Using R, try your hand at reproducing the analysis of variance of tables you worked with in problem #2. Let’s load the data.

```r
library(foreign)
prob3.URL <- "http://people.umass.edu/biep640w/datasets/larvae.dta"
dat <- read.dta(file = prob3.URL)
```

Let’s get information on the dataset using commands `str(...)` and `summary(...)`.

```r
str(dat)
```

```
## 'data.frame': 15 obs. of 4 variables:
## $ id  : num 1 2 3 4 5 6 7 8 9 10 ...
## $ y   : num 2.84 2.97 2.69 2.68 2.83 ...
## $ x1  : num 0.15 0.214 0.487 0.509 0.57 ...
## $ x2  : num 0.425 0.439 0.301 0.325 0.371 ...
## - attr(*, "datalabel")= chr "PubHlth 640 Unit 2 Regression - Larvae data"
## - attr(*, "time.stamp")= chr "10 Feb 2013 16:36"
## - attr(*, "formats")= chr "%9.0g" "%9.0g" "%9.0g" "%9.0g"
## - attr(*, "types")= int 254 254 254 254
## - attr(*, "val.labels")= chr "" "" "" ""
## - attr(*, "var.labels")= chr "larva id" "log10(survival)" "log10(dose)" "log10(weight)"
## - attr(*, "expansion.fields")=List of 2
##   ..$ : chr ":_dta" "note1" "Week 3 homework assignment exercises 2 and 3"
##   ..$ : chr ":_dta" "note0" "1"
## - attr(*, "version")= int 12
```

```r
summary(dat)
```

```
## id  y  x1  x2
## Min. :1.0 Min. :2.35 Min. :0.150 Min. :0.093
## 1st Qu.:4.5 1st Qu.:2.43 1st Qu.:0.539 1st Qu.:0.174
## Median :8.0 Median :2.45 Median :0.739 Median :0.289
## Mean :8.0 Mean :2.57 Mean :0.701 Mean :0.278
## 3rd Qu.:11.5 3rd Qu.:2.68 3rd Qu.:0.884 3rd Qu.:0.367
## Max. :15.0 Max. :2.97 Max. :1.194 Max. :0.439
```

The commands `str(...)` and `summary(...)` give quite a bit of information, which can be an overload. I always look for the number of observations, number and names of variables, as well as the types of variables. Are they factors? Are they numeric or character variables, etc.?

From the Stata handout, we see that its `regress` function gives a nice amount of information, including:

- ANOVA table (SS, df, MS)
- linear regression table (Coefficients, Std. Error, t value, P > |t| (p-values), and 95% CIs)
• F-statistic and $R^2$

Let’s find this information individually through R commands and then we will create a function to deliver us this information in a nicely formatted package tied with a bright and shiny red bow.

First, let us fit the model as such:

```r
fit1 <- lm(y ~ x1, data = dat)
```

By simply running `fit` into the Console, we are only given a preview of what we actually need.

We see the *formula* and the *coefficients*, but obviously we need more. If you recall from the last assignment, the `summary(...)` command is quite useful.

```r
summary(fit1)
```

```
##
## Call:
## lm(formula = y ~ x1, data = dat)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##-0.18413 -0.05148  0.00258  0.06728  0.18822
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.9522 0.07360 40.143 5.1e-15 ***
## x1         -0.5499 0.09725 -5.650 7.9e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.107 on 13 degrees of freedom
## Multiple R-squared:  0.711, Adjusted R-squared:  0.688
## F-statistic: 31.9 on 1 and 13 DF,  p-value: 7.94e-05
```

Again, we see the *formula* and the *coefficients*, but also the *Std. Error*, *t value*, *p-value*, and *F-statistic* and $R^2$. We are missing an *ANOVA table* and confidence intervals.
anova(fit1)

## Analysis of Variance Table
##
## Response: y
## Df Sum Sq Mean Sq F value Pr(>F)
## x1 1 0.363 0.363 31.9 7.9e-05 ***
## Residuals 13 0.148 0.011
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

confint(fit1)

## 2.5 % 97.5 %
## (Intercept)  2.7933 3.1111
## x1          -0.7601 -0.3396

With the three commands we just entered, we now have all of the information we need to replicate the answer of the Stata command `regress`

You should be able to replicate the output, but now using a model containing \textit{x2} alone.

fit2 <- \texttt{lm(y ~ x2, data = dat)}

summary(fit2)

##
## Call:
## lm(formula = y ~ x2, data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.1412 -0.1163  0.0417  0.0775  0.1774
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.185      0.082   26.64  9.9e-13 ***
## x2            1.376      0.275    5.01  0.00024 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.116 on 13 degrees of freedom
## Multiple R-squared: 0.659, Adjusted R-squared: 0.632
## F-statistic: 25.1 on 1 and 13 DF, p-value: 0.00024

anova(fit2)

## Analysis of Variance Table
##
## Response: y
## Df Sum Sq Mean Sq F value Pr(>F)
## x2 1 0.337 0.337 25.1 0.00024 ***
## Residuals 13 0.175 0.013
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
confint(fit2)

## 2.5 % 97.5 %
## (Intercept) 2.008  2.362
## x2       0.782  1.969

Lastly, let’s replicate the output using the model containing \( x_1 \) and \( x_2 \).

```r
fit3 <- lm(y ~ x1 + x2, data = dat)
summary(fit3)
```

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.589050   0.083626  30.970   <2e-16 ***
## x1          -0.378496   0.066406  -5.708   9.9e-05 ***
## x2           0.875000   0.172524   5.070  0.00027 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## Residual standard error: 0.0626 on 12 degrees of freedom
## Multiple R-squared: 0.908,  Adjusted R-squared: 0.893
## F-statistic: 59.2 on 2 and 12 DF,  p-value: 6.09e-07
```

```r
anova(fit3)
```

```
## Analysis of Variance Table
## Response: y
##            Df Sum Sq Mean Sq F value Pr(>F)
## x1         1 0.3630  0.3630  92.60 5.4e-07 ***
## x2         1 0.1010  0.1010  25.65 0.00027 ***
## Residuals 12 0.0470  0.0039
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
confint(fit3)
```

```
## 2.5 % 97.5 %
## (Intercept)  2.4068  2.7712
## x1         -0.5231 -0.2339
## x2         0.4992  1.2508
```
Bonus: Functions!

Now, this is something new, but it shouldn’t be too hard to follow along. We’re going to create a function to get us all the information that Stata’s `regress` command outputs.

```
# x will represent our fitted objects (fit1-fit3)
stata.regress <- function(x) {
  # syntax: function(x1, x2,...) followed by a bracket
  a <- summary(x)  # we are going to pass x to three commands as listed
  b <- anova(x)
  d <- confint(x)

  # the last line typically specifies what one wants the function to print
  print(a); print(b); print(d)
}

stata.regress(fit1)  # voila
```

```
## Call:
## lm(formula = y ~ x1, data = dat)
##
## Residuals:
##    Min   1Q Median   3Q    Max
## -0.18413 -0.05148  0.00258  0.06728  0.18822
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.9522     0.0736   40.14 5.1e-15 ***
## x1          -0.5499     0.0973   -5.65 7.9e-05 ***
## ---
## Signif. codes:  < 0.001 ***  0.001 **  0.01 *  0.05 .  1
##
## Residual standard error: 0.107 on 13 degrees of freedom
## Multiple R-squared: 0.711, Adjusted R-squared: 0.688
## F-statistic: 31.9 on 1 and 13 DF,  p-value: 7.94e-05
##
## Analysis of Variance Table
##
## Response: y
##            Df Sum Sq Mean Sq  F value Pr(>F)
## x1          1 0.363  0.363  31.9    7.9e-05 ***
## Residuals   13 0.148  0.011
## ---
## Signif. codes:  < 0.001 ***  0.001 **  0.01 *  0.05 .  1
## (Intercept) 2.7933  3.1111
## x1         -0.7601 -0.3396
```

This is not easy to display in a short amount of space, but I simply tried to create a function that mimics Stata’s `regress` command. After creating one function, most are not going to understand the syntax. It takes practice and that is why I am only giving a glimpse as to what functions are and how they work.

If you’d like to practice a few functions, try to recreate the `mean(...)` and `confint(...)` functions, but with different names. By naming your own created functions the same names as the base functions in R, things start to get screwy. Give it a shot, run both functions, and see if they match up.
PLEASE email me for any confusions as I try my best to answer back as quickly as I can.