Using Mean Bootstrap Weights in Stata: A BSWREG Revision

By James Chowhan and Neil J. Buckley

Abstract

This article presents revisions to a Stata "bswreg" ado file that calculates variance estimates using bootstrap weights. This revision adds new output and analytic features. The main feature added to the program enables researches to use mean bootstrap weights while accounting for the number of weights used to generate the average bootstrap weight. The Workplace and Employee Survey dataset will be used to illustrate the usefulness of this program. This revised version of the "bswreg" command is still an easy to use flexible tool, which is compatible with a wide variety of regression analytical techniques and datasets. The bswreg command and design-based bootstrap weights should only be used for inference when it is theoretically valid.

Introduction

This article presents revisions to "bswreg". BSWREG is a Stata ado file that was developed to calculate variance estimates using bootstrap weights. Piérard et al [2004] developed this program to provide researchers with an easy to use and flexible tool within Stata that can be employed with bootstrap weights to make use of complex survey design information and to calculate sampling variance estimates that account for survey design. Refer to Piérard et al [2004] for details on how to use the bswreg program, its unique features, and for tests validating the program's robustness. This article assumes some familiarity with this previous report.

The revised version of the program adds new features to the output displayed by the program after command execution, but more importantly the revisions allow researchers to use mean bootstrap weights while accounting for the number of weights used to generate the average bootstrap weight. Thus, the program has been designed to account for the fact that some Statistic Canada surveys provide average bootstrap weights. The BSWREG program is provided in Appendix 1.

The Workplace and Employee Survey (WES) data are used for this article to present an example of how important it is to account for the mean bootstrap when calculating design-based variance estimates, in comparison to the method used for standard bootstrap weights.

II. A Brief Comparison of Standard and Mean Bootstrap

Many of Statistics Canada's surveys provide a final weight (or final design weight) and bootstrap weights, which can be used by researchers to generate consistent estimates of population parameters, and sampling variances that account for sample design, respectively.

The standard bootstrap variance estimator for $\hat{\theta}$, used in this program, is given by [Yeo et al., 1999; 3]:

$$v_B\left(\hat{\theta}\right)=\frac{1}{B}\sum_b\left(\hat{\theta}^*_{(b)}-\hat{\theta}^*_{(.)}\right)^2 \qquad \text{where } \hat{\theta}^*_{(.)}=\left(1\middle/B\right)\sum_b\hat{\theta}^*_{(b)} \tag{1}$$

However, this variance estimator is inappropriate when the bootstrap weights are *mean bootstrap* weights. Mean bootstrap weights are bootstrap weights that have been averaged over C iterations usually to protect the confidentiality of survey respondents.

An argument can be made to use the coefficient estimates generated with the full sample final weight as $\hat{\theta}^*_{(.)}$, as opposed to the average of $\hat{\theta}^*_{(b)}$, which are the coefficient estimates generated from the repeated estimation of $\hat{\theta}$ using B bootstrap weights. The bswreg command uses the latter estimate.

Generally, bootstrap weights are generated by randomly drawing samples from each stratum of primary sampling units, with replacement; each sample drawn is equal in size to the number of units in the data set; and then the weight is assigned, using the same clustering and multi-stage sampling that is used to generate the final (design) weight, to each unit in the selected primary sampling unit, the weight is adjusted to reflect the probability of selection into the random sample. Further, observations or sampling units selected into the random sample receive a positive bootstrap weight and units not selected receive a weight of zero [Satin and Shastry, 1993]. This sampling is replicated many times in order to generate a set of bootstrap weights that is large enough to be consistent; the number of times this process is repeated equals the number of bootstrap samples. In equation 1 above there are B bootstrap samples. For example, in the National Population Health Survey there are B=500 bootstrap samples.

Many surveys provide this final set of weights (B samples) for variance analysis. However, after calculating the bootstrap weight samples, some surveys take the additional step of averaging the bootstrap weights over C bootstrap samples. Modifying the variance estimator presented in equation 1, the mean bootstrap variance estimator is as follows:

$$v_{\bar{B}}\left(\hat{\theta}\right)=\frac{C}{B}\sum_b\left(\hat{\theta}^*_{(b)}-\hat{\theta}^*_{(.)}\right)^2 \qquad \text{where } \hat{\theta}^*_{(.)}=\left(1\middle/B\right)\sum_b\hat{\theta}^*_{(b)} \tag{2}$$

Where each b^{th} mean bootstrap sample set of weights is equal to the means of C bootstrap weights. In this specification, the term $\hat{\theta}^*_{(b)}$ is obtained using the b^{th} mean bootstrap weight variable as opposed to the standard bootstrap weight variable used in equation 1 [Phillips, 2004 and Yeo et al., 1999].

For the standard bootstrap weight any single bootstrap replicate, which will include some zero weights, does not pose a confidentiality risk. However, when B is large all standard bootstrap replicates could be examined to identify the pattern of zero weights, and thereby identify cluster membership of observations or records. The mean bootstrap with non-zero

averages, comes from the practice of ensuring that at least one weight in C is non-zero [Yeo et al., 1999]. Since calculating average bootstrap weights in this way helps to mask cluster membership, this reduces the risk to contravening confidentiality.

For example, for the WES data the initial number of standard bootstrap weights samples is equal to B=5000. However, for confidentiality proposes average bootstrap weights were derived. In the WES, the bootstrap weight samples were averaged over groups of C=50. Thus, each of the 100 mean bootstrap weights provided for the WES is an average bootstrap weight of 50 other bootstrap weights.

By inserting the integer C into the numerator of the variance estimator an adjustment is being made which re-introduces the variability that had been removed by using an average bootstrap weight. Thus, the C reflects the fact that the set of bootstrap weights are mean bootstrap weights that have been averaged over C iterations [Statistics Canada, 2003]. Further, the inclusion of the scalar C in the BSWREG revision also expands the breadth and functionality of the program. The revised variance estimator and program can be used to account for variants of the standard Balanced Repeated Replication method. Specifically, this can be used with surveys where only two primary sampling units are selected per stratum (for our PISA illustration that follows, the two PSUs per stratum are schools).

Researchers wishing to use achievement data from Programme for International Student Assessment (PISA) should also account for the added sampling variance that arises from the measurement error inherent in the use of plausible value achievement scales to arrive at a final (total) sampling variance estimator. The bswreg program is only useful when the user is not using achievement data. Refer to Lauzon [2004] for a discussion on the estimation of variance when plausible value achievement data, available in YITS/PISA, are used as dependent variables. Lauzon discusses in detail when the bootstrap should be used with PISA instead of the Balanced Repeated Replication (BRR), and he provides a Stata program for these applications.

An example of this is the Programme for International Student Assessment (PISA) survey and Fay's replicates, which can be used to compute unbiased-standard error estimates to accompany population estimates. In Fay's Balanced Repeated Replication method T half samples are randomly drawn with replacement, similar to the procedure above, from each stratum, of primary sampling units; the sample drawn is equal in size to half the number of units in the data set; then the final weights are adjusted by multiplying the selected half by (2-K) and the other half by K, where K is a number between 0 and 1. For the PISA data K is equal to 0.5 [OECD, 2001]. The Fay's variance estimator is as follows:

$$v_{Fay}\left(\hat{\theta}\right)=\frac{1}{T(1-K)^2}\sum_t\left(\hat{\theta}^*_{(t)}-\hat{\theta}^*_{(.)}\right)^2 \qquad \text{where } \hat{\theta}^*_{(.)}=\left(1/T\right)\sum_t\hat{\theta}^*_{(t)} \qquad (3)$$

Thus, as discussed by Phillips [2004], the mean bootstrap and Fay's method can employ the same variance estimator. For example, in equation 2, C could be set equal to $C=(1-K)^{-2}$, to accommodate for Fay's Method. Using the PISA example, in equation 2, C is equal to 4. For a more detailed discussion see Phillips [2004] and OECD [2001]. Researchers will want to be

careful when using Fay's method with PISA data due to the measurement error inherent to the PV achievement data, as discussed above.

III. Revised Features

The revised BSWREG stata ado program has many useful additional features (see Appendix 2 for a complete list of options). These features include: the added possibility of accounting for mean bootstrap weights or other types of non-standard balance repeated replication techniques, this can be done by using the c*meanbs* option. This new option can be used to specify the number of bootstrap weight samples used to calculate an average bootstrap weight. In the case of WES, the bootstrap weight samples were averaged over groups of C=50, and as such the option c*meanbs* should be set equal to 50 (see example below).

The bootstrap count algorithm has been modified to notify users of the completion of the first few bootstrap repetitions so that researchers can verify the iterations are incrementally stepping forward and not "frozen". The new count may also help researchers better estimate an expected completion time. Also the display form has been changed to a fixed statistic display format/layout.

There were also several new results in e( ) that have been created for the bswreg e-class Stata command, these include: the e(numofbs) variable that is available after running bswreg and contains the number of bootstraps successfully run, the e(N) variable that contains the number of observations in the plain unbootstrapped regression, and the e(cmd) variable that contains "bswreg". All these are in addition to the coefficient and bootstrapped variance-covariance matrices: e(b) and e(V), that continue to be available. Use the "ereturn list" command to display other scalars, macros, matrices, and functions that are available with BSWREG.

In addition to the new features listed above bswreg now also works with additional regression commands including, but not limited to, commands like: reg, areg, qreg, intreg, ivreg, reg3, probit, biprobit, heckprob, heckman, glm, cox, etc... The program now works with all regression commands that support weights. The "xt" series of commands that do not support weights cannot be run with bswreg.

IV. How to – An Example

The revised Stata program is as easy to use as the original bswreg program. Simply copy the "bswreg.ado" and "bswreg.hlp" files, which are described in Appendix 1, to your Stata ADO folder, (type the command "adopath" at the Stata command prompt for a list of ado directory paths in which to place this program), then employ the program by using the following syntax command:

bswreg depvar [varlist] weighttype=full_sample_weight [if exp] [in range],
cmd(STATA_regression_command) [cmdops(options_for_regression_command)]

bsweights(bootstrap_weights_varlist) [cmeanbs(integer)] [level(integer)] [bsci] [saving(path_and_filename[,replace])];

Underlines indicate short forms for the options. To illustrate the use of this syntax command, using the Workplace and Employee Survey 1999, suppose you wished to investigate the effect of location size (small, medium, and large), payroll per employee, percentage of workers covered at the location by a collective bargaining agreement, a flag to distinguish non-profit workplaces from those operating for profit, and in-house dedicated human resources personnel on the availability of individual incentive systems.

Individual incentive systems are one of the areas where the WES focuses its questions. The question: "Does your compensation system include the following incentives? [Including]…Individual incentive systems such as bonuses, piece rate, and commissions are systems that reward individuals on the basis of individual output or performance" [Statistics Canada, 2001]. This is a binary variable where the availability of incentives equals 1 and 0 otherwise. The existence of incentives and the factors that may affect their offering are the essence of this example.

In this example, plant size is determined by each workplace's total employment count. Locations with a total number of employees ranging between 0 and 100 are classified as small; between 101-500 as medium, and 501 or more are large. This is the traditionally category classification used in the Canadian System of National Accounts. In all, three location size dummy variables are defined. Small workplaces are the most numerous group accounting for 98.2% of the population, followed by medium and large workplaces comprising 1.58% and 0.22%, respectively.

Payroll per employee is the average return per workplace to the workforce for labour and human capital services (payroll_per_person), and is calculated by dividing gross-payroll by total employment for each location.

The percentage of workers covered at the location by a collective bargaining agreement is picked-up by the union status variable (pct_union). It is presumed that the degree of unionization in a location may affect the incentive systems offered by workplaces.

The WES does not include the public sector; however, both private sector for-profit and non-profit workplaces are included (binary variable nonprft_flag, where non-profit is indicated by the variable equalling one). The locations not motivated by profit maximization are expected to have different emphasis placed on incentive systems.

The human resources variable "hr_in" attempts to get at whether or not there is a person dedicated to human resource activities at the workplace. The question is phrased as: "Which statement best describes the responsibility for human resource matters at this location?" and the responses are: "(1) there is a separate human resources unit in this workplace employing more than one person; (2) one full-time person in this workplace is responsible for human resources matters; (3) human resources matters comprise part of one person's job in this workplace, such as owner or manager; (4) human resources matters for this workplace are the responsibility of a

person or unit in another workplace; (5) human resources matters are handled as they arise in this workplace (i.e. are not assigned to one person in particular); or (6) Some other arrangement, specify;" where hr_in equals one when respondents selected 1, 2, or 3 and zero otherwise. Locations with in-house dedicated human resources may be more likely to have incentive systems in place.

The example is a logit regression of incentives on a list of size dummies, payroll per employee, unionization, profit motive, and dedicated human resources using WES workplace data 1999. To begin, ensure that your analytical data file and the appropriate bootstrap weight files have been merged correctly (use the appropriate unique identifier). The BSWREG program does not require the bootstrap weights to have any naming scheme. To get design-based standard errors, all 100 mean bootstrap weights will be used in this regression. The command to use these weights is as follows:

bswreg incentives medium large payroll_per_person pct_union nonprft_flag hr_in [pw=wkp_final_wt], cmd(logit) bsweights(wkp_bsw1-wkp_bsw100) cmeanbs(50) level(95); (4)

The results from this regress are as follows:

Output 1

```
. bswreg incentives medium large payroll_per_person pct_union nonprft_flag hr_in
> [pw=wkp_final_wt], cmd(logit) bsweights(wkp_bsw1-wkp_bsw100) cmeanbs(50) level(95) ;


1 bootstraps completed
2 bootstraps completed
3 bootstraps completed
4 bootstraps completed
5 bootstraps completed
25 bootstraps completed
50 bootstraps completed
100 bootstraps completed

Results from BSWREG
-------------------

* The confidence intervals below are based on the normal distribution

  +-------------------------------------------------------------------------------+
  |   Var_name        Coef       BSse     BSzstat   BSpvalue    BSilow95    BSiup95 |
  |-------------------------------------------------------------------------------|
  |     medium    1.038241   0.150151    6.914630   0.000000    0.743950    1.332533 |
  |      large    1.175536   0.243483    4.828008   0.000001    0.698319    1.652753 |
  | payroll_pe    0.000024   0.000004    5.803496   0.000000    0.000016    0.000032 |
  |  pct_union   -0.930827   0.300417   -3.098455   0.001945   -1.519633   -0.342022 |
  | nonprft_fl   -1.089882   0.237791   -4.583371   0.000005   -1.555943   -0.623821 |
  |-------------------------------------------------------------------------------|
  |      hr_in   -0.283383   0.149053   -1.901218   0.057274   -0.575522    0.008756 |
  |      _cons   -1.231138   0.168566   -7.303616   0.000000   -1.561520   -0.900755 |
  +-------------------------------------------------------------------------------+

Total bootstraps completed: 100
```

This is inference appropriate output, because we have used the design-based bootstrap weights. All of our explanatory variables are statistically significant at the 95% level except the dedicated human resources variable (hr_in). Notice how this output differs from the bswreg output that does not adjust for the mean bootstrap using cmeanbs(50). In other words, how is our inference

effected if the cmeanbs(50) option is excluded for the WES data that uses mean bootstraps (see Output 2).

Output 2

```
. bswreg incentives medium large payroll_per_person pct_union nonprft_flag hr_in
> [pw=wkp_final_wt], cmd(logit) bsw(wkp_bsw*) l(95) ;


1 bootstraps completed
2 bootstraps completed
3 bootstraps completed
4 bootstraps completed
5 bootstraps completed
25 bootstraps completed
50 bootstraps completed
100 bootstraps completed

Results from BSWREG
-------------------

* The confidence intervals below are based on the normal distribution

  +-------------------------------------------------------------------------------------+
  |   Var_name        Coef       BSse      BSzstat   BSpvalue    BSilow95     BSiup95 |
  |-------------------------------------------------------------------------------------|
  |     medium    1.038241   0.021235    48.893818   0.000000    0.996622    1.079860 |
  |      large    1.175536   0.034434    34.139172   0.000000    1.108047    1.243024 |
  | payroll_pe    0.000024   0.000001    41.036919   0.000000    0.000023    0.000025 |
  |  pct_union   -0.930827   0.042485   -21.909389   0.000000   -1.014097   -0.847558 |
  | nonprft_fl   -1.089882   0.033629   -32.409325   0.000000   -1.155793   -1.023972 |
  |-------------------------------------------------------------------------------------|
  |      hr_in   -0.283383   0.021079   -13.443637   0.000000   -0.324698   -0.242068 |
  |      _cons   -1.231138   0.023839   -51.644360   0.000000   -1.277861   -1.184415 |
  +-------------------------------------------------------------------------------------+

Total bootstraps completed: 100
```

The above output is clearly problematic. Even though the coefficient estimates are the same, which they should be, the standard errors are substantially lower in Output 2 than they are in Output 1. This is because the scalar factor, where C=50, is being left out of equation 2 and thus the variances are being underestimated by a factor of C. In other words, the standard errors are being underestimated by a factor of $\sqrt{C}$ *or* $\sqrt{50}$. Thus, the above output leads to inappropriate inference. In Output 2 we are led to the conclusion that dedicated human resources is also statistically significant at the 95% level.

Notice in the Output 2 command above the bootstrap weight variable list is specified as "wkp_bsw*". Researchers may want to use the wild card or asterisk when specifying a list of variables that may not be in numerical order in the Stata data set being used. This will avoid a problem with Stata's built in algorithm, which selects variables over the specified range from the order that they occur in the data set rather than the logical range implied by the boundaries. For example, if the boundaries are "bsw1-bsw100" and the first four (of one hundred) weights specified in the data set are bsw1, bsw10, bsw100, and bsw2, then stating the *varlist* as "bsw1-bsw100" in any Stata command will result in only the first three variables (weights) being selected (bsw1, bsw10, bsw100) as opposed the full range. While stating the *varlist* as "bsw*" implies the full range of 100 weights to be selected.

Output 3

```
> logit incentives medium large payroll_per_person pct_union nonprft_flag hr_in
> [pw=wkp_final_wt];


(sum of wgt is   7.1789e+05)

Iteration 0:   log pseudo-likelihood = -3817.6905
Iteration 1:   log pseudo-likelihood = -3635.8102
Iteration 2:   log pseudo-likelihood = -3631.4552
Iteration 3:   log pseudo-likelihood = -3631.4242
Iteration 4:   log pseudo-likelihood = -3631.4242

Logit estimates                                   Number of obs   =       6271
                                                  Wald chi2(6)    =     103.01
                                                  Prob > chi2     =     0.0000
Log pseudo-likelihood = -3631.4242                Pseudo R2       =     0.0488

------------------------------------------------------------------------------
             |               Robust
  incentives |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      medium |   1.038241   .1580988     6.57   0.000     .7283732    1.348109
       large |   1.175536   .2388543     4.92   0.000     .7073897    1.643681
payroll_pe~n |   .0000242   3.82e-06     6.33   0.000     .0000167    .0000317
   pct_union |  -.9308272   .2927079    -3.18   0.001    -1.504524   -.3571304
nonprft_flag |  -1.089882   .2453014    -4.44   0.000    -1.570664   -.6091007
       hr_in |   -.283383   .1420044    -2.00   0.046    -.5617065   -.0050594
       _cons |  -1.231138   .1660942    -7.41   0.000    -1.556676   -.9055991
------------------------------------------------------------------------------
```

It is also important to note that the logit regression with robust standard errors would also lead to incorrect inference, because it does not use the bootstrap weights at all. From the information in Output 3 it appears that all variables are significant at the 95% level, however the standard errors here are biased and do lead to inappropriate inference. The output generated by equation 4 (Output 1) contains design-based standard errors and associated p-values.

The program is not only useful for regression techniques, but can be used to calculate various summary statistics such as frequencies, means, and ratios. See Piérard et al [2004] for a discussion of limitations and for examples of how these statistics can be calculated.

V. Concluding Remarks

This program focuses on design-based (and inference appropriate) variance estimation across various Statistics Canada social surveys. The program can now be used with any survey that has bootstrap weights, This includes a wide spectrum of datasets from the General Social Survey (GSS), National Longitudinal Survey of Children and Youth (NLSCY), National Population Health Survey (NPHS), Survey of Labour and Income Dynamics (SLID), Workplace and Employee Survey (WES), and with some limitations, the Programme for International Student Assessment (PISA) and the Youth in Transition Survey (YITS) just to name a few.

The revisions to this program build on the available features and continue to provide researchers, who use Stata, with a flexible tool that is easy to use and accurate.

References

Lauzon, Darren. 2004. "Variance estimation with plausible value achievement data: Two STATA programs for use with the YITS/PISA data." The Research Data Centres Information and Technical Bulletin. (Spring) 1(1):37-62. Statistics Canada Catalogue no. 12-002-XIE.

Organization for Economic Co-operation and Development (OECD). 2001. "Manual for the PISA 2000 Database." Programme for International Student Assessment (PISA) 2000.

Phillips, Owen. 2004. "Using Bootstrap Weights with WesVar and SUDAAN." The Research Data Centres Information and Technical Bulletin. (Fall) 1(2):1-10. Statistics Canada Catalogue no. 12-002-XIE.

Piérard, Emmanuelle, Neil Buckley, and James Chowhan. 2004. "Bootstrapping Made Easy: A Stata ADO File." The Research Data Centres Information and Technical Bulletin. (Spring) 1(1):23-40. Statistics Canada Catalogue no. 12-002-XIE.

Satin, Alvin, and Wilma Shastry. (1993) Survey Sampling: A Non-Mathematical Guide. Ottawa: Ministry of Industry, Statistics Canada, Social Survey Methods Division. Catalogue No. 12-602-XPE.

Statistics Canada. 2003. "Guide to the Analysis of Workplace and Employee Survey 2001" Business and Labour Market Analysis Division & Labour Statistics Division. (June) Ottawa.

Statistics Canada. *1999 Workplace and Employee Survey*. Business and Labour Market Analysis Division and Labour Statistics Division. 4-4700-2.1: 1999-04-01. STC/LAB-075-75055. Ottawa: Statistics Canada.

Yeo, Douglas, Harold Mantel, and Tzen-Ping Liu. 1999. "Bootstrap Variance Estimation For the National Population Health Survey." American Statistical Association: Proceedings of the Survey Research Methods Section. Baltimore, August.

Appendix 1

Ado File:

```
*                                    WARNING
* The authors are the owners of all intellectual
* property rights (including copyright) in this software.  Subject to the terms below,
* you are granted a non-exclusive and non-transferable license to use this software.
*
* This software is provided "as-is", and the owner makes no warranty, either express
* or implied, including but not limited to, warranties of merchantability and fitness
* for any particular purpose.  In no event will the owner be liable for any indirect,
* special, consequential or other similar damages.  This agreement will terminate
* automatically without notice to you if you fail to comply with any term of this
* agreement.

* TO CHANGE THE DECIMAL DISPLAY FORMAT OF THE BOOTSTRAPPED OUTPUT SEARCH FOR THE "FORMAT" COMMAND NEAR THE
BOTTOM OF THIS PROGRAM;

program define bswreg, eclass sortpreserve byable(recall)

* October 21st, 2004 Buckley, Chowhan
*    BSWREG should now work with any regression command that accepts a weight
*      (including, but not limited to commands like: reg, qreg, intreg, ivreg, reg3, probit, biprobit,
heckprob, heckman, glm etc...)
*    fixed problem with running BSWREG with regression methods that analyze censored data containing missing
values (e.g. INTREG is now fully functional within BSWREG)
* September 30th, 2004 Buckley, Chowhan
*    added possibility of mean bootstrap weights
*    changed bootstrap count algorithm
*    created e(numofbs) variable that is available after running bswreg and contains the number of bootstraps
successfully run
*    created e(N) variable that contains number of observations in plain unbootstrapped regression
*    created e(cmd) variable that contains "bswreg"
*    fixed statistic display format/layout
* August 8th, 2003 Pierard, Buckley, Chowhan (original)

# delimit;
version 7.0;

syntax anything [aweight pweight fweight iweight] [if] [in], cmd(string) [cmdops(string)]
  BSWeights(varlist numeric) [Cmeanbs(integer 1)] [Level(integer 95)] [bsci] [SAVing(string)];

*This sets the touse variable = 1 if observation is in our sample;
marksample touse;
*Error check to make sure a weight was used;
if "`weight'"==""
  {;
  noi di in red "BSWREG error: You must specify a weight!";
  exit;
  };

quietly
{;

*Preserve the original dataset and set parameter values and setup temporary matrices;
preserve;
set more 1;
tempvar esamplevar;
tempname bhat bsVC bsbhat bsbetas;

*The next line runs the wanted regression and checks for errors;
capture `cmd' `anything' [`weight'`exp'] `if' `in', `cmdops';

if _rc ~= 0
   {;
   noi di in red " ";
   noi di in red "Error doing: `cmd' `anything' [`weight'`exp'] `if' `in', `cmdops'";
   noi di in red " ";
   noi di in red "The regression command you have typed in resulted in an error, please investigate";
   noi di in red "this error outside of the 'bswreg' program by typing in the regression command itself";
   noi di in red "with the options you specified.";
   noi di in red " ";
   exit;
   };

*The next line runs the wanted regression and we store the coefficients in a matrix for later use;
`cmd' `anything' [`weight'`exp'] `if' `in', `cmdops';
local _numofobs = e(N);
gen `esamplevar'=e(sample);

*e(b) is a 1x(k+1) coefficient vector if the model has a constant and k is the number of variables other than
the constant;

matrix `bhat'=e(b);
matrix list `bhat';
matrix `bsVC'=e(V);
```

```
*we store the variable names of the regressors and the number of regressors in local macros;
local _varnames : colfullnames(`bhat');
local _k=colsof(`bhat')-1;
local _k1=`_k'+1;

*Generate concatenated list of placeholder regressor variable names xc1-xck1, later to be turned into
variables;
local _xclist="";
forvalues _i = 1/`_k1'
   {;
   local _xclist `_xclist' _xc`_i';
   };
*We assigned these placeholder variable names to the regressors in the coefficient vector;
matrix colnames `bhat' = `_xclist';
*Each "true estimate of beta" is saved under it's own variable name;

svmat double `bhat', name(col);
matrix colnames `bhat' = `_varnames';

*Realboot is the actual number of successful bootstrap regressions run in case we get any
convergence/regression errors etc., it starts off at the specified number of bootstrap weights;
local _realboot: word count `bsweights';
noi di " ";

*The main bootstrap loop will run with each bootstrap weight in the supplied bsweight varlist and exit with the
matrix named BETAS containing all the bootstraps of our coefficients, a (boot)x(k+1) dimensional matrix;
local _i 1;
*Start of bootstrap loop;
foreach bswvar of local bsweights
 {;

 *Display notice of number of completed bootstraps every time 50 are completed;
 if (mod(`_i',100)==0 | `_i'<6 | `_i'==25 | `_i'==50)
  {;
  noi di in green `_i' " bootstraps completed";
  };

 *Run the regression with the chosen set of bootstrap weights, only use the coefficients if there are no
errors;

 capture `cmd' `anything' [`weight'=`bswvar'] `if' `in', `cmdops';
 if _rc==0
  {;

   *Store coefficients in the bootstrap matrix;
   matrix `bsbhat'=get(_b);
   *bsbhat is a 1x(`k'+1) (row) vector if the model has a constant.  Need to transpose;
   matrix `bsbhat'=`bsbhat'';

   *If we have the proper number of coefficients then add them to the bootstrap matrix, otherwise do not add
them (this most likely arises due to a regressor being dropped due to multicollinearity;
   if rowsof(`bsbhat')==`_k1'
        {;
        *If we are on the first bootstrap then create the bsbetas matrix, otherwise append to it;
        if `_i'==1
           {;
           matrix `bsbetas'=(`bsbhat');
           };
        else
           {;
           matrix `bsbetas'=(`bsbetas',`bsbhat');
           };
        };
   else
        {;
        matrix drop `bsbhat';
        local _realboot=`_realboot'-1;
        noi di "Bootstrap #`_i' has been dropped for not having the correct number of coefficients";
        };
  };
 else
  {;
  local _realboot=`_realboot'-1;
  noi di "bootstrap #`_i' has been dropped due to an error estimating the regression";
  };
 local _i=`_i'+1;
 };

*End of bootstrap loop;

*All the bootstraps have been completed now calculate the new standard errors and display relevant statistics;
*We must transpose the matrix to make each row now, then column, a new variable;
matrix `bsbetas'=`bsbetas'';
*Generate concatenated list of colnames, later to be turned into variables;
local _xvlist="";
forvalues _i = 1/`_k1'
   {;
   local _xvlist `_xvlist' _xv`_i';
   };
```

```
*Calls each row of the matrix by the name of the independent variable it corresponds to (we call them _xv'_i'
so that they are not mixed up with the "real" variables);
matrix colnames `bsbetas'= `_xvlist';

*Separate each column as a new variable.  The format of the data must be specified.  It renames each variable
by the name of the column;
svmat double `bsbetas', name(col);

*Generate the bootstrapped variance-covariance matrix, you can access this in e(V) after running the BSWREG ado
file;
*CmeanBS is the number of bootstrap weight samples used to calculate an average bootstrap weight sample;
*When CmeanBS is not equal to 1 a mean bootstrap factor exists dependent on the survey, the default value is 1;
forvalues _i = 1/`_k1'
 {;
 forvalues _j = 1/`_k1'
  {;
    correlate _xv`_i' _xv`_j', covariance;
    matrix `bsVC'[`_i',`_j'] = (((`_realboot'-1)*(`cmeanbs'))/`_realboot')*r(cov_12);
  };
 };

*Generate the standard deviation, t-stat, conf. int. etc. for each variable;
tempvar _bsobs _uniqobs _coefnum;
gen `_bsobs'=_n;
forvalues _i = 1/`_k1'
 {;

 sum _xv`_i';
 * Like the SAS bootvar program, we use (boot-1)/boot because variance and standard error have different
denominators;

 * See above for description of CmeanBS;
 gen _sdx`_i'=sqrt((((`_realboot'-1)*(`cmeanbs'))/`_realboot')*r(Var)) in 1/1;
 gen _t`_i'=_xc`_i'/_sdx`_i' in 1/1;
 gen _abst`_i'=abs(_t`_i') in 1/1;
 gen _p`_i'=2*norm((-1)*_abst`_i') in 1/1;
* gen _p`_i'=2*ttail(`_realboot'-1,_abst`_i') in 1/1;
 if "`bsci'"==""
 {;
   gen _low`level'`_i'=_xc`_i'-invnorm(1-((1-(`level'/100))/2))*_sdx`_i';
   gen _high`level'`_i'=_xc`_i'+invnorm(1-((1-(`level'/100))/2))*_sdx`_i';
 };
 if "`bsci'"=="bsci"
   {;
   sort _xv`_i';
   local _obslow= max(1,round(((1-(`level'/100))/2)*`_realboot',1));
   local _obshigh= max(1,round((1-((1-(`level'/100))/2))*`_realboot',1));
   local _obslow2= _xv`_i'[`_obslow'];
   local _obshigh2= _xv`_i'[`_obshigh'];
   sort `_bsobs';
   gen _low`level'`_i'= `_obslow2' in 1/1;
   gen _high`level'`_i'= `_obshigh2' in 1/1;
   };
 };

*Assign each coefficient its true regressor name stored at the beginning of this program;
local _i=1;
foreach _curname in `_varnames'
 {;
 gen str10 _xname`_i'="`_curname'";
 local _i=`_i'+1;
 };

*Reshape the data so that the bootstrapped stats can be displayed easily, and then display the results;
keep _xname* _xc* _sdx* _t* _p* _low`level'* _high`level'*;
drop if _n>1;
gen `_uniqobs'=1;

reshape long _xname _xc _sdx _t _p _low`level' _high`level', i(`_uniqobs') j(`_coefnum');

*The %9.4f tells stata to display the bootstrapped results to 6 decimals using 15 numbers total -- this can be
changed to suit tastes;
format _xc _sdx _t _p _low`level' _high`level' %11.6f;
*creates nice labels for variables
label var _xname "Name of variable";
ren _xname Var_name;
label var _xc "Coefficient estimate";
ren _xc Coef;
label var _sdx "Bootstrap standard error of coefficient";
ren _sdx BSse;
label var _t "Bootstrap z-statistic";
ren _t BSzstat;
label var _p "Bootstrap p-value";
ren _p BSpvalue;
if "`bsci'"==""
 {;
 label var _low`level' "Bootstrap lower confidence interval assuming a normal distribution";
 label var _high`level' "Bootstrap upper confidence interval assuming a normal distribution";
 };
if "`bsci'"=="bsci"
 {;
```

```
 label var _low`level' "Bootstrap lower confidence interval using bootstrap sample distribution";
 label var _high`level' "Bootstrap upper confidence interval using bootstrap sample distribution";
 };
ren _low`level' BSilow`level';
ren _high`level' BSiup`level';

*Display RESULTS!;
noi display in green " ";
noi display in green "Results from BSWREG";
noi display in green "--------------------";
noi display in green " ";
if "`bsci'"=="bsci"
 {;
 noi display in green "* The confidence intervals below are based on the bootstrapped distribution";
 };
else noi display in green "* The confidence intervals below are based on the normal distribution";
*noi display in green " ";
format Coef BSse BSzstat BSpvalue BSilow`level' BSiup`level' %10.6f;
format Var_name %10s;
noi list Var_name Coef BSse BSzstat BSpvalue BSilow`level' BSiup`level', nodisplay noobs;

noi di " ";

noi di "Total bootstraps completed: `_realboot'";

*Set the eclass variables like the coefficients and the variance-covariance matrix into their appropriate
matrices so that F-tests and the like can be run;
*If you wish the TEST command to produce F-tests after the BSWREG command then add ", dof(`_realboot')" to the
line below;
estimates post `bhat' `bsVC';
*This next line creates a e(numofbs) scalar available after running bswreg that contains the number of
boostraps run, di e(numofbs);
estimates scalar numofbs = `_realboot';
estimates scalar N = `_numofobs';
estimates local cmd = "bswreg";

*Save the bootstrap raw data is the "SAVING" option has been used;
if "`saving'"~=""
 {;
  drop _*;
  save "`saving'", `replace';
 };

*Restore the original dataset
restore;
};
end;
```

BSWREG Help File

```
{smcl}
{* 21October2004 Buckley/Chowhan}

{* 30September2004 Buckley/Chowhan}
{* 8August2003 Pierard/Buckley/Chowhan}
{hline}
help for {hi:BSWREG}
{hline}

{title:BSWREG - uses bootstrap weights to calculate standard errors in models involving complex survey data.}

{p 8 13}{cmd:bswreg} depvar [varlist] {it:weighttype}={it:full_sample_weight} [{cmd:if} {it:exp}] [{cmd:in}
{it:range}]{cmd:,} {cmd:cmd(}{it:STATA_regression_command}{cmd:)}
[{cmd:cmdops(}{it:options_for_regression_command}{cmd:)}]
  {cmdab:bsw:eights(}{it:bootstrap_weights_varlist}{cmd:)} [{cmdab:c:meanbs(}{it:integer}{cmd:)}]
[{cmdab:l:evel(}{it:integer}{cmd:)}] [{cmd:bsci}]
[{cmdab:sav:ing(}{it:path_and_filename}[{cmd:,replace}]{cmd:)}];
{p} {cmd:cmd()} and {cmd:bsweights()} are required options for the {cmd:BSWREG} command.
{p} {cmd:by ...: } and {cmd:bysort ...:} can be used with {cmd:BSWREG}. See help {help by}.
{p} {cmd:aweight}s, {cmd:fweight}s, {cmd:iweight}s, and {cmd:pweight}s are allowed as long as the
      given regression command is compatible with them. See help {help weights}.
{p} As {cmd:BSWREG} is an eclass STATA program, it provides STATA with the {cmd:e(b)} coefficient vector and
the {cmd:e(V)} bootstrapped variance-covariance matrix.
    The {cmd:test} command can be used immediately following the {cmd:BSWREG} command to conduct Wald tests
based on the chi-squared distribution.

{inp:The software is provided "as-is" and the authors are not responsible for any misuse.}

{title:Description}

(used to calculate regression statistics using Statistics Canada's bootstrap weights)

{p}{cmd:bswreg} runs a number of regressions, each with a particular bootstrap
```

```
weight so that bootstrapped standard errors on the coefficients can be calculated
and displayed. Use of bootstrap weights is recommended for calculating reliable
standard errors, confidence intervals etc. on data from complex
household surveys.

The user provides the names of the bootstrap weights to the {cmd:BSWREG} command
in the {cmdab:bsw:eights(varlist)} option. You must already have the appropriate
bootstrap weights merged into your datafile for this command file to work.  NPHS
merges on REALUKEY and SLID merges on PERSONID. Below is a sample .DO file that
merges NPHS bootstrap weights into a datafile named data.dta:

{inp:use data.dta, replace"}
{inp:sort realukey"}
{inp:save data.dta, replace"}
{inp:use bootstrap/sas_bs_wt_1_4.dta, replace"}
{inp:destring realukey, replace"}
{inp:sort realukey"}
{inp:merge realukey using data.dta"}
{inp:keep if _merge==3"}

{title:Options}

{p 0 4}{cmd:cmd(}{it:STATA_regression_command}{cmd:)} specifies the Stata regression command to bootstrap. This
is a {cmd:required} option. "regress", "probit" and "logit" are a few possibilities.

{p 0 4}{cmd:bsweights(}{it:varlist}{cmd:)} specifies a variable list of the bootstrap weight names. This is a
{cmd:required} option. For instance, if your bootstrap weights are named bsw1 to bsw500, you may wish to use
the
{cmd:bsweights(bsw1-bsw500)} option. In order to avoid Stata variable ordering problems it might be better to
specify {cmd:bsweights(bsw*)} when using all weights.

{p 0 4}{cmd:cmdops(}{it:options_for_regression_command}{cmd:)} specifies the options you wish to use on the
Stata regression command provided in {cmd:cmd()}. Some options are useful and others are meaningless in a
bootstrap weighting context.
For instance, if you wish to run the REGRESS command with no constant then use the {cmd:cmd(regress)
cmdops(noconstant)} options.  Options like {cmd:robust} are meaningless in this context since the command
computes bootstrap weighted
standard errors not robust ones.

{p 0 4}{cmd:cmeanbs(}{it:integer}{cmd:)} specifies the number of bootstrap weight samples each mean bootstrap
weight is averaged over, in the case of surveys that use mean boostrap weights. The default is
that the bootstraps privided are not mean bootstrap weights, {cmd:cmeanbs(1)}.

{p 0 4}{cmd:level(}{it:integer}{cmd:)} specifies the confidence level, in percent,
for confidence intervals.  The default is {cmd:level(95)}. See help {help level}.

{p 0 4}{cmd:bsci} specifies that the confidence intervals be calculated from the raw bootstrapped distribution
of coefficients rather than using the standard formula
based on the bootstrapped standard error and the normal distribution.

{p 0 4}{cmd:saving(}{it:filename}[{cmd:,replace}]{cmd:)} saves the bootstrap statistics in a
separate Stata dataset file that can later be loaded and used by other .DO and .ADO files.
If you do not specify an extension, {cmd:.dta} will
be assumed. Include the {cmd:,replace} option to overwrite an existing file.

{title:Outputed variables}
{inp: Var_name:}  This is the STATA variable name of the regressor.
{inp: Coef:}      This is the coefficient from the specified regression.
{inp: BSse:}      This is the new standard error of the coefficient,
            calculated using bootstrap weights.
{inp: BSzstat:}   This is the new z-stat of the coefficient,
            calculated as the coefficient divided by the bootstrapped standard error.
{inp: BSpvalue:}  This is the new p-value of the coefficient,
            calculated using the z-statistic.
{inp: BSilow(level):} This is the lower (level)% confidence interval around the coefficient
            using the bootstrapped std. error.
{inp: BSiup(level):}  This is the upper (level)% confidence interval around the coefficient
            using the bootstrapped std. error.

{title: e-class results}
{inp: e(numofbs): Scalar} The number of successful bootstrap replications.
{inp: e(N): Scalar}       The number of observations in the underlying survey sample.
{inp: e(cmd): Macro}      Contains "bswreg".
{inp: e(b): Matrix}       This is the vector of coefficients.
{inp: e(V): Matrix}       This is the bootstrapped variance-covariance matrix.

{title:Examples}

{p 8 12}{inp:. bswreg income education rural [aw=wt] if married==1, cmd(regress) bsw(bsw1-bsw500)}
{p 8 12}{inp:. bswreg employed education rural [aw=wt66], cmd(probit) bsw(bsw50-bsw100)}
{p 8 12}{inp:. bysort maritalstatus: bswreg income education rural [aw=wt], cmd(reg) bsw(bsw1-bsw500)}
```

```
  {inp:cmdops(noconstant) level(99) bsci saving(c:\data\bsw1.dta,replace)}
{p 8 12}{inp:. bswreg wesemployeeincentives wesworksize [aw=wt], cmd(logit) bsw(bsw1-bsw500) cmeanbs(50)}
```

Appendix 2

The bswreg command allows for the use of options. The program has several options available:

cmd: specifies the Stata regression command to bootstrap. This is a required option. The following regression commands have been tested explicitly: regress, logit, probit, tobit, ologit, oprobit, biprobit, mlogit, qreg, glm, intreg, boxcox, (basically any single stage estimation technique should work with this program) and non-twostage "xt" commands that support weights.

bsweights: specifies a variable list of the bootstrap weight names. This is a required option. For instance, if your bootstrap weights are named bsw1 to bsw500, you could specify the option as bsweights(bsw1-bsw500). In order to avoid Stata variable ordering problems it might be better to specify bsweights(bsw*) when using all weights.

cmdops: specifies the options you wish to use on the Stata regression command provided in cmd(). Some options are useful and others are meaningless in a bootstrap weighting context. For instance, if you wish to run the REGRESS command with no constant then use the cmd(regress) cmdops(noconstant) options. Options like robust are meaningless in this context since the command computes bootstrap weighted standard errors not robust ones.

cmeanbs: specifies the number of bootstrap weight samples used to calculate an average bootstrap weight sample, mean bootstrap weight factors are dependent on the survey. The default is equal to 1, implying that the bootstrap weights are not mean bootstrap weights.

level: specifies the confidence level, in percent, for confidence intervals. The default is level(95).

bsci: specifies that the confidence intervals be calculated from the raw bootstrapped distribution of coefficients rather than using the standard formula based on the bootstrapped standard error and the normal distribution. This option is provided for users that may have a theoretical reason for employing the confidence intervals derived from the bootstrapped distribution of coefficients.

saving: saves the bootstrap statistics in a separate Stata dataset file that can later be loaded and used by other .DO and .ADO files. If you do not specify an extension, .dta will be assumed. Include the replace option to overwrite an existing file.