sae: The Manual

Paul Corral Rodas and Sandra Segovia Juarez*

Key words: Small area estimation, ELL, Poverty mapping, Poverty map, Big Data, Geospatial
JEL classification: C55, C87, C15

*The author acknowledge financial support from the World Bank. This is a product of the Poverty and Equity Policy Lab of the Poverty and Equity Global Practice.
Any error or omission is the authors' responsiblility alone.

1 Introduction

Household surveys that are designed for national or sub-national (i.e. regions or states) level parameter estimates often lack a sufficiently large sample size to generate accurate direct estimates for smaller domains or sub-populations.[1] Any area for which the sample is not sufficiently large to produce an estimate of adequate precision is referred to as a small area. Consequently small area methods attempt to address low representativeness of surveys within areas, or the lack of data for specific areas/sub-populations. This is accomplished by incorporating information from supplementary sources. Poverty estimates are often produced by statistical agencies, and commonly are the product of a household survey. Household surveys usually are the main source of welfare (expenditure or income) used to produce poverty estimates, yet most are only reliable up to a certain geographical level. Therefore, a commonly adopted solution has been to borrow strength from a national census, administrative records, and/or geospatial information which allows for representativeness for small areas (or any domain for which the survey lacks the necessary precission). Nevertheless, these outside data sources often lack detailed expenditure or income information required for producing estimates of poverty. Small area methods attempt to exploit each data's attributes to obtain estimators that can be used at dis-aggregated levels.

Under unit-level small area estimation the desired estimate is obtained by first fitting a model using survey data whose parameters are then linked to outside information such as a population census, administrative records, and/or geospatial information obtained through remote sensing with the goal of obtaining estimates for the entire population. As Rao and Molina (2015) state, the availability of these auxiliary data as well as a valid model are essential for obtaining successful small area estimates. Since poverty measures are nonlinear parameters of the welfare distribution, small area estimation methods coupled with Monte Carlo simulations are a useful statistical technique for monitoring poverty along with its spatial distribution and evolution.[2]

Poverty at lower geographical levels can be used to identify areas that are in need of attention, or that may be lagging behind the rest of the country. For example, the government of Ecuador after an earthquake that occurred on April 16, 2016 relied on small area estimates of poverty to decide where help was needed most. The small area estimates of poverty for Ecuador, which were released not long before the earthquake, proved to be an invaluable resource for the rebuilding effort in the country.

One of the most common small area methods used for poverty estimates is the one proposed by Elbers, Lanjouw, and Lanjouw (2003, henceforth ELL).[3] This methodology has been widely adopted by the World Bank and has been applied in numerous poverty mapsconducted by the institution.[4] In its efforts to make the implementation of the ELL methodology as straight forward as possible, the World Bank created a software package that could be easily used by anyone. The software, PovMap (Zhao, 2006),[5] has proven to be an invaluable resource for the World Bank as well as for many statistical agencies, line ministries, and other international organizations seeking to create their own small area estimates of poverty. The software is freely available and has a graphical user interface which simplifies its use. Nevertheless, many advances in

[1] For example districts, municipalities, migrant populations, or populations with disabilities.

[2] Poverty is a nonlinear function of welfare, consequently unit-level small area estimation methods attempt to replicate the welfare distribution as closely as possible relying on model assumptions. For simple poverty measures such as headcount poverty, the expected value of the indicator may be calculated without having to resort to Monte Carlo simulations (Molina 2019)

[3] For a detailed discussion of early applications of small area poverty estimates by the World Bank, readers should refer to Bedi et al. (2007). For other applications beyond poverty, readers should refer to Rao and Molina (2015).

[4] Poverty map is the common name within the World Bank for the methodology where the obtained estimates are mapped for illustrative purposes.

[5] downloadable from: http://iresearch.worldbank.org/PovMap/PovMap2/setup.zip

the field have taken place since the release and the ELL methodology has been replaced by methods which under some scenarios are deemed to yield estimates of better quality.

Empirical Best (EB) methods for small area estimation of poverty by Molina and Rao (2010) has arguably taken the limelight away from ELL. In their paper Molina and Rao (2010) illustrate the considerable improvement in precision and accuracy that EB methods offer for small area estimation of poverty. An early attempt to add EB methods to PovMap detailed by Van der Weide (2014), was later shown to be less than an ideal implementation (see Corral et al. 2021). An early version of the Stata package presented here implemented the adaptation detailed in Van der Weide (2014), although it is now only available for replication purposes. Corral et al. (2021) present the advances in methodology offered by empirical best methods. Those advances are also included into the package presented here.

The purpose of this document is to provide users with a detailed manual on using the sae package for Stata. Through examples and a discussion of the available options and their functions, this guide offers a comprehensive overview of the package.

In the following section we discuss the different methods implemented in the `sae` Stata package in the order these were introduced in the literature. After presenting the method, the method's application in Stata is presented with the different options available under each method.

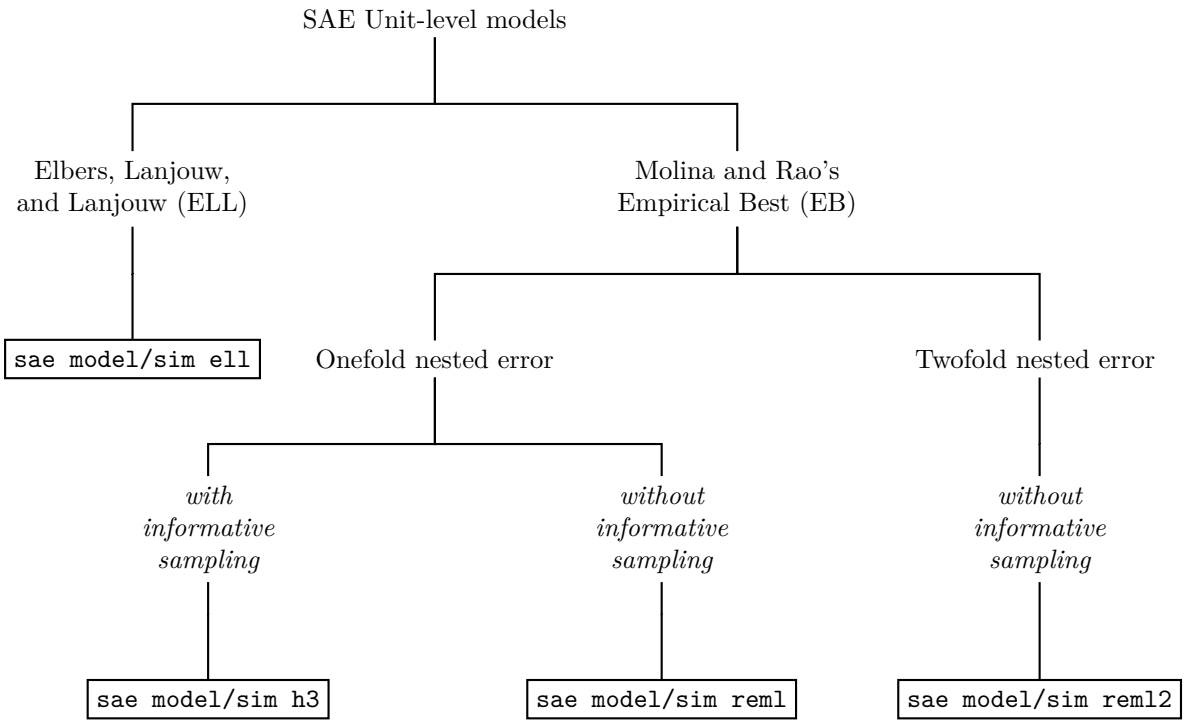

Figure 1: Approaches to Estimating SAE Unit-Level Models

Figure 1 showcases different approaches to estimating unit-level SAE models, emphasizing the transition from the ELL method to Empirical Best (EB) methods introduced by Molina and Rao (2010). EB models enhance the accuracy of poverty estimation and are categorized based on their error structure: onefold models, which are further divided into informative sampling with weights (estimated using Henderson's method - `sae`

h3) and non-informative sampling (estimated using restricted maximum likelihood -sae reml); and twofold models, estimated using restricted maximum likelihood for twofold nested error models - sae reml2.

2 ELL approach

The section presented here lifts directly from Corral et al. (2021) which presented the updates implemented to the Emprical Best (EB) methods originally implemented in the sae Stata package as detailed in Nguyen et al. (2018).

The Elbers, Lanjouw and Lanjouw (2002, 2003) methodology was the de facto small area estimation approach utilized by the World Bank until the updates presented in Corral et al. (2021). Before the updates, the ELL approach was widely applied across the globe to produce poverty maps conducted by the institution.[6] The popularity of the method can be attributed, to some degree, to the availability of the PovMap software (Zhao, 2006),[7] which was programmed in C and offers users a simple point and click interface. The PovMap software is also incredibly computationally efficient and fast, allowing users, even with limited computing power, to work with census data without facing memory limitations. In 2018, a Stata implementation of the PovMap software was released (Nguyen et al., 2018).

The ELL method assumes that the natural log of welfare y_{ch} for each household h within each location c in the population is linearly related to a $1 \times K$ vector of characteristics (or correlates) x_{ch} for that household, according to the nested error model:

$$\ln(y_{ch}) = x_{ch}\beta + \eta_c + e_{ch},\ h = 1, \ldots, N_c,\ c = 1, \ldots, C, \tag{1}$$

where η_c and e_{ch} are respectively location and household-specific idiosyncratic errors, assumed to be independent from each other, following (although not explicitly noted by ELL):

$$\eta_c \overset{iid}{\sim} N\left(0, \sigma_\eta^2\right),\ e_{ch} \overset{iid}{\sim} N\left(0, \sigma_e^2\right).$$

Here, C is the number of locations in which the population is divided and N_c is the number of households in location c, for $c = 1, \ldots, C$. Finally, β is the $K \times 1$ vector of coefficients.

A key feature of the ELL approach as implemented in the sae Stata package is that it considers heteroscedasticity and fits a preliminary model for that, called alpha model. In most applications, the alpha model has usually a small adjusted R^2 (which may not reach 0.05) yet tends to play a considerable role in the obtained point estimates, especially for parameters beyond poverty rate such as Gini index, Theil index, Poverty Gap, etc. Readers interested in understanding how the alpha model is implemented can refer to Nguyen et al. (2018), section 3.1.1.

The nested error model in (1) was originally fit by Generalized Least Squares (GLS) accounting for heteroscedasticity. Afterwards, Van der Weide (2014) updated the original GLS method to properly account for survey weights.[8] Currently, the implemented method fits the model by Feasible Generalized Least Squares (FGLS). In this procedure, a simple linear regression obtained by considering as model errors $u_{ch} = \eta_h + e_{ch}$

[6] Poverty mapping is the common name within the World Bank for SAE methodology, where the obtained estimates are mapped for illustrative purposes

[7] downloadable from: http://iresearch.worldbank.org/PovMap/PovMap2/setup.zip

[8] Haslett et al. (2010) presents the problems in the original GLS implementation of ELL.

is fit using ordinary least squares (OLS), and afterwards the appropriate covariance matrix of regression parameter estimators is estimated.[9]

The variances of the model parameter estimators are required in the considered ELL bootstrap approach which borrows from the multiple imputation literature. Originally, to estimate these variances, ELL (2002) proposed to use the delta method. The actual implementation of the delta method is through a computationally intense simulated numerical gradient. An estimate of the gradient vector is obtained by making perturbations of the parameters, and this is used to estimate the variances of the estimators via the delta method. ELL (2002) mentions also the possibility of drawing from the sampling distribution (parametric) as a way to incorporate the estimation error into the total prediction error. This latter method has become the de facto approach used under the ELL method, and is what is implemented in the `sae` Stata package.[10]

The first implementation of the World Bank poverty mapping software was actually done in SAS by Demombynes (2002). Poverty indicators and their corresponding standard errors were obtained by a bootstrap procedure relying on simulation. This simulation approach is very reminiscent of multiple imputation methods, where every single relevant parameter necessary for simulating vectors of welfare is drawn from its corresponding estimated asymptotic distribution (or an approximation to it). This approach is the one used. Specifically, the steps of the implemented bootstrap procedure designed to obtain the traditional ELL point estimates and their estimated noise are:

1. Fit the nested error model (1) to the survey data. This yields the vector of initial parameter estimates

$$\hat{\theta}_0 = (\hat{\beta}_0, \hat{\sigma}^2_{\eta 0}, \hat{\sigma}^2_{e0}),$$

   where the 0 subscript is used hereafter to indicate that the estimates come from the original household survey. In the implemented version, the model is fit via FGLS as specified in Nguyen et al. (2018) Section 3.3.

2. Draw new model parameters as follows. First, regression coefficients are drawn from

$$\beta^* \sim MVN\left(\hat{\beta}_0, \widehat{\text{vcov}}(\hat{\beta}_0)\right),$$

   where $\widehat{\text{vcov}}(\hat{\beta}_0)$ is the variance covariance matrix for $\hat{\beta}_0$. The variance of the household-level errors is drawn, according to Gelman et al. (2004, pp. 364-365), from

$$\sigma^{2*}_e \sim \hat{\sigma}^2_{e0} \frac{(n-K)}{\chi^{2*}_{n-K}},$$

   where χ^{2*}_{n-K} denotes a random number from a chi-squared distribution with $n-K$ degrees of freedom. Here, n is the number of observations in the survey data used to fit the model and K is the number of correlates used in the model. Alternatively, when heteroskedasticity is assumed, the variance of the household-level errors is given by Nguyen et al. (2018). The variance of the cluster, or location effects

[9] For a detailed look into the ELL approach, interested readers should refer to the original ELL papers (ELL, 2003; 2002) and Section 3.3 of Nguyen et al. (2018), which presents the current GLS estimator from Van der Weide (2014)

[10] In a comparison of the simulation methods proposed by ELL (2002), Demombynes et al. (2008) shows that the delta method from ELL and the parametric drawing of the parameters provide similar results. In tests with pseudo surveys, the delta method seems to provide wider standard errors than the parametric approach (Demombynes et al. 2008), suggesting that perhaps the parametric estimates are too optimistic when compared to the delta method

σ_η^{2*} is drawn, according to Demombynes (2008) and Demombynes et al. (2002), from

$$\sigma_\eta^{2*} \sim Gamma\left(\hat{\sigma}_{\eta 0}^2, \widehat{\text{var}}(\hat{\sigma}_{\eta 0}^2)\right).$$

3. Using the simulated model parameters in step 2, calculate the welfare for every household in the census y_{ch}^* from the model as

$$\ln(y_{ch}^*) = x_{ch}\beta^* + \eta_c^* + e_{ch}^*,$$

where the household-specific errors are generated as

$$e_{ch}^* \overset{iid}{\sim} N\left(0, \sigma_e^{2*}\right),$$

and the location effects are generated as

$$\eta_c^* \overset{iid}{\sim} N\left(0, \sigma_\eta^{2*}\right).$$

Then construct the vector containing the N_c simulated welfares for the households in location c, denoted $y_c^* = (y_{c1}^*, \ldots, y_{cN_c}^*)^T$, where N_c is the number of census households in location c.

4. With the vectors y_c^*, $c = 1, \ldots, C$, of simulated census welfares, indicators can be produced for all the locations. Let $\tau_c^* = f(y_c^*)$ be the indicator of interest calculated based on the simulated vector for location c; for example, for the Foster, Greer and Thorbecke (1984,FGT) class of decomposable poverty measures, the function $f(y_c^*)$ is defined as

$$f_\alpha(y_c^*) = \sum_{h=1}^{N_c} \frac{p_{ch}}{\sum_\ell p_{c\ell}} I\left(y_{ch}^* < z\right) \left(1 - \frac{y_{ch}^*}{z}\right)^\alpha, \; \alpha \geq 0,$$

where z is the poverty line, p_{ch} is the size of household h from location c in the census and $I(y_{ch}^* < z) = 1$ if $y_{ch}^* < z$ and is equal to 0 otherwise.

5. Repeat steps 2 to 4 M times. Although traditionally $M = 100$, a larger number of replicates M is recommended. Let $\tau_c^{*(m)}$ be the indicator of interest obtained in m_{th} replicate of the bootstrap. The ELL estimator is then given by the average across the M bootstrap replicates,

$$\hat{\tau}_c^{ELL} = \frac{1}{M} \sum_{m=1}^{M} \tau_c^{*(m)}$$

and the ELL estimated variance of the ELL estimator is given by

$$\text{var}_{ELL}(\hat{\tau}_c^{ELL}) = \frac{1}{M-1} \sum_{m=1}^{M} \left(\tau_c^{*(m)} - \hat{\tau}_c^{ELL}\right)^2.$$

As opposed to the EB method of Molina and Rao (2010), which uses a Monte Carlo simulation procedure and a separate bootstrap procedure for MSE (see Section 3), in the above ELL procedure, a single computational algorithm tries to capture the noise of the initial model parameter estimates in the ELL standard error by varying the model parameters across simulations, which is aligned to Rubin's rules (Rubin 2004, 1996). This means that, in each replicate of the bootstrap, different values of the model parameters β^*, σ_η^{2*} and σ_e^{2*}

are used to generate the welfare data. However, this step might entail an increased noise of the final ELL estimators (see Corral et al. 2021).

2.1 ELL in the `sae` command

The `sae` command is split into a model and a simulation stage. The model stage can be used to initially fit the model following its underlying assumptions. The simulation stage will also fit the model but will also run the simulations and produce as a dta dataset the final small area estimates.

2.1.1 ELL Model-fitting under the `sae` command

The ELL Model-fitting follows the GLS approach presented by Van der Weide 2014 and further detailed in Nguyen et al. (2018).

`sae model ell` *`depvar indepvars`* `[if] [in] [aw],` *`options`*

Required

- `area()`: The `area` option is necessary and specifies at which level the clustering is done, it indicates at which level the η_c, the random location effect, is obtained at. The only constraint is that the variable must be numeric, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006).
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[11]

Optional

- `zvar()`: The `zvar` option is necessary for specifying the alpha model, the user must place the independent variables under the option
- `yhat()`: The `yhat` option is also a part of the alpha model (heteroskedasticity). Variables listed here will be interacted with the predicted $\hat{y} = X\beta$ from the OLS model.
- `yhat2()`: The `yhat2` option is also a part of the alpha model. Variables listed here will be interacted with the predicted $\hat{y}^2 = (X\beta)^2$ from the OLS model.
- `alfatest()`: The `alfatest` option is useful for alpha model selection. It requests the command to output the the dependent variable of the alpha model, which depends on the current model fit, for users to select a model for heteroskedasticity.

Example 1 Model-fitting under ELL is done by calling the `sae` command and the `model ell` subcommands. The user must have loaded their data beforehand. The simulated data used throught these examples are created as illustrated in the appendix 4.1 at the end of this document.

[11] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

As can be seen in the example below, the command requires the user to specify the area level for the random effects. The command also provides information that allows evaluating the model fit, including key information about the random effects. Aside from removing perfectly collinear covariates, the command will also exclude from the model areas where there are less than 3 observations.

```
. *================================================================================
. // Model fitting under ELL
. *================================================================================
. //Determine personal folder - this will only work if you´ve created the data
. //beforehand
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. //import data
. use "`p´/sae/survey.dta", clear

.         //fit model
.         sae model ell Y x1 x2, area(HID)
WARNING: 0 observations removed due to less than 3 observations in the cluster.

OLS model:
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

GLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378435	.0089481	4.23	0.000	.0203056	.0553814
x2	-.0362157	.0106211	-3.41	0.001	-.0570327	-.0153988
_cons	2.989547	.0107443	278.24	0.000	2.968489	3.010606

Comparison between OLS and GLS models:

Variable	bOLS	bGLS
x1	.03085459	.0378435
x2	-.03264075	-.03621573
_cons	2.9932721	2.9895472

```
Model settings
------------------------------------------------------------------
Error decomposition                             ELL

Beta model diagnostics
------------------------------------------------------------------
Number of observations                  =    4000
Adjusted R-squared                      =    .00390058
R-squared                               =    .00439875
Root MSE                                =    .29774216
F-stat                                  =    8.8297455

Model parameters
```

```
Sigma ETA sq.                                =    .02549255
Ratio of sigma eta sq over MSE               =    .28756276
Variance of epsilon                          =    .06315784
Sampling variance of Sigma eta sq.           =    5.152e-06
```

<End of first stage>

Example 2 In addition, a key contribution of Elbers et al. (2002) was the introduction of the alpha model, which allows users to model for heteroskedasticity of the household specific residuals. The dependent variable used for the alpha model can be created by specifying the `alfatest()` option combined with `zvar()`. Doing this creates 3 variables in the data that can be used to simplify selection of the alpha model. The first two variables will have as prefix the string specified in the `alfatest()` option: 1) `het_alpha`, the dependent variable for the alpha model, and 2) `het_xb`, the predicted linear fit of the main model that may be used as a covariate in the alpha model. The third and final variable is `_est_bGLS`, which indicates the observations that were used for the GLS model. When the alpha model is run, the command gives additional information related specifically to the alpha model.

```
. //Determine personal folder - this will only work if you´ve created the data
. //beforehand
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. //import data
. use "`p´/sae/survey.dta", clear

.
.         //Create dependent variable for alpha model
.         // This is useful for alpha model selection
.         // Just add any covariate to the zvar() option and specify alfatest()
.         sae model ell Y x1 x2, area(HID) zvar(x1) alfatest(het)
WARNING: 0 observations removed due to less than 3 observations in the cluster.

OLS model:
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

Alpha model:

Residual	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0334407	.0720811	0.46	0.643	-.1078356	.1747169
_cons	-4.037482	.0572511	-70.52	0.000	-4.149692	-3.925272

GLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378119	.0090123	4.20	0.000	.0201482	.0554756

```
          x2 |  -.0363414   .0107331    -3.39   0.001    -.0573778   -.0153049
       _cons |   2.989591   .0107564   277.94   0.000     2.968508    3.010673
------------------------------------------------------------------------------

Comparison between OLS and GLS models:

------------------------------------
    Variable |      bOLS         bGLS
-------------+----------------------
          x1 |   .03085459    .03781188
          x2 |  -.03264075   -.03634138
       _cons |   2.9932721    2.9895906
------------------------------------

Model settings
------------------------------------------------------------------------------
Error decomposition                               ELL

Beta model diagnostics
------------------------------------------------------------------------------
Number of observations                      =    4000
Adjusted R-squared                          =    .00390058
R-squared                                   =    .00439875
Root MSE                                    =    .29774216
F-stat                                      =    8.8297455

Alpha model diagnostics
------------------------------------------------------------------------------
Number of observations                      =    4000
Adjusted R-squared                          =    -.00019615
R-squared                                   =    .00005396
Root MSE                                    =    2.199986
F-stat                                      =    .215732

Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                               =    .02549255
Ratio of sigma eta sq over MSE              =    .28756276
Variance of epsilon                         =    .06315784
Sampling variance of Sigma eta sq.          =    5.152e-06
------------------------------------------------------------------------------

                  <End of first stage>

.         //new variables added to the data:
. // 1: het_alpha, the alpha model´s dependent variable
. // 2: het_xb, the predicted linear fit of the OLS model
. // 3: _est_bGLS, dummy variable indicating which observations are use
```

Example 3 Finally, modeling for heteroskedasticity is simple under the `sae model ell` command as users can easily add the `yhat()` and `yhat2()` options where variables specified under the option are interacted with the first stage OLS's linear fit. In the case of the `yhat2()` option, the variable indicated will be interacted with the square of the OLS linear fit. If interactions of the variable are not desired, then the user can specify these variables under the `zvar()` option.

```
. //Determine personal folder - this will only work if you´ve created the data
. //beforehand
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)
```

```
. //import data
. use "`p´/sae/survey.dta", clear

.
.         //fit model with alpha model including interactions with linear fit
.         sae model ell Y x1 x2, area(HID) yhat(x1)
WARNING: 0 observations removed due to less than 3 observations in the cluster.
```

OLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

Alpha model:

Residual	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1_yhat	.0111583	.02388	0.47	0.640	-.0356456	.0579622
_cons	-4.037631	.0572503	-70.53	0.000	-4.14984	-3.925423

GLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378119	.009012	4.20	0.000	.0201486	.0554752
x2	-.0363416	.0107324	-3.39	0.001	-.0573767	-.0153065
_cons	2.989591	.0107561	277.94	0.000	2.968509	3.010672

Comparison between OLS and GLS models:

Variable	bOLS	bGLS
x1	.03085459	.03781192
x2	-.03264075	-.03634162
_cons	2.9932721	2.9895908

Model settings

```
Error decomposition                             ELL
```

Beta model diagnostics

```
Number of observations                  =    4000
Adjusted R-squared                      =    .00390058
R-squared                               =    .00439875
Root MSE                                =    .29774216
F-stat                                  =    8.8297455
```

Alpha model diagnostics

```
Number of observations                  =    4000
Adjusted R-squared                      =    -.00019538
R-squared                               =    .00005473
Root MSE                                =    2.1999851
```

```
F-stat                                    =    .21881613

Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                             =    .02549255
Ratio of sigma eta sq over MSE            =    .28756276
Variance of epsilon                       =    .06315784
Sampling variance of Sigma eta sq.        =    5.152e-06
------------------------------------------------------------------------------

                  <End of first stage>

.
```

2.1.2 ELL Small Area Estimates under the sae command

The simulation stage can be used to obtain SAE point estimates as well as estimates of the noise. The main command for simulations is:

`sae sim ell` *depvar indepvars* `[if] [in] [aw],` *options*

Required

- `area()`: The `area` option is necessary and specifies at which level the clustering is done, it indicates at which level the η_c is obtained at. The only constraint is that the variable must be numeric, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006).
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[12]
- `uniqid()`: option specifies the numeric variable that indicates the unique identifiers in the census and survey dataset. This is necessary to ensure replicability of the analysis, and the name should match the one of the unique identifier from the source dataset.
- `rep()`: Indicates the number of Montecarlo Simulations to be executed and must be an integer.
- `matin()`: The matin option indicates the path and filename of the Mata format target dataset. The dataset must be created with the `sae data import` command.
- `aggids()`: The aggids option indicates the different aggregation levels for which the indicators are to be obtained, values placed here tell the command how many digits to the left to move to get the indicators at that level. Using the hierarchical id specified in the area option, AAMMEEE, if the user specifies 0, 3, 5, and 7, it would lead to aggregates at each of the levels E, M, A and the national level. Note that it is NOT advised to report results at a higher level than area(), it is likely that the MSE is underestimated (see Corral et al. 2021). Users should specify 0 if they are not using a hierarchical id.
- `pwcensus()`: Indicates the variable which corresponds to the expansion factors to be used in the target (census) dataset, it must always be specified. The user must have added the variable to the imported data (`sae data import`) i.e. the data.
- `indicators()`: The `indicators` option is used to request the indicators to be estimated from the simulated vectors of welfare. The list of possible indicators is:

[12] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

  - The set of Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) FGT_0, FGT_1, and FGT_2 ; also known as poverty head count, poverty gap, and poverty severity respectively.
  - The set of inequality indexes: Gini, and Generalized Entropy Index with $\alpha = 0, 1, 2$

- `eta()`: available options are normal and non-normal, indicates how eta is simulated.
- `epsilon()`: available options are normal and non-normal, indicates how epsilon is simulated.

Optional

- `zvar()`: The `zvar` option is necessary for specifying the alpha model, the user must place the independent variables under the option
- `yhat()`: The `yhat` option is also a part of the alpha model (heteroskedasticity). Variables listed here will be interacted with the predicted $\hat{y} = X\beta$ from the OLS model.
- `yhat2()`: The `yhat2` option is also a part of the alpha model. Variables listed here will be interacted with the predicted $\hat{y}^2 = (X\beta)^2$ from the OLS model.
- `ydump()`: The user must provide path and filename for a Mata format dataset to be created with the simulated dependent variables at the `uniqid()` level.
- `addvars()`: The addvars option allows users to add variables to the dataset created from the simulations. These variables must have been included into the target dataset created with the sae data import command.
- `lny`: The option indicates that the dependent variable in the welfare model is in log form. Specifying this indicates that the simulated vectors need to be transformed before pp the poverty line specified in `pline` or `plinevar`.
- `seed()`: The option ensures replicability. Users should be aware that Stata's default pseudo-random number generator in Stata 14 is different than that of previous versions. If not specified, the default seed of 123456789 is used.
- `plinevar()`: Option allows users to indicate a variable in the target data set which is to be used as the threshold for the Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) to be estimated from the simulated vectors. Te user must have added the variable in the `sae data import` command when preparing the target dataset. Only one variable may be specified.
- `plines()`: The option allows users to explicitly indicate the threshold to be used, this option is preferred when the threshold is constant across all observations. Additionally, it is possible to specify multiple lines, separated by a space.
- `results()`: The `results` option specifies the path and filename for users to save as a txt file the results the analysis specified in the `indicators` option.
- `allmata`: The `allmata` option skips use of the C plugin and does all poverty calculations in Mata.

Example 1 The ELL implementation in the sae Stata package aims to replicate the World Bank's PovMap software Zhao (2006). Hence, many of the options available for this implementation and the others are inspired on PovMap.

The simulation stage for ELL based small area estimates require the user first importing the census data into a mata data file. This is done because it is more simple to manage mata files and these can be processed more quickly.

The simulation process involves several more options that control how the simulation is undertaken as well as how results are obtained. The most important options control how the location and household effects are drawn. Only under the ELL do users have the option of drawing residuals from a normal distribution, `eps(normal)` and `eta(normal)`, as assumed under the model (1), or from the empirical distribution, `eps(nonnormal)` and `eta(nonnormal)`.

```
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. *================================================================================
. // Importing the census for small area estimates
. // Note that for the census we add the population expansion factors
. // to the varlist(). We´ve also added a variable containing the poverty line
. // "pvar".
. *================================================================================
.
. tempfile census_mata

.         sae data import, datain("`p´/sae/census.dta") varlist(x1 x2 hhsize pvar) ///
>         area(HID) uniqid(hhid) dataout(`census_mata´)
Saving data variables into mata matrix file (6)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
......
.
. *================================================================================
. //Obtain final small area estimates based on ELL approach
. *================================================================================
. //Bring in the survey data
. use "`p´/sae/survey.dta", clear

.         /*
>           The following model specifications are used:
>           1: Assume normal errors - eta(normal) and eps(normal)
>           2: Assumes that the Y variable is in log terms, we request to back //
>              transform before obtaining poverty and other indicators (lny)
>           3: The uniqid() option is necessary to ensure replicability
>           4: We indicate that the simulations are done at the household level
>              and indicate population expansion factors to obtain values for the
>              population, pwcensus().
>           5: We indicate that the poverty line to be used is in the variable
>              "pvar" within the census data, plinevar()
>           6: We´ve requested for the estimates to be obtained from 100 simulated
>              vectors, rep()
>           7: We´ve requested to estimate FGT0, headcount poverty, indicators(fgt0)
>           8: We´ve requested for results to be produced at the same level as the
>              area() variable, aggids(0).
>           9: We´ve indicated that the census data is in the mata file ,atin(`census_mata´),
>              which was created using the sae data import command.
```

```
>           10: We add survey weights to the model fit via [aw=hhweight]
>         */
.         sae sim ell Y x1 x2 [aw=hhweight], area(HID) eta(normal) eps(normal) lny ///
>         uniqid(hhid) pwcensus(hhsize) plinevar(pvar) rep(100) indicators(fgt0) ///
>         aggids(0) matin(`census_mata´)

Note: Dependent variable in logarithmic form
WARNING: 0 observations removed due to less than 3 observations in the cluster.

OLS model:
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

GLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378435	.0089481	4.23	0.000	.0203056	.0553814
x2	-.0362157	.0106211	-3.41	0.001	-.0570327	-.0153988
_cons	2.989547	.0107443	278.24	0.000	2.968489	3.010606

Comparison between OLS and GLS models:

Variable	bOLS	bGLS
x1	.03085459	.0378435
x2	-.03264075	-.03621573
_cons	2.9932721	2.9895472

```
Model settings
------------------------------------------------------------------------------
Error decomposition                             ELL
Beta drawing                                    Parametric
Eta drawing method                              normal
Epsilon drawing method                          normal
Empirical best method                           No

Beta model diagnostics
------------------------------------------------------------------------------
Number of observations                  =       4000
Adjusted R-squared                      =       .00390058
R-squared                               =       .00439875
Root MSE                                =       .29774216
F-stat                                  =       8.8297455

Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                           =       .02549255
Ratio of sigma eta sq over MSE          =       .28756276
Variance of epsilon                     =       .06315784
Sampling variance of Sigma eta sq.      =       5.152e-06
------------------------------------------------------------------------------
                        <End of first stage>
------------------------------------------------------------------------------
```

```
      Initializing the Second Stage, this may take a while...
Number of observations in target dataset:

        20000
Number of clusters in target dataset:

        400
      Parametric drawing of betas
      Number of simulations: 100
      Each dot (.) represents 1 simulation(s).
      ----+--- 1 ----+--- 2 ----+--- 3 ----+--- 4 ----+--- 5
      ..................................................    50
      ..................................................   100
      Finished running the Second Stage
      ------------------------------------------------------------------------------

      Opening plugin
      Output bas been loaded into Stata. Please see the results.

      .
      . cap:texdoc stlog close
```

Example 2 Users have two options for specifying the poverty thresholds in the simulation stage: `plinevar()` and `plines()`. The first, `plinevar()` allows users to indicate a variable in the census microdata with the threshold. The variable must have been imported when using the `sae data import` command and only allows for specifying one variable. The second, `plines()`, allows users to indicate multiple thresholds with the caveat that these apply equally to all households in the census microdata.

Because the model assumes that errors are normally distributed, and for poverty the replication of the model's assumptions is crucial to ensure unbiased estimates, transformation of the dependent variable is recommended. under the ELL implementation, the only transformation available is the natural logarithm. Specifying the `lny` option informs the command that the provided dependent variable for the model is in natural logs, and that the poverty line is not in natural logarithms. This tells the command to back transform the simulated welfare vectors. under the other methods implemented in the `sae` package additional transofrmation approaches are included.

An important feature of the original PovMap software by Zhao (2006) is its speed in calculating final indicators. To some extent the speed was due to the use of C. Stata's `sae` implementation of ELL uses a C plugin to calculate the indicators. Users have the option of not using the C plugin and instead use a Mata implementation by specifying the `allmata` option.

Finally, just like the original PovMap, users have the option to extract the simulated vectors of welfare for all observations in the census. This is achieved by the ydump() option. Users should keep in mind that this file will be quite large, depending on the census population. Hence, the file is saved as a Mata file and must be loaded on to Stata using the sae data export command as shown below. Due to the potential size of the data, users have the option to split the file into multiple dta files. When requesting to split the ydump into multiple files, users must provide a path and prefix for the new dta files. See the appendix 4.2.1 for more information on the `sae data export` and `sae data import` utilities provided with the `sae` Stata package.

```
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. *================================================================================
. // Importing the census for small area estimates
. // Note that for the census we add the population expansion factors
. // to the varlist(). We´ve also added a variable containing the poverty line
. // "pvar".
. *================================================================================
.
. tempfile census_mata

.         sae data import, datain("`p´/sae/census.dta") varlist(x1 x2 hhsize pvar) ///
>         area(HID) uniqid(hhid) dataout(`census_mata´)
Saving data variables into mata matrix file (6)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
......
.
. *================================================================================
. // Attempt to obtain estimates drawing errors from the empirical residuals and
. // obtain poverty rates unde multiple lines
. *================================================================================
. //Bring in the survey data
. use "`p´/sae/survey.dta", clear

.         //obtain 2 poverty lines, 25th and 75th percentile
.         sum E_Y, d

                            E_Y
-------------------------------------------------------------

      Percentiles      Smallest
 1%     10.05694       5.593755
 5%     12.33652       8.128839
10%     13.81096       8.526508       Obs               4,000
25%     16.57419       8.614201       Sum of wgt.       4,000

50%     20.21931                      Mean           21.14125
                        Largest       Std. dev.      6.499419
75%     24.65453       52.95024
90%     29.72983       59.15125       Variance       42.24245
95%     32.95321       61.11948       Skewness       1.043196
99%     41.65685       61.42175       Kurtosis        5.16971

.         local pline25 = r(p25)

.         local pline75 = r(p75)

.         //The following model specifications are modified from before:
.         //1: Assume nonnormal errors - eta(normal) and eps(normal)
.         //2: We indicate that poverty must be estimated at 2 lines, pline()
.         //3: We´d also like for estimates to ALSO be aggregated to the national
.         //   level using aggids() and the hierarchical ID.
.         tempfile thedump

.         sae sim ell Y x1 x2 [aw=hhweight], area(HID) eta(nonnormal)  ///
>         eps(nonnormal) lny uniqid(hhid) pwcensus(hhsize)  ///
>         pline(`pline25´ `pline75´) rep(100) indicators(fgt0) ///
>         aggids(0 5) matin(`census_mata´) allmata ydump(`thedump´)

Note: Dependent variable in logarithmic form
WARNING: 0 observations removed due to less than 3 observations in the cluster.

OLS model:
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

GLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378435	.0089481	4.23	0.000	.0203056	.0553814
x2	-.0362157	.0106211	-3.41	0.001	-.0570327	-.0153988
_cons	2.989547	.0107443	278.24	0.000	2.968489	3.010606

Comparison between OLS and GLS models:

Variable	bOLS	bGLS
x1	.03085459	.0378435
x2	-.03264075	-.03621573
_cons	2.9932721	2.9895472

Model settings

Error decomposition	ELL
Beta drawing	Parametric
Eta drawing method	nonnormal
Epsilon drawing method	nonnormal
Empirical best method	No

Beta model diagnostics

Number of observations	=	4000
Adjusted R-squared	=	.00390058
R-squared	=	.00439875
Root MSE	=	.29774216
F-stat	=	8.8297455

Model parameters

Sigma ETA sq.	=	.02549255
Ratio of sigma eta sq over MSE	=	.28756276
Variance of epsilon	=	.06315784
Sampling variance of Sigma eta sq.	=	5.152e-06

<End of first stage>

Initializing the Second Stage, this may take a while...

Number of observations in target dataset:

20000

Number of clusters in target dataset:

400

```
Parametric drawing of betas

Number of simulations: 100
Each dot (.) represents 1 simulation(s).
----+--- 1 ----+--- 2 ----+--- 3 ----+--- 4 ----+--- 5
..................................................    50
..................................................   100

Finished running the Second Stage
------------------------------------------------------------

Computing indicators for 100 simulation(s).
----+--- 1 ----+--- 2 ----+--- 3 ----+--- 4 ----+--- 5
..................................................    50
..................................................   100

.
. //import the ydump - this is the rep(100) vectors of imputed welfare on the
. // census.
. sae data export, matasource(`thedump´)
Your output will not be saved, please use the datasave() option

.
```

3 Empirical Best Approaches

This section borrows from Corral et al. (2021) which is a background paper for the updated `sae` command.

Molina and Rao (2010) also consider a nested error model (Eq.1), much like that of ELL. The main difference with the empirical best (EB) approaches presented in this section with ELL is that EB methods condition on the survey sample data making more efficient use of the survey data. The survey data is the only available information on welfare available and EB methods attempt to make the most out of what is available.

The EB method presented by Molina and Rao (2010) requires areas and households across survey and census to be matched. Thus, if P_c denotes the area's population of size N_c, then the survey sample, s_c, is a sample of size $n_c \leq N_c$ drawn from P_c. The complement to the sampled population is referred to as r_c, which is just $P_c - s_c$. Thus, the census welfare vector for any area c is defined as $y_c = (y_{c,s}^T, y_{c,r}^T)^T$, which contains the welfare for sampled and non-sampled observations in area c, denoted $y_{c,s}$ and $y_{c,r}$ respectively. Typically, the non-sampled population (of size $N_c - n_c$) is much larger than the sampled population (of size n_c). It may also happen that $n_c = 0$ for some areas, meaning that the area is not sampled in the survey. The EB predictor of τ_c for an area c that is sampled ($n_c > 0$) is defined as the conditional expectation $E(\tau_c|y_{c,s};\hat{\theta})$. For an out-of-sample area ($n_c = 0$), the EB predictor of τ_c is $E(\tau_c;\hat{\theta})$, which is similar to ELL estimator. Here, $\hat{\theta}$ is a consistent estimator of θ as the overall sample size n tends to infinity. In contrast with this procedure, the traditional ELL approach does not require to link the census and survey observations.

3.1 Molina and Rao's EB

When the expectation $E(\tau_c|y_{c,s};\hat{\theta})$ has no explicit form, the empirical best (EB) predictor of τ_c can be approximated using a Monte Carlo (MC) simulation method. The MC approximation to the EB predictor $\hat{\tau}_c^{EB} = E(\tau_c|y_{c,s};\hat{\theta})$, as described in Rao and Molina (2015), is obtained as follows:

1. Using the survey data, fit model (1) via any method providing consistent estimators. This yields the vector of parameter estimates:

$$\hat{\theta}_0 = \left(\hat{\beta}_0, \hat{\sigma}^2_{\eta 0}, \hat{\sigma}^2_{e0}\right).$$

   Usual fitting methods under this appr hod, which does not require to specify a distribution. In the implemented sae R package, REML is used.

2. Use the parameter estimates obtained in step 1 as true values to simulate a vector of welfare in the census. First, the welfare from each out-of-sample household is generated as follows,

$$\ln(y^*_{ch,r}) = x_{ch,r}\hat{\beta}_0 + \eta^*_c + e^*_{ch,r},$$

   where we add the subscript r to indicate that the household is not sampled. Here, for an area c that is included in the sample, the area effect η^*_c is generated as

$$\eta^*_c \sim N\left(\hat{\eta}_{c0}, \hat{\sigma}^2_{\eta 0}(1-\hat{\gamma}_c)\right),$$

   where $\hat{\eta}_{c0}$ and $\hat{\gamma}_c$ are respectively given by

$$\hat{\eta}_{c0} = \hat{\gamma}_c\left(\bar{y}_{c,s} - \bar{x}_{c,s}\hat{\beta}_0\right), \quad \hat{\gamma}_c = \frac{\hat{\sigma}^2_{\eta 0}}{\hat{\sigma}^2_{\eta 0} + \hat{\sigma}^2_{e0}/n_c}.$$

   Here, $\bar{y}_{c,s}$ and $\bar{x}_{c,s}$ are respectively the sample means of the welfare variable and the correlates in area c. If area c is not sampled, the area effect is generated as $\eta^*_c \sim N\left(0, \hat{\sigma}^2_{\eta 0}\right)$. Finally, the household error $e^*_{ch,r}$ is generated as:

$$e^*_{ch,r} \sim N\left(0, \hat{\sigma}^2_{e0}\right).$$

   For an area c that is sampled in the survey, the generated non-sample vector $y^*_{c,r}$ is then augmented by the survey data $y_{c,s}$. Therefore, the final vector for the whole census in area c is $y^*_c = (y^T_{c,s}, y^{*T}_{c,r})^T$, which is made up of the generated out-of-sample welfares and the survey ones.

3. The previous step yields a census vector of welfare y^*_c, simulated using the fitted model parameters. This census vector is then used to calculate the indicator for each area $c = 1, \ldots, C$, as

$$\tau^*_c = f\left(y^*_c\right),$$

   where $f(\cdot)$ can be any indicator function such as the FGT poverty indicators.

4. Repeat steps 1 to 3 a large number of times M. If $y^{*(m)}_c$ denotes the m_{th} replicate of the census vector $y^*_c = (y^T_{c,s}, y^{*T}_{c,r})^T$, then $\tau^{*(m)}_c = f(y^{*(m)}_c)$ is the corresponding indicator, $m = 1, \ldots, M$. The final MC approximation to the EB estimator of τ_c is just the average across de M simulations of these indicators

$$\hat{\tau}^{EB}_c = \frac{1}{M}\sum_{m=1}^{M} \tau^{*(m)}_c.$$

The steps for the parametric bootstrap estimation of the MSE are as follows (MR, 2015) and is based on González-Manteiga et al. (2008):

1. Using the survey data, fit model (1) using a method providing consistent estimators. This yields the set of parameter estimates from the observed sample:

$$\hat{\theta}_0 = \left(\hat{\beta}_0, \hat{\sigma}^2_{\eta 0}, \hat{\sigma}^2_{e0}\right).$$

   In the implemented sae R package, REML method is used.

2. Use the estimates in $\hat{\theta}_0$ to simulate census welfares[13] from the fitted model as follows:

$$\ln(y^*_{ch}) = x_{ch}\hat{\beta}_0 + \eta^*_c + e^*_{ch},$$

   where the area effects η^*_c are generated as

$$\eta^*_c \sim N\left(0, \hat{\sigma}^2_{\eta 0}\right)$$

   and the household-specific errors are generated as

$$e^*_{ch} \sim N\left(0, \hat{\sigma}^2_{e0}\right).$$

3. From the simulated census vector y^*_c, we calculate the true value of the indicator of interest for each area c:

$$\tau^*_c = f\left(y^*_c\right),$$

   where $f(y^*_c)$ can be any indicator function such as the FGT indicators.

4. Since the survey is regarded as a subset of the census, the survey sample s_c is now extracted and, using the corresponding sample welfares $y^*_{c,s}$ (newly generated in step 2), one fits the model (1). This yields bootstrap estimates of the model parameters,

$$\hat{\theta}^* = \left(\hat{\beta}^*, \hat{\sigma}^{2*}_{\eta}, \hat{\sigma}^{2*}_{e}\right).$$

5. With the bootstrap vectors of sample welfare $y^*_{c,s}$, $c = 1, \ldots, C$, obtain the EB estimators $\hat{\tau}^{EB*}_c$, using MC simulation if needed. The estimation procedure is exactly the same EB procedure based on the original sample, but using the bootstrap sample data $y^*_{c,s}$, $c = 1, \ldots, C$ and the corresponding bootstrap model parameter estimates $\hat{\theta}^*$ from step 4.

6. Repeat steps 2 to 5 a sufficiently large number of times B. In each bootstrap replicate b, $\tau^{*(b)}_c$ is the true value of the indicator obtained from the $\mathrm{b_{th}}$ simulated census and $\hat{\tau}^{EB*(b)}_c$ is the corresponding EB estimator obtained from the extracted sample. The parametric bootstrap approximation of the MSE is then given by:

$$mse_B\left(\hat{\tau}^{EB}_c\right) = \frac{1}{B}\sum_{b=1}^{B}\left(\hat{\tau}^{EB*(b)}_c - \tau^{*(b)}_c\right)^2.$$

[13] Note that here welfares are generated for all households in the census, sampled and non-sampled

3.2 H3 CensusEB

The addition of EB prediction developed by Van der Weide (2014) represented a landmark update to the PovMap project of the World Bank and its poverty mapping agenda.

Let $e_{c,s}$ be the vector with the marginal residuals $e_{ch} = \ln(y_{ch}) - x_{ch}\beta$ for the households in the survey. The predictor of the location effect η_c proposed by Van der Weide (2014) is an extension of the best predictor $E[\eta_c|e_{c,s}]$ for the heteroscedastic nested error model with normality, incorporating also the survey weights to account for complex sampling designs. This predictor is given by

$$\hat{\eta}_c = \hat{\gamma}_c \left(\sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} \right)^{-1} \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} \hat{e}_{ch} \tag{2}$$

where w_{ch} is the survey weight for household h in location c, $\hat{\sigma}^2_{ech}$ is the estimated household-specific error variance using the alpha model as in ELL (2002),

$$\hat{\gamma}_c = \frac{\hat{\sigma}^2_\eta}{\hat{\sigma}^2_\eta + \sum_h w^2_{ch} \left(\sum_h w_{ch} \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} \right)^{-1}}$$

and $\hat{\sigma}^2_\eta$ is an estimator of σ^2_η. For this, Van der Weide (2014) considers the extended version of Henderson's method III (Henderson, 1953) obtained accounting for survey weights as proposed in Huang and Hidroglou (2003). Finally, $\hat{e}_{ch} = \ln(y_{ch}) - x_{ch}\hat{\beta}$ is the estimated marginal residual from the GLS estimator $\hat{\beta}$ that accounts for heteroscedasticity and survey weights, given by

$$\hat{\beta} = \left\{ \sum_c \left(\sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} x_{ch} x^T_{ch} - \gamma_c \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} \bar{x}_{c,w} \bar{x}^T_{c,w} \right) \right\}^{-1} \sum_c \left(\sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} x_{ch} y_{ch} - \gamma_c \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} \bar{x}_{c,w} \bar{y}_{c,w} \right),$$

where all the sums are in the survey data, and $\bar{y}_{c,w}$ and $\bar{x}_{c,w}$ are the weighted sample means,

$$\bar{y}_{c,w} = \left(\sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} \right)^{-1} \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} y_{ch}, \quad \bar{x}_{c,w} = \left(\sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} \right)^{-1} \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech}} x_{ch}.$$

Van der Weide (2014) also proposed the following estimator of the variance of $\hat{\eta}_c$:

$$\widehat{\text{var}}[\hat{\eta}_c] = \hat{\sigma}^2_\eta - \hat{\gamma}^2_c \left\{ \hat{\sigma}^2_\eta + \sum_h \left(\frac{w_{ch}}{\hat{\sigma}^2_{ech}} \right)^2 \hat{\sigma}^2_{ech} \right\}. \tag{3}$$

The EB estimator in MR (2010) requires to link the survey and census households, which may not be possible in real applications. In fact, the survey sample is likely not a subset of the available census. Hence, here we consider the CensusEB estimator, which avoids this step. In fact, in most cases, the number of sample households for a given area is much smaller than the number of census households and, in such cases, the CensusEB estimator is expected to perform very much like the original EB.

CensusEB small area estimators are obtained with a similar MC simulation approach as the one used by MR (2010) and described in Section 3.1, but simulating the vectors of welfare for all the census households (instead of the non-sampled ones only) and hence not appending the welfare for the survey units. The

CensusEB is extended by accounting for the sampling design and heteroscedasticity, similarly as proposed by Van der Weide 2014. This procedure is implemented by also incorporating household expansion factors (household sizes) taken from the census. The proposed MC simulation procedure for the approximation of the extended CensusEB estimator is:

1. Fit model (1) to the survey data. This yields the set of parameter estimates:

$$\hat{\theta}_0 = \left(\hat{\beta}_0, \hat{\sigma}^2_{\eta 0}, \hat{\sigma}^2_{e0}\right).$$

   The currently implemented procedure fits the model either via FGLS and decomposing residuals as in the traditional ELL procedure, or using H3 method.

2. Use the model parameter estimates obtained in step 1 to simulate a vector of welfare for the N_c census households in area c, $y^*_c = (y^*_{c1}, \ldots, y^*_{cN_c})^T$, where each y^*_{ch} is obtained as

$$\ln(y^*_{ch}) = x_{ch}\hat{\beta}_0 + \eta^*_c + e^*_{ch},$$

   where, if area c is in the sample, its location effect is generated as

$$\eta^*_c \sim N\left(\hat{\eta}_{c0}, \widehat{\text{var}}\left[\hat{\eta}_{c0}\right]\right),$$

   with $\hat{\eta}_{c0}$ given by

$$\hat{\eta}_{c0} = \hat{\gamma}_c \left(\sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech0}}\right)^{-1} \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech0}} \hat{e}_{ch0},$$

   for $\hat{e}_{ch0} = \ln(y^*_{ch}) - x_{ch}\hat{\beta}_0$,

$$\hat{\gamma}_{c0} = \frac{\hat{\sigma}^2_{\eta 0}}{\hat{\sigma}^2_{\eta 0} + \sum_h w^2_{ch} \left(\sum_h w_{ch} \sum_h \frac{w_{ch}}{\hat{\sigma}^2_{ech0}}\right)^{-1}}.$$

   and

$$\widehat{\text{var}}\left[\hat{\eta}_{c0}\right] = \hat{\sigma}^2_{\eta 0} - \hat{\gamma}^2_{c0} \left\{\hat{\sigma}^2_{\eta 0} + \sum_h \left(\frac{w_{ch}}{\hat{\sigma}^2_{ech0}}\right)^2 \hat{\sigma}^2_{ech0}\right\}.$$

   If area c is not included in the sample, then its effect is generated as $\eta^*_c \sim N\left(0, \hat{\sigma}^2_{\eta 0}\right)$. In absence of heteroscedasticity, the household-specific residuals come from

$$e^*_{ch} \sim N\left(0, \hat{\sigma}^2_{e0}\right)$$

   and, in the case of heteroscedasticity, from

$$e^*_{ch} \sim N\left(0, \hat{\sigma}^2_{ech0}\right).$$

3. The previous step yields a simulated vector of census welfare for each area, $y^*_c = (y^*_{c1}, \ldots, y^*_{cN_c})^T$, which makes use of the fitted model parameters. With the census y^*_c, the indicator in area c is calculated as

$$\tau^*_c = f(y^*_c),$$

where, for the FGT indicator of order $\alpha \geq 0$, we have

$$f_\alpha(y_c^*) = \sum_{h=1}^{N_c} \frac{p_{ch}}{\sum_\ell p_{c\ell}} I\left(y_{ch}^* < z\right) \left(1 - \frac{y_{ch}^*}{z}\right)^\alpha .$$

4. Repeat steps 1 to 3 a large number of times M. Let $\tau_c^{*(m)}$ be indicator obtained in $\mathrm{m_{th}}$ replicate, for $m = 1, \ldots, M$. The extended Census EB estimate is just the average of the M indicators,

$$\hat{\tau}_c^{CEB} = \frac{1}{M} \sum_{m=1}^{M} \tau_c^{*(m)}.$$

As a measure of noise of the extended CensusEB estimators, the MSE under the nested error model is considered. To estimate this MSE, a parametric bootstrap procedure similar to the one in MR (2010), but adapted to the case where the survey is not necessarily a subset of the census is used. This procedure is also described in Molina (2019) for the CensusEB predictor, but it is applied to the extended CensusEB predictor that includes heteroscedasticity and sampling weights:

1. Fit model (1) to the survey data. This yields the set of parameter estimates:

$$\hat{\theta}_0 = \left(\hat{\beta}_0, \hat{\sigma}_{\eta 0}^2, \hat{\sigma}_{e0}^2\right).$$

   In the implemented version, the model is fit either via FGLS and decomposing residuals as in the traditional ELL fitting approach, or by H3.

2. Using the estimates obtained in step 1 as true values of the model parameters, simulate a vector of census welfare $y_c^* = (y_{c1}, \ldots, y_{cN_c})^T$ as follows:

$$\ln(y_{ch}^*) = x_{ch}\hat{\beta}_0 + \eta_c^* + e_{ch}^*,\ h = 1, \ldots, N_c,$$

   where the location effect is generated as

$$\eta_c^* \sim N\left(0, \hat{\sigma}_{\eta 0}^2\right).$$

   In the homoscedastic case, household-specific errors are generated as

$$e_{ch}^* \sim N\left(0, \hat{\sigma}_{e0}^2\right)$$

   and, in the case of heteroscedasticity, as

$$e_{ch}^* \sim N\left(0, \hat{\sigma}_{ech0}^2\right).$$

3. With the simulated census of welfare $y_c^* = (y_{c1}, \ldots, y_{cN_c})^T$ for each area $c = 1, \ldots, C$, produce indicators of interest:

$$\tau_c^* = f\left(y_c^*\right).$$

   where $f\left(y_c^*\right)$ can be any indicator function, such as one of the FGT poverty indicators.

4. Use the model parameter estimates obtained in step 1 to obtain new sample welfares, $y^*_{ch,s}$, for every location c as follows:

$$\ln(y^*_{ch,s}) = x_{ch,s}\hat{\beta}_0 + \eta^*_c + e^*_{ch,s}.$$

   (a) The estimates $\hat{\beta}_0$ come from step 1.

   (b) η^*_c is the same one generated in step 2. Specifically, all locations present in the survey are matched to the census locations and the same value of η^*_c that was simulated for that location in step 2 is applied to the survey households within the same location.

   (c) The household-specific errors are simulated as

$$e^*_{ch,s} \sim N\left(0, \hat{\sigma}^2_{e0}\right)$$

   in the homoscedastic case. When there is heteroscedasticity, they are simulated as

$$e^*_{ch,s} \sim N\left(0, \hat{\sigma}^2_{ech0}\right).$$

   Unlike the original parametric bootstrap procedure in MR (2010) described in Section 3, here the sample errors $e^*_{ch,s}$ are not a subset of the census errors simulated in step 2, although they are generated from exactly the same model (with the same variance).

5. Fit the model (1) to the newly simulated survey data from step 4. This yields a bootstrap vector of estimated model parameters:

$$\hat{\theta}^* = \left(\hat{\beta}^*, \hat{\sigma}^{2*}_{\eta}, \hat{\sigma}^{2*}_{e}\right).$$

6. Obtain the bootstrap extended CensusEB estimator $\hat{\tau}^{CEB*}_{c,b}$ through MC simulation using the bootstrap parameter estimates $\hat{\theta}^*$ from step 5, by the approach described in the previous section.

7. Repeat steps 2 to 6 a sufficiently large number of times, B. Let $\tau^{*(b)}_c$ be the true value and $\hat{\tau}^{CEB*(b)}_{c,b}$ be the CensusEB estimator obtained in the $\mathrm{b_{th}}$ replicate of the bootstrap procedure. A parametric bootstrap estimator of the MSE is given by:

$$mse_B\left(\hat{\tau}^{CEB}_c\right) = \frac{1}{B}\sum_{b=1}^{B}\left(\hat{\tau}^{CEB*(b)}_c - \tau^{*(b)}_c\right)^2.$$

3.3 Marhuenda et al's twofold nested model extension

If the model assumes only subdomain random effects and aggregates them to the domain level, there is a risk of incorrect model specification. This risk is particularly likely when the covariates used in the model do not fully explain the variation between domains, leading to a significant underestimation of the true Mean Squared Error (MSE), as emphasized by Marhuenda et al. (2017). The twofold nested error model proposed by Marhuenda et al. (2017) extends the model presented by Molina and Rao's EB (2010). The primary advantage of the twofold-nested error model over Molina and Rao's onefold nested error models is its ability to provide point and noise estimates at two different levels of aggregation.

The twofold-nested error model that expands upon the onefold-nested error model by including a subarea random effect:

$$y_{ach} = x_{ach}\beta + \eta_a + \eta_{ac} + e_{ach}, \quad h = 1, \ldots, N_{ac}; \quad c = 1, \ldots, C_a; \quad a = 1, \ldots, A$$

In this modified model, a subdomain c within area a, and its random location effect, is captured by η_{ac}. As in Molina and Rao's (2010) approach, it is essential to highlight that the random location effects and the household-specific errors are considered to be statistically independent in this context:

$$\eta_a \sim N(0, \sigma_\eta^2), \quad \eta_{ac} \sim N(0, \sigma_{\eta_c}^2), \quad e_{ach} \sim N(0, \sigma_e^2)$$

Predictors for households within a sampled subarea c within the larger area a are provided by the EB method:

$$E[y_{ach}|\eta_a + \eta_{ac}] = x_{ach}\hat{\beta} + \gamma_{ac}\left(\overline{y}_{ac} - \overline{x}_{ac}\hat{\beta}\right) + \left(\frac{\sigma_e^2}{\sigma_{\eta_c}^2}\right)^2 \frac{\psi_a}{w_{ac}} \sum_{i=1}^{C_a} \gamma_{ai}\left(\overline{y}_{ai} - \overline{x}_{ai}\hat{\beta}\right)$$

In the absence of heteroscedasticity, w_{ac} corresponds to the number of units sampled in subarea c within area a. Furthermore, $\overline{x}_{ac}$ and $\overline{y}_{ac}$ represent sample means of x and y within the subarea ac. The shrinkage parameters, denoted as γ and ψ, are determined as follows:

$$\gamma_{ac} = \frac{\sigma_{\eta_c}^2}{\sigma_{\eta_c}^2 + \frac{\sigma_e^2}{w_{ac}}}$$

$$\psi_a = \frac{\sigma_{\eta_c}^2}{\sigma_e^2 + \sigma_\eta^2 \sum_{c=1}^{C_a} (1 - \gamma_{ac})\, w_{ac}}$$

For households situated in an unsampled subarea within a sampled geographical area, the data from the survey is leveraged to influence the random location effect for that area. As a result, the Empirical Best (EB) predictor for such households is as follows:

$$E[y_{ach}|\hat{\eta}_a + \hat{\eta}_{ac}] = x_{ach}\hat{\beta} + \left(\frac{\sigma_e^2}{\sigma_{\eta_c}^2}\right)^2 \psi_d \sum_{i=1}^{C_a} \gamma_{ai}\left(\overline{y}_{ai} - \overline{x}_{ai}\hat{\beta}\right)$$

This represents a notable benefit of the approach as it allows residences within non-sampled subareas to utilize the information gathered from the survey within the larger geographical area. In instances where an area has not been sampled, it becomes impractical to account for model error. Consequently, the random location effects for those unsampled areas align with those of the ELL method, leading to a simplified predictor, which can be expressed as:

$$E[y_{ach}] = x_{ach}\hat{\beta}$$

The conditional variances for households depend on whether they are within sampled or non-sampled subareas, and the estimation of various parameters is carried out using the restricted maximum likelihood fitting method for linear mixed models. The best predictor for poverty is approximated through a Monte Carlo simulation, and the mean squared error (MSE) estimate is obtained via a parametric bootstrap procedure, following the methodology presented in González-Manteiga et al. (2008), but adapted to the two-fold case.

Conditional variances for households within a sampled subarea are determined by:

$$\sigma_e^2 \left[1 + \psi_a \left(1 + \gamma_{ac}(\gamma_{ac} - 2)\right)\right] + \sigma_{\eta_c}^2 (1 - \gamma_{ac})$$

Conditional variance for households located in an unsampled subarea within a sampled larger one is specified as:

$$\sigma_e^2(1 + \psi_a) + \sigma_{\eta_c}^2$$

Conditional variance for households in areas that are not included in the sampling is determined by:

$$\sigma_e^2 + \sigma_{\eta_c}^2 + \sigma_\eta^2$$

The conditional variances and EB predictors provided rely on the presence of a consistent estimator for $\hat{\beta}, \hat{\sigma}_{\eta_c}^2, \hat{\sigma}_\eta^2$, and $\hat{\sigma}_e^2$. In the SAE Stata package, these parameters are calculated using a restricted maximum likelihood (REML) fitting technique for linear mixed models relying on Stata's `xtmixed` command. It is important to note that the twofold model in Stata cannot handle survey weights or heteroskedasticity.[14]

Monte Carlo simulations for point estimates and bootstraps for MSE estimates are similar to those presented in Section 3.1.

3.4 Empirical Best (EB) in the `sae` command

The `sae model/sim h3`, `sae model/sim reml` and `sae model/sim reml2` subcommands, like the `sae model/sim ell` subcommand, consist of two stages: a model stage and a simulation stage. The model stage fits the model to the data, while the simulation stage not only fits the model but also runs simulations based on the underlying model's assumptions, producing final small area estimates as a `.dta` dataset.

The Empirical Best (EB) method developed by Molina and Rao (2010) and implemented via the `sae model/sim reml` subcommand in the `sae` package, employs Restricted Maximum Likelihood (REML) for mmodel fitting. Meanwhile, the CensusEB approach, which accounts for heteroskedasticity and incorporates survey weights during model fitting and parameter estimation, is implemented via the `sae model/sim h3` subcommand. This approach uses Generalized Least Squares (GLS), with variance parameters estimated through Henderson's Method III (H3) (Henderson 1953). Notably, both approaches produce similar results when heteroskedasticity and survey weights are not considered (see Corral et al. (2021), Figure A4).

The latest edition of the Stata sae package also supports the use of twofold nested-error models, following the work of Marhuenda et al. (2017). However, this implementation shares the same limitations as the `sae model/sim reml` subcommand, in that it does not account for survey weights or heteroskedasticity. Despite this drawback, twofold nested-error models offer the advantage of producing estimates that are optimal at two levels of aggregation. This results in accurately estimated mean squared errors (MSEs) at both levels under the assumed model, compared to onefold nested-error models that include only cluster or area effects (for further discussion, see Corral et al. (2022), Figure 4.2 and Marhuenda et al. (2017)).

The specific subcommands are designed as follows:

[14] Although heteroscedasticity doesn't introduce bias into the coefficient estimates, it does reduce their accuracy. This reduced accuracy raises the probability that the coefficient estimates deviate further from the true population values.

- `sae model/sim h3`: Fits the CensusEB model using GLS with Henderson's Method III, accounting for heteroskedasticity and survey weights within onefold nested-error models.
- `sae model/sim reml`: Implements EB and CensusEB estimation under the REML approach with onefold nested-error models.
- `sae model/sim reml2`: Applies CensusEB estimation under REML for twofold nested-error models.

3.4.1 EB model-fitting under the `h3` subcommand

The `sae model/sim h3` subcommand enables CensusEB estimation under Generalized Least Squares (GLS) with Henderson's Method III, specifically designed to account for survey weights and heteroskedasticity within one-fold nested error models. To obtain GLS estimates, the approach requires estimating variance components, which is done using a modified version of Henderson's Method III (Henderson 1953). This modification, which accounts for the use of survey weights, was presented by Huang and Hidiroglou (2003) and is further developed and elaborated by Van der Weide (2014) and presented Nguyen et al. (2018).

The method produces CensusEB estimates for areas included in both the survey and the census. When neither weights nor heteroskedasticity are specified, the results are nearly identical to those obtained using `sae model/sim reml` (see Corral, Molina, and Nguyen 2021).

The command to fit the model is:

`sae model h3` *`depvar indepvars`* `[if] [in] [aw],` *`options`*

Required

- `area()`: The `area` option is necessary and specifies at which level the clustering is done, it indicates at which level the η_c is obtained at. The only constraint is that the variable must be numeric, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006).
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[15]

Optional

- `zvar()`: The `zvar` option is necessary for specifying the alpha model, the user must place the independent variables under the option
- `yhat()`: The `yhat` option is also a part of the alpha model (heteroskedasticity). Variables listed here will be interacted with the predicted $\hat{y} = X\beta$ from the OLS model.
- `yhat2()`: The `yhat2` option is also a part of the alpha model. Variables listed here will be interacted with the predicted $\hat{y}^2 = (X\beta)^2$ from the OLS model.
- `alfatest()`: The `alfatest` option is useful for alpha model selection. It requests the command to output the the dependent variable of the alpha model, which depends on the current model fit, for users to model for heteroskedasticity.

[15] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

- `method()`: The method option allows users to specify different inverters for the GLS estimation. Options include `cholinv, cholinv_la, luinv, luinv_la`. Changing the inverter can yield considerable speed gains.

Example 1 Model-fitting is done by calling the `sae` command and the `model h3` subcommands. The user must have loaded their data beforehand. The simulated data used throught these examples are created as illustrated in the appendix 4.1 at the end of this document.

As can be seen in the example below, the command requires the user to specify the area level for the random effects. The command also provides information that allows evaluating the model fit, including key information about the random effects. Aside from removing perfectly collinear covariates, the command will also exclude from the model areas where there are less than 3 observations.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. *================================================================================
. // Model fitting under H3 (see Van der Weide 2014; Nguyen et al. 2018)
. *================================================================================
. //import data
. use "`p´/sae/survey.dta", clear

.         //fit model
.         sae model h3 Y x1 x2, area(HID)
You chose H3, parameters must be obtained via bootstrap I changed it for you.
WARNING: 0 observations removed due to less than 3 observations in the cluster.

OLS model:
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

```
GLS model:
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378468	.0089458	4.23	0.000	.0203133	.0553803
x2	-.0362173	.0106182	-3.41	0.001	-.0570286	-.0154059
_cons	2.989545	.0107473	278.17	0.000	2.968481	3.01061

```
Comparison between OLS and GLS models:
```

Variable	bOLS	bGLS
x1	.03085459	.03784677
x2	-.03264075	-.03621727
_cons	2.9932721	2.9895455

```
Model settings
```

```
Error decomposition                             H3

Beta model diagnostics
------------------------------------------------------------------------------
Number of observations                  =    4000
Adjusted R-squared                      =    .00390058
R-squared                               =    .00439875
Root MSE                                =    .29774216
F-stat                                  =    8.8297455

Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                           =    .0255287
Ratio of sigma eta sq over MSE          =    .28797048
Variance of epsilon                     =    .06318513
------------------------------------------------------------------------------
                  <End of first stage>
```

Example 2 The `alfatest()` option allows users to obtain the dependent variable for the alpha model, a model for heteroskedasticity introduced in Elbers, Lanjouw, and Lanjouw (2002). The dependent variable used for the alpha model can be created by specifying the `alfatest()` option combined with `zvar()`. Doing this creates 3 variables in the data that can be used to simplify selection of the alpha model. The first two variables will have as prefix the string specified in the `alfatest()` option: 1) `het_alpha`, the dependent variable for the alpha model, and 2) `het_xb`, the predicted linear fit of the main model that may be used as a covariate in the alpha model. The third and final variable is `_est_bGLS`, which indicates the observations that were used for the GLS model. When the alpha model is run, the command gives additional information related specifically to the alpha model.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. //import data
. use "`p´/sae/survey.dta", clear

.
.         //Create dependent variable for alpha model
.         // This is useful for alpha model selection
.         // Just add any covariate to the zvar() option and specify alfatest()
.         sae model h3 Y x1 x2, area(HID) zvar(x1) alfatest(alpha)
You chose H3, parameters must be obtained via bootstrap I changed it for you.
WARNING: 0 observations removed due to less than 3 observations in the cluster.

OLS model:
------------------------------------------------------------------------------
           Y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
          x1 |   .0308546   .0096996     3.18   0.001     .0118438    .0498654
          x2 |  -.0326407   .0120273    -2.71   0.007    -.0562138   -.0090677
       _cons |   2.993272   .0079874   374.75   0.000     2.977617    3.008927
------------------------------------------------------------------------------

Alpha model:
------------------------------------------------------------------------------
    Residual | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
```

```
          x1 |   .0877675   .0744967     1.18   0.239    -.0582433    .2337782
       _cons |  -4.139203   .0605114   -68.40   0.000    -4.257803   -4.020602
------------------------------------------------------------------------------

GLS model:

------------------------------------------------------------------------------
           Y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
          x1 |   .0378593   .0088692     4.27   0.000      .020476    .0552427
          x2 |  -.0365954   .0106099    -3.45   0.001    -.0573905   -.0158003
       _cons |   2.989608   .0106462   280.82   0.000     2.968742    3.010474
------------------------------------------------------------------------------

Comparison between OLS and GLS models:

------------------------------------
    Variable |      bOLS         bGLS
-------------+----------------------
          x1 |  .03085459    .03785934
          x2 | -.03264075    -.0365954
       _cons |  2.9932721    2.9896084
------------------------------------

Model settings
------------------------------------------------------------------------------
Error decomposition                             H3

Beta model diagnostics
------------------------------------------------------------------------------
Number of observations                    =    4000
Adjusted R-squared                        =    .00390058
R-squared                                 =    .00439875
Root MSE                                  =    .29774216
F-stat                                    =    8.8297455

Alpha model diagnostics
------------------------------------------------------------------------------
Number of observations                    =    4000
Adjusted R-squared                        =    .00010932
R-squared                                 =    .00035935
Root MSE                                  =    2.2370467
F-stat                                    =    1.437214

Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                             =    .0255287
Ratio of sigma eta sq over MSE            =    .28797048
Variance of epsilon                       =    .06318513
------------------------------------------------------------------------------

                  <End of first stage>

.         //new variables added to the data:
.         // 1: alpha_alpha, the alpha model´s dependent variable
.         // 2: alpha_xb, the predicted linear fit of the OLS model
.         // 3: _est_bGLS, dummy variable indicating which observations are use
```

Example 3 Finally, modeling for heteroskedasticity is simple under the `sae model h3` command as users can easily add the `yhat()` and `yhat2()` options where variables specified under the option are interacted with the first stage OLS's linear fit. In the case of the `yhat2()` option, the variable indicated will be interacted with the square of the OLS linear fit. If interactions of the variable are not desired, then the user can specify

these variables under the `zvar()` option.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. //import data
. use "`p´/sae/survey.dta", clear

.
.         //fit model with alpha model including interactions with linear fit
.         sae model h3 Y x1 x2, area(HID) yhat(x1)
You chose H3, parameters must be obtained via bootstrap I changed it for you.
WARNING: 0 observations removed due to less than 3 observations in the cluster.

OLS model:
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

Alpha model:

Residual	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1_yhat	.0292174	.0246752	1.18	0.236	-.0191451	.0775799
_cons	-4.139464	.0605105	-68.41	0.000	-4.258063	-4.020866

GLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378595	.0088688	4.27	0.000	.020477	.055242
x2	-.0365952	.0106081	-3.45	0.001	-.0573867	-.0158036
_cons	2.989609	.0106456	280.83	0.000	2.968744	3.010474

Comparison between OLS and GLS models:

Variable	bOLS	bGLS
x1	.03085459	.03785946
x2	-.03264075	-.03659516
_cons	2.9932721	2.9896087

```
Model settings
------------------------------------------------------------------------------
Error decomposition                             H3

Beta model diagnostics
------------------------------------------------------------------------------
Number of observations                    =    4000
Adjusted R-squared                        =    .00390058
R-squared                                 =    .00439875
Root MSE                                  =    .29774216
F-stat                                    =    8.8297455
```

```
Alpha model diagnostics
------------------------------------------------------------------------------
Number of observations                    =    4000
Adjusted R-squared                        =    .00011275
R-squared                                 =    .00036279
Root MSE                                  =    2.2370429
F-stat                                    =    1.4509535

Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                             =    .0255287
Ratio of sigma eta sq over MSE            =    .28797048
Variance of epsilon                       =    .06318513
------------------------------------------------------------------------------

                  <End of first stage>

.
```

Example 4 The option `method` has been added to the `sae package` to improve model fitting and simulation speed. Inversion techniques include: `invsym` (the default), `cholinv` and `cholinv_la` use Cholesky decomposition for symmetric, positive-definite matrices, while `luinv` and `luinv_la` rely on LU decomposition and are more general, applicable to any square matrix. Each inverter has its advantages depending on the type of matrix and the structure of the problem. In the example below, the methods seem to be equally as efficient for the given data.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

.
. *================================================================================
. // Test different inverters
. // Changing inverters can lead to faster model fitting and faster simulations
. *================================================================================
. // The list below includes all the possible inverters
. // invsym is the default method
. local inverters invsym cholinv cholinv_la luinv luinv_la

. local counter = 1

. foreach inverter of local inverters{
  2.         timer on `counter´
  3.         forval z = 1/100 {
  4.                 qui:sae model h3 Y x1 x2, area(HID) method(`inverter´)
  5.         }
  6.         timer off `counter´
  7.         local counter = `counter´+1
  8. }

.
. timer list
   1:      4.02 /        1 =       4.0170
   2:      4.56 /        1 =       4.5600
   3:      4.04 /        1 =       4.0400
   4:      4.03 /        1 =       4.0300
   5:      4.03 /        1 =       4.0340
```

3.4.2 EB Small Area Estimates under the `h3` subcommand

The simulation stage can be used to obtain SAE point estimates as well as estimates of the noise - note that these can be done separately. The command for fitting the model and proceeding to the simulation stage, includes Monte Carlo for CensusEB estimation and Bootstrap MSE. CensusEB estimates differ from EB when survey estimates for a given indicator are not combined with predictions based solely on census data to obtain final point estimates by area. This approach is similar to the one outlined by Corral et al. (2021). MSEs are estimated via a parametric bootstrap following the approach from González-Manteiga et al. (2008) and noted in the relevant section.The main command for simulations is:

`sae sim h3` *`depvar indepvars`* `[if] [in] [aw],` *`options`*

Required

- `area()`: The `area` option is necessary and specifies at which level the clustering is done, it indicates at which level the η_c is obtained at. The only constraint is that the variable must be numeric, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006).
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[16]
- `uniqid()`: option specifies the numeric variable that indicates the unique identifiers in the census and survey dataset. This is necessary to ensure replicability of the analysis, and the name should match the one of the unique ier from the source dataset.
- `mcrep(integer)`: Indicates the number of Montecarlo Simulations to be executed.
- `bsrep(integer)`: Indicates the number of Bootstrap populations to be executed to obtain the MSE estimate. Note that every bootstrap population will execute the number of Montecarlo simulations indicated in mcrep().
- `matin()`: The matin option indicates the path and filename of the Mata format target dataset. The dataset must be created with the `sae data import` command.
- `aggids()`: The aggids option indicates the different aggregation levels for which the indicators are to be obtained, values placed here treml2EB the command how many digits to the left to move to get the indicators at that level. Using the hierarchical id specified in the area option, AAMMEEE, if the user specifies 0, 3, 5, and 7, it would lead to aggregates at each of the levels E, M, A and the national level. Note that it is NOT advised to report results at a higher level than area(), it is likely that the MSE is underestimated (see Corral et al. 2021). Users should specify 0 if they are not using a hierarchical id.
- `pwcensus()`: Indicates the variable which corresponds to the expansion factors to be used in the target (census) dataset, it must always be specified. The user must have added the variable to the imported data (`sae data import`) i.e. the data.
- `indicators()`: The `indicators` option is used to request the indicators to be estimated from the simulated vectors of welfare. The list of possible indicators is:

[16] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

- The set of Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) FGT_0, FGT_1, and FGT_2 ; also known as poverty head count, poverty gap, and poverty severity respectively.
- The set of inequality indexes: Gini, and Generalized Entropy Index with $\alpha = 0, 1, 2$

Optional

- `zvar()`: The `zvar` option is necessary for specifying the alpha model, the user must place the independent variables under the option
- `yhat()`: The `yhat` option is also a part of the alpha model (heteroskedasticity). Variables listed here will be interacted with the predicted $\hat{y} = X\beta$ from the OLS model.
- `yhat2()`: The `yhat2` option is also a part of the alpha model. Variables listed here will be interacted with the predicted $\hat{y}^2 = (X\beta)^2$ from the OLS model.
- `ydump()`: The user must provide path and filename for a Mata format dataset to be created with the simulated dependent variables at the `uniqid()` level.
- `addvars()`: The addvars option allows users to add variables to the dataset created from the simulations. These variables must have been included into the target dataset created with the sae data import command.
- `lny`: The option indicates that the dependent variable in the welfare model is in log form. Specifying this indicates that the simulated vectors need to be transformed before applying the poverty line specified in `pline` or `plinevar`.
- `bcox`: option tells command to execute bcskew0 (Box-Cox transform - Zero skewness) on the dependent variable of your model.
- `lnskew`: option tells command to execute lnskew0 (log shift transform - Zero skewness) on the dependent variable of your model.
- `lnskew_w`: option tells command to execute lnskew0 (log shift transform - Zero skewness) on the dependent variable of your model with survey weights.The `_w` suffix requests the command to use survey weights for the transformation.
- `s2s_spec`: option indicates that the survey is not a subset of the census and allows for cases where an area used for the model is not present in the census.
- `seed()`: The option ensures replicability. Users should be aware that Stata's default pseudo-random number generator in Stata 14 is different than that of previous versions. If not specified, the default seed of 123456789 is used.
- `plinevar()`: Option allows users to indicate a variable in the target data set which is to be used as the threshold for the Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) to be estimated from the simulated vectors. Te user must have added the variable in the `sae data import` command when preparing the target dataset. Only one variable may be specified.

- `plines()`: The option allows users to explicitly indicate the threshold to be used, this option is preferred when the threshold is constant across all observations. Additionally, it is possible to specify multiple lines, separated by a space.
- `method()`: The method option allows users to specify different inverters for the GLS estimation. Options include cholinv, cholinv_la, luinv, luinv_la. Changing the inverter can yield considerable speed gains.

Example 1 First, the `sae data import` command is used to bring in the census data, specifying relevant variables from the census (x1, x2, hhsize, pvar for the poverty line), along with the area (HID) and unique household ID (`hhid`). This data is saved to a Mata file.

Users have two options for specifying the poverty thresholds in the simulation stage: `plinevar()` and `plines()`. The first, `plinevar()` allows users to indicate a variable in the census microdata with the threshold. The variable must have been imported when using the `sae data import` command and only allows for specifying one variable. The second, `plines()`, allows users to indicate multiple thresholds with the caveat that these apply equally to all households in the census microdata.

In the example below the dependent variable is modeled in log terms, with a back-transformation applied to calculate poverty and other indicators (`lny`). The `uniqid(hhid)` option ensures replicability by using a unique household ID, while `pwcensus(hhsize)` adjusts for household-level population expansion factors. The poverty line is defined by the `plinevar(pvar)` variable, and simulations are run using 100 Monte Carlo and 100 bootstrap replications. Poverty indicators, including the headcount poverty ratio (`indicators(fgt0)`), are calculated and aggregated at the area level (`aggids(0)`).

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

.
.
. *================================================================================
. // Importing the census for small area estimates
. // Note that for the census we add the population expansion factors
. // to the varlist(). We´ve also added a variable containing the poverty line
. // "pvar".
. *================================================================================
. tempfile census_mata

.         sae data import, datain("`p´/sae/census.dta") varlist(x1 x2 hhsize pvar) ///
>         area(HID) uniqid(hhid) dataout(`census_mata´)
Saving data variables into mata matrix file (6)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
.....
.
. *================================================================================
. //Obtain final small area estimates based on H3 census EB approach
. *================================================================================
. //Bring in the survey data
. use "`p´/sae/survey.dta", clear

.         //The following model specifications are used:
.         //1: Assumes that the Y variable is in log terms, we request to back //
```

```
.         //   transform before obtaining poverty and other indicators (lny)
.         //2: The uniqid() option is necessary to ensure replicability
.         //3: We indicate that the simulations are done at the household level
.         //   and indicate population expansion factors to obtain values for the
.         //   population
.         //4: We indicate that the poverty line to be used is in the variable
.         //   "pvar" within the census data, plinevar()
.         //5: We´ve requested for the estimates to be obtained from 100 Monte
.         //       Carlo repetitions, mcrep()
.         //6: We´ve requested noise estimates (MSE) from 100 bootstrap replications,
.         //   bsrep()
.         //7: We´ve requested to estimate FGT0, headcount poverty, indicators(fgt0)
.         //8: We´ve requested for results to be produced at the same level as the
.         //   area() variable, aggids(0).
.         //9: We´ve indicated that the census data is in the mata file `census_mata´,
.         //   which was created using the sae data import command.
.         sae sim h3 Y x1 x2, area(HID)  lny uniqid(hhid) pwcensus(hhsize) ///
>         plinevar(pvar) mcrep(100) bsrep(100) indicators(fgt0) ///
>         aggids(0) matin(`census_mata´)
You chose H3, parameters must be obtained via bootstrap I changed it for you.
WARNING: 0 observations removed due to less than 3 observations in the cluster.
```

OLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0308546	.0096996	3.18	0.001	.0118438	.0498654
x2	-.0326407	.0120273	-2.71	0.007	-.0562138	-.0090677
_cons	2.993272	.0079874	374.75	0.000	2.977617	3.008927

GLS model:

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378468	.0089458	4.23	0.000	.0203133	.0553803
x2	-.0362173	.0106182	-3.41	0.001	-.0570286	-.0154059
_cons	2.989545	.0107473	278.17	0.000	2.968481	3.01061

Comparison between OLS and GLS models:

Variable	bOLS	bGLS
x1	.03085459	.03784677
x2	-.03264075	-.03621727
_cons	2.9932721	2.9895455

Model settings

```
Error decomposition                             H3
```

Beta model diagnostics

```
Number of observations                  =    4000
Adjusted R-squared                      =    .00390058
R-squared                               =    .00439875
Root MSE                                =    .29774216
F-stat                                  =    8.8297455
```

```
Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                                 =    .0255287
Ratio of sigma eta sq over MSE                =    .28797048
Variance of epsilon                           =    .06318513
------------------------------------------------------------------------------

                    <End of first stage>
file C:\Users\WB378870\AppData\Local\Temp\ST_6348_000004.tmp saved as .dta format
Number of observations in target dataset:

  20000

Number of clusters in target dataset:

  400

Number of simulations: 100
Each dot (.) represents 1 simulation(s).
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
..................................................    50
..................................................   100

Number of bootstraps: 100

----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
..................................................  50
..................................................  100

.
. cap: texdoc stlog close
```

Example 2

Because the model assumes that errors are normally distributed, and for poverty the replication of the model's assumptions is crucial to ensure unbiased estimates, transformation of the dependent variable is recommended. In this example, the dependent variable is transformed using `lnskew` option, which uses Stata's `lnskew0` command in the background. If users wish to use survey weights for the transformation, they may call the `lnskew_w` option, which uses a modified version of `lnskew0` in the background. Alternatively, users may also use a Zero-skewness Box-Cox transformation using the option `bcox` which calls Stata's `bcskew0` option in the background.

In this example, we modify the previous model to include a skewed natural log transformation (`lnskew_w`) with survey weights (`aw=hhweight`). The model is adjusted to include the alpha model and additional covariates as specified by the `zvar(x2)` option. In the presence of informative sampling `lnskew_w` allows for a more accurate transformation of the dependent variable. The introduction of the alpha model allows for better replication of the distribution of welfare in the presence of heteroskedasticity.

```
. //Determine personal folder
. local p : sysdir PERSONAL
. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)
.
. *==============================================================================
. // Importing the census for small area estimates
. // Note that for the census we add the population expansion factors
```

```
. // to the varlist(). We´ve also added a variable containing the poverty line
. // "pvar".
. *================================================================================
. tempfile census_mata

.         sae data import, datain("`p´/sae/census.dta") varlist(x1 x2 hhsize pvar) ///
>         area(HID) uniqid(hhid) dataout(`census_mata´)
Saving data variables into mata matrix file (6)
----+--- 1 ----+--- 2 ----+--- 3 ----+--- 4 ----+--- 5
......
.
.
. *================================================================================
. //Obtain final small area estimates based on H3 census EB approach
. //Modify to use skewed natural log transformation with survey weights
. *================================================================================
. //Bring in the survey data
. use "`p´/sae/survey.dta", clear

.         //The following model specifications are modified from before:
.         //1: Use lnskew_w option, note that we use the untransformed dependent
.         //   variable, E_Y. The command will transform the dependent variable
.         //   for the model and will backtransform the simulations. The _w suffix
.         //   requests the command to use survey weights for the transformation.
.         //2: Added alpha model to the simulation.
.         sae sim h3 E_Y  x1 x2 [aw=hhweight], area(HID) zvar(x2) lnskew_w ///
>         uniqid(hhid) pwcensus(hhsize) plinevar(pvar) mcrep(100) bsrep(100) ///
>         indicators(fgt0) aggids(0) matin(`census_mata´)
```

Transform	k	[95% conf. interval]	Skewness
ln(E_Y-k)	.6876876	(not calculated)	4.02e-06

```
You chose H3, parameters must be obtained via bootstrap I changed it for you.
WARNING: 0 observations removed due to less than 3 observations in the cluster.
```

OLS model:

__000001	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0319461	.0100605	3.18	0.001	.0122278	.0516643
x2	-.0337976	.012468	-2.71	0.007	-.0582344	-.0093608
_cons	2.956535	.0082887	356.70	0.000	2.940289	2.97278

Alpha model:

Residual	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x2	-.0352418	.0940514	-0.37	0.708	-.2195792	.1490956
_cons	-4.049284	.0391123	-103.53	0.000	-4.125942	-3.972625

GLS model:

__000001	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0392475	.0092778	4.23	0.000	.0210633	.0574316
x2	-.0374684	.0109095	-3.43	0.001	-.0588505	-.0160862
_cons	2.952631	.011152	264.76	0.000	2.930774	2.974489

```
Comparison between OLS and GLS models:

------------------------------------
    Variable |      bOLS        bGLS
-------------+----------------------
          x1 |   .03194605    .03924746
          x2 |  -.03379757   -.03746837
       _cons |   2.9565347    2.9526313
------------------------------------

Model settings
------------------------------------------------------------------------------
Error decomposition                                 H3

Beta model diagnostics
------------------------------------------------------------------------------
Number of observations                        =    4000
Adjusted R-squared                            =    .00388803
R-squared                                     =    .00438621
Root MSE                                      =    .30872543
F-stat                                        =    8.8044572

Alpha model diagnostics
------------------------------------------------------------------------------
Number of observations                        =    4000
Adjusted R-squared                            =    -.0002128
R-squared                                     =    .00003731
Root MSE                                      =    2.2512436
F-stat                                        =    .14917995

Model parameters
------------------------------------------------------------------------------
Sigma ETA sq.                                 =    .02745597
Ratio of sigma eta sq over MSE                =    .28806601
Variance of epsilon                           =    .06792364
------------------------------------------------------------------------------

                    <End of first stage>
      file C:\Users\WB378870\AppData\Local\Temp\ST_6348_000005.tmp saved as .dta format
Number of observations in target dataset:

        20000

Number of clusters in target dataset:

        400

      Number of simulations: 100
      Each dot (.) represents 1 simulation(s).
      ----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
      ..................................................    50
      ..................................................   100

      Number of bootstraps: 100

      ----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
      ..................................................  50
      ..................................................  100

      .
      . cap:  texdoc stlog close
```

3.4.3 EB model-fitting under the `reml` subcommand

Under the `sae model reml` subcommand, Restricted Maximum Likelihood (REML) fitting is implemented assuming a one-fold nested error model structure. This implementation follows Molina and Rao's (2010) method for EB estimation and Bootstrap Mean Squared Error (MSE). The command is a translation from R's `sae` library, developed by Molina and Marhuenda (2015).

Models fit by **reml do not** incorporate survey weights nor heteroskedasticity. The command uses Stata's `xtmixed` command in the background to fit the model. When choosing this method, the one-fold nested-error model considered is the same one used in Molina and Marhuenda's (2015) `sae` package in R.

The command to fit the model is:

`sae model reml` *depvar indepvars* `[if] [in],` *options*

Required

- `area()`: The `area` option is necessary and specifies at which level the clustering is done, it indicates at which level the η_c is obtained at. The only constraint is that the variable must be numeric, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006).
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[17]

Example 1 Model-fitting under reml is done by calling the `sae` command and the `model reml` subcommands. The user must have loaded their data beforehand. The simulated data used throughout these examples are created as illustrated in the appendix 4.1 at the end of this document. The model is applied to the survey data, where the dependent variable (`Y`) is modeled against independent variables (`x1, x2`), with `HID` as the area variable. The `reml` subcommand does not allow the inclusion of weights or a heteroskedasticity model.

As can be seen in the example below, the command requires the user to specify the area level for the random effects. The command also provides information that allows evaluating the model fit, including key information about the random effects - which comes directly from Stata's `xtmixed`.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

.
. *================================================================================
. // Model fitting under REML
. // The model uses Stata´s xtmixed in the background
. *================================================================================
. //import data
. use "`p´/sae/survey.dta", clear

.         //fit model
.         sae model reml Y x1 x2, area(HID)

Performing EM optimization:
```

[17] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

```
Performing gradient-based optimization:

Iteration 0:  Log restricted-likelihood = -485.59116
Iteration 1:  Log restricted-likelihood = -485.59116

Computing standard errors:

Mixed-effects REML regression                       Number of obs    =  4,000
Group variable: HID                                 Number of groups =    400
                                                    Obs per group:
                                                                 min =     10
                                                                 avg =   10.0
                                                                 max =     10
                                                    Wald chi2(2)     =  29.91
Log restricted-likelihood = -485.59116              Prob > chi2      = 0.0000
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378451	.0089501	4.23	0.000	.0203031	.055387
x2	-.0362165	.0106235	-3.41	0.001	-.0570381	-.0153949
_cons	2.989546	.0107495	278.11	0.000	2.968478	3.010615

Random-effects parameters	Estimate	Std. err.	[95% conf.	interval]
HID: Identity				
sd(_cons)	.1597743	.0070724	.146497	.174255
sd(Residual)	.2513664	.0029632	.2456252	.2572418

```
LR test vs. linear model: chibar2(01) = 708.41        Prob >= chibar2 = 0.0000

.         //The use of REML does not allow for the inclusion of a model for
.         //heteroskedasticity.
.
. cap:  texdoc stlog close
```

3.4.4 EB Small Area Estimates under the `reml` subcommand

The use of the `sae sim reml` subcommand to obtain small area estimates are presented here. The command supports Molina and Rao's (2010) EB small area estimation methods and is a translation from R's sae library from Molina and Marhuenda (2015).

`sae sim reml` *depvar indepvars* `[if] [in],` *options*

Required

- `area()`: The `area` option is necessary and specifies at which level the clustering is done, it indicates at which level the η_c is obtained. The only constraint is that the variable must be numeric, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006).
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[18]

[18] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

- uniqid(): option specifies the numeric variable that indicates the unique identifiers in the census and survey dataset. This is necessary to ensure replicability of the analysis, and the name should match the one of the unique ier from the source dataset.
- mcrep(integer): Indicates the number of Monte Carlo Simulations to be executed.
- bsrep(integer): Indicates the number of Bootstrap populations to be executed to obtain the MSE estimate. Note that every bootstrap population will execute the number of Monte Carlo simulations indicated in mcrep().
- matin(): The matin option indicates the path and filename of the Mata format target dataset. The dataset must be created with the `sae data import` command.
- aggids(): The aggids option indicates the different aggregation levels for which the indicators are to be obtained, values placed here tell the command how many digits to the left to move to get the indicators at that level. Using the hierarchical id specified in the area option, AAMMEEE, if the user specifies 0, 3, 5, and 7, it would lead to aggregates at each of the levels E, M, A and the national level. Note that it is NOT advised to report results at a higher level than area(), it is likely that the MSE is underestimated (see Corral et al. 2021). Users should specify 0 if they are not using a hierarchical id.
- pwcensus(): Indicates the variable which corresponds to the expansion factors to be used in the target (census) dataset, it must always be specified. The user must have added the variable to the imported data (`sae data import`) i.e. the census.
- indicators(): The `indicators` option is used to request the indicators to be estimated from the simulated vectors of welfare. The list of possible indicators is:
  - The set of Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) FGT_0, FGT_1, and FGT_2 ; also known as poverty head count, poverty gap, and poverty severity respectively.
  - The set of inequality indexes: Gini, and Generalized Entropy Index with $\alpha = 0, 1, 2$

Optional

- pwsurvey(string): optional, and only necessary if users are using the appendsvy option.
- lny: The option indicates that the dependent variable in the welfare model is in log form. Specifying this indicates that the simulated vectors need to be transformed before applying the poverty line specified in `pline` or `plinevar`.
- bcox: option tells command to execute bcskew0 (Box-Cox transform - Zero skewness) on the dependent variable of your model.
- lnskew: option tells command to execute lnskew0 (log shift transform - Zero skewness) on the dependent variable of your model.
- CONStant(real 0.0): option adds constant value to dependent variable when doing Box-Cox transformation. If bcox is not specified this is irrelevant.

- `seed()`: The option ensures replicability. Users should be aware that Stata's default pseudo-random number generator in Stata 14 is different than that of previous versions. If not specified, the default seed of 123456789 is used.
- `plinevar()`: Option allows users to indicate a variable in the target data set which is to be used as the threshold for the Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) to be estimated from the simulated vectors. The user must have added the variable in the `sae data import` command when preparing the target dataset. Only one variable may be specified.
- `plines()`: The option allows users to explicitly indicate the threshold to be used, this option is preferred when the threshold is constant across all observations. Additionally, it is possible to specify multiple lines, separated by a space.
- **`appendsvy:`** option allows for full implementation of Molina and Rao's EB predictor, and ensures EB estimates utilize sampled observations and non-sampled observations. If option is not specified it will implement Census EB, see Corral et al. (2021).

Example 1 We first import the census data using the `sae data import` command. This data includes variables such as x1, x2, hhsize (household size), and pvar (poverty line). The census data is saved as a Mata file, which will be used later in the estimation process.

This part of the script generates small area estimates using the Empirical Best (EB) method following Molina and Rao (2010). The poverty lines for the 25th and 75th percentiles are calculated using the `summarize` command and stored as pline25 and pline75. The `sae sim reml` command is used to run simulations and obtain small area estimates for poverty indicators such as the headcount ratio (fgt0).

The implemented example estimates poverty indicators using the `sae sim reml` subcommand with the following specifications:

1. The dependent variable (`Y`) is assumed to be in log terms and is back-transformed before deriving poverty and other indicators, using the `lny` option.
2. The `uniqid()` option ensures replicability by assigning unique identifiers to households.
3. Simulations are performed at the household level, with population expansion factors specified for population-level estimates (`pwcensus()`).
4. Poverty lines are provided through local macros specified under the `pline()` option.
5. Estimates are derived from 100 Monte Carlo repetitions (`mcrep(100)`). Noise estimates (MSE) are obtained from 100 bootstrap replications (`bsrep(100)`).
6. The poverty measure estimated is the FGT0 (headcount poverty) indicator (`indicators(fgt0)`).
7. Results are generated at the same aggregation level as the `area()` variable (`aggids(0)`).
8. The census data used for estimation is stored in the Mata file `census_mata`, created with the `sae data import` command. The `appendsvy` option ensures that the Empirical Best (EB) estimates are consistent with those of Molina and Rao (2010).

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

.
. *================================================================================
. // Importing the census for small area estimates
. // Note that for the census we add the population expansion factors
. // to the varlist(). We´ve also added a variable containing the poverty line
. // "pvar".
. *================================================================================
. tempfile census_mata

.         sae data import, datain("`p´/sae/census.dta") varlist(x1 x2 hhsize pvar) ///
>         area(HID) uniqid(hhid) dataout(`census_mata´)
Saving data variables into mata matrix file (6)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
......
.
. *================================================================================
. //Obtain final small area estimates based on EB (Molina and Rao 2010)
. *================================================================================
. //Bring in the survey data
. use "`p´/sae/survey.dta", clear

.         //obtain 2 poverty lines, 25th and 75th percentile
.         sum E_Y [aw=hhweight], d

                            E_Y
-------------------------------------------------------------

      Percentiles      Smallest
 1%     10.05694       5.593755
 5%     12.33652       8.128839
10%     13.81096       8.526508       Obs               4,000
25%     16.57419       8.614201       Sum of wgt.  1.00000005

50%     20.21931                      Mean           21.14125
                        Largest       Std. dev.      6.499419
75%     24.65453       52.95024
90%     29.72983       59.15125       Variance       42.24245
95%     32.95321       61.11948       Skewness       1.043196
99%     41.65685       61.42175       Kurtosis        5.16971

.         local pline25 = r(p25)

.         local pline75 = r(p75)

.
.         //The following model specifications are used:
.         //1: Assumes that the Y variable is in log terms, we request to back //
.         //   transform before obtaining poverty and other indicators (lny)
.         //2: The uniqid() option is necessary to ensure replicability
.         //3: We indicate that the simulations are done at the household level
.         //   and indicate population expansion factors to obtain values for the
.         //   population
.         //4: We indicate that the poverty line to be used is in the locals
.         //   specified under plines()
.         //5: We´ve requested for the estimates to be obtained from 100 Monte
.         //       Carlo repetitions, mcrep()
.         //6: We´ve requested noise estimates (MSE) from 100 bootstrap replications,
.         //   bsrep()
```

```
.         //7: We´ve requested to estimate FGT0, headcount poverty, indicators(fgt0)
.         //8: We´ve requested for results to be produced at the same level as the
.         //   area() variable, aggids(0).
.         //9: We´ve indicated that the census data is in the mata file `census_mata´,
.         //   which was created using the sae data import command.
.         //10:We add the appendsvy option to ensure EB estimates similar to Molina
.         //   and Rao (2010)
.         sae sim reml Y x1 x2, area(HID)  lny uniqid(hhid) pwcensus(hhsize) ///
>         pline(`pline25´ `pline75´) mcrep(100) bsrep(100) indicators(fgt0) ///
>         aggids(0) matin(`census_mata´) appendsvy

Performing EM optimization:

Performing gradient-based optimization:

Iteration 0:  Log restricted-likelihood = -485.59116
Iteration 1:  Log restricted-likelihood = -485.59116

Computing standard errors:

Mixed-effects REML regression                   Number of obs     =  4,000
Group variable: HID                             Number of groups  =    400
                                                Obs per group:
                                                              min =     10
                                                              avg =   10.0
                                                              max =     10
                                                Wald chi2(2)      =  29.91
Log restricted-likelihood = -485.59116          Prob > chi2       = 0.0000
```

Y	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0378451	.0089501	4.23	0.000	.0203031	.055387
x2	-.0362165	.0106235	-3.41	0.001	-.0570381	-.0153949
_cons	2.989546	.0107495	278.11	0.000	2.968478	3.010615

Random-effects parameters	Estimate	Std. err.	[95% conf.	interval]
HID: Identity				
sd(_cons)	.1597743	.0070724	.146497	.174255
sd(Residual)	.2513664	.0029632	.2456252	.2572418

```
      LR test vs. linear model: chibar2(01) = 708.41        Prob >= chibar2 = 0.0000
      file C:\Users\WB378870\AppData\Local\Temp\ST_42f8_000004.tmp saved as .dta format
Number of observations in target dataset:

        20000

Number of clusters in target dataset:

        400

      Number of simulations: 100
      Each dot (.) represents 1 simulation(s).
      ----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
      ..................................................    50
      ..................................................   100

      Number of bootstraps: 100
```

```
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
.................................................. 50
.................................................. 100

.
. cap:  texdoc stlog close
```

Example 2 In this section, the model is modified to obtain CensusEB estimates, which are slightly different from the EB estimates. The key difference is that the `appendsvy` option is removed to produce CensusEB estimates which relies entirely on the census data for prediction of point and noise estimates. Additionally, the model applies a zero skewness log transformation (`lnskew`) to the dependent variable (`E_Y`) instead of the log transformation used previously. The poverty lines (`pline25` and `pline75`) are once again used for the simulations, and the census data is referenced via the `matin` option.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

.
. *================================================================================
. // Importing the census for small area estimates
. // Note that for the census we add the population expansion factors
. // to the varlist(). We´ve also added a variable containing the poverty line
. // "pvar".
. *================================================================================
. tempfile census_mata

.         sae data import, datain("`p´/sae/census.dta") varlist(x1 x2 hhsize pvar) ///
>         area(HID) uniqid(hhid) dataout(`census_mata´)
Saving data variables into mata matrix file (6)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
......
.
.
. *================================================================================
. //Obtain final small area estimates based on CensusEB
. *================================================================================
. //Bring in the survey data
. use "`p´/sae/survey.dta", clear

.         //obtain 2 poverty lines, 25th and 75th percentile
.         sum E_Y [aw=hhweight], d

                            E_Y
-------------------------------------------------------------

      Percentiles      Smallest
 1%     10.05694       5.593755
 5%     12.33652       8.128839
10%     13.81096       8.526508       Obs               4,000
25%     16.57419       8.614201       Sum of wgt.  1.00000005

50%     20.21931                      Mean           21.14125
                       Largest       Std. dev.      6.499419
75%     24.65453       52.95024
90%     29.72983       59.15125       Variance       42.24245
95%     32.95321       61.11948       Skewness       1.043196
99%     41.65685       61.42175       Kurtosis        5.16971
```

```
.         local pline25 = r(p25)

.         local pline75 = r(p75)

.
.         //The following model specifications are modified from before:
.         //1: We´ve removed the appendsvy option to obtain censusEB estimates.
.         //2. We´ve requested to apply the zero skewness log transformation
.         sae sim reml E_Y x1 x2, area(HID)  lnskew uniqid(hhid) pwcensus(hhsize) ///
>         pline(`pline25´ `pline75´) mcrep(100) bsrep(100) indicators(fgt0) ///
>         aggids(0) matin(`census_mata´)
```

Transform	k	[95% conf. interval]	Skewness
ln(E_Y-k)	.6876876	(not calculated)	4.02e-06

```
Performing EM optimization:

Performing gradient-based optimization:

Iteration 0:  Log restricted-likelihood = -630.18783
Iteration 1:  Log restricted-likelihood = -630.18783

Computing standard errors:

Mixed-effects REML regression                   Number of obs    =  4,000
Group variable: HID                             Number of groups =    400
                                                Obs per group:
                                                              min =     10
                                                              avg =   10.0
                                                              max =     10
                                                Wald chi2(2)     =  29.85
Log restricted-likelihood = -630.18783          Prob > chi2      = 0.0000
```

__000002	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
x1	.0392165	.0092798	4.23	0.000	.0210285	.0574044
x2	-.037493	.0110146	-3.40	0.001	-.0590813	-.0159047
_cons	2.952655	.0111468	264.89	0.000	2.930808	2.974503

Random-effects parameters	Estimate	Std. err.	[95% conf.	interval]
HID: Identity				
sd(_cons)	.1656958	.0073338	.1519276	.1807117
sd(Residual)	.2606216	.0030723	.254669	.2667133

```
LR test vs. linear model: chibar2(01) = 708.79        Prob >= chibar2 = 0.0000
file C:\Users\WB378870\AppData\Local\Temp\ST_42f8_000005.tmp saved as .dta format
```

Number of observations in target dataset:

```
  20000
```

Number of clusters in target dataset:

```
  400

Number of simulations: 100
Each dot (.) represents 1 simulation(s).
```

```
──┼── 1 ──┼── 2 ──┼── 3 ──┼── 4 ──┼── 5
..................................................    50
..................................................   100

Number of bootstraps: 100

──┼── 1 ──┼── 2 ──┼── 3 ──┼── 4 ──┼── 5
.................................................. 50
.................................................. 100

.
. cap:  texdoc stlog close
```

3.4.5 EB model-fitting under the `reml2` subcommand

The Stata `sae` package supports twofold nested error models featuring cluster effects nested within area effects, following the specification by Marhuenda et al. (2017). This model is particularly effective when estimates are needed at two different levels of aggregation. Model fitting is done via the `sae model reml2` subcommand which fits a Restricted Maximum Likelihood (REML) model based on the structure proposed by Marhuenda et al. (2017). The command calls Stata's `xtmixed` command in the background.

The command has a key difference with the previous ones presented since it requires users to specify the `subarea` option that gives an identifier for sub-areas. Unlike other commands presented here, the use of a hierarchical ID is necessary, ensuring identifiers for every location have an equal number of digits. Then, for the `area` option, users must indicate how many digits to trim from the rightmost part of the hierarchical identifier to obtain area-level identifiers. Detailed discussions of these hierarchical identifiers can be found in Nguyen et al. (2018) and Zhao (2006).

The command to fit the model is:

`sae model reml2` *`depvar indepvars`* `[if] [in],` *`options`*

Required

- `subarea():` The `area` option is necessary and specifies at which level the clustering is done, it indicates at which level the η_c is obtained at. The only constraint is that the variable must be numeric, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006).
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[19]
- `area():` Option is necessary and indicates the aggregation level for the larger area. Values placed here tell the command how many digits to the left to move the hierarchical id placed in the `subarea()` option to arrive at the larger ar example using the id from above, if the random effect for the larger area should correspond to areas, then here you should place a value of 4. Note that the structure of the id needs to be the same across Census and survey data.

[19] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

Example 1 The following code fits a model using the `reml2` subcommand, which leverages Stata's `xtmixed` function for modeling the assumed two-fold error structure. This step represents the first stage of modeling, where the basic two-fold model is applied to the survey data.

The `sae model reml2` command implements the two-fold estimation method to model the relationship between the dependent variable (`Y`) and independent variables (`x1`, `x2`). The options `subarea(HID)` and `area(2)` specify the hierarchical structure of the data, with `HID` identifying sub-areas and `area(2)` indicating the broader areas or regions for model application.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. *================================================================================
. // Model fitting under REML2
. // The model uses Stata´s xtmixed in the background
. *================================================================================
. //import data
. use "`p´/sae/survey.dta", clear

.         //fit model
.         sae model reml2 Y x1 x2, subarea(HID) area(2)

Performing EM optimization:

Performing gradient-based optimization:

Iteration 0:  Log restricted-likelihood = -323.82823
Iteration 1:  Log restricted-likelihood = -323.80238
Iteration 2:  Log restricted-likelihood = -323.80238

Computing standard errors:

Mixed-effects REML regression                           Number of obs =  4,000

        Grouping information
------------------------------------------------------------------------------
                |     No. of       Observations per group
 Group variable |     groups    Minimum    Average    Maximum
----------------+-------------------------------------------------------------
       __000003 |         40        100      100.0        100
       __000002 |        400         10       10.0         10
------------------------------------------------------------------------------

                                                        Wald chi2(2)  =  34.88
Log restricted-likelihood = -323.80238                  Prob > chi2   = 0.0000
------------------------------------------------------------------------------
           Y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
          x1 |   .0405492   .0087255     4.65   0.000     .0234476    .0576508
          x2 |  -.0372861   .0104355    -3.57   0.000    -.0577393    -.016833
       _cons |   2.988046   .0246479   121.23   0.000     2.939738    3.036355
------------------------------------------------------------------------------

------------------------------------------------------------------------------
  Random-effects parameters |   Estimate   Std. err.      [95% conf. interval]
----------------------------+-------------------------------------------------
__000003: Identity          |
                  sd(_cons) |   .1479327   .0175541      .1172353     .186668
------------------------------------------------------------------------------
```

```
__000002: Identity
                     sd(_cons)    .0644235   .0061654      .0534051     .077715
-----------------------------+------------------------------------------------
                  sd(Residual)    .2513615    .002963      .2456206    .2572365
------------------------------------------------------------------------------
LR test vs. linear model: chi2(2) = 1031.98                Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

.         //The use of REML does not allow for the inclusion of a model for
.         //heteroskedasticity.
. cap: texdoc stlog close
```

3.4.6 EB Small Area Estimates under the `reml2` subcommand

CensusEB small area estimation under the assumed twofold nested error model structure proposed by Marhuenda et al. (2017) is implemented using `sae sim reml2` . This approach generates CensusEB estimates through Monte Carlo simulation based on the twofold nested error model, the MSE is estimated following a parametric bootstrap aligned to the model's assumptions following González-Manteiga et al. (2008). The primary command for running simulations is:

`sae sim reml2` *depvar indepvars* `[if] [in],` *options*

Required

- `subarea(varname):` Option is necessary and indicates variable denoting level of area effect. The only constraint is that the variable must be numeric and should match across datasets (survey and census). The variable must follow a hierarchical structure similar to the one proposed by Zhao (2006). SSAAAMMMM, where SS indicate states, AA indicate areas, and MMMM indicate municipalities.
  - The hierarchical id should be of the same length for all observations for example: AAMMEEE.[20]
- `area():` Option is necessary and indicates the aggregation level for the larger area. Values placed here tell the command how many digits to the left to move the hierarchical id placed in the subarea() option to arrive at the larger ar example using the id from above, if the random effect for the larger area should correspond to areas, then here you should place a value of 4. Note that the structure of the id needs to be the same across Census and survey data.
- `uniqid():` option specifies the numeric variable that indicates the unique identifiers in the census and survey dataset. This is necessary to ensure replicability of the analysis, and the name should match the one of the unique identifier from the source dataset.
- `mcrep(integer):` Indicates the number of Montecarlo Simulations to be executed.
- `bsrep(integer):` Indicates the number of Bootstrap populations to be executed to obtain the MSE estimate. Note that every bootstrap population will execute the number of Montecarlo simulations indicated in mcrep().
- `matin():` The matin option indicates the path and filename of the Mata format target dataset. The dataset must be created with the `sae data import` command.

[20] In the case this were done for a specific country, AA stands for the highest aggregation level, MM stands for the second highest aggregation level, and EEE, stands for the lowest aggregation level.

- `aggids()`: The aggids option indicates the different aggregation levels for which the indicators are to be obtained, values placed here tell the command how many digits to the left to move to get the indicators at that level. Using the hierarchical id specified in the area option, AAMMEEE, if the user specifies 0, 3, 5, and 7, it would lead to aggregates at each of the levels E, M, A and the national level. Note that it is NOT advised to report results at a higher level than area(), it is likely that the MSE is underestimated (see Corral et al. 2021). Users should specify 0 if they are not using a hierarchical id.
- `pwcensus()`: Indicates the variable which corresponds to the expansion factors to be used in the target (census) dataset, it must always be specified. The user must have added the variable to the imported data (`sae data import`) i.e. the census data.
- `indicators()`: The `indicators` option is used to request the indicators to be estimated from the simulated vectors of welfare. The list of possible indicators is:
  - The set of Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) FGT_0, FGT_1, and FGT_2 ; also known as poverty head count, poverty gap, and poverty severity respectively.
  - The set of inequality indexes: Gini, and Generalized Entropy Index with $\alpha = 0, 1, 2$

Optional

- `pwsurvey(string)`: optional, and only necessary if users are using the appendsvy option.
- `lny`: The option indicates that the dependent variable in the welfare model is in log form. Specifying this indicates that the simulated vectors need to be transformed before applying the poverty line specified in `pline` or `plinevar`.
- `bcox`: option tells command to execute bcskew0 (Box-Cox transform - Zero skewness) on the dependent variable of your model.
- `lnskew`: option tells command to execute lnskew0 (log shift transform - Zero skewness) on the dependent variable of your model.
- `CONStant(real 0.0)`: option adds constant value to dependent variable when doing Box-Cox transformation. If bcox is not specified this is irrelevant.
- `seed()`: The option ensures replicability. Users should be aware that Stata's default pseudo-random number generator in Stata 14 is different than that of previous versions. If not specified, the default seed of 123456789 is used.
- `plinevar()`: Option allows users to indicate a variable in the target data set which is to be used as the threshold for the Foster Greer Thorbecke indexes (Foster, Greer, and Thorbecke 1984) to be estimated from the simulated vectors. Te user must have added the variable in the `sae data import` command when preparing the target dataset. Only one variable may be specified.
- `plines()`: The option allows users to explicitly indicate the threshold to be used, this option is preferred when the threshold is constant across all observations. Additionally, it is possible to specify multiple lines, separated by a space.

Example 1 The `sae data import` command imports a dataset (`census.dta`) containing additional information, such as variables related to household size (`hhsize`) and the poverty line (`pvar`). The `area(HID)` and `uniqid(hhid)` options are used to specify the area (`subarea` identifier HID) and the unique household identifier (`hhid`). The dataout option creates a temporary file (`census_mata`) to store the processed census data.

The command `sae sim reml2` performs CensusEB estimation under REML using a twofold nested error model. The process follows these steps:

1. Log-Transformation & Back-Transformation:
   (a) The dependent variable (`Y`) is assumed to be in log terms (`lny`).
   (b) Estimates are back-transformed before deriving poverty and other indicators.
2. Replicability:
   (a) The `uniqid()` option assigns unique household identifiers (`hhid`) to ensure reproducibility of results.
3. Simulation Level & Population Expansion:
   (a) Simulations are conducted at the household level, with population expansion factors specified via the `pwcensus()` option using `hhsize`. This ensures that results are scaled appropriately to represent the population.
4. Poverty Line Specification:
   (a) The poverty line is defined by the variable `pvar` within the census data, specified via the `plinevar()` option.
5. Monte Carlo Repetitions:
   (a) Estimates are obtained from 100 Monte Carlo repetitions (`mcrep(100)`)
6. Bootstrap MSE Calculation:
   (a) Noise estimates (Mean Squared Errors) are computed using 100 bootstrap replications (`bsrep(100)`).
7. Poverty Indicator Calculation:
   (a) The FGT0 (headcount poverty) indicator is estimated using the `indicators(fgt0)` option.
8. Aggregation Level Specification:
   (a) Results are produced at two levels of aggregation, as specified by:
      i. `subarea()`: Identifies sub-areas using the HID variable.
      ii. `area()`: Uses the value 2, indicating that two digits are removed from the rightmost part of the hierarchical identifier to define area-level identifiers.
      iii. `aggids(0 2)`: Ensures results are generated at both the subarea and area levels.
9. Census Data Source:

(a) The census data used for estimation is stored in the Mata file `census_mata`, created through the `sae data import` command and specified in the `sae sim reml2` command using `matin()`.

```
. //Determine personal folder
. local p : sysdir PERSONAL

. if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)

. *================================================================================
. // Importing the census for small area estimates
. // Note that for the census we add the population expansion factors
. // to the varlist(). We´ve also added a variable containing the poverty line
. // "pvar".
. *================================================================================
. tempfile census_mata

.         sae data import, datain("`p´/sae/census.dta") varlist(x1 x2 hhsize pvar) ///
>         area(HID) uniqid(hhid) dataout(`census_mata´)
Saving data variables into mata matrix file (6)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
......
.
. *================================================================================
. //Obtain final small area estimates based on twofold census EB
. //Results are to be obtained at the PSU and the domain level
. *================================================================================
. //Bring in the survey data
. use "`p´/sae/survey.dta", clear

.         //obtain 2 poverty lines, 25th and 75th percentile
.         sum E_Y [aw=hhweight], d

                            E_Y
-------------------------------------------------------------

      Percentiles      Smallest
 1%     10.05694       5.593755
 5%     12.33652       8.128839
10%     13.81096       8.526508       Obs               4,000
25%     16.57419       8.614201       Sum of wgt.  1.00000005

50%     20.21931                      Mean           21.14125
                        Largest       Std. dev.      6.499419
75%     24.65453       52.95024
90%     29.72983       59.15125       Variance       42.24245
95%     32.95321       61.11948       Skewness       1.043196
99%     41.65685       61.42175       Kurtosis        5.16971

.         local pline25 = r(p25)

.         local pline75 = r(p75)

.
.         //The following model specifications are used:
.         //1: Assumes that the Y variable is in log terms, we request to back //
.         //   transform before obtaining poverty and other indicators (lny)
.         //2: The uniqid() option is necessary to ensure replicability
.         //3: We indicate that the simulations are done at the household level
.         //   and indicate population expansion factors to obtain values for the
.         //   population
.         //4: We indicate that the poverty line to be used is in the variable
.         //   "pvar" within the census data, plinevar()
.         //5: We´ve requested for the estimates to be obtained from 100 Monte
```

```
.         //       Carlo repetitions, mcrep()
.         //6: We´ve requested noise estimates (MSE) from 100 bootstrap replications,
.         //   bsrep()
.         //7: We´ve requested to estimate FGT0, headcount poverty, indicators(fgt0)
.         //8: We´ve requested for results to be produced at the same level as noted
.         //   under area() and subarea() options, aggids(0 2).
.         //9: We´ve indicated that the census data is in the mata file `census_mata´,
.         //   which was created using the sae data import command.
.         sae sim reml2 Y x1 x2, subarea(HID) area(2)  lny uniqid(hhid) pwcensus(hhsize) ///
>         plinevar(pvar) mcrep(100) bsrep(100) indicators(fgt0) ///
>         aggids(0 2) matin(`census_mata´)

Performing EM optimization:

Performing gradient-based optimization:

Iteration 0:  Log restricted-likelihood = -323.82823
Iteration 1:  Log restricted-likelihood = -323.80238
Iteration 2:  Log restricted-likelihood = -323.80238

Computing standard errors:

Mixed-effects REML regression                           Number of obs =  4,000

        Grouping information
------------------------------------------------------------------------------
 Group variable |     No. of       Observations per group
                |     groups    Minimum    Average    Maximum
----------------+-------------------------------------------------------------
       __000003 |         40        100      100.0        100
       __000002 |        400         10       10.0         10
------------------------------------------------------------------------------

                                                Wald chi2(2)  =  34.88
Log restricted-likelihood = -323.80238          Prob > chi2   = 0.0000
------------------------------------------------------------------------------
           Y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
          x1 |   .0405492   .0087255     4.65   0.000     .0234476    .0576508
          x2 |  -.0372861   .0104355    -3.57   0.000    -.0577393    -.016833
       _cons |   2.988046   .0246479   121.23   0.000     2.939738    3.036355
------------------------------------------------------------------------------

------------------------------------------------------------------------------
  Random-effects parameters |   Estimate   Std. err.      [95% conf. interval]
----------------------------+-------------------------------------------------
__000003: Identity          |
                  sd(_cons) |   .1479327   .0175541      .1172353     .186668
----------------------------+-------------------------------------------------
__000002: Identity          |
                  sd(_cons) |   .0644235   .0061654      .0534051     .077715
----------------------------+-------------------------------------------------
               sd(Residual) |   .2513615    .002963      .2456206    .2572365
------------------------------------------------------------------------------

LR test vs. linear model: chi2(2) = 1031.98               Prob > chi2 = 0.0000

     Note: LR test is conservative and provided only for reference.
Number of observations in target dataset:

       20000

Number of subareas in target dataset:
```

```
  400
Number of areas in target dataset:

  40
...

Number of simulations: 100
Each dot (.) represents 1 simulation(s).
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
..................................................    50
..................................................   100

Number of bootstraps: 100

----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
.................................................. 50
.................................................. 100

. cap: texdoc stlog close
```

4 Appendix

4.1 Creating simulated data

```
/*
Annex, building simulated data for simulations

Do file below constructs data for a two fold nested error model. It follows the
approach illustrated in the paper from Molina and others in the link below.

We start off by creating a simulated data set as illustrated in that same paper.
 https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/rssa.12306
*/

set more off
clear all

//Determine personal folder
local p : sysdir PERSONAL
if (`"`c(os)´"´ == "MacOSX") local p = subinstr("`p´","\","/",.)
cap: mkdir "`p´/sae/"

*================================================================================
// Parameters for simulated data set
*================================================================================
        set seed 734137
        global numobs = 20000
        global outsample = 50
        global areasize  = 500
        global psusize   = 50

        //We have 2 location effects below
        global sigmaeta   = 0.15
        global sigmapsu  = 0.07
        //We have household specific errors
        global sigmaeps   = 0.25
        global model x1 x2
```

```
        local obsnum    = $numobs
        local areasize  = $areasize
        local psusize   = $psusize

*================================================================================
//Create fake dataset
*================================================================================
        set obs `obsnum´

        //Random location effects
        gen dom = (mod(_n,`areasize´)==1)*_n
        gen psu  = (mod(_n,`psusize´)==1)*_n

        gen u1 = rnormal(0,$sigmaeta) if dom!=0
        gen u2 = rnormal(0, $sigmapsu)  if psu!=0

        replace dom = dom[_n-1] if dom==0
        replace psu = psu[_n-1] if psu==0

        replace u1 = u1[_n-1] if u1==.
        replace u2 = u2[_n-1] if u2==.
        gen e      = rnormal(0,$sigmaeps)

        foreach i in dom psu{
                egen g2 = group(`i´)
                replace `i´ = g2
                drop g2
        }

        gen tdom=real(substr(string(psu),-1,1))
        replace tdom = 10 if tdom==0
        drop psu

        //Covariates
        gen x1 = runiform()<(0.2+0.4*dom/40+0.4*tdom/10)
        gen x2 = runiform()<0.2

        //Generate income
        gen Y = 3 + 0.03*x1 -0.04*x2 + u1 + u2 + e

        gen E_Y = exp(Y)

        //Get poverty line
        sum E_Y,d
        local povline = r(p25)

        //Hierarchical identifiers
        gen HID = 100*(100+dom) +tdom
        gen hhid = _n
        gen hhsize = 1

        gen double pvar = `povline´

        gen uno =1
        count
        preserve
                sort hhid
```

```
        sample 20, by(HID)

        //Household weights, every household has the same likelihood of being
        //sampled
        gen hhweight = 1/4000

        save "`p´/sae/survey.dta", replace
restore

save "`p´/sae/census.dta", replace
```

4.2 Using the data export and import utilities

4.2.1 Importing data

The `sae data import` command imports the sae data in a Stata file to one disk file in Mata matrix format. The file will be named based on the `dataout(string)`. All options are required for the command.

`sae data import, matasource() datain() varlist() area() uniqid() dataout()`

Required options

- `datain(string):` option indicates the path and file name of the Stata format dataset to be converted into a Mata format dataset.
- `varlist(string):` option specifies all the variables to be imported into the Mata format dataset. The variables specified will be available for the simulation stage of the analysis. Variables must have similar names between datasets. Additionally, users should include here any additional variables they wish to include, as well as expansion factors for the target data.
- `uniqid(string):` option specifies the numeric variable that indicates the unique identifiers in the target dataset. This is necessary to ensure replicability of the analysis, and the name should match the one of the unique identifier from the source dataset.
- `area(string):` option is necessary and specifies at which level the clustering is done, it indicates at which level the `eta_c` is obtained at. The only constraint is that the variable must be numeric and should match across datasets, although it is recommended it follows a hierarchical structure similar to the one proposed by Zhao (2006). The hierarchical id should be of the same length for all observations. For example: AAMMEEE. This structure facilitiates getting final estimates at different aggregation levels.
- `dataout(string):` option indicates the path and filename of the Mata format dataset that will be used when running the Monte Carlo simulations.

Example See example in section 2.1.2 ELL Small Area Estimates under the sae command.

4.2.2 Exporting data

The `sae data export` command is useful to bring in the simulated vectors created by `sae sim lmm` into Stata. Note that this function requires the user to specify the creation of a `ydump` file. The `ydump` file is only available under `sae sim lmm` and the `sae sim ell` functions.

The function takes the `ydump` and imports it into Stata data in memory and gives the ability to further manipulate these vectors. Due to the unique structure of the Mata data, this command will only work with data created in the simulation stage.

Due to the unique structure of the Mata data, this command will only work with data created in the simulation stage

`sae data export, matasource()` *options*

Options

- `matasource(string):` The matasource option allows users to specify the source ydump file created by the sae simulate routine. Because the size of the file can be quite large, it is advisable to use this with the numfiles option.
- `numfiles(integer 1):` The numfiles option is to be used in conjunction with the ydumpdta option; it specifies the number of datasets to be created from the simulations.
- `prefix(string):` The prefix option may be used to give a prefix to the simulated vectors.
- `saveold:` The saveold option can be specified in conjunction with the ydumpdta option, and makes the files readable by older versions of Stata.
- `datasave(string):` The datasave option allows users to specify a path where to save the exported data, this is recommended when using the numfiles option.

Example See example in section 2.1.2 ELL Small Area Estimates under the sae command.

References

Bedi, T., A. Coudouel, and K. Simler (2007). *More than a pretty picture: using poverty maps to design better policies and interventions*. World Bank Publications.

Corral, P., K. Himelein, K. McGee, and I. Molina (2021). A map of the poor or a poor map? *Mathematics 9*(21), 2780.

Corral, P., I. Molina, A. Cojocaru, and S. Segovia (2022). *Guidelines to small area estimation for poverty mapping*. Washington, DC: The World Bank.

Corral, P., I. Molina, and M. Nguyen (2021). Pull your small area estimates up by the bootstraps. *Journal of Statistical Computation and Simulation 91*(16), 3304–3357.

Demombynes, G. (2002). A manual for the poverty and inequality mapper module. *University of California, Berkeley and Development Research Group, the World Bank*.

Demombynes, G., C. Elbers, J. Lanjouw, P. Lanjouw, J. Mistiaen, and B. Ozler (2002). Producing an Improved Geographic Profile of Poverty: Methodology and Evidence from Three Developing Countries. WIDER Working Paper Series 039, World Institute for Development Economic Research (UNU-WIDER).

Demombynes, G., C. Elbers, J. O. Lanjouw, and P. Lanjouw (2008). How good is a map? putting small area estimation to the test. *Rivista Internazionale di Scienze Sociali*, 465–494.

Elbers, C., J. O. Lanjouw, and P. Lanjouw (2002). Micro-level estimation of welfare. *2911*.

Elbers, C., J. O. Lanjouw, and P. Lanjouw (2003). Micro–level estimation of poverty and inequality. *Econometrica 71*(1), 355–364.

Foster, J., J. Greer, and E. Thorbecke (1984). A class of decomposable poverty measures. *Econometrica: Journal of the Econometric Society*, 761–766.

Gelman, A., J. Carlin, H. Stern, D. Dunson, A. Vehtari, and D. Rubin (2004). *Bayesian Data Analysis.*, Volume 2nd ed of *Texts in Statistical Science*. CRC Press [CAM].

González-Manteiga, W., M. J. Lombardía, I. Molina, D. Morales, and L. Santamaría (2008). Bootstrap mean squared error of a small-area eblup. *Journal of Statistical Computation and Simulation 78*(5), 443–462.

Haslett, S., M. Isidro, and G. Jones (2010). Comparison of survey regression techniques in the context of small area estimation of poverty. *Survey Methodology 36*(2), 157–170.

Henderson, C. R. (1953). Estimation of variance and covariance components. *Biometrics 9*(2), 226–252.

Huang, R. and M. Hidiroglou (2003). Design consistent estimators for a mixed linear model on survey data. *Proceedings of the Survey Research Methods Section, American Statistical Association (2003)*, 1897–1904.

Marhuenda, Y., I. Molina, D. Morales, and J. Rao (2017). Poverty mapping in small areas under a twofold nested error regression model. *Journal of the Royal Statistical Society: Series A (Statistics in Society) 180*(4), 1111–1136.

Molina, I. (2019). Desagregación de datos en encuestas de hogares: metodologías de estimación en áreas pequeñas.

Molina, I. and Y. Marhuenda (2015). R package sae: Methodology. *The R Journal 7*(1), 81–98.

Molina, I. and J. Rao (2010). Small area estimation of poverty indicators. *Canadian Journal of Statistics 38*(3), 369–385.

Nguyen, M. C., P. Corral, J. P. Azevedo, and Q. Zhao (2018). sae: A stata package for unit level small area estimation. *World Bank Policy Research Working Paper* (8630).

Rao, J. and I. Molina (2015). *Small area estimation* (2nd ed.). Hoboken, NJ: John Wiley & Sons.

Rubin, D. B. (1996). Multiple imputation after 18+ years. *Journal of the American statistical Association 91*(434), 473–489.

Rubin, D. B. (2004). *Multiple imputation for nonresponse in surveys*, Volume 81. John Wiley & Sons.

Van der Weide, R. (2014). Gls estimation and empirical bayes prediction for linear mixed models with heteroskedasticity and sampling weights: a background study for the povmap project. *World Bank Policy Research Working Paper* (7028).

Zhao, Q. (2006). User manual for povmap. *World Bank.http://siteresources.worldbank.org/INTPGI/Resources/342674-1092157888460/Zhao_ManualPovMap.pdf*.