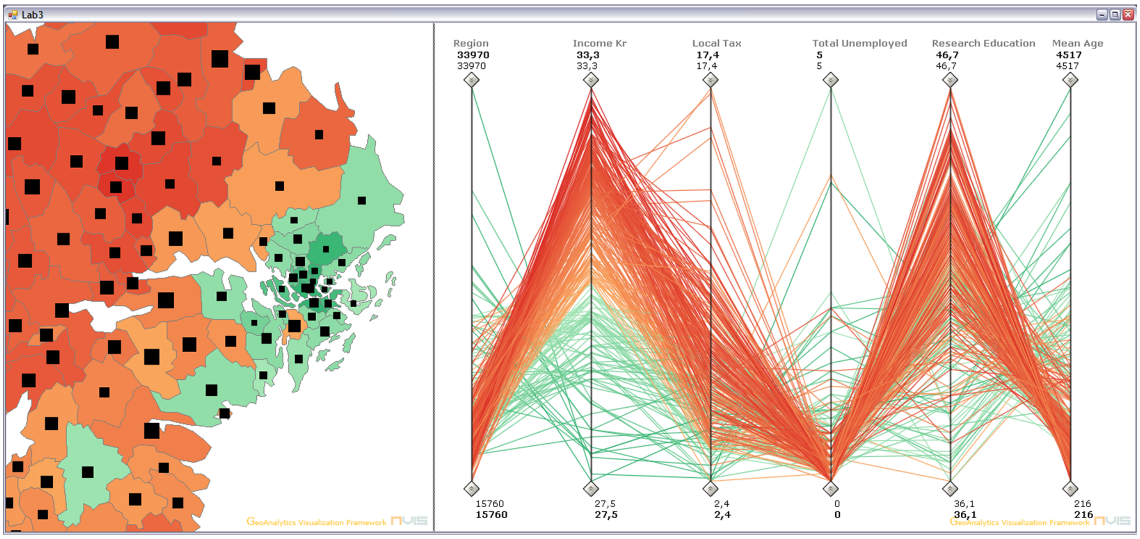
TNM048 – Information Visualization
3. GeoVisualization

February 13, 2007



1 Introduction

This lab exercise will introduce the spatial dimension by combining geovisualization with information visualization. A choropleth map will be used in combination with a parallel coordinates plot to visualize Swedish social science data.

To pass this exercise, functional code as well as your explanations of how it works are required. The laboratory exercise should be performed individually or in a group of maximum two students. After completing all tasks, present your results to the laboratory assistant.

2 Project

A standard windows application project is already created and can be found in the same zip-file as this document.

Task 1:
Open the Lab3 project.

3 Data

The dataset in this lab is a Swedish social science dataset. Every row in the dataset corresponds to one Swedish municipality.

Task 2:
Insert the data from the excel file into a new DataCube by using the LabExcelReader class. Before inserting the data into the DataCube, the first column of the dataset has to be removed. Store this column's data in a list or array for later use.

4 Maps in GAV

In GAV the ChoropletMap component is the base class for all maps. A choropleth map is a map in which the areas are shaded or patterned in proportion to a statistical variable. In this lab, both the shading and pattern techniques will be used.

GAV uses shapefiles (.shp) to provide non-topological geometry data for visualization of geographical data. The ShapFileReader class can read shapefiles and insert the data into a GAV MapData object. Beside the geographical information, metadata describing each item in the shapefile is needed by the MapData class. This information is stored in .shx and .dbf files.

Task 3:
Use the ShapeFileReader class to load the geographical data into a GAV MapData object.

The GAV ChoroplethMap contains one or more map layers. The following layer classes will be used:

- Gav.Graphics.MapPolygonBorderLayer - Renders region borders.
- Gav.Graphics.MapPolygonLayer - Renders regions.
- LabGlyphLayer - Renders one glyph per map region.

The layers in the map are order-dependent. This means that the layers will be rendered in the order they were added. Adding a layer to the map is done by calling the AddLayer method.

Task 4:
Create a MapPolygonBorderLayer and add the MapData object recieved from the ShapeFileReader. Also create a ChoroplethMap and add the border layer to it.

Task 5:
Visualize the map by creating a renderer and the other components needed for a GAV VizComponent to be rendered.

As described earlier a choropleth map uses two techniques to visualize statistical variables. The first one, area shading, is done with the MapPolygonLayer.

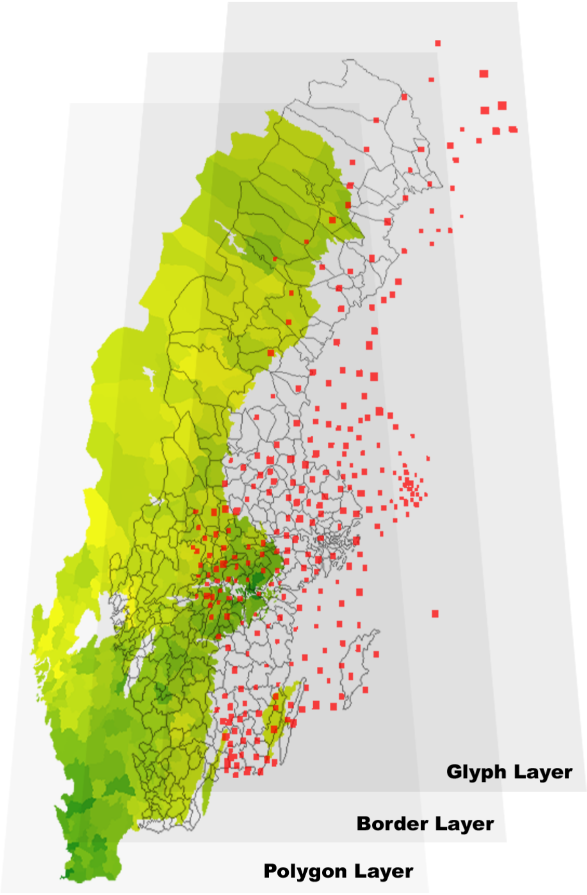


Figure 1: One ore more layers are combined into a choropleth map.

Task 6:
Create a MapPolygonLayer, connect a MapData object, and add it to the map. *Tip:* Remember that the order in which the layers are added determines the rendering order.

The map is now rendered with the same color for every area. To accomplish shading of the map areas, a ColorMap needs to be connected to the polygon layer.

Task 7:
Connect a ColorMap to the polygon layer.

The second technique applies a pattern to every glyph in the map. The LabGlyphLayer class is specifically created for this lab and is already added to the project. This class renders one glyph per map region and uses a GlyphPositioner to calculate the glyph positions. The GlyphPositioner needs the MapData property to be set. The MapData object provides the positioner with spatial data needed for position calculations.

Task 8:
Familiarize yourself with the LabGlyphLayer class and add it to the map. Use the CenterGlyphPositioner.

Task 9:
The current implementation of the LabGlyphLayer class visualizes one dimension (size). Extend the functionality of the class to visualize star glyphs with at least five dimensions per glyph. One of the dimensions should be color.

5 Interaction - Linked Views

Task 10:
Add a parallel coordinates plot (PC) to the application. Use the same ColorMap for the PC and the polygon layer in the map. The headers in the PC should be set from the headers array recieved from the excel reader.

GAV uses an IndexVisibilityHandler class to determine if an index (row) is visibile or not. The PC creates an instance of this class by default to handle the internal thresholding.

Task 11:
Get the PC's IndexVisibilityHandler and assign it to the polygon layer. Now, thresholding in the PC should affect the visibility for the polygons in the polygon layer. *Tip:* Listen to the PC's FilterChanged event and trigger the Invalidate method on the map.

6 Optional

It is possible to detect when a PC header is clicked by listening to the HeaderClicked event. The index of the header clicked is stored in the ClickedHeader property.

Task 12: (optional)
Use the headers in the PC to select which column to map in the color map.