# RDAV Tutorial: Hands-on with R on Nautilus

`http://rdav.nics.tennessee.edu/`

The data that we are using for one of the examples today comes from:
`http://iridl.ldeo.columbia.edu/SOURCES/.CARTON-GIESE/.SODA/.v1p2/`

You don't need to download the data for today's session on Nautilus, as I already have it saved on Nautilus as `/gpfs/medusa/szczepa/tutorial/ssh_global.cdf`.

Before we can use R, we need to set some things up.

1. Connect to Nautilus (if you have not yet done so):

   ```
   ssh -Y username@login.nautilus.nics.utk.edu
   ```

   Windows users who will be using a VNC session should leave off the `-Y` and follow the directions from the other handout.

   When prompted for your `PASSCODE`: enter your PIN and the number on the password token.

2. We will set up some directories for our work today.

   ```
   mkdir images
   mkdir scripts
   mkdir rscripts
   ```

3. We need to make local copies of a few scripts

   ```
   nautilus:> cp /gpfs/medusa/szczepa/tutorial/makegraph2.R ~/rscripts/makegraph2.R
   nautilus:> cp /gpfs/medusa/szczepa/tutorial/makegraph3.R ~/rscripts/makegraph3.R
   nautilus:> cp /gpfs/medusa/szczepa/tutorial/myscript ~/scripts/myscript.pbs
   ```

Once we have a VNC session to Nautilus, we are ready to launch R and run an interactive R session on Nautilus. All data analysis and visualization done on Nautilus needs to be submitted to the job queue. That is our next step.

4. You can see the status of the queue with the `showq` command.

5. Submit a request to the analysis queue asking for an interactive job, with X-forwarding, using one processor and 4 GB of memory for an hour, charging to the account UT-NTNLEDU:

   ```
   qsub -A UT-NTNLEDU -lncpus=1 -lmem=4GB -lwalltime=1:00:00 -q analysis -X -I
   ```

6. Once your interactive session starts, we need to make sure we have everything we need loaded and then launch R:

   ```
   nautilus:> module unload PE-intel
   nautilus:> module load PE-gnu
   nautilus:> module load netcdf
   nautilus:> module load r
   nautilus:> R
   ```

7. We will now run an interactive session in R. The first data set that we'll work with describes some students who took science classes; it is stored as a spreadsheet. (Note: This is randomly generated data.) This data will be represented in R as a **data frame**, which is an extremely common format within R.

   ```
   > studentdata <- read.csv("/gpfs/medusa/szczepa/tutorial/studentgrades.csv")
   > ls()
   > head(studentdata)
   > tail(studentdata)
   > str(studentdata)
   ```

```
> summary(studentdata)
> studentdata$SUCCESS <- as.factor(studentdata$SUCCESS)
> str(studentdata)
> summary(studentdata$SUCCESS)
```

8. We've used several commands. How can we find out about which commands to use, what they do, and how to use them? R has an extensive built-in help system.

```
> ?summary
> ??summary
> example(summary)
> example(barplot)
```

9. Let's make a few graphs using a package called ggplot2 that uses the ideas from *The Grammar of Graphics*. For more information about this style of graphing, see:
   - *The Grammar of Graphics* by Leland Wilkinson
   - *ggplot2: Elegant Graphics for Data Analysis* by Hadley Wickham

```
> library(ggplot2)
> qplot(COURSE, data=studentdata)
> qplot(SUCCESS, data=studentdata)
> qplot(GENDER, data=studentdata)
> qplot(SAT.MATH, data=studentdata)
> qplot(HS.GPA, data=studentdata)
> qplot(COURSE, HS.GPA, data=studentdata)
> qplot(COURSE, HS.GPA, data=studentdata, geom="jitter")
> qplot(COURSE, HS.GPA, data=studentdata, geom="boxplot")
> qplot(SUCCESS, SAT.MATH, data=studentdata, geom="jitter", facets = GENDER ~ COURSE)
> qplot(GENDER, SAT.MATH, data=studentdata, geom="jitter", facets = SUCCESS ~ COURSE)
> qplot(GENDER, SAT.MATH, data=studentdata, geom="jitter", facets = SUCCESS ~ COURSE, color=HS.GPA
```

We will next open a NetCDF file of sea surface height and make some graphs based on the data. We will save some graphs as files and copy them off Nautilus with scp. Because R doesn't have a built-in reader for NetCDF files, we need to tell R to load the package ncdf. Once we've done that we can open the file, assign it a name, and extract the data from the file. For more information about getting data into R and working with NetCDF files in R, see the following resources:

- http://cran.r-project.org/doc/manuals/R-data.html
- http://www.image.ucar.edu/GSP/Software/Netcdf/

The data that we are using today comes from
http://iridl.ldeo.columbia.edu/SOURCES/.CARTON-GIESE/.SODA/.v1p2/

10. We now load the data:

```
> library(ncdf)
> mydata <- open.ncdf("/gpfs/medusa/szczepa/tutorial/ssh_global.cdf")
> ls()
> print(mydata)
> x <- get.var.ncdf(mydata, "lon")
> y <- get.var.ncdf(mydata, "lat")
> ti <- get.var.ncdf(mydata, "time")
> seasurface <- get.var.ncdf(mydata, "ssh")
> ls()
```

We're now ready to examine the data.

```
> str(x)
> str(seasurface)
> summary(x)
> summary(seasurface)
> attributes(mydata)
> attributes(x)
> attributes(y)
> attributes(ti)
> attributes(seasurface)
> class(x)
> class(seasurface)
> x[50]
> seasurface[50,50,50]
```

11. We can make a histogram of sea surface height. We will first look at the help file for the `hist()` command that draws the histogram.

    ```
    > ?hist
    > hist(seasurface)
    > hist(seasurface, main="This is the Title of my Histogram")
    > hist(seasurface, main="This is the Title of my Histogram", xlab = "This is the
      label of my x variable")
    ```

    More interestingly, we can make a graphic that uses colors to show the sea surface at all the latitudes and longitudes in our data. Since the data represents 528 time steps, we will pick one of them for graphing the sea surface height.

    ```
    > zslice <- seasurface[,,1]
    > filled.countour(x, y, zslice, color.palette=heat.colors, asp=1)
    ```

12. These graphs have all been displayed by Nautilus at screen resolution. We can tell R that we want it to save the graphs as a file. We'll do an example saving the filled contour graph a PDF file. We can use the `pdf` command to tell R to save the graphics output to a PDF file instead of displaying it on the screen. There is also a command for `png` and for other graphics formats.

    ```
    > pdf("~/images/filledcontour1.pdf")
    > filled.contour(x, y, zslice, color.palette=heat.colors, asp=1)
    > dev.off()
    ```

13. We can use `scp` to copy this graph back to our local laptop:

    ```
    localhost% scp username@nautilus.nics.utk.edu:~/images/filledcountour1.pdf
    local-filename
    ```

    Enter your *PASSCODE* when prompted.

If we already know what we want to do, we don't need to work interactively with R from its command line. It can be a lot more efficient to submit commands with `Rscript`. In our next example we will also see how to change the proportions of the graph so that it is shorter and wider.

14. Quit the R interactive session. You do not want to save the workspace.

    ```
    > quit()
    ```

    You can also use the command `quit(save="no")`

15. We will run the R script `makegraph2.R` that you copied over at the beginning of this session. First let's take a look at what the script does, and then we will run it with the `Rscript` command. Run the following commands on Nautilus:

    ```
    cd rscripts
    more makegraph2.R
    Rscript makegraph2.R
    ```

```
cd ..
ls images
```

Then give the following command on your own computer (Mac or Linux) or use an scp or sftp client on Windows to copy the file back to your own computer.

```
scp username@nautilus.nics.utk.edu:images/filledcountour2.pdf local-filename
```

We don't even need to be on an interactive session on Nautilus to do this. We can submit this request to the job queue via a script.

16. Exit from the interactive job on Nautilus.

    ```
    exit
    ```

17. Take a look at the scripts that we are going to run by giving these commands on Nautilus:

    ```
    more rscripts/makegraph3.R
    more scripts/myscript.pbs
    ```

18. Submit the script to the queue:

    ```
    cd scripts
    qsub -V myscript.pbs
    ```

19. You can keep track of the status of your job with `showq`. Once the job has run, you should see the new graph in your images directory. You can copy it to your local machine with `scp` as before:

    ```
    scp username@nautilus.nics.utk.edu:images/filledcountour3.png local-filename
    ```

20. Finally, disconnect from Nautilus. Mac and Linux users can just `exit`. Windows users who have a VNC session will have to close the VNC session (close all programs in the VNC WINDOW), kill the ssh-tunnel in PuTTY, and then tell the server that they're done (`vncserver -kill :DISPLAY_NUMBER` on Nautilus).