GRAPHING DATA II

Last lab you made plots of copter data, using basic **plot**, **qplot**, **ggplot**, and the **cowplot** controller of ggplot2. Here we move on to the **lattice** package, which makes grids of plots so you can compare multiple variables. This "trellis graph" is especially useful when you first want to squint at data to see what pops out, or when you want to prepare a complex array of graphs for publication (e.g., a 2X3 set of graphs). Other packages do this too (e.g., cowplot), but lattice is made for this. In addition, interaction plots are great way to see how one variable affects another. We use our copter data again, plus a large data set that comes with R.

PRELIMINARIES:

You will likely want to have these links open in a web browser [click on each link to open]:

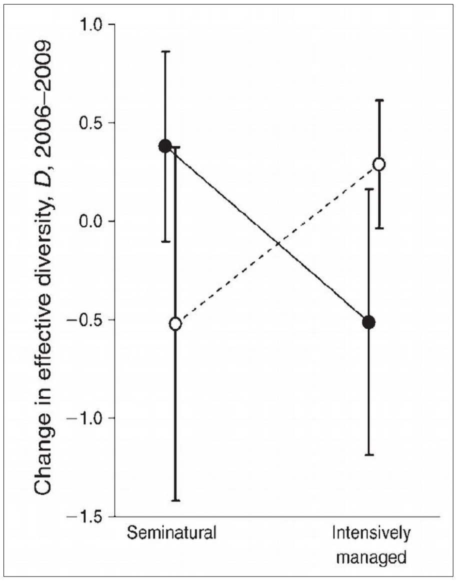
- Lattice Graphs
- Simple interaction plots

Start RStudio, set your Working Directory, import the copter data at:
http://jenkins.cos.ucf.edu/wordpress/wp-content/uploads/copter-data-F16.csv

Here we assume you called it "data" (because we are so creative...). This time we do not attach because we use a few data sets here, and we can specify them each time. Also, notice, that instructions, etc. are comment lines below, as if all this was already in R code.

```
# It will be easier if we designate our group IDs, Designs, and Fold as categorical factors:

fgroup <- factor(data$Group)
fdesign <- factor(data$Design)
ffold <- factor(data$Fold)
fwing <- factor(data$Wing)

# The lattice package should already be installed  - check out packages, and scroll down
# below the System Library heading to look for "lattice." It may even already be checked.
# If not, check it now.

demo(lattice) # to see some examples of cool graphs you could make with the
lattice package

# Let's try one of those for our copter data - a density plot, which shows a histogram.
# We'll use Time data:

densityplot(~data$Time)

# Our data (overall - not per treatment!): are they normal? This is actually
# NOT relevant for statistics, but you can now see how alter that command to
# show "bell curves" per design - which IS relevant for stats

densityplot(~data$Time | fdesign)

# This shows "bell curves" per categorical design, using | to say "for each"
# Is each data set "normal-ish"?
# Now let's stack these vertically:

densityplot(~data$Time | fdesign, layout=c(1,8))

# Was Time displaced with designs?

# Or how about scatter plots per each combination of Wing Length and Group?

xyplot(data$Time ~ data$Wing | fgroup, layout=c(2,3))

# Does the effect of Wing on Time look ~ consistent among groups?

# Ooh! Ooh! how about a 3D scatter plot of Time ~ Wing, Step, and Fold?

cloud(data$Time~data$Wing*data$Step | data$ffold)

# you get the idea - you have lots of options in lattice

# Perhaps more useful will be grids of multiple variable - great for first
# squinting at data to see patterns, correlations, etc.
# Here we have to list columns of interest (yup by counting across, or just listing all).
# For example:

splom(data[c(1:6)])

# The first row is most relevant: Time is the Y-axis there.

# What if we had more complex data, like the mtcars data set in R?
# This has a bunch of measures of 1993 cars.
# To see the first few rows in that data set,type

head(mtcars) # note that car names are row labels, not a first column

# Now let's make a big grid of scatter plots for all those data.
# If you wanted to predict mpg, which variables might be most useful?

# So winnow that list down to the best apparent predictors of mpg and make a smaller scatter
# plot grid.

# Do some of these potential predictors look closely correlated themselves (i.e. are they
# redundant)? If yes, this is called collinearity - something to avoid. You may not need 11
# predictors (or even 5) to predict mpg well, especially if some predictors are redundant.
# Seeing data is the first step toward effective stats modeling.

# Or imagine you have data to explore and you seek patterns. A classic data set is for irises
# (the flowers), where lengths and widths were measured for sepals and petals in 3 species.

# Check out the iris data set with head(iris).
# Then make some lattice plots to explore those data.
```

Interaction plots help understand interactive effects between experimental treatments. For example, the figure to the right is from a paper from Shawn Kelly's (UCF Bug Closet) thesis, where he found changes in beetle diversity due to prescribed fire depended on the land use; unburned sites increased in diversity with intensive management (fertilizer, etc.), but burned sites decreased in diversity with the same management. Those opposite effects of fire and management is an interaction between those treatments: one treatment changed the effect of the other treatment. Get it? The graph helped, right?



This is important for factorial experiments, and even sampling designs across complex landscapes, where factors may counter-act each other to yield complex outcomes.

Let's explore the possible interaction between Wing length and Folding in our copter data. Because this works with experiments, it assumes treatments are categories. We already made categorical treatments with the factor commands at the start.

```
# But first we have to remove NAs from the data (to avoid weird graphs). So do this:

cleandata <- na.omit(data)

# Now we can make an interaction plot:

interaction.plot(cleandata$Fold,cleandata$Wing,cleandata$Time,type="b")

# Did wing folding and wing length interact to alter each other's effects?
# What if you replace Fold with other factors?

Now let's try some more complex data.

interaction.plot(mtcars$am,mtcars$cyl,mtcars$mpg,type="b")

# meh. Still not a strong interaction; can you find one in the mtcars data set?
```

And don't forget the graph contest:
"best" gets a pass for up to 3 days late on a homework assignment