Paper ST-10

Using a SAS/IML Nonlinear Programming Procedure To Determine a Single Uniform Weighting Scheme For a Complex Survey Design

Richard A. Moore, Jr., U.S. Bureau of the Census, Washington, DC

Abstract

In probability-based surveys, each sampling unit is assigned a probability of being selected into the sample. Each selected unit is assigned a weight equal to the inverse of the probability of selection. For estimation, all responses are typically inflated by the corresponding weight and summed. In a simple design, the same weight is used to weight all variables. In more complex designs, this weight can vary with the variable tabulated (3.00 for payroll, 3.50 for sales, ...). Although multiple weights produce more accurate estimates, they make it very difficult for estimating other statistics (e.g., ratios or regressions), since traditional procedures require that the same weight be used for all variables.

Data from the 1997 Surveys Of Minority- and Women-Owned Business Enterprises are tabulated using a multiple weighting scheme. Each record contains up to 12 different weights, corresponding to different published characteristics. Using the SAS/IML procedure, **Proc NLPNMS**, a single weight is assigned to each unit. When units are tabulated using this single weight, nearly all published values are preserved. This talk focuses on (1) the appropriate constraints and objective function, (2) diagnostic measures of efficiency, and (3) characterizing situations where the routine provides inaccurate solutions and offering some possible work around solutions.

Introduction

Most probabilistic survey designs assign a single weight to each selected sampling unit. The weight is usually the inverse of the probability of selection. For estimation, all responses are inflated by the corresponding weight and summed. In a simple design, the same weight is used to weight all survey characteristics. In some more complex designs, there are situtations where weights vary not only from record to record, but also by the different characteristics on each record. The 1997 Surveys of Minority- and Women-Owned Business Enterprises are examples of programs which use different weights for different response variables.

Complex weighting makes the data set very difficult to use. Have you ever tried to use a **weight** statement in a **Proc Reg** where different variables have different weights? This paper offers a solution to this problem. Using a SAS/IML non-linear programming routine, we assign a single weight to each observation, so that all key published estimates are “virtually” the same, regardless of which weighting system is used.

The 1997 Surveys of Minority- and Women-Owned Enterprises — SMOBE/SWOBE

Every five years, United States Bureau of the Census conducts the Surveys of Minority- and Women-Owned Business Enterprises (SMOBE/SWOBE). These surveys were last conducted for 1997. Their major function is to publish a comprehensive set of statistics for minority- and women-owned businesses. These include the number of businesses, total receipts, payroll, and employment at various geographic and industrial level combinations for businesses primarily owned by (1) one of the 3 major minority races (Black, Asian - Pacific Islander (API), American Indian - Alaskan Native (AIAN), (2) Hispanics (HISP), and (3) Women. Note that these categories are not mutually exclusive. SMOBE/SWOBE often identifies a business which is owned by an American Indian Hispanic or an Asian-Pacific Islander Female, etc. Each such business is tabulated in all relevant publications. (e.g., A Black Hispanic female-owned business would contribute to the Black-owned, the Hispanic-owned, and the Women-owned numbers.) Basic “building block” estimates are obtained for aggregate totals at the state-industry-minority category (e,.g., American Indian-owned restaurants in Delaware) . The 1997 SMOBE/SWOBE used 50 states (and D.C.) and 71 different industry categories. Other estimates (e.g., total number of Hispanic-owned businesses in Georgia) can then be obtained by adding the estimates from the appropriate building blocks.

A Simplified Explanation of the Sample Design

Each year, the U.S. Census Bureau constructs a register of all business operations that file payroll and/or income tax returns. The 1997 register contained 20.8 million businesses. For each business, the register contains (1) the location(s) where the business operates, (2) the type of operation conducted at each location, (3) the employment, payroll, and receipts for each location, and (4) in many cases, a listing of (at least) several of the owners. The register contains most of the components (state, industry, employment, payroll, and receipts) required to produce the building block numbers. Only the ownership demographics are missing. Unfortunately, a complete enumeration of all 20.8 million businesses for the sole purpose of obtaining this information is cost prohibitive.

SMOBE/SWOBE, therefore, selects a sample of 2.5 million businesses and solicits gender, race, and Hispanic ownership information from each. The universe is first stratified by state and type of operation. Even though exact ownership demographic information is not available for most businesses, there are a number of sources from which ownership race, Hispanic, and gender inferences can be made. Although these inferences are too unreliable by which to tabulate, they can be used to systematically list the businesses within each stratum.

The list can be constructed so that the business in Position X is very likely to give similar demographic responses as the those listed in the neighboring positions (Position X - K, X - K + 1, ..., X + L - 1, X + L; with K and L both non-negative and both can vary with the value of X). Consequently, we use a stratified systematic design to select our sample. Selected businesses are then weighted by the inverse of their probability of selection. We determine the sampling rates high enough to ensure that the responses of the selected cases when inflated by their sampling weight will provide accurate estimates for each state-industry-classification building block.

Compensation for Non-Response

About 2.0 million (or about 4 out of every 5) of the selected businesses provided SMOBE/SWOBE with the requested information. However, it was found that non-response was far from uniform for three reasons. First, businesses selected from portions of the listing where most responses were expected to be "White non-Hispanic-owned" responded much better than cases selected from other portions of the listing (e.g., cases expected to respond as "White Hispanic-owned"). This problem was addressed by identifying all respondents on the original listing and increasing the number of neighboring cases that certain respondents would represent. Suppose, for example, the Position 10 respondent was to originally represent the 6 businesses listed in Positions 9 through 14. Due to non-response, he is now expected to represent 9 businesses listed in Positions 6 through 14. Due to non-response, his tabulation weight increases by a factor of 9/6 or 1.500.

Upon further inspection, we next noticed that businesses with a high dollar volume of payroll and receipts were much more likely to respond than those with lower volumes. While the business listed in Position 10 had $100,000 in receipts and he was initially representing 6 businesses whose total receipts were approximately $600,000, the additional cases may have only increased the total for the 9 business block to $700,000. To produce better estimates for receipts, SMOBE/SWOBE set this respondent's receipts tabulation weight to 7 (700,000/100,000)– an increase of 7/6 or 1.167. Different weights were also calculated for employment and payroll, currently giving the record 4 different weights.

Since Hispanic- and Women-owned businesses crossed race categories, it is difficult to reassign cases in a manner that accurately compensates for race, Hispanic, and gender non-response simultaneously. Consequently, different reassignments were made for each group, and we ended up with a set of 12 weights per record. These weights were used to produce estimates for (1) number of firms, (2) aggregate receipts, (3) number of firms with paid employees, (4) aggregate receipts of firms with employees, (5) aggregate payroll, and (6) aggregate employment at each state and industry level for each of the 3 minority race groups, the Hispanic group, and the Women-owned group.

The Need for a Single Weight

Although the publications containing estimates by state and industry, by gender, race and Hispanic ownership are the major products from these surveys, the data are also used in a variety of other ways, in particular, special tabulations and microdata analysis (correlations, regressions, ratios, etc.) The 12-weight scheme does not work well for these secondary products. If, for example, someone wants a special tabulation of Black Hispanic-owned businesses, he would first extract the businesses which responded as both Black- and Hispanic-owned. He would then choose the "correct" set of weights (the 4 race weights or the 4 Hispanic weights). Since race-weighted and Hispanic-weighted data give different estimates, care must be taken to ensure that all users know which set of weights apply for their particular needs.

Now suppose a user wants to run a regression between payroll and receipts for API-owned firms. In a simple design, each respondent has a single different weight, say TABWT. The user may want to run a SAS **Proc Reg** with a "**weight tabwt**;" statement. Unfortunately **Proc Reg** cannot handle different weights for each variable on the same record. This problem would not exist if each record contained only a single tabulation weight.

The Problem Simply Stated

Given a set of K respondents that contribute to a building block, can we replace the 12 different tabulation weights on each record with a single tabulation weight TABWT and "nearly" preserve the 6 basic estimates (number of firms, aggregate receipts, ...) for each building block?

The SAS Solution Using an IML Routine

This problem is nothing more than a linear or non-linear programming problem which minimizes an objective function subject to a set of linear constraints. Rather than write such a routine from scratch, we searched and found several routines in the SAS statistical and data analysis libraries. All were written in Interactive Matrix Language (IML). We selected the IML routine **Proc NLPNMS** — Non-linear Programming, Nelder-Mead Simplex Method.

Proc NLPNMS has several arguments. You supply it with the data in matrix form, an initial guess, minimum and maximum acceptable values for each component of the final solution, and a single objective function. Using the matrix algebra methods, the routine invokes the multivariate equivalent of Newton's method for finding roots of functions in one variable to find a second solution. It then tests to verify that the value of the objective function has decreased. If it has not, it takes the initial solution as the optimal solution. If the value of the objective function has decreased by at least 0.0001, it substitutes the solution for the initial guess and starts over looking for an even better solution. It continues this process until it finds "the optimal" solution. In addition to the "optimal" solution, the routine also returns evaluation code which varies in value from -8 to +10. Negative values of this code are deemed to be unstable solutions, positive values are stable solutions. The higher the absolute value of the magnitude of the evaluation code, the more unstable or stable the solution.

Data Preparation

Having committed to use of **Proc NLPNMS**, we placed the data in the matrix format required by the SAS/IML routine. The steps below describe the sequence used.

Matrices B and A for The Linear Constraints: $AN_F = B$

This involved the following sequence of processes.

1 Construct **B** as a 6 x 1 matrix that contains the 6 basic published estimates of Building Block #1.

2 Identify the K respondents that contribute to Building Block #1.

3 Construct **D** as a 6 x K matrix that has unweighted data for each respondent listed as a column vector.

4 Construct **W** as a K x K matrix that contains the original sampling weight of each respondent, W_i, on the diagonal and zeros else where.

5 Set **A = DW**. **A** will be a 6 x K matrix. One column for each respondent in the block. The entries for Column I of Row j, will be the I-th respondent's data used to compile published estimate j, multiplied by the Respondent i's original sampling weight, W_i*Dji).

Note: $\mathbf{N_F}$ is the unknown matrix for which we need a solution. It is a K X 1 matrix, which contains the single weight adjustment factor for each of the K respondents. $\mathbf{N_F}$ contains the non-response adjustment factors, so the matrix $\mathbf{WN_F}$ will contain final adjusted weights of all respondents.

The Objective Function, ERR

Linear programming assumes that an "approximate solution" which nearly gives the correct result is a suitable substitute for the exact solution. For our objective function, we need to develop a function that determines how close the result of the "approximate solution" is from being exactly correct.

6 Suppose, for example, that we have an approximate solutions, $\mathbf{N_1}$, so that $\mathbf{AN_1} = \mathbf{B_1}$. **B**. We need a measure of the differences between $\mathbf{B_1}$ and **B**. A good candidate is the mean of the squares of the component-by-component differences, scaled by the value of the component of B,

$$\mathbf{ERR_1} = |\mathbf{B_1} - \mathbf{B}| = (3\ [(B_{1i} - B_i)/ B_i]^2) /6.$$

7 If there are M equations in T unknowns with M … T, there are an infinite number of solutions. The Euclidean distances in R^K between pairs of these may vary greatly. Suppose also that we have made a good initial guess for a solution, $\mathbf{N_0}$. We want to choose a solution which is very close to $\mathbf{N_0}$. We can calculate a second error function based on the Euclidean distance from $\mathbf{N_0}$ to any $\mathbf{N_1}$. If we scale by the number of cases **,** then $\mathbf{ERR_2} = |\mathbf{N_1} - \mathbf{N_0}| = 3(N_{1i} - N_{0i})^2 / K$.

8 Our main goal is to obtain the solution with the lowest value of $\mathbf{ERR_1}$. When multiple solutions with low values of $\mathbf{ERR_1}$ exist, we prefer the one closest to our initial guess. To get a single objective function, we introduce a scaling factor of one-millionth and add the two error functions to get the our final objective function,

$$\mathbf{ERR = ERR_1} + 10^{-6} * \mathbf{ERR_2}.$$

A Suitable Bounds Matrix for the Solution

9 In probabilistic sampling, a case with weight "W" means the responding case represents itself and "W-1" other similar cases. Weights less than 1.00 are inconsistent with this definition. Since the final weight of the j-th respondent is $N_{Fj} * W_j$, we want $N_{Fj} \geq 1/ W_j$. So $MIN_j = 1/W_j$.

10 For an upper bound, we recommend an integer multiple of the initial guess, $N_{Fj} \leq MAX_j * N_{0j}$. Using $MAX_j = 9999$ (essentially no upper bound). results in a large number of poor quality solutions. After experimenting with a variety of values for MAX_j, we settled an upper bound of $MAX_j = 5$ throughout. Although we held MAX_j constant, it may vary from observation to observation.

11 Construct the K x 2 bounds matrix by placing the MIN_j and desired MAX_j values (calculated above) in first and second columns, respectively, of the j-th row of the bounds matrix.

The Initial Guess Matrix $\mathbf{N_0}$

It only remains to deterimne the components, $N_{0j,}$ of the initial non-response guess matrix, $\mathbf{N_0}$, for each respondent. To do so, we used the following algorithm.

12 Determine the proportion that Case j contributes to each of the 6 published estimates (number of firms, total sales, ...) to get the set $\{p_{j1}, p_{j2}, ..., p_{j6}\}$.

13 Isolate the case to its own sub-building block (K = 1), and determine the value of N_{oj} which would minimize the square error, provided that each of the 6 variables had weight p_{ij}. That is, minimize $Q = 3_I (W_j N_{oj} - W_{ji})^2 * p_{ji}^2$; where W_j is the initial weight and W_{ji} is the weight used to tabulate the I-th estimate in the 12 weight scheme. You get

$$N_{oj} = 3_I [p_{ji}^2 W_{ji}/W_j] / 3_I [p_{ji}^2].$$

Invoke Proc NLPNMS to Solve Block 1

We are now ready to invoke **Proc NLPNMS** and solve our problem for Block 1. It only remains to execute the same process on the other 16,529 non-empty blocks. We obviously did not want to do these building blocks interactively. The next section describes setting up a %do loop macro for processing the entire set of respondents.

Using Proc NLPNMS Inside a Macro

Our current program will process one building block at a time. The following algorithm describes the procedure that SMOBE/SWOBE used to process all blocks with a single invocation.

1 Pass the file of respondents and assign each observation to the building block(s) to which it contributes. Sort and number these blocks sequentially from 1 to 16,530.

2 Append the building block number to each respondent record. If a respondent is in more than one building block. Replicate the record with each replicate having a different building block number. This will be our master input data set for the processing.

3 Set up a %do loop macro. The first operation will read the master input data set and pull off all respondent records whose building block number corresponds to the number of the iteration which is currently being performed.

4 The processing described in the Data Preparation section will then occur. At its conclusion, we will have solutions for all cases in this building block.

5 Before the next iteration of the %do loop begins,

use **Proc append** to append the solutions for cases in this block to those for respondents in the previous blocks. Each record contains (1) the respondent's unique identification number, (2) the building block number, (3) the solution (i.e., the final non-response adjustment factor), (4) the initial guess for the non-response factor, and (5) the initial sampling weight.

6 It is also a good idea to keep a performance log. It should contain one record for each block processed. Each record should contain (1) the block number (2) the number of records in the block, (3) the final value of the objective function, (4) the evaluation code returned by **Proc NLPNMS**, and (5) the clock time when the block finished. As with the solution data set, the log data set should be augmented with a **Proc append** statement before next %do loop iteration. **Note:** The importance of this log file will be illustrated later in the Results, Problems, Hints, ... section.

One More Subtle Twist

Suppose a business responded as Black Hispanic Female-owned. In the preceding macro section, we place it in three — a Black, a Hispanic, and a Female -- different blocks. Shouldn't that give us three different non-adjustment factors? The answer is "Yes". However, this nifty little trick fixed that.

Ideally, we should have used an **A** matrix with 18 rows — 6 for the basic race data, 6 for the basic Hispanic data, and 6 for the gender data. However, we worried that an 18 constraint problem would be very unlikely to converge for most blocks to a reasonable solution, since it would take at least 18 linearly independent observations to have a unique solution. Even if it did, the routine could possibly take a considerable amount of time to converge for each block. To ensure better and quicker convergence, we "subsetted" the 18-variable problem into a set of 6-variables problems as follows.

1 Process the Hispanic-owned blocks as described in the Data Preparation section. Determine the non-response adjustments for all businesses that responded Hispanic-owned. (Note that only a small percentage of each minority race-owned businesses are also Hispanic-owned.)

2a Process the Black-owned respondents. For Black non-Hispanic-owned businesses use the process described in the Data Preparation section. For Black-Hispanic-owned respondents, we already have a non-response adjustment factor N_{oj}. For these cases, set the initial guess and the two values in the bounds matrix to N_{oj}. This forces the routine to use the Hispanic non-adjustment factors for the Black Hispanic-owned cases and solve for suitable values for the N_{oi}'s of the Black-non-Hispanic cases.

2b Repeat process 2a for Asian/Pacific Islander-owned businesses.

2c Repeat process 2a for the American Indian/Alaska Native-owned businesses.

3 Process the Women-owned respondent cases. For White-non-Hispanic-Women-owned businesses, use the procedure described in the Data Preparation section. Otherwise, fix the non-response adjustment factor determined in either Stage 1, 2a, 2b, or 2c. For most of the Women-owned building blocks, note that at least 50 percent of the responding Women-owned businesses are also White non-Hispanic owned.

This trick requires that we run the macro three times instead of just one. It also requires that we write code to fix factors for some of the cases before starting the next macro. Nevertheless, it worked well. The observations with fixed factors did not appear to hinder cell convergence and it probably saved us a lot of headaches.

Results, Problems, Hints, Recommendations, and Questions

This section analyzes of our results, identifies situations where the software performed well and those where it did not, and offers some hints based on our observations from our analysis. The author welcomes other ideas and observations from readers who have also used SAS linear and non-linear programming routines.

How quickly does the software process a block?

This depends on the number of observations in the block. In general, the processing time is reasonable, provided the number of observations do not exceed 150. Table 1 was complied using information from the log file of the three most difficult groups to solve — Hispanics, API, and AIAN. (Blacks and Women building blocks solved more quickly, because their weights did not vary as drastically from variable to variable.) The table shows that the number of seconds required to process a block seems to increase exponentially with the number of observations.

We tried to process a cell with 1500 observations. It ran over 8 hours and had not completed when we terminated the run.

Table 1. Mean Times By Cell Size Hispanic, API, and AIAN 6,349 Non-Empty Blocks

Obs Per Cell	1 to 25	26 to 50	51 to 100	101 to 200	201 or more
Cells	4,943	722	515	129	40
Sec.	0.7	1.4	4.8	25.9	343.0

Does accuracy improve as the block size increases?
In general, it does. The more observations a cell has, the more degrees of freedom that the software can use to solve the problem. Consequently, expect better solutions as the number of observations increase. The statistics in Table 2 were obtained from information stored in the Hispanic, API, and AIAN log files. As you can see, the Relative Mean Square Error (RMSE) decreases at a logarithmic rate with the number of observations per cell. Additional reductions in RSME appear to be negligible for block sizes greater than 200.

Table 2. Mean RMSEs (%$\overline{ERR_1}$) By Cell Size Hispanic, API, and AIAN 6,349 Non-Empty Blocks

Obs/ Cell	1 to 25	26 to 50	51 to 100	101 to 200	201 or more
Cells	4,943	722	515	129	40
Mean RMSE (Pct)	10.5	5.0	2.7	2.2	1.3
Max RMSE (Pct)	505.2	63.2	64.4	74.6	12.3

Is there an optimal block size for large blocks?
Based on the information in the previous two tables, we recommend restricting blocks to 150 observations or less. At these sizes, each large block should converge quickly with average RMSEs of approximately 2.5 percent. For blocks with more than 200 observations, subset the blocks into pieces with 100 to 150 observations each. Then solve for each piece. Using these size restrictions for our SMOBE/SWOBE application, the software processes about 700 observations per minute.

What about accuracy at the published estimate level?
The RMSE is a composite estimate of accuracy over the 6 published estimates (number of firms, total receipts, ...). When we compare each published cell value with its value using the single weight convention, about 90 percent of the cells differ by less than 10 percent. Table 3 shows the accuracy comparison by race. As previously noted, the Black- and Women-owned businesses have weights which vary little from variable to variable. They converge quite nicely. The Hispanic-, and API-AIAN categories had weights with more substantial variation. This made convergence much more difficult. Table 3 illustrates the disparity well. While over 98 percent of the Women- and Black-owned cells converged to values within 10 percent of the published estimates, only about 70 percent of the Hispanic-, API-, and AIAN-owned cells converged within 10 percent of the corresponding published values.

Table 3. Percentage of Estimates Within a Percentage Difference On a Published Estimate by Estimate Basis

	Within 1%	Within 5%	Within 10%	Within 20%	Within 50%
Women	92	96	98	99	~100
Black	99	~100	~100	~100	~100
API-AIAN	34	55	71	86	98
Hispanic	35	52	67	84	98
Overall	77	85	90	95	99

Explain the problem with the Hispanic, and the API-AIAN rows in Table 3 further.
The Hispanic-, API-, and AIAN-owned rows give the perception that the software failed to give good results. The instability of the various weights, mentioned above, is only part of the problem. Each building block has 6 constraints, yet 3,179 (or 50.1percent) of these 6,349 non-empty blocks contained 5 or fewer observations. Consequently, there was no unique solution for half of these cells. Many of these probably have no very good approximate solution. If a good approximate solution exists, the software generally finds one. Note that 55 percent of the API-AIAN and 52 percent of the Hispanic cells converge totals within 5 percent of the published estimate, even though over half of the cells lacked a sufficient number of observations for a unique solution.

Does the accuracy of Proc NLPNMS compare with that of other commercially available software products?
Carl Eric Sarndal (1992) developed a routine for a similar weighting application for Statistics France. While at Statistics Canada, he and Victor Estavao (2000) developed software (known as CALMAR and CALJACK) which implements this method. The software is widely used in Statistic Canada programs, yet there are known instances where it did not work well. Whitridge (2000) used it to weight an e-business supplement to a capital expenditures survey. The results were so poor that they could not be used to tabulate the data. The U.S. National Agriculture Statistics Service (NASS) purchased the CALMAR and CALJACK software and regularly uses it to adjust their sampling weights. According to Phil Kott (2001), the program gives good results when applied immediately after sample selection to adjust weights for <u>all</u> units in the sample. When used only on the responding cases, it rarely works well. Consequently, We are happy with the results of the SMOBE/SWOBE application.

What caused cells to diverge badly when convergence was possible?
During the routine, **Proc NLPNMS** uses a generalized matrix inverse routine to "invert" the matrix **A.** This generalized inverse is used to determine the next solution. If **A** has any row vectors that are identically the **ZERO** vector, then the solutions tend to be really poor.

Hint: If you pre-identify building blocks with one or more variables that are identically zero for each observation,

you improve precision by separating these blocks and then

reformulating the problem so as to eliminate the zero rows in **A** and **B**. In our application, many of the Black- and Women-owned cells which did not converge contained no employer records. Hence the rows for employer firm indicator, employer receipts, payroll, and employment were all zero in the matrix **A**.

How important is the determination of a good initial guess?
Very important! The CALMAR software uses the vector with all entries set to 1 as its initial guess. We used the CALMAR initial guess for 39 different building blocks in Hawaii. We then compared those solutions with solutions using our initial guess. For 10 (or 26 percent) of the blocks, the solutions were equally accurate. The SMOBE/SWOBE initial guess produced much more accurate results for 26 (or 67 percent) of the blocks. The CALMAR guess only outperformed the our guess for 1 block. There was also 1 block where neither guess gave a good solution.

Any advice for setting the upper and lower solution bounds?
The bounds matrix is optional. We strongly recommend using one. In a probabilistic survey, you must use it to ensure that the final weights are all greater than 1. At first we were very lenient in setting the upper bound to a value of 9,999 times the initial guess. This gave very poor convergence for a large number of blocks. We experimented with several values on 2,289 non-empty Black-owned blocks, which we knew should converge well. We then tried multipliers of 99, 6, 5, 4, and 3. Using the multiplier of 5, we maximized the percentage of blocks that converged within 1 percent to 99.8. After testing the same multipliers with very similar results on non-Black-owned business cells, we settled on an upper bound of 5 for the SMOBE/SWOBE application. **This is an area where the author would greatly appreciate feedback from those who have developed an algorithm for setting these bounds effectively.**

Any hints on setting the objective function?
We had two objective functions. ERR_1 measured the relative distance from the result of the approximate solution to the desired result. ERR_2 measured the distance from the initial guess to the solution. Since ERR_2 was the less significant of the two, we scaled it by a factor of one-millionth and added it to ERR_1 to get the resulting single objective function.

In our opinion, the term 10^{-6}*ERR_2 had almost no effect on the final solution. Although we have done no further research in this area, we are left with the following questions:

1 Should we have increased the scaling factor to 10^{-4} or 10^{-3}?
2 Should we have dropped the ERR_2 term completely?
3 Are there another objective functions which give a low ERR_1 value and do not allow the final solution to stray from the initial guess?

If any of the readers have any suggestions or results, please feel free to contact the author.

Conclusions

It takes a considerable amount of programming to transform the data into the IML format required by the integer programming routine **Proc NLPNMS**. However, its overall performance for solving the 1997 SMOBE/SWOBE multiple to single weight problem was more than adequate. After restricting the number of linear constraints to under 150 per system, the routine provided stable solutions for most blocks in a reasonable amount of time. Further analysis of the blocks with unsatisfactory results provided methods for identifying these situations before they were processed, and modifying the data preparation routine so that **Proc NLPNMS** would give stable solutions for these systems also.

References

Deville, J.C. and Sarndal, C.E. (1992) . Calibration Estimators in Survey Sampling. *Journal of the American Statistical Association.* pp 376-382.

Kott, Philip S. (Feb. 16, 2001). Comments made during a Washington Statistical Society short-course on *An Introduction to Model -Based and Model-Assisted Sampling Theory* at the Bureau of Labor Statistics, Washington, DC.

Sarndal, Carl-Eric and Victor Estevao (2000). A Functional Form Approach to Calibration. *Journal of Official Statistics, Vol 16, No. 4 (Dec. 2000)*. Stockholm, Sweden. (to appear)

Whitridge, Patricia (Oct. 23, 2000). Comments made during the 2000 US Bureau of the Census / Statistics Canada Interchange during a talk on *Internet Sales Supplemental Survey to the Capital Expenditures Survey* at the US Census Bureau, Suitland, MD.

Disclaimer

This paper reports the results of research and analysis undertaken by Census Bureau staff. It has undergone a more limited review than official Census Bureau publications. This report is released to inform interested parties of research and encourage discussion.

Contact Information

For more information about the surveys or the techniques described in this paper, feel free to contact the author. The author also values and encourages comments and questions.

Richard A. Moore, Jr.
U.S. Bureau of the Census
Company Statistics Division — 6400
Washington, DC 20233-6400
Phone: 301-457-3313
Fax: 301-457-3396
E-mail: Richard.A.Moore.Jr@census.gov