1 Lattice Graphics

Lattice graphics are a collection of clever high level graphics commands for displaying relationships among 3 or more variables. Given the usual 2 dimension limit that paper imposes on us, this is indeed a challenge for the creative scientist.

Lattice graphics functions do this generally, by presenting an array of similar graphs showing a relationship between two variables (e.g. a scatter plot) for cleverly selected subsets of the other variables. The clever sub-setting of the data is referred to as \textit{conditioning}—think of “conditional” probability.

1.1 A cooked up example

To illustrate the wonder of lattice graphics, consider this data from the planet Zorch.

> ## load the data stored in R's special binary format
> ## this gets us a data.frame called "society"
> load(file="/hdir/0/carlm/213/LatticeGraphics/PlanetZorch.rsave")
> ## summary statistics
> summary(society)

<table>
<thead>
<tr>
<th></th>
<th>income</th>
<th>height</th>
<th>IQ</th>
<th>religion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min.</td>
<td>0</td>
<td>59.32</td>
<td>9.482</td>
<td>amana</td>
</tr>
<tr>
<td>1st Qu.</td>
<td>39459</td>
<td>64.58</td>
<td>82.687</td>
<td>oneida</td>
</tr>
<tr>
<td>Median</td>
<td>50212</td>
<td>65.96</td>
<td>101.067</td>
<td>shaker</td>
</tr>
<tr>
<td>Mean</td>
<td>50296</td>
<td>65.95</td>
<td>100.305</td>
<td>swedenborgian</td>
</tr>
<tr>
<td>3rd Qu.</td>
<td>62416</td>
<td>67.39</td>
<td>117.736</td>
<td></td>
</tr>
<tr>
<td>Max.</td>
<td>94272</td>
<td>72.04</td>
<td>184.911</td>
<td></td>
</tr>
</tbody>
</table>

> ##
> ########################################################################
> ## And wow -- look at the power of this regression
> ## check out the T value on that intercept!
> ########################################################################
> summary(lm(income~IQ+height+religion,data=society))

Call:
  lm(formula = income ~ IQ + height + religion, data = society)

Residuals:
    Min     1Q    Median     3Q    Max
-50164  -11128    -39  11885  43907

Coefficients:  Estimate Std. Error t value Pr(>|t|)
(Intercept)  32434.04    16716.13   1.940  0.0526 .
IQ          -11.35      20.56    -0.552  0.5810
height      277.42      251.38    1.104  0.2700
religiononeida  387.07    1460.68   0.265  0.7911
religionshaker  1874.92   1458.39   1.286  0.1989
religionswedenborgian  556.10   1465.70   0.379  0.7045
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16300 on 994 degrees of freedom
Multiple R-squared:  0.003438,    Adjusted R-squared:  -0.001575
F-statistic: 0.6859 on 5 and 994 DF,  p-value: 0.6342

Hmmm perhaps the above regression coefficients are not so compelling after all. Well then how about some plots showing the effect of stature and IQ on income?
Oh well, I guess reasonably careful scientist might conclude from this that studying the effects of height and IQ on income is perhaps not so promising a path to tenure. And yet that scientist would be wrong. Figure 1 shows an example of a xyplot() function. xyplot is the lattice version of the basic scatter plot produced by the plot() function.

Notice that in Figure 1, in place of a single scatter plot, we have a matrix of plots. Each cell of the matrix contains one plot show the relationship between “income” and “IQ” (on the planet Zorch) for a subset of the data. The lowest and leftest plot shows the relationship only for members of the “Amana” “religion” having a certain range of height The range of height is shown by the darker shading in the region above the plot. Since all of the plots have the same X and Y range, you can scan across the rows or up and down the columns to compare the shapes and general levels of the relationship between income and

1well not really–these are after all entirely made up data
IQ _conditioning_ on religion and stature.

If you scan up any column, you _might_ notice that the height range displayed in each of the plots overlap the range in the plots above and below. This represents a sort of compromise between treating height as a categorical, vs continuous variable.

There are many other lattice plots which share many syntactical features. All of them require that you load the lattice package via the command:

\[
> \text{library(lattice)}
\]

The lattice graphics functions are a bit less intuitive than the graphics functions that we learned last week (if you can believe that). But once you get the hang of _xypot_() you will find the others work the same way.

Note that because lattice graphics generally create many similar graphs at once, the wonderful paradigm of creating a graph and then adding stuff to it is lost. I am sad about this, but so it goes.
> ## an xyplot example using made up data
> library(lattice)
> print(
> + xyplot(income~IQ | religion + equal.count(height, n=4, overlap=.25),
> + data=society,
> + type=c("p","smooth"), cex=.5, col="black", col.line="blue",
> + main="Income and IQ conditioning on height and religion")
> + )
> )

**Figure 1:** an xyplot showing relationship among four variables
2 Lattice plots with IPUMS data

Social scientists seldom stumble across relationships as strong as those from the planet Zorch, but lattice graphics are good tool for doing the best we can.

```r
library(foreign)
acs09<-read.dta(file="/data/commons/carlm/ACS2009/acs09.dta")
save(file="/72hours/acs09.Rsave",acs09)
names(acs09)
[1] "year"   "datanum"  "serial"   "hhwt"   "region"  "statefip"
[7] "gq"     "pernum"   "perwt"    "famsize" "nchild"  "nchlt5"
[13] "age"    "sex"      "marst"    "marrno"  "marrinyr" "widinyr"
[19] "divinyr" "fertyr"   "race"     "raced"   "bpl"     "bp1l"
[25] "citizen" "yrnatur"  "yrimmig"  "hispan"  "hispand" "hcovany"
[31] "hcopriv" "hinsemp"  "hinspur"  "occ"     "migrate1" "migrate1d"
[37] "diffmob"

## create a numeric age variable
acs09$ageN<-as.numeric(as.character(acs09$age))
## The "N/A" here is NOT the same as NA..
## create a new "logical" ie TRUE/FALSE variable
acs09$fertyrL<-ifelse(acs09$fertyr=="Yes",TRUE,FALSE)
acs09$citizenF<-factor(as.character(acs09$citizen))
levels(acs09$citizenF)
[1] "Born abroad of American parents" "N/A"  
[3] "Naturalized citizen"   "Not a citizen"
levels(acs09$citizenF)[2]<-"US Born"
acs09$bpl.immig<-as.character(acs09$bpl)
acs09$bpl.immig[acs09$citizenF == "US Born"]<- "US"
acs09$bpl.immig<-factor(acs09$bpl.immig)
## Load the lattice library
```

```r
library(lattice)
acs09$yrnatur[acs09$yrnatur==9999]<-NA
summary(acs09$yrnatur)
```
Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

> #################################################################
> ## A box-whisker plot to examine the distribution of duration
> ## between immigration and naturalization across different immigrant groups
> #################################################################
> acs09$natur.yrs<-with(acs09, yrnatur - yrimmig)
> print(bwplot(natur.yrs~marst|hispan, data=acs09))
> ## A plot of immigrants by nationality
> ## exclude "US Born" of course
> top10<-names(sort(table(acs09$bpl.immig),decreasing=TRUE))[2:11]
> print(bwplot( nchild~natur.yrs|bpl.immig,horizontal=TRUE,
> + data=acs09,subset=bpl.immig %in% top10))
> ## note the odd negative values are quite unlikely given that
> ## America's naturalization law has required a five year wait since
> ## 1795.
> > sum(acs09$natur.yrs <=0 & acs09$citizenF == "Naturalized citizen")
> [1] 2651
>
```r
> head(ynr.bpl)

<table>
<thead>
<tr>
<th>head(ynr.bpl)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ynr.bpl &lt;- with(acs09,</td>
</tr>
<tr>
<td>tapply(natur.yrs, bpl, median, na.rm = TRUE)</td>
</tr>
<tr>
<td>head(ynr.bpl)</td>
</tr>
</tbody>
</table>
```
> ynr.bpl <- ynr.bpl[!is.na(ynr.bpl)]
> dim(ynr.bpl)

[1] 70

> ## ynr.bpl is a 1X163 "martix" with one value for each birth place.
> ## use stack to convert ynr.bpl to form more useful for lattice plots
> dim(ynr.bpl)

[1] 70

> ynr.bpl <- stack(ynr.bpl)
> dim(ynr.bpl)

[1] 70 2

> names(ynr.bpl)

[1] "values" "ind"

> head(ynr.bpl)

   values     ind
1    13 American Samoa
2    11       Canada
3    10 Atlantic Islands
4    14       Mexico
5    13 Central America
6    10        Cuba

> print(dotplot(ind~values, data=ynr.bpl))
## OK but this is better
```r
ynr.bpl$country<-reorder(ynr.bpl$ind,ynr.bpl$values)
print(dotplot(country~values,data=ynr.bpl))
```
3 Assignment

Your job this week is to download (or reuse) an IPUMS data set and create several interesting lattice graphic showing the relationship among three or more variables. The fastest and best way to proceed is:

1. Launch RStudio – perhaps by visiting [www.demog.berkeley.edu/rstudio](http://www.demog.berkeley.edu/rstudio) if you have not tried that before.
2. Create, as usual, a new project in the (existing) 213/weekN directory which you created for this purpose.


4. Create several lattice graphs using data from IPUMS. The best students will visit IPUMS and create a new dataset for this project. Students who are secure enough to not care what the instructor thinks, can save time by using the ACS 2009 extract that can be loaded from the demonstration.r program.

3.1 This week’s reading: most of Lattice Graphics: an Introduction

This document by Deepayan Sakar does a nice job of introducing the big ideas of Lattice Graphics and give nice examples which you can and should do as you read it. Here’s the URL http://www.bioconductor.org/help/course-materials/2008/advanced_R/latticeLab.pdf

As always when reading computer related stuff, the most important thing is to understand what is possible rather than to memorize the gritty details. If you know what’s possible you can always figure out how to do it.

When cutting and pasting from Lattice Graphics: an Introduction into Rstudio, be sure to drop the commands into the source pane rather than the console where R is running. This will work better and allow you to easily change things and experiment.

3.2 Create graphs from IPUMS data

Once again, the main point is learning how to use the lattice graphics system making an original and significant contribution to science is optional.

You are by now expert at extracting data from IPUMS and reading it into R. You are encouragies to ply that skill and grab a new IPUMS extract. However, you may also reuse the data set that you used for calculating an interesting rate – or even use the ACS 2009 extract, if you can take the hit to your self-respect.

Now, using the ACS data, create several lattice plots each of which should reveal an important relationship in the data.

You are encouraged to experiment with the following plots of which the xyplot is the most useful:

- xyplot
- bwplot
- histogram
- dotplot

2well, a relationship anyway
3.3 Hints and advice
You are now an expert on all things related to IPUMS, but I shall nonetheless presume to remind you of a few things:

- In your extract, it cannot hurt to include the `statefip` variable as this can allow you to easily select a single state to work on – at least while you perfect your code. There is no particular value in this exercise, in using a large dataset when a small one will do just as well.

- After importing the data from your .dta file, many factors include levels with zero observations. These phantom levels can be removed most easily by running:

  ```r
  > acs$nameOfVariable <- factor(as.character(acs$nameOfVariable
  ```

- Many IPUMS variables have missing value codes that are numeric. Check the codebook and then turn those missing values into NAs:

  ```r
  > acs$someVariable[acs$someVariable == 99999999999999] <- NA
  ```