Lab 1 - Learning to use R and initial data exploration.

If you have never used Splus or R before, check out these texts and help pages;
For help with general computing and basic stats; Modern applied statistics with Splus (Venables and Ripley) is a good text. Phil Spector (`http://www.stat.berkeley.edu/users/spector`) has an on-line introduction to R and Splus. Other online tutorials are linked from the class homepage. For help with installing R on your computer, check out `http://cran.us.r-project.org/`. You can download R for windows, linux and unix at this site, as well as the add-on packages. There is an extensive manual text (pdf-file), though I would regard this as a reference, not a text to study.
This lab is meant to help you get started. I have included most of the commands you will need. Try to figure out the rest yourself by reading the help files and online tutorials.

1 Getting Started

START: Install R from the R-project.org website. To start R, find the R icon and click. This will open up R in your home directory. To be able to save your work, go to the file menu and scroll down to change directories. First create a new directory for the lab called e.g. LAB1. Then change the directory to this, or you can browse until you find the directory you want to work from. R will be running from the current directory.
HELP: You can always get information about a command by typing `help(command)` at the prompt. If you don't know the command name, try `help.search(''phrase'')`. If you have used Splus before, but not R, most command names are the same, but some are not which can be confusing.
EDITORS: For the projects, and the labs, it is best to use an editor so you don't have to retype the commands. You can always cut-and-paste into the R command window. An alternative is to write a series of commands in your editor of choice, and save the file as "file.r". You can execute the commands in the file by writing
`source(''file.r'')`
at the prompt.
PLOTS AND GRAPHICS: The graphics window will open automatically when you plot something. If you want to display multiple plots in the graphics window, use commands `par(mfrow=c(m,n))` or `split.screen(c(m,n))` to divide into m by n small graphics displays. Read the help files on these commands.
ENDING THE SESSION: You quit R by typing q() at the prompt. You can elect to save the workspace in which case all functions you've created during the lab will be available when you start a new session. The results and functions are stored in a file called .RData. To review the commands you issued in a session look at the file .Rhistory. When you start another session you can launch R from the directory of choice, or change to this directory once you've got R running. To access an old workspace you need to first make dot-files visible by changing the preference in the "My documents" display to show hidden files. In R, go to the file menu and load workspace. You can browse to the directory of choice and mark the .RData file you wish to load. Note, you can save workspaces this way too. Note also that you can load multiple workspaces to combine work from different directories and sessions.

2 Regression analysis of the larynx cancer data.

In this lab you will perform an initial exploration of the larynx cancer data (KM, chapter 1.8). You can find the data set under "labs" on the class homepage. The data set consists of 90 males diagnosed with cancer of the larynx. For each individual you get the time on study (time), the outcome status at the end of the study (dead=1, alive=0), the stage of the initial diagnosis (1-4 in order of seriousness from low to high), and the age of the patient at diagnosis (also the year of diagnosis).
In this lab, the focus is to get familiar with R primarily and also to perform some initial exploration of survival data.
Open up the data file `larynx.dat` in an editor of choice. Please take note of the data structure. To read the data into R, remove the header information at the top of the data set and preserve only the data.

```
larynx<-as.data.frame(scan("larynx.dat",what=list(stage=0,time=0,y1=0,y2=0,status=0)))
```

Look at the data in R by simply writing `larynx` at the prompt.
Now, if this was a regression type data and you didn't know about censoring (that some patients are still alive at the end of the study), what would you do? You would start by examining the relationship with survival time and the other variables... Please go ahead and plot survival time versus other variables and comment on your findings.
Example: `plot(larynx$y1,larynx$time,xlab="age",ylab="time")`.
Please note, to access a variable in a data set you use the $ sign. That is, `larynx$time` tells R to retrieve the time variable from the data set lynx. You can check the names of the variables that are available using the `names(larynx)` command.

Consider data transformations like `log()` and plot the transformed data. Comment on your findings. Is stage of tumor related to survival time? In what way? Is age of the patient at diagnosis related to survival time? In what way?
Example:

```
plot(larynx$stage,log(larynx$time),xlab="stage",ylab="time")
```

or `boxplot(log(larynx$time)~larynx$stage)`

Try fitting a simple regression model to the log of the survival times. As we discussed in class though, this ignores that some of the patients were still alive at the end of study.

```
model1<-lm(log(larynx$time)~larynx$stage)
boxplot(model1$residuals~larynx$stage)
```

Examine the boxplot of the residuals. Did the linear regression model explain the dependency of survival time on tumor stage?

Look at the summary of the regression fit:

```
print(summary(model1))
```

How much of the variability of survival time did stage explain (R-squared and adjusted R-squared). Is the relationship between survival and stage significant? Please comment.

3 How to handle survival data objects, and plotting survival curves.

3.1 Saving a plot

To save a plot as a postscript file to be used in a report, do the following;
```
postscript(‘‘figure1.ps’’)
the plotting commands go here
dev.off()
```
Alternatively, when the graphics window is active, simply go to the file menu and save the figure in the desired format. Make sure to save as a file and not print hard-copies when you compile a report though. I want the graphs to be included inside the report, not as an appendix.

3.2 Survival objects

To manipulate survival type data you need to install the package `survival` in R. Simply issue the command
`install.packages('survival')`
at the prompt. After following the instructions and installing the package, activate the library by issuing the command
`library(survival)`
at the prompt.
Descriptions of the library content are available at the R-project website or in the R helpfile system (`library(help='survival')`).

The function `Surv()` in R creates a survival data object, which records if the survival time is a time of event or a censored time. Look at the output of
`with(larynx,Surv(time,status))`
(The `with()` function is a handy way of telling R which data set or subset of a data set to apply a function to). Notice how R has added the + symbol to all censored observations.

The function `survfit()` estimates the survival function using the Kaplan-Meier method (see book and lecture notes). In this lab, you don't need to worry too much about the estimation procedure, but try to get some intuition form the graphical and numerical output.
```
sf.larynx<-survfit(Surv(larynx$time,larynx$status), conf.type="none")
summary(sf.larynx)
print(sf.larynx)
```

Please note that the `survfit()` function estimates the survival at each point where an event occurred (time), and also outputs the number of individuals at risk at these time points and number of events that occurred. Were there multiple deaths at any time point? The function also outputs the standard error of the survival estimate. We will talk more about this in class.

Let's look at the estimated survival curve.
`plot(sf.larynx,main="Larynx Tumor Survival")`
Comment on the shape of the survival curve. If the hazard rate is constant, the survival function is exponential. Does an exponential model look plausible here? Try plotting `log(survival)` against time - it should look like a line.
`plot(sf.larynx,main="Larynx Tumor Survival", fun=log)`
Comment on the above figure.

Let's compare the patients with different tumor stages.
`sf.larynx<-survfit(with(larynx,Surv(time,status))~larynx$stage,conf.type="none")`
`plot(sf.larynx,col=1:4,lty=1:4,main="Larynx Tumor Survival")`
Comment on the similarities and differences of survival for the different stages of tumor.

Summary

Please write up a short report outlining your work in this lab. Include only the relevant figures.