LIS Self Teaching Package 2022

SPSS version

Part I

Inequality, poverty, and social policy

Part I: Inequality, poverty, and social policy

Overall Plan and Structure of the Exercise

The next eight exercises demonstrate the use of the LIS data. These exercises will lead you through the process of developing a comparative research project that examines inequality and poverty across countries. Each of the exercises introduces new concepts related to the datasets and the programming techniques needed to make use of the data. By the end of the last exercise, you will produce a complete program that returns results on poverty and inequality for a selection of five LIS countries.

Each exercise builds on the one that comes before it. It is intended that you will begin every new activity by returning to the program you have written in the previous exercise, and modifying it to satisfy the requirements of the new exercise. Each exercise contains questions that you can answer with the new results you produce. The solutions[1] available for each exercise include an example program, with **bolded** sections indicating the code that has been added for that exercise.

The analysis shown here is simplified somewhat, compared to what might be done in an actual LIS Working Paper. Some choices have also been made in order to demonstrate particular aspects of the LIS data. However, these exercises provide a starting point for researchers who want to develop an analysis of the data based on their own research questions.

Research Questions

Since the beginning of the LIS project, one of the most prominent objects of research using the data has been the effect of government tax and transfer programs on poverty and inequality. The first substantive paper in the LIS Working Papers series, published in 1985, analysed the pre- and post-transfer poverty rate in Sweden, the United Kingdom, Israel, the United States, Norway, Canada, and West Germany.[2] The second substantive paper compares the

[1] Please note that results calculated by you might differ from the ones presented here given that the LIS data is subject to updates from time to time.

[2] Richard Hauser, Lee Rainwater, Martin Rein, Gaston Schaber, Timothy Smeeding. "Poverty in Major Industrialized Countries". LIS Working Paper No. 2 – Jul 1985. Available at: http://www.lisdatacenter.org/wps/liswps/2.pdf

distribution of income across these same seven countries.[3]

States affect the income distribution through several different types of policy, which are captured in the LIS data:

- Income taxes and social insurance contributions
- Social insurance programmes linked to employment, such as public pensions, unemployment insurance benefits, and sickness pay
- Universal benefits provided irrespective of employment, income or assets
- Social assistance benefits for especially needy individuals or households

The exercises in this section assess the impact of these policies in different countries on both poverty and income inequality. We will measure income from labour, capital, private pensions, and private transfers. Then, we will compare them to income after income taxes and social insurance contributions as well as government transfers are accounted for. Within the category of government policies, we will separate the effect of payroll taxes, social insurance, and universal benefits from the effect of social assistance benefits. We will measure the proportion of the population that is poor according to these different income measures, in which poverty is defined relative to median level of income within a country. We will also compare income inequality using one of the most popular and longstanding inequality measures, the Gini coefficient.

As LIS has grown, the analysis of government policy, poverty, and inequality has been updated for more countries and more recent years.[4] Recently, LIS expanded beyond the rich countries that have long made up the core of the project, and began adding data from middle income countries. This gives researchers the opportunity to compare income and poverty in these countries to the patterns seen in rich countries that have been much more heavily studied.

For this exercise, we will begin by analysing the data from Guatemala. We will then compare Guatemala to four other countries: The United States, Denmark, Hungary, and Israel. These countries have been chosen in part for pedagogical reasons. As well as, they allow for substantively interesting comparisons as they

3 Michael O'Higgins, Gunther Schmaus, Geoffrey Stephenson. "Income Distribution and Redistribution". LIS Working Paper No. 3 – Jun 1985. Available at: http://www.lisdatacenter.org/wps/liswps/3.pdf

4 For a recent example, see Timothy Smeeding, "Government Programs and Social Outcomes: The United States in Comparative Perspective". LIS Working Paper No. 426 – May 2005. Available at: http://www.lisdatacenter.org/wps/liswps/426.pdf

represent a wide range of national income levels and welfare state regime types.

After completing the final exercise, you will be able to answer the following questions:

- Which of the five countries in the study have the highest levels of income inequality and poverty before taxes and transfers are accounted for? Does this change when taxes and transfers are included?
- Which type of government policy has a larger impact on inequality and poverty in each country: taxes, social insurance, and universal benefits, or targeted social assistance?

Before you begin

Before you begin the exercises, take a look at the 2019 Template LIS User Guide, which can be accessed through *LIS Website* → *Our Data* → *LIS Database*. The User Guide provides an overview of the structure of LIS data and some data management practices, such as missing values policy and aggregation rules, which will be useful in working with LISSY.

In addition to this, an overview of the datasets and variables is provided through the METadata Information System (METIS) without having to login to LISSY. You can access METIS via the *LIS Website* → *METIS* → *Enter METIS* → *LIS*. After selecting the datasets and variables, consult the *Results tab* for information on variables and definitions, dataset-specific information and variable availability across datasets.

Contents

1. Accessing the LIS Database: LISSY Interface

Goal

This exercise introduces LISSY Interface, which we will be using to work with LIS data in all of the subsequent exercises.

LISSY secure web-based interface allows researchers to:

- write, submit and view job requests (and corresponding outputs);
- track the status of the job requests in process ('received', 'processing', 'set for review', 'refused', etc.); and
- access the history of all job requests ever sent.

In this exercise we will use LISSY interface to open a dataset and produce basic descriptive statistics.

Activity

Go to *LIS website* → *Login LISSY* tab with your LISSY account. Submit a simple program to display descriptive statistics (number of valid observations, mean, minimum, maximum) for the household-level income variable **dhi**, for Guatemala 2006. **dhi**, or disposable household income, contains the total monetary and non-monetary current income for the household, net of income taxes and social security contributions. It is a harmonised variable that is available for all datasets.

Questions

1.1. How many valid observations (non-missing) are in this dataset for **dhi**?

Guidelines

- Once connected to LISSY Interface, there are three main tasks that may be carried out:
  1. Submit jobs through the Job Session window>>edit job pane.
     - Select a project (LIS, LWS).
     - Select a statistical package (SAS, **SPSS**, Stata or R).
     - When submitting a job (edit job pane), always add a subject line.
     - Write your code.
     - Click on the submit button.
  2. Work with Today's Jobs (Recent Jobs window.)
     - Watch the status of jobs currently sent to LISSY in the 'recent jobs'

pane.

- View the jobs returned by LISSY.
- Click on the received job (marked with green arrow).
- Click on the 'job text' or 'listing' tabs, respectively, of the right panel to see the request and its output.
- Download the received results and the job code by either:
  - clicking on 'save all results to pdf file' button in the middle of the upper tool bar to save the results in pdf format.
  - Clicking on 'save all results to a zipped file' button to save the results in txt format or png format for the produced graphs.
- Re-submit a selected job by clicking on the 'upload all or selected text to the actual job editor' button at the left hand side of the upper tool bar.

3. Manage (view, clean and search) all job requests ever sent in the Job Archive window.
   - View jobs sent over a specific time period.
   - Clean the library by discarding useless job requests (remove job from active archive button).
   - Search jobs by keywords.
   - Re-submit a selected job by clicking on the 'upload all or selected text to the actual job editor' button at the left hand side of the upper tool bar.

- When you open a LIS dataset, use the correct file reference for the country/year you wish to use. For example:

  get file = gt06h

  For more information about the syntax of country/year file reference, see the job submission instructions on the LIS web site (*Data Access → Job Submission*), accessed through this link.

  For a list of available data sets and their 2digit country codes, go to:

  Our Data → LIS Database → List of Datasets, accessible through LIS

  Our Data → LWS Database → List of Datasets, accessible through LWS

- SPSS reminder: to run descriptive statistics, use **descriptive variables = <varlist>**

- If you do not receive your job in the expected amount of time, it means that there is a long queue of jobs on LISSY. In that case, resending your job, or

sending several other ones while waiting to receive the first one, will only increase the queue and hence your waiting time. Remember to wait to get your results before sending a new job!

- Please keep in mind that SPSS is sensitive to special characters. If you experience issues when copy-pasting the code from the self-teaching packages, you can first paste it to a Notepad file and then to LISSY interface.

Program

get file = gt06h.

descriptive variables = dhi.

Results

	Number of valid observations	Mean	Minimum	Maximum
dhi	13,664	40,670	-186,010	2,729,298

Solution

1.1. How many valid observations (non-missing) are in this dataset for **dhi**?

- There are 13,664 valid observations.

Comments

- It is important to pay close attention to sample sizes in order to make sure you have enough data to make stable estimates. When working with small datasets, or small sub-samples of datasets, always check the size of the sample underlying each statistic you have computed.
- As you can see in the results, the disposable household income **dhi** can be negative. This happens in cases where the data provider kept losses as such rather than applying bottom coding techniques. This typically happens with incomes from self-employment or capital income; in rare circumstances it happens that taxes are higher than gross income due to different income reference periods or miscalculation of taxes.

2. Sample selection and weighting

Goal

This exercise introduces the variables that we will be using in the rest of our analysis and concentrates on sample selection and the use of weights.

Sample selection – The final objective of this exercise is to compare incomes before and after government intervention. In order to be sure that the comparison is correct, users should ensure that they use exactly the same sample when calculating statistics for the pre- and post-government intervention. It is thus important to begin by selecting a sample that can be used for the entire analysis. For this reason, we will drop from the analysis not only cases which have missing values in the variable of interest to the specific statistic being calculated at each step, but all the cases that have a missing value in any of the variables of interest for the whole analysis.

Use of weights - Comparative researchers are typically interested in the characteristics of national populations, not the samples provided. It is very important to understand and use sample weights correctly in order to get representative results for the total underlying population. This exercise shows the differences in statistics between the unweighted sample and the weighted population.

Activity

As in the previous exercise, we will continue working with the Guatemala 2006 (GT06) data. Modify your previous program to select only the following variables in addition to **dhi**: household weight (**hpopwgt**), number of household members (**nhhmem**), gross or net income information (**grossnet**), factor income (**hifactor**), work-related insurance transfers (**hpub_i**), universal benefits (**hpub_u**), social assistance benefits (**hpub_a**), private transfers (**hiprivate**) and tax and social security contribution expenditures (**hxitsc**).

(1) Create an indicator variable (e.g. miss_comp) that is equal to 1 if **dhi** or any of its income and expenditure sub-components is missing, and equal to zero otherwise.

(2) Produce a frequency table for this variable to see how many cases have missing data.

(3) Produce two different sets of descriptive statistics for the variables you have selected: unweighted, and weighted by person. For continuous variables, the statistics should include the number of observations, the mean, the median, the minimum and the maximum. For categorical variables, you should produce a frequency table.

(4) Drop all cases for which the indicator variable is equal to 1 and produce again the set of weighted descriptive statistics for all variables selected.

Questions

2.1. What currency unit is used for the income variables in this dataset?

2.2. What effect does applying the weights have on the median of disposable household income?

2.3. What percentage of cases is being dropped from the dataset?

2.4. In subsequent exercises, we will compare incomes before and after accounting for different kinds of government transfers. What is the difference, in terms of definition, between work-related insurance transfers, universal benefits, and assistance benefits?

2.5. In subsequent exercises, we will separate social assistance transfers from work-related insurance and from universal transfers. Based on the information you have so far, which type of transfer do you think has a larger effect on inequality and poverty measures in Guatemala?

2.6. How many different values does the variable **grossnet** take in this dataset? How might this variable be useful?

Guidelines

- To keep only the variables you will be using, use the **keep** options within the file command:

  /keep = <varlist>

  This avoids unnecessary burden on the machine so that submitted jobs will run faster. Combine the option and the command:

  get file = gt06h / keep = <varlist>

- To run summary statistics use the **descriptives** command:

  descriptives variables = <varlist>

- The descriptive command does not produce the median. A simple way to get the median is by using the **frequencies** command with the **/statistics=median** option. **Be aware** that when applying this to a continuous variable, you will get the full frequency table which will produce huge output. To avoid this, you need to add a further option **/format=notable**.

  frequencies variables = <varlist>

- LIS Weights:
  - LIS records the household level weights in the variable **hpopwgt.** The person-level file contains a person-level weight variable, **ppopwgt.** When we are using the household file but want to weight by person rather than household, we can multiply the household weight by the number of household members, contained in the variable **nhhmem**.

- Both of these are *inflated* weights. This means that for this dataset, the weight inflates to the total population in Guatemala in 2006. You can find the population size by looking at the weighted number of observations.

- To get weighted descriptive statistics, you need to tell SPSS to weight by the variable that contains the weighting factor **weight by <varname>**. The variable name must be an existing one, or should have been created in advance. Your final command should look something like this:

  weight by <varname> .

  or in case of a composite weight:

  compute populat = hpopwgt * nhhmem.

  weight by populat.

Program

```
get file = gt06h /keep=did dhi hifactor hpub_i hpub_u hpub_a hiprivate hxitsc
hpopwgt nhhmem grossnet.

compute miss_comp = 0.

if (missing(dhi) or missing(hifactor) or missing(hpub_i) or missing(hpub_u) or
missing(hpub_a) or missing(hiprivate) or missing(hxitsc)) miss_comp = 1.

frequencies variables =  miss_comp.

descriptives variables = dhi hifactor hpub_i hpub_u hpub_a hiprivate hxitsc
hpopwgt.

frequencies variables = dhi /statistics=median /format=notable.

frequencies variables = grossnet.

compute populat = hpopwgt * nhhmem.

weight by populat.

descriptives variables = dhi hifactor hpub_i hpub_a hiprivate hxitsc hpopwgt.

frequencies variables = dhi /statistics=median /format=notable.

select if miss_comp eq 0.

descriptives variables = dhi hifactor hpub_i hpub_a hiprivate hxitsc hpopwgt.

frequencies variables = dhi /statistics=median /format=notable.
```

Results

miss_comp	number of observations	percent	cumulative percent
0	13,664	99.84	99.84
1	22	0.16	100
Total	13,686	100	

Unweighted, all cases

	number of observations	mean	median	minimum	maximum
hpopwgt	13,686	193.8477	119	2	2,657
dhi	13,664	40670.24	27,092	-186,010	2,729,298
hifactor	13,664	39,924.08	24,777.50	0	3,866,644
hpub_i	13,664	803.47	0	0	134,400
hpub_u	13,686	0	0	0	0
hpub_a	13,686	695.49	0	0	62,400
hiprivate	13,664	1,438.65	0	0	128,020
hxitsc	13,664	2,191.45	0	0	1,137,346

grossnet	number of observations	percent	cumulative percent
[120]gross, taxes and contributions imputed	13,686	100	100
Total	13,686	100	

Weighted, all cases

	number of (weighted) observations	mean	median	minimum	maximum
hpopwgt	12,964,197				
dhi	12,938,575	49,556.30	33,658	-186,010	2,729,298
hifactor	12,938,575	49,298.82	30,819	0	3,866,644
hpub_i	12,938,575	777.41	0	0	134,400
hpub_u	12,964,197	0	0	0	0
hpub_a	12,964,197	755.96	0	0	62,400
hiprivate	12,938,575	1,675.62	0	0	128,020
hxitsc	12,938,575	2,952.58	0	0	1,137,346

Weighted, missing income cases dropped

	number of (weighted) observations	mean	median	minimum	maximum
hpopwgt	12,938,575				
dhi	12,938,575	49,556.3040	33,658	-186,010	2,729,298
hifactor	12,938,575	49,298.8200	30,819	0	3,866,644
hpub_i	12,938,575	777.4066	0	0	134,400
hpub_u	12,938,575	0	0	0	0
hpub_a	12,938,575	757.0184	0	0	62,400
hiprivate	12,938,575	1,675.6224	0	0	128,020
hxitsc	12,938,575	2,952.58	0	0	1,137,346

Solution

2.1. What currency unit is used for the income variables in this dataset?

- The currency unit in this dataset is the Guatemalan Quetzal. This information can be found on the LIS METIS documentation system at *Our Data → METIS,* accessed through this *link*. One way to find currency information in METIS is by *clicking on enter Metis → selecting LIS → selecting a country → Results → Dataset information → [LIS] Dataset*.
- Each LIS dataset has a variable named "currency". This variable indicates the currency unit used for the income/expenditure variables. This variable is available in both the household and the person data files.

```
get file = gt06h
frequencies variables = currency.
```

currency units	number of observations	Percent	Cum.
[320]GTQ - Quetzal	13,664	100	100
Total	13,664	100	

2.2. What effect does applying the weights have on the median of disposable household income?

- The median of unweighted **dhi** is 27,092 Quetzals, while the median of weighted **dhi** is 33,658 Quetzals, suggesting that low-income households are over-represented in the sample.

2.3. What percentage of cases is being dropped from the dataset?

- There are 22 cases with missing data, or 0.16 percent of the total number of cases in the dataset. Note that you should always be careful if you see a large amount of missing data, as it could bias your estimates.

2.4. In subsequent exercises, we will be comparing incomes before and after accounting for different kinds of government transfers. What is the difference between work-related insurance transfers, universal benefits, and assistance benefits?

- These concepts are defined in the **Interactive METadata Information System (METIS)**.

  LIS Database → Select variables → Major economic aggregates → hpub_i/hpub_u/hpub_a → click on the information icon ⓘ → a pop-up menu will appear displaying all the information related to these variables (definition and comments):

  - Work-related insurance transfers: Monetary transfers stemming from systems where the eligibility is based on the existence and/or the length of an employment relationship; in most cases the benefits are financed by contributions paid by employers, workers or both, and their amount is usually dependent on either the previous earnings or the previous contributions (not including occupational and voluntary pensions);
  - Universal transfers: Monetary transfers stemming from public programmes that provide flat-rate benefits to certain residents or citizens, provided that they are in a certain situation, but without consideration of income, employment or assets; note that in some cases the benefit amount may also depend on the other incomes of the individuals, which at the limit may result on some proportion of the population at the upper end of the income distribution to be excluded from receipt;
  - Assistance transfers: Monetary and non-monetary transfers stemming from public programmes that provide benefits especially targeted to needy individuals or households (i.e. with a strict income or assets test); the amount of the benefits is either flat rate or based on the difference between the recipient income and a standard amount representing the minimum subsistence needs as guaranteed by the government.

2.5. In subsequent exercises, we will separate social assistance transfers from work-related insurance and from universal transfers. Based on the information you have so far, which type of transfer do you think has a larger effect on inequality and poverty in Guatemala?

- Since universal transfers are not available at the level of detail required for Guatemala's dataset. Therefore, inequality and poverty will only be impacted by social assistance transfers and work-related insurance.

2.6. How many different values does the variable **grossnet** take in this dataset? How might this variable be useful?

- The variable **grossnet** has the same value for every case ([120] gross, taxes and contributions imputed). By looking at this variable, you can learn what type of gross or net income information is contained in this variable, even without looking at the documentation. This variable will also be useful when working with multiple datasets that may contain either gross or net income. This information can be found in METIS under the *Results* → *Dataset information* → *Code Books* tab.

3. Working with household income variables: top and bottom coding and equivalence scales

Goal

In order to compare incomes across countries, we need to make sure that our variables are fully comparable. In this exercise, you will apply top- and bottom-codes to remove extreme values. You will then create an *equivalised* income variable that adjusts for household size.

Top and bottom coding - Many inequality measures are sensitive to the values at the bottom and/or top of the income distribution, and some are not defined for non-positive values of income (e.g., any measure that calculates a logarithm). Applying top and bottom-codes (often referred to as 'winsorising') will avoid this problem, as well as ensuring comparability between datasets that may have originally had different top- and bottom-coding. At LIS, we use the interquartile range (IQR) method to estimate income measures for the LIS Key figures and Data Access Research Tool (DART). The method detects extreme values and applies lower and upper boundaries as bottom and top code, smoothing inequality trends within and between countries by consistently reducing the influence of extreme values in the income distributions for inequality measures. The first step required is to define a new log transformed variable disposable household income, the second, is to calculate the log values for the interquartile range. Finally, we use the exponential of the log values in the original income distribution before the log transformation: EXP [log Q1 - 3*(logQ3 –logQ1)] for the lower boundary and EXP [log Q3 + 3*(logQ3 –logQ1)] for the upper boundary.[5]

Equivalence scales - In order to get measures of poverty and/or income inequality in a population, it is necessary to compare income across different types of households. It is not logical to compare total household income between households of different sizes and composition.

Suppose you observe three levels of income (A, B, and C), where A>B>C. You cannot state that a household earning A is better off than one earning B unless you know the two households are similar in composition. For example, a family of four adult members receiving A is not necessarily better off than a couple with two children who receive B, and the family receiving B may not be better off than the childless couple receiving C.

For this reason, total household income needs to be adjusted to make it comparable across different households. This exercise gives one example of "equalising" households using one specific equivalence scale.

[5] See LIS Technical Working Paper No. 9, accessible here.

Activity

(1) Keep the code from your previous exercise that you used to drop cases with missing data.

(2) Create a new variable, **dhi_tb**, a top- and bottom-coded version of **dhi** using the interquartile range technique.

(3) Create another new variable, **edhi_tb**, an equivalised version of top- and bottom-coded disposable household income. We will correct for household size by applying the "LIS equivalence scale" (i.e., the square root of the number of household members).

(4) Create a measure of per-capita income, **pcdhi_tb**, by dividing household income (un-equivalised, but top- and bottom-coded) by the number of household members.

(5) Produce summary statistics showing the mean, median, minimum, and maximum of the four income variables: **dhi**, **dhi_tb**, **pcdhi_tb**, and **edhi_tb**.

Using the results returned from LISSY, fill out the following table:

	Household income (no top or bottom codes)	Household income	Per capita income	Equivalised income
Mean				
Median				
Minimum				
Maximum				

Questions

3.1. Which of these four versions of the income variable contain negative values?

3.2. Relative to household income and per capita income, how large are the mean and median of equivalised income?

3.3. How does applying the top and bottom code affect the mean and median of household income?

Guidelines

- As we continue to build up to our final program, some of the code from the previous exercises will no longer be necessary. (For example, the code that produced the summary statistics in the previous exercise). You can choose to delete this code from your program in order to make it shorter. However, if you would like to keep a line code but stop it from being executed, simply place a * before it.
- This exercise does not use all of the variables that were used in the previous section. But, you should continue to keep all of those variables when you open the dataset as they will be needed in future exercises.
- To equivalise income, divide the total household income by the value of the equivalence scale for each observation. To create LIS equivalised income:

  compute edhi_tb = dhi_tb /(nhhmem0.5)**

- Be careful when using weights. Make sure that the weight matches your unit of analysis. Weigh by **hpopwgt** for variables which are intrinsically at the household level (e.g., **dhi**) and by **hpopwgt*nhhmem** (to account for household size) for variables that are conceptually meaningful at the person level (e.g., per capita and equivalised income).

Program

define dataprepare ().

SET ERRORS OFF.

compute miss_comp = 0.

if (missing(dhi) or missing(hifactor) or missing(hpub_i) or missing(hpub_u) or missing(hpub_a) or missing(hiprivate) or missing(hxitsc)) miss_comp = 1.

select if miss_comp eq 0.

*** select only records if dhi filled.**

select if not missing(dhi) .

*** create top and bottom coded household disposable income.**

compute dhi_tb = dhi.

*** recode negaive dhi into zero**

if (dhi_tb<0) dhi_tb=0.

EXECUTE.

compute dhi_log = ln(dhi_tb).

EXECUTE.

if (missing(dhi_log) & NOT(missing(dhi_tb))) dhi_log=0.

EXECUTE.

*** create person weight as hwgt times number of household members.**

compute wt = hpopwgt*nhhmem .

weight by wt.

!enddefine .

define decilecalc ().

preserve .

set tvars names tnumbers values.

dataset declare decileratio.

WEIGHT BY hpopwgt.

sort cases by did.

split file by did.

```
OMS
 / select tables
 / if command = ['Frequencies'] subtypes=['Statistics']
 /destination format = sav       outfile = 'decileratio'
 /columns sequence = [l1 r2] .
frequencies variables = dhi_log
  /percentiles = 25 50 75
  /format = notable .
OMSEND.
weights off.
restore.
match files file = *
  /table = 'decileratio'
  /rename (var1 = did)
  /by did .
!enddefine .
define topbottom ().
weight by wt.
COMPUTE iqr=dhi_log_75-dhi_log_25.
EXECUTE.
* detect upper bound for extreme values
COMPUTE upper_bound=dhi_log_75 + (iqr * 3).
EXECUTE.
COMPUTE lower_bound=dhi_log_25 - (iqr * 3).
EXECUTE.
* top code income at upper bound for extreme values
if dhi_tb>exp(upper_bound) dhi_tb=exp(upper_bound).
EXECUTE.
```

*** bottom code income at lower bound for extreme values**

if dhi_tb<exp(lower_bound) dhi_tb=exp(lower_bound).

EXECUTE.

*** create equivalised income, set equivalence scale as square root of household members**

compute edhi_tb = dhi_tb/(nhhmem0.5).**

compute pcdhi_tb = dhi_tb/nhhmem.

!enddefine .

get file = gt06h /keep=did dhi hifactor hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem grossnet.

*** run the dataprepare, decilecalc and topbottom**

dataprepare.

decilecalc.

topbottom.

weight by hpopwgt.

descriptives variables = dhi dhi_tb.

frequencies variables = dhi dhi_tb /statistics=median /format=notable.

compute populat = hpopwgt * nhhmem.

weight by populat.

descriptives variables = pcdhi_tb edhi_tb.

frequencies variables = pcdhi_tb edhi_tb /statistics=median /format=notable.

Results

	Household income (no top or bottom codes) **(dhi)**	Household income (top or bottom coded) **(dhi_tb)**	Per capita income (top or bottom coded) **(pcdhi_tb)**	Equivalised income **(edhi_tb)**
Mean	47,681	47,750	9,768	21,407
Median	31,096	31,096	5,766	14,007
Minimum	-186,010	358	60	146
Maximum	2,729,298	2,541,775	691,527	1,270,888

Solution

3.1. Which of these four versions of the income variable contain negative values?

- Only the income variable without top or bottom codes contains negative values, which are removed by applying a bottom-code. This will be important in the next exercise on inequality. The measure of inequality we will use, the Gini coefficient, does not allow negative values. Removing negative values also allows for the commonly used logarithmic transformation of income.

3.2. Relative to household income and per capita income, how large are the mean and median of equivalised income?

- The mean and median values for equivalised income fall between those for household income and those for per capita income. The equivalising formula of **dhi**/(**nhhmem**^0.5) is a compromise between assigning all individuals their household income (**dhi**/**nhhmem**^0) and assigning them a per capita income (**dhi**/**nhhmem**^1).

3.3. How does applying the top and bottom code affect the mean and median of household income?

- Applying top and bottom codes makes the mean lower but does not affect the median (31,096). Means can be very sensitive to extreme values, so median values are often preferred as a measure of central tendency.

4. Inequality: The Gini Index

Goal

This exercise introduces the Gini index, which is one of the most commonly used income inequality indicators. We will be using the Gini coefficient as our measure of inequality in subsequent exercises, in order to compare inequality across countries and across different concepts of income.

Activity

Calculate the Gini index on total disposable income for Guatemala in 2006, using variables created in the previous exercise.

(1) Start with your program from the previous exercise, which will drop observations with missing data,

(2) Apply top and bottom codes,

(3) Create variables containing equivalised disposable income and disposable income per capita,

(4) Calculate the Gini coefficient for the winsorised (or bottom- and top-coded) versions of household income, per capita income, and equivalised income, and fill out the following table:

	Household income	Per capita income	Equivalised income
Gini coefficient			

Questions

4.1. Which measure shows greater inequality: household income, per capita income or equivalised income? What does this suggest about the possible relationship between income and household size?

Guidelines

- SPSS does not provide a ready-made command that will calculate the Gini directly, and actually, this requires some coding. Accordingly, the codes for generating gini is provided in the **ginicalc** block, all you need to do is to specify what income variable (inc_var) the routine should run on, as well as the weighting factor (wt_var) that should be applied. One important note is to make sure that the dataset is weighted by the weight factor as well.

```
compute wt_var = <varname>
compute inc_var = <varname>
```

ginicalc.

- In order to define the ginicalc block and loop it over different income concepts, this leads us to introducing SPSS macros.

- Looping through datasets: introducing SPSS macros

In SPSS, the **define** -- **!enddefine** commands can be used to mark the start and ending of blocks of code. All what is in between these marks can than easily be repeated as many times as needed. If we use the program of the last exercise, and put these marks around it, we can create a macro that performs the same analysis multiple times. Such a macro needs a name. Whenever the macroname is called, the block of code is executed:

```
define <macroname> ( )
…commands…
!enddefine
get file = first_dataset
<macroname>
get file = second_dataset
<macroname>
get file = third_dataset
<macroname>
etc. etc.
```

BE AWARE to write **!enddefine** in the correct spelling, NO space after exclamation mark.

Program

```
define ginicalc ()
sort cases by inc_var (a).
compute cumwgt = cumwgt + wt_var.
leave cumwgt.
aggregate outfile= *
  mode = addvariables
   /break= did
   /meany= mean(inc_var)
   /meanr= mean (cumwgt)
   /n=n.
compute devy= inc_var - meany.
compute rank= cumwgt/n.
compute devr = (rank - 0.5).
compute prod= devy*devr.
aggregate outfile= *
   /break=did
   /sumprod= sum(prod)
   /meany= mean(inc_var)
   /n=n.
compute cov= sumprod/(n-1).
compute gini=cov*2/meany.
formats gini (f10.4).
descriptives var=gini .
!enddefine .
```

```
define dataprepare ().
SET ERRORS OFF.
compute miss_comp = 0.
if (missing(dhi) or missing(hifactor) or missing(hpub_i) or missing(hpub_u) or missing(hpub_a)
or missing(hiprivate) or missing(hxitsc)) miss_comp = 1.
select if miss_comp eq 0.
* select only records if dhi filled.
select if not missing(dhi) .
* create top and bottom coded household disposable income.
compute dhi_tb = dhi.
* recode negaive dhi into zero
```

```
if (dhi_tb<0) dhi_tb=0.
EXECUTE.
compute dhi_log = ln(dhi_tb).
EXECUTE.
if (missing(dhi_log) & NOT(missing(dhi_tb))) dhi_log=0.
EXECUTE.
* create person weight as hwgt times number of household members.
compute wt = hpopwgt*nhhmem .
* create child weight as hwgt times number of household members under 18.
weight by wt.
!enddefine .

define decilecalc ().
preserve .
set tvars names tnumbers values.
dataset declare decileratio.
WEIGHT BY hpopwgt.
sort cases by did.
split file by did.
OMS
 / select tables
 / if command = ['Frequencies'] subtypes=['Statistics']
 /destination format = sav        outfile = 'decileratio'
 /columns sequence = [l1 r2] .
frequencies variables = dhi_log
  /percentiles = 25 50 75
  /format = notable .
OMSEND.
weights off.
restore.
match files file = *
  /table = 'decileratio'
  /rename (var1 = did)
  /by did .
!enddefine .

define topbottom ().
```

```
weight by wt.
COMPUTE iqr=dhi_log_75-dhi_log_25.
EXECUTE.
* detect upper bound for extreme values
COMPUTE upper_bound=dhi_log_75 + (iqr * 3).
EXECUTE.
COMPUTE lower_bound=dhi_log_25 - (iqr * 3).
EXECUTE.
* top code income at upper bound for extreme values
if dhi_tb>exp(upper_bound) dhi_tb=exp(upper_bound).
EXECUTE.
* bottom code income at lower bound for extreme values
if dhi_tb<exp(lower_bound) dhi_tb=exp(lower_bound).
EXECUTE.
* create equivalised income, set equivalence scale as square root of household members
compute edhi_tb = dhi_tb/(nhhmem**0.5).
compute pcdhi_tb = dhi_tb/nhhmem.
!enddefine .
```

get file = gt06h /keep=did dhi hifactor hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem grossnet.

*** run the dataprepare, decilecalc and topbottom**

dataprepare.

decilecalc.

topbottom.

*** set two input parameters (weighting and income variable).**

compute wt_var = hpopwgt.

compute inc_var = dhi_tb.

*** weight the dataset with the weight variable complying with the inequality measure**

weight by hpopwgt .

*** run the gini calculation block**

ginicalc.

get file = gt06h /keep=did dhi hifactor hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem grossnet.

*** set two input parameters (weighting and income variable).**

dataprepare.

decilecalc.

topbottom.

compute wt_var = hpopwgt*nhhmem.

compute inc_var = pcdhi_tb.

weight by wt_var.

ginicalc.

get file = gt06h /keep=did dhi hifactor hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem grossnet.

*** set two input parameters (weighting and income variable).**

dataprepare.

decilecalc.

topbottom.

compute wt_var = hpopwgt*nhhmem.

compute inc_var = edhi_tb.

weight by wt_var.

ginicalc.

Results

	Household income	Per capita income	Equivalised income
Gini coefficient	0.498	0.527	0.489

Solution

4.1. Which measure shows greater inequality: household income, per capita income or equivalised income? What does this suggest about the possible relationship between income and household size?

- The Gini for equivalised income is lower than the one for household income, which in turn is lower than the Gini for per-capita income. This reflects the fact that poorer households in Guatemala are on average larger than richer households. Because the LIS equivalence scale assumes some economies of scale in large households, it produces a lower estimate of inequality than a per-capita measure.

5. Relative poverty rates

Goal

In order to calculate any measure of poverty, it is essential to make some assumptions concerning the criteria used to define poverty. The approach used by LIS (and most commonly adopted in the literature), is that of creating a relative poverty line based on the level and distribution of (equivalised) household disposable income in the total population. Households are classified as poor or non-poor on the basis of whether their income is lower or higher than the relative poverty line.

Once poor households are identified, you can create an indicator to help identify the proportion of poor households (or individuals) and to measure the level of poverty. The choice of the indicator used will mainly depend on the purpose of the research. In this exercise, we will calculate the relative poverty rates of households and individuals in the Guatemala 2006 data.

Activity

(1) Add code to your program to produce an indicator for poverty.

(2) Define the poverty line as 50% of the median equivalised income.

(3) Calculate both the percentage of households in poverty and the head count ratio (defined as the percentage of individuals living in poor households), and complete the following table.

	Households	Individuals
Relative poverty rate		

Question

5.1. Are there more poor *households* or more poor *individuals?* What can you infer from this?

Guidelines

- From this point forward, we will be working exclusively with equivalised income, so the sections of your code relating to per-capita income can now be commented out or removed.
- You can create an indicator variable indicating that an individual is poor (**poor** = 0 or = 1). The mean of this variable will give you the proportion in poverty.
- Whether this is the proportion of *individuals* or *households* in poverty depends on which weighting you use. Use **hpopwgt** if you want to measure household poverty, and **hpopwgt*nhhmem** if you are interested in individual poverty.

If you use the individual-level weighting, you will produce the Head Count Ratio (HCR), which is the percentage of poor individuals in the total population.

- To get the median equivalised income, you can use again the command aggregate:

```
aggregate outfile = *
    mode=addvariables
    /break=<varname>
    /newvar=median(oldvar).
```

The principle is the same, fill **oldvar** now with **edhi_tb**.

- Create a variable povline that is equal to half the median of equivalised household income:

```
compute povline = medianedhi_tb * 0.5 .
```

- To determine the poverty rate, we create a dummy which is initialized at zero, and takes the value of 1 if a certain condition is satisfied (income below poverty line). The frequency table of that dummy will display the poverty rate.

```
compute poor = 0
if edhi_tb lt povline poor = 1
```

Program

```
define dataprepare ().
SET ERRORS OFF.
compute miss_comp = 0.
if (missing(dhi) or missing(hifactor) or missing(hpub_i) or missing(hpub_u) or missing(hpub_a)
     or missing(hiprivate) or missing(hxitsc)) miss_comp = 1.
select if miss_comp eq 0.
* select only records if dhi filled.
select if not missing(dhi) .
* create top and bottom coded household disposable income.
compute dhi_tb = dhi.
* recode negaive dhi into zero
if (dhi_tb<0) dhi_tb=0.
EXECUTE.
compute dhi_log = ln(dhi_tb).
EXECUTE.
if (missing(dhi_log) & NOT(missing(dhi_tb))) dhi_log=0.
EXECUTE.
* create person weight as hwgt times number of household members.
compute wt = hpopwgt*nhhmem .
weight by wt.
!enddefine .

define decilecalc ().
preserve .
set tvars names tnumbers values.
dataset declare decileratio.
WEIGHT BY hpopwgt.
sort cases by did.
split file by did.
OMS
 / select tables
 / if command = ['Frequencies'] subtypes=['Statistics']
 /destination format = sav        outfile = 'decileratio'
 /columns sequence = [l1 r2] .
```

```
frequencies variables = dhi_log
  /percentiles = 25 50 75
  /format = notable .
OMSEND.
weights off.
restore.
match files file = *
  /table = 'decileratio'
  /rename (var1 = did)
  /by did .
!enddefine .

define topbottom ().
weight by wt.
COMPUTE iqr=dhi_log_75-dhi_log_25.
EXECUTE.
* detect upper bound for extreme values
COMPUTE upper_bound=dhi_log_75 + (iqr * 3).
EXECUTE.
COMPUTE lower_bound=dhi_log_25 - (iqr * 3).
EXECUTE.
* top code income at upper bound for extreme values
if dhi_tb>exp(upper_bound) dhi_tb=exp(upper_bound).
EXECUTE.
* bottom code income at lower bound for extreme values
if dhi_tb<exp(lower_bound) dhi_tb=exp(lower_bound).
EXECUTE.
* create equivalised income, set equivalence scale as square root of household members
compute edhi_tb = dhi_tb/(nhhmem**0.5).
compute pcdhi_tb = dhi_tb/nhhmem.
!enddefine .

get file = gt06h /keep=did dhi hifactor hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem
      grossnet.

* run the dataprepare, decilecalc and topbottom
```

dataprepare.

decilecalc.

topbottom.

compute populat = hpopwgt * nhhmem.

weight by populat.

aggregate outfile = *

** mode = addvariables**

** /break = did**

** /medianedhi_tb = median(edhi_tb) .**

compute povline = medianedhi_tb * 0.5 .

descriptives variables = povline .

compute poor = 0 .

if edhi_tb lt povline poor = 1 .

weight by hpopwgt.

frequencies variables = poor .

weight by populat.

frequencies variables = poor .

Results

	Households	Individuals
Relative poverty rate	22.1%	22.5%

Solution

5.1. Are there more poor households or more poor individuals? What can you infer from this?

- There is a slightly greater proportion of poor individuals than poor households. This is because poor households are larger on average than non-poor households. This is another reason why the use of the equivalence scale is important.

Comments

- The head count ratio (HCR) measures poverty incidence (i.e., the number or proportion of poor people), but gives every person equal weight no matter how far they fall from the poverty line.
- Another measure, the Income Gap Ratio (IGR) measures poverty intensity or depth (how poor are the poor), but one poor person with an income of an amount x counts the same as two poor people each with an income of x/2. That is, the IGR measures the average income gap, but not its distribution among the poor.
- There are many other indicators of poverty that may be useful for different purposes. These include, among the most common, the whole family of Foster-Greer-Thorbecke indicators (of which the HCR is only one), the Sen index, the Takayama index, the Clark index, and the Thon index. It is important to note that a country may score better in comparison to a second country when using a particular index, but could score worse if another index was used instead.

6. Comparing income concepts

Goal

Now that we have calculated Gini coefficients and poverty rates based on equivalised household disposable income, we can easily apply this same code to three other income concepts: income before any taxes and government transfers, income after taxes, social insurance, and universal benefit transfers, and income after social assistance transfers. Starting from this exercise, we will also introduce some programming techniques that will make it easier for you to repeat a series of commands several times without having to repeat the code.

Different income concepts - The income variable we have been working with, **dhi**, combines multiple income and expenditure flows. It is the sum of labour and capital income, private transfers, private pensions, work-related insurance transfers, universal benefits, and social assistance benefits, minus any taxes and social insurance contributions paid. We will now define three new concepts of income. One of them is income before *any* government redistribution, but including private pensions. The second is income *after* taxes, social insurance, and universal benefits, but *before* social assistance is included. The third one is after social assistance transfers, but before taxes, social insurance, and universal benefits.

By calculating the Gini coefficient and the poverty rate using each of these four income concepts (income before government intervention, income after non-assistance government redistribution, income after social assistance benefits, and income after all government redistribution, i.e. our original disposable household income variable, **dhi**), we gain some insight into the effect of government programmes on inequality and poverty.

Efficient programming techniques - This exercise also introduces some programming techniques that allow looping the same code over several variables.

Activity

As always, begin with the program you developed for the last exercise. Modify it to create three new income variables. The first, **mi**, is the sum of factor income (**hifactor**), private transfers (**hiprivate**) and private pensions (**hi33**). Because we are specifically interested in the role of *government* transfers, we add private transfers and private pensions to our measure of "market income" from labour and capital.

The second, **siti**, adds **mi** together with social insurance transfers (**hpub_i**) and universal benefits (**hpub_u**), while subtracting taxes and social contributions paid (**hxitsc**).

The third measure, **sa**, adds together the market income **mi** with social assistance benefits (**hpub_a**).

The income variable we have been using up to now, disposable household income, adds together the variables contained in **siti** along with social assistance transfers (which are also contained in the variable **hpub_a**).

Make sure you apply bottom codes and the equivalence scale to the new variables, producing the final variables **emi_b, esiti_b, esa_b** and **edhi_b**.

Write a loop to calculate the Gini coefficient and the poverty rate for all four income variables, based on the code from the previous two exercises. Use it to fill out the table below. Make sure you use the *same* poverty line for all four income variables. That is, the poverty line should be defined as 50 percent of the median equivalised disposable household income, and that same poverty definition should be applied to the other three income variables.

	Before taxes and government transfers (mi)	After taxes, social insurance benefits and universal benefits (siti)	After social assistance benefits (sa)	After taxes and all government transfers (dhi)
Gini coefficient				
Poverty rate				

Question

6.1. Which government instruments have a greater impact on inequality and poverty in Guatemala: taxes/social insurance/universal benefits, or social assistance?

Guidelines

- Detailed information about income aggregates used in the exercise can be found in METIS: *METIS → LIS database → select <variables> → Results → Variable definitions*. In addition to this, it may be useful to consult the current list of variables on LIS website: *LIS website → Our data → LIS database → list of variables.*

- Note that social assistance, social insurance and universal benefits (**hpub_a**, **hpub_i** and **hpub_u**) together comprise the concept of public transfers. However, the *total sum of these benefits may be lower than the total public transfers reported at the higher-level variable **hpublic**.* This is because some amounts that do not clearly belong to one of the transfer categories may be reported directly in the variable **hpublic**. This may also be the case for other higher-level income concepts with an exception of the five blocks of the current income that always add up to the total income (labour income, capital

income, pensions, public social benefits and private transfers). For more information, you can consult the aggregation rules in the LIS User Guide.

You can easily check whether this is the case by summing up the sub-components and the higher-level variable of public transfers **hpublic**:

components and the higher-level variable of public transfers **hpublic**:

descriptives variables = hpublic hpub_i hpub_u hpub_a /statistics=mean /format=notable.

Program

```
define dataprepare ().
SET ERRORS OFF.
compute miss_comp = 0.
if (missing(dhi) or missing(hifactor) or missing(hi33) or missing(hpub_i) or missing(hpub_u) or missing(hpub_a) or missing(hiprivate) or missing(hxitsc)) miss_comp = 1.
select if miss_comp eq 0.
* create person weight as hwgt times number of household members.
compute wt = hwgt*nhhmem .
compute mi_tb = hifactor + hi33 + hiprivate.
compute siti_tb = hifactor +hi33 +  hiprivate + hpub_i + hpub_u - hxitsc.
compute sa_tb = hifactor + hi33 + hiprivate + hpub_a.
!enddefine .

define prepare ().
SET ERRORS OFF.
* select only records if dhi filled.
select if not missing(inc_var) .
* recode negaive dhi into zero
if (inc_var <0) inc_var =0.
EXECUTE.
compute inc_var_log = ln(inc_var).
EXECUTE.
if (missing(inc_var_log) & NOT(missing(inc_var))) inc_var_log =0.
EXECUTE.
weight by wt.
!enddefine .
```

```
define decilecalc ().
*** decile procedure.
preserve .
set tvars names tnumbers values.
dataset declare decileratio.
WEIGHT BY hwgt.
sort cases by did.
split file by did.
OMS
 / select tables
 / if command = ['Frequencies'] subtypes=['Statistics']
 /destination format = sav        outfile = 'decileratio'
 /columns sequence = [l1 r2] .
frequencies variables = inc_var_log
  /percentiles = 25 50 75
  /format = notable .
OMSEND.
weights off.
restore.
match files file = *
  /table = 'decileratio'
  /rename (var1 = did)
  /by did .
DATASET CLOSE decileratio.
!enddefine .

define topbottom ().
weight by wt.
COMPUTE iqr= inc_var_log_75- inc_var_log_25.
EXECUTE.
* detect upper bound for extreme values
COMPUTE upper_bound= inc_var_log_75+ (iqr * 3).
EXECUTE.
COMPUTE lower_bound= inc_var_log_25- (iqr * 3).
EXECUTE.
* top code income at upper bound for extreme values
if inc_var >exp(upper_bound) inc_var =exp(upper_bound).
```

```
EXECUTE.
* bottom code income at lower bound for extreme values
if inc_var <exp(lower_bound) inc_var =exp(lower_bound).
EXECUTE.
* create equivalised income, set equivalence scale as square root of household members
compute ey = inc_var /(nhhmem**0.5).
!enddefine .

define povcalc ().
aggregate outfile = *
   mode = addvariables
   / break = did
   / ey_median = median(ey)  .
compute povline5 = ey_median * 0.5 .
match files file=*/keep
did povline5.
Dataset copy poverty.
!enddefine .

define povcalccnt ().
MATCH FILES /FILE=*
  /FILE='poverty'
  /RENAME (did = d0)
  /DROP= d0.
EXECUTE.
compute poor50 = 0 .
if ey lt povline5 poor50 = 100 .
variable labels poor50 'overall poverty rate 50%'.
weight by wt.
descriptives  poor50 /statistics=mean /format=notable.
!enddefine .

define ginicalc ()
sort cases by ey (a).
compute cumwgt = cumwgt + wt.
leave cumwgt.
aggregate outfile= *
```

```
   mode = addvariables
    /break= did
    /meany= mean(ey)
    /meanr= mean (cumwgt)
    /n=n.
compute devy= ey - meany.
compute rank= cumwgt/n.
compute devr = (rank - 0.5).
compute prod= devy*devr.
aggregate outfile= *
    /break=did
    /sumprod= sum(prod)
    /meany= mean(ey)
    /n=n.
compute cov= sumprod/(n-1).
compute gini=cov*2/meany.
formats gini (f10.4).
descriptives var=gini .
!enddefine .


get file = gt06h /keep=did dhi hifactor hi33 hpublic hpub_i hpub_u hpub_a hiprivate hxitsc
hpopwgt nhhmem grossnet hwgt nhhmem nhhmem17 nhhmem65  .
dataprepare.
* set one input parameter (income variable).
compute inc_var =dhi.
prepare.
decilecalc.
topbottom.
povcalc.
compute poor50 = 0 .
if ey lt povline5 poor50 = 100 .
variable labels poor50 'overall poverty rate 50%'.
weight by wt.
descriptives  poor50 /statistics=mean /format=notable.
TITLE Gini using dhi.
ginicalc.
```

```
get file = gt06h /keep=did dhi hifactor hi33 hpublic hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem grossnet hwgt nhhmem nhhmem17 nhhmem65 .

dataprepare.

* set one input parameter (income variable).

compute inc_var = mi_tb.

prepare.

decilecalc.

topbottom.

TITLE Poverty using mi_tb.

povcalccnt.

TITLE Gini using mi_tb.

ginicalc.


get file = gt06h /keep=did dhi hifactor hi33 hpublic hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem grossnet hwgt nhhmem nhhmem17 nhhmem65 .

dataprepare.

* set one input parameter (income variable).

compute inc_var = siti_tb.

prepare.

decilecalc.

topbottom.

TITLE Gini using siti_tb.

povcalccnt.

TITLE Gini using siti_tb.

ginicalc.


get file = gt06h /keep=did dhi hifactor hi33 hpublic hpub_i hpub_u hpub_a hiprivate hxitsc hpopwgt nhhmem grossnet hwgt nhhmem nhhmem17 nhhmem65 .

dataprepare.

* set one input parameter (income variable).

compute inc_var = sa_tb.

prepare.

decilecalc.

topbottom.

TITLE Poverty using sa_tb.

povcalccnt.

TITLE Gini using sa_tb.

ginicalc.
```

Results

	Before taxes and government transfers (mi)	After taxes, social insurance benefits and universal benefits (siti)	After social assistance benefits (sa)	After taxes and all transfers (dhi)
Gini coefficient	0.516	*0.496* *	0.509	0.489
Poverty rate	24.5%	*24.0%* *	22.9%	22.5%

* Calculation excludes universal benefits since it is not filled in gt06.

Solution

6.2. Which government instruments have a greater impact on inequality and poverty in Guatemala: taxes/social insurance/universal benefits, or social assistance?

- Taxes, social insurance benefits and universal benefits play a larger role in reducing income inequality in Guatemala in 2006 than social assistance benefits, and the reverse is true for poverty reduction.

Comments

- When comparing incomes before and after taxes and transfers, be careful when interpreting the meaning of the pre-tax and transfer figure. It is tempting to interpret this number as a representation of the income distribution that would exist in the absence of government programs. However, since outcomes in the private sector are conditioned by the presence or absence of government programs, this is not generally a reasonable inference.

- There are several online databases containing detailed information on social security systems around the world that may be useful in your analysis:

  - The Mutual Information System on Social Protection (MISSOC), which provides up-to-date information on social protection legislation, benefits and conditions for 28 EU Member States, Iceland, Liechtenstein, Norway and Switzerland (and MISSCEO for countries outside of the EU's MISSOC network);
  - Social Security Programs Throughout the World – a biannual publication highlighting the principal features of social security programs in more than 170 countries;

- Other databases and publications from institutional websites, such as the International Labour Organisation (ILO), the Organisation for Economic Co-operation and Development (OECD) and EUROMOD – the tax-benefit microsimulation model for European Union;
- National social security websites.

7. Comparing multiple countries

Goal

Now that we have written code to compute all of our statistics of interest, it is time to calculate these quantities for multiple countries. Building up on the previous exercise, we will introduce different programming techniques that break the code into logic sub-routines and generalise our program to loop through multiple datasets.

Adding more countries – Before adding a new country/year to an analysis, it is important to check that the dataset in question has all information necessary for the analysis you are performing. In this case, one should carefully check that the income sub-components variables used in the previous exercise are filled for all the new datasets to be introduced (and if not, whether the analysis can be slightly modified to take into account a different situation).

Efficient programming techniques – In the previous exercise we have introduced some programming techniques that allowed to loop the same code over several variables. In this exercise, we will introduce some other techniques that allow to organise the code in an efficient way and easily loop over both variables and datasets.

Activity

(1) Take the code from the previous exercise and modify it so that it loops through five datasets: Guatemala 2006 (**gt06**), United States 2004 (**us04**), Denmark 2004 (**dk04**), Hungary 2005 (**hu05**) and Israel 2005 (**il05**).

(2) The code that creates the income variables can be placed in a subroutine that is called from the main loop, as can the code that applies the equivalence scale and the bottom code.

Use your results to fill in the following tables:

Gini Coefficient

Dataset	Before taxes and government transfers	After taxes, social insurance benefits and universal benefits	After social assistance benefits	After taxes and all transfers
GT06				
US04				
DK04				
HU05				
IL05				

Poverty Rate

Dataset	Before taxes and government transfers	After taxes, social insurance benefits and universal benefits	After social assistance benefits	After taxes and all transfers
GT06				
US04				
DK04				
HU05				
IL05				

Keep in mind that even if all cells can technically be constructed, the result may not necessarily be comparable conceptually! Think carefully about whether the dataset you are looking at contains the necessary information to calculate the quantity in each column.

Question

7.1. In what cells does the figure you produced not match the income concept described in the column header?

7.2. In which country do government programmes do the most to reduce inequality and poverty, in percentage terms? In which country do they do the least?

7.3. In which countries do social assistance benefits do more to reduce poverty than social insurance plus universal benefits and taxes?

Guidelines

- Looping through datasets: defining a macro with arguments

  If you want to perform a loop through many countries, you could go even one step further, and extend the macro with an argument (in our example number of repetitions). You have to specify the number of repetitions in the start, and list the names of the datasets when you call the macroname. BE AWARE: make sure that the number **#** matches the number of dataset that you list when calling the macroname. We only recommend this further step to those who feel comfortable with macro language:

```
define <macroname> (!positional  !tokens(#)
!do !i !in (!1)
get file = !i /keep= etc….
…commands…
!doend
!enddefine
```

```
<macroname>    first_dataset second_dataset etc. etc.
```

- Noting irregular cases

The variable **grossnet** reports whether the incomes in a dataset are gross income, before taxes, or whether they only report post-tax values. Within a single dataset, all cases will have the same value for this variable. In datasets containing gross incomes, the **grossnet** variable can take values 111, 110 or 120. In a purely net income dataset, **grossnet** will be 200. Note, however, that a few datasets have **grossnet** codes of 300, 310 or 320 because they contain a mixture of gross and net incomes. More information about whether the LIS datasets report gross or net values can be found on the LIS METIS documentation system *at Our Data → METIS → LIS database → select "ccyy" → Dataset information*, while the *Crossed-compare* tab allows you to check the variable value taken for the selected dataset(s). METIS is accessed through this link.

You can check whether gross incomes are available in each dataset by tabulating the **grossnet** variable:

```
frequencies variables = grossnet.
```

As in the previous exercise, it is worth noting whether social benefits add up to the concept of public transfers in every country:

```
descriptives variables = hpublic hpub_i hpub_u hpub_a /statistics=mean
/format=notable.
```

Program

```
define gt06data ()
get file = gt06h / keep = did dhi hi33 hpublic hifactor hpub_i hpub_u hpub_a
hiprivate hxitsc hpopwgt nhhmem grossnet hwgt.
!enddefine
define us04data ()
get file = us04h / keep = did dhi hi33 hpublic hifactor hpub_i hpub_u hpub_a
hiprivate hxitsc hpopwgt nhhmem grossnet hwgt.
!enddefine
define dk04data ()
get file = dk04h / keep = did dhi hi33 hpublic hifactor hpub_i hpub_u hpub_a
hiprivate hxitsc hpopwgt nhhmem grossnet hwgt.
!enddefine
define hu05data ()
get file = hu05h / keep = did dhi hi33 hpublic hifactor hpub_i hpub_u hpub_a
hiprivate hxitsc hpopwgt nhhmem grossnet hwgt.
!enddefine
define il05data ()
get file = il05h / keep = did dhi hi33 hpublic hifactor hpub_i hpub_u hpub_a
hiprivate hxitsc hpopwgt nhhmem grossnet hwgt.
!enddefine
```

```
define dataprepare ().

SET ERRORS OFF.

compute miss_comp = 0.

if (missing(dhi) or missing(hifactor) or missing(hi33) or missing(hpub_i) or missing(hpub_u) or
missing(hpub_a) or missing(hiprivate) or missing(hxitsc)) miss_comp = 1.

select if miss_comp eq 0.

* create person weight as hwgt times number of household members.

compute wt = hwgt*nhhmem .

compute mi_tb = hifactor + hi33 + hiprivate.

compute siti_tb = hifactor +hi33 +  hiprivate + hpub_i + hpub_u - hxitsc.

compute sa_tb = hifactor + hi33 + hiprivate + hpub_a.

!enddefine .


define prepare ().

SET ERRORS OFF.

* select only records if dhi filled.

select if not missing(inc_var) .

* recode negaive dhi into zero

if (inc_var <0) inc_var =0.

EXECUTE.

compute inc_var_log = ln(inc_var).
```

```
EXECUTE.
if (missing(inc_var_log) & NOT(missing(inc_var))) inc_var_log =0.
EXECUTE.
weight by wt.
!enddefine .

define decilecalc ().
*** decile procedure.
preserve .
set tvars names tnumbers values.
dataset declare decileratio.
WEIGHT BY hwgt.
sort cases by did.
split file by did.
OMS
 / select tables
 / if command = ['Frequencies'] subtypes=['Statistics']
 /destination format = sav        outfile = 'decileratio'
 /columns sequence = [l1 r2] .
frequencies variables = inc_var_log
  /percentiles = 25 50 75
  /format = notable .
OMSEND.
weights off.
restore.
match files file = *
  /table = 'decileratio'
  /rename (var1 = did)
  /by did .
DATASET CLOSE decileratio.
!enddefine .

define topbottom ().
weight by wt.
```

```
COMPUTE iqr= inc_var_log_75- inc_var_log_25.
EXECUTE.
* detect upper bound for extreme values
COMPUTE upper_bound= inc_var_log_75+ (iqr * 3).
EXECUTE.
COMPUTE lower_bound= inc_var_log_25- (iqr * 3).
EXECUTE.
* top code income at upper bound for extreme values
if inc_var >exp(upper_bound) inc_var =exp(upper_bound).
EXECUTE.
* bottom code income at lower bound for extreme values
if inc_var <exp(lower_bound) inc_var =exp(lower_bound).
EXECUTE.
* create equivalised income, set equivalence scale as square root of household members
compute ey = inc_var /(nhhmem**0.5).
!enddefine .

define run ().
prepare.
decilecalc.
topbottom.
!enddefine.

define povcalc ().
aggregate outfile = *
  mode = addvariables
  / break = did
  / ey_median = median(ey)  .
compute povline5 = ey_median * 0.5 .
match files file=*/keep
did povline5.
Dataset copy poverty.
!enddefine .
```

```
define povcalccnt ().
MATCH FILES /FILE=*
  /FILE='poverty'
  /RENAME (did = d0)
  /DROP= d0.
EXECUTE.
compute poor50 = 0 .
if ey lt povline5 poor50 = 100 .
variable labels poor50 'overall poverty rate 50%'.
weight by wt.
descriptives  poor50 /statistics=mean /format=notable.
!enddefine .

define ginicalc ()
sort cases by ey (a).
compute cumwgt = cumwgt + wt.
leave cumwgt.
aggregate outfile= *
   mode = addvariables
    /break= did
    /meany= mean(ey)
    /meanr= mean (cumwgt)
    /n=n.
compute devy= ey - meany.
compute rank= cumwgt/n.
compute devr = (rank - 0.5).
compute prod= devy*devr.
aggregate outfile= *
    /break=did
    /sumprod= sum(prod)
    /meany= mean(ey)
    /n=n.
compute cov= sumprod/(n-1).
compute gini=cov*2/meany.
```

```
formats gini (f10.4).
descriptives var=gini .
!enddefine .

define edhi ()
dataprepare.
compute inc_var = dhi.
run.
povcalc.
compute poor50 = 0 .
if ey lt povline5 poor50 = 100 .
variable labels poor50 'overall poverty rate 50%'.
weight by wt.
descriptives  poor50 /statistics=mean /format=notable.
TITLE Gini using dhi.
ginicalc.
!enddefine.


define emi ()
dataprepare.
compute inc_var = mi_tb.
run.
TITLE Poverty using mi_tb.
povcalccnt.
TITLE Gini using mi_tb.
ginicalc.
!enddefine.


define esiti ()
dataprepare.
compute inc_var = siti_tb.
run.
```

```
TITLE Gini using siti_tb.
povcalccnt.
TITLE Gini using siti_tb.
ginicalc.
!enddefine.


define esa ()
dataprepare.
compute inc_var = sa_tb.
run.
TITLE Poverty using sa_tb.
povcalccnt.
TITLE Gini using sa_tb.
ginicalc.
!enddefine.

define stat ()
frequencies variables = grossnet.
descriptives variables = hpublic hpub_i hpub_u hpub_a /statistics=mean /format=notable.
!enddefine

TITLE ================================================.
TITLE Guatemala.
TITLE ================================================.
gt06data.
edhi.
gt06data.
emi.
gt06data.
esiti.
gt06data.
esa.
gt06data.
```

stat.

TITLE ==========.

TITLE United States.

TITLE ==========.

us04data.

edhi.

us04data.

emi.

us04data.

esiti.

us04data.

esa.

us04data.

stat.

TITLE ==========.

TITLE Denmark.

TITLE ==========.

dk04data.

edhi.

dk04data.

emi.

dk04data.

esiti.

dk04data.

esa.

dk04data.

stat.

TITLE ==========.

TITLE Hungary.

TITLE ==========.

hu05data.

edhi.

hu05data.

emi.

hu05data.

esiti.

hu05data.

esa.

hu05data.

stat.

TITLE ==.

TITLE Israel.

TITLE ==.

il05data.

edhi.

il05data.

emi.

il05data.

esiti.

il05data.

esa.

il05data.

stat.

Results

Gini Coefficient

Dataset	Before taxes and government transfers	After taxes, social insurance benefits and universal benefits	After social assistance benefits	After taxes and all transfers
GT06	0.516	0.496 **	0.509	0.489
US04	0.488	0.399	0.468	0.374
DK04	0.422	0.262	0.395	0.230
HU05	0.527 *	0.305	0.510	0.288
IL05	0.506	0.408	0.486	0.383

Poverty Rate

Dataset	Before taxes and government transfers	After taxes, social insurance benefits and universal benefits	After social assistance benefits	After taxes and all transfers
GT06	24.5	24.0 **	22.9	22.5
US04	26.2	21.3	22.9	17.2
DK04	25.3	12.7	22.2	5.6
HU05	42.6 *	9.6	41.3	6.7
IL05	28.6	22.6	26.8	19.3

*Calculation based on post-tax income

**Calculation excludes universal benefits

Solution

7.1. In what cells does the figure you produced not match the income concept described in the column header?

- Because Hungary 2005 is a net income dataset, the Gini before taxes and transfers cannot be included. While the example program below does produce a result for Hungary, it is not actually comparable to the other countries because it is post-tax. That cell should therefore be left blank.

7.2. In which country do government programmes do the most to reduce inequality and poverty, in percentage terms? In which country do they do the least?

- Government programmes have the largest impact on reducing inequality in Denmark, where they reduce the Gini coefficient by 45 percent, and poverty in Hungary by 84 percent.

- Government programs have the lowest impact on reducing inequality and poverty in Guatemala, where they reduce the Gini coefficient by 6 percent and poverty by 8 percent.

7.3. In which countries do social assistance benefits do more to reduce poverty than social insurance plus universal benefits and taxes?

- In Guatemala, social assistance reduces poverty more than social insurance plus taxes and universal benefits.

Comments

- The datasets in these exercises were chosen because they allow social assistance to be separated from other kinds of government transfers. In many datasets, unfortunately, this separation is not possible due to the limitations of the original data. In such cases, the total amount of transfer income will be contained in a higher-level variable such as **hpublic**, and lower level variables such as **hpub_i** and **hpub_u** will be unfilled. You can consult the **Crossed-compare** function in the METIS documentation tool, to determine which variables are available in each dataset. This is the **main functionality** of METIS. Upon entering the tool (**METIS**), it is possible to view the availability of the selected variable(s) in the selected dataset(s) and to **compare information about this selection**. To access this information, *Select datasets → Select variables (under Income Aggregates) → Results → Crossed-Compare*.

Additional guidelines on producing graphs with LISSY

This additional section introduces LISSY's graphing feature that allows users to generate, display and export graphs based on LIS data.

To generate graphs in LISSY, the syntax below needs to be used after the SPSS graphing syntax. It allows users to display graphs on the Web-based Job Submission Interface and download them in PDF or png format.

OUTPUT EXPORT /JPG DOCUMENTFILE="mypdf\graphtestspss.jpg

Program

```
GET FILE = lu10p.
GRAPH
    /BAR(SIMPLE)=relation BY edyrs.
OUTPUT EXPORT /JPG DOCUMENTFILE="mypdf\graphtestspss.jpg
```