228.371 Computer Lab: Comparing means and simple linear regression

Semester One 2015 - Week 2

Instructions: Read each section and try the commands. Then try the Stream worksheet questions suggested to test your knowledge. The worksheet is "adaptive" which means if you get an answer wrong, you can try again. This quiz is to help you monitor your progress, it does not count toward your mark.

Note that because of fonts, especially for symbols like quotation marks, cutting and pasting commands from this document occasionally will not work - you may have to retype.

1 Comparing the mean between two independent samples

Download the file `house_selling_prices.txt` from our stream site and read it into `R`. For example, if you place the file in your working directory the command would be

```
houses <- read.table(file="house_selling_prices.txt", header=TRUE)
```

You can also use

```
houses <- read.table(file.choose(), header=TRUE)
```

to select the file interactively.

First, let's do a test to compare the prices of houses inside and outside the northwest quadrant of the city being studied. This is a particularly desirable quadrant - we use the alternative hypothesis that house prices there are *h*igher. We will also do a boxplot to test our assumptions.

```
t.test (houses$price[houses$Quadrant=='NW'],
         houses$price[houses$Quadrant!='NW'],
         alternative='greater')              # Square brackets subset the data
boxplot(houses$price ~ houses$NW)            # 1=northwest house, 0=else
```

The boxplot shows a few moderate outliers; however for the Northwest quadrant the sample size is fairly large, so these should not have too much influence on that sample mean. For the non-northwest quadrant, the outliers are on the high side - if anything we

would expect them to diminish the significance of the result. So a conclusion that there is a difference between the Northwest and other quadrants seems reasonable, even if the test assumptions are not perfectly satisfied.

The previous test assumed the variance was different between the two groups. Do the results change much if we assume equal variance? Try the command:

```
t.test(houses$price[houses$Quadrant=='NW'], houses$price[houses$Quadrant!='NW'],
        alternative='greater', var.equal=TRUE)
```

Complete Worksheet questions 1-2.

2 Paired data

A frog jumping club is trying to determine the optimal time to feed their frogs before a race. Some say 4 hours before the race is the optimal time, others 8 hours before. They perform an experiment to see. They take five of the club's frogs and race each of them twice (both races have the same length). For one race, a frog is fed four hours before, for the other, the frog is fed 8 hours before. They randomise which frogs get which treatment before which race. The two times for each frog, in seconds, are recorded below.

Fred	31	36
Ingo	24	29
Mona	33	33
LuLu	25	31
Tom	40	42

Note there are some fairly large differences between the frog (e.g. Tom is a lot slower) but the *differences* are similar. This is the point of the paired test, to control for innate individual differences and isolate the difference due to the different treatments. Read the data into R and perform the test with the option "paired=TRUE":

```
fed4 <-c(31,24,33,25,40)
fed8 <-c(36,29,33,31,42)
t.test (fed4, fed8, paired=TRUE)
```

What is your conclusion?

Try worksheet questions 3-4.

3 Simple Linear Regression

We will revisit the data set used in the first assignment. In that case, we considered the correlation, R^2, and fitted values for the slope and intercept. We will now consider making *inferences* (tests and confidence intervals) for the slope. Recall that we load the data with the command `data(trees)`. We will use `Volume` as the response variable (y variable). For now we will use `Girth` as the only predictor (x variable).

Regression in `R` uses the function `lm()` (`lm` stands for *linear model*). The response is listed first, followed by the symbol $\sim$. This symbol can be read "predicted by." We assign the output of the function to an object, in this case called `TreeModel`. For the next section (Prediction) it is important that we indicate the data set as an argument, rather than repeating it for each variable.

```
TreeModel <- lm(Volume~Girth, data=trees)
## Instead of TreeModel <- lm(trees$Volume~trees$Girth)
```

Looking at `TreeModel` gives the most basic information about the model. We can get more information using the `summary` function. Try the commands

```
TreeModel
summary(TreeModel)
```

The last column in the table of coefficients, `Pr(>|t|)`, contains the p-value for a test of whether the coefficient for that row is zero. Generally we only look at these for the second row (the slope). What is the conclusion in this case?

We can also construct a confidence interval for the slope and/or intercept using the standard error given in the second column. The multiplier is obtained from a t distribution with $n-2$ degrees of freedom (e.g. via the `qt` function). The degrees of freedom is also given in the summary information, on the line where it says `Residual standard error`.

The `TreeModel` object also contains the fitted values and residuals for each observation, which can be used to check our assumptions. What are we looking for in the following plots?

```
hist(TreeModel$resid)
qqnorm(TreeModel$resid)
plot(TreeModel$fitted, TreeModel$resid)
```

Complete Worksheet questions 5-6

4 Predicting with the Simple Linear Regression Model

The function `predict.lm()` can use a model object to give point predictions, prediction intervals, and confidence intervals at particular values of the predictor variables. (Confidence intervals contain the mean value at a particular x value at the specified confidence level; a $P\%$ prediction interval should contain $P\%$ of future *observations* at that x value. Both are centred at the point prediction.) The default predictions are at the x values used to fit the model. Recall the `trees` data used in the last lab:

```
data(trees)
TreeModel <- lm(Volume~Girth, data=trees)
predict.lm (TreeModel)
predict.lm (TreeModel, interval="confidence")
predict.lm (TreeModel, interval="prediction")
predict.lm (TreeModel, interval="prediction", conf.level=.9)
## Default confidence level is 95%
```

Alternately, the predictions can be for any values of x. These new values for prediction should be stored in a data frame where x is given the same variable name as in the original data set. For instance:

```
newdata <- data.frame(Girth=c(9,12,15,18,21))
predict.lm (TreeModel, newdata, interval="confidence")
```

Complete worksheet questions 7-8