Motivating the `uwIntroStats` Package and Using the `regress` Function

Author: Brian D. Williamson

Contents

Chapter 1

Introduction

Teaching statistical software in introductory courses has become ubiquitous in the last decade. As there is no universal standard language, each individual department or school must decide for themselves which software they will teach to students. A variety of factors have led the University of Washington to teach STATA to all students who enroll in Biostatistics and Epidemiology courses. STATA is a very comfortable language - commands are fairly intuitive, the workspace can only involve one data set at a time, and as a proprietary language the functions are all controlled and are guaranteed to work. However, many researchers use R to analyze data. As a open source language, R offers flexibility and breadth that STATA cannot. Also, the large variety of developers working on R packages allow a user to perform a multitude of tasks.

The motivation for the `uwIntrostats` package is to bridge the gap between these two software packages. This package was developed by Scott S. Emerson, M.D. Ph.D., Andrew J. Spieker, and Brian D. Williamson, a professor and two graduate students, respectively, in the University of Washington Department of Biostatistics. We developed functions for descriptive statistics, plotting, regression, and inference. Some of these functions are completely novel, while some add novel functionality onto existing R functions. All functions, however, were designed to be easy to use and intuitive, and display results in a manner consistent with what the authors deemed as the helpful parts of both STATA and R output.

In this document, we present example analyses of the `mri` data set and the `salary` data set, available at `http://emersonstatistics.com/datasets/mri.txt` and `http://emersonstatistics.com/datasets/salary.txt` respectively. We use the `regress` function both from the `uwIntroStats` package and from STATA, and compare the output from each. In each example, we will first motivate the analysis, and then present the STATA code and output followed by the R code and output. For a more comprehensive introduction to the R language, see `http://www.emersonstatistics.com/GeneralMaterials/R/IntroToR.pdf`.

Chapter 2

The regress Function

The name for the function in `uwIntroStats` was borrowed from its STATA equivalent. However, we have combined the functionality of the `lm`, `glm`, `coxph`, and `geeglm` functions to make this possible within one function in R. We have also added functionality to these R functions, like calculating robust standard error estimates and F statistics by default, and allowing the user to specify multiple-partial F-tests within the call to `regress`. The following examples are meant to show the power and flexibility of this function. For all analyses, we will be using robust standard error estimates to allow for unequal errors (in linear regression), to help protect against non-proportional hazards (in Cox regression), and to help protect against the mean-variance relationship in logistic regression. However, before we go on, we stress that all analyses must be prespecified in order to preserve the correct Type I error rate. The analyses presented below are for teaching purposes only.

2.1 Syntax for regress

In STATA, the syntax for `regress` is the following (taken from http://www.stata.com/manuals13/rregress.pdf):

```
regress depvar [indepvars] [if] [in] [weight] [, options]
```

Here `depvar` is the dependent (response) variable, and `[indepvars]` is an arbitrary list of independent variables. The other variables help to specify weights and other options (like subsetting, or robust standard error estimates). Similar syntax is followed in functions for logistic or Cox proportional hazards regression. Interactions can be specified using the `##` operator, as in `var1##var2`, in the `indepvars` string.

In R, the syntax is:

```
regress(functional, formula, data, options)
```

Here `functional` specifies the type of regression to perform. Table 2.1 displays the allowed functionals and their corresponding type of regression. Formula is an R formula that might be entered into any of the common regression functions in R, of the form

```
y ~ x1 + x2 + ...
```

R formulas can also specify interactions using the syntax `x1*x2`. The argument `data` allows the user to enter a data set and use this data set to find the variables entered in the formula. There are many options that `regress` takes; we invite the reader to look at the documentation for `regress` to see the exhaustive list. We will not be changing any of these options from their default values in this document.

Functional	Type of Regression
`"mean"`	Linear Regression
`"geometric mean"`	Linear Regression on Log Transformed Y
`"odds"`	Logistic Regression
`"rate"`	Poisson Regression
`"hazard"`	Cox Proportional Hazards Regression

Table 2.1: Functionals and Regression Types for `regress`.

2.2 Examples using the `mri` Data

There are many potentially interesting response variables in the `mri` data set, but for all of the following analyses besides those using a Cox proportional hazards model we will be using `atrophy` as our response. If we look at the documentation for this data set (available at http://emersonstatistics.com/datasets/mri.pdf), we note that determining whether brain atrophy measured by MRI represented some form of nervous disease, whether it was a sign of some other disease process, or whether it was just a result of the aging process.

Before we do any analysis, we first must read in the data. In STATA, this is done by running the command

```
infile ptid mridate age male race weight height packyrs yrsquit alcoh physact chf chd stroke
diabetes genhlth ldl alb crt plt sbp aai fev dsst atrophy whgrd numinf volinf obstime death
using "http://www.emersonstatistics.com/datasets/mri.txt"
```

However, this creates a row of missing data at the top, because the names are in the text file. Thus we drop the first row with

```
drop in 1
```

In R, the data set is supplied with the `uwIntroStats` package. Thus we simply have to load the package (and the packages it depends on) and then call in the data:

```
library(Exact)
library(geepack)
library(plyr)
library(sandwich)
library(survival)
library(uwIntroStats)
data(mri)
```

2.2.1 Example 1: `atrophy` on `age`

A natural question is this: is age associated with brain atrophy? To address this question, we can run a linear regression of age against atrophy as measured by MRI. The coefficient from this model will tell us the average increase (or decrease) in atrophy for each year increase in age. In STATA, we run the command

```
regress atrophy age, robust
```

and obtain the following output:

```
Linear regression                                      Number of obs =     735
                                                       F(  1,   733) =   60.12
                                                       Prob > F      =  0.0000
                                                       R-squared     =  0.0867
                                                       Root MSE      =  12.359

------------------------------------------------------------------------------
             |               Robust
     atrophy |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |   .6979831   .0900192     7.75   0.000      .521257    .8747093
       _cons |  -16.06213   6.700595    -2.40   0.017    -29.21677   -2.907482
------------------------------------------------------------------------------
```

We can glean a lot of information from this output. Above the table, we see that there are 735 observations in this data set. We also note that the overall F-statistic (here testing that the coefficients for the intercept and age are simultaneously zero) is 60.12. If we calculate the probability of seeing this value or more extreme on an F distribution with 1 and 733 degrees of freedom, we get a very small p-value (the third line down). We skip over R-squared and Root MSE.

Now we come to the table. In these data, we have people between ages 65 and 99. However, the intercept is interpreted as the average atrophy level for newborns, which is clearly outside of the range of our data. Therefore, we cannot interpret the intercept in a meaningful manner. If we wanted an interpretable intercept, we could reparametrize our model by subtracting the mean of the age data from each observation. This would place the intercept at the average age, and thus it would have meaning to us. Moving on to the coefficient for age, we see that it is 0.697. This means that for each 1 year increase in age, we expect to see an average increase of 0.697 in an individual's atrophy score. Based on the 95% confidence interval of an increase of between 0.52 and 0.87 in atrophy score for each one year increase in age computed by STATA, we reject the null hypothesis that age is not associated with atrophy in favor of the alternative hypothesis, that increasing age is associated with an increase in average atrophy score.

In R, we run

```
regress("mean", atrophy~age, data=mri)
```

and obtain the following output:

```
Call:
regress(fnctl = "mean", formula = atrophy ~ age, data = mri)

Residuals:
    Min      1Q  Median      3Q     Max
-36.870  -8.589  -0.870   7.666  51.203

Coefficients:
                 Estimate  Naive SE  Robust SE    95%L      95%H          F stat    df Pr(>F)
[1] Intercept     -16.06     6.256     6.701      -29.22    -2.907           5.75 1    0.0168
[2] age            0.6980   0.08368   0.09002      0.5213    0.8747         60.12 1  < 0.00005

Residual standard error: 12.36 on 733 degrees of freedom
Multiple R-squared:  0.08669,Adjusted R-squared:  0.08545
F-statistic: 60.12 on 1 and 733 DF,  p-value: 2.988e-14
```

Note that in this printout, the overall tests - Root MSE, which here is displayed as Residual standard error; R-squared, here Multiple R-squared and Adjusted R-squared; and the overall F test

- are all displayed at the bottom of the printout. We also now see the estimates and inference for the intercept followed by the slope. Last, the only other difference between the two tables is that R displays an F-statistic and p-value by default, while STATA displays a t-statistic and p-value. Other than that, all of the estimates and inference are the same.

2.2.2 Example 2: `atrophy` on `age` and `male`