GRAPHING OUR HELICOPTER DATA I

Last week you generated helicopter flight times for the experiment designed by the class. Here we will use helicopter data to make boxplots, which compare categories well. For continuous predictors, we will also use scatter plots.

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

- Charts in R
- Plot: generic X-Y plotting
- qplot
- ggplot2 cheatsheet at RStudio
- Cowplot intro

Actual commands you type are shown below in `this font`. Mouse-based actions through drop-down windows are indicated with … (e.g., File...).

Start RStudio – if a prior session was saved, you may need to close the R script and/or data files (click the X on tabs at the top of the upper left, and click on the broom icon under Environment)

Set your working directory – Session... Set Working Directory... Choose Directory
And select the Desktop folder (for today's in-class work)

Import Dataset – In the Environment tab of the upper right window, click on Import Dataset...Desktop...From Web URL and then copy-past this link in:
`http://jenkins.cos.ucf.edu/wordpress/wp-content/uploads/copter-data-F16.csv`

Here you can change the name of the input data matrix (for convenience here we call it "data" below). Also ensure that Heading is Yes. Click **Import** (bottom right of window). You should see the data in the upper left window.

Attach data – this tells R the data set is the default (vs. loading multiple data sets to work with).
Type:

`attach(data) #` and hit Enter.

NOTE: You have now set the working directory, imported and renamed a data set, and told R to work with that data set. As before, make a script file (File...New File...R Script) that records the commands below that you like, with comments in English flagged with a #. Save that script file, email it to yourself, etc. to start keeping sets of instructions you can re-use.

This will help a lot in the future, because we won't tell you again how to do the steps above.

BOXPLOTS: To compare medians and quartiles of groups
First let's squint at the data by treatment categories using boxplots
Type:

```
boxplot(Time ~ Design)
```
 # This specifies one boxplot per each level of Design.

What do you see in the data?

Now add axis labels by typing:

```
boxplot(Time ~ Design, xlab='Helicopter Treatments', ylab='Flight Times
(sec)')
```
 # where xlab and ylab specify x- and y-axis labels

What can you infer about the helicopter experiment already?

Also try:

```
plot(Design, Time)
```
 # for a simple scatter plot

This specifies X then Y axes. If design was alphanumeric (A1, etc.), R would automatically make a boxplot based on categories. Instead, Designs are numeric so R made a scatterplot. You can already glimpse how the simple plot command can be quite versatile.

Now let's instead convert designs (numbers) into categories, or factors. **Type:**

```
fdesign <- factor(Design)
```

This made a new variable called “fdesign” that converted continuous design numbers into categories, or factors. *This will also be helpful in later labs*. Now watch what happens if you repeat the most recent plot command above but substitute `fdesign` for `Design`. Go ahead – you know how...

Now we'll try ***ggplot2*** – a graphing package that can get fancy [read "more complex commands"]. Actually ggplot2 has two levels of "fancy": quickplots and full fancy. Click the Packages tab in the bottom right window, scroll down and see if ggplot2 is already there. If so, click the box next to ggplot2. You just loaded (i.e., made active) the ggplot2 package. If ggplot2 is not listed, then click on the Install button (just above the list and type `ggplot2` in the pop-up box. Once it is all installed (it may take a minute or 2), click on the box next to ggplot2 in the list.

Lets first try a quick plot (`qplot`) in the ggplot2 package. **Try this command:**

```
qplot(Design, Time, data=data, group=Design, geom=c("boxplot", "jitter"),
main="Copter Flight Times", xlab="Designs", ylab="Flight Time (sec)")
```

Not bad, huh? **Play with options** in there to see what happens. For example, take out "jitter". And try some of the qplot tricks at the link listed at the start (above)

Now we go with full featured ggplot2 – which seems to dominate graphing in R. Lotta options [read lotta command possbilities]. Always look at examples in R Help, and check out the ggplot2 cheatsheet linked near the top here.

Type this command:

```
ggplot(data = data, aes(x=Design, y=Time, group=Design)) + geom_boxplot()
# where aes specifies basic aesthetics, and geom_boxplot calls for that form
# there are LOTS more details available to make plots fancy – see links above
```

Here is a more full version, showing all the defaults, which you can edit:

```
ggplot(data = data, aes(x=Design, y=Time, group=Design)) + geom_boxplot(stat =
"boxplot", position = "dodge", outlier.colour = "grey", outlier.shape = 19,
outlier.size = 1.5, outlier.stroke = 0.5, notch = T, notchwidth = 0.5,
varwidth = FALSE, na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)
```

Try changing some of the parameters inside geom_boxplot(___)

One final approach: **install and load the package cowplot**. Much advice abounds about elegant graphs, but Edward Tufte advises: strive to **maximize the data/ink ratio**. For example, the gray background in ggplot2 does not represent data. Also, axis labels are too small for publishing. Cowplot aims to make it easy to graph well; once you load the cowplot package, it essentially takes over defaults for ggplot2. So now re-run the above ggplot command and compare.
To turn off cowplot for ggplot2 for runs below, just click the cowplot box again to remove the X. And cowplot does not affect simple plot commands (used below).

Save your favorite boxplot so far: In the Plots tab (lower right window), **click on Export...** and follow choices to save a graph of your choice on the Desktop.

SCATTER PLOTS - use to compare bivariate, continuous data or compare raw values of categorical data.

Type:

```
plot(Time)   # This simply graphs Time as a function of row number, which is not as
    informative as

plot(Group,Time)   # This plots groups on the X axis, and flight time on the Y axis, or
plot(Time~Group)   # Plots Time "as a function of" Group: Is there a trend?
```

Type:
```
plot(Time~Design)
```

or to look more fancy, you could enter

```
ggplot(data = data, aes(x=Design, y=Time, colour = factor(Fold))) +
geom_point(size=1.5)
```

Do you think designs differed in helicopter times? Does a boxplot of the same comparison tell you anything different?

Type:
```
designtimes <- lm(Time~Design)
```

This computes a linear regression model (lm = linear model) of time as a function of design, to test to see if designs had different helicopter times. Note that designs here are handled as continuous data. We will later analyze this better.

Type:
```
summary(designtimes)
```
to see the results of that regression. Did design significantly affect flight time? How can you tell from the output table? We'll help you read this if you don't know...

Now **arrow up** and **click Enter** to re-run `plot(Time~Design)`
Then **type:**
```
abline(designtimes)
```

You should see the regression line on the basic scatter plot. Abline makes a simple linear regression, based on the designtimes linear model already specified, and superimposes it on a scatterplot also already specified. Abline has to immediately follow a `plot` command to work.

Does the regression line support your conclusion from the output table?

Now let's try this a different way. **Type:**
```
fdesigntimes <- lm(Time~fdesign)
summary(fdesigntimes)
```
Do you get a different answer? Why?
What does this tell you about how we should treat designs?

<u>And Now... A Contest</u>! Make the best looking graph of ***our copter data*** that you can. We will pick a winner, where the "best" graph most clearly conveys the most information and is visually-appealing (colors are welcome – use web resources for guides).

Winner gets one free pass to turn in a homework up to 3 days late.