Epi Lab color code

Software/Packages/Add-ins required

Software/Packages/Add-ins recommended

Description text

R code to copy/paste into console

R code to copy/paste into console that needs adjustment to your personal workspace

Websites where you can download requirements

Lab #10 requirements

- R - http://cran.r-project.org/bin/windows/base/
- R Studio - www.rstudio.com/ide/download/desktop
- Internet connection
- R packages "lattice"

Lattice graphs

JDG

Lattice/trellis graphs have always been something that has interested me. They are useful in depicting data of varying types over time and analysing trends visually. We have tried to do this using panel graphs in MSExcel® in the past (http://www.elsenburg.com/vetepi/epireport_pdf/April2013.pdf) but found that, while useful, the data manipulation prior to getting a final product just took too long. In R, however, there is a package called "lattice" which can perform this visualisation with relative ease. While some data manipulation prior to starting this visualisation is necessary you will see that it is quite simple to do.

In this example we take the African horse sickness cases that have occurred in our AHS control zones in the Western Cape and plot the epidemic curves of these outbreaks in a lattice fashion.

Functions and Code covered - Lab 10

barchart in the lattice package, **print** function in the R base package with the use of the **more = T/F** for multiple plots in the same graphic as well as the **panel.width/height** function for multiple plot manipulation.

The code

```
ahsdata <- read.csv("http://www.jdata.co.za/backpagelabs/backpagelabs_jdg_latticeahs.csv")
#Lets just briefly look at the data that is imported
head(ahsdata)
summary(ahsdata)
View(ahsdata)
```

#We have 3 columns of data: The outbreak year/reference, the case week and then the number of cases associated with that case week. This is enough information to make an epidemic curve. Instead of using the `epicurve.weeks` function in the `epitools` package as discussed in the 1st and 7th back page labs (see www.elsenburg.com – Epi Labs) we are using a more standard bar plot but we'll be using it within the `lattice` package. So install the lattice package with its dependencies.

```
install.packages("lattice", dependencies = TRUE)
#Activate the lattice package
library("lattice")
```

#Lets look at the very basic `bar chart` type lattice graph with minimum manipulation. Here we want to plot the number of AHS cases per case week with a panel per outbreak year.

A

```
barchart(ahsdata$Count~ahsdata$CaseWeek | ahsdata$Outbreak)
```

#So you'll see the plot is a side plot which is not something that I think is as helpful as a normal epidemic curve format, so let's sort that out:

B

```
barchart(ahsdata$Count~ahsdata$CaseWeek | ahsdata$Outbreak,
         horizontal = FALSE
         )
```

#The layout is automatically forced to 2 panels across and 3 panels high. We have 6 outbreak years so this works out well for us…you can use the layout function to manipulate this, we'll change it back but check how this works out

C

```
barchart(ahsdata$Count~ahsdata$CaseWeek | ahsdata$Outbreak,
         horizontal = FALSE,
         layout = c(6,1)
        )
```

#While you won't add a main heading in a publication you'd certainly add axis labels…I've put both here for completeness sake…I've also gone back to 2 panels across and 3 high

D

```
barchart(ahsdata$Count~ahsdata$CaseWeek | ahsdata$Outbreak,
         horizontal = FALSE,
         layout = c(2,3),
         main="AHS Outbreak Epidemic curves\n 1999 - 2014",
         ylab="Number of cases",
         xlab="Week of Outbreak"
        )
```

#Note above the "/n" in the main title adds a new line.

If you want to change font sizes then for your labels you can use the list function so set parameters for the labels – here the `cex` option changes fonts as you wish – try a couple of options to see how it changes

E

```
barchart(ahsdata$Count~ahsdata$CaseWeek | ahsdata$Outbreak,
         horizontal = FALSE,
         layout = c(2,3),
         main=list("AHS Outbreak Epidemic curves\n 1999 - 2014", cex=3),
         ylab=list("Number of cases", cex=2),
         xlab=list("Week of Outbreak", cex=2),
         )
```

#Since its an outbreak I think red is a better colour than the default light blue. Also you may have noticed that the y-axis was starting just below 0… here we set the limits of the Y axis (ylim) to force it to zero.

F

```
barchart(ahsdata$Count~ahsdata$CaseWeek | ahsdata$Outbreak,
         horizontal = FALSE,
         layout = c(2,3),
         main=list("AHS Outbreak Epidemic curves\n 1999 - 2014", cex=3),
         ylab=list("Number of cases", cex=2),
         xlab=list("Week of Outbreak", cex=2),
         col="red",
         ylim=c(0,30)
        )
```

#You can also manipulate the scales in terms of where they are labelled. I've set it to show the Y axis labels on all axes and the x axis labels just on the bottom axis: try some options between 0-3 to see how it impacts the labels. This graph is pretty much complete for this example.

```
barchart(ahsdata$Count~ahsdata$CaseWeek |
ahsdata$Outbreak,
         horizontal = FALSE,
         layout = c(2,3),
         main=list("AHS Outbreak Epidemic curves\n 1999
- 2014", cex=3),
         ylab=list("Number of cases", cex=2),
         xlab=list("Week of Outbreak", cex=2),
         col="red",
         ylim=c(0,30),
         scales = list(y=list(alternating ="3"),
                       x=list(alternating = "1")
                       )
          )
```

G

#To get slightly more involved on my computer the panels seemed a little too wide, so that if I showed 6 curves on top of each other (even if I change the size of my plotting window in R Studio) there was little to be seen because of the severe width. Below I use the `print` function to force the plot to be made in a specific part of the plot window and with specific dimensions per panel. Your computer will be slightly different so play around till you get something that works. To start with I need to make the graph in a format that works – below I've changed the format to 6 panels on top of each other, the y labels are only on the left, the width of the bars are changed a bit (`box.width`) and I've removed the title.

```
barchart(ahsdata$Count~ahsdata$CaseWeek |
ahsdata$Outbreak,
         horizontal = FALSE,
         layout = c(1,6), box.width = 0.8,
         ylab=list("Number of cases", cex=1),
         xlab=list("Week of Outbreak", cex=1),
         col="red",
         ylim=c(0,30),
         scales = list(y=list(alternating ="1"),
                       x=list(alternating = "1")
         )
)
```

H

#Now to manipulate the dimensions. Firstly for the `print` function its easier to assign the entire chart to a value list as shown below. We call this value list "`ahschart`"

```
ahschart<-barchart(ahsdata$Count~ahsdata$CaseWeek |
ahsdata$Outbreak,
         horizontal = FALSE,
         layout = c(1,6), box.width = 0.8,
         ylab=list("Number of cases", cex=1),
         xlab=list("Week of Outbreak", cex=1),
         col="red",
         ylim=c(0,30),
         scales = list(y=list(alternating ="1"),
                       x=list(alternating = "1")
         )
)
```

#Now we `print` the `ahschart` using the `panel.height` and `panel.width` function to make it look better. In my case it was better to make each panel 2 cm high and 10 cm wide (my RStudio window was set to its maximum height and about half the screen width. You can play with it until you are comfortable with the result , which you can then export out of the RStudio plot window.

```
print(ahschart, panel.height=list(2, "cm"),
                panel.width=list(10,"cm")
      )
```

I

#for fun try this : you can manipulate the starting position of your plots so that you can essentially put multiple plots at different parts of the plot area. To plot multiple plots you need to add a "`MORE = T`" to your print string until you get to the last one you plot which is "`more = F`"
I make another 2 plots of the same data but in different depictions - `ahschart2` and `ahschart3`

```
ahschart2<-barchart(ahsdata$Count~ahsdata$CaseWeek |
ahsdata$Outbreak,
                    horizontal = FALSE,
                    layout = c(2,3), box.width = 0.8,
                    ylab=list("Number of cases", cex=1),
                    xlab=list("Week of Outbreak", cex=1),
                    col="red",
                    ylim=c(0,30),
                    scales = list(y=list(alternating
                                         ="1"),
                                  x=list(alternating =
                                         "1")
                                  )
                )

ahschart3<-barchart(ahsdata$Count~ahsdata$CaseWeek |
ahsdata$Outbreak,
                     horizontal = FALSE,
                    layout = c(6,1), box.width = 0.8,
                    ylab=list("Number of cases", cex=1),
                    xlab=list("Week of Outbreak", cex=1),
                    col="red",
                    ylim=c(0,30),
                    scales = list(y=list(alternating
                                         ="1"),
                                  x=list(alternating =
                                         "1")
                                  )
                )
```

#Now I want to plot all three plots…

```
print(ahschart,position = c(0.3,1,0,0),
panel.height=list(2, "cm"), panel.width=list(4,"cm"),
more = T)

print(ahschart2,position = c(1.3,1.2,0,0),
panel.height=list(2.5, "cm"), panel.width=list(5,"cm"),
more = T)

print(ahschart3,position = c(1.3,0.35,0,0),
panel.height=list(2, "cm"), panel.width=list
(2.5,"cm"), more = F)
```

J

#Mine came out as shown on the results page in this example (again – my RStudio plot window was at its highest height…manipulate your position and panel heights and widths to try get it similar.)

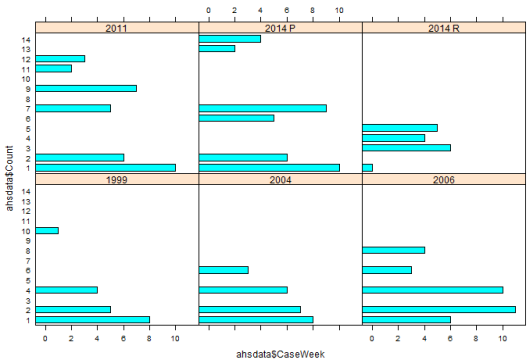
References

R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/

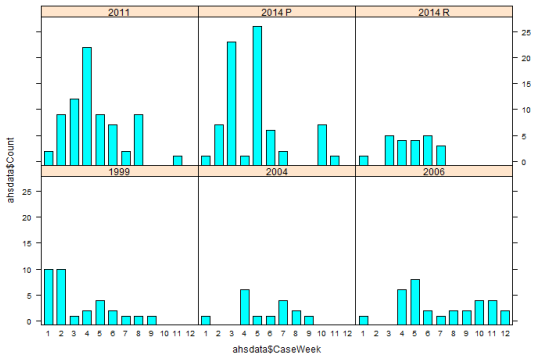
Sarkar, Deepayan (2008) Lattice: Multivariate Data Visualization with R. Springer, New York. ISBN 978-0-387-75968-5

The result

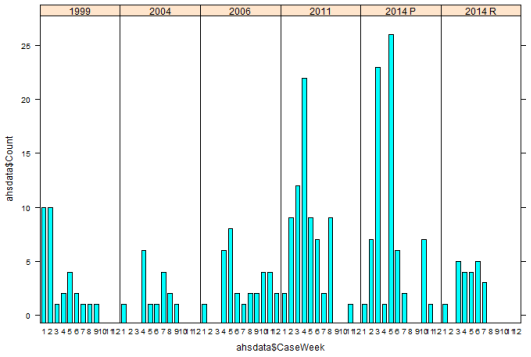
A



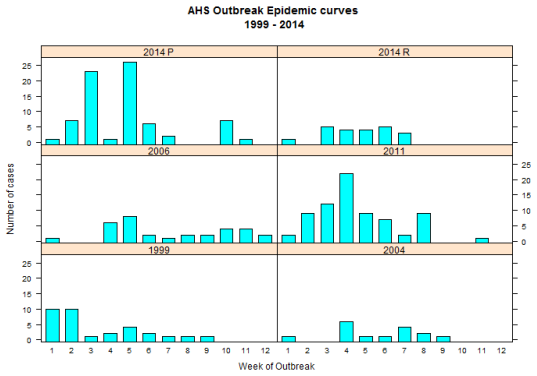
B



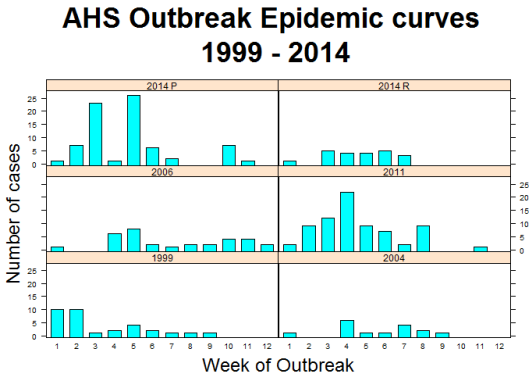
C



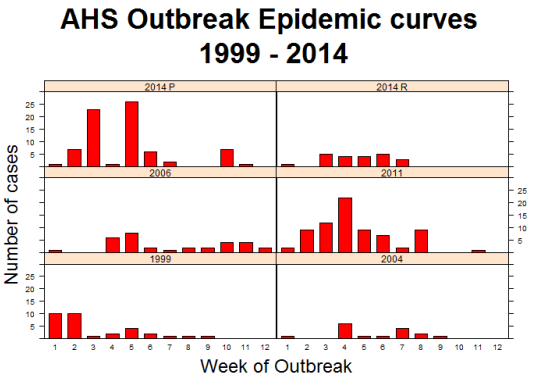
D



E

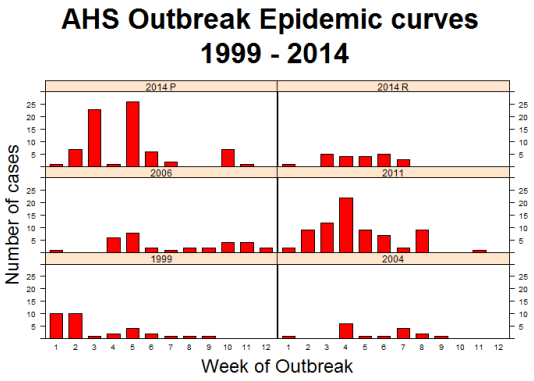


F

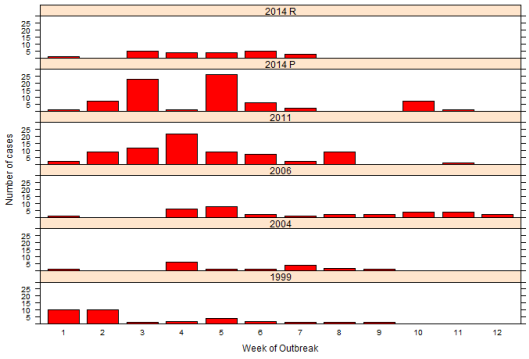


The result cont...

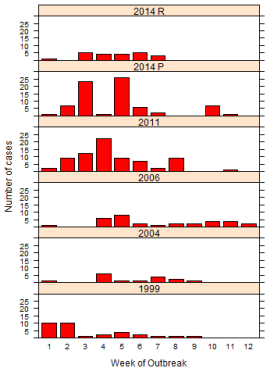
G



H



I



J