Post02 - Lattice

Yash Sanghrajka

November 27, 2017

Introduction

In this course, we have used a number of different data visualization methods in order to portray data the way we want. We have used the base R plotting methods, including plot, hist and bar. We have used ggplot, which has an assortment of different plotting techniques that make it very easy to plot and visualize data easily and effectively. Most recently, we have used ggvis, which not only "Declaratively describe data graphics with a syntax similar in spirit to ggplot2", but also can "Leverage shiny's infrastructure to publish interactive graphics usable from any browser", according to the ggvis site.

However, there is one library that we haven't covered in depth that can still be very useful, which is lattice.

According to Deepayan Sarkar, "The lattice add-on package is an implementation of Trellis graphics for R. It is a powerful and elegant high-level data visualization system with an emphasis on multivariate data. It is designed to meet most typical graphics needs with minimal tuning, but can also be easily extended to handle most nonstandard requirements."

As we can see, lattice takes a very in-depth approach to visualization, and lattice can be used to visualize all different sorts of data. We will be getting into the data in the examples.

Audience

Before I dive straight into the lattice package, I want to discuss the audience I am trying to reach for this post. This post can be applied for people in Statistics 133, as this provides another tool for them to visualize their data accurately and effectively. On top of that, it can be applied to anybody learning R and looking to learn more about data analysis and visualization. Lattice is a package that has a lot of applications, and can be used for a lot of different datasets, so it has a wide audience.

Syntax and Examples

To use the functions in lattice, we have to first import the lattice library.If you have not installed it yet, install it first.

```
#calling the lattice library
library(lattice)
```

The general format of lattice graphs is graph_type(formula).

Here are some formulas that lattice has to offer, according to Robert I. Kabacoff.

graph_type	description	formula examples
barchart	bar chart	x~A or A~x
bwplot	boxplot	x~A or A~x
cloud	3D scatterplot	z~x*y|A
contourplot	3D contour plot	z~x*y
densityplot	kernal density plot	~x|A*B
dotplot	dotplot	~x|A
histogram	histogram	~x
levelplot	3D level plot	z~y*x
parallel	parallel coordinates plot	data frame
splom	scatterplot matrix	data frame
stripplot	strip plots	A~x or x~A
xyplot	scatterplot	y~x|A
wireframe	3D wireframe graph	z~y*x

Let's now dive into some of these examples, and show how we can use them.

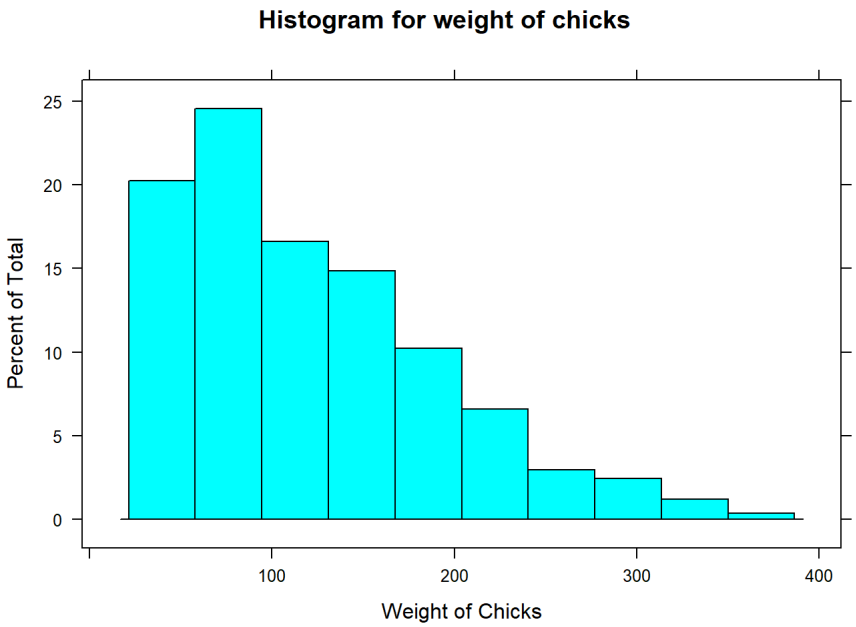
The first dataset we are going to visualize is the ChickWeight dataframe built in to R. This dataset is shown below and has four different columns, the weight of the chick, what time the chick was measured at, which chick it is (ID number), and most importantly for us, which of the four diets the chicks are on (designated by 1,2,3,4).

```
#ChickWeight dataframe
head(ChickWeight)
```

```
##   weight Time Chick Diet
## 1     42    0     1    1
## 2     51    2     1    1
## 3     59    4     1    1
## 4     64    6     1    1
## 5     76    8     1    1
## 6     93   10     1    1
```

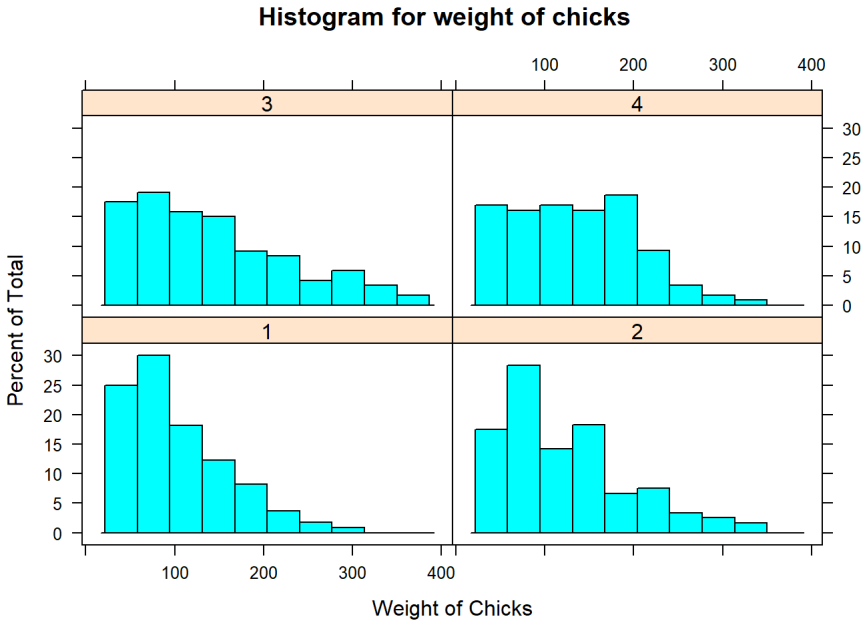
Let us first use the histogram visualization. As shown in the table of formulas above, we only need to pass in the column as the formula. We use data to signify what the source is, main for the title, and xlab for the label of the x-axis.

```
#histogram of weights from ChickWeights
histogram(~weight,data = ChickWeight,main = "Histogram for weight of chicks", xlab = "Weight of Chicks")
```



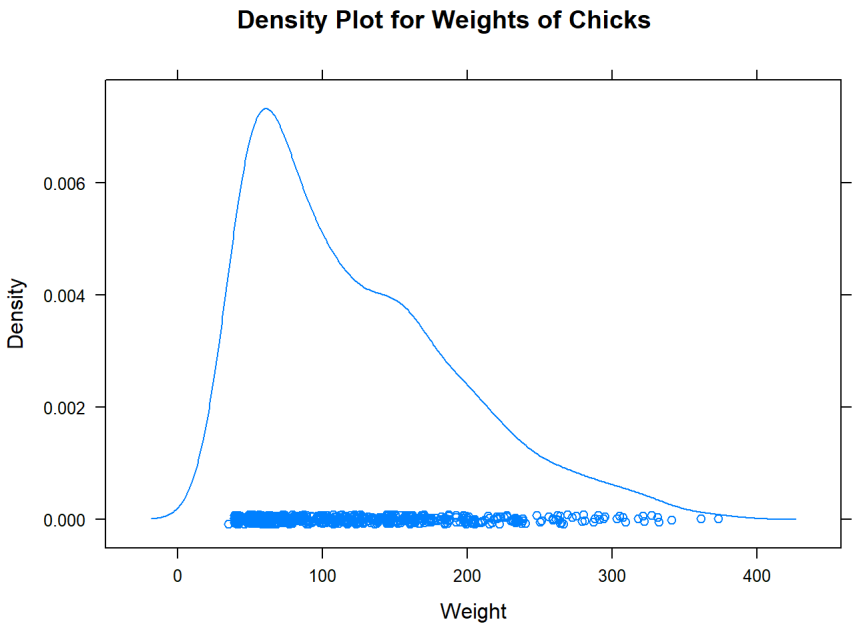
Now, let's show off some of the functionality of lattice. What if I wanted to see the distribution of weights, but by which diet the chick was on. We would just add that column as a factor, as shown by the |Diet added below.

```
#histogram of weights from ChickWeights with factors
histogram(~weight|Diet,data = ChickWeight,main = "Histogram for weight of chicks", xlab = "Weight of Chicks")
```



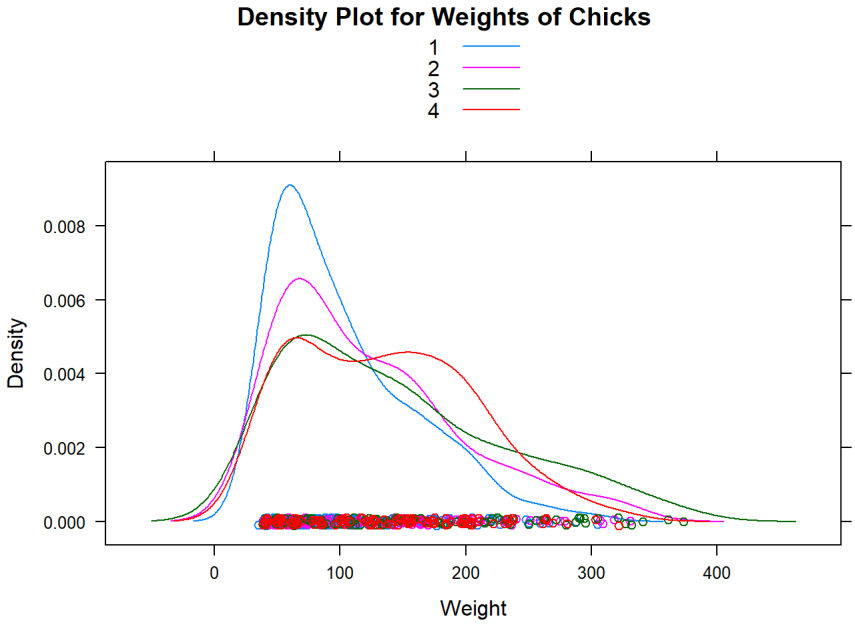
Now let's make a density plot of those weights. Similar to the histogram, the formula remains constant, and all that changes is the function.

```
#density of weights from ChickWeights
densityplot(~weight,data = ChickWeight,main = "Density Plot for Weights of Chicks",  xlab = "Weight")
```



Let's now try to divide it up by diet. However, if we used the way above (|Diet), that would return us four quadrants of the density plot, divided up by Diet. But this time, we want to overlay it. If we want to overlay the four density functions, we would use groups, as shown below. We also want a key, so we would add auto.key to show the color differential between the density plots.

```
#density of weights from ChickWeights grouped by Diet
densityplot(~weight,data = ChickWeight,main = "Density Plot for Weights of Chicks", xlab = "Weight",groups = Diet,
auto.key = TRUE)
```



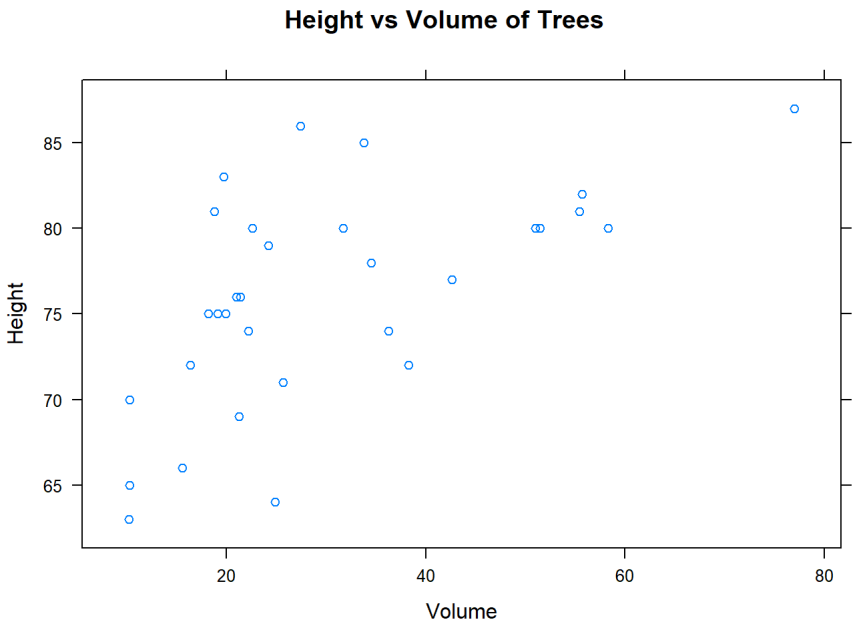
Now, we want to show off some of the other functionality of lattice. For that, we will be using a different dataset. We will now be using trees, a built in data-set that has the girth, height and volume of different trees that they have measured.

```
head(trees)
```

```
##   Girth Height Volume
## 1   8.3     70   10.3
## 2   8.6     65   10.3
## 3   8.8     63   10.2
## 4  10.5     72   16.4
## 5  10.7     81   18.8
## 6  10.8     83   19.7
```

First, let's create a generic XY-plot in lattice. As we can see below, because we do not have a xlab or ylab assignment, it automatically uses the name of what is being inputted.

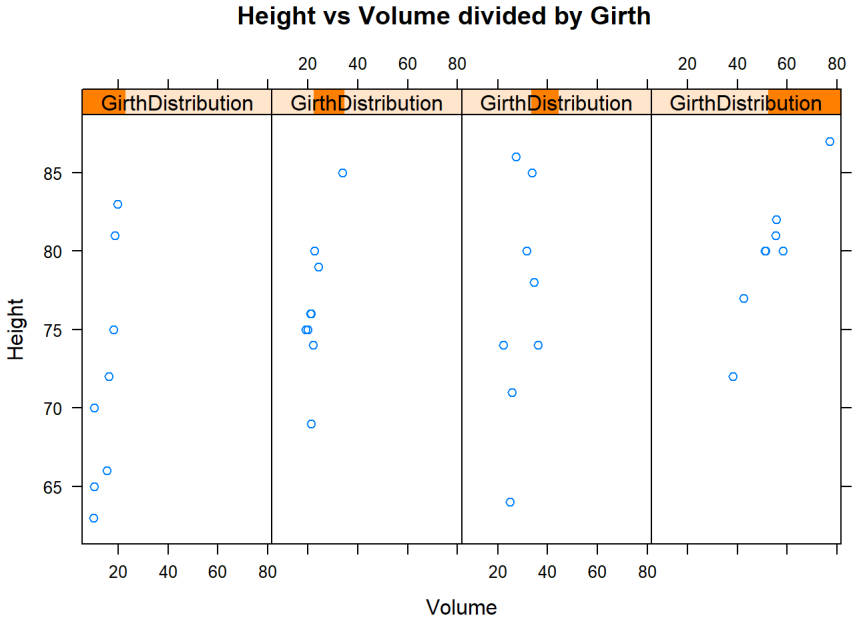
```
#XY-plot comparing height and volume.
xyplot(Height~Volume, data=trees,main = "Height vs Volume of Trees")
```



Now, we want to do something interesting. What if we were able to show the XY-plot above, but the 4 quantiles of Girth. We can do that by factorizing like above, but also using equal.count to divide up the four quantiles.

We need to use layout to show that there 4 different graphs that need to be shown.

```
#Quantile distribution of Girth
GirthDistribution <- equal.count(trees$Girth, number=4, overlap=0)

#XY-plot divided up by quantile
xyplot(Height~Volume|GirthDistribution, data=trees,
 main = "Height vs Volume divided by Girth",
 layout=c(4, 1))
```



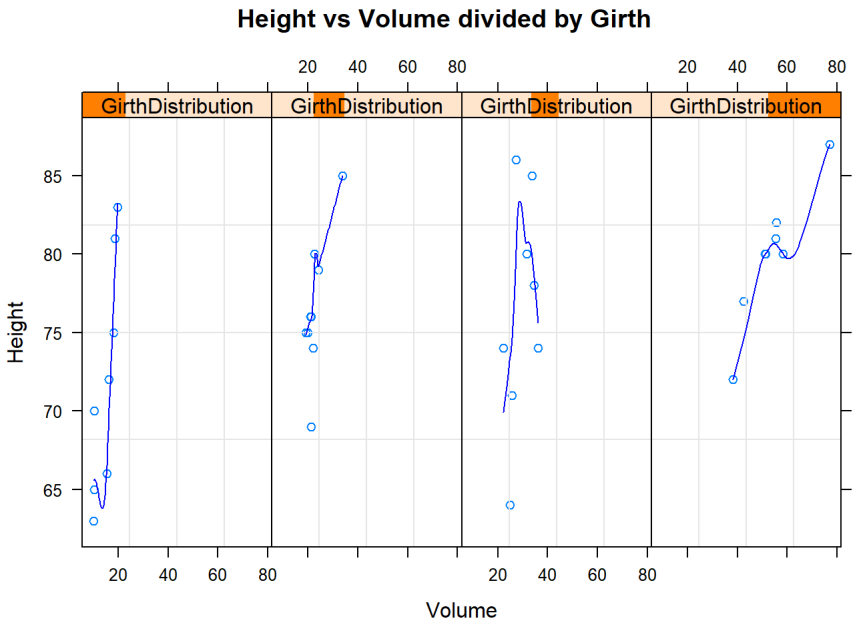
As we can see by the Red boxes shown at the top of the graph, we can see that the quantiles are not uniform, as they should be, and that the height and weight correlation can change based on its girth.

The key aspect that we haven't covered yet, and one that is a major reason why people use lattice is its use of functions.

You can use functions to change the attributes of the graph.

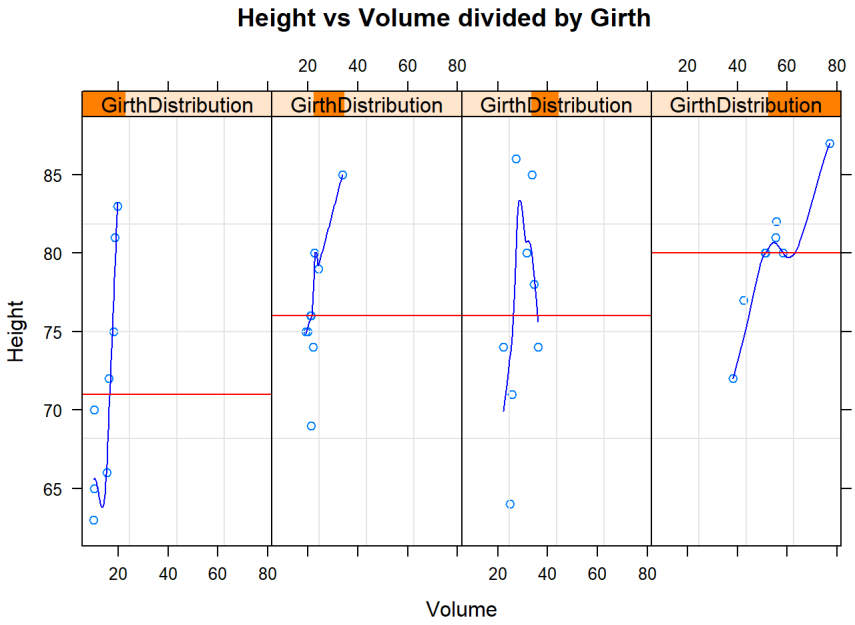
For example, we are taking the same table as above, and adding a grid and loess line in panel_graph.

```
#Function for graph
panel_graph <- function(x, y) {
 panel.xyplot(x, y)
 panel.grid()
 panel.loess(x, y, col="blue")
}

#Quantile distribution of Girth
GirthDistribution <- equal.count(trees$Girth, number=4, overlap=0)

#XY-plot divided up by quantile
xyplot(Height~Volume|GirthDistribution, data=trees,
 main = "Height vs Volume divided by Girth",
 layout=c(4, 1),panel = panel_graph)
```



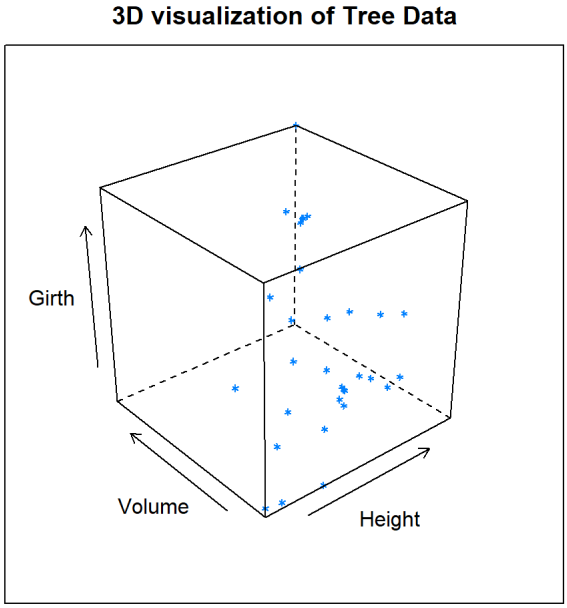
In addition, we can have conditionals in the function. In the graph below, we have the median of the y-value for each quantile. We can use the panel.abline function to add an abline to the graph.

```
#Function for graph
panel_graph <- function(x, y) {
 panel.xyplot(x, y)
 panel.grid()
 panel.loess(x, y, col="blue")
 panel.abline(h = median(y),col="red")
}

#Quantile distribution of Girth
GirthDistribution <- equal.count(trees$Girth, number=4, overlap=0)

#XY-plot divided up by quantile
xyplot(Height~Volume|GirthDistribution, data=trees,
 main = "Height vs Volume divided by Girth",
 layout=c(4, 1) , panel = panel_graph)
```



There are also 3D visualizations in lattice. We can use the cloud function in lattice to compare the girth, height, and volume all on one graph.

```
#3D visualization of tree data
cloud(Girth ~ Height * Volume, data = trees, main = "3D visualization of Tree Data")
```



These are just some of the examples of how lattice can be used to visualize data in a very unique way.

Advantages of Lattice

Lattice has many advantages, but I am going to focus on three major benefits that lattice has that set it apart.

1. It is intuitive.

The syntax of lattice is very easy to pick up, and follows a general structure that is very easy to use. While it has many features, each feature is worked into the syntax of lattice seamlessly.

2. It is extensive.

What I covered today has a lot of content, but may be the tip of the iceberg when it comes to lattice. When we start dealing with really intense and complex datasets, lattice has the tools to be able to analyze and visualize it in a way that other packages can't. One reason why this is has to do with the panels feature inside lattice. By being able to integrate panels into lattice, we are able to take on a wide variety of different visualizations and make them work seamlessly together, which is fantastic when it comes to very complex data.

3. It is continuously growing.

Lattice is continuously growing, and the developers of lattice are adding to the package in order to cover more and more styles of visualizations. This ensures that on top of the many types of visualizations that are already available in lattice, we can see that there will be more features and graphs in the future, allowing us to move past base graphs and visualize higher level data.

Take Home Message

This post has two major take-home messages. The first one is fairly self-explanatory. Lattice is a library in R that has a lot of practical

applications, and can help you better visualize your data. The second take home is somewhat more implicit, however. When visualizing your data, there isn't a clear-cut way to do it. Different packages can provide different benefits when visualizing your data, and there is no 1 correct way to do it. However, the onus is on you to do research and look into the different ways you can visualize better, as there are many tools out there, including lattice.

Thank you for reading this post, I hope you learned more about lattice.

References

http://lattice.r-forge.r-project.org/index.php

http://rstudio-pubs-static.s3.amazonaws.com/7953_4e3efd5b9415444ca065b1167862c349.html

https://ggvis.rstudio.com/

https://www.statmethods.net/advgraphs/trellis.html

https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html

http://www.dummies.com/programming/r/how-to-make-common-graphs-with-lattice-in-r/

http://www.unige.ch/ses/sococ/cl/r/lattice.hist.e.html

https://www.statmethods.net/RiA/lattice.pdf

http://lattice.r-forge.r-project.org/Vignettes/src/lattice-intro/lattice-intro.pdf