Instructions for cartogram.c

Michael T. Gastner

December 17, 2004

Cartograms are maps in which the sizes of geographic regions such as countries or provinces appear in proportion to their population or some other analogous property. The construction of cartograms is a non-trivial undertaking whose theory is reviewed in our article in PNAS, vol. 101, and the references therein. After we published the article several readers expressed interest in making cartograms, but were concerned about the programming involved.

To meet these concerns we make a general-purpose version of our algorithm publicly available. In what follows we will demonstrate the necessary steps to prepare a cartogram from a conventional map with one concrete example. We will assume that the original map exists as ESRI shapefile or "generate" file. You will also need the GNU C compiler (which is already installed on most Linux distributions).

We decided to use the ESRI formats because you will most likely find a shapefile (extension .shp) suitable for your problem after a quick search on the world wide web. In section 1 we explain how to "ungenerate" a shapefile with ArcInfo. There are other methods to convert the polygon data from a shapefile to an ASCII file, see http://arcscripts.esri.com/, but you must make sure that your input to cartogram.c looks exactly like a "generate" file. This should not need anything but a text editor and a few "replace" commands.

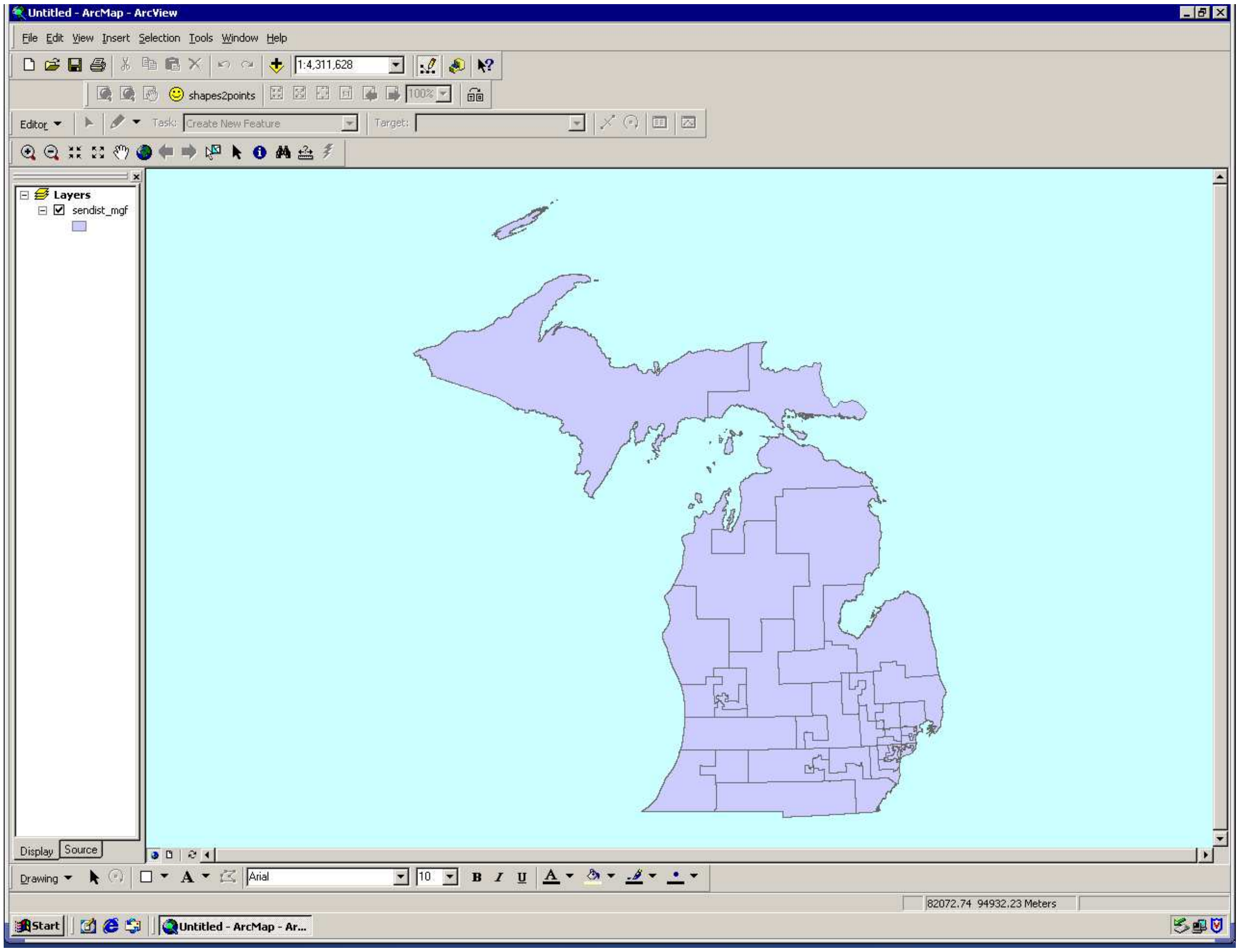
Our example will be a cartogram of senate districts in the state of

Michigan in 2002. There are 38 senate districts containing a population of approximately 212 000 to 264 000 residents, but they are of very different geographic size. Each district sends exactly one senator to the state senate. On an equal-area map a district in Detroit will appear much smaller than in the rural north of the state simply because the population densities are different. For representing the actual power distribution in the senate such a map might be rather misleading because Detroit's districts appear to be small, but their power in the senate is just as big as that of any other district. A cartogram on the other hand, with each district appearing equally large, will reflect the regional influence much better.

1 Obtaining the polygon coordinates from a shapefile with ArcInfo

Your first task would be to find a shapefile for the senate districts. To make your life easier, we already did this for you. Among the files you downloaded with this document you should find the following three files: sendist_mgf.shp, sendist_mgf.shx, and sendist_mgf.dbf. If you have ArcMap available you can take a look at it. (To my knowledge there is no Linux version of ArcGIS, so you may have to go to a Windows machine for this part.)

You will see exactly what we tried to describe before. A few sizeable districts in the north and many tiny districts in the southeast around Detroit, some of them too small to be visible on this scale. For creating our cartogram we need the boundaries for all these districts. We do this by "ungenerating" the shapefile.



Open ArcInfo Workstation and at the ArcPrompt navigate to the directory of the three above mentioned files with the "w" command. Then enter the following sequence of commands:

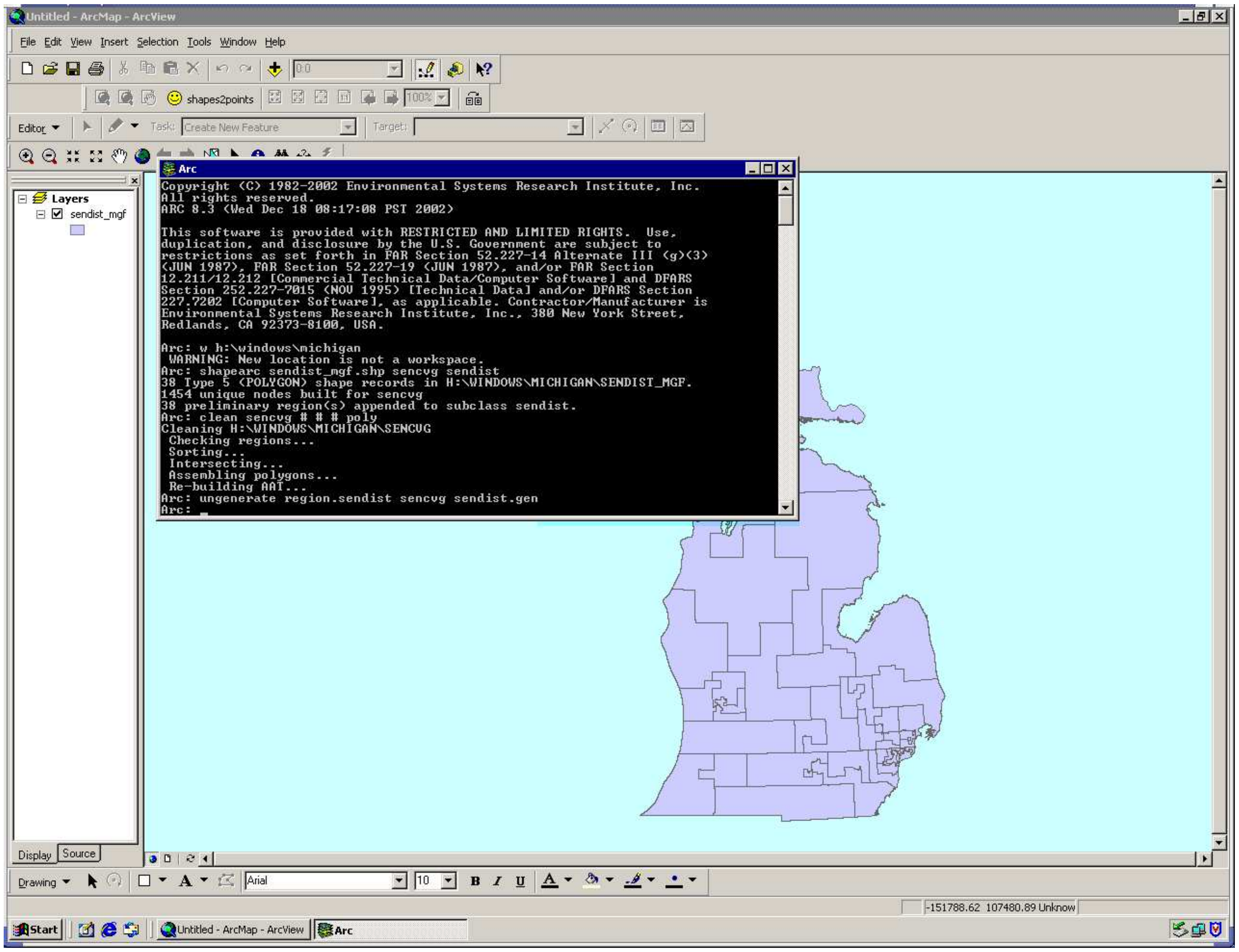
```
shapearc sendist_mgf.shp sencvg sendist
ungenerate region.sendist sencvg sendist.gen
```

If ArcInfo complains about these commands you may try this instead:

```
shapearc sendist_mgf.shp sencvg sendist
clean sencvg # # # poly
ungenerate region.sendist sencvg sendist.gen
```

The next screen-shot shows how ArcInfo should react to each command.

The whole procedure should not take longer than three to four minutes.


Untitled - ArcMap - ArcView
File Edit View Insert Selection Tools Window Help
shapes2points
Editor
Task: Create New Feature
Target:
Layers
sendist_mgf
Display
Source
Drawing
Arial
10
-151788.62 107480.89 Unknow
Start
Untitled - ArcMap - ArcView
Arc

```
Copyright (C) 1982-2002 Environmental Systems Research Institute, Inc.
All rights reserved.
ARC 8.3 (Wed Dec 18 08:17:08 PST 2002)

This software is provided with RESTRICTED AND LIMITED RIGHTS.  Use,
duplication, and disclosure by the U.S. Government are subject to
restrictions as set forth in FAR Section 52.227-14 Alternate III (g)(3)
(JUN 1987), FAR Section 52.227-19 (JUN 1987), and/or FAR Section
12.211/12.212 [Commercial Technical Data/Computer Software] and DFARS
Section 252.227-7015 (NOV 1995) [Technical Data] and/or DFARS Section
227.7202 [Computer Software], as applicable. Contractor/Manufacturer is
Environmental Systems Research Institute, Inc., 380 New York Street,
Redlands, CA 92373-8100, USA.

Arc: w h:\windows\michigan
 WARNING: New location is not a workspace.
Arc: shapearc sendist_mgf.shp sencvg sendist
38 Type 5 (POLYGON) shape records in H:\WINDOWS\MICHIGAN\SENDIST_MGF.
1454 unique nodes built for sencvg
38 preliminary region(s) appended to subclass sendist.
Arc: clean sencvg # # # poly
Cleaning H:\WINDOWS\MICHIGAN\SENCVG
 Checking regions...
 Sorting...
 Intersecting...
 Assembling polygons...
 Re-building AAT...
Arc: ungenerate region.sendist sencvg sendist.gen
Arc:
```

In the last step you created the ASCII file sendist.gen. Open it with a text editor and take a closer look at it. Here is what you should see:

```
   1
0.3248690E+06 0.8558454E+06
0.3248376E+06 0.8557575E+06
0.3248171E+06 0.8556783E+06
0.3247582E+06 0.8556348E+06
...
```

```
  0.3248690E+06 0.8558454E+06
END
     1
  0.3248690E+06 0.8558454E+06
  0.3249651E+06 0.8558901E+06
  ...
  ...
END
END
```

On the first line you find an integer followed by several lines with two floating point numbers each. The integer identifies the region (here: senate district 1), and the two columns of floating point numbers are x- and y-coordinates around the region's boundary. The first and last two numbers are the same because the line around the boundary closes upon itself. The line is then terminated with "END", and the next polygon begins. Note that the second polygon has again identifier "1" because senate district 1 consists of several disjoint polygons. The end of the file is marked by two consecutive lines "END". [1]

You do not have to start from shapefiles if the boundary coordinates are given in some other format. But then you must make sure that you create a file that follows the rules for "generate" files. Only files in this "generate" format can be used as input for cartogram.c.

2 Preparing the cartogram

We have to tell the computer how we want to rescale each region. In our example we simply want all senate districts to be equally large. The way

[1] Polygons can also contain "holes" like a donut, e.g. lakes. To distinguish these holes from the enclosing polygon the orientation is clockwise for the exterior and anticlockwise for the interior boundary.

to do this is to write a simple text file with two numbers on each line. The first number is the region identifier from the .gen file. If you have the database file (extension .dbf) that came with the shapefile you can find out the identifiers from there. The second number is the number of senators. (In your applications it might be the number of residents, disease cases, gross regional income, etc. You can also use floating point numbers.) This can be followed by an optional comment. Everything following the second number of each line will be ignored by the code, so you can write whatever you consider useful information, For 38 senate districts the text file simply looks like this

```
1 1 Senate District 1
2 1 Senate District 2
...
38 1 Senate District 38
```

If e.g. the second district could elect four senators we would replace the second line by

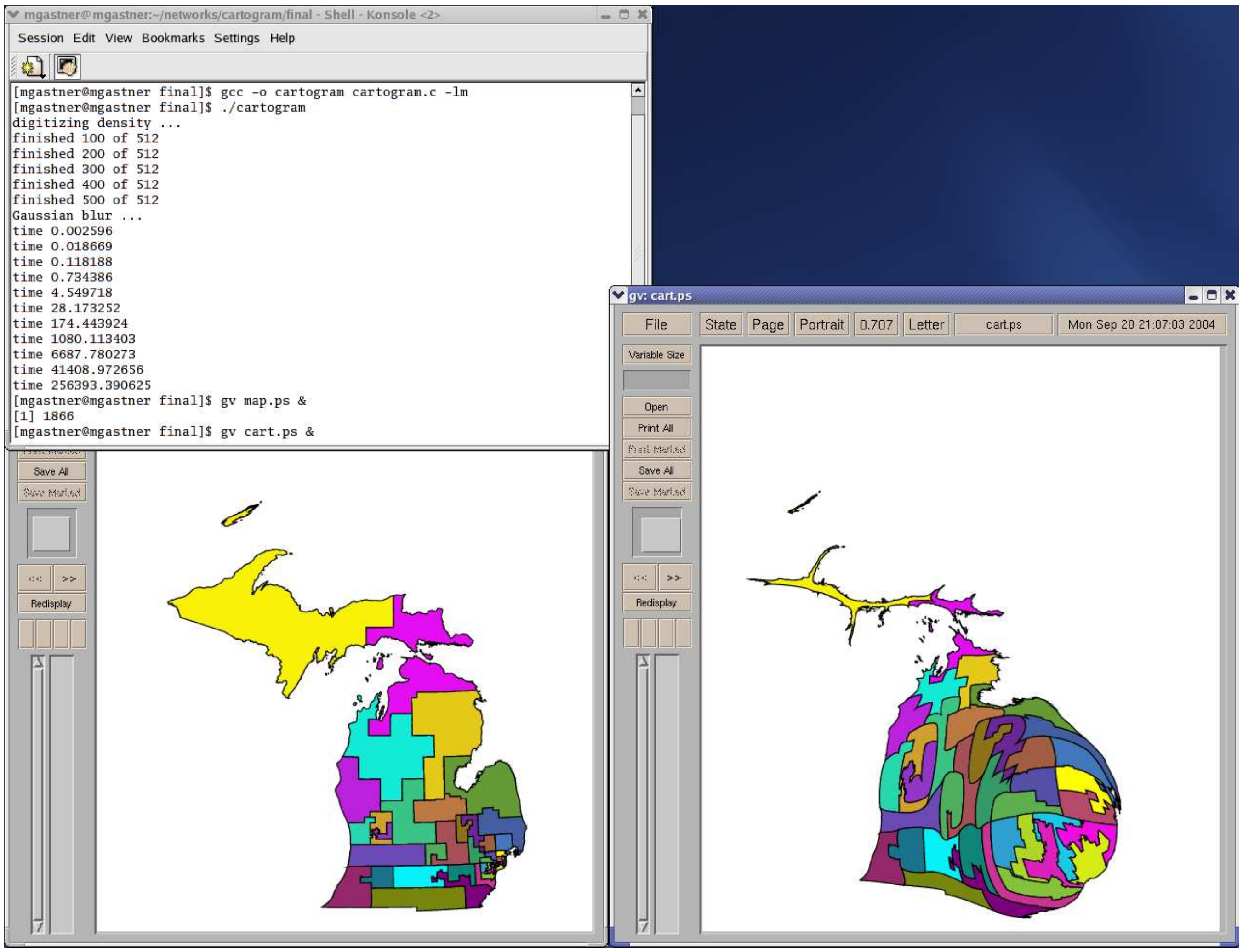
```
2 4 Senate District 2
```

and so on. For every region identifier appearing in the .gen file there *must* be a corresponding line here. If not, cartogram.c will return an error message. You can find the text file for this example as census.dat among the files you downloaded with this document.

Now we are almost ready to run the cartogram algorithm. Make a copy of sendist.gen in the same directory where you saved cartogram.c and rename it “map.gen”. If census.dat is not yet in this directory move it there. Open a Linux terminal, navigate with “cd” to said directory, and compile and run the cartogram code by entering the following commands in the terminal window:

```
gcc -o cartogram cartogram.c -lm
```

```
./cartogram
```



```
[mgastner@mgastner final]$ gcc -o cartogram cartogram.c -lm
[mgastner@mgastner final]$ ./cartogram
digitizing density ...
finished 100 of 512
finished 200 of 512
finished 300 of 512
finished 400 of 512
finished 500 of 512
Gaussian blur ...
time 0.002596
time 0.018669
time 0.118188
time 0.734386
time 4.549718
time 28.173252
time 174.443924
time 1080.113403
time 6687.780273
time 41408.972656
time 256393.390625
[mgastner@mgastner final]$ gv map.ps &
[1] 1866
[mgastner@mgastner final]$ gv cart.ps &
```

On my computer the program finishes after 90 seconds. (There are ways to make it faster at the expense of accuracy, or to make it more accurate with higher demands on your memory. We suppose the default parameters declared in the code define a reasonable compromise for most users.) There should be three new files in the current directory. First, you will find the cartogram in "generate" format as cartogram.gen which you can use to create a new shapefile.[2] If you are only interested in images of

[2]You could for example use gen2shp, see http://www.intevation.de/ jan/gen2shp.

map and cartogram, open the other two new files, map.ps and cart.ps.

```
gv map.ps &
gv cart.ps &
```

They contain postscript images of the beginning and end product. A comparison shows a hugely expanded southeast while the upper peninsula has become very thin.

3 Bug reports etc.

We hope this introduction will be of some help to you. The code is designed to be relatively robust to different inputs. If you receive cryptic warnings or error messages, suspect a bug, are dissatisfied with the performance, have questions or suggestions please do not hesitate to send an email to *mgastner@umich.edu* describing your problem. Your feedback will help us improving the code.