Guide to Matrix Routines in Stata
By Kevin Sweeney
Spring, 2003

Getting Started

First of all, this entire discussion is drawn (basically) from the Stata User's Guide and Stata Programming manual. For folks who use Stata: there is nothing in the reference manuals about matrix routines.

You start by getting data. There are ***two*** ways to tell Stata you are going to be working with matrix algebra.

One more note before we get started: anything you need to type will be in *italics*; anything that Stata spits out will be in **courier bold**.

Inputting Data by Hand

The most labor intensive way to do matrix algebra in Stata is to input the matrices by hand. Since I am writing this explicitly for the folks in 686 (and I know your data sets are relatively small), this is a feasible way to go. (Not-so-subliminal message: Before doing anything open a log file!) If you do input data by hand, you type:

matrix define X = (2,7,4\3,1,5\6,4,3)

Notice that commas separate the row elements and the rows are separated by the backslash. Stata names your rows and columns rather unimaginatively, but these names are important to know for stuff you will do later on. Once you have type this matrix in, you can list it to see what it looks like so type:

matrix list X

And you get:

```
      X[3,3]
      c1  c2  c3
 r1    2   7   4
 r2    3   1   5
 r3    6   4   3
```

Stata tells you that matrix X_{nxk} is 3 x 3, and names the rows r1, r2,…r_n, and the columns c1, c2,…c_k.

Once you have defined a matrix X, and a conformable column vector Y, you will be ready to rock and roll. So, type:

matrix define Y = (1\2\3)
matrix list Y

And you get:

```
Y[3,1]
    c1
r1   1
r2   2
r3   3
```

Matrix Operations

The Usual Suspects

1. matrix *C =(A+B)* -- Addition
2. matrix *C =(A-B)* -- Subtraction
3. *matrix C =(A*B)* -- Multiplication
4. *matrix C =A/b* -- Division by a scalar

Operations Specific to Matrix Algebra

1. *matrix T =A'* --Transposition
2. matrix I =inv(A) --Inversion of Non-Singular Matrices

Other Useful Things – You need some programming to get some of these…

1. *matrix A = I(d)* --returns d x d identity matrix

2. *matrix define X = (2,7,4\3,1,5\6,4,3)*
local r = rowsof(X)
di `r' --returns number of rows in X

(Note: `r' is typed with the key next to the 1 on your keyboard, r, and single close quote.)

3. *matrix define X = (2,7,4\3,1,5\6,4,3)*
local c = rowsof(X)
di `c' --returns number of columns in X

4. *matrix define X = (2,7,4\3,1,5\6,4,3)*
matrix b = vecdiag(X) --returns the main diagonal of X as a row vector.
matrix c = b' --transposes the row vector into a column vector.

5. sqrt(x) --returns the square root of X.

6. *matrix define X = (2,7,4\3,1,5\6,4,3)*
local e1 = el(X,1,1) --obtains the first element from the first row of X
local e2 = el(X,2,1) --obtains the first element from the 2nd row of X
local e3 = el(X,3,1) --obtains the first element from the 3rd row of X
di `e1' + `e2' + `e3'--displays, in effect, the column sum from the 1st column from X.

There are many more operations and functions that may be of use. Consult the Stata Programming Manual for details.

Example

Suppose I had the same 3 x 3 matrix as in the SPSS example:

2	7	4
3	1	5
6	4	3

The following few commands are necessary to define the matrix, transpose it, find the inverse of the original matrix, and check to make sure the inverse is correct.

matrix define X = (2,7,4\3,1,5\6,4,3)

matrix define T = X'

matrix list T

```
T[3,3]
     r1  r2  r3
c1    2   3   6
c2    7   1   4
c3    4   5   3
```

matrix define I = inv(X)

matrix list I

```
I[3,3]
             r1           r2           r3
c1   -.12408759   -.03649635    .22627737
c2    .15328467   -.13138686    .01459854
c3    .04379562    .24817518   -.13868613
```

*matrix define C = X*I*

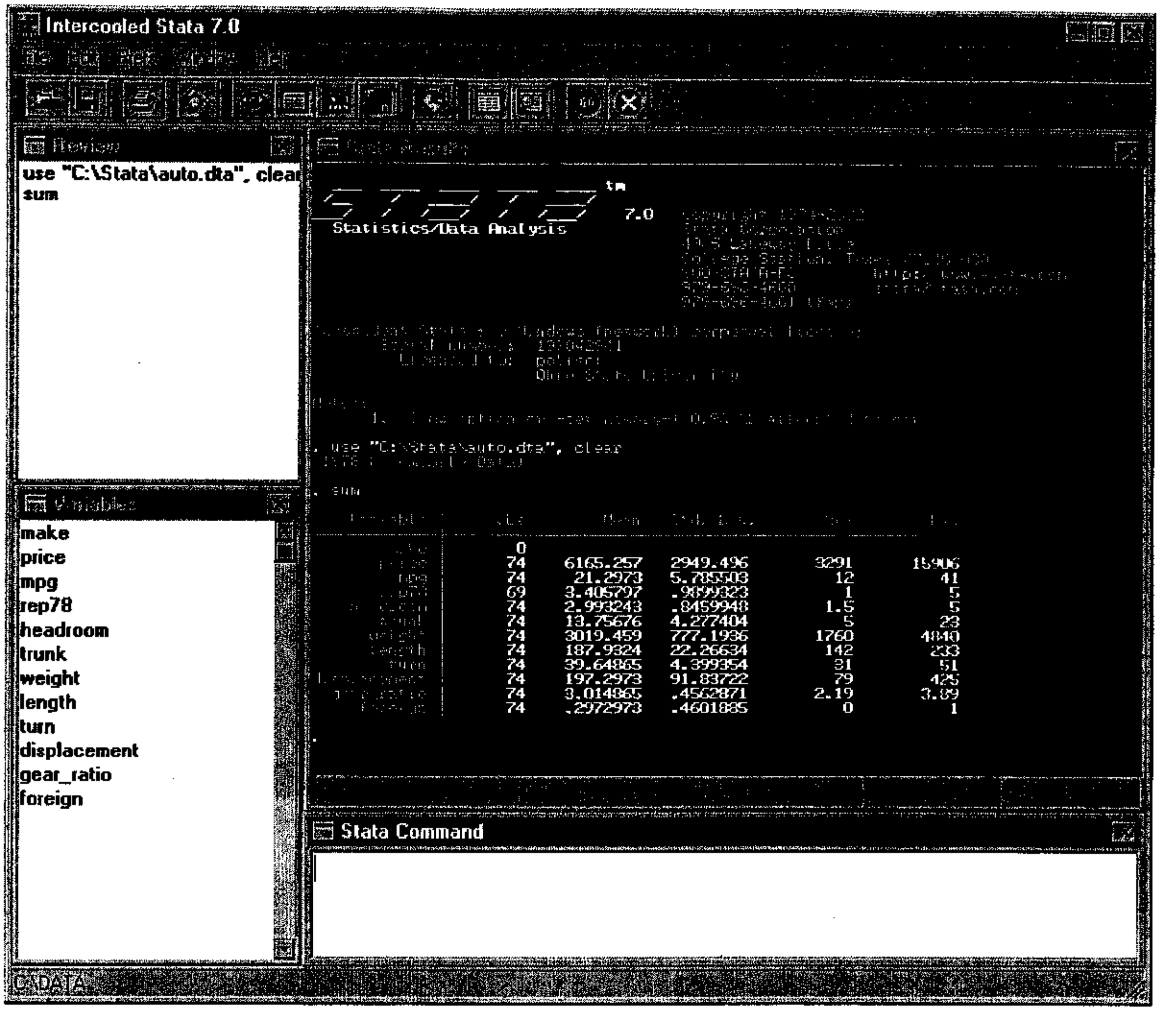
matrix list C

```
symmetric C[3,3]
           r1          r2          r3
r1          1
r2          0           1
r3          0           0           1
```

Accumulating Your Matrices from Pre-Existing Data (beyond 686)
Alright, the above three pages detail pretty much all you will need to complete the assignments in 686, but for those of you who are going to be pursuing statistics beyond 686 I figured it would be a good idea to let you know that Stata can do much more than what is above.

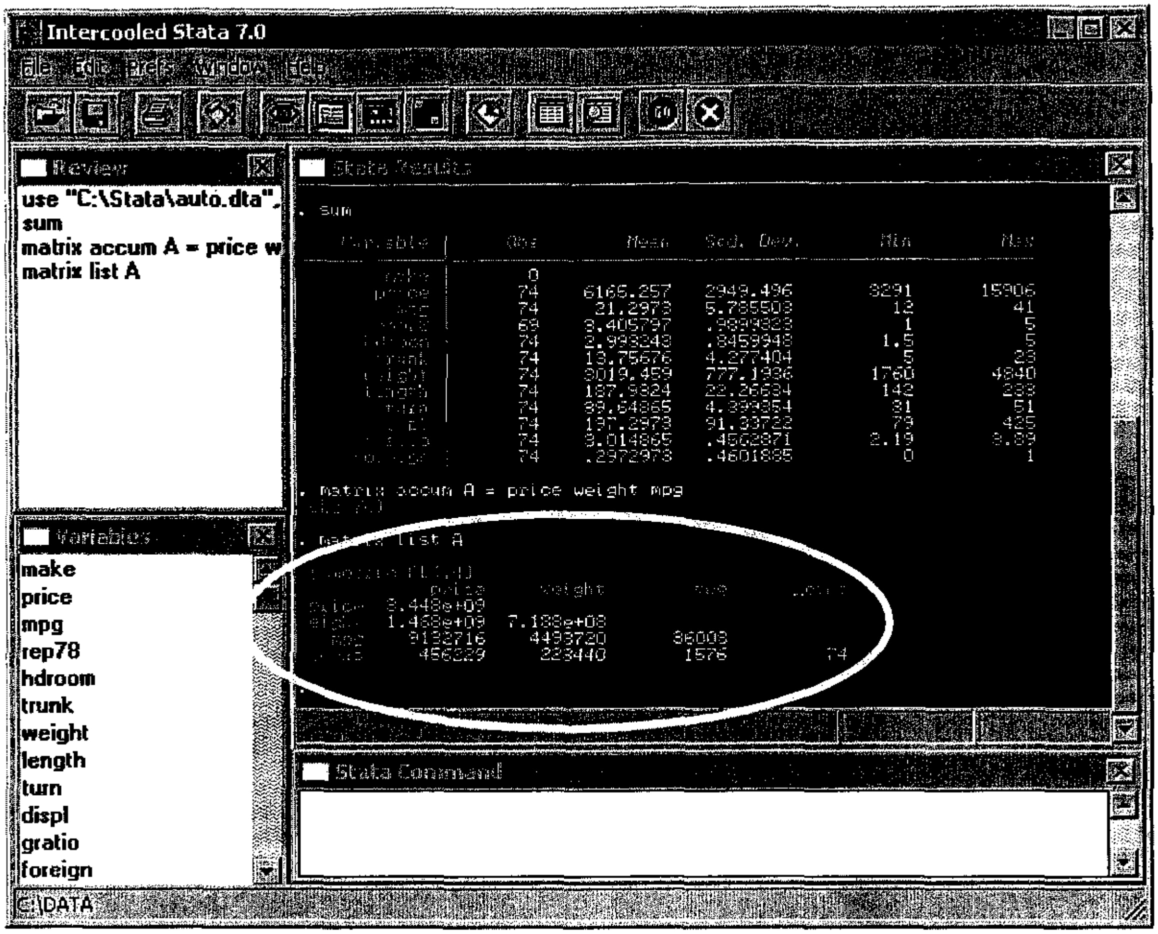
Rather than typing in the data by hand it is ***much*** easier to generate matrices from existing data – particularly the case if you have more many variables and/or observations. Do not do this for your assignments in this class, because the point of the assignments is for you to demonstrate you know how to do OLS (and later GLS) step by step, but as you move beyond 686 what is below will be useful.

Lets begin by opening one of the canned datasets that comes with Stata. The example they use in the programming manual is called auto.dta, so we will use it here also – so you can play along at home. Any machine that has Stata on it has this dataset as well. This is basically what the data looks like and where you get it. I typed the sum command so that we could see how many observations, and what each variable looked like.



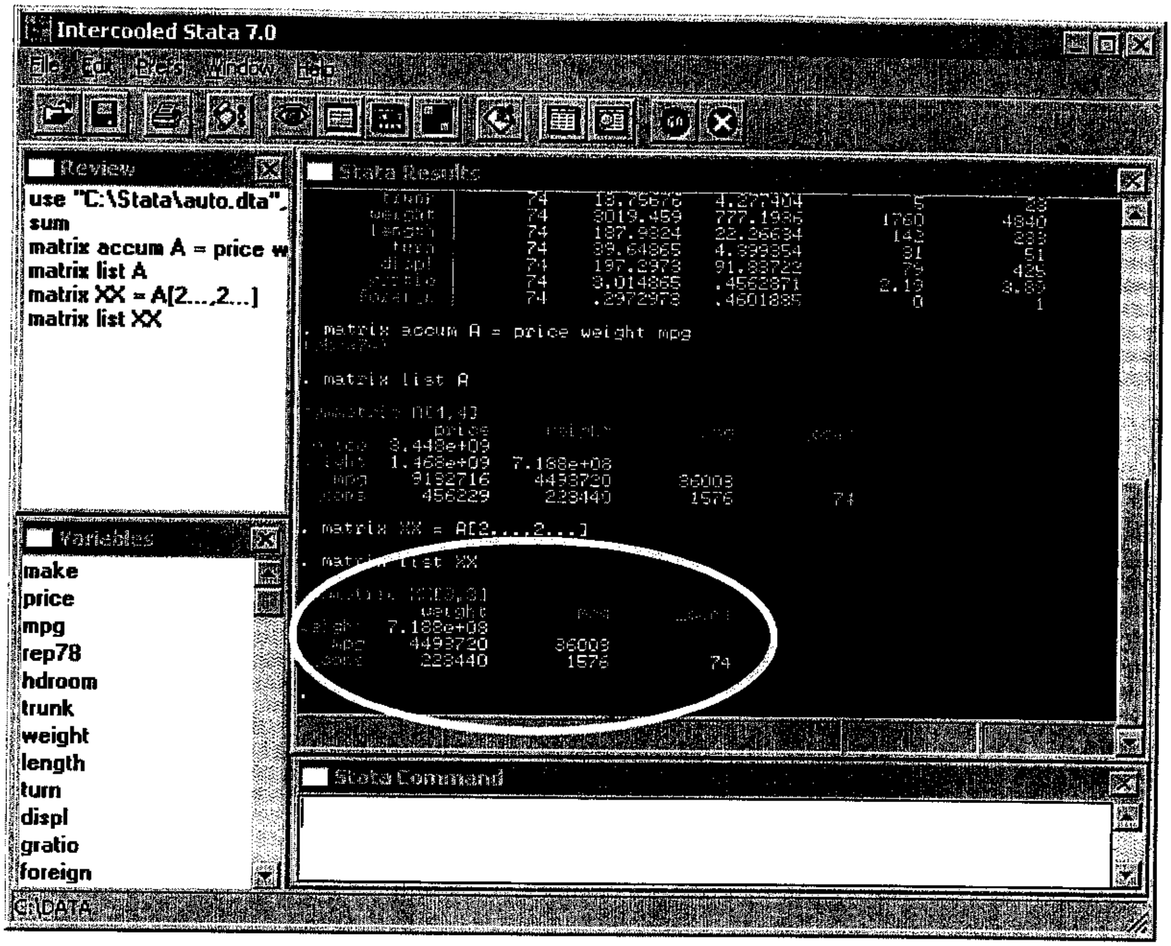
Lets say we wanted to regress the variable price on the variables mpg and weight. We would use the *matrix accum* command to create a matrix directly from the data set. Matrix accum is the command that accumulates a single matrix that holds X'X and X'y, which is all we would need to get $b = (X'X)^{-1}X'y$. You would type:

matrix accum A = price weight mpg
matrix list A



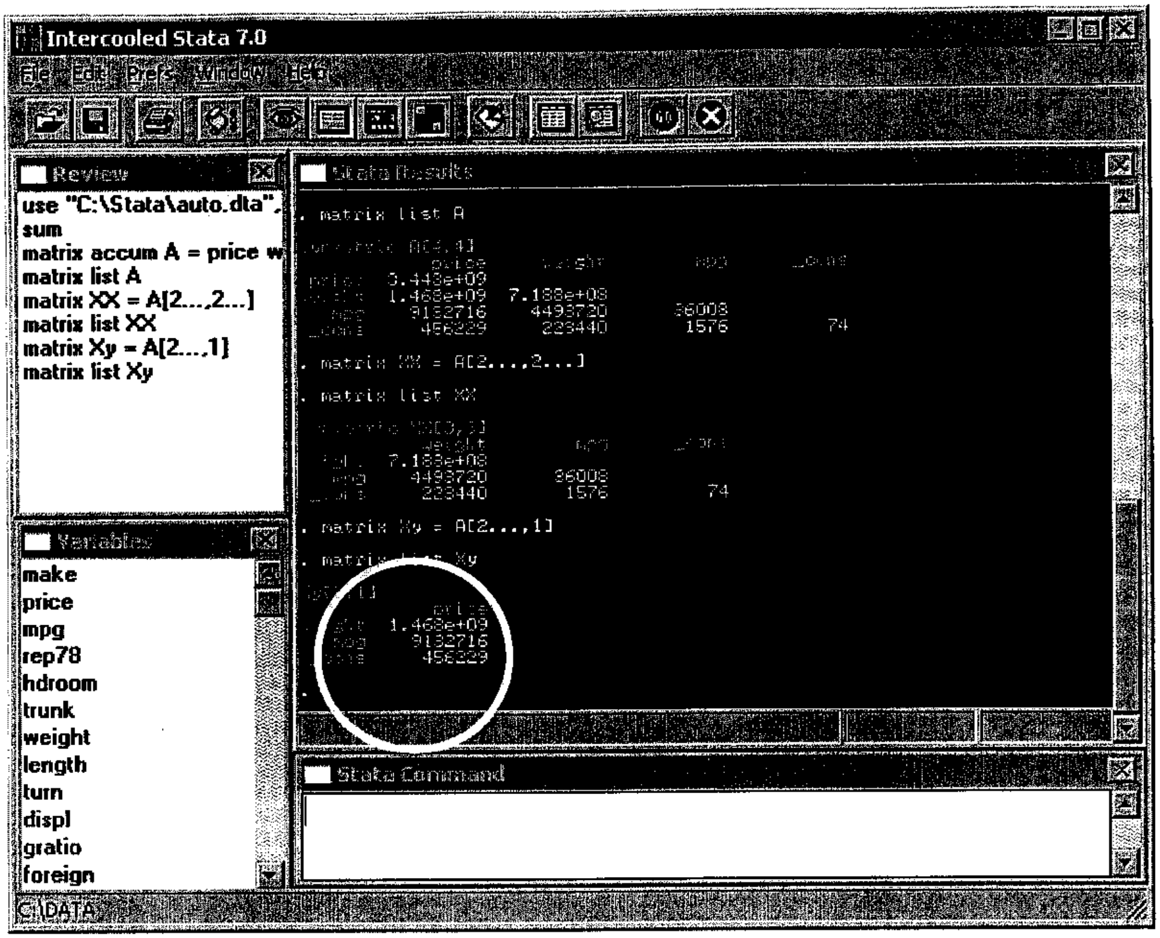
The matrix we just accumulated can be written $A = \begin{pmatrix} y'y & y'X \\ X'y & X'X \end{pmatrix}$ contains everything we need to get the OLS coefficient estimates. We can extract X'X from A beginning at the second row and column, and we can extract X'y from the first column of A, omitting the first row. So type:

matrix XX = A[2...,2...]
matrix list XX



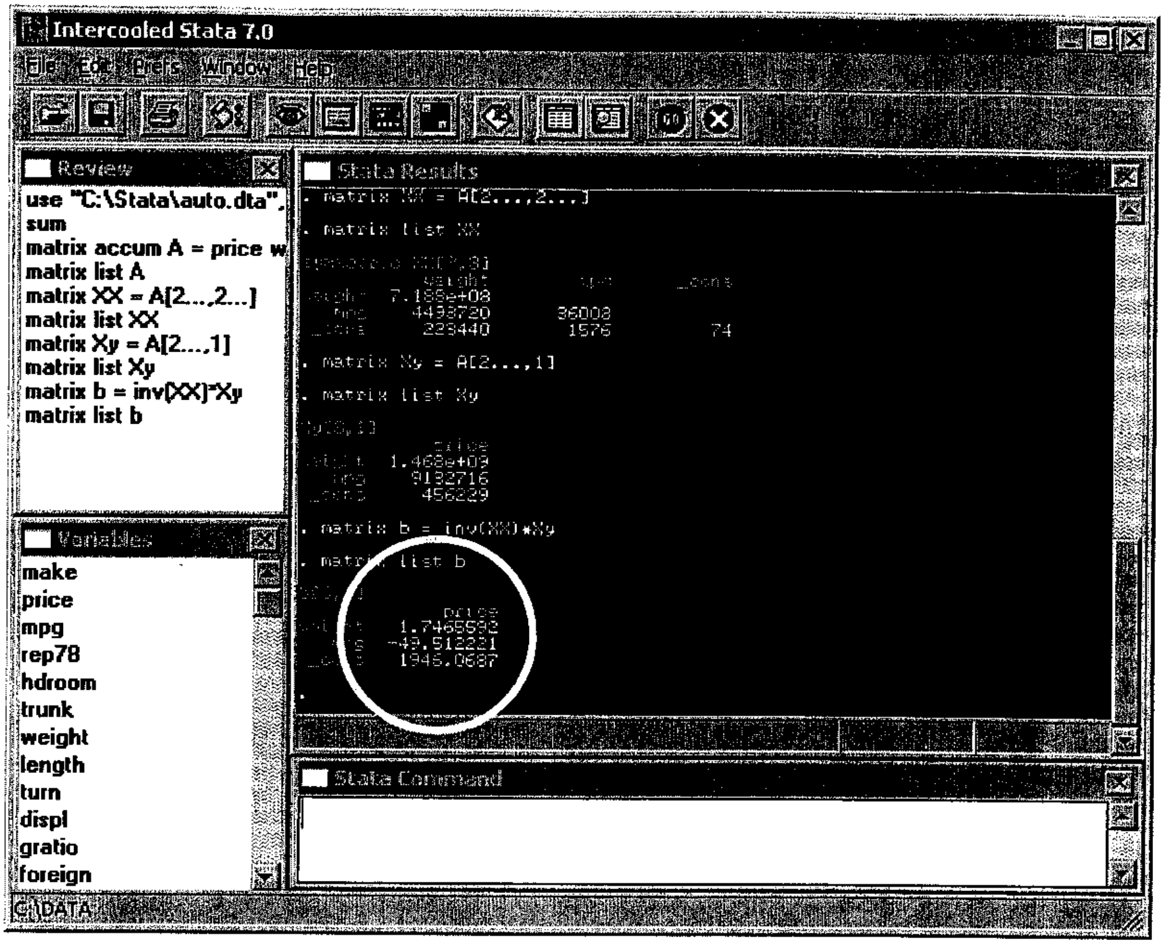
Then, extract X'y by typing:

matrix Xy = A[2...,1]
matrix list Xy



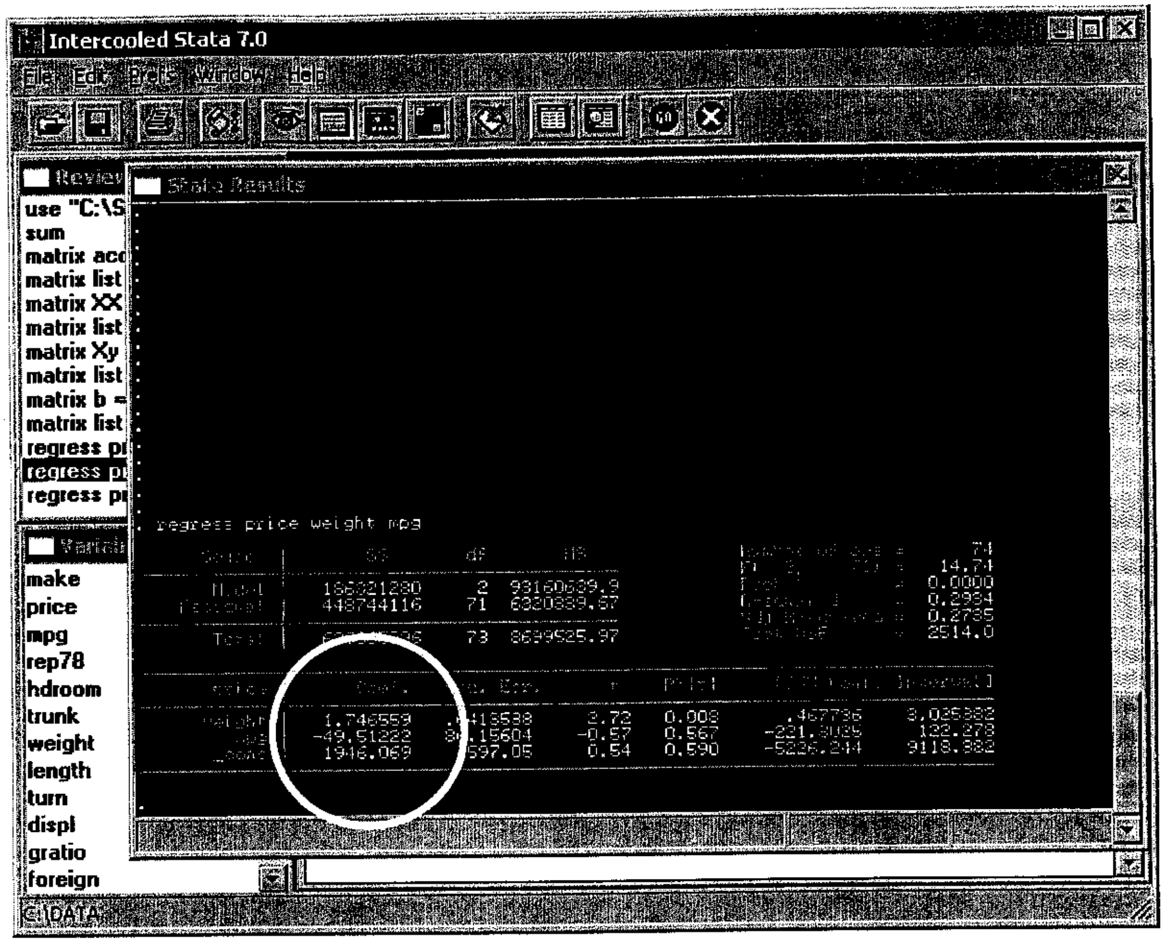
It is now very simple to get the coefficient vector. Type:

*matrix b = inv(XX)*Xy*
matrix list b



We can double check our work here by using the canned OLS command to verify that the coefficients and constants are correct. So type:

regress price weight mpg



Looks pretty good. If you found this little exercise interesting you should know that Stata also has another pretty useful command for matrix operations. The command glsaccum is a generalization of the matrix accum command in that it produces matrices of the form X'BX, where

$$B = \begin{pmatrix} W_1 & 0 & 0 \\ 0 & W_2 & 0 \\ 0 & 0 & W_3 \end{pmatrix}$$

I leave it to you to figure out how best to use this command to do GLS with matrix commands, but the process would be basically the same. Consult the Stata programming manual for details.