36-315: Statistical Graphics and Visualization

Lab 13

Date: April 14, 2003 **Due: end of lab**

In this lab, you are given the commands to type in, but with question marks (??) denoting things you must provide, based on the dataset you are using.

1. Download all the files for this lab into `My Documents`. Other folders will not necessarily work.
2. Unzip `maps.zip` by pressing right mouse and selecting `Extract All...`. In the wizard, click `Next`, `Next`, and `Finish`. If the computer complains about WinZip, move to another.
3. Open a Word document to record your work (plots and commands).

Start R

4. `Start -> Programs -> Class software -> R 1.5.1`
5. Set the working directory to `My Documents`:

   ```
   File -> Change dir...
   ```

6. Load the special functions for this lab:

   ```
   source("lab12.r")
   ```

Load your data

7. From the class web page, download the census data files for your state. Unzip them into `My Documents`. Then they can be loaded via

   ```
   a = read.csv("tracta.csv")
   b = read.csv("tractb.csv")
   frame = clean.census.data(a,b)
   ```

8. From the class web page, download the census tract boundary files for your state. Unzip the files into `My Documents`. Then project the map:

   ```
   mt.orig = read.cob("tr??_d90_p.dat","tr??_d90_pa.dat")
   mt = map(mt.orig,proj='albers',par=c(29,45))
   ```

Identifying outliers

9. Make a scatterplot of your response versus one predictor for your state. Use `identify` and select an outlier on the plot:

```
i = identify(??,frame)
```

The transformation you use will affect your ability to find the outliers. (If there turns out to be no real outlier, pick some point anyway.)

10. Highlight the outlier on the plot:

```
points(??,frame[i,],col="red",cex=2)
```

Save this plot.

11. Make a map, and color the corresponding tract:

```
map(mt)
map(mt,i,??)
```

If you have a hard time finding the tract, make an outline of your state instead, then paste over it:

```
map("state",??,proj='albers',par=c(29,45))
map(mt,i,??)
```

12. To make the tract more visible, zoom in using `locate.bbox`:

```
bbox = locate.bbox()
map(mt,xlim=bbox[1:2],ylim=bbox[3:4])
map(mt,i,??)
```

Save this plot.

Explaining outliers

13. Before you can use projection, all variables must be transformed. The file `variables.r` contains a suggested set of variables and transformations, in a formula called `goodvars`.

```
source("variables.r")
```

You can make a new frame containing these transformed variables as follows:

```
x = model.frame(goodvars,frame)
```

The validity of the transformations for your state can be checked via

```
hist(x[1:40])
hist(x[41:76])
```

You can edit `variables.r` later to use any other transformations you'd like.

14. To explain the outlier, you want to use variables other than the two that originally identified the outlier. You can remove them from `x` using `not`. For example:

```
x = not(x,"log(POPPSQMI)")
x = not(x,income.vars)
x = not(x,age.vars)
```

`income.vars` and `age.vars` are defined in `variables.r` and name all variables involving income or age, respectively. To see what variables are left:

```
x[1,]
```

The name you give to `not` must exactly match the column name in `x`, including its transformation and spacing, or `not` will silently remove nothing.

15. Now use `separate` to find a projection that separates `i` from the rest of the tracts:

    ```
    separate(x,i)
    ```

    This will return the weights for the horizontal axis of the projection. Large magnitude weights indicate possible explanatory variables. Note that the weights are only a suggestion for what variables to look at—they should not be taken as an explanation of the outlier by themselves.

16. Returning to the original scatterplot, color the points according to your new explanatory variable:

    ```
    color.plot(? ~ ? + ?, frame)
    ```

    Is the outlier explained?

17. Show us your graphs.

Identification Suppose you have a scatterplot of `y` versus `x`. If you run this command, then click on a point, the name of that point will be placed in `i`:

```
i = identify(y ~ x, frame)
```

In general, the usage is:

```
i = identify(<formula>,frame,<number of points>)
```

where `formula` names the two axes of the plot. This also works in maps:

```
i = identify(<map object>)
```

When you click on a region, that region's name is placed in the variable `i`.

Highlighting a point Suppose you have a scatterplot of `y` versus `x`. This command will highlight the points named in `i`:

```
points(y~x,frame[i,],col="red",cex=2)
```

In general:

```
points(<formula>,frame[i,],col=<color>,cex=<size>)
```

Zooming a map `locate.bbox()` waits for you to click on two places in the map. These are assumed to be opposite corners of a bounding box. The two locations are returned in a vector.

Filling or outlining a tract Given the name of one or more tracts in the variable `i`, you can fill or outline them on a map via

```
map(<map object>,i,fill=T,col="red",border="red",add=T)
map(<map object>,i,col="red",add=T)
```

Since you are adding to an existing map, you don't need to specify limits.

Constructing a transformed frame In general:

```
x = model.frame(<formula>,frame)
x = not(x,<character vector>)
x = not(x,<another character vector>)
```