User Manual

for

PovMap

Version 1.1a

Qinghua Zhao

Development Research Group

The World Bank

1818 H Street, N.W.

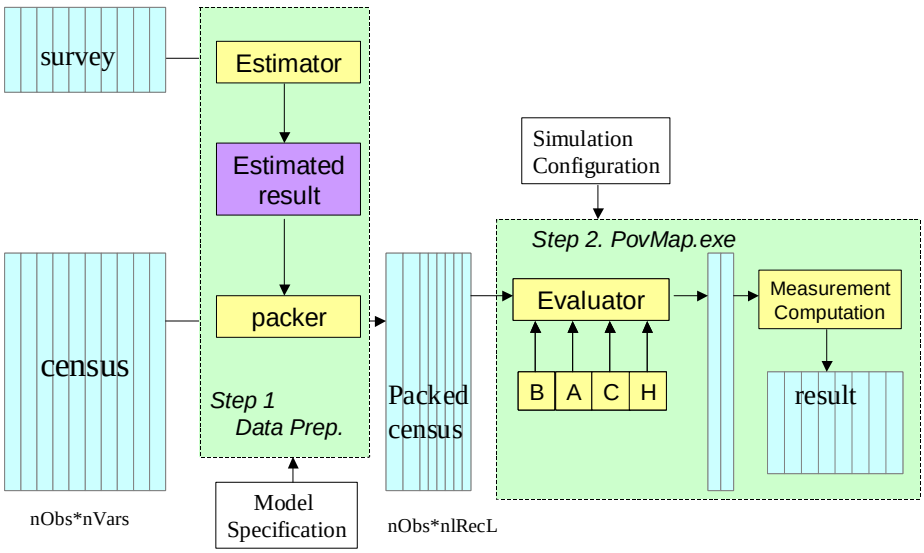
Washington, D.C. 20433

Introduction

PovMap is a software package that computes poverty and inequality indicators at a spatially disaggregated level. "Poverty mapping" is a method that uses a model of household expenditure model from a survey dataset to estimate household welfare in a census dataset which typically do not include household expenditure or income information. Poverty indicators at the community level are then formed as aggregats. Bootstrapping is used to improve the accuracy of the estimation.

The method consists of two stages. During the first stage, a series regression is run to model the expenditure and decompose the random unexplained component. In order to apply this to the census data, the regressors in the model need to exist both in the survey and census datasets. A special dataset that combine census data and model parameters is then produced. The second stage of poverty mapping is the simulation stage. This stage uses the model parameters but performs repeated drawings on different random components to bootstrap the household expenditure.

The basic structure of this package is shown in the chart below. The "data packer" estimates the model and organizes the information for simulation. Two packers are provided for the user, one that uses SAS software and the other that does not. SAS users will use a set of SAS programs to handle all tasks in stage 1. Others will use a stand alone executable *PovMapPacker.exe* to prepare their data.



Whay this design ?

1. SAS is widely used to perform data processing of census data.
2. SAS has all the necessary modules built in to perform tasks such as GLS and singular decomposition.
3. Without SAS software it is difficult to read SAS datasets. SAS users must perform the data packing within SAS.
4. Users who do not have access to SAS must utilize other statistical packages for model building. Presumably their dataset is much smaller. The stand alone data packer PovMapPacker.exe can access Stata, DBF and ASCII formats.
5. The packed dataset between stage 1 and stage 2 can be made more compact to allow the dataset to be fully loaded into memory. This option is much quicker than throughdisk operation.
6. A stand alone EXE executable can efficiently satisfy these demanding computations.

It is necessary for the poverty mapping software to run efficiently during the simulation stage to ensure its success. Our version of PovMap requires, at most, a few minutes to run. When executing the PovMap software, most of the computer's system resources will be occupied, however the user will still be able to perform other tasks and switch between programs.. PovMap can also be run in conjunction with other processes that are currently reading the same data file and outputting the result to that same file.

This package is designed to be used by the general user. Users do not need to be able to program in SAS as long as an easy specification file can be made.

Computing Model

In this section the computing process for poverty mapping will be summarized. Users of this manual should always refer to the paper by Elbers, Lanjouw and Lanjouw (2001) for theoretical background and statistical inference.

The computing of poverty mapping begins during the estimation of the expenditure function. For simplicity, we assume per capita expenditure of a household is the basic left hand side variable

and the word ‘cluster’ is an aggregation level in survey and census datasets.

(1) $\ln y_{ch} = E[\ln y_{ch} | \mathbf{x}_{ch}] + u_{ch}$

where

c is the subscript for the cluster

h is the subscript for the household within cluster *c*.

y_{ch} is the per capita expenditure of household *h* in cluster *c*.

$\mathbf{x}_{ch}$ is the household characteristics for household *h* in cluster *c*.

a linear approximation of model (1) is then written as:

(2) $\ln y_{ch} = \mathbf{x}_{ch}'\boldsymbol{\beta} + u_{ch}$ *(also referred to as **Beta** model)*

since survey data is just a sub sample of the whole population, the location information is not available for all regions in the census data. Thus we cannot include the location variable in the survey model. Thus, the residual of (2) must contains the location variance.

(3) $u_{ch} = \eta_c + \varepsilon_{ch}$

Here η_c is the cluster component and ε_{ch} is the household component. As mentioned above, the estimate of η_c for each cluster in the census dataset is not applicable, therefore we must estimate the deviation of η_c. Taking the arithmetic expectation of (3) over cluster *c*

(4) $u_{c.} = \eta_c + \varepsilon_{c.}$

Hence

$$\mathrm{E}[u_{c.}^2] = \sigma_\eta^2 + \mathrm{var}(\varepsilon_{c.}) = \sigma_\eta^2 + \tau_c^2.$$

Assuming η_c and ε_{ch} are normally distributed and independent each other, Elbers et al gave a estimate of variance of the distribution of the locational effect η_c:

(5) $$\mathrm{var}\left(\widehat{\sigma_\eta^2}\right) \approx \sum_c [a_c^2 \mathrm{var}\left(u_{c.}^2\right) + b_c^2 \mathrm{var}(\widehat{\tau_c^2})] \approx \sum_c 2[a_c^2\{(\widehat{\sigma_\eta^2})^2 + (\widehat{\tau_c^2})^2 + 2\widehat{\sigma_\eta^2}\widehat{\tau_c^2}\} + b_c^2 \frac{(\widehat{\tau_c^2})^2}{n_c - 1}].$$

When the location effect η_c does not exist, equation (3) is reduced to $u_{ch} = \varepsilon_{ch}$.

According to Elbers et al, the remaining residual ε_{ch} can be fitted with a logistic model and will

regress a transformed ε_{ch} on household characteristics:

(6) $$\ln\left[\frac{e_{ch}^2}{A-e_{ch}^2}\right]=\mathbf{z}_{ch}^T\widehat{\boldsymbol{\alpha}}+r_{ch}.$$ (*also referred to as **Alpha** model*)

where A set to equal 1.05*max{ε_{ch}^2}. The variance estimator for ε_{ch} can be solved as

(7) $$\widehat{\sigma}_{\varepsilon ch}^2=\left[\frac{AB}{1+B}\right]+\frac{1}{2}\widehat{\mathrm{Var}}(r)\left[\frac{AB(1-B)}{(1+B)^3}\right].$$

The result from above indicates a violation of assumptions for using the OLS in model (2), so a GLS regression is needed. In GLS the variance-covariance matrix is a diagonal block matrix with structure:

(8) $$\begin{bmatrix}\sigma_{\eta c}+\sigma_\varepsilon & \sigma_\varepsilon & \sigma_\varepsilon & \sigma_\varepsilon\\ \sigma_\varepsilon & \sigma_{\eta c}+\sigma_\varepsilon & \sigma_\varepsilon & \sigma_\varepsilon\\ \sigma_\varepsilon & \sigma_\varepsilon & \sigma_{\eta c}+\sigma_\varepsilon & \sigma_\varepsilon\\ \sigma_\varepsilon & \sigma_\varepsilon & \sigma_\varepsilon & \sigma_{\eta c}+\sigma_\varepsilon\end{bmatrix}$$

Overall, the procedure for stage 1 of of the poverty mapping computation can be listed as:

s1. estimate *"Beta"* model (2)

s2. calculate the location effect η_c (3)

s3. calculate the variance estimator $\mathrm{var}(\sigma_\eta^2)$ (4)

s4. prepare the residual term ε_{ch} for estimating *"Alpha"* model (6)

s5. estimate GLS model with (8)

s6. use a singular value decomposition to break down the variance-covariance matrix from previous step. This will be used for generating a vector of a normally distributed random variable such that the joint variance-covariance matrix will be in the form of (8)

s7. read in census data, eliminate records containing missing values, generate all census variables needed for both *Beta* and *Alpha* models.

s8. save all datasets needed for the simulation (the "PDA" file).

Dataset and Memory Usage

The dataset generated during the data preparation stage is in a proprietary format with an extension of PDA. The dataset includes the regression results from the *Beta*, and *Alpha* models, decomposed

variance-covariance matrix from step s6, decomposed variance-covariance matrix from step s4, household count of each cluster and other parameters estimated in the data preparation step, and finally, the census data in binary format. The goal is to organize all intermediate results into one file which would then take up less hard disk space.

Even though users may never need to see the binary dataset directly, it is still necessary to explain the structure of this dataset. Since we are dealing with census data the program should not be limited by the size of this data For the sake of computing speed the data will be read into memory, however this is always limited. Thus, it is important to compress the data efficiently. In this package, a 'bit' type variable is provided implying this variable will occupy only one bit. Since there are typically a lot of dummy variables within the census data, using the *bit* data type will allow us to compress up to 8 dummy variables in to one single byte. To further conserve space, the cluster ID is not stored as a column since they are constant within the cluster.

PovMap may pack the cluster level information separately to save space. Cluster only variables are those that are invariant within a cluster, and do not need to be repeated in the household record. In PovMap.exe, cluster-only variables will be added to household record. The savings associated with using cluster-only variables is tremendous. For example, the PovMap dataset for South Africa stratum 7 with 1.8 million household and 10 variables is barely 25M.

In order to achieve the fastest speed and most efficient use of memory, four memory usage modes were designed into the PovMap.exe simulator. They will be selected by automatically program depending on the available memory set by the user. These four modes are:
Mode I--data is internally organized as a matrix of X and Z in double precision. Demands $n*8*(m_a+m_b+2)$ bytes of memory. When n=1,000,000, m_a=20 and m_b=10, PovMap needs at least 256 Mega bytes of memory. This is the fastest mode.

Mode II--demands much less memory as it needs only $n*(m_t+16)$ bytes of memory. Here m_t is the record length of census data. If m_t=60, and n=1 million, PovMap needs 76 Mega bytes of memory space. This mode is about 1.5 times slower than the mode 1.

Mode III--the highest memory saver but the slowest mode. PovMap needs only n*16 bytes of memory. Data will be read directly from the census data and processed on the fly. The speed of mode III is dependent on the computer's throughput. Thus, when a larger dataset is used, it is better to process it on the local drive.

Stage 1. Preparing Data

As previously mentioned, there are two packages users can select from when preparing a PovMap dataset. SAS allows users to perform data preparation and process census data The necessary components are SAS/Base, SAS/IML and SAS/Stat. For those who do not use SAS for census preparation, we also provide a tool that can read datasets in Stata, dBase and ASCII formats.

Data Preparation with SAS

Users do not to be SAS experts as long as the model can be specified into the following format:

```
*************************************************************************
*            Data preparation for PovMap
*************************************************************************;
%include "C:\projects\PovMap\fileaux.sas";
%include "C:\projects\PovMap\dataprep.sas";

*============= Specify the Survey information ==================;
**** dir of survey data **;   %let sdir=;
**** input dataset name **;   %let srvdata=LSMSGABE;
**** clustering variable *;   %let Cluster=CLUSTER;
**** survey weight *******;   %let sWeight=FACTORES;
**** LHS variable ********;   %let lhs= LRPCEXP;
**** Beta RHS variables **;   %let rhs=
EDUCHD HEADAGE NOWIFE EDUCWF WIFEAGE PPD YGIENE1
TOPER2 TOPER3 PPD2 PPD3 CMEAN2 CMEAN3 CMEAN25 CMEAN40 SCMN22;
**** Alpha RHS vars ******;   %let arhs=VAR4 VAR11 XBETA4 S12 S112 S23 S45 S46;
**** Locational effect? **;   %let LOCERR=YES;

*================ Specify the census information ===============;
**** dir of Census data***;   %let cdir=;
**** input dataset name **;   %let cendata=smallcengabe;
```

```
**** census weight  ******;   %let cWeight=OPERSON;
**** ID Vars in census ***;   %let cKeyVar=Cluster;
**** Cluster only vars ***;   %let cOnlyVar=CMEAN2 CMEAN3 CMEAN25 CMEAN40 SCMN22;
**** Cluster only file ***;   *%let cOnlyDat=cclusterOnly;

*================= Output Directory ============================;
**** output directory ***;    %let outdir=;
%let LOCERR=Yes;
%let dataout=small;
%dataprep;
```

The above program is written in SAS macro language. In this SAS code, statements starting with '*' and end with ';' are a comment statements, and have no programming meaning. SAS will ignore all comment lines. The meanings of other statements are explained below:

%include statement will retrieve an external SAS code from a specified file. Two files to be included are FileAux.sas and DataPrep.sas. Full path specification is allowed. If the user places these two file into SAS's macro directory, these two %include statements can be omitted.

%let SDIR=*SurveyDirectory*; specifies the directory of the survey dataset. If the survey dataset is in the same directory leave it empty between '=' and ';'. Please note that no quote sign is needed (even if there are spaces in the directory name), i.e. **sDir="c:\temp";** will cause problems but **sDir=c:\temp;** is correct.

%let SrvData=*SurveyDatasetName;* to specify the name of survey dataset (in SAS format). Must be supplied.

%let Cluster=*NameOfClusterID;* to specify the name of cluster ID. This variable should exist in both the survey and census datasets (and have the same name).

%let sWeight=*NameOfWeightingVariableInSurvey;* to specify the name of the weighting variable within the survey, typically compounded by household size and cluster weight.

%let LHS=*PerCapitaExpenditureInLogrithm;* to specify the name of the variable that holds the per capita expenditure figure in logarithm .

%let RHS=*List of Variables in Beta model;* to specify the variable names in *Beta* model. The variables must exist in the SAS dataset.

%let aRHS=*List of Variables in Alpha model;* to specify regressors in *Alpha* model. Regressors can be either a variable that already exists or an expression that uses variables in the survey dataset. Variable name *Yhat* can be used in the formula to represent the predicted value of *Beta* model.

%let CDIR=*CensusDirectory*; to specify the directory where the census data resides. Full path name is allowed. Omit the path name if the census dataset is in the same directory as this code.

%let CenData=*CensusDataset;* to specify the name of census dataset;

%let cWeight=*NameOfWeightingVariableInCensus;* to specify the weighting variable in census data. Must exist already.

%let cKeyVar=*NameOfIDinCensus.* If the cluster ID variable already exists in the census dataset, place its name here. This can only be used when the cluster ID in the census is different from the cluster ID in the survey. cKeyVar need not to exist in survey dataset.

%let cOnlyVar=*Variables* Optional. It defines the variables at cluster level and does not vary within a cluster. Variables of this type will be stored separately and restored during simulation.

%let cOnlyDat=*DatasetNameWithCOnlyVar*. cOnlyVar as defined above need not exist in the census data because it will significantly increase its size. This dataset can be read in from another (census cluster mean) file with this option.

%let OutDir=*OutputDirectory;* to specify a directory to store the output datasets.

%let DataOut=*OutputPovmapDataFile;* the output dataset will have extension PDA.

%let LocErr=*YesOrNo;* to specify whether the locational random component should be modeled. When equal to NO, the location effect will not be modeled.

%Dataprop; the last statement to execute the SAS macro %DataPrep.

When editing this file, the case of variable names or reserved words are not important, and neither is their order except the statement %DataPrep which must be the last. Please make sure that the name YHAT must not exist in the survey dataset because it is reserved to represent the estimated per capita expenditure. Names start with _Z and followed by a number should also be avoided because they are internal variables for the *Alpha* regression.

Data Preparation without SAS

When SAS is not available or the census dataset can be processed by another package, PovMap supplies --PovMapPacker.exe-- to prepare the dataset. **PovMapPacker** performs the same task as the version with SAS. Currently three types of datasets can be used: Stata (any version), dBase(III and IV) and ASCII with header line. To use **PovMapPacker** the user needs to prepare a model specification file which looks like:

```
*****  Model *****
srvdata=LSMSGABE4.dta
lhs=LRPCEXP
rhs=EDUCHD HEADAGE NOWIFE EDUCWF WIFEAGE PPD YGIENE1
    YGIENE2 YGIENE3 MATER1 MATER3 COCINA1 COCINA4 VIVI1 VIVI2 NATIVE OPERSON
    TOPER2 TOPER3 PPD2 PPD3 CMEAN2 CMEAN3 CMEAN25 CMEAN40 SCMN22
arhs=VAR4 VAR11 XBETA4 S12 S112 S23 S45 S46 S49 S55 S810 S314 S417 S420 S421
    S721 S1019 S1121 S1219 S1221 _Yhat_*_Yhat_
Cluster=CLUSTER
sWeight=FACTORES
CenData=smallcengabe.dat
cWeight=OPERSON
cKeyVar=CLUSTER
LocErr=YES
DataOut=small
```

The specification is identical to the SAS version. All previous explanations for SAS apply here except the following:

sDir= or cDir= clause is not needed. User can put the path of file in front of the dataset name; i.e.`srvdata=c:\project1\stratum3\LSMSGABE4.dta`. However, cOnlyVar is not supported.

The following dataset formats are supported by PovMapPacker:
Stata file (*.DTA) Any Stata format between version 2.0 to 8.0 is supported. However, the Stata file generated with an Unix system may not be read by PovMapPacker.
dBase file (*.DBF). DBF III or IV are supported including other variations such as Fox pro's DBF file.
ASCII file with header. This is an ASCII file in which fields are separated by comma ',', missing value is coded with no symbol. The following example has a missing value on field INCOME at first record:

```
HHID,HHAge,INCOME,EXPEND,EDUC,HHSIZE
100101,35,,3000,10,4
100102,50,2000,1600,7,5
```

Please keep in mind that all records containing missing values will be dropped.

Compounded ID in Census

Cluster ID is the ID that identifies the lowest level in survey dataset and is typically the county level. It may not be the lowest level in which you want to produce the poverty map measurements. Even though PovMap provides aggregations for the cluster level and the levels below, going lower than cluster level is typically associated with larger standard error.

Only one identifier is allowed in the PovMap's "PDA" dataset in order to save space and to run faster. This identifier could be as long as 16 digits and is typically compounded with identification of different administrative units. Suppose the dataset (survey and census) have multiple identifiers such as STRATUM (ranged from 1 to 9), COUNTY (ranged from 1 to 99), DISTRICT (ranged from 1 to 125) and TOWNSHIP (ranged from 1 to 999), a compounded ID at district level can be created with

```
DISTID=(STRATUM*100+COUNTY)*1000+DISTRICT
```

or

```
DISTID=STRATUM*100000+COUNTY*1000+DISTRICT .
```

Similarly, a compouneded township ID could be defined as

```
TOWNID=((STRATUM*100+COUNTY)*1000+DISTRICT)*1000+TOWNSHIP
```

or

```
TOWNID=STRATUM*100000000+COUNTY*1000000+DISTRICT*1000+TOWNSHIP
```

The multiplier (e.g.100 or 10000) in the above example should be carefully chosen according to the range of each identication.

When users want to estimate the poverty and inequality at levels lower than cluster level, they should use both CLUSTER and CKEYVAR clauses such as

```
%let Cluster=DISTID;
%let cKeyVar=TOWNID;
```

Then the cluster ID has the form of **SCCDDD** and the ID on each record of census data has the form of **SCCDDDTTT.** During the simulation, cluster ID will be used to determine when a new cluster level redraw should happen. this ID will be used to produce aggregations at different levels by shifting the ID to the right.

Stage 2. Bootstrap (Simulation)

Methodology

The fully specified simulation model is defined as follows:

(9) $$\ln \tilde{y}_{ch} = \mathbf{x}_{ch}' \tilde{\beta} + \tilde{\eta}_c + \tilde{\varepsilon}_{ch}$$

where $\tilde{\beta} \sim N(\hat{\beta}, \hat{\Sigma}_\beta)$

$\tilde{\eta}_c$ is a random variable (could be normally distributed or T-distributed) with a variance defined in (5)

$\tilde{\varepsilon}_{ch}$ is a random variable (either normally distributed or T-distributed) with a variance defined in (7) , B= $\exp(\tilde{Z}_{ch}^T \tilde{\alpha})$ and $\tilde{\alpha} \sim N(\hat{\alpha}, \hat{\Sigma}_\alpha)$

Trimming could be applied to the random variable $\tilde{\eta}_c$ and $\tilde{\varepsilon}_{ch}$ as well as to random vector $\tilde{\beta}$

and $\tilde{\alpha}$. In the case of a normal distributed random variable, a range (-1.96, 1.96) will make 10% of random --$N(0,1)$--drawing to be redrawn. For random vector of size *m*, the vector will be redrawn if the mode of the vector (a χ^2 distributed random variable) is outside the specified range.

After estimating $\ln \tilde{y}_{ch}$, several poverty and inequality measures will be computed. They include Generalized Entropy class :

$$GE(\lambda) = \frac{1}{\lambda^2 - \lambda}\left[\frac{1}{W}\sum w_i\left(\frac{y_i}{\bar{y}}\right)^{\lambda} - 1\right] \qquad (\lambda \neq 0\ ,\ \lambda \neq 1)$$

$$GE(0) = \frac{1}{W}\sum w_i \log\frac{y_i}{\bar{y}} \quad \text{and} \quad GE(1) = \frac{1}{W}\sum w_i \frac{y_i}{\bar{y}} \log\frac{y_i}{\bar{y}}$$

Atkinson class of measures : $A(c) = 1 - \left[\frac{1}{W}\sum w_i\left[\frac{y_i}{\bar{y}}\right]^{1-c}\right]^{1/1-c}$ $(c \neq 0\ ,\ c \neq 1)$

and Gini index:

$$Gini = \frac{W+1}{W-1} - \frac{2}{W(W-1)\bar{y}}\sum w_i y_i[\rho_i + 0.5(w_i - 1)] \quad \text{where} \quad \rho_{i+1} = \rho_i + w_i$$

In the above definitions, w_i is the weight of household *i* and *W* is the total population.

Simulation Configuration

To run the simulation, the user needs to create a configuration file with extension PCF:

```
DataSource=Stra1.pda
nSim=100
PovLine=45476
MemorySize=128
SEED=1234567
CDist=T(5)
HDist=N
INDICES=FGT0 FGT1 FGT2 GE0 GE1 GE2 ATK2 GINI
```

```
minimpute=none
MAXimpute=none
HBound=auto
CBound=auto
ABound=NONE
BBound=0.99
Simulation=0 3 6
```

Explanation:

- DataSource: Optional. The filename of the input PDA file. If omitted, a PDA file with same name as PCF is assumed.
- NSim=*n*: Required. Number of simulation.
- PovLine=*n*: Required. Poverty line.
- MemorySize=*n*: Optional. Memory size available to the simulation. Default is 128M.
- Seed=*n*: Optional. The user specifies seed for random number generator on *Beta* vector. When omitted or set to 0, internally produced random seed will be used. This seed is derived by the system clock in 1/1000 second resolution. Thus no two simulations will be equal if seed is set to 0;
- CDist=v: Required if cluster effect is modeled. Type of distribution for cluster effect. Value selected from T(*n*)--T with DF of *n*, N--normal, NP--non parametric, HNP--hierarchical non parametric.
- HDist=v: Required. Type of distribution for cluster effect. Similar to CDist.
- MinImpute=*v* : Optional. Lower boundary for trimming simulated LHS variable. *v* could be a numeric value, AUTO or NONE. When AUTO is used, the lower bound of per capita expenditure in survey dataset will be used to eliminate that household in that simulation.
- MaxImpute=*v*: Optional. Upper boundary for trimming simulated LHS variable. *v* could be a numeric value, AUTO or NONE. When AUTO is used, the upper bound of per capita expenditure in survey dataset will be used to eliminate that household in that simulation. Default is NONE.
- cBound=*v*: Optional. Specifies the range for cluster effect (location effect) trimming. The value *v* could be *AUTO, NONE* or a specified value. When cBound=*AUTO*, the range will be *(-ScBound, ScBound)* where *ScBound* is the highest absolute value of

cluster effect in survey. When *v* is specified as a number, the trimming range will be *(-v, v).* When random number is outside of this range, repeat drawing will occur until a random value is inside the boundary. When cBound=*NONE*, no trimming will be used, in fact, it is done by setting up a boundary from negative infinity to positive infinity. The other boundary setting described below are implemented in this way too.

- hBound=*v*: Optional. Specifies the range for household effect trimming. The value *v* could be *AUTO*, *NONE* or a specified value. When hBound=*AUTO*, the range will be *(-hBoundInSurvey*, *hBoundInSurvey)* where *hBoundInSurvey* is the highest absolute value of household effect in survey. When hBound=*NONE*, no trimming will be used. When *v* is specified as a number, the trimming range will be *(-v, v).*
- bBound=*v*: Optional. ($0<v \leq 1$) The accepting probability for drawing random vector *Bet*l. Value *v* will be internally converted to range according to the distribution of the random variable.
- aBound=*v*: Optional. ($0<v \leq 1$) The accepting probability for drawing random vector *Alpha* model. Value *v* will be internally converted to range according to the distribution of the random variable.
- INDICES: Required. Indices of poverty and inequality measurements. User can choose from FGT0 FGT1 FGT2 GE0 GE1 GE2 ATK2 GINI in any order. General Entropy measurements with fraction parameter can be specified like GE1.75. User can also require the distribution to be outputted as percentile. Three different styles can be used here: DIST, DIST:20 or DIST(1,5,10,25,50,75,90,95,99). The first specification produce the distribution as decile value, the second format allows for 20 groups of percentile equally spaced. The third format will produce an uneven group at 1%, 5%, 10% ... 95% 99% levels.
- Simulation=*n1 [n2 ..]* : Required. This is the way to run simulations on different aggregate levels. When *n*=0, the record identifier in PDA file will be used. When this ID changes, a new aggregation will be outputed. When *n*>0, the ID in census dataset will be shifted *n* digits to the right to produce a shorter ID that represents an aggregation on higher level, new aggregation will be outputed when this value changes. For example, if ID is the form SCCDDD, then SIMULATION=3 will produce a estimates at the county level (SCC).

When multi-level simulation is requested such as *Simulation=0 3 5*, estimates of district level (SCCDDD), county level (SCC) and stratum level (S) will be produced in one simulation.

Please note that characters used in simulation configuration file are not case sensitive. Users may also use "*" in front of each line to disable that line (set it as a comment line).

Other Options

All items listed here are optional.

- END: To terminate the execution. Any statement after END will be discarded.
- yDump=*v*: In order to"dump" the estimated Y. This is useful when further measurement is wanted. The output file is binary. (See below for more information).
- Beta(*VarName*)*=1.01: To manually alter the value of Beta model. *VarName* identify the variable name whose value will be set to 1% higher. Similar notation could be /=, +=, and -=. Their impact can be shown with concrete example: let X=1.0

  X*=1.01 <===> assign X*1.01 to X <===> X become 1.01

  X/=1.01 <===> assign X/1.01 to X <===> X become 0.99

  X+=0.02 <===> assign X+0.02 to X <===> X become 1.02

  X-=0.02 <===> assign X-1.02 to X <===> X become 0.98

  An number can also be assigned to the Beta with Beta(*VarName*)=1.23.
- Alpha(VAR11)=0: Alter the value of Alpha model. Similar to Beta.
- seed=last: Set the random seed to be the same as last simulation
- yBound=0.999: Another way to trim Y. Specify the proportion to be kept.

Identify the Distribution of Random Component

An earlier version of this program included a module that identified the distribution of random components. Unfortunately, this module does not work well. In the current version, this function is excluded. User have to do analysis manually to identify the best fitted distribution. For SAS user, the data preparation will produce two SAS datasets with name ClusterRes and hhldRes.

ClusterRes is the random component correspondent to cluster effect $\hat{\eta}_c$, hhldRes is the random component correspondent to idiosyncratic component $\hat{\varepsilon}_{ch}$. For user of PovMapPacker, the two files produced have extension pResC and pResH, and both are in ASCII format:

Obs	Weight	Residual
1	3.2776626e-003	0.17330292
2	4.8687609e-003	6.4200165e-002
3	2.2275377e-003	-0.57137692
4	1.3126561e-003	0.39374108
5	3.5004163e-004	-4.3999818e-002
6	2.0684278e-003	0.275217
7	1.4850251e-004	-0.54797756
8	2.1320718e-003	0.14737068
9	..…	

Format of yDump file

User can collect all the estimated Y into a binary file. The option YDUMP

The yDump file is organized as

double for cluster ID, float for household size, float for y_{11}, float for y_{12}, ... float for $y_{1\ nSim}$,
double for cluster ID, float for household size, float for y_{21}, float for y_{22}, ... float for $y_{2\ nSim}$,
double for cluster ID, float for household size, float for y_{31}, float for y_{32}, ... float for $y_{3n\ nSim}$,
.....
double for cluster ID, float for household size, float for y_{n1}, float for y_{n2}, ... float for $y_{n\ nSim}$,

This file can be read with SAS with following code.

```
Data YDUMP;
  infile "c:\project1\stratum1\ydump.bin" recfm=n;
  input  ClusterID rb8.0   (hhsize  x1-x100) (float4.0) @@;
```

here X100 will be replaced by X300 if 300 simulation were run. 'recfm=n' along with '@@' read the binary data as a stream. User who wants to read the YDUMP file in Stata please contact me.

Run PovMap under Command Mode

PovMap.exe can be run in Windows' command mode. Users can keep PovMap.EXE anywhere on the PC or on a network drive. For simplicity, let us assume the program is placed on the network drive s:\PovMap\, and user has already change directory to c:\PovMap\Stratum1\ and prepared a simulation configuration file `STRATUM1.PCF`. To begin running PovMap, type the following:

```
c:\PovMap\Stratum1>s:\povMap\PovMap STRATUM1.PCF
```

For the complete list of PovMap's syntax, type option `–h` in command line

```
PovMap –h
```

The syntax will be displayed as

```
PovMap ConfigFileName [-vABVLT] [-h] [-s] [-r] [-oOutputFileName] [-0nnn]
  ConfigFileName     PovMap configuration file name.
  -v                 View input parameters of Alpha and Beta.
    A                  showing Alpha vector.
    B                  showing Beta vector.
     V                 showing varian-covarian matrix.
     L                 showing linkage vector.
    T                  showing raw data (only for memory mode 1).
  -h                 Show this help screen.
  -r                 Reset output file. Existing file will be erased.
  -nN                Override the number of simulation defined in config file.
  -oOutputFileName   Alternative output file name. Default is Result.dat.
  -0                 Set switches to random drawing.
    0                suppress random drawing on Beta.
     0               suppress random drawing on locational effect.
      0              suppress random drawing of household effect.
```

The command line options need further explanation:

- -r Reset the output file. By default the output of bootstrapping will always append to the end of existing result file *.POU. To clear the content in the result file, use -r option.
- -n Override the number of simulation specified in *.PCF file.
- -0 Testing switch. Is followed by three digits of 1 or 0. When 0 is shown in that position, the correspondent random variable will be shut down. This is useful when testing the result against an existing measurement.
- -v Requests a display of parameters including estimation of *Beta* and *Alpha*, its variant-covariant matrix.

Run PovMap under Windows

PovMap can also easily run under Windows. User can associate the file type *.PCF with the executable PovMap.exe and then double lick over PCF file to run PovMap.exe. To do that, open windows explorer, right-click over the PCF file, then click the **`Open With…`** Under the Open With menu, check the **Always use this program to open these files,** then click Other to location the PovMap.exe file. Click OK once the PovMap.exe is identified. This will permanently associate PCF file with PovMap.exe.

Result of Simulation

Result of bootstrapping is stored in file of type POU. The result file is a tab delimited text file. The first nine columns are always provided no matter whether you choose the measurement.

Type	Unit	nHHLDs	nIndividuals	nSim	Min_Y	Max_Y	avg_MEAN	se_MEAN
Point Estimate	3227	145	760	100	5698.7599	1628399.4	94299.803	12172.398
Point Estimate	3228	120	605	100	7082.1178	1253250.2	85392.196	10001.359
Point Estimate	3229	134	782	100	8621.7361	1034800.7	74580.428	10317.259
Point Estimate	3230	143	802	100	8850.0348	2258822.8	85437.389	9125.4147
Point Estimate	3231	112	634	100	2629.6289	611199.37	43242.205	4512.5221
Point Estimate	3232	59	382	100	2134.0325	302344.98	31155.461	2907.7902
Point Estimate	3233	57	288	100	5045.925	711865.91	57055.221	7955.2547

The remaining columns are optional depending on whether it is specified in INDICES= statement of PCF file. Each index selected will have two columns, the first one (denoted with prefix “avg_”) is the average and the second (denoted with prefix “se_”) is the standard error.

The best way to open POU file is to associate the POU file with Excel.

Remark

This software package is provided free of charge. PovMap is intended for use by the World Bank and its clients and is not intended to be sold or used for commercial purposes. Under no circumstances shall The World Bank be liable for any loss, damage, liability or expense incurred or suffered which is claimed to result from use of PovMap.