An Application of SASTM Software in Auditing and Finance:
Monetary Unit Sampling
Pankaj N. Patel, General Electric Corporation, Albany, NY
Laura G. Bacon, General Electric Corporation, Albany, NY

Abstract
This paper describes an interactive use of SAS for the auditing and finance area. Monetary Unit Sampling is described in detail to show SAS's efficiency in programming and sampling. SAS's built-in functions and procedures make the program efficient and easy to understand. The program was written and developed on the IBM-PC, MSDOS, using base SAS for the personal computer version 6.04 and IBM 3090, MVS, running on base SAS Version 6.0.

Introduction
In recent years more public accounting firms and internal audit departments in large companies in the USA and Canada have introduced Monetary Unit Sampling as a standard statistical technique in their auditing department.

Sampling is one of the most efficient tools auditors use in their work. Today almost all auditing is conducted on a sampling basis. For financial applications, simple attribute sampling is modified to calculate possible overstatements/understatements in accounts such as receivables, payables, inventory, etc. This modified attribute sampling is called Monetary Unit Sampling (MUS). In MUS the sampling unit is a dollar rather than an invoice or an item. It is one of the principal sampling methods to assess audit risk. With the help of statistical sampling substantive sampling risk can be quantified and controlled.

MUS is best explained by an example. Assume we have the following information about an account to be audited.

Total Receivables: $1,000,000
Number of invoices: 10,000
Sample Size: 100
Beta risk: .05

If we consider a dollar as a sampling unit, and take a random sample of 100 individual dollars and find no errors, we can make the statement that "We are 95 percent confident that there is no more than a 3.0 percent error in $1,000,000 dollars or the total error does not exceed $30,000 dollars. Note that MUS allows us to statistically state the conclusion in dollars. There are various methods to evaluate monetary unit sampling results, however, we will only discuss the Stringer method, which is most commonly used.

For MUS selection each invoice is weighted by the dollar amount of the invoice, i.e. a $200 dollar invoice will have twice the chance of selection into the sample as a $100 dollar invoice. We accomplish this by randomly selecting individual one-dollar bills rather than invoices. But as we cannot verify the individual dollar by itself, we select and audit the invoice attached to the dollar selected in the sample. There are various methods by which we can select an MUS sample. We will discuss the "interval" method of sample selection.

Sample Selection
The SAS program to select an MUS sample uses the "interval method" of sample selection. In this method every nth dollar is selected for the sample and the physical unit attached to that dollar is included in the sample file. The sampling interval is calculated by dividing the total debits in the population by the calculated sample size. (Credits may also be sampled but they must be sampled separately.)

The first section of the selection program interactively asks for information about the population to be sampled. The following screen is produced using the WINDOW statement.

Exhibit I(Select Screen I)

This window asks for the name of the SAS dataset containing the population, the variable name which contains the dollar value to be sampled, the DOS filename to output the sample, the DOS filename to output the sample documentation and a brief description of the population. This information is saved as macro variables and used later to document the sample.

The next screen requests the sampling interval and a seed for the RANUNI function used later in the program.

Exhibit II(Select Screen II)

The main body of the program is the selection procedure itself. It uses base SAS to stratify the population into debits, credits, units with a value of zero and units with a value greater than the sampling interval(100% item). Only the debits are sampled.
The program starts with a random number between zero and the negative value of the sampling interval. This number is used to initialize an accumulator variable (i).

```sas
  data &output (drop=i start seed credit hpct debit kmnz ndn sample select total o si);
  set &infile end=j;
  if _n_=1 then set work.data1;
  if _n_=1 then do;
    start=round(ranuni(seed)*si,2);
    i=start;
  end;
```

It then runs through the population dataset sequentially checking to see which category the item falls into. If it is a zero, a credit, or 100% item, the number of records and total book value information is accumulated. Items greater than sampling interval are also automatically included in the sample without a change to (i).

Debits are added to (i) until (i) becomes positive. The item that causes (i) to become positive is included in the sample and written to the sample file. The sampling interval is subtracted from (i) to make (i) negative again.

```sas
  m+1;
  sample + &bookval;
  i + &bookval;
  if i>=0 then do;
    output;
    o + 1;
    select + &bookval;
    i + -si;
```

The process repeats itself until the last item in the dataset is read.

When the end of the population file is reached, the program documents the sample for audit workpapers.

Exhibit III (Sample Log)

This documentation is written to a DOS file for storage and is useful in the event that the sample has to be recreated. The sample files are given to the auditor and can also be saved for documentation.

Sample Evaluation

This program extrapolates the results of the sample to the population using the Stringer method of evaluation. Depending on the results of the evaluation, an auditor will conclude whether or not the overstatement/understatement in the account is material to the business.

The Stringer method of MUS uses a modified form of attribute sampling based upon the binomial distribution. While in classic attribute sampling a unit is either right or wrong, black or white, in the Stringer method and other tainted dollar approaches a unit may be partially correct. This amount of partial correctness is called a taint. (The tainting percentage is the absolute value of the book value minus the audit value, divided by the book value.) The Stringer method differs from other MUS evaluation methods in that it ranks errors in descending tainting percentage.

The evaluation begins with interactive input of the sample results and details. The first screen asks for the name of the SAS dataset containing the sample errors. (Items where the audited value did not match the values on the books.) The dataset must contain variables called bv (book value) and av (audit value) but may also contain other variables. This screen also asks for a DOS filename to output the results and a brief description of the sample.

Exhibit IV (Evaluation Screen I)

The second screen asks for information about the population. Most of this information is available from the log printed in the selection program. This screen also asks for the beta risk which is the risk that the actual error in the population may exceed the projected upper error limit. (Confidence is equal to 1-beta risk.)

Exhibit V (Evaluation Screen II)

Using this information, the program uses base SAS to extrapolate the results of the sample. The first step is to separate the errors into overstatement (bv > av), understatement (bv < av), and errors from 1.00% items and calculate the taints (abs(bv-av)/bv). The maximum number of errors for a category is found using the MAX function and the taints are sorted from largest to smallest.

The heart of the program is the calculation of the precision gap widening factors (PGW). A PGW factor is the incremental amount of error that is extrapolated to the population due to an additional error in the sample. When no errors are found in the sample, the PGW factor represents the sampling precision and is usually called the basic precision (BP). This section of code uses the binomial probability function (PROBBNML) to find the error rate at which the binomial probability is equal to beta given m errors in a sample size of n. (The Poisson distribution is often used as an approximation for calculating PGW, however, the PROBBNML function allows for more precise calculations and is an advantage of using SAS over other software languages.)
/*CALCULATES PGW FACTORS*/
data work.pgw(keep=pgw);
set work.details;
set work.ref;
uel=0; p=.5;i=.5;
do m=0 to number;
i=(1-p);
/*Use trial and error with error rate (p%) to find the binomial probability which
is equal to beta given m errors in a
sample of n*/;
do until(abs(phat-beta)<.00000001);
i=il2;
phat=probbnml(p,n,m);
if phat<beta then p=p-1;
else p=p+1;
end;
pgw=n*p-uel-min(m,1);
uel=n*p;
output;
end;
run;

Once the error rate is found the PGW is calculated by multiplying the sample size by the error rate to get the maximum number of errors in the population (Upper Error Limit). Because PROBNML is a cumulative function, the prior UEL and 1 (when m<>0) is subtracted from this calculation. The process is repeated until the number of PGWs generated equals the maximum number of errors for overstatement or understatement.

Using the PGWs, separate most likely errors (MLE, upper precision limits (UPL), and upper error limits are calculated for overstatement and understatement.

upl=taint"pgw"sl;
if _n_ =1 then mle=0;
else mle=taint"sl";
uel=mle+upl;
smle+mle; tupl+upl;uel+uel;
output work.puel;

Finally, a most likely error and adjusted precision and upper error limit bounds are calculated combining the results of the overstatement and understatement errors. Using the method suggested by Leslie, Teitlebaum and Anderson, the program nets the over and under statement MLEs and reduces the UPL and UEL by the MLE in the opposite direction.

The final piece of the program prints out the results of the evaluation to the screen and to the DOS filename designated in the first screen. Like the selection program, the evaluation program uses macro variables for filenames and descriptions.

Exhibit VI (Evaluation Results)

Other Applications

MUS is just one of the statistical audit techniques for which we have SAS programs. Additional programs include determining binomial and hypergeometric sample sizes, selecting samples and evaluating the results of these samples, producing sequential sampling plans, random sampling and sample selection for book/line data.

The flexibility of SAS to read and summarize a variety of data makes it an ideal tool for the diverse environment in which we work.

References


SAS™ is a registered trademark of SAS Institute in the USA and other countries. ™ indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.
This program selects a MUS sample using the interval method. To use this program you should have your data already in a SAS data set. Please enter the following information:

- Brief name to appear in log: SAMPLE FOR SUGI 17
- Name of SAS data set to be sampled: SMPSUGI
- Variable name containing the Book value: AMOUNT
- File name of sample output file: SMPOUT
- File name for documentation output (in quotes): SMPOOC

Exhibit I (Select Screen I)

Please enter the following data:

- Sampling Interval: 12500000000
- Enter a random seed between 1 and 99,999,999 808480

Exhibit II (Select Screen II)

MUS selection log for SAMPLE FOR SUGI 17
Population file name: SMPSUGI
Book Value variable Name: AMOUNT
Sample written to: SMPOUT
Sampling Interval: $125,000,000.00
Random Seed 808,480

Zero Balances
Credit 32
Debits 627
$11,004,433,285.46
$8,217,616,005.01
$3,686,817,280.45
$3,424,130,091.43
$5,119,004,433,284.46

Exhibit III (Sample Log)
You should have your error data in a SAS data set with the variables bv for the book value and av for the audit value.

File containing error data: SUG117P

Type CTRL-BREAK to abort if you have not created this file.

DOS filename to put output (in single quotes): 'SUG117E'

Brief description of sample: SUG 17 EXAMPLE

Exhibit IV (Evaluation Screen I)

This program calculates the Stringer bounds for MUS samples for both over and understatements using the binomial distribution for calculating the precision gap widening factors.

Please enter the following information:

Total Book Value of Account Sampled
Including Items Sampled 100%: 1190433285.46

Book Value of Items Sampled 100% 3686817280.45
Sampling Interval: 12500000

Beta Risk: .05

NOTE: The program assumes that all items greater than the sampling interval or gauge are selected 100% if a different criteria was used, do not include these items in total book value and put 0 for 100% items.

Exhibit V (Evaluation Screen II)
Input Data:

Input file name: SUG17P

<table>
<thead>
<tr>
<th>obs</th>
<th>Book Value</th>
<th>Audit Value</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$21,538,541.31</td>
<td>$16,153,905.98</td>
<td>$5,384,635.33</td>
</tr>
<tr>
<td>2</td>
<td>$61,525,897.20</td>
<td>$49,220,717.76</td>
<td>$12,305,179.44</td>
</tr>
<tr>
<td>3</td>
<td>$411,963.31</td>
<td>$5,149,791.13</td>
<td>$-4,737,827.82</td>
</tr>
<tr>
<td>4</td>
<td>$45,000,000.00</td>
<td>$54,000,000.00</td>
<td>$-9,000,000.00</td>
</tr>
<tr>
<td>5</td>
<td>$71,101,362.35</td>
<td>$79,191,456.80</td>
<td>$-8,090,094.45</td>
</tr>
</tbody>
</table>

| Total Book Value of Items Selected 100%: $3,586,817,280.45 |

| Sampling Interval: $500,000,000.00 |

| Beta Risk: 0.050 |

Stringer Bound On Over Statement Errors:

<table>
<thead>
<tr>
<th>obs</th>
<th>Most Likely Error</th>
<th>Upper Precision Limit</th>
<th>Upper Error Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$0.00</td>
<td>$366,095,152.68</td>
<td>$366,095,152.68</td>
</tr>
<tr>
<td>1</td>
<td>$31,250,000.01</td>
<td>$213,329,980.28</td>
<td>$52,582,980.29</td>
</tr>
<tr>
<td>2</td>
<td>$25,000,000.00</td>
<td>$12,064,927.77</td>
<td>$37,006,927.77</td>
</tr>
</tbody>
</table>

| Upper Precision Limit: $56,250,000.01 |

| Upper Error Limit: $599,492,660.73 |

Stringer Bound On Under Statement Errors:

<table>
<thead>
<tr>
<th>obs</th>
<th>Most Likely Error</th>
<th>Upper Precision Limit</th>
<th>Upper Error Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$0.00</td>
<td>$366,095,152.68</td>
<td>$366,095,152.68</td>
</tr>
<tr>
<td>3</td>
<td>$1,437,499,925.66</td>
<td>$981,317,041.59</td>
<td>$2,418,816,967.26</td>
</tr>
<tr>
<td>4</td>
<td>$25,000,000.00</td>
<td>$12,064,927.77</td>
<td>$37,006,927.77</td>
</tr>
<tr>
<td>5</td>
<td>$14,222,818.98</td>
<td>$5,466,444.75</td>
<td>$19,689,263.73</td>
</tr>
</tbody>
</table>

| Combined Most Likely Error (- = understatement): $-1,420,472,744.63 |

| Adjusted Stringer Bound for Overstatement Errors: |

| Upper Precision Limit: $-1,020,980,083.90 |

| Upper Error Limit: $-964,730,083.89 |

Exhibit VI (Evaluation Results)