SOUTHMOD User Manual

A manual for the harmonized SOUTHMOD model bundle

Last updated: 7 March 2024

This manual is prepared with reference to SOUTHMOD bundle version A2.0 and will be equally applicable to future iterations of the harmonized model.

This document is a user manual for the harmonized SOUTHMOD model bundle, incorporating country models for Africa (Ethiopia, Ghana, Mozambique, Rwanda, Mainland Tanzania, Uganda, Zambia, and Zanzibar); Latin America (Bolivia, Colombia, Ecuador, and Peru); and Southeast Asia (Viet Nam). The manual complements the SOUTHMOD Modelling Conventions[1] as well as Country Reports developed for each individual model.

This manual is designed as an introductory guide for new users and a reference for those already familiar with the basic operations of SOUTHMOD models. The manual provides comprehensive instructions for using any of the country models for the first time, along with more complex tasks such as building new policies. The focus of the manual is on the technicalities of how to use the SOUTHMOD models in practice, rather than on the many processes that could be undertaken using the EUROMOD software more generally.

The manual is shared publicly as a courtesy for any interested SOUTHMOD model users. Users should request access to the bundle, with access to input data for required country models[2], and download the newest release of the EUROMOD software[3]. We also recommend that the relevant Country Reports and associated Data Requirement Documents (available from UNU-WIDER by request along with the model bundle) are studied carefully.

Acknowledgments

This manual was produced as part of the SOUTHMOD programme at UNU-WIDER, a research project focusing on the development of tax-benefit microsimulation models for developing countries. This user manual has been amended and unified from country-specific manuals so that it applies to the SOUTHMOD model bundle, developed in the collaboration between UNU-WIDER, SASPRI, and national teams of the respective countries in the SOUTHMOD project. Much of the material in the original manuals, used as a basis for this document, has been drawn from documentation prepared for the EUROMOD model. The authors are grateful to the EUROMOD team for granting their permission to use this material.

---

[1] SOUTHMOD Modelling Conventions: https://www.wider.unu.edu/sites/default/files/Projects/PDF/SOUTHMOD_Modelling_Conventions_20240307.pdf

[2] Request access to the SOUTHMOD model bundle, along with input data, Country Reports and DRDs: https://www.wider.unu.edu/about/accessing-southmod-models

[3] Latest EUROMOD software release: https://euromod-web.jrc.ec.europa.eu/access-euromod - inline-nav-1

Contents

Table of figures

List of tables

1 Introduction

1.1 About SOUTHMOD models

Tax-benefit microsimulation models for developing countries

Tax-benefit microsimulation models combine:

- Representative household-level data on incomes and expenditures ('micro data'); and
- Detailed coding of tax and benefit legislation ('policy rules').

The models apply the user-defined policy rules to the micro data and calculate the effects of these rules on household income, and thereby on poverty, inequality, and government budget. A range of policy scenarios or reforms can be evaluated and compared.

Many developing countries are gradually building up their social protection systems, and the financing of public spending will need to be increasingly based on domestic tax revenues. In this process, understanding the system-wide impacts of different policy choices is critically important, and tax-benefit microsimulation models are very well suited for this purpose. Tax-benefit microsimulation models are useful for policymakers and researchers alike.

Against this backdrop, UNU-WIDER and Southern African Social Policy Research Insights (SASPRI) have launched SOUTHMOD. It is a research project that focuses on the development of tax-benefit microsimulation models for selected developing countries, namely, those in:

- Africa *(Ethiopia, Ghana, Mozambique, Rwanda, Mainland Tanzania, Uganda, Zambia, and Zanzibar);*
- Latin America *(Bolivia, Colombia, Ecuador, and Peru);* and
- Southeast Asia *(Viet Nam)*

Each model has been developed in cooperation with local partners, including researchers and staff members from academia, tax authorities, and government ministries. The models are based on national household surveys that allow for representative results on both the national and sub-national level.

SOUTHMOD models

Each SOUTHMOD model constitutes a versatile yet easy-to-use tool for both policymakers and researchers. The models allow for applying a set of policy rules to household survey data and then calculate individual entitlements to benefits and liabilities for taxation. The resulting output at the individual and household level can then be analysed to provide national data on, for example, impact of social benefits on poverty and inequality.

The models also allow for calculating the effects of policy changes on the government spending and revenue, the latter accounting for both direct and indirect taxes. Possible research questions may include, for example, how specific tax rates could be increased to offset any additional expenditures on social protection.

In addition to economy-wide and population-wide effects, it is possible to estimate the number of beneficiaries of certain social protection policies, or the number of citizens who pay taxes, along with the characteristics of these populations (e.g. gender, formality status, or industry).

The base models simulate the existing tax-and-benefit arrangements within each country. Their real strength, however, is that they make it possible to simulate hypothetical changes in social protection benefits and taxes (e.g. alternative or planned tax-benefit reforms).

Some of the limitations of SOUTHMOD models (which apply to effectively all tax-benefit microsimulation models) include the following:

- The survey data underlying each model only captures individuals and households in a given country, not corporate firms, so policy questions regarding firms (e.g. corporate taxation and subsidies) are outside the scope of these models.
- Certain tax-benefit policies typically cannot be modelled due to limitations in the underlying survey data. To properly model pension payments in some countries, for example, the micro data would have to contain detailed information on individuals' contributory history. This is not the case in most household surveys, including those underpinning SOUTHMOD models.
- The models are static and as such do not simulate changes in behaviour. For example, they do not take into account changes in labour supply or transitions between formal and informal sector following changes in tax rates or benefit amounts. Advanced users with statistical knowledge may wish to model such behavioural changes.
- The models do not account for macroeconomic effects. As with behavioural responses, advanced users can opt to model micro-macro linkages using the standard models as a starting point. In SOUTHMOD models, the one exception where macroeconomic effects are considered is the COVID-19 pandemic. For details, see 'COVID-19 transitions and shocks (lma_cc)' in Section 6.1.

EUROMOD platform

All SOUTHMOD models are built on the EUROMOD platform, originally built by Professor Holly Sutherland and colleagues at the University of Essex to simulate policies for the European Union countries.[4] EUROMOD has been built and developed over more than two decades, and is now available for 28 European countries.

EUROMOD offers a suitable basis for SOUTHMOD models due to the fact that all calculations are transparent and can be easily modified by the user; models built on EUROMOD are very flexible, allowing policies to be modified and almost any type of new policy to be created.

[4] Sutherland, H. and F. Figari (2013). EUROMOD: The European Union tax-benefit microsimulation model. *International Journal of Microsimulation*, 6(1): 4–26.

EUROMOD also has a stand-alone user-friendly interface, which is stable and compatible with computers running on Windows operating systems and provides greater control and guidance over user actions.

SOUTHMOD model bundle

This harmonized SOUTHMOD model bundle takes advantage of EUROMOD's functionality to work with several countries in a single project and thereby conduct comparative analysis using different models. The SOUTHMOD bundle, discussed in this manual, incorporates country models for all countries listed above.

1.2 Introduction to this manual

After this first section, this manual is organized into the following interlinked sections:

2) Getting started (the structure of the SOUTHMOD model bundle, user interface, and running a given country model)
3) Introduction to datasets, systems and policies
4) Menus, settings and actions
5) Tax-benefit policies and their modelling (with details on functions and parameters)
6) Definitional policies
7) Variables
8) Modelling in practice
9) Using the Statistics Presenter

Special modelling terms are selectively printed in blue bold italics (e.g. ***policy spine*** or ***income list***), while EUROMOD buttons, controls and functionalities are printed in green bold italics (e.g. ***Countries*** menu tab or ***Run SOUTHMOD*** button). Other important terms, such as file names and datasets, are printed in ***bold italics***. 'cc' is used across the document as a placeholder that can refer to any country model (e.g. 'ug' for UGAMOD, the model for Uganda or 'gh' for GHAMOD, the model for Ghana).

1.3 Complementary materials

SOUTHMOD Modelling Conventions

The SOUTHMOD Modelling Conventions (SMC) are intended to ensure that all models part of the SOUTHMOD project are based on the same assumptions and conventions regarding the modelling and input data. The harmonized methodology allows for the comparability of model outcomes across different countries, thereby facilitating comparative research and analysis. The conventions are furthermore linked as closely as possible with the conventions of the European EUROMOD models. In principle this makes it possible to include European countries in such comparative analysis as well. Lastly, the modelling conventions are also intended to support the development of a new tax-benefit microsimulation model and the upkeep of existing models.

SMC is shared publicly in order to facilitate transparency about the underlying assumptions and conventions of the SOUTHMOD tax-benefit microsimulation models. The document may also be helpful for researchers aspiring to build their own model based on the freely available EUROMOD software either as part of the SOUTHMOD project or independently.

The treatment of most topics covered in the SMC are kept to a minimum in this User Manual. The reader is advised to consult the SMC especially on the following subjects:

- Required variables with associated categories as well as naming conventions;
- Details on system and database configuration;
- Required policies and related conventions;
- Default switchable policies (extensions);
- Income and expenditure concepts and their use in income lists; and
- Guidance on indirect taxes, model output, and validation.

SOUTHMOD Country Reports

SOUTHMOD Country Reports describe individual country models (e.g. UGAMOD for Uganda, RWAMOD for Rwanda, VNMOD for Viet Nam, etc.) in greater detail. The treatment of most topics covered in these Country Reports are kept to a minimum in this User Manual. The reader is advised to consult the Country Reports especially on the following subjects:

1. Taxes and benefits in place in the relevant country;
2. The implementation of these taxes and benefits in the given model;
3. Survey data used in the given model; and
4. Validation of model outputs against external statistics.

SOUTHMOD Data Requirement Documents

SOUTHMOD Data Requirement Documents (DRDs) describe the variables used in all country models (namely, the input data for each model), including personal or demographic, labour market, income, asset, and expenditure variables. In addition to the SMC, which covers the mandatory variables required for each SOUTHMOD model, the reader is advised to consult the DRDs for information on the country-specific variables (sample size, variable names, details on derivations from original survey data, relevant categories, and summary statistics).

2 Getting started

2.1 Structure of the SOUTHMOD model bundle

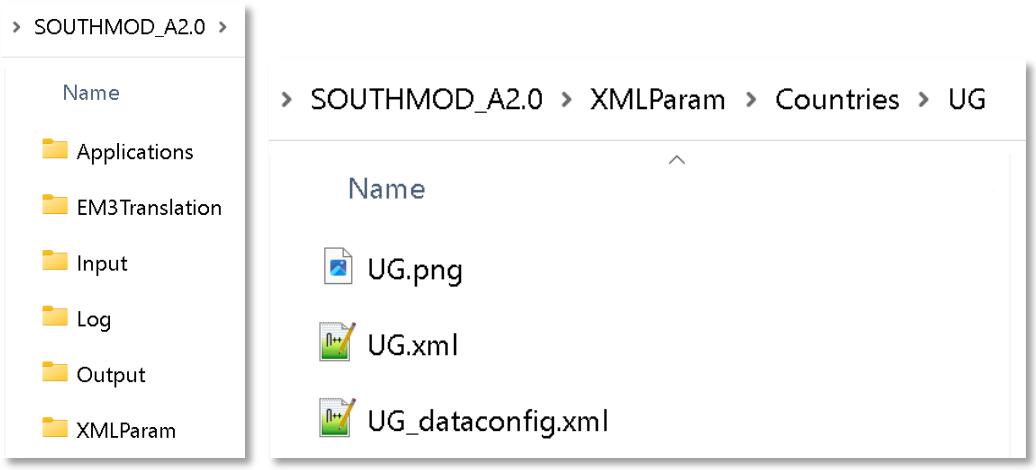
Each tax-benefit microsimulation model consists of:

1) ***A software file*** (i.e. the EUROMOD software installed on Windows), which includes the standard user interface, executable, and integrated help menu; and
2) ***Content files***, including the policy rules for each country, as well as various other tools and applications.

The software (available from the EUROMOD website maintained by the JRC[5,6]) and content files can be updated separately from each other allowing greater flexibility.

For the SOUTHMOD model bundle (and all other EUROMOD models), the content files are stored in a model folder, shown in Figure 2.1.

Figure 2.1: Model folder structure (left) and main content files for UGAMOD (right)



The folder structure includes:

- ***Applications:*** Specific to European EUROMOD models, not used for SOUTHMOD
- ***EM3 Translation:*** Translations of main content files, such as model rules and variables, to the EUROMOD software
- ***Input:*** Input micro data files underpinning each country model
- ***Log:*** A change log describing all changes to individual country models
- ***Output*****:** Output micro data files produced when running individual country models
- ***XMLParam:*** Main content files, such as model rules and variables, as well as general project files

[5] Download EUROMOD from here: https://euromod-web.jrc.ec.europa.eu/access-euromod#inline-nav-1

[6] Visit the main website of the European Commission's Joint Research Centre (JRC) here: https://joint-research-centre.ec.europa.eu/index_en

The information that the model needs for its calculations is stored in two main content files (e.g. ***ug.xml*** and ***ug_DataConfig.xml*** in the *UGAMOD* model for Uganda). These files contain information for the implementation of both the framework of the tax-benefit model and the particular policies that make up the tax-benefit system. The files are not accessed directly by the user, but are stored in the model folder (e.g. in ***/XMLParam/Countries/UG/*** for UGAMOD, and similarly for other individual country models).

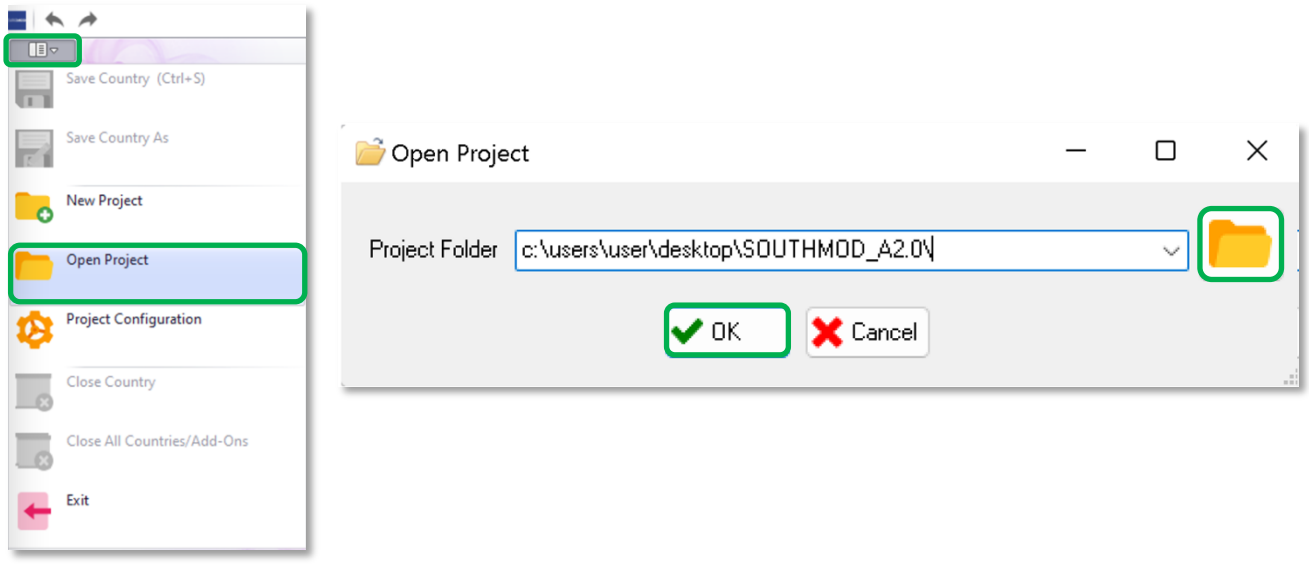
Each model draws on input data stored in text files (found in folder ***/Input***) and returns the output to a text file (in folder ***/Output***). The output data can be analysed in STATA, Excel, or any other statistical programme.

2.2 Opening the SOUTHMOD model bundle

As illustrated in Figure 2.2, the SOUTHMOD bundle – or any model – can be accessed by:

1. Opening the *EUROMOD* software installed on a Windows machine;
2. Navigating to the file menu on the top-left corner;
3. Clicking ***Open Project***;
4. Clicking the yellow folder in the pop-up menu and navigating to the main *SOUTHMOD_A2.0* model folder; and
5. Clicking ***OK***.

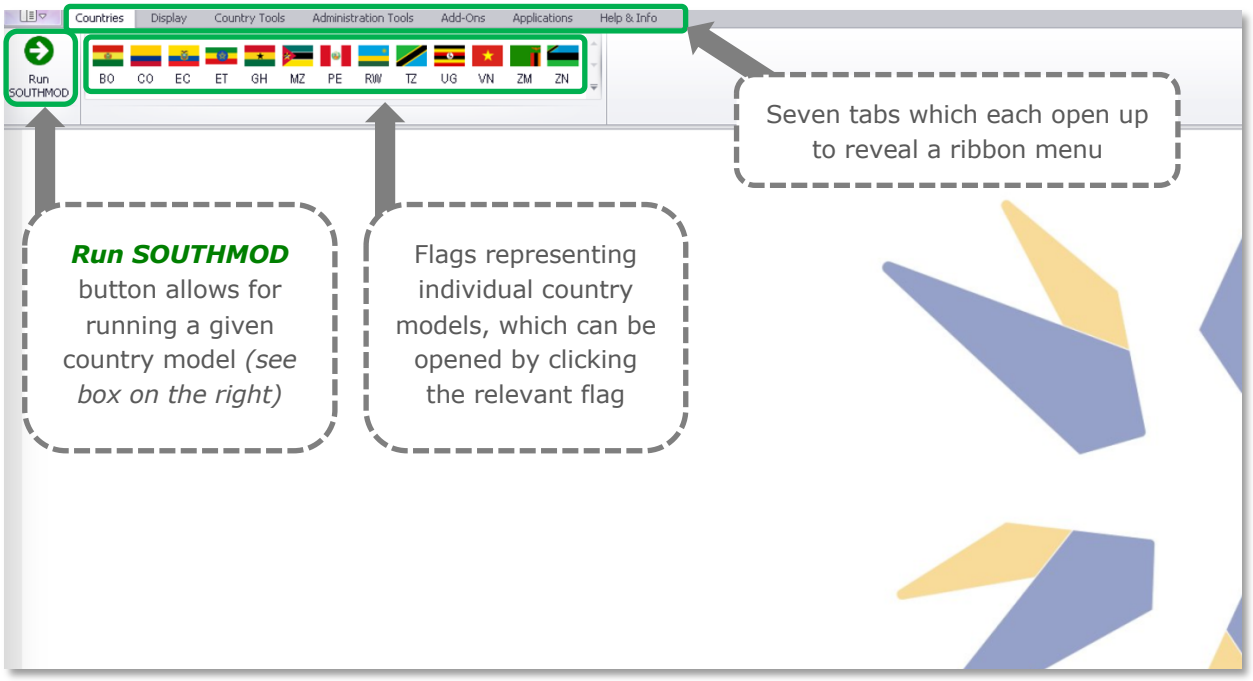
Figure 2.2: Opening the SOUTHMOD model bundle



The EUROMOD software interface should now show the flags of each country model included in the SOUTHMOD model bundle, as shown in Figure 2.3. The user interface has seven tabs – ***Countries***, ***Display***, ***Country Tools***, ***Administration Tools***, ***Add-ons***, ***Applications***, and ***Help & Info***. Each tab opens up to reveal a ribbon menu with a number of functionalities; see section 4.2 for details.

The ***Countries*** tab contains *flags for available models* (allowing to open specific country models) and a ***Run SOUTHMOD*** button (allowing to run a given model). It is also possible to run a model or models via ***Run SOUTHMOD*** *before* opening them from the user interface.

Figure 2.3: EUROMOD interface after opening the SOUTHMOD bundle



2.3 Model interface and key definitions

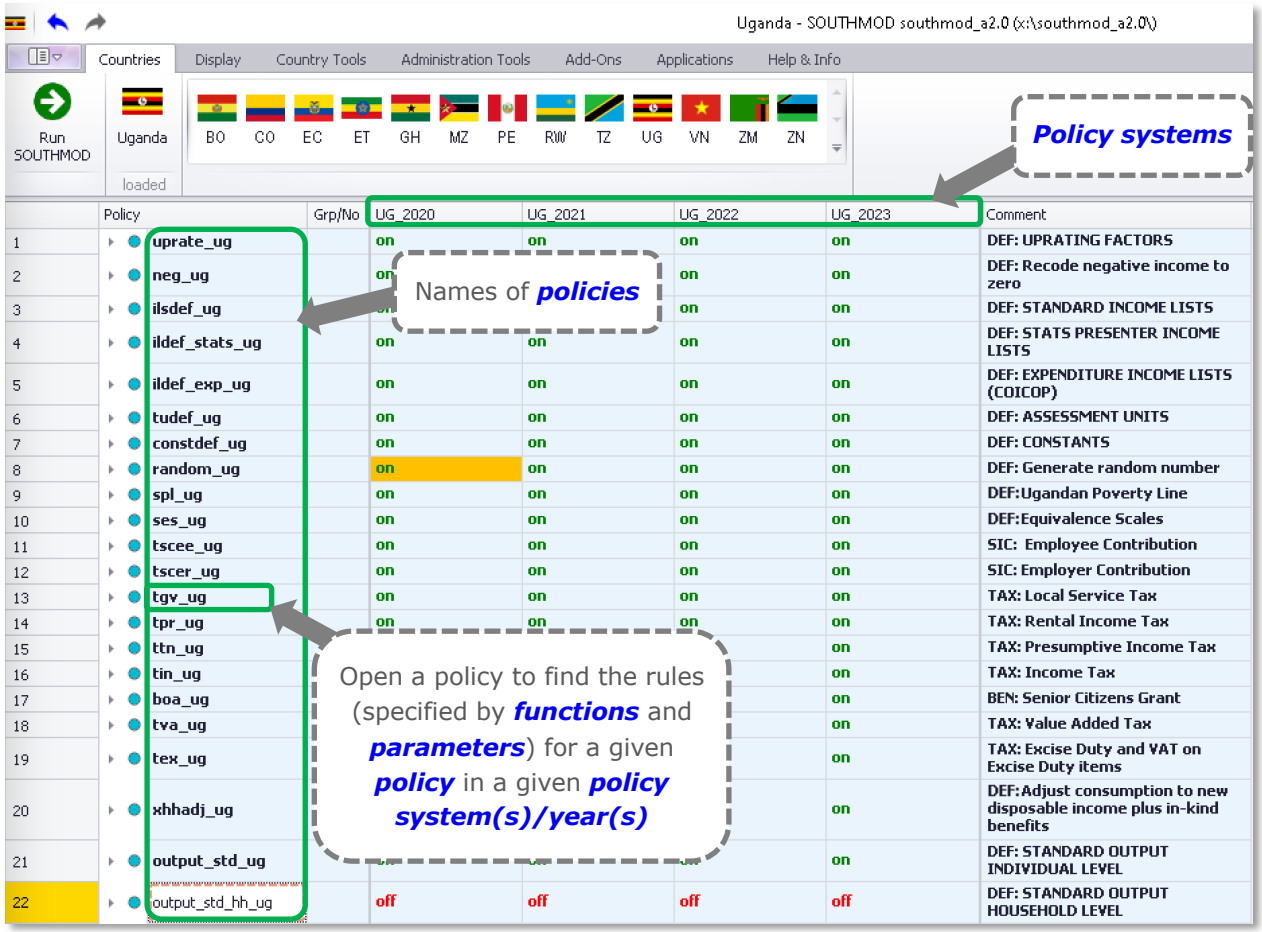
In order to access the model for a given country, click on the relevant country flag (e.g. Ugandan flag to open UGAMOD). This opens the associated model workspace.

The main part of the window displays the representation of the country's tax-benefit system, which when opened is in a 'collapsed' format. In EUROMOD terminology, this is frequently referred to as the ***policy spine***, or simply ***spine***.

As an example, the UGAMOD user interface showing the *spine*, with *policies* in collapsed form, is shown in Figure 2.4. Policies can be opened by double-clicking them, or from the triangle on the left. All user input for each model occurs via this user interface. Within the interface, information is mainly written into *Policies*, which are directly embedded in the *policy spine*. Some related definitions are provided below, with more information in Section 3:

- ***Policy Spine*** is the set of all policies in the given model, displayed all at once (across different rows) in a single workspace in the *EUROMOD* software.
- ***Policy systems*** *or policy years* reflect the policy rules generally as of 30 June or 1 July in a given year and are represented by each column (e.g. ***UG_2016*** and ***UG_2019*** in Figure 2.4).
- ***Policies*** may be related to either the general definitions and settings of the model or to particular tax-benefit policies that are modelled.
- ***Functions*** are the building blocks for implementing a country's tax-benefit system. Each *policy* is described by one or more *functions*.
- ***Parameters*** represent a particular element of the *policy's* functionality. Usually more than one *function,* each including a number of *parameters,* is used to calculate a tax or benefit. *Functions* and their *parameters* can also interact with each other.

Figure 2.4: UGAMOD interface after opening the UGAMOD model



	Policy	Grp/No	UG_2020	UG_2021	UG_2022	UG_2023	Comment
1	uprate_ug		on	on	on	on	DEF: UPRATING FACTORS
2	neg_ug		on	on	on	on	DEF: Recode negative income to zero
3	ilsdef_ug		on	on	on	on	DEF: STANDARD INCOME LISTS
4	ildef_stats_ug		on	on	on	on	DEF: STATS PRESENTER INCOME LISTS
5	ildef_exp_ug		on	on	on	on	DEF: EXPENDITURE INCOME LISTS (COICOP)
6	tudef_ug		on	on	on	on	DEF: ASSESSMENT UNITS
7	constdef_ug		on	on	on	on	DEF: CONSTANTS
8	random_ug		on	on	on	on	DEF: Generate random number
9	spl_ug		on	on	on	on	DEF:Ugandan Poverty Line
10	ses_ug		on	on	on	on	DEF:Equivalence Scales
11	tscee_ug		on	on	on	on	SIC: Employee Contribution
12	tscer_ug		on	on	on	on	SIC: Employer Contribution
13	tgv_ug		on	on	on	on	TAX: Local Service Tax
14	tpr_ug		on	on	on	on	TAX: Rental Income Tax
15	ttn_ug					on	TAX: Presumptive Income Tax
16	tin_ug					on	TAX: Income Tax
17	boa_ug					on	BEN: Senior Citizens Grant
18	tva_ug					on	TAX: Value Added Tax
19	tex_ug					on	TAX: Excise Duty and VAT on Excise Duty items
20	xhhadj_ug					on	DEF:Adjust consumption to new disposable income plus in-kind benefits
21	output_std_ug					on	DEF: STANDARD OUTPUT INDIVIDUAL LEVEL
22	output_std_hh_ug		off	off	off	off	DEF: STANDARD OUTPUT HOUSEHOLD LEVEL

2.4 What happens when a user runs a model

In practice, tax-benefit microsimulation models apply a set of policy rules (specified in the user interface) to household survey data (so-called ***input dataset***). When the user runs the model, via the ***Run SOUTHMOD*** button, it calculates individual and household-level entitlements to benefits and liabilities for taxes in each ***policy year***, and thus, enriches the input dataset with additional variables.

In the process, the model creates an ***output dataset*** that contains those simulated variables for each individual or household. This output dataset can be directly analysed via the model interface (***Applications*** → ***EUROMOD Statistics*** → ***Statistics Presenter***) or opened and examined using statistical software such as Excel, STATA, or any such programme.

Table 2.1 and Table 2.2 illustrate how the model applies a given policy rule to the input data, and adds a new, simulated variable to the output data, based on these rules. Table 2.1 shows selected information available in any ***input dataset*** for 3 households and their household members, including the relevant identifiers, age, formality status, and employment income.

Table 2.1: Example of model input data

Idhh	***idperson***	***dag***	***lfo***	***yem***
Household identifier	*Person identifier*	*Age*	*Formality status (1=formal worker, 0=informal or not relevant)*	*Gross employment income*
1	101	44	1	50,000
1	102	41	1	10,000
1	103	9	0	0
2	201	60	1	100,000
2	202	57	0	500
3	301	30	0	0

One policy in UGAMOD, for instance, calculates the annual amount of social insurance contributions paid by each worker. The policy rule is simple: formal workers (those with ***lfo=1***) pay 5% of their gross employment income (***yem***) to a relevant social insurance scheme.

After the user runs the UGAMOD model, variable ***tscee_s*** is added to the ***output data***, representing social insurance contributions paid by each employee. As can be seen in Table 2.2, this variable is created by applying the above-mentioned rules. Positive contributions are simulated for each *formal* employee with *positive gross employment income*, and contributions are always 5% of gross income.

Table 2.2: Example of model output data with one simulated variable

Idhh	***idperson***	***dag***	***lfo***	***yem***	***tscee_s***
Household identifier	*Person identifier*	*Age*	*Formality status (1=formal worker, 0=informal or not relevant)*	*Gross employment income*	*Employee social insurance contributions*
1	101	44	1	50,000	2,500
1	102	41	1	10,000	500
1	103	9	0	0	0
2	201	60	1	100,000	5,000
2	202	57	0	500	0
3	301	30	0	0	0

Similar calculations are applied – based on a range of variables in the input data – to simulate relevant entitlements to benefits and liabilities for taxes for all individuals and households.

3 Introduction to datasets, systems and policies

3.1 A single dataset is often used for several policy systems

Key definitions

The first step in microsimulation is to identify and compile appropriate secondary data on the incomes and expenditures of individuals in a representative survey of households. Such data should also contain information on personal characteristics and labour market characteristics of household members, as detailed in the SOUTHMOD Modelling Conventions. The second step is to compile a series of policy rules that can be applied to the individuals in the data to determine what benefits they are entitled to and what taxes they should pay.

In the model, there can be:

- One or more ***datasets*** (i.e. modified survey data collected in different years); and
- One or more ***systems*** (i.e. tax-benefit rules for different policy years).

Ideally, to calculate taxes and benefits for the year 2022, for example, you would use the 2022 policy rules together with data referring to the year 2022.

However, corresponding data is not always available and even if so, preparing and integrating new data in the model is a very laborious task. Therefore datasets are used for simulating several policy years, by uprating monetary values to the corresponding policy year. For each system, there is a dataset that is most suitable, normally the one whose collection year is nearest to the policy year (the 'best match'). Related definitions are listed in Table 3.1 based on the SOUTHMOD Modelling Conventions.

Table 3.1: Key definitions related to datasets and policy systems

Term	Description
Policy year / policy system	Tax and benefit policy rules as of 30 June or 1st July in a given year
Income / expenditure reference period	The time period to which the income and expenditure information in the input data refer
Base-year simulation	Refers to the case in which the year of policy rules matches the input data income reference period *(e.g. in Uganda, 2020 policy rules and Uganda National Household Survey data from 2019/20)* The purpose of base-year simulation is to provide information about the actual income distribution that is as accurate as possible
Target-year simulation	Refers to the case in which the year of policy rules does not match the input data income reference period *(e.g. 2022 policy rules and 2020 input data).* The purpose of target-year simulation is to provide accurate policy simulations for the policy year based on certain assumptions about growth in incomes and expenditures in the input data (uprating) rather than deriving the actual income distribution for the policy year
Baseline simulation	Refers to the best possible combination between years of policy rules and input data; see also 'Configuration: Databases' in Section 4.4

Examples from the SOUTHMOD bundle: Datasets in UGAMOD

As an example, UGAMOD, the tax-benefit microsimulation model for Uganda, is currently underpinned by two micro-datasets, and covers policy systems for years 2016–2023.

The input datasets are constructed using two waves of the Uganda National Household Survey (UNHS). The first dataset is from the 2016/17 UNHS, and relates to the 1 July 2016 timepoint, while the second dataset from the 2020/19 UNHS refers to the 1 July 2020 timepoint.

2016–19 policy systems are underpinned by the 2016/17 dataset, while 2020–23 policy systems are underpinned by the 2020/19 dataset.

The 2016/17 dataset can be used with the systems after 2016 because the ***uprate_ug*** policy uprates the monetary values in the input dataset (e.g. labour market income) to 2017, 2018, and 2019 using the Consumer Price Index (CPI). Similarly, the 2019/20 dataset is used for 2020 and also for the subsequent systems with the help of uprating monetary values.

3.2 Policies included in SOUTHMOD models

There are generally between 20 and 40 ***policies*** in the SOUTHMOD country models. These policies can be grouped into definitional and tax-benefit policies. The policies are processed by the EUROMOD software in the order they appear in the spine. While most policies and the ordering of policies are country-specific, there are also policies that are generally include in all models (see Table 3.2). Note that 'cc' at the end of each policy name in the table refers to a given country model, e.g. 'ug' for UGAMOD. The definition of a policy in the table (and each model) starts with either ***DEF*** for definitional, ***TAX*** for tax, ***BEN*** for benefit, or ***SIC*** for social insurance contributions.

In descriptive terms, the order of the policies in the spine is as follows:

1) **All adjustments to variables in the input data** are applied before modelling of taxes and benefits. This includes uprating monetary variables to the year to which the policy system refers, recoding all negative income values to zero, and specifying labour market transitions and shocks resulting from the COVID pandemic in 2020–21.
2) **Definitions or values** are specified for income concepts (defined as income lists), assessment units, model constants, poverty lines, and equivalence scales.
3) **Social insurance contributions, direct taxes and benefits** are modelled.
4) **Indirect taxes** are modelled.
5) **Consumption expenditures are adjusted** to the year to which the policy system refers based on absolute changes in disposable income.
6) **Finally, the results** are outputted to a single output dataset for each system.

The order in which the ***policies*** are simulated (and thus defined in the ***policy spine***) is crucial because some policies draw on variables produced in other policies and so these need to be created first.

Under the ***Display*** tab, you can choose whether to view the ***Full Spine*** or a ***Single Policy*** by checking the appropriate box.

Table 3.2: List of policies in SOUTHMOD models and relevant sections in the Manual

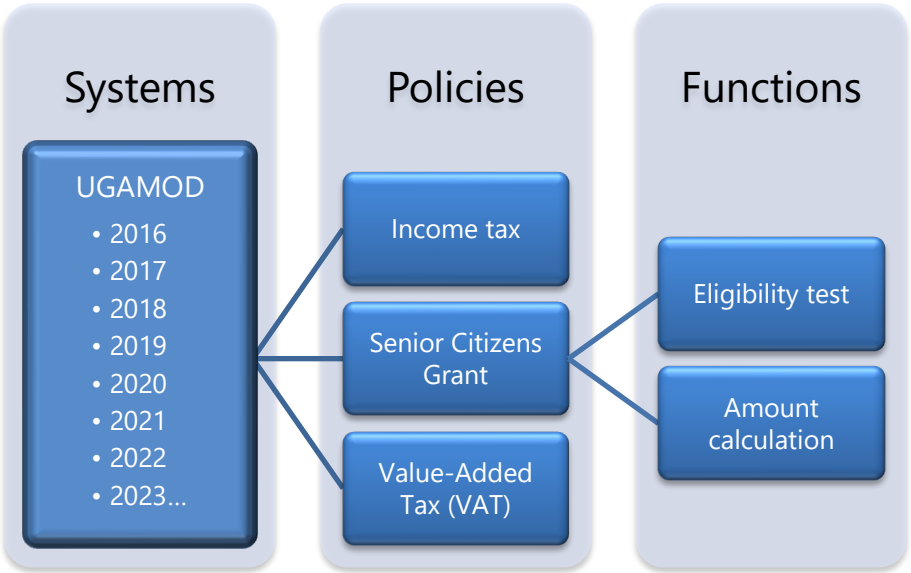
Name of policy	Type and purpose of the policy
Apply data adjustments (Section 6.1)	
uprate_cc	DEF: Uprating factors
neg_cc	DEF: Recode negative incomes to zero
lma_cc	DEF: COVID-19 transitions and shocks (may be ***on*** for systems 2020–21 when there are no input datasets covering those years)
Define income lists (Section 6.2)	
ilsdef_cc	DEF: Standard income lists
ildef_cc	DEF: Model-specific income lists
ildef_stats_cc	DEF: Statistics Presenter income lists
ildef_exp_cc	DEF: Expenditure income lists (COICOP)
Define assessment units, constants, poverty lines, equivalence scales (Section 6.3)	
tudef_cc	DEF: Define assessment units
constdef_cc	DEF: Define constants
spl_cc	DEF: Define poverty lines
ses_cc	DEF: Define equivalence scales
Model social insurance contributions, direct taxes and benefits (several policies, depending on the model) (Section 5)	
tsc*_cc	SIC: Employee and employer social insurance contributions (SICs)
t*_cc	TAX: Direct taxes
b*_cc	BEN: Benefits
Model indirect taxes (generally the two policies listed below) (Section 5)	
tva_cc	TAX: Value-added tax (VAT)
tex_cc	TAX: Excise duty and VAT on excise duty items
Uprate consumption based on changes in disposable income (Section 6.4)	
xhhadj_cc	DEF: Adjust consumption to new disposable income
Define variables included in the output datasets (Section 6.5)	
output_std_cc	DEF: Standard output, individual level
output_std_hh_cc	DEF: Standard output, household level (generally ***off*** for all systems)

3.3 Relationship between systems, policies and functions

All policies, whether definitional or tax-benefit policies, have the same general structure, shown in Figure 3.1 and summarized below:

- ***Systems*** contain groups of *policies*.
- ***Policies*** are constructed from *functions*.
- ***Functions*** (containing different ***parameters***) perform calculations to generate the model output, stored in the ***Output*** folder under the main folder for the SOUTHMOD bundle.

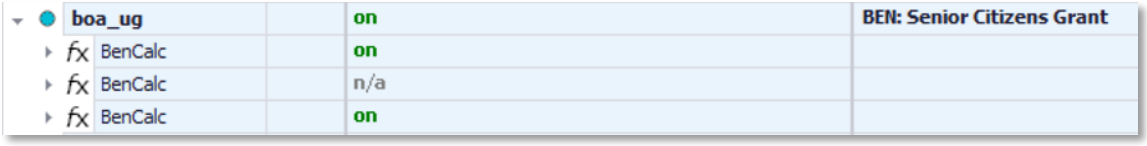
Figure 3.1: Relationship between systems, policies and functions



The structure is described below using the *Senior Citizens Grant* from UGAMOD as an example (using the simpler 2018 rules). This is an example of a fairly typical benefit policy that can be implemented in tax-benefit microsimulation models, including those in the SOUTHMOD model bundle.

This policy, ***boa_ug*** is shown in Figure 3.2 and models this grant. Additional information on tax-benefit policies are described in Section 5, and on definitional policies in Section 6.

Figure 3.2: Example of a policy: boa_ug in UGAMOD



The main functionalities of this policy – and policies in general – are described below:

- Each ***policy*** consists of:
    1. A *header* displaying the name of the *policy* (***boa_ug*** in the example); and
    2. A *switch* defining whether the *policy* is activated or not in each ***system*** (only *system* UG_2018 is shown above, where the policy is ***on***).
        - This facility to switch ***off*** a *policy* may, for example, be used if a reform scenario is implemented where the *policy* is not required.
        - The *policy* can also be switched to setting ***n/a***, which has the same functionality as switching the *policy* ***off***. It is generally used simply to reflect that the policy is not applicable in a given policy system. For instance, a policy that was introduced in 2022 would be ***n/a*** in 2021.

- Each ***policy*** is composed of ***functions****:*
    - You can view all the *functions* of the *policy* by right-clicking on the blue dot next to the *policy* name and selecting ***Expand All Functions***.

- You can also do this in a stepwise way by using the grey arrow heads next to the *policy* name and the functions within.
- It is also possible to expand all *policies* in the model at once by right-clicking on the word ***Policy*** and selecting ***Expand All Policies***.
- Like *policies*, *functions* can be set ***on*** or ***off*** (or ***n/a***). For instance, the second *BenCalc* function in ***boa_ug*** is set to ***n/a*** for the 2018 system year as it is not relevant for the 2018 policy year.

> In the ***boa_ug*** *policy* (Figure 3.3), there are three *functions* – each ***BenCalc*** – each of which determines eligibility for the benefit and assigns the relevant amount to each eligible person.
>
> The ***BenCalc*** *function* is one of the most frequently used *functions* in SOUTHMOD models, including *UGAMOD*. More complex *policies* may contain a number of different *functions*.

- Each ***function*** is a self-contained building block that has its own ***parameters*** and represents a particular operation of the *policy's* functionality.
  - Each *function* carries out a specific operation based on the specified *parameters* within the function. For example, a monthly benefit for eligible individuals requires specifying the eligibility conditions (e.g. age bracket) and benefit amounts inside the relevant function.
  - The purpose of using *functions* as building blocks of the model is to provide a general structure and standardized language to describe policy instruments.

> In the ***boa_ug*** *policy* (Figure 3.3) a BenCalc function is used to define individuals that are eligible for the benefit and to assign the benefit amount to those eligible. Eligibility for the benefit is based on district *(**drgn2**)* and age *(**dag**)*, as shown in the ***Comp_Cond*** parameter under the first ***BenCalc*** function.
>
> Benefit amount is set in ***Comp_perTU*** parameter, specified as '***$senior_grant_amount'***, a constant defined in the ***constdef_ug*** policy.

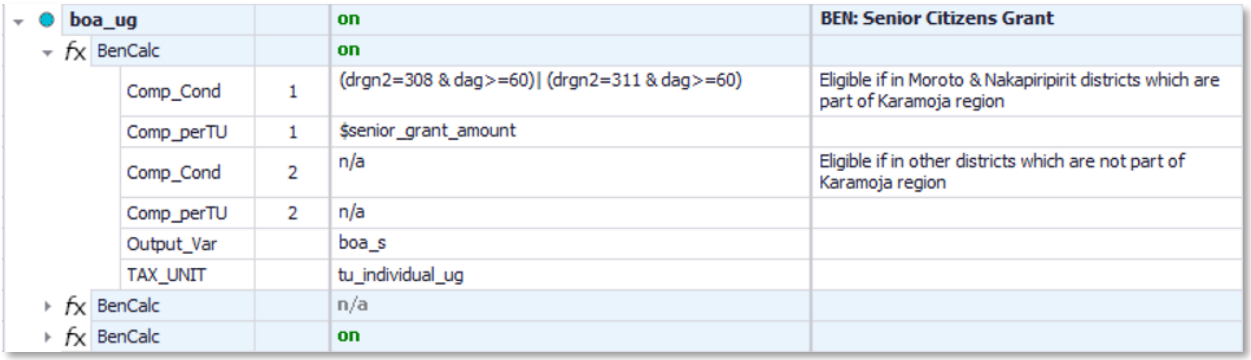
- ***Functions*** can be classified in three categories:
  1. Policy functions for implementing tax-benefit policies (including e.g. ***ArithOp***, ***BenCalc*** and ***SchedCalc***);
  2. System functions for implementing the general definitions and settings of the model (including e.g. ***Uprate***, ***DefVar***, ***DefIL***, ***DefTU***, ***DefConst*** and ***DefOutput***); and
  3. Special functions, discussed later in the User Manual.

- A summary of all the *functions* and their *parameters* available for use can be found in the ***EUROMOD Functions*** section of the *EUROMOD Help* (accessed from the ***Help & Info*** tab in the main user interface).

- ***Functions*** include both *compulsory* and *optional* ***parameters***. Many parameters appear within multiple functions, while some are specific to particular functions.
  - For example, the *parameter* ***TAX_UNIT*** must be included in all policy *functions*, as otherwise UGAMOD will issue an error message.
  - *Parameter values* can take a number of different forms, for example yes/no (1/0) values, amounts, variables and formulas.
  - The *parameter values* may change for each *system*, for instance if tax rates, benefit amounts, or eligibility conditions change over time.

- Figure 3.3 shows ***boa_ug*** with the first ***BenCalc*** *function* expanded.
  - Each *policy* is named in the final column, ***Comment***.
  - The name starts with either ***DEF*** for definitional, ***TAX*** for tax, ***BEN*** for benefit (as with this policy), or ***SIC*** for social insurance contributions.
  - Each *function* and *parameter* can also be described if necessary.

Figure 3.3: Example of a function: boa_ug in UGAMOD with the first function expanded



boa_ug		**on**	**BEN: Senior Citizens Grant**
fx BenCalc		**on**	
Comp_Cond	1	(drgn2=308 & dag>=60)| (drgn2=311 & dag>=60)	Eligible if in Moroto & Nakapiripirit districts which are part of Karamoja region
Comp_perTU	1	$senior_grant_amount	
Comp_Cond	2	n/a	Eligible if in other districts which are not part of Karamoja region
Comp_perTU	2	n/a	
Output_Var		boa_s	
TAX_UNIT		tu_individual_ug	
fx BenCalc		n/a	
fx BenCalc		**on**	

4 Menus, settings and actions

4.1 EUROMOD file menu: Opening a new project and saving changes

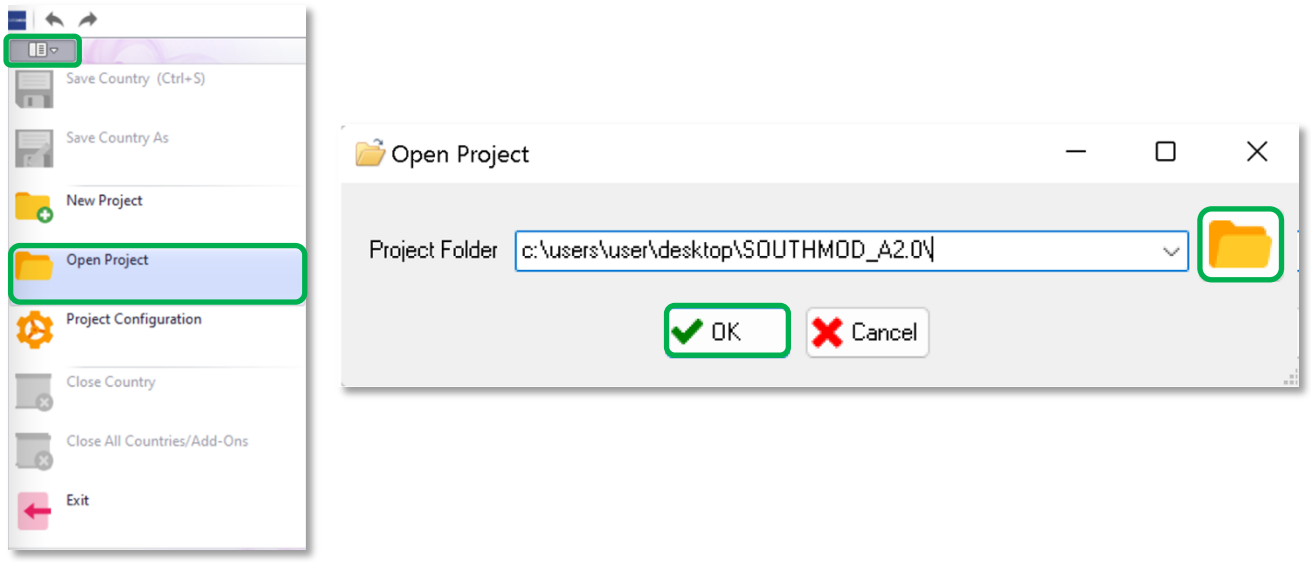
As discussed, the main menu has an option to ***Open Project***. Clicking on it opens a dialog where you can specify a file path to a project – such as the SOUTHMOD model bundle – to be opened in the EUROMOD software (see Figure 4.1).

Opening a new project is required:

1. When opening the SOUTHMOD model bundle in EUROMOD software, or any model, for the first time (after doing this once, the previous model is opened automatically upon opening EUROMOD);
2. If a new version of the SOUTHMOD bundle is issued and you wish to switch to using the new version.

In the dialog, you are asked to provide the location of the model folder (e.g. ***C:\Desktop\SOUTHMOD_A2.0***).

Figure 4.1: Opening a new project



To save your changes, open the main menu (above the ***Run SOUTHMOD*** button) and select the menu item ***Save Country***. Alternatively press ***Ctrl-S***.

New Project, not covered here in detail, may be used when developing an entirely new model.

Project Configuration in the file menu offers general options for configuring the opened project, such as default locations of the folders for input and output data, enabling autosave, and showing warnings.

4.2 EUROMOD menu tabs: Main contents

Figure 4.2 shows the EUROMOD menu tabs, each of which opens expands to reveal a ribbon menu with specified actions. These tabs and related actions are listed in Table 4.1.

Figure 4.2: EUROMOD menu tabs

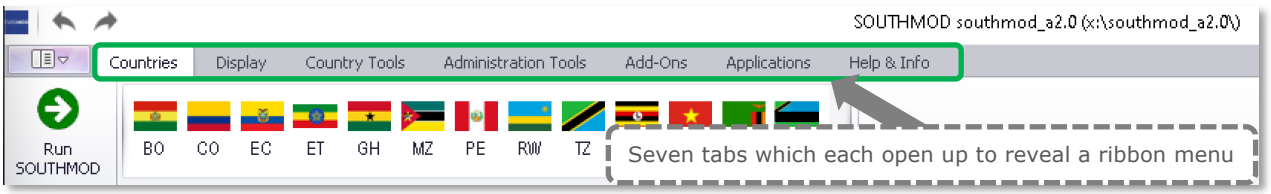


Table 4.1: EUROMOD menu tabs, main contents and relevant sections in the Manual

Menu tab	Notable contents in the ribbon menu	More information in the User Manual
Countries	• Opening a country model • Running a model	Section 2 on 'Getting started' Section 8 on 'Modelling in practice'
Display	• Highlighting policies/functions with colors • Setting bookmarks • Showing the whole spine versus a selected policy only	Section 4.3
Country Tools	• Configuring ***Country*** settings such as long and short names • Configuring ***System*** settings such as currencies used • Configuring input datasets (***Databases***) that can be used for simulating a country's tax-benefit system • Adding or deleting systems • Configuring extension switches • Search functionalities	Section 4.4
Administration Tools	• Managing variables for all country models • Setting exchange rates for all country models	Section 4.5 Section 7 on 'Variables'
Add-Ons	• *Not used with the SOUTHMOD bundle*	-
Applications	• Accessing ***Statistics Presenter*** to summarize model output, along with other applications	Section 4.6 Section 9 on 'Using the Statistics Presenter'
Help & Info	• ***Help*** library for EUROMOD software • Software ***version*** • Software ***licence***	-

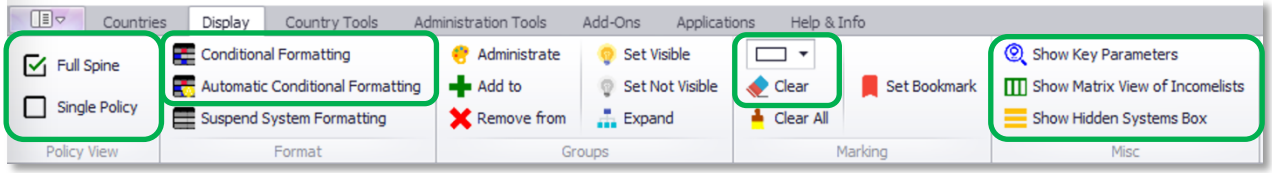
The following sections provide additional details on the most important actions, focusing on the ***Display***, ***Country Tools***, ***Administration Tools*** and ***Applications*** tabs.

4.3 EUROMOD menu tab: 'Display'

The ribbon the ***Display tab*** (Figure 4.3) allows for the following actions:

- **Policy View:** Check either box to view the ***Full Spine*** or a ***Single Policy***.
- **Format:**
  - Click on ***Conditional Formatting*** to specify how differences between a *base* and a derived *system* are highlighted using background and/or text colours. The default setting highlights changes in a policy system from the system in the previous year in yellow.
  - Generally, it is advised to click on the ***Automatic Conditional Format*** button to apply the default settings.
  - Conditional formatting is very useful for highlighting the changes introduced by a given reform. By selecting "expand differences", all functions with differences between a system and its specified base year are expanded.
- **Groups:** Generally not used with SOUTHMOD models; see EUROMOD ***Help*** for details.
- **Marking:** Click on a policy or function and then select a highlight color, or clear it. For instance, different policies in progress may be highlighted whilst working on a model. Different income lists in SOUTHMOD models (see Section 6.2) are also highlighted by default. Bookmarks are generally not used with SOUTHMOD models; see EUROMOD ***Help*** for details.
- **Misc:**
  - Click ***Show Key Parameters*** to show notable parameters (e.g. output variables) below the main rows of functions.
  - Click ***Show Matrix View of Incomelists*** to show the income lists in which different variables are included.
  - Click ***Show Hidden Systems Box*** to list and if needed reveal any hidden policy systems.

Figure 4.3: Ribbon in the 'Display' tab, with key actions highlighted

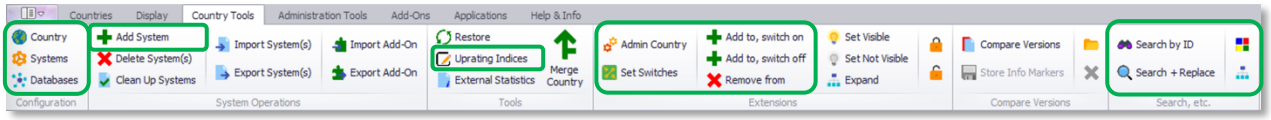


4.4 EUROMOD menu tab: 'Country Tools'

The ribbon for the ***Country Tools*** tab (Figure 4.4) includes the following subsections, with the main functionalities for SOUTHMOD users explained in the subsections below.

- **Configuration:** The configuration dialogs ***Country***, ***Systems*** and ***Databases*** are explained in the following subsections (see below).
- **System Operations:** Click ***Add System*** to add a new policy system based on an existing system (this can also be achieved by right-clicking any system header and clicking ***Copy-Paste System***). Other actions are generally not relevant for SOUTHMOD models; see EUROMOD ***Help*** for details.
- **Tools:** The configuration dialog for ***Uprating Indices*** is explained below.
- **Extensions:** Switches are explained below.
- **Compare Versions:** Generally not relevant for SOUTHMOD models; see EUROMOD ***Help*** for details.
- **Search, etc.:** Main search tools are explained below.

Figure 4.4: Ribbon in the 'Country Tools' tab, with key actions highlighted

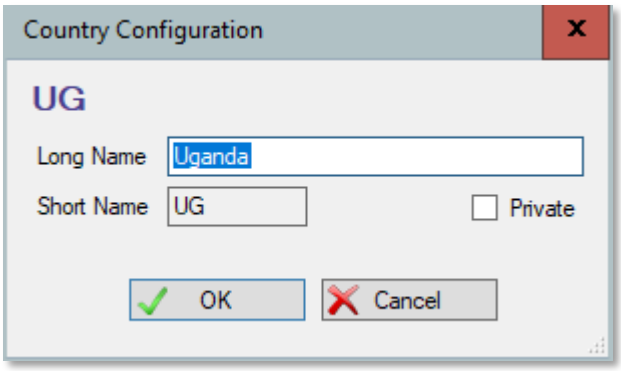


First, there are three buttons to the left hand side of the ***Country Tools*** ribbon: ***Country***, ***Systems*** and ***Databases***, discussed below in detail.

Configuration: Country settings

In the ***Country configuration*** dialog (Figure 4.5), the *Long Name* of the country can be changed. The *Short Name* of the country cannot however be changed by the user, as it needs to correspond to the filenames of the country's XML files (model rules).

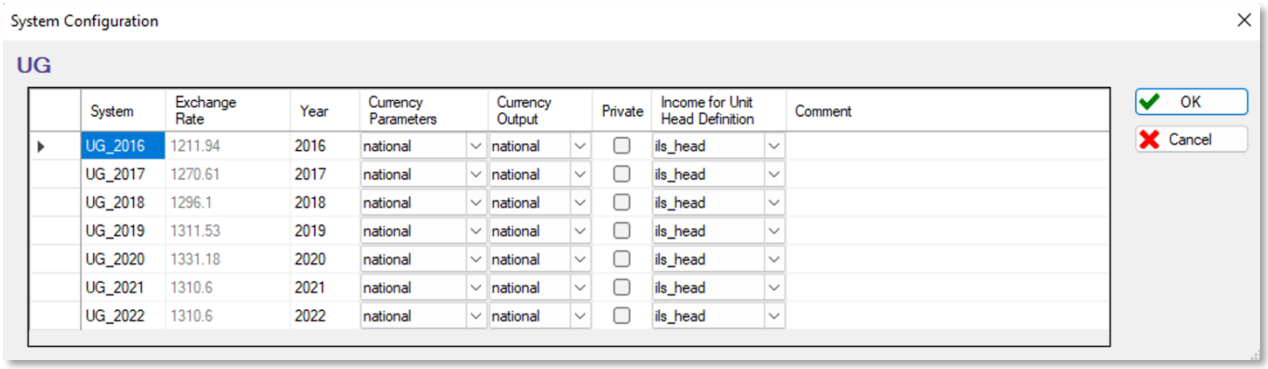
Figure 4.5: The 'Country Configuration' dialog: Example from UGAMOD



Configuration: Systems

The ***System Configuration*** dialog (Figure 4.6) lists all systems and allows for changing their settings, discussed below.

Figure 4.6: The 'System Configuration' dialog: Example from UGAMOD



The settings include:

- **Exchange Rate** indicates the rates used by the model to convert national currency into international dollars or vice versa across different systems (i.e. dollar amounts are multiplied by the rate to obtain amounts in national currency). See Section 4.5 on editing exchange rates. (Note that the selection for 'euro' refers to 'dollar' in SOUTHMOD.)
- **Currency Parameters** indicate the currency used for monetary ***parameter*** values. It is set to national in all models in the SOUTHMOD bundle, which means the national currency is used. The alternative in the SOUTHMOD bundle would be dollar. This setting should not be changed unless monetary amounts in the model are adjusted too.
- **Currency Output** indicates the currency used within the output file (e.g. UG_2022_std.txt for the 2022 policy year in UGAMOD), which again is set to national. The alternative in the SOUTHMOD bundle would be international dollar (using purchasing power parity conversion factors to take differences in costs of living between the countries into account, see World Bank's World Development Indicators database).
- **Income for Unit Head Definition** indicates which ***income list*** (from the dropdown menu) is to be used as a default for determining the head of the assessment unit. This may be set as *ils_head* (which refers to the head of the household as specified in the original survey data) or specified based on original income.

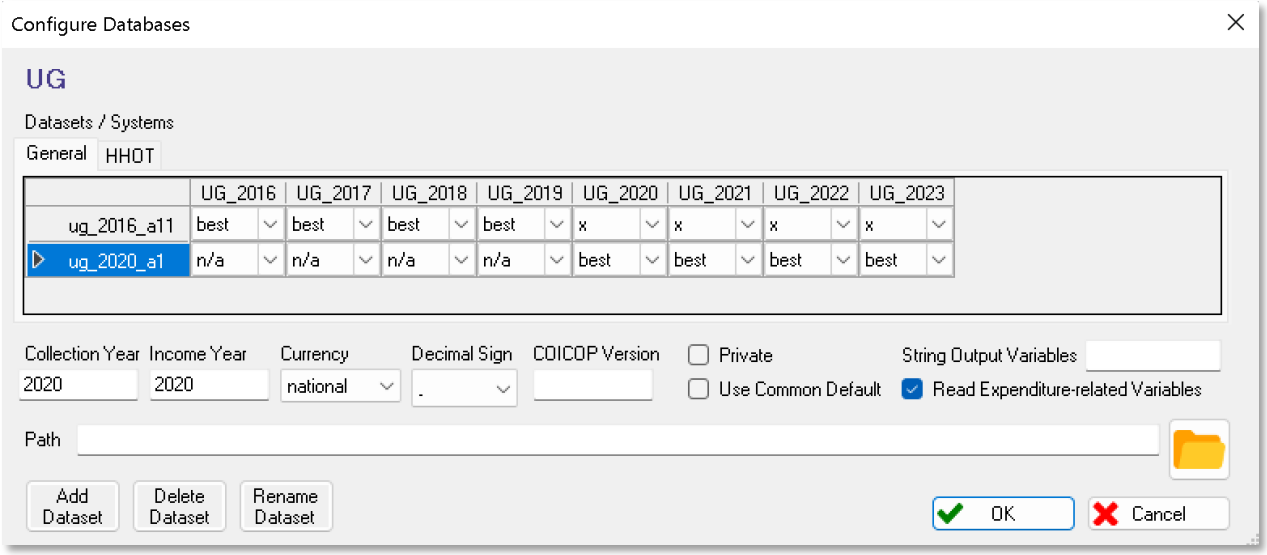
Configuration: Databases

Figure 4.7 shows the ***Configure Databases*** dialog. The dialog allows for:

1. Assigning databases to systems;
2. Adding, removing or renaming a dataset; and
3. Adjusting the settings for different datasets.

These options are discussed in greater detail below.

Figure 4.7: The 'Configure Databases' dialog: Example from UGAMOD



1. Assigning databases to systems

The upper part of the *Configure Databases* dialog shows a table where the row headers list all datasets available for the country, while the column headers list all available ***systems***. The intersection of a dataset (row) and system (column) indicates whether the system can be run with the dataset, in other words whether they form a so-called system-dataset combination

There are three possible settings:

- **A cross (x)** indicates that the dataset and the system form a system-dataset combination;
- **Best** denotes a system-dataset combination that is the 'best match'; and
- **n/a** means that the system cannot be run with the dataset.

2. Adding, removing or renaming a dataset

The following actions can be undertaken:

- **To add a dataset**, click on the ***Add Dataset*** button. This opens a file search dialog allowing for the search of a text file that contains data suitable to run one or more of the ***systems***.
- **To delete a dataset**, select it and click on the ***Delete Dataset*** button. Note that the dataset is removed without any further warning, but you can still undo this action by closing the dialog with the ***Cancel*** button or by using the undo functionality. The dataset is not deleted physically; the removal concerns only the ability to use the dataset within the relevant SOUTHMOD model.
- **To rename a dataset**, select it and click on the ***Rename Dataset*** button. This opens a textbox *where* the new name can be entered.

3. Settings of a selected dataset

Below the 'Datasets / Systems' table, the dialog shows the settings of the selected dataset:

- **Collection Year** indicates the year the data were collected.
- **Income Year** indicates the year to which the monetary values within the data refer.
- **Currency** indicates the currency in which data are stored. This is set to national in SOUTHMOD models.
- **Decimal Sign** indicates whether data use point (.) or comma (,) as the decimal sign. Generally a point is used in SOUTHMOD models.
- **Path** indicates a specific path to locate the dataset. Usually it is left empty, required to instruct the model to locate the dataset at the default path.
- **Use Common Default** can be checked, meaning that any variable not in the data but used by the system is set to zero (i.e. no error message is issued).
- **Read Expenditure-related Variables** is an important functionality developed for SOUTHMOD models, whose input datasets typically contain detailed expenditure-related variables (for instance, household expenditure on bread, or litres of gas purchased by a household). Checking this option will cause all such variables to be automatically imported from the input datasets and used within the model. The definitions of expenditure-related variables are as follows:
    - *Expenditure variables:* all variables whose name starts with "x" followed by numbers only; imported as monetary variables.
    - *Quantity variables:* all variables whose name starts with "q" followed by numbers only; imported as non-monetary variables.
    - *Price variables:* all variables whose name starts with "p" followed by numbers only; imported as monetary variables.

With all of these buttons, click ***OK*** to confirm any changes or ***Cancel*** to close the dialog without any consequences. Changes are only definite once the project is saved. Before that, you can still use the undo functionality or close the project without saving (see ***Working with EUROMOD – Undo and redo*** section of the EUROMOD ***Help*** from the ***Help & Info*** tab).

For further information see the ***Working with EUROMOD – Changing Countries*** Settings section of the EUROMOD ***Help***.

Tools: Uprating Indices

Datasets are usually used for simulating several policy years, by uprating monetary values in the survey data to the corresponding system year. The ***policy uprate_cc*** (see Section 6 for details) contains information for uprating monetary variables in the dataset – specifically, which uprating factors are applied to which variables.

The numerical values of the uprating factors are specified in the ***Uprating Indices*** dialog called from the ***Country Tools*** tab (see Figure 4.8; note that in some computers you may only see the notebook icon). This resulting dialog, shown in Figure 4.9, defines the uprating factors, gives a reference name, and describes the source. Additional indices, for example a

wage inflator, can be added as additional rows. New years of data can be added by pressing the ***Add Year*** button, which will result in a new column being added to the table.

Figure 4.8: 'Uprating Indices' in the 'Country Tools' tab

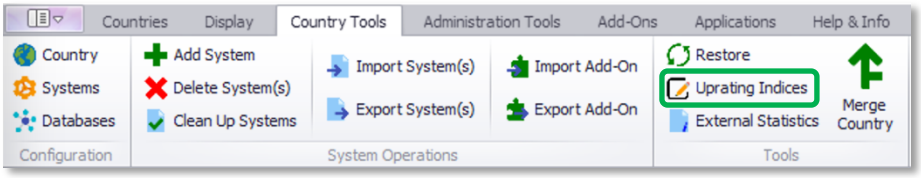
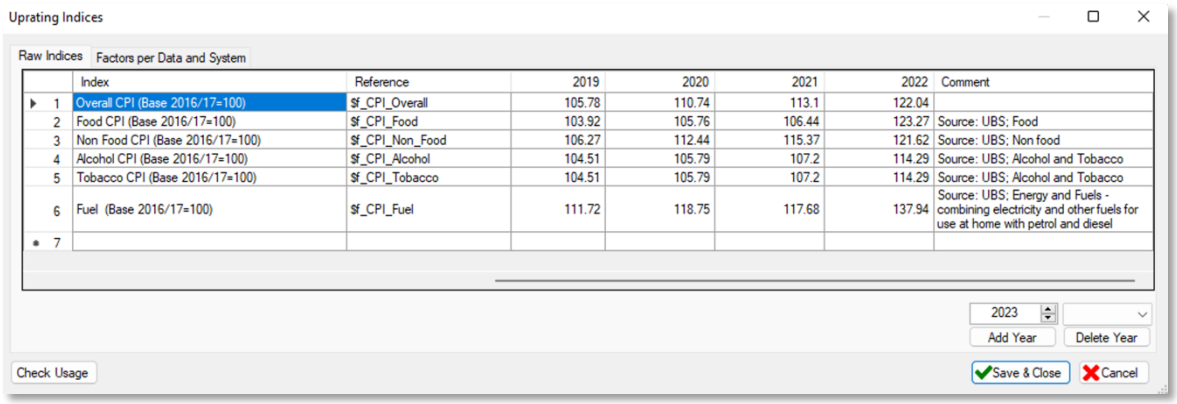


Figure 4.9: The 'Uprating Indices' dialog: Example from UGAMOD



Most SOUTHMOD models uprate income and expenditure values using the overall Consumer Price Index (CPI) and apply specific CPIs to relevant consumption expenditures (such as Food CPI to food items in UGAMOD).

The table shows the annual raw indices. To see the actual uprating factors that are applied to the monetary variables in the input dataset (the change from the data year to the policy year), go to the second tab called "Factors per Data and System" and select a dataset.

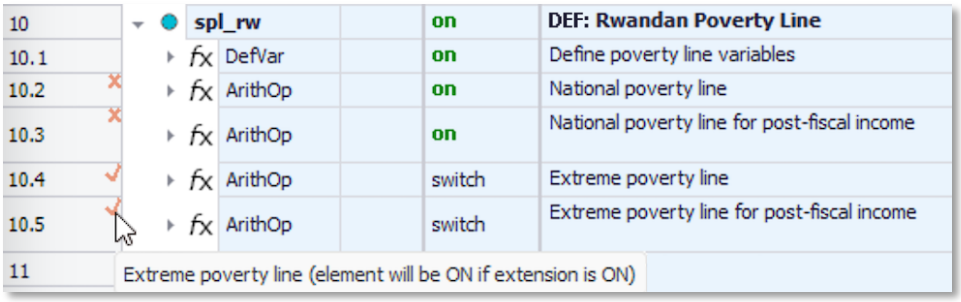
Extensions: Switches

Extensions or ***switches*** allow for various additions to the default ***system*** for a given ***policy year***. For instance, in most SOUTHMOD models the user can use switches to choose whether a standard or an alternative poverty line is used to determine whether individuals or households are defined as poor. In that way, any policy system can be run using either poverty line without requiring any further modifications, and the user is not required to have two separate systems and/or policies.

Figure 4.10 below shows an example of the 'Extreme poverty line' (POV) extension in the Rwandan model. The standard national poverty line is used when the POV extension is ***off*** (the default), while the extreme poverty line is used when the POV extension is ***on***.

The extension is applied to the relevant *functions* in the poverty line *policy* (spl_rw) in the model. It is illustrated by the crosses and check marks next to the relevant row numbers (10.2–10.5) as well as the word ***switch*** on rows 10.4 and 10.5, indicating that the functions in those rows will be ***on*** (and extreme poverty line used) when the extension switch is ***on***. This is also shown by EUROMOD when hovering over one of the check marks.

Figure 4.10: Switches in the poverty line policy: Example from RWAMOD



In practice, EUROMOD allows for an extension switch to be attached to a given policy, function or parameter, or several of them at once. Extensions can be switched ***on*** or ***off*** in the baseline (i.e. as a default option when running a given model). For example, an extension that allows users to account for informality when simulating social insurance contributions can be switched ***on*** in the baseline (i.e. to assume that informal workers do not pay social contributions) but can be switched ***off*** by users (i.e. simulating contributions also for those in the informal sector).

Extensions can carry out two actions. They can:

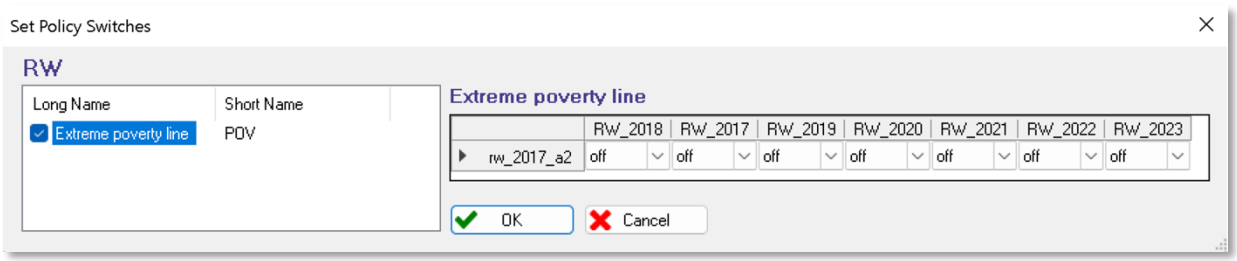
1. switch ***off*** policies, functions or parameters that are ***on*** in the baseline; or
2. switch ***on*** policies, functions or parameters that are ***off*** in the baseline (as in Figure 4.10).

Both actions can be carried out by the same extension as it can be specified for each policy, function or parameter separately (i.e. switching an extension ***on*** can switch one function ***off*** but another one ***on***).

The actions available in the ***Extensions*** menu in the ***Country Tools*** tab include the following:

- ***Admin Country*** allows for adding (and naming) or deleting an extension, along with setting a colour to be used with the extension.
- ***Set Switches*** (see Figure 4.11) allows for specifying default settings for the extensions for different policy systems (***on***, ***off***, or ***n/a***). When the default is ***on***, the relevant extension is ***on*** when the model is run. When the default is ***off***, the relevant extension is ***off*** when the model is run, as is the case with the prior example from Rwanda. ***n/a*** signals that the extension is not available for the specific policy system.

Figure 4.11: The 'Set Policy Switches' dialog: Example from RWAMOD



- ***Add to, switch on/off*** can be used by selecting any policy or function, clicking the button, and selecting the extension that is used with the policy or function. When the former is used, the relevant policy or function is set ***on*** when the extension is ***on;*** when the latter is used, the policy or function is set ***off*** when the extension is ***on***.
- ***Remove from*** removes an extension from a policy or function. It can be used by selecting any policy or function with an existing extension attached to it, clicking the button, and selecting the relevant extension to be removed.
- ***Set Visible and Set Not Visible*** allow for either showing or not showing policies or functions to which extensions are applied. This only applies to policies/functions/parameters that are only ***on*** when the extension is switched ***on***.
- ***Expand*** allows for expanding policies and functions that are part of the selected extension.

Note that users can select whether to run the tax-benefit simulations with the extension being ***on*** or ***off*** in the Run SOUTHMOD dialog, as follows:

Countries tab → ***Run SOUTHMOD*** → ***View / Filter / Add-ons*** → ***Extensions*** → Check relevant extensions to be included in the main run dialog → ***Main*** → Set relevant extensions ***on***, ***off***, or ***all*** for relevant systems. Selecting ***all*** produces output datasets with both ***on*** and ***off*** settings for the relevant switch.

See the SOUTHMOD Modelling Conventions for standardized policy extensions and relevant setting used in SOUTHMOD models (including e.g. switches for poverty lines and tax non-compliance).

Search options

The main functionalities of the ***Search, etc.*** menu in the ***Countries*** tab include the following:

- ***Search by ID*** allows for searching by an identifier. Identifiers are combinations of letters and numbers identified in the main dialog when running a policy system from a given model.
- ***Search + Replace*** allows for standard searching functionalities across the user interface, including system, policy and comment columns.

4.5 EUROMOD menu tab: 'Administration Tools'

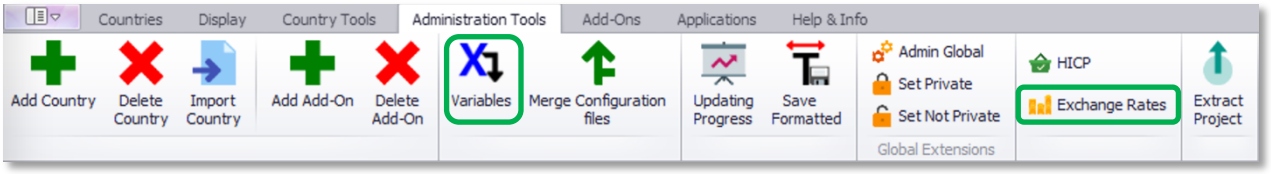
The main actions in the ribbon for the ***Administration tab*** (Figure 4.12) include:

- ***Variables:*** Clicking on ***Variables*** opens up the *variable manager*. It contains information on all variables and their properties stored in the ***variable description file*** *(**VarConfig.xml**)* and allows for a number of operations, such as adding, deleting and searching for variables. Note that any changes apply to all country models included in the SOUTHMOD model bundle. Section 7 offers a detailed description of the variable manager.
- ***Exchange Rates:*** Clicking on ***Exchange Rates*** opens the exchange rate configuration dialog. For each country in the SOUTHMOD model bundle, exchange

rates are defined under the ***Year Average*** column for all available systems as local currency units per international dollar (using purchasing power parity conversion factors to take differences in costs of living between the countries into account, see World Bank's World Development Indicators database). Exchange rates are defined for each country included in the SOUTHMOD model bundle and each policy system included within each country model.

- Other functionalities are not generally required by standard SOUTHMOD model users; see EUROMOD ***Help*** for details.

Figure 4.12: Ribbon in the 'Administration Tools' tab, with key actions highlighted

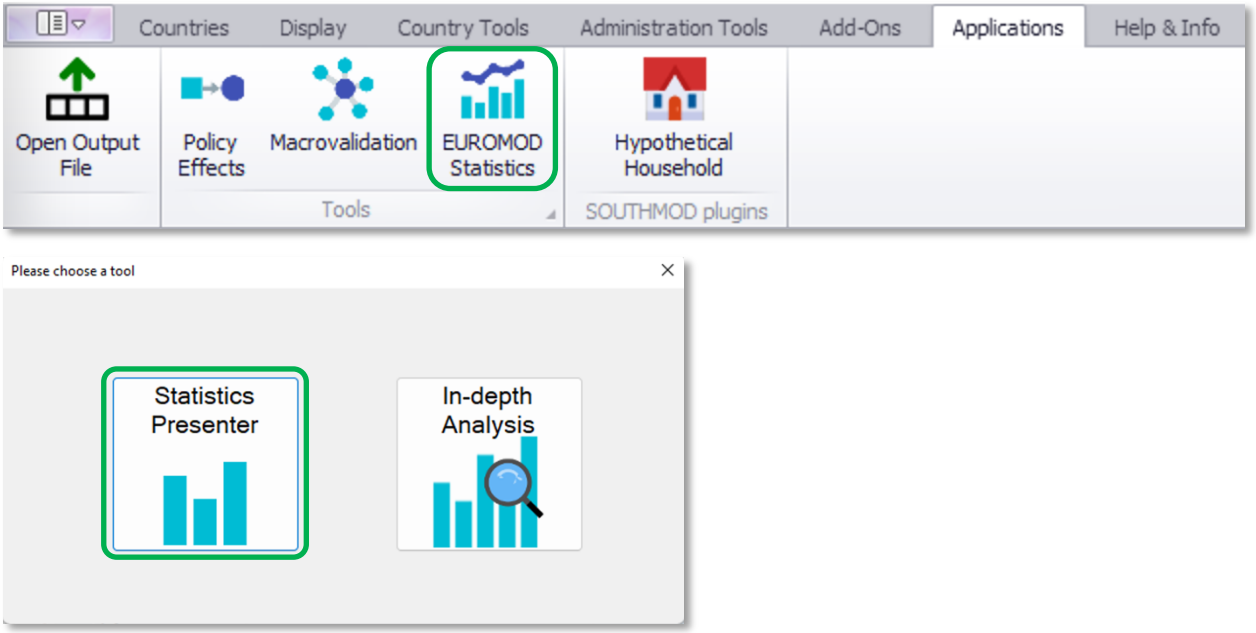


4.6 EUROMOD menu tab: 'Applications'

The main functionality in the ribbon for the ***Applications tab*** (Figure 4.13) includes:

- ***EUROMOD Statistics***, which opens a dialog with access to the ***Statistics Presenter***. This tool is used to derive aggregate statistics from model output datasets, including total government taxes and revenues, poverty measures, and inequality measures, as well as comparisons of these outcomes between different systems. Section 9 offers details on using the Statistics Presenter.
- Other functionalities are not generally required by standard SOUTHMOD model users; see EUROMOD ***Help*** for details.

Figure 4.13: Ribbon in the 'Applications' tab, with EUROMOD Statistics highlighted



5 Tax-benefit policies and their modelling

The country models included in the SOUTHMOD bundle include a large array of tax and benefit policies in place in those countries. The most common policies in SOUTHMOD models include:

- Employee and employer social insurance contributions (SICs);
- Labour income tax (a direct tax); and
- Value-added tax (an indirect tax).

Additionally, several models simulate presumptive or turnover tax, excise tax as well as a number of means-tested social cash transfers, pension or senior citizen benefits, and school feeding programmes (where the value of food rations is provided in monetary terms). Additional policies simulated in particular country models include rental income tax, local service tax, capital income tax, electricity subsidies along with COVID-related tax measures, utility tariff reliefs and social protection benefits.

The rest of this section offers details on how such policies can be implemented in the EUROMOD software, offering insights into the use of functions (5.1) and parameters (5.2).

Tax-benefit ***policies*** are illustrated below using the **PSSN Fixed Basic Cash Transfer** *(**bsa_tz**)* from the TAZMOD model as an example.

This policy is shown in Figure 5.1 in the section Figure 5.1: Example of a benefit policy: PSSN Fixed Basic Cash Transfer from TAZMOD, with rules shown only for the ***policy year*** 2018 as an illustration. The benefit amounts used are defined as constants in the ***constdef_tz*** policy, shown in Figure 5.2.

References to this policy below are shown in **boxes** such as this one.

5.1 Functions

This section on functions discusses key functions and interactions between functions, including an example of benefit policy from TAZMOD, the microsimulation model for Tanzania.

Key functions and an example from TAZMOD

Tax and benefit ***policies*** are modelled using one or more ***functions***. Examples of some functions are given below.

- ***Elig*** determines *eligibility* or *liability* for benefits and taxes.
- ***BenCalc*** calculates the benefit or tax *amount* for all *eligible* units.
- ***ArithOp*** is a simple calculator, allowing for the most common arithmetical operations.
- ***SchedCalc*** allows for the implementation of the most common (often progressive) tax schedules.

- ***Allocate*** allows reallocating amounts (incomes, benefits, taxes) between members of assessment units.
- ***Min*** *and* ***Max*** are simple minimum and maximum calculators.

Functions are described in detail in the EUROMOD Functions section of the EUROMOD **Help** (accessed from the ***Help & Info*** tab) and they are not elaborated in great detail here.

Many of the social benefit policies in the SOUTHMOD bundle follow the same general structure, using the ***function BenCalc*** to calculate eligibility for and amount of the grant (including the cash transfer from TAZMOD shown in Figure 5.1).

Alternatively, the same results can be achieved by using the ***function Elig*** to calculate eligibility for the grant and then ***function ArithOp*** to calculate the amount of grant for eligible individuals.

Figure 5.1: Example of a benefit policy: PSSN Fixed Basic Cash Transfer from TAZMOD

	Policy	Grp/No	TZ_2022	TZ_2023	Comment
18	**bsa_tz**		on	on	**BEN: PSSN Fixed Basic Cash Transfer**
18.1	BenCalc		on	on	
18.1.1	Comp_Cond	1	dfp=1 & dag>=0 & dag<=17	dfp=1 & dag>=0 & dag<=17	capture infants and children
18.1.2	Comp_perTU	1	$fbct_child_amnt	$fbct_child_amnt	
18.1.3	Comp_Cond	2	dfp=1 & dag>=18	dfp=1 & dag>=18	capture adults
18.1.4	Comp_perTU	2	$fbct_adult_amnt	$fbct_adult_amnt	
18.1.5	Output_Var		bsa_s	bsa_s	Total bsa amount for adult and child
18.1.6	TAX_UNIT		tu_household_tz	tu_household_tz	

Figure 5.2: Relevant constants for the PSSN Fixed Basic Cash Transfer from TAZMOD

	Policy	Grp/No	TZ_2022	TZ_2023	Comment
9	**constdef_tz**		on	on	**DEF: CONSTANTS**
9.1	DefConst		on	on	Define constants
9.1.1	$presumptive_upper_limit		100000000#y	100000000#y	Upper limit for presumptive tax
9.1.2	$fbct_child_amnt		5000#m	5000#m	FBCT Child Amount
9.1.3	$fbct_adult_amnt		12000#m	12000#m	FBCT Adult amount

In the ***bsa_tz*** *policy* in Figure 5.1, there is one ***BenCalc*** *function.* It determines *eligibility* for the benefit and assigns the relevant *amounts* to each eligible person.

The parameter ***Comp_Cond*** (*Component Condition*) is the eligibility condition that has to be fulfilled before an amount specified in ***Comp_perTU*** can be allocated.

In ***bsa_tz***, there are two of these parameters each, connected by group numbers 1 and 2 on the right to the parameter names:

- **The first two parameters (group 1)** determine the eligibility condition and benefit amount for infants and children: The first ***Comp_Cond*** specifies that those in food-poor households *(**dfp=1**)* who are between 0 and 17 of age *(**dag**)* are eligible to receive the benefit specified by constant ***$fbct_child_amnt*** (4,000 per month), specified in the constants policy, ***constdef_tz*** (Figure 5.2).
- **The latter two parameters (group 2)** determine the eligibility condition and benefit amount for adults: The second ***Comp_Cond*** specifies that people in food-poor households *(**dfp=1**)* who are at least 18 years of age *(**dag**)* are eligible to receive the benefit specified by constant ***$fbct_adult_amnt*** (10,000 per month), again specified in the constants policy, ***constdef_tz*** (Figure 5.2).

The calculations of this ***BenCalc*** *function* are at the household level. This is specified in the final parameter, ***TAX_UNIT***, which has the value ***tu_household_tz***.

The output variable *(**Output_Var**)* is called ***bsa_s***, which sums up and records the total monthly benefits received by each household in 2018.

Interactions between functions

Usually more than one ***function*** is used to calculate a benefit or tax, which means that the *functions* interact in some way. These interactions can be classified in four categories, shown in Table 5.1.

Table 5.1: Interactions between functions

Category	Use
Condition	One *function* (usually ***Elig***) evaluates a condition and a subsequent *function* operates on the basis of the result of this evaluation, using *parameter* ***who_must_be_elig***
Input	One *function* calculates a result which is stored in a variable and this variable is used in a subsequent *function*
Addition	One *function* calculates a part of a policy, and a subsequent *function* calculates another part of the policy, and needs to be added to the first part (e.g. a benefit amount and an extra top-up to the benefit amount for specific groups)
Replacement	A subsequent *function* replaces the result of a precedent *function*, which of course only makes sense if the result of the first *function* is used in between

5.2 Parameters

This section on parameters discusses common parameters, parameter syntax and operations, queries, interpreting conditions with respect to assessment units, footnotes, and amounts.

Common parameters

Common parameters are used by several ***functions***. They can be classified into four categories, shown in Table 5.2.

Table 5.2: Common parameters

Category and availability in functions	Common parameter	Use
1. Common parameters affecting output Provided by all tax-benefit policy functions, except ***Elig***	***output_var***	Records allocated tax or benefit amounts to a specified variable Overwrites any existing value of ***output_var***
	output_add_var	Records allocated tax or benefit amounts to a selected variable Result is added to any existing value of ***output_var***
	result_var	Allows a second output variable Generally used in combination with ***output_add_var***
2. Common parameters affecting eligibility Provided by all tax-benefit policy functions	***who_must_be_elig***	***Elig*** function sets a variable (usually ***sel_s***) to 0 or 1, based on a condition defined by the parameter ***elig_cond***. Subsequent functions use this information, together with the setting of parameter ***who_must_be_elig***, to determine whether or not their calculations should be carried out for an assessment unit
	elig_var	Indicates whether a person is eligible
3. Common parameters limiting results Provided by all tax-benefit policy functions	***lowlim***	Used to set a lower limit
	uplim	Used to set an upper limit
	threshold	Used to define a threshold
4. Common parameter TAX_UNIT Compulsory for all tax-benefit policy functions	***TAX_UNIT***	Defines the assessment unit (individual, household, family, etc.) to which the function refers. Assessment units are needed because the eligibility criteria for benefits and taxes do not always focus only on the individual but may take the entire household or several household members into account. Assessment units range from individual units *(each person builds their own unit)* to various definitions of family and household units.

There are also features, provided by all policy functions, which control whether a ***function*** is processed, including:

- The ***on/off*** switch (for turning *functions* ***on*** or ***off*** – not a parameter per se); and
- The parameter ***run_cond***, which allows for conditional processing of the *function*. In other words, the function is only carried out if the respective condition is fulfilled

Further detail on common parameters can be found in the ***EUROMOD Functions*** section of the EUROMOD ***Help*** (accessed from the ***Help & Info*** tab).

Parameter syntax and operations

Certain syntax rules have to be followed when writing the *parameter* formulas for ***Comp_Cond*** or ***Elig_Cond****:*

- Commonly, these conditions are based on variables included in the input data (or variables generated in the model).
    - For instance, such a condition might be as simple as ***ddi=1***, indicating that a person needs to be disabled, or ***dag<16***, indicating that a person needs to be less than 16 years of age (for instance in order to receive a benefit).
        - The so-called comparison operators that can be used in such formulas include **>** (greater than), **<** (less than), **>=** (greater than or equal to), **<=** (less than or equal to), **=** (equal to), and ***!=*** (not equal to).
    - Conditions can also be combined. For example, ***ddi=1 & dag<16*** indicates that the person needs to be *both* disabled *and* under the age of 16.
        - Conditions can be combined using logical operators **&** (and) and **|** (or).
- It is also possible for a condition to only have *one* component, namely a yes/no query such as ***IsParent***. Here, the condition is fulfilled if the person is a parent based on the input data. There can also be reverse conditions such as ***!IsParent***, which indicates that the condition is fulfilled if a person is *not* a parent. See 'Queries' on page 33 for more information on queries.
- Parentheses can be used to group conditions. For instance, condition ***(ddi=1 & dag<16) | dag>=60*** indicates that the person needs to either (a) both disabled and under the age of 16, or (b) at least 60 years of age (e.g. to receive a benefit).

The *parameter formula* in the *function* ***ArithOp*** allows for the operations shown in Table 5.3.

Table 5.3: ArithOp operations

Operation	**Operator**	**Examples**
Addition	+	
Subtraction	-	
Multiplication	*	
Division	/	
Raising to a power	^	2 ^ 3 → result: 8
Percentage	%	yem*3% → result: yem*(3/100)
Reminder of division	\	22\5 → result: 2
Minimum and maximum	<min> or <max>	10 <min> 15 → result: 10 0 <max> 1 → result: 1
Absolute value	<abs>()	<abs>(-22) → result: 22 <abs>(50-70) → result: 20
Negation	!()	!(IsMarried) → result: 0 !(17) → result: 0 !(0) → result: 1

These ***operations*** are to be used with the following ***operands***:

- Numeric values, e.g. 10, 0.3 or -25
- Numeric values with a period, e.g. ***12000#m*** or ***1000#y***
    - Note that ***#m*** in this example refers to a monthly value, while ***#y*** refers to a yearly value. It is good practice to always indicate such a period. See section 'Amounts' on page 36 for further details.
- Constants (specified in a separate constants function), e.g. ***$fbct_child_amnt***
- Income lists (specified in a separate income list function), e.g. ***ils_dispy***
- Queries, e.g. ***IsParent***
- Random numbers, ***rand***

Order of ***operation rules*** is as follows:

1. First: ^, <min>, <max>, <abs>, (), !(), %
2. Which come before multiplicative operations: *, /, \
3. Which come before additive operations: + −

Parentheses can be used to group operations, e.g. (2+3)*4.

Queries

Queries such as ***IsParent*** and ***IsMarried*** are EUROMOD expressions used effectively as shorthand for determining the answers to particular questions. The result of a query is either 1 (i.e. true) or 0 (i.e. false), or some numeric (monetary or non-monetary) value, as shown in examples below:

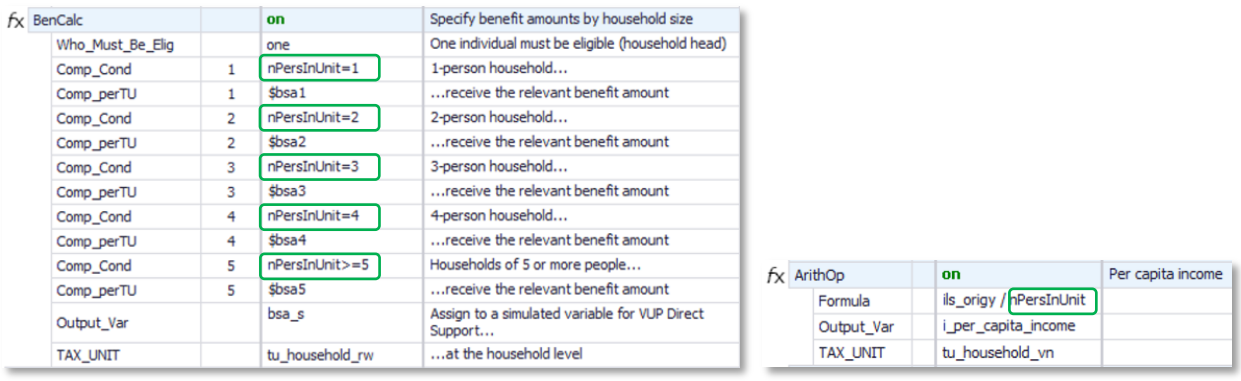
- ***IsParent*** answers the question 'Is the person a parent to a child?'. The result is either 1 (true) or 0 (false).
- ***IsMarried*** answers the question 'Is the person married?'. The result is either 1 (true) or 0 (false).
- ***GetPartnerIncome*** is short hand for the income of the partner. The result is a numeric monetary value.
- ***nDepChildrenInTu*** calculates the number of children in the assessment unit, such as household (see Table 5.2 for a definition of assessment units). The result is a numeric value.
- ***nPersInUnit*** calculates the number of people in the assessment unit. The result is a numeric value.

Queries save the user from having to specify all the variables and codes needed to provide the answer each time.

Figure 5.3 shows two ways of using queries in the models:

- On the left, an example of a social cash transfer from the Rwandan model shows how benefit amounts differ based on the size of the household. For instance, households with 2 members (***nPersInUnit=2***) receive different benefit amounts than households with 3 members (***nPersInUnit=3***).
- On the right, an example from the Vietnamese models shows how per-capita income for households are calculated by dividing total original income within a household (***ils_origy***) by the number of household members (***nPersInUnit***). Note that, as opposed to the above example, here the query is used without specifying households of a particular size.

Figure 5.3: Using queries in functions: Examples from RWAMOD and VNMOD



fx BenCalc		on	Specify benefit amounts by household size
Who_Must_Be_Elig		one	One individual must be eligible (household head)
Comp_Cond	1	nPersInUnit=1	1-person household...
Comp_perTU	1	$bsa1	...receive the relevant benefit amount
Comp_Cond	2	nPersInUnit=2	2-person household...
Comp_perTU	2	$bsa2	...receive the relevant benefit amount
Comp_Cond	3	nPersInUnit=3	3-person household...
Comp_perTU	3	$bsa3	...receive the relevant benefit amount
Comp_Cond	4	nPersInUnit=4	4-person household...
Comp_perTU	4	$bsa4	...receive the relevant benefit amount
Comp_Cond	5	nPersInUnit>=5	Households of 5 or more people...
Comp_perTU	5	$bsa5	...receive the relevant benefit amount
Output_Var		bsa_s	Assign to a simulated variable for VUP Direct Support...
TAX_UNIT		tu_household_rw	...at the household level

fx ArithOp	on	Per capita income
Formula	ils_origy / nPersInUnit	
Output_Var	i_per_capita_income	
TAX_UNIT	tu_household_vn	

Further examples are shown in Section 6.3 with 'Defining tax units or assessment units (tudef_cc)'.

The full list of queries can be found in the ***EUROMOD Functions*** section of the EUROMOD ***Help*** (accessed from the ***Help & Info*** tab).

Interpreting conditions with respect to assessment units

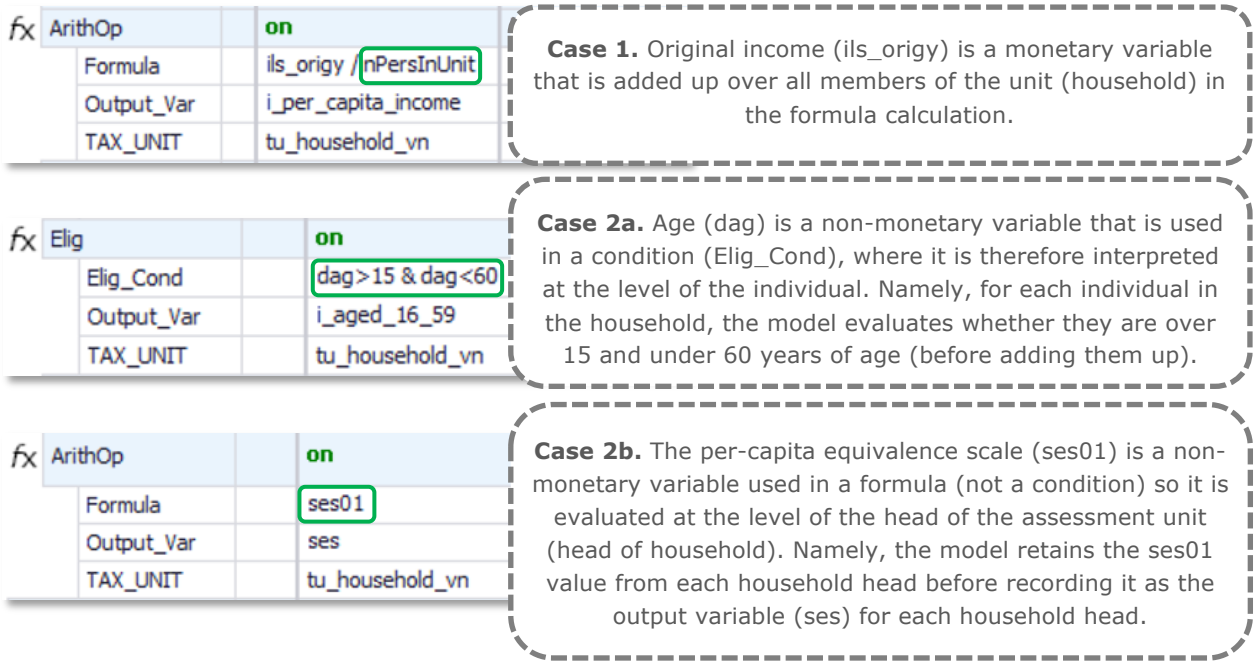
An assessment unit or tax unit refers to the unit of assessment (e.g. individual, household or family) used in a given function. Assessment units are required because the eligibility criteria of benefits and taxes do not always focus on the individual but may take in account the entire household or several household members.

There are differences in interpreting how the values for ***parameters***, ***income lists*** and ***queries*** are used with assessment units with more than one person.

The following rules of interpretation apply in such cases, with examples in Figure 5.4:

1. ***Parameters*** relating to monetary variables as well as ***income lists*** are interpreted at the level of the assessment unit, defined by the *parameter* ***TAX_UNIT***. Values are therefore added up over all members of the unit (case 1).
2. ***Parameters*** relating to non-monetary variables and individual-level ***queries*** are interpreted at the level of the individual if there is a condition (case 2a), or at the level of the head of the assessment unit for all other *parameters* (case 2b).
3. ***Parameters*** relating to non-individual level ***queries*** are interpreted in varying ways.

Figure 5.4: Interpreting conditions with respect to assessment units: Examples from VNMOD



It is possible to change the assessment unit generally used by the *function* for single variables, *income lists*, or *queries*.

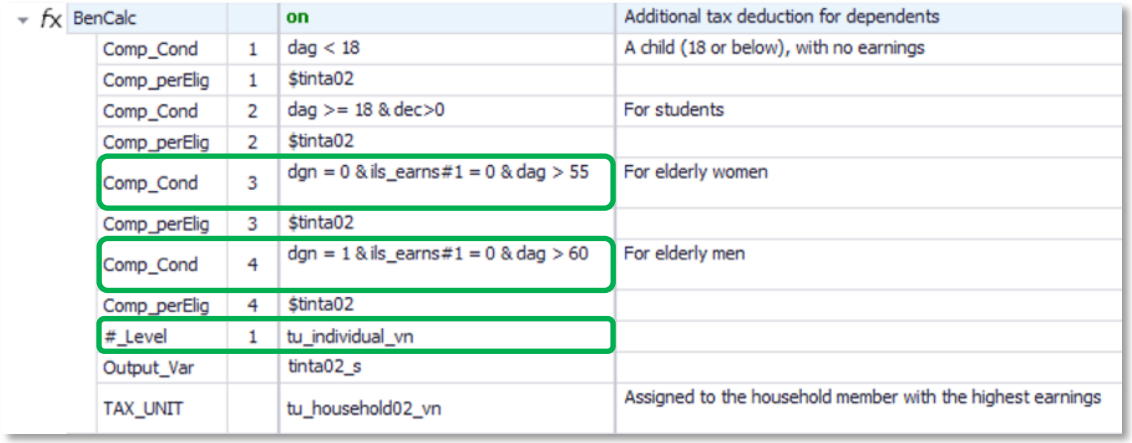
Footnotes

Footnotes are ***parameters*** which serve to further specify other *parameters*. They are referred to as ***footnote parameters*** or ***footnotes***.

- *Footnotes* can be easily identified by names starting with the character ***#*** with an integer number given in the ***Grp/No*** column, for instance ***#_Level*** and ***1***.
- *Footnotes* are applicable with several functions, specifically all functions providing formula or condition *parameters*.

Figure 5.5 shows an example of footnotes used in the Vietnamese model, where a function's assessment unit is changed for particular operands. Specifically, earnings (***ils_earns***) are used at the individual level (***TAX_UNIT*** is ***tu_individual_vn***) when calculating deductions for elderly women and men. In the example, ***ils_earns#1 = 0*** is a condition that checks whether earnings of an individual are zero. ***#1*** corresponds to the ***#_Level*** parameter set to ***1***. The other conditions and the total deduction are however calculated at the household level.

Figure 5.5: Using footnotes: Example from VNMOD



fx BenCalc		on	Additional tax deduction for dependents
Comp_Cond	1	dag < 18	A child (18 or below), with no earnings
Comp_perElig	1	$tinta02	
Comp_Cond	2	dag >= 18 & dec>0	For students
Comp_perElig	2	$tinta02	
Comp_Cond	3	dgn = 0 & ils_earns#1 = 0 & dag > 55	For elderly women
Comp_perElig	3	$tinta02	
Comp_Cond	4	dgn = 1 & ils_earns#1 = 0 & dag > 60	For elderly men
Comp_perElig	4	$tinta02	
#_Level	1	tu_individual_vn	
Output_Var		tinta02_s	
TAX_UNIT		tu_household02_vn	Assigned to the household member with the highest earnings

The four most commonly used types of footnotes are described in Table 5.4.

Table 5.4: Commonly used footnote types

Type of footnote	**Use**
Assessment Units	It is possible to change the function's *assessment unit* (indicated by the parameter ***TAX_UNIT***) for a single operand (as in Figure 5.5)
Limits	It is sometimes necessary to set limits (lower, upper, threshold) to *function* results and single operands.
Amounts	It is sometimes more transparent to indicate amounts outside the formula (particularly in a complex formula or implementation of several *policy years*).
Queries	A few queries need further specification. For instance, ***nDepChildrenInTaxunit*** counts the number of dependent children in the assessment unit. It has two optional *parameters*, ***#_AgeMini*** and ***#_AgeMaxi***, which allow you to specify the age of the dependent children.

Amounts

Amount ***parameter*** values are usually followed by their 'period'. It is good practice to always indicate a period, although ***#m*** (monthly) has no real effect as EUROMOD internally converts all amounts to monthly and would assume a monthly amount if no period was specified.

Most models use at least ***#m*** (monthly – no conversion) and ***#y*** (yearly – divided by 12 when generating outputs from the model). Other options are also possible, for example quarterly *(**#q**),* weekly *(**#w**)* and daily *(**#d**)*.

In general, it makes sense to specify amounts as the time period in which they are commonly discussed (e.g. benefit or grant amounts in monthly terms, and income tax thresholds in annual terms).

As an example, Figure 5.6 shows selected constants (including those shown in Figure 5.2) from TAZMOD. The upper limit for presumptive tax is defined in annual terms, while the two benefit amounts *(**$fbct***)* are defined in monthly terms. See also 'Specifying constants (constdef_cc)' in Section 6.3 for the use of amounts in the ***constdef_cc*** policy.

Figure 5.6: Amounts: Examples of selected constants from TAZMOD

	Policy	Grp/No	TZ_2022	TZ_2023	Comment
9	**constdef_tz**		on	on	**DEF: CONSTANTS**
9.1	DefConst		on	on	Define constants
9.1.1	$presumptive_upper_limit		100000000#y	100000000#y	Upper limit for presumptive tax
9.1.2	$fbct_child_amnt		5000#m	5000#m	FBCT Child Amount
9.1.3	$fbct_adult_amnt		12000#m	12000#m	FBCT Adult amount

6 Definitional policies

6.1 Data adjustment

Uprating factors (uprate_cc)

Overview: A single input dataset is often used for simulating several policy years, by uprating monetary values of variables included in the input dataset to the corresponding policy year. For instance, in cases where newer data is not available, input data for 2020 can be used not only for the 2020 policy system but also for 2021, 2022 and subsequent policy systems.

The *function* ***Uprate*** under *policy* ***uprate_cc*** – included in all models in the SOUTHMOD model bundle – allows for the uprating of monetary dataset variables to the price level of the policy year. This *function* allows for specifying the following parameters:

- ***dataset:*** The *input dataset* to which uprating settings apply;
- ***def_factor:*** Default *uprating factor* (generally the overall CPI), applied to all monetary variables for which uprating factors have not been specified separately;
- The specific *uprating factors* that are used for specific monetary variables, namely different income and expenditure items.

As an example, Figure 6.1 shows the first parameters in *policy* ***uprate_mz*** from MOZMOD, the tax-benefit microsimulation model for Mozambique. Namely:

- ***dataset:*** Uprating settings in the 2018 *policy system* apply to the 2015 *input dataset* (***mz_2015_a****, where the asterisk allows for the user of different data versions);
- ***def_factor:*** The default *uprating factor* is the overall CPI *(**$f_CPI_total**);*
- The general wage inflator *(**$f_general_wage_inflator**)* is used for uprating employment income *(**yem**)* and self-employment income *(**yse**);* and
- The food CPI *(**$f_CPI_food**)* is used to uprate expenditures on food items, such as rice *(**x011111** and **x011113**).* Other such CPIs that are specific for different types of products and services are used as uprating factors for the related expenditure items.

Figure 6.1: Example of uprating factors: uprate_mz in MOZMOD

	Policy		Grp/No	MZ_2018	Comment
1	uprate_mz			on	DEF: UPRATING FACTORS (FACTORES DE ACTUALIZAÇÃO)
1.1	Uprate			on	Define uprating factors
1.1.1		Dataset		mz_2015_a*	
1.1.2		Def_Factor		$f_CPI_total	
1.1.3		yem	1	$f_general_wage_inflator	
1.1.4		yse	2	$f_general_wage_inflator	
1.1.5		x011111	3	$f_CPI_food	RICE WITHOUT BARKING
1.1.6		x011113	4	$f_CPI_food	Rice in shell
1.1.7		x011118	5	$f_CPI_food	AMARULA

As discussed in 'Tools: Uprating Indices' in Section 4.4, the numerical values of the uprating factors are specified in the ***Uprating indices*** dialog, called from the ***Country Tools*** tab.

Recode negative incomes to zero (neg_cc)

Overview: Following the SOUTHMOD Modelling Conventions, each model in the SOUTHMOD bundle records any negative values for market incomes in the input data as zero before applying further calculations. These income types are employment income *(**yem**),* self-employment income *(**yse**), and* agricultural income *(**yag**).*

The related implementation in the country models, relying on ***ArithOp*** functions and ***<max>*** formulas, is shown in Figure 6.2, using MOZMOD as an example.

Figure 6.2: Example of recoding negative incomes to zero: neg_mz in MOZMOD

2	**neg_mz**		**on**	**DEF: Recode negative income to zero**
2.1	ArithOp		**on**	Recoding negative employment income to 0
2.1.1	Formula		0<max>yem	
2.1.2	Output_Var		yem	
2.1.3	TAX_UNIT		tu_individual_mz	
2.2	ArithOp		**on**	Recoding negative self-employment income to 0
2.2.1	Formula		0<max>yse	
2.2.2	Output_Var		yse	
2.2.3	TAX_UNIT		tu_individual_mz	
2.3	ArithOp		**on**	Recoding negative agricultural income to 0
2.3.1	Formula		0<max>yag	
2.3.2	Output_Var		yag	
2.3.3	TAX_UNIT		tu_individual_mz	

COVID-19 transitions and shocks (lma_cc)

Overview: Most models in the SOUTHMOD bundle use *input datasets* from years preceding the COVID-19 pandemic that began in 2020. Simple uprating of monetary values from these older datasets to the *policy years* associated with the pandemic would generally imply *higher* nominal incomes; the documented income losses during the crisis would not be accounted for. For the courtesy of users, most models in the SOUTHMOD bundle contain a definitional policy called ***lma_cc**,* which applies shocks to incomes 'on-model' in 2020 and 2021.

When the policy is set ***on*** (default in the 2020 and 2021 policy systems), a portion of workers in each industry transitions from paid employment to unemployment with no market income. Household consumption expenditures are adjusted downwards accordingly based on absolute reductions in disposable income (see Section 6.4). Labour market characteristics are adjusted as well; as an example, labour market status is changed to 'unemployed' for the transitioning workers. The adjusted incomes and labour market characteristics are then taken into account in the simulation of taxes, social insurance contributions and benefits.

The shares of workers who transition to unemployment are derived from changes in each industry's GDP from its counterfactual values in 2020 and 2021, computed based on the pre-pandemic, 2017–19 linear trend. GDP shocks are used as a proxy for average losses of market income in each sector. Specifically, it is assumed that the size of the proportional GDP shock

in a given sector is equivalent to the share of workers who transition to unemployment with no market income. See national Country Reports and Lastunen (2022) for details.[7]

6.2 Income lists

Overview: An ***income list*** is the aggregate of several variables that are summed together to build the aggregate. Perhaps the most common application of this concept is ***income definitions***. As an example, the income list for *disposable income* is defined as *original (market) income + benefits – taxes – employee social insurance contributions*.

When an *income list* is defined, it is available for all *functions* and *policies and re-calculated every time it is used*. All *income lists* generated in the model are included in ***model output files***, along with individual simulated variables.

In a multi-country tax-benefit model such as SOUTHMOD, it is important to have a standardized output format for all countries that still allows for country-specific definitions of the particular variables; for instance, each model defines a standard income list for means-tested benefits, but the specific benefit variables names included in that list differ between models based on the existing benefits in the country.

In each model, *income lists* themselves are named in an identical manner and organized into four separate income list *policies*, discussed in detail in the next subsections*:*

1. Standard income lists *(**ilsdef_cc**)*
2. Model-specific income lists *(**ildef_cc**)*
3. Statistics Presenter income lists *(**ildef_stats_cc**)*
4. Expenditure income lists *(**ildef_exp_cc**)*

Note that any other *policy* (other than ***uprate_cc***) may also contain income list definitions. An example of an *income list* not included in the main four policies above is the *income list* for standard-rated VAT items *(**ils_vat_std**),* generally included in the VAT policy. It is available in all SOUTHMOD country models and covered at the end of this section.

Further information is available in the SOUTHMOD Modelling Conventions (Section 11). Required *edits* to *income lists* when amending policies are discussed in Section 9.

Standard income lists (ilsdef_cc)

The policy specifying standard income lists *(**ilsdef_cc**)* includes the definitions of:

- The household head *(**ils_head**);*
- Earnings *(**ils_earns**)* and original (market) income *(**ils_origy**);*
- Direct taxes *(**ils_tax**)* and social insurance contributions *(**ils_sicee**, **ils_sicer**, **ils_sicse**);*

[7] Lastunen, J. (2022). On-Model Adjustment of Incomes During COVID-19 in SOUTHMOD Tax-Benefit Microsimulation Models. *WIDER Technical Note 2022/4*. Helsinki: UNU-WIDER.

- Pension-related income *(**ils_pen**)* and benefits (different components based on the respective model);
- Three different definitions of disposable income (see the last three lists in Figure 6.3).

The income lists specified in ***ilsdef_cc*** and their composition are explained in great detail in the SOUTHMOD Modelling Conventions (see especially Table 4 on page 26).

For illustration, however, Figure 6.3 shows an example of a standard income list*, **ilsdef_et**,* from ETMOD, the model for Ethiopia. Some functions have been expanded for illustration.

Figure 6.3: Example of a standard income list: ilsdef_et in ETMOD

Policy	Grp/No	ET_2020	Comment
ildef_std_et		**on**	**DEF: STANDARD INCOME LISTS**
fx DefIl		on	Household Head
fx DefIl		on	Earnings
Name		ils_earns	
yem		+	employment income
yse		+	self employment income
yag		+	farmer income
fx DefIl		on	Original income
Name		ils_origy	
ils_earns		+	Earnings
yiy		+	Investment income
ypr		+	Income from property
ypt		+	Income from private transfer
fx DefIl		on	Taxes ("Direct taxes" in Statistics Presenter)"
Name		ils_tax	
tin01_s		+	Employment income tax
tin02_s		+	Self-employment income tax
fx DefIl		on	Employee social contribution
fx DefIl		on	Employer social contribution
fx DefIl		on	Self-employed social insurance contributions
fx DefIl		on	Pension benefits ("Pension benefits" in Statistics Presenter)
Name		ils_pen	
poa		+	Contributory old-age pension
fx DefIl		on	Means tested benefits
fx DefIl		on	Non means tested benefits
fx DefIl		on	Cash benefits
fx DefIl		on	In-kind Benefits
fx DefIl		on	Disposable income without imputed home produce
Name		ils_dispy	
ils_origy		+	Original income
ils_ben		+	All benefits
ils_tax		-	Taxes
ils_sicee		-	Employee social insurance contributions
ils_sicse		-	Self-employed social insurance contributions
fx DefIl		on	Post-fiscal disposable income without imputed home produce
fx DefIl		on	Disposable income accounting for in-kind benefits

Note the following list of details about the implementation of the ***ilsdef_et*** policy, which are relevant for all country models included in the SOUTHMOD bundle:

- The income list policy is composed of ***DefIl*** functions, which allow for defining income lists consisting of different variables.
    - In the example in Figure 6.3, some of these functions have been expanded.

- The name provided for an income list in each ***DefIl*** function starts by '***ils_****', where '***s***' stands for 'standard'. This is to distinguish them from ordinary variables.

- As shown, income lists are defined by summing up values of different variables, using a '+' operator, or subtracting variable values from the total, using a '-' operator.
  - As an example, the income list for earnings, ***ils_earns***, comprises employment income *(**yem**),* self-employment income *(**yse**), and* agricultural income *(**yag**).*

- Income lists generated can be used as inputs in other income lists. For instance, the ***ils_earns*** is used in the subsequent income list for original income, ***ils_origy***, which includes earnings but also other types of income.

- The income list for disposable income without imputed home produce, ***ils_dispy***, is calculated by adding (+) or subtracting (-) different income components, listed by their variable name. Specifically, disposable income is computed as:
  - Original income;
  - *Plus* benefits;
  - *Minus* taxes;
  - *Minus* employee and self-employed social insurance contributions.

- The above is a fairly standard definition of disposable income but could easily be modified if a slightly different definition was required, by either amending the existing income list (not required for ***ils_dispy***) or adding a new income list.
  - For instance, policy ***ildef_stats_cc*** discussed below includes a corresponding income list for disposable income that *additionally* includes imputed home produce.

- Income list ***ils_dispy***, and many other income lists that are specified, are not used within any policy but are particularly important in the analysis stage (when analysing model output datasets that contain all income list variables). The use of income lists with the same definition across countries allows for and simplifies comparable cross-country research as researchers only need to know the name of the standard income list and do not need to be aware of every single component in each country.

- There are two standard income lists, ***ils_tax*** and ***ils_pen***, that are part of the standard income list policy but also used by the ***Statistics Presenter*** (as are those specified directly in Statistics Presenter income lists in policy ***ildef_stats_cc***).

Model-specific income lists (ildef_cc)

The policy specifying model-specific income lists *(**ildef_cc**)* may include any income lists that are not part of the standard income lists *(**ilsdef_cc**, **ildef_stats_cc*** and ***ildef_exp_cc**)* nor

specified within particular policies. Non-standard income lists that could be included, depending on what policies are used, include taxable income for income tax purposes and the sum of incomes relevant for means-testing of benefits. The former is defined for instance in ETMOD.

As opposed to the other income list policies, the name of each country-specific income list must start with a prefix '***il_****' (as opposed to '***ils_****' for standard, Statistics Presenter and expenditure income lists, as well as income lists for standard-rated VAT items).

Statistics Presenter income lists (ildef_stats_cc)

The income lists required (directly or indirectly) by the ***Statistics Presenter*** tool and collected in the Statistics Presenter income list policy *(**ildef_stats_cc**)* include:

- Total simulated indirect taxes *(**ils_taxind**);*
- Total simulated social security contributions *(**ils_sic**);*
- Simulated benefits by type (not including pension benefits, ***ils_pen***, contained in the ***ilsdef_cc***), include:
    - Child-related benefits, including school feeding *(**ils_bch**)*;
    - Social-assistance related benefits *(**ils_bsa**)*;
    - Orphan and widowhood-related benefits *(**ils_bsu**)*;
    - Disability-related benefits *(**ils_bdi**)*;
    - Unemployment-related benefits *(**ils_bun**)*; and
    - Agricultural benefits *(**ils_bag**).*
- Disposable income accounting for own produce (***ils_dispyx***, and ***ils_dispyx_pf***, the post-fiscal definition that subtracts indirect taxes from the total); and
- Simulated consumption (***ils_con***, and ***ils_con_pf***, the post-fiscal definition that subtracts indirect taxes from the total).

Apart from new variables generated within the data preparation stage (see Section 9.3), the ***Statistics Presenter*** uses the income lists above to generate the requisite output (namely, information shown in the result panels for ***Tax-ben policy***, ***Poverty*** *and* ***Inequality***). The income lists in ***ildef_stats_cc*** and their content are explained in detail in the SOUTHMOD Modelling Conventions (see especially Table 5 on page 27), including which lists are used in different panels of the ***Statistics Presenter***.

Expenditure income lists (ildef_exp_cc)

The income list policy for household expenditure items *(**ildef_exp_cc**)* includes 13 income lists following the two-digit level of the 2018 COICOP classification, for instance:

- ***ils_coicop01*** for items belonging to COICOP 01 (food products);
- ***ils_coicop02*** for items belonging to COICOP 02 (alcohol and tobacco);
- etc.

The expenditure income lists are included in all SOUTHMOD country models in order to have harmonized expenditure lists and thereby to facilitate comparative research using the models.

Income list for standard-rated VAT items (ils_vat_std)

While not specified in the four main income lists policies, all countries in the bundle contain an income list that collects standard-rated items subject to VAT. This income list is called ***ils_vat_std*** and sits directly within the relevant VAT policy.

In fact, ***ils_vat_std*** generally lists *all* expenditure items, putting a ***'+'*** sign against items subject to standard rate VAT and an ***'n/a'*** against items that are exempt from VAT or zero-rated. Note that items that are VAT exempt or zero-rated should not have a ***'-'*** sign alongside them, as this would mean they are *subtracted* from the total income list, which is not correct.

In the case of a two-tier VAT system, ***ils_vat01*** and ***ils_vat02*** collect the respective expenditure items subject to the two different tax rates. Further lists may be included in case of more than two tax rates.

6.3 Assessment units, constants, poverty lines and equivalence scales

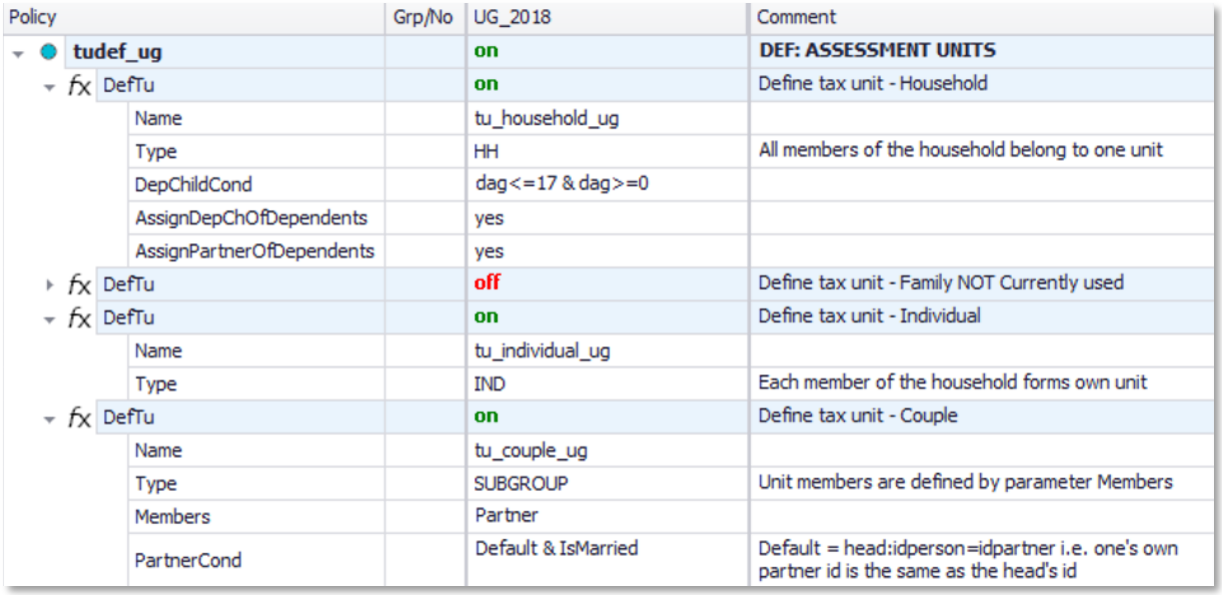
Defining tax units or assessment units (tudef_cc)

Overview: The *policy* ***tudef_cc*** in each model contains the definitions of ***tax units*** or ***assessment units***. Eligibility criteria of many benefits and sometimes also taxes do not focus on the individual only but take the whole household or several household members into account. *Tax units* specify which household member belongs to the assessment units, who is defined as a child, and so on. *Tax units* are defined by function ***DefTu*** and are used in tax-benefit *policies* for the implementation of the respective tax or benefit.

In principle any *policy* other than ***uprate_cc*** may contain assessment unit definitions. However, for reasons of transparency, assessment units are usually defined centrally in the *policy* ***tudef_cc***. This rule may be disregarded if a particular assessment unit is used temporarily in one special *policy*. An assessment unit, once defined, is available for all subsequent *functions* and *policies*.

As with income lists, a multi-country tax-benefit model such as SOUTHMOD includes several, sometimes very different assessment unit definitions and therefore must allow for flexible specification on-model (instead of hard-coding). This is allowed by specifying tax units. The main definitions are however generally the same across all SOUTHMOD models in the bundle. Figure 6.4 shows an example of three ***DefTu*** functions in the ***tudef_ug*** policy in UGAMOD, the model for Uganda.

Figure 6.4: Example of defining assessment units: tudef_ug in UGAMOD



Policy	Grp/No	UG_2018	Comment
tudef_ug		on	**DEF: ASSESSMENT UNITS**
DefTu		on	Define tax unit - Household
Name		tu_household_ug	
Type		HH	All members of the household belong to one unit
DepChildCond		dag<=17 & dag>=0	
AssignDepChOfDependents		yes	
AssignPartnerOfDependents		yes	
DefTu		off	Define tax unit - Family NOT Currently used
DefTu		on	Define tax unit - Individual
Name		tu_individual_ug	
Type		IND	Each member of the household forms own unit
DefTu		on	Define tax unit - Couple
Name		tu_couple_ug	
Type		SUBGROUP	Unit members are defined by parameter Members
Members		Partner	
PartnerCond		Default & IsMarried	Default = head:idperson=idpartner i.e. one's own partner id is the same as the head's id

Details on these definitions are covered in Table 6.1. The information applies to all country models in the SOUTHMOD bundle. For more information on queries, used in this policy, is available in subsection 'Queries' in Section 5.2.

Table 6.1: Definitions of assessment units: Example from UGAMOD

Tax unit	Name and description	Details of function contents
Household	***tu_household_ug*** A unit consisting of all members in the household, as specified in the input data (as individuals having the same household identifier, ***idhh***)	• Parameter *Type* is set to ***HH*** (one of 3 possible values) • ***HH*** refers to household type units, which means that all members of the household belong to the same unit • Values for parameter ***DepChildCond*** specify that any household members between the ages of 0 and 17 are treated as dependent children in the household • Parameters ***AssignDepChOfDependents*** and ***AssignPartnerOfDependents*** mean that dependent children or partners of dependent unit members (i.e. persons who are not the head or partner of the unit) are assigned to the unit
Individual	***tu_individual_ug*** Each individual (i.e. member of a household) forms their own unit (***idperson*** in the input data)	• Parameter *Type* is set to ***IND*** • ***IND*** denotes individual type units, which means that each member of the household forms its own unit • Other than ***Name*** and ***Type***, no other parameters are required; this is the simplest form of tax unit definition • This definition is used, for example, for personal income tax, where tax is calculated for each person and child/adult definitions are not relevant

Couple	***tu_couple_ug*** A unit consisting of one individual and their partner	• Parameter *Type* is set to ***SUBGROUP*** • ***SUBGROUP*** means that the household may be split into several units of different size • In this case, ***Partner*** is set as the value for parameter ***Members***, specifying that a couple consists of the head of the unit and their partner • The parameter ***Members*** usually defines relations with respect to the head of the unit, which is the richest person in the unit, or if there are several equally rich persons, the oldest, or if there are several equally rich and equally old persons, the one with the lowest number for the variable ***idperson*** • The partner is defined in parameter ***PartnerCond***, which requires that queries ***Default*** and ***IsMarried*** are both equal to 1 (true), following information in the input data.

Specifying constants (constdef_cc)

Overview: The *policy* ***constdef_cc*** contains definitions of ***constants***. A *constant* is simply a way of storing a number (usually a rate or a monetary amount) for use in one or more tax-benefit *policies*. Because most *constants* usually reside in one place, this *policy* is a particularly good place to define those benefit, tax or contribution amounts that are amended each year. Note that any other *policy* (other than ***uprate_cc***) may also contain constant definitions.

Figure 6.5 shows this policy, ***constdef_rw****,* from RWAMOD, the model for Rwanda.

Figure 6.5: Example of constants: constdef_rw in RWAMOD

9	constdef_rw		on	DEF: CONSTANTS
9.1	DefConst		on	Define constants: Poverty rates
9.2	DefConst		on	Define constants: Social insurance contributions
9.2.1		$tsceepi_ra	0.03	Contribution rate of gross salary for employees in the Pension Scheme
9.2.2		$tscerpi_ra	0.03	Contribution rate of gross salary for employers in the Pension Scheme
9.2.3		$tscerac_ra	0.02	Contribution rate of gross salary for employers in the Occupational Hazards Scheme
9.2.4		$tsceeml_ra	0.003	Contribution rate of gross salary for employees in the Maternity Leave Benefit Scheme
9.2.5		$tscerml_ra	0.003	Contribution rate of gross salary for employers in the Maternity Leave Benefit Scheme
9.2.6		$tsceehl_rama_ra	0.075	Contribution rate of basic earnings for employees in RAMA
9.2.7		$tscerhl_rama_ra	0.075	Contribution rate of basic earnings for employers in RAMA
9.2.8		$tscpehl_rama_ra	0.075	Contribution rate of pension for pensioners in RAMA
9.2.9		$tsceehl_cbhi_ubu1_pr	2000#y	Annual premium for CBHI insurance for poorest Ubudehe group 1; may be subsidized (RWF)
9.2.10		$tsceehl_cbhi_ubu2_pr	3000#y	Annual premium for CBHI insurance for Ubudehe group 2 (RWF)
9.2.11		$tsceehl_cbhi_ubu3_pr	7000#y	Annual premium for CBHI insurance for Ubudehe group 3 (RWF)
9.2.12		$tsceehl_cbhi_ra	0.005	Contribution rate of 0.5% of net salary of employees to CBHI (since 2020)
9.2.13		$tsceehl_mmi_ra	0.05	Contribution rate of gross salary for employees in the MMI
9.2.14		$tscerhl_mmi_ra	0.175	Contribution rate of gross salary for employers (Government) in the MMI
9.3	DefConst		on	Define constants: Indirect taxes
9.4	DefConst		on	Define constants: Benefits
9.4.1		$bsa1	7500#m	Direct VUP support for households with 1 member (RWF, monthly)
9.4.2		$bsa2	12000#m	Direct VUP support for households with 2 members (RWF, monthly)
9.4.3		$bsa3	15000#m	Direct VUP support for households with 3 members (RWF, monthly)
9.4.4		$bsa4	18000#m	Direct VUP support for households with 4 members (RWF, monthly)
9.4.5		$bsa5	21000#m	Direct VUP support for households with 5 or more members (RWF, monthly)
9.4.6		$cow_value	9333#m	Value (average price) of a cow divided by expected production lifetime (5 years or 60 months) (RWF, monthly)

In RWAMOD, constants – defined in *function* ***DefConst*** – are grouped in four different ***DefConst*** functions based on the type of policies where they are used (poverty rates, social insurance contributions, indirect taxes and benefits). In Figure 6.5, constants for social insurance contributions and benefits are expanded. The grouping used in RWAMOD is not mandatory but makes it easier to identify and edit the required constants. Note that:

- The *name* of each ***constant*** is defined in the ***Policy*** column, where the first character of a *constant's* name should always be ***$***.
- The *value* of each ***constant*** is defined in the respective ***System*** column. As explained in Section 5.2, the ***#m*** after the value indicates that it is a monthly amount and the ***#y*** indicates that it is an annual amount.

By default, a constant is created as a monetary variable. However, as shown in Figure 6.5, constants may also be rates or other non-monetary variables. For instance, the contribution rate for employees in the Pension Scheme *(**$tsceepi_ra**)* is 3 per cent (0.03), which could also be defined as 3%. No period definition is needed after percentages.

Defining poverty lines (spl_cc)

Overview: In each model in the SOUTHMOD bundle, poverty line used to calculate poverty outcomes is specified in function ***spl_cc***.

This *policy* is shown in Figure 6.6, using the policy ***spl_rw*** from RWAMOD as an example; the implementations are similar for all models in the bundle, although the types of poverty lines differ to some extent.

Figure 6.6: Example of poverty lines: spl_rw in RWAMOD

	Policy	Gr...	RW_2020	Comment
10	**spl_rw**		**on**	**DEF: Rwandan Poverty Line**
10.1	DefVar		**on**	Define poverty line variables
10.1.1	spl		0	Poverty line, initiate at 0
10.1.2	splpf		0	Post-fiscal poverty line, initiate at 0
10.2	ArithOp		**on**	National poverty line
10.2.1	Formula		$povline* $f_CPI_Overall	Inflated using the overall CPI from the uprating indices
10.2.2	Output_Var		spl	
10.2.3	TAX_UNIT		tu_individual_rw	
10.3	ArithOp		**on**	National poverty line for post-fiscal income
10.3.1	Formula		$povline_indir* $f_CPI_Overall	Calculated in Stata and then uprated with the CPI
10.3.2	Output_Var		splpf	
10.3.3	TAX_UNIT		tu_individual_rw	
10.4	ArithOp		switch	Extreme poverty line
10.4.1	Formula		$povline_extr* $f_CPI_Overall	Inflated using the overall CPI from the uprating indices
10.4.2	Output_Var		spl	
10.4.3	TAX_UNIT		tu_individual_rw	
10.5	ArithOp		switch	Extreme poverty line for post-fiscal income
10.5.1	Formula		$povline_extr_indir*	Calculated in Stata and then uprated with the CPI
10.5.2	Output_Var		splpf	
10.5.3	TAX_UNIT		tu_individual_rw	

The first function, ***DefVar***, initiates variables ***spl*** (poverty line) and ***splpf*** (post-fiscal poverty line) at value 0. These values are set to desired values later in the policy using ***ArithOp*** *functions*, as explained below.

For each poverty line, there are two ***ArithOp*** *functions*. The first two *functions* specify the national poverty line as well as its post-fiscal equivalent, i.e. one that accounts for indirect taxes. The latter two *functions* specify the extreme poverty line and its post-fiscal equivalent. In some other SOUTHMOD models, the extreme poverty line is set as the default poverty line given that it is more commonly used in the country in question.

The ***ArithOp*** *functions* are very straightforward. The key *parameter* is a formula that takes the constant for the poverty line (the regular poverty line or the one taking indirect taxes into account) and multiplies it by the overall CPI factor, ***$f_CPI_Overall***. This is necessary because national poverty lines are often only available in selected years and thus need to be uprated to the year of the policy system. The output variables are ***spl*** and ***splpf*** (which are fed into the ***Statistics Presenter*** to calculate poverty rates).

The user can choose to produce model outputs using either the standard or the alternative poverty line (for example, either the national or the extreme poverty line in Rwanda). See 'Extensions: Switches' in Section 4.4 for details. Further, as discussed in Section 9, the user can choose to produce aggregate outputs in the ***Statistics Presenter*** using either the standard or the post-fiscal version of the selected poverty line.

Defining equivalence scales (ses_cc)

Overview: In most models in the SOUTHMOD bundle, *policy* ***ses_cc*** is used to define a household-level equivalence scale (in adult equivalents) used in the given country for poverty and inequality measurement. Such nationally defined equivalence scales used are often kilojoule/calorie-based, with associated weights assigned for individuals based on their age and gender. The ***ses_cc*** policy also allows for adopting a per-capita or square root equivalence scale directly from the model input data. In fact, per-capita equivalence scales are used by default in some SOUTHMOD country models. Figure 6.7 shows an example of the ***ses_rw*** policy from RWAMOD, which uses a kilojoule/calorie-based scale.

Figure 6.7: Example of equivalence scales: ses_rw in RWAMOD

	Policy	Grp/No	RW_2020	Comment
11	**ses_rw**		**on**	**DEF: Equivalence Scales**
11.1	DefVar		**on**	Define equivalence scale variable
11.1.1	ses	1	0	Initiate at 0
11.1.2	Var_Monetary	1	no	Set non-monetary
11.2	BenCalc		**on**	Define weights based on age
11.2.1	Comp_Cond	1	dag=0	<1 year, male and female
11.2.2	Comp_perElig	1	0.41	
11.2.3	Comp_Cond	2	dag>=1 & dag<=3	1-3 years, male and female
11.2.4	Comp_perElig	2	0.56	
11.2.5	Comp_Cond	3	dag>=4 & dag<=6	4-6 years, male and female
11.2.6	Comp_perElig	3	0.76	
11.2.7	Comp_Cond	4	dag>=7 & dag<=9	7-9 years, male and female
11.2.8	Comp_perElig	4	0.91	
11.2.9	Comp_Cond	5	dag>=10 & dag<=12 & dgn=1	10-12 years, male
11.2.10	Comp_perElig	5	0.97	
11.2.11	Comp_Cond	6	dag>=10 & dag<=12 & dgn=0	10-12 years, female
11.2.12	Comp_perElig	6	1.08	
11.2.13	Comp_Cond	7	dag>=13 & dag<=15 & dgn=1	13-15 years, male
11.2.14	Comp_perElig	7	0.97	
11.2.15	Comp_Cond	8	dag>=13 & dag<=15 & dgn=0	13-15 years, female
11.2.16	Comp_perElig	8	1.13	
11.2.17	Comp_Cond	9	dag>=16 & dag<=19 & dgn=1	16-19 years, male
11.2.18	Comp_perElig	9	1.02	
11.2.19	Comp_Cond	10	dag>=16 & dag<=19 & dgn=0	16-19 years, female
11.2.20	Comp_perElig	10	1.05	
11.2.21	Comp_Cond	11	dag>=20 & dag<=39	20-39 years, male and female
11.2.22	Comp_perElig	11	1.00	
11.2.23	Comp_Cond	12	dag>=40 & dag<=49	40-49 years, male and female
11.2.24	Comp_perElig	12	0.95	
11.2.25	Comp_Cond	13	dag>=50 & dag<=59	50-59 years, male and female
11.2.26	Comp_perElig	13	0.90	
11.2.27	Comp_Cond	14	dag>=60 & dag<=69 & dgn=1	60-69 years, male
11.2.28	Comp_perElig	14	0.90	
11.2.29	Comp_Cond	15	dag>=60 & dag<=69 & dgn=0	60-69 years, female
11.2.30	Comp_perElig	15	0.80	
11.2.31	Comp_Cond	16	dag>=70	70 years and over, male and female
11.2.32	Comp_perElig	16	0.70	
11.2.33	Comp_Cond	17	dag=-1 | dgn=-1	No age and/or no gender
11.2.34	Comp_perElig	17	0.9064	Insert population weighted average where age or gender is missing
11.2.35	Output_Var		ses	Assign to the equivalence scale variable...
11.2.36	TAX_UNIT		tu_household_rw	...at the household level
11.3	ArithOp		**off**	Option to select per capita equivalence scale (ses01) or square root equivalence scale (ses02) from data
11.3.1	Formula		ses01	Use ses01 or ses02 from data
11.3.2	Output_Var		ses	Assign to the equivalence scale variable...
11.3.3	TAX_UNIT		tu_household_rw	...at the household level

The policy includes the following components:

- The first function, ***DefVar***, initiates the variable ***ses*** (equivalence scale) at value 0 and specifies it as a non-monetary variable using parameter ***Var_Monetary*** with value ***no***

- The value for ***ses*** is then set to the desired value for each individual using a ***BenCalc*** *function*. This function contains 17 groups of ***Comp_Cond*** and ***Comp_perElig*** parameters. The former specify the population to which a particular equivalence

value is given, while the latter assigns the associated value to these individuals based on national definitions of the equivalence scale. For instance, males *(**dgn=1**)* between the age *(**dag**)* of 10 and 12 are assigned a value of 0.97.

- ***Output_Var*** is defined as ***ses*** and the ***TAX_UNIT*** as ***tu_household***, meaning that the ***ses*** variable sums up and then records the above values for each household.

- The last function in the *policy*, ***ArithOp***, allows for adopting a per-capita *(**ses01**)* or square root *(**ses02**)* equivalence scale directly from the model input data (to replace the national definition). These two scales are generally more suited for cross-country comparisons. While the function is set ***off*** by default, the alternative scales can be used by setting the function ***on*** and specifying either ***ses01*** or ***ses02*** in the ***Formula***.

6.4 Adjusting consumption expenditures

Overview: In each model in the bundle, consumption levels are based on the original reported consumption levels in the input data *(**xhh**)*. These levels are adjusted from the *base year* to the *policy year* by absolute changes in disposable income from the base year to the policy year. The adjustment is implemented in policy ***xhhadj_cc***.

The change in disposable income takes into account changes in market incomes (e.g., COVID-related income losses and uprating) as well as changes in benefits and contributions (e.g. increase in benefit amounts, new benefits, changes in income tax brackets). The underlying assumption is that changes in disposable incomes lead to the same changes in consumption levels. In recognition of the fact that there may be some consumption of own-account produced food, in cases where the base year disposable income is higher than the disposable income in the policy year, a proportion of the original consumption is assumed to be unaffected. This proportion is assumed to be 25 per cent of the original consumption following Tschirley et al. (2015)[8] but can be revised by the user.

6.5 Model outputs

Overview: The *policy* ***output_std_cc*** contains the specification of the output microdata file, stored in the model **Output** folder.

This *policy* is shown in Figure 6.8, using the policy ***output_std_zm*** from MicroZAMOD, the model for Zambia, as an example. The implementations are equivalent for all models in the bundle. Table 6.2 describes the main parameters used in the policy.

[8] Tschirley, D., T. Reardon, M. Dolislager and J. Snyder (2015). The Rise of a Middle Class in East and Southern Africa: Implications for Food System Transformation. *Journal of International Development*, 27(5): 628–46.

Figure 6.8: Example of an output policy: output_std_zm in MicroZAMOD

Policy	Grp/No	ZM_2020	Comment
output_std_zm		**on**	**DEF: STANDARD OUTPUT INDIVIDUAL LEVEL**
fx DefOutput		**on**	Define output at individual level
file		ZM_2020_std	
vargroup		id*	Identification
VarGroup		s*	System Variables eg SPL and SES
vargroup		d*	Demographics
vargroup		l*	Labour market
vargroup		y*	Incomes
vargroup		p*	Pensions
vargroup		b*	Benefits
vargroup		t*	Taxes and SIC
vargroup		a*	Assets
VarGroup		x*	x* output for checking
VarGroup		xhh*	Consumption
VarGroup		q*	q* output for checking
ILGroup		ils*	Standard income lists
ILGroup		il*	Other income lists
unitinfo_tu	1	tu_household_zm	Household
unitinfo_id	1	HeadID	
unitinfo_id	1	IsDependentChild	
unitinfo_id	1	IsLoneParent	
unitinfo_id	1	IsPartner	
nDecimals		2	
TAX_UNIT		tu_individual_zm	

Table 6.2: Parameters for defining outputs: Examples from MicroZAMOD

Parameter	Use
file	• Defines the name of the **output file** (e.g. ***ZM_2020_std***) • Standard output for the 2020 system shown here is then written to a text file named ***ZM_2020_std.txt*** • For the 2023 system, it would be named ***ZM_2023_std.txt***, etc.
VarGroup (vargroup)	• Allows for a group of ***variables*** to be output indicated by the initial letter, followed by an asterisk (the ******* symbol), the 'wild card' • For example, vargroup ***b**** will output all variables in the input data set along with any simulated variables that begin with the letter ***b*** (i.e. all benefit variables, both original and simulated)
Var	• While not used in MicroZAMOD, it is also possible to output individual variables by using the *parameter* ***var*** and specifying the name of the variable
ILGroup	• Allows for a group of ***income lists*** to be output indicated by the initial letter, again followed by an asterisk (the ******* symbol), the 'wild card'
unitinfo_tu	• Allows for outputting tax unit information
unitinfo_id	• Allows for outputting selected definitions for individuals within a tax unit
nDecimals	• Determines the number of decimal places for monetary variables • Any amounts with more decimal places are rounded
defIL	• While not used in MicroZAMOD, this *parameter* allows for all *variables* included within a particular income list to be outputted • Stands for ***definition Income List*** • For example, specifying this parameter ***ils_dispy*** would mean that the output file would contain all variables included in list for standard disposable income • Using the parameter ***ILGroup*** and specifying the value ***ils_dispy*** would instead result in only the overall value for disposable income to be outputted
TAX_UNIT	• Defines the level of output • Set to an individual assessment unit *(**tu_individual_cc**)* in all country models, which means that the output contains one row for each individual • If this *parameter* is set to some larger unit (e.g. a household), there would be one row for each unit, where *monetary values* refer to the sum over all unit members and *non-monetary values* refer to the head of the unit • This is the case for the ***output_std_hh_cc*** policy, which is available but set off by default in all models in the SOUTHMOD bundle

7 Variables

All countries implemented in EUROMOD and based on the EUROMOD framework use the same set of variables to store information taken from the input data and information generated by the model.

7.1 Variable naming conventions

Variables follow specific naming conventions. All variables generated by the model end with ***_s*** to denote that they are simulated. Variables in the input data do not have such an ending. For instance, ***bch*** is a child benefit carried out from the data (reported receipt), whereas ***bch_s*** is a simulated child benefit. The first character of a variable's name indicates its type and the rest of the name is composed of two-character acronyms, as shown in Table 7.1.

Table 7.1: Variable naming conventions

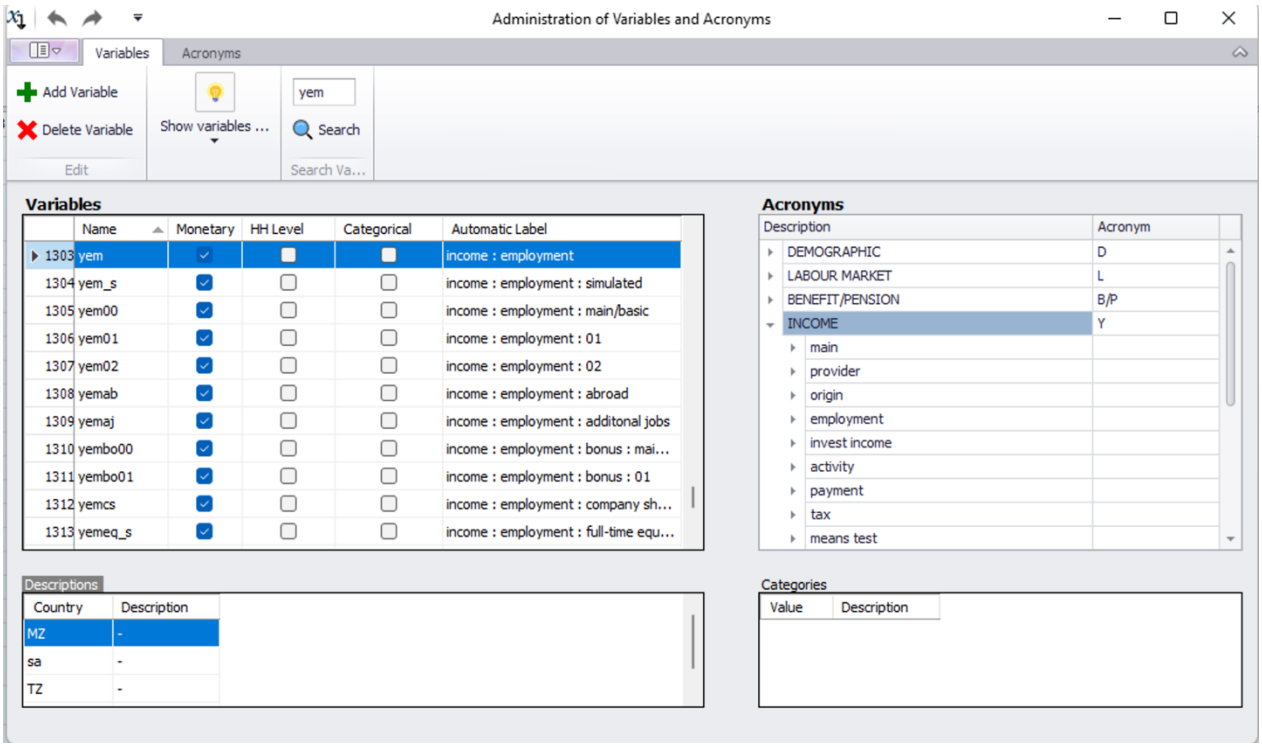
First letter	Type	Variable examples		
b	benefit	***bsa**:*	***b*** = benefit	***sa*** = social assistance
t	tax	***tin**:*	***t*** = tax	***in*** = income
p	pension	***poa**:*	***p*** = pension	***oa*** = old age
d	demographic	***dag**:*	***d*** = demographic	***ag*** = age
l	labour market	***les**:*	***l*** = labour market	***es*** = economic status
y	income	***yem**:*	***y*** = income	***em*** = employment
a	asset	***afc**:*	***a*** = asset	***fc*** = financial capital
x	expenditure	***xhh**:*	***x*** = expenditure	***hh*** = household
k	in kind	***ked**:*	***k*** = in kind	***ed*** = education

7.2 Variable storage, access and modifications

Variables are stored in the ***variable description file*** (***VarConfig.xml***). The file contains the names and properties of all variables available in the SOUTHMOD bundle. Some of the properties help the model to distinguish whether certain routines should be applied to the variable (e.g. only monetary variables are uprated). Other properties are descriptions of the variables, including an automatically generated description of each variable. The file also stores the acronyms of which the variable names are composed.

As mentioned in Section 4.5, the SOUTHMOD user interface provides a tool, a *variable manager*, for administering the information stored in ***variable description file***. To access this tool, shown in Figure 7.1, click on the button ***Variables*** in the ***Administration Tools*** tab.

Figure 7.1: Variable configuration manager



Main contents of the tool are described below:

- **To the left-hand side** is a list of all available ***variables***.
    - For each variable, there is a checkbox *(**Monetary**)* indicating whether the variable is monetary (checked) or not (not checked). This can be amended.
    - Other information is available indicating whether or not the variable is at the level of the household *(**HH Level**)* and whether it is categorical *(**Categorical**)*.
    - There is also a verbal description *(**Automatic Label**)* which is automatically generated out of the acronyms building the variable's name and the user cannot edit it.
- **To the right-hand side** is a list of all available ***acronyms***, organized into three levels:
    1. The first level is the type as described above (benefit, tax, income, etc.);
    2. The second level subdivides variable types into different categories; and
    3. The third level shows the acronyms themselves together with a verbal description, which is used to generate the automatic labels of the variables. If an acronym is categorical (e.g. gender has two categories, male and female), selecting the acronym lists the respective categories below the list of acronyms.

A number of operations can be performed within the tool, all of which are described in the ***Working with EUROMOD – Administrating variables*** section of the EUROMOD ***Help***. They include adding, deleting, filtering, sorting and searching variables; changing the name, monetary state and country-specific description of a variable; and adding acronyms.

7.3 Intermediate and consumption-related variables

Intermediate variables created on-model

As has been shown, it is also possible to add variables using the ***DefVar*** function. Such variables fall outside the standard EUROMOD variable naming conventions. They are used sparingly and specified by adding ***'i_'*** (for intermediate) at the start of the variable name.

Such intermediate or temporary variables are not included in the model output datasets (except for the purpose of debugging) as per SOUTHMOD Modelling Conventions. This is mainly to reduce the size of the model output datasets.

Expenditure-related variables in the model input data

Consumption-related variables – monetary expenditures and consumption quantities – also do not follow the naming conventions discussed in Section 7.1. *Expenditure variables* used for modelling VAT and excise duties all begin with an ***x*** and the associated *quantity variables* for excise duties begin with ***q***. These are then followed by the COICOP code for the item in question.

Because there are a very large number of these items, the underpinning software has been amended to enable all ***x*** and ***q*** variables to be automatically read into the model from the input data, provided that the box ***Read expenditure related variables*** has been checked in the ***Configure Databases*** dialog box (activated by clicking ***Databases*** in the in the ***Country Tools*** tab). For these cases, the *variable tool* is not required.

8 Modelling in practice

This section explains the practical steps for running a model (8.1), adding a new system (8.2), implementing a policy reform (8.3), and adding new variables (8.4). Analysis of the output data using the Statistics Presenter is discussed in Section 9.

8.1 Running a model

Background

As explained in this User Manual, model output is based on two types of inputs in the case of each model:

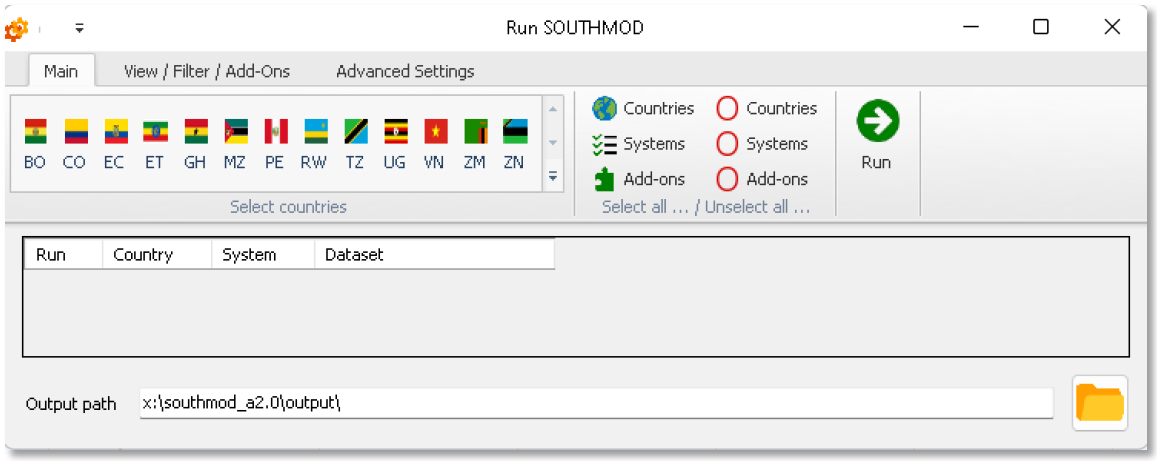
1. Household micro data (***Input dataset*** or ***datasets***); and
2. Rules on how to calculate taxes and benefits (stored in the ***content file*** and specified in the ***user interface***).

Using these two information sources, running the model calculates all modelled taxes and benefits as well as other components (e.g. poverty line, equivalence scale) and saves them under the specified variable names (see parameter output_var). All variables are outputted along with relevant aggregate variables (e.g. income lists for disposable income) and variables from the input dataset as specified in the output policy. The calculations are carried out for each individual and household in the relevant input datasets. The result for each ***policy system***, including a number of existing and simulated variables, is written to an **output file** (e.g. ***ZM_2020_std.txt*** for the 2020 policy system in MicroZAMOD), generated in the ***Output*** folder. The output is at the individual level unless specified otherwise in the output policy.

After the SOUTHMOD bundle has been opened *(**file menu** → **Open Project** →* click on the yellow folder → identify the SOUTHMOD model folder → click ***OK**),* the user can run any of the models in the bundle. Running a model (i.e. simulating the policies that have been modelled) requires clicking the ***Run SOUTHMOD*** button. It is of course possible, although not necessary, to first open a given model by clicking the respective flag.

Figure 8.1 shows the resulting ***Run SOUTHMOD*** dialog.

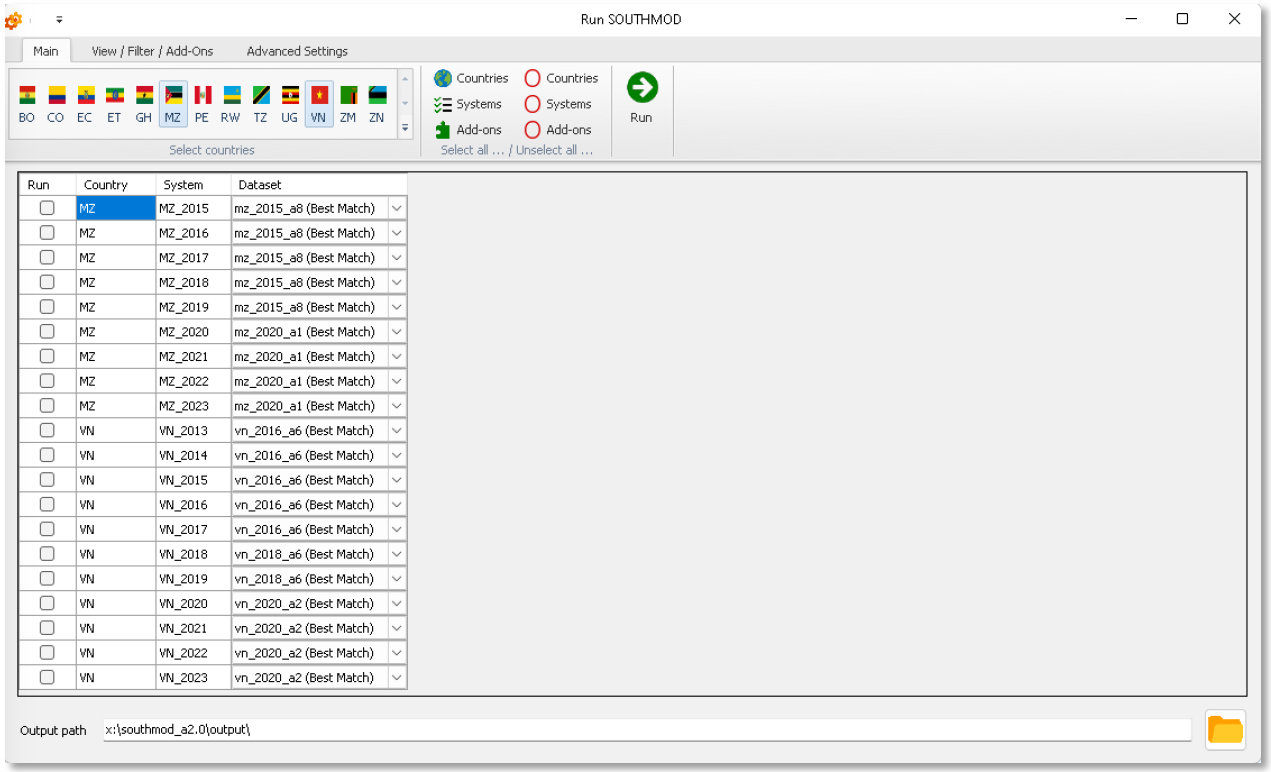
Figure 8.1: The 'Run SOUTHMOD' dialog



Main selections

Now the user can click one of the flags in the ***Main*** menu to open a dialog that includes descriptions of all ***policy systems*** that can be run from the respective model (including relevant ***input datasets*** used). These selections are already visible if the user has opened a particular country before accessing the ***Run SOUTHMOD*** dialog. Figure 8.2 shows the dialog after clicking the flags of Mozambique (MZ) and Viet Nam (VN). One or more countries can be selected.

Figure 8.2: The 'Run SOUTHMOD' dialog: Example from VNMOD and MOZMOD



The dialog includes columns with;

- Check boxes for systems that will be run *(**Run**)*
    - To select a system for running, check the box to the right of the system.
- Countries associated with the systems *(**Country**)*
- Systems *(**System**)*
- Selection menus for associated datasets *(**Dataset**)*
    - The list provides a drop-down menu for each system that contains all available datasets. As a default, these boxes show the best matching datasets.
    - It is however possible to choose another dataset by selecting it from the list.

The field ***Output path*** at the bottom of the dialog defines the folder where the model writes its output. By default this is the ***Output*** folder defined when opening the SOUTHMOD bundle. The folder can be changed by clicking the folder button right of the field to browse and select the appropriate folder, or by typing. Please note that the output folder must exist, otherwise the model issues an error message.

Advanced selections

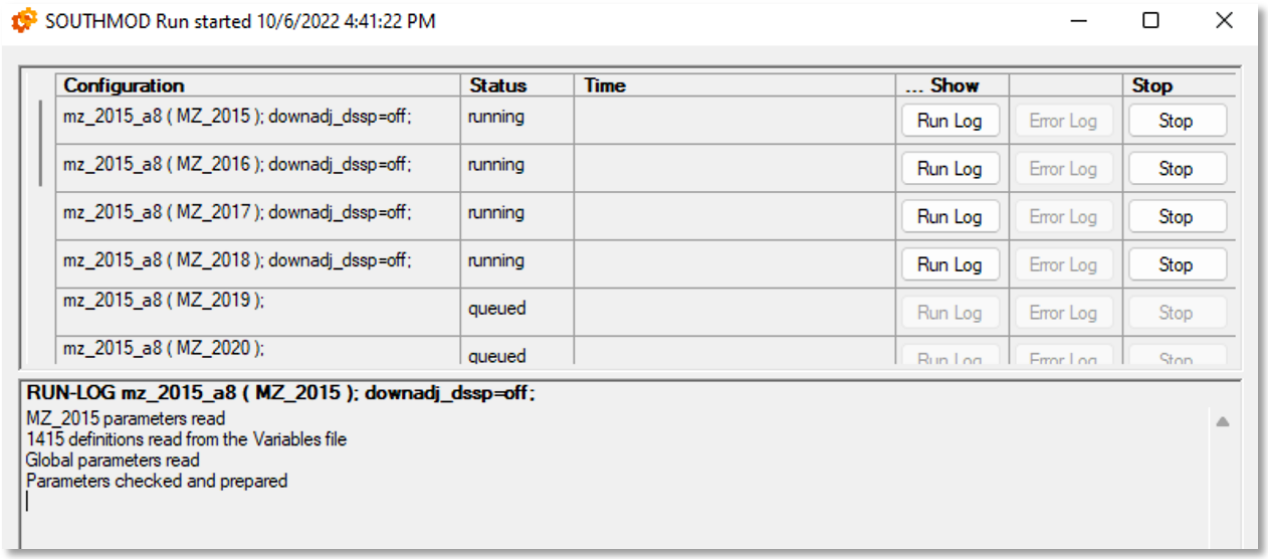
Other options accessible from the ***Run SOUTHMOD*** dialog include the following:

- ***Main*** → ***Countries***: Open (left) or close (right) accessible systems for all countries
- ***Main*** → ***Systems***: Check (left) or uncheck (right) all systems
- ***Main*** → ***Add-ons***: Not used with the SOUTHMOD bundle
- ***View / Filter / Add-ons***: Checking ***Show selected HH options*** allows for running the model only for selected households, specified by first and last household identifiers *(**idhh**)*
- ***View / Filter / Add-ons***: ***Filter Datasets*** and ***Filter Systems*** for filtering
- ***View / Filter / Add-ons***: Check ***Best Match Only*** to only use the best matching input datasets for all systems
- ***View / Filter / Add-ons***: Select any ***Extensions*** to add respective columns for their settings (***on*** or ***off***) next to the relevant systems in the main dialog (see also 'Extensions: Switches' in Section 4.4)
- Further options are available in the Advanced ***Settings*** tab.

Running the model

Once the required selections have been made (namely the desired systems), click on the ***Run*** button to start the simulation process. A window appears (see Figure 8.3 for an example), providing information about the progress of the run and allowing for some manipulation.

Figure 8.3: The 'SOUTHMOD Run started' dialog: Example from MOZMOD



Note the following:

- All system-dataset combinations selected for running are listed and their status is shown as either ***running***, ***queued***, ***finished*** or ***aborted*** (either by the user or due to an error).

- Once a run is started, its starting time is displayed, and once it is finished (or aborted), the finishing time is indicated as well, together with the time taken. The total time needed by the simulation depends on:
    1. The processing speed of the computer,
    2. The size of the dataset (i.e. how many households must be processed); and
    3. The extensiveness of the system (i.e. the number and complexity of implemented taxes and benefits).

- There are three buttons for each run:
    1. The ***Stop*** button enables the user to abort the run.
    2. Once a run is started, the ***Run Log*** button is activated. If it is clicked, the field below the list of runs shows progress information.
    3. If a run produces an error (stopping the run) or a warning (allowing the run to continue) the ***Error Log*** button is activated. Clicking the button shows the run's warnings and/or errors in the field below the list of runs.

- Note that the content of this field is determined by the most recently clicked ***Run Log*** or ***Error Log*** button – its heading indicates what is currently displayed.
    - If any warnings or errors are issued, an error log file is generated in the ***Output*** folder, named ***yyyymmddhhmm_errlog.txt*** (e.g. ***202212081530_errlog.txt***).
    - The information window will stay open until it is closed by the user, even if all runs are finalized, to allow possible error logs to be checked and to inform about the times taken.

- If the user closes the window before all runs are finished (after a warning), the still active runs are aborted and the queued runs are taken from the queue. To hide the window, use the ***Minimize*** button.

After calculations have been finished

When EUROMOD has finished its calculations, the output is stored as one or more text files in the place of storage defined in the field ***Output path*** (generally the **output** folder in the main SOUTHMOD model folder).

Output files can be viewed in a text editor program, such as Notepad, or imported into any statistical analysis package, for example STATA, for more detailed analysis. The output can also be viewed in Excel by using the in-built tool ***Open Output File*** (accessed from the ***Applications*** tab). The output data can also be analysed using the ***Statistics Presenter*** (See Section 9).

In addition to the ***output file***, the model produces a ***header file*** relating to the output file, named ***yyyymmddhhmm_EMHeader.txt*** (e.g. ***202212081530_EMHeader.txt***). The header file contains the following information:

- System;
- Database;
- EUROMOD version number;
- User interface version number;
- Executable version number;
- Start date and time;
- End date and time;
- Name and location of the output file;
- Currency; and
- Exchange rate.

Additional systems in all country models will need to be added for future years. In addition, there may be a need to test the impact of reforms to the current tax and benefit rules. These two related tasks are discussed next.

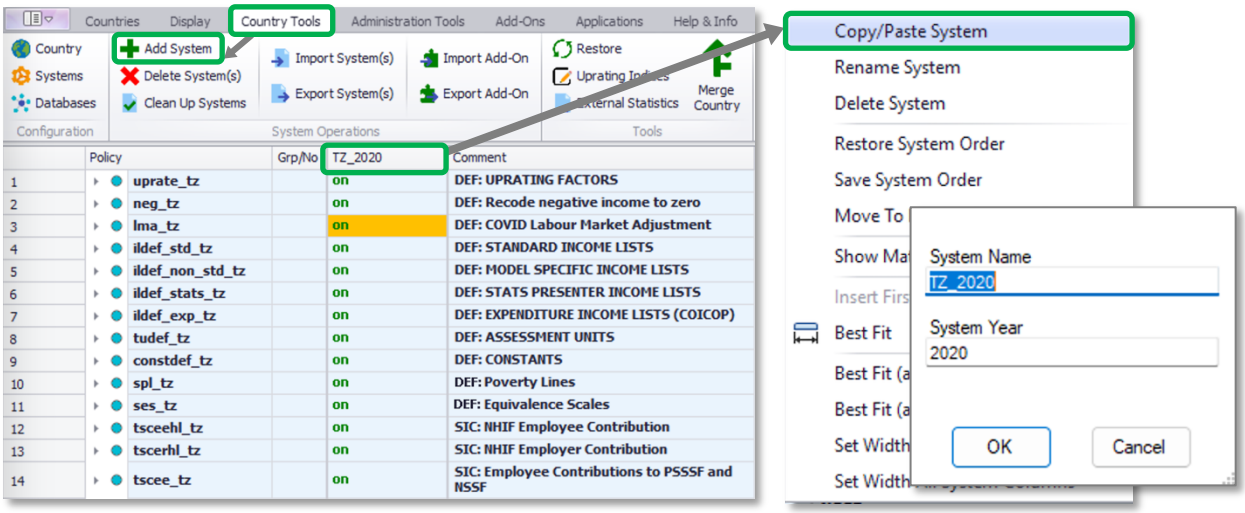
8.2 Adding a new system

Two options of adding a new system

Each model in the SOUTHMOD bundle allows the user to add a new ***system***. As shown in Figure 8.4, there are two ways to achieve this:

- Click on the ***Add System*** button in the ***Country Tools*** tab; or
- Right-click on the header of an existing ***system*** and select ***Copy/Paste System***.

Figure 8.4: Two options of adding a new system



Both options open a dialog which asks for the new ***system name*** and the ***system year***. For instance, a *system* for 2021 in TAZMOD created based on the 2020 system could – and should – be named TZ_2021. The *system year* in this example would be 2021.

Note that the *system's* name must not contain any other characters than letters, numbers and underscores. If any other character is used, or if the chosen name is equal to an existing *system's* name, an error message is issued and you are asked to change the name. Clicking ***OK*** adds the new *system*.

Differences between base systems and their copies

In the first instance, the new ***system*** is almost an exact copy of the base ***system*** and is automatically configured to run with the same *dataset* as in the base *system*.

There is only one small difference between base *systems* and their copies: the ***Add System*** tool changes the names of the **output files** for the new *system* to reflect the new *system's* name. For instance, the output file for system ***GH_2022*** in GHAMOD is called ***GH_2022_std.txt***, but in a new *system*, say ***GH_2023***, the name of the output file would be ***GH_2023_std.txt***.

As discussed in Section 4.3 on the options in the ***Display*** tab, the user interface allows the differences between a base and a derived system to be highlighted, using background or text color (***Display*** → ***Automatic Conditional Format*** button). This is very useful in highlighting the changes introduced by a reform (see Section 8.3). Alternatively you can specify your own ***Conditional Formatting***.

After the new system has been created (e.g. ***GH_2023*** based on ***GH_2022***), it is possible to make changes to ***parameters*** in the new system. In the example, this may include adjusting tax rates or benefit amounts to reflect any values that have changes from 2022 to 2023.

As discussed in Section 8.3 below, the new *system* may involve a planned or hypothetical reform to tax-benefit policies in an existing *system*. In this case, this new system (for instance based on ***GH_2022***) might be called ***GH_2022_reform*** or similar.

Deleting a system

The models also allow for the user to *delete* a new ***system*** in two ways:

- Click on the ***Delete System(s)*** button in the ***Country Tools*** tab, which opens a dialog *(**Select Systems**)* where the system(s) to be deleted can be selected; or
- Right click on the header of an existing ***system*** and select ***Delete System***.

Note the warning that *systems* will be deleted permanently. Take great care when using the ***Delete System*** tool in order to avoid inadvertently deleting a system that you did not intend to delete.

8.3 Implementing a policy reform

Background

Understanding how to correctly implement a policy reform is a crucial part of using models in the SOUTHMOD bundle. The process involves two broader steps, which are described below along with supplementary guidelines:

1. The process of implementing a reform should begin with adding a new ***system*** to a given model (as described above in Section 8.2).
    - Continuing from the previous example, the new system could be called ***GH_2022_reform***, for example, and created as a copy of an existing 2022 system in GHAMOD *(**GH_2022**)*.
    - When implementing a policy reform, remember that a *system* is a collection of tax-benefit policies that apply to a particular point in time. For example the ***GH_2022*** *system* in GHAMOD records the policy rules in existence in 2022 (along with relevant uprating), which is also carried on to the reform system as it is the copy of the original..

2. The next step is making changes to the reform ***system***, including for instance changes in ***parameter*** values, or adding new ***functions*** or ***policies***.
    - Such changes may reflect a past, planned or hypothetical policy reform adopted in the respective country.
    - Note that all changes should be made only in the reform *system.* There are two reasons for this:
        - First, this allows for simple comparisons of rules (e.g. parameter values) with the base *system*, which remains unaffected. Simulations can also be easily run and then compared between both systems.
        - Second, if changes were made to particular policies in an existing system, the record of *actual* policy parameters in place at a particular time point would be lost. It would not be possible to run simulations on the existing system without undoing the changes made in the reform scenario. While it is possible to also make changes to an existing system, it is not recommended.

The exact process of implementing a policy reform will then depend on whether:

1. The reform is a change to an existing policy (e.g. amending a means test threshold or grant amount, removing a means test, or varying the means test amount by certain criteria); or
2. The reform involves the introduction of a new policy.

The possible options are too many and varied to explain in detail here. Instead, the next subsection describes the main operations that will be required when implementing a policy reform.

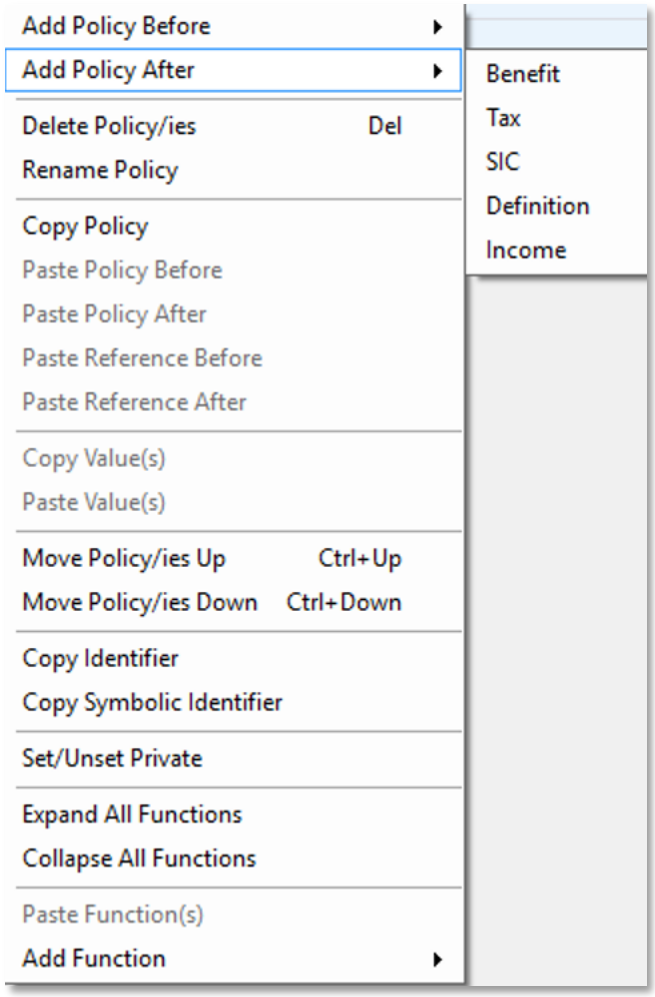
Adding a policy

The following steps are required to add a ***policy*** in any model:

1. Right click on the name of an existing *policy* that is located either before or after which you want to insert the new policy.
2. This opens the policy's context menu, where you select the menu item ***Add Policy Before*** or ***Add Policy After*** (see Figure 8.5).

3. A sub menu opens where the type of the new policy that is added can be chosen (***Benefit***, ***Tax***, ***SIC***, ***Definition*** *or* ***Income***).
4. Click on the respective *policy* type to open a dialog where you are asked to indicate the new *policy's* name. Clicking ***OK*** adds the new *policy*.
    - Note that the *policy's* name must not contain any characters other than letters, numbers and underscores. If any other character is used or if the chosen name is equal to an existing policy's name, an error message is issued, and you are asked to change the name.
    - Moreover, the *policy* name is by convention expected to end with ***_cc***, where ***cc*** reforms the relevant country abbreviation. If this is not the case, the user interface asks whether it should add this ending for you. While you may answer this question with ***No*** to use a non-standard policy name, it is recommended to answer with ***Yes***.

Figure 8.5: The 'Add Policy' Tool



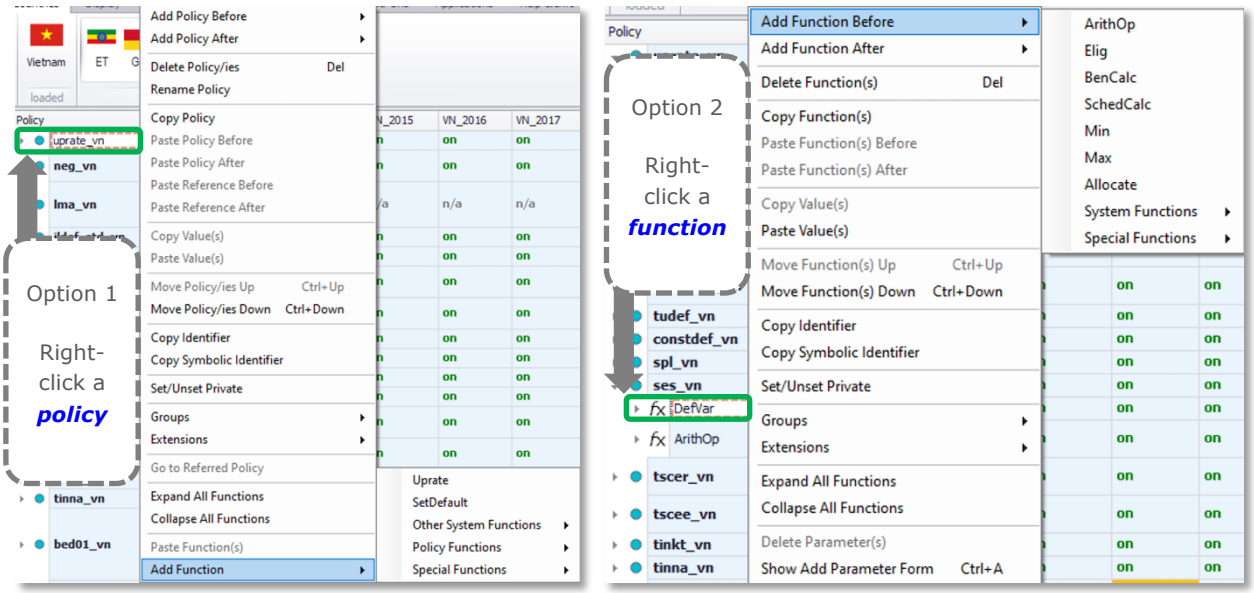
Adding a function

There are two alternative options for adding a ***function*** in a selected *policy:*

1. Right-click on the name of the ***policy*** in which you want to add a function. This opens the policy's context menu with an item ***Add Function*** (see the left panel in Figure 8.6). There are additional sub-menus for instance for different ***Policy Functions*** and menu items for two system functions *(**Uprate*** and ***SetDefault**).* Using this menu adds the selected *function* as the very last *function* of the *policy*.
2. Right-click on the name of the ***function*** either before or after which you want to insert the new *function* (the right panel in Figure 8.6). This opens the *function's* context menu, where you select either of the menu items ***Add Function Before*** or ***Add Function After***. A sub-menu opens where the function can be chosen. Note that this sub-menu is slightly different depending on the *policy* that the new *function* will be part of.

In both cases, clicking the respective *function* add both the ***function*** itself as well as the ***compulsory parameters*** of this *function* (for most functions, ***TAX_UNIT*** and ***Output_var***).

Figure 8.6: The 'Add Function' Tool from menus of a policy and a function

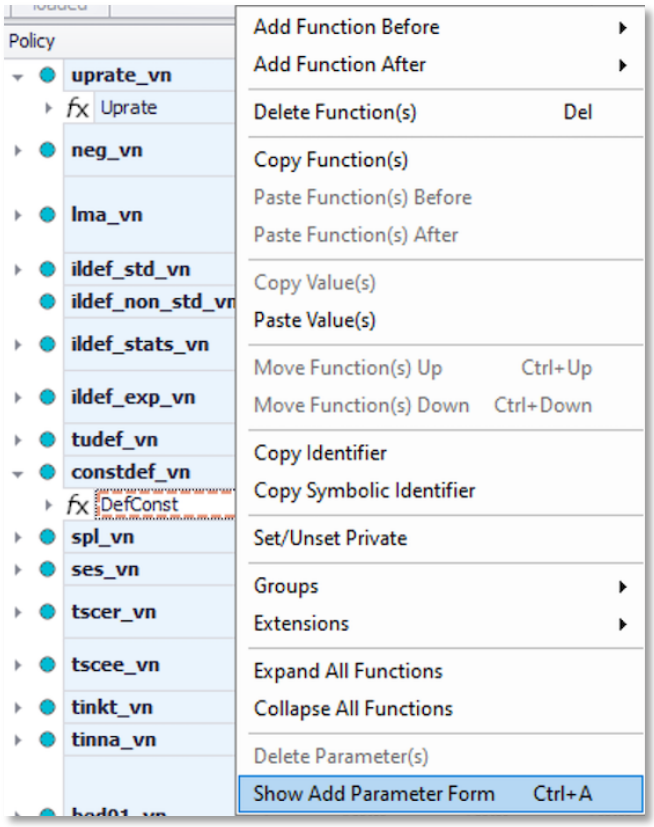


Adding a parameter

The following steps are required to add a ***parameter*** in any relevant function:

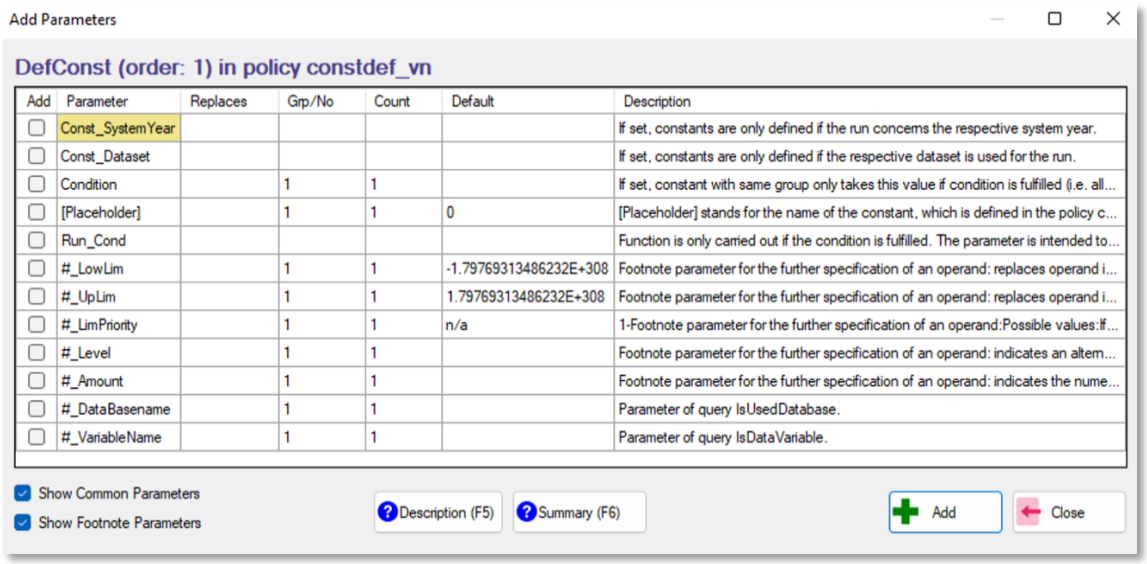
1. Right-click on the respective ***function*** to open its context menu. Select the menu item ***Show Add Parameter Form*** (see Figure 8.7).

   Figure 8.7: Accessing the 'Add Parameter' form



2. The form that opens (Figure 8.8) shows all ***parameters*** that can be added to the ***function***. The form includes:
   - Boxes that need to checked to add a given parameter *(**Add** column)*;
   - Name of the parameter *(**Parameter**)*;
   - Optional cells where you can specify which parameter is replaced *(**Replaces**)*;
   - Optional cells where you can specify a group number *(**Grp/No**)*;
   - Cells where you can specify how many of each parameter are added *(**Count**)*; note that the parameter is added once by default;
   - Default value of the parameter when added *(**Default**)*; and
   - The description of each parameter *(**Description**)*.

Figure 8.8: The 'Add Parameter' form: DefConst in VNMOD



3. Note that the form observes what the user is doing and adapts its content respectively. Relatedly, the form does not list ***compulsory parameters*** of the respective *function* (e.g. the parameter ***TAX_UNIT***), as they are automatically a part of the function and do not need to be separately added. Exceptions to this rule are 'special' *parameters* such as:
    - Those which can be added more than once (e.g. *parameter* ***Var*** of *function* ***DefOutput*** or the *parameter* ***Comp_Cond*** of function ***BenCalc***); and
    - Those which have 'aliases' (e.g. the *parameter* ***Output_Var*** with its alias ***Output_Add_Var***).

4. To add one or more *parameters* to the *function*, select them by ticking the corresponding check boxes. Then click on the ***Add*** button with the green plus.
    - It is also possible to add *parameters* in bulk (in cases where more than one incidence is allowed, such as the *parameter* ***Var*** of the *function* ***DefOutput***). This can be achieved by amending the relevant cells under the ***Count*** column.
    - Note that if in the main view a *parameter* is selected, new *parameters* are added after this *parameters*. If instead a function is selected (as in Figure 8.7), new parameters are added at the end of the function.

To understand the full functionality of the ***Add Parameter Form*** see the ***Working with EUROMOD – Changing countries' tax-benefit systems – Adding parameters*** section of the EUROMOD ***Help*** (accessed from the ***Help & Info*** tab).

Amending income lists

When the user has introduced a new policy, it is necessary to amend certain income lists:

1. If the new policy is a new benefit:
    a. One of the standard (and mutually exclusive) income lists ***ils_pen***, ***ils_benmt***, or ***ils_bennt*** needs to be amended to include the new benefit output variable.
    b. Additionally, the benefit must also be added to one of the following (again mutually exclusive) income lists: ***ils_pen*** (as above), or one of ***ils_bch***, ***ils_bsa***, ***ils_bsu***, ***ils_bdi***, ***ils_bun***, and ***ils_bag***.
2. If the new policy is a direct tax, the new output variable must be added to the income list ***ils_tax.***
3. If the new policy is a new indirect tax, the new output variable must be added to the income list ***ils_taxind.***
4. If the new policy is a social insurance contribution, the new output variable must be added to income list to one of the related (mutually exclusive) income lists: ***ils_sicee***, ***ils_sicse***, *or* ***ils_sicer***.

More detailed accounts of income lists can be found in Sections 6.2 (Income lists) 9.3 and (Compulsory information required by the ***Statistics Presenter***) as well as in Section 11 of the SOUTHMOD Modelling Conventions.

For further information on adding policies, functions and parameters, see the ***Working with EUROMOD – Changing countries' tax-benefit systems*** section of the EUROMOD ***Help*** (accessed from the ***Help & Info*** tab).

Hiding systems

Finally, to make the process of implementing a policy reform – or simply working with an existing system – more manageable, it is sometimes helpful to concentrate only on the reform ***system*** and perhaps also its base ***system***. To facilitate this, *systems* can be hidden as follows:

- Right click on a *system's* header and then move the mouse over the menu item **Move to Hidden Systems Box...** This reveals various sub-menu items for hiding and unhiding *systems* from the main view.
- *Systems* hidden from the main view are listed in the ***Hidden Systems Box***. This is a small window, which is displayed by choosing the sub-menu item ***Show Hidden Systems Box*** or any of the other sub-menu items except ***Unhide All Systems***.
- The sub-menu item ***Unhide all Systems*** redisplays all hidden *systems* (i.e. the *systems* listed in the ***Hidden System Box***).
- To redisplay a single *system*, double click on the *system* in the ***Hidden System Box***.
- It is also possible to hide and unhide a *system* by dragging it into the ***Hidden System Box*** or by dragging it from the ***Hidden System Box*** to its old or any other position.

Note that the user interface draws attention to any changes which could affect hidden systems.

8.4 Adding new variables

As part of implementing a reform or a new *system* where there has been a change to a policy, it may be necessary to incorporate new *variables*. Please note that it is good practice to check available variables already included in the variable menu first and to use the name of an already existing variable if possible. Related guidelines are listed below:

1. As described earlier in Section 7.2, the ***variable description file*** *(**VarConfig.xml**)* is accessed via the button ***Variables*** in the ***Administration Tools*** tab.
2. In order to add a *variable*, click on the button ***Add Variable*** in the top left-hand corner of the ***Variables*** tab. Alternatively, press the keys ***Alt*** and ***V*** simultaneously.
3. This adds an empty row to the list of *variables*.
    a. Initially the row is added below the selected row.
    b. Re-sorting the list (manually or by an automatic update due to another change) moves empty rows to the beginning (ascended sorting) or end (descended sorting) of the list of variables.
4. The *variable* name can be entered by typing directly into the cell of the ***Name*** column, using the listed acronyms.
    a. If the required acronym is not in the list, it can be added using the buttons in the ***Acronyms*** tab.
    b. The ***Monetary*** box is checked by default but can be unchecked.
    c. The ***Automatic Label*** is automatically generated from the acronyms used in the name of the variable and cannot be edited.
    d. A country-specific description of the variable can be added by typing directly into the ***Description*** cell.
5. Remember to save the file before closing.

9 Using the Statistics Presenter

The ***Statistics Presenter*** is an extremely powerful and flexible tool that allows for immediate analysis of the output data from SOUTHMOD. The tool calculates the costs of benefits and the amount of taxes simulated for the government and produces estimates of both consumption-based and income-based poverty and inequality levels, taking into account simulated taxes and social transfers. ***Statistics Presenter*** also allows comparisons between two or more systems.

9.1 Access and selections

The ***Statistics Presenter*** is accessed from the ***Applications*** menu by clicking on the ***EUROMOD Statistics*** button and selecting the ***Statistics Presenter*** tool from the opening dialog (Figure 9.1). After this – and after also running the policy systems for relevant country models – follow the steps below to obtain summary results:

1. In the first dialog (Figure 9.2), select either:
    - ***Southmod Statistics – Default*** to show results for individual policy systems at a time; or
    - ***Southmod Statistics – Baseline/Reform*** to show comparisons between different policy systems, an option that is particularly useful when examining the impact of policy reforms. Click ***OK*** to accept the selection.

2. In the second dialog, choose the possible ***output datasets*** to be analysed:
    - Following the ***Default*** selection (left panel in Figure 9.3), one panel opens where you can select the desired datasets.
    - Following the ***Baseline/Reform*** selection (right panel in Figure 9.3), two panels open where you can select both the baseline dataset and one or form reform datasets. The result then takes the form of a comparison between the base system (i.e. the output file selected in the leftmost part of the dialog box shown above) and the comparator system or systems.
    - Apart from the baseline dataset under the ***Baseline/Reform*** selection, one or more may datasets can be selected in these panels. To select more than one, use the control key and mouse click to select the desired output files.
    - Click ***OK*** to accept the selections.

3. The third dialog enables the user to select whether poverty and inequality statistics in the ***Statistics Presenter*** should be based on consumption or income (Figure 9.4).
    - For most models in the SOUTHMOD bundle, consumption-based outcomes are generally preferred, as they follow national definitions. Post-fiscal definitions ('net of direct taxes') are also available, although not commonly used.
    - Click ***OK*** to accept the selection.

4. After the user has selected the distribution statistics used, the ***Statistics Presenter*** will begin to process the results, displaying gear wheel whilst it is doing so.

Figure 9.1: Opening Statistics Presenter

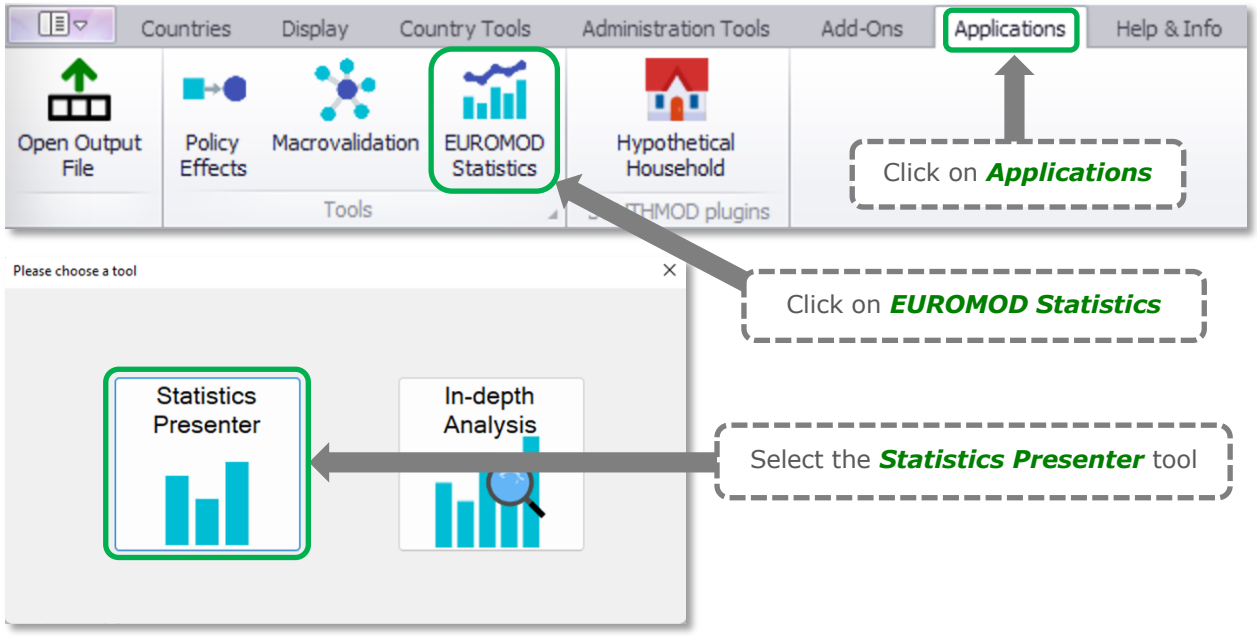


Figure 9.2: Selecting a Statistic Template

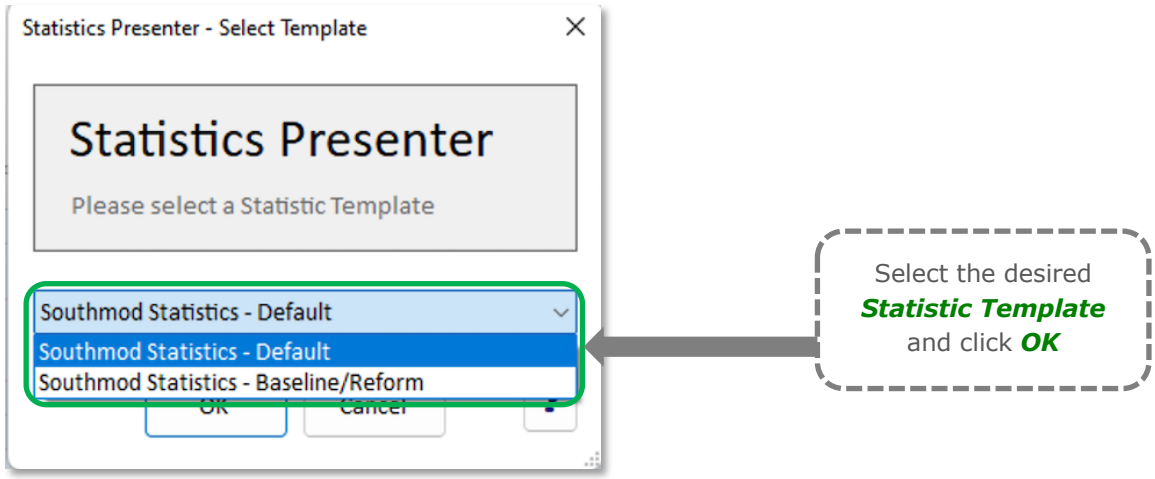


Figure 9.3: Selecting output datasets to use in Statistics Presenter

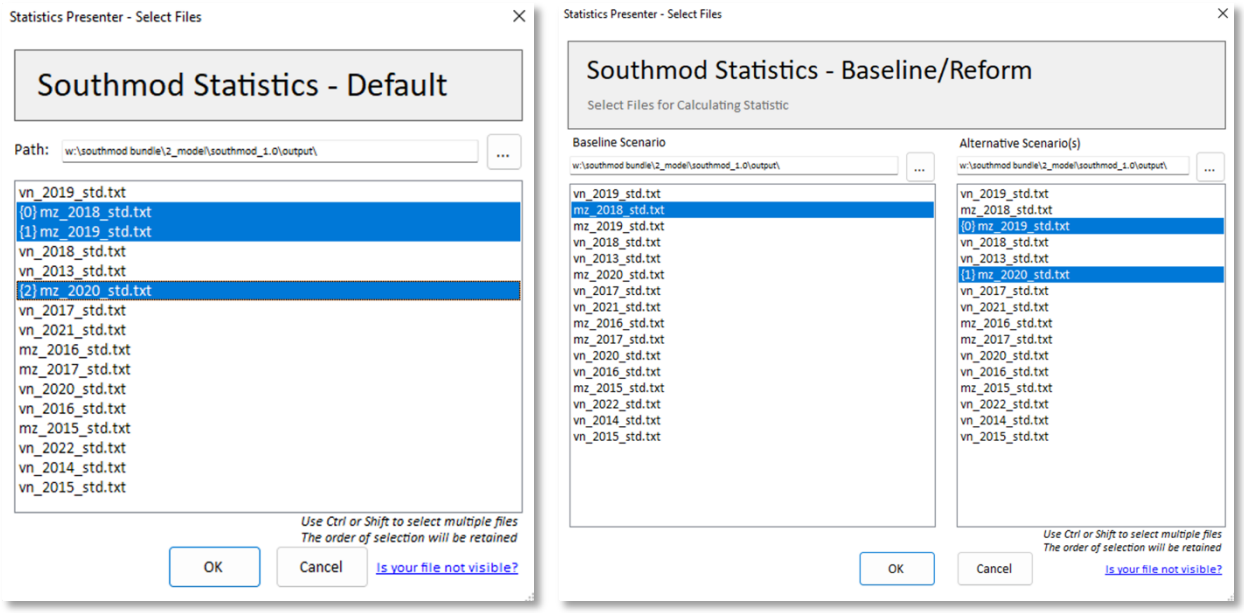
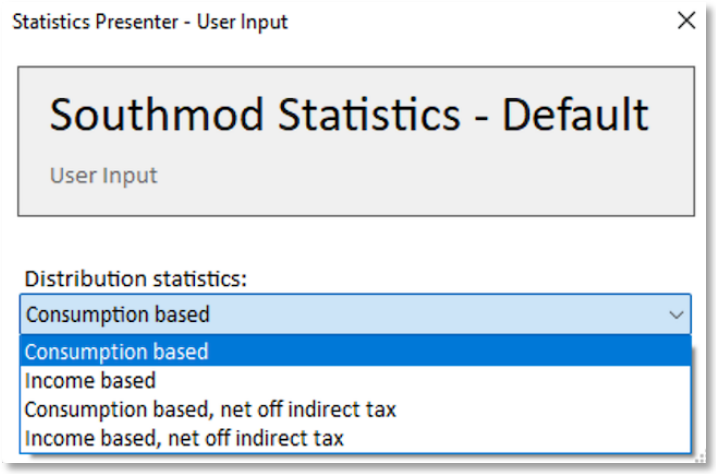


Figure 9.4: Selecting distribution statistics to use in Statistics Presenter



9.2 Results

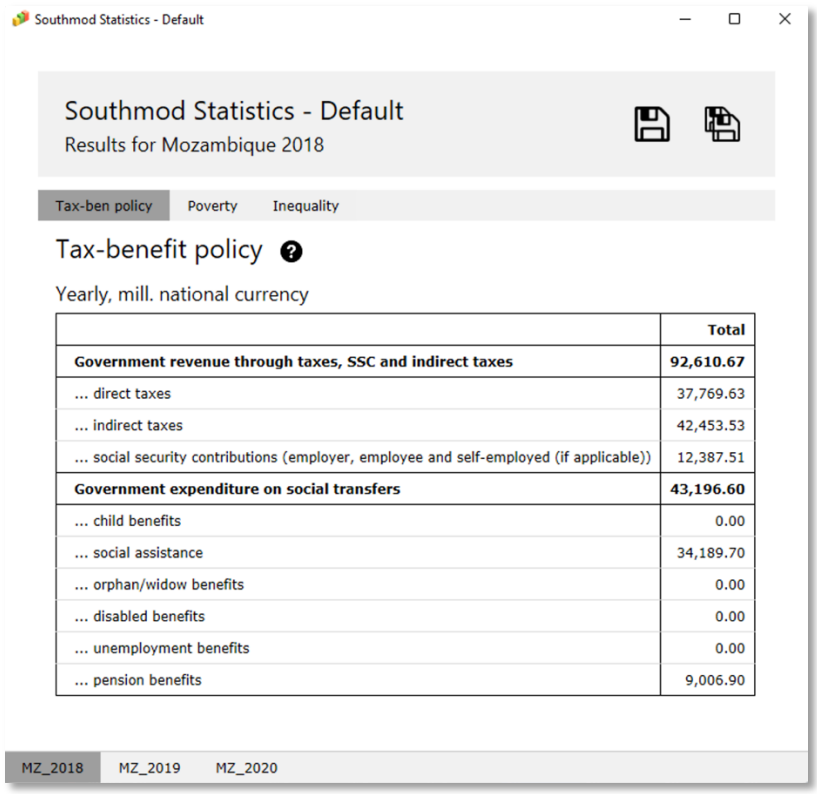
After calculations are complete, it is possible to analyse the results using the tool:

1. The initial screen shows the ***Tax-benefit policy*** panel. It presents information on modelled annual government revenue through taxes (direct and indirect) and social security contributions. It also displays annual government expenditure on various kinds of social transfer. What is displayed under different headings is controlled through ***income lists*** (discussed above). Note that:
    - Following the ***Default*** selection, estimates are shown for a single *system* (one of the selected output datasets), as seen in Figure 9.5. Results for other *systems* (if the user had selected more than one output file) can be accessed by clicking one of the other *system* names at the bottom of the screens.

- Following the ***Baseline/Reform*** selection, estimates are shown for two or more systems (baseline and reform datasets) at once, as seen in Figure 9.6. The final output also shows the difference between the base system and the comparator system or systems.

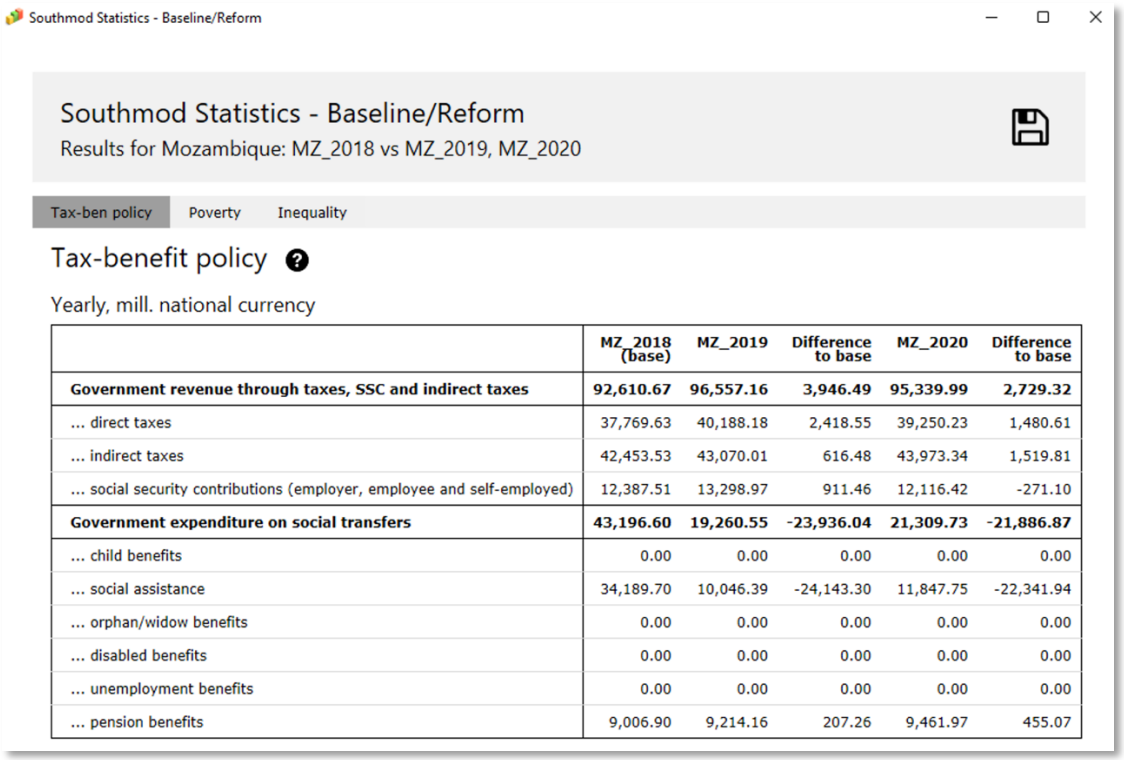
2. The user can also access summary output for ***Poverty*** and ***Inequality*** indicators using the associated tabs to the right of the ***Tax-benefit policy*** panel.
    - The ***Poverty*** panel shows the percentage of the population in poverty. This is also shown broken down by different types of households. The ***Poverty*** panel also displays the average normalized poverty gap, FGT(1). The poverty line used in the ***Poverty*** panel (reported in the last row of information) is specified in a special policy in each model *(**spl_cc**)* and can be modified as needed (see Section 6.3).
    - The ***Inequality*** panel shows the Gini coefficient, the P80/P20 ratio (the ratio of the income of those at the 80th percentile of the distribution compared to the income of those at the 20th percentile). The income at each quintile of the income distribution is also shown.
    - As with the ***Tax-benefit policy*** panel, the ***Poverty*** and ***Inequality*** panels show results and comparisons between two or more systems following the ***Baseline/Reform*** selection.

Figure 9.5: Initial screen in Statistics Presenter (Default): Tax-ben policy panel



	Total
Government revenue through taxes, SSC and indirect taxes	**92,610.67**
... direct taxes	37,769.63
... indirect taxes	42,453.53
... social security contributions (employer, employee and self-employed (if applicable))	12,387.51
Government expenditure on social transfers	**43,196.60**
... child benefits	0.00
... social assistance	34,189.70
... orphan/widow benefits	0.00
... disabled benefits	0.00
... unemployment benefits	0.00
... pension benefits	9,006.90

Figure 9.6: Initial screen in Statistics Presenter (Baseline/Reform): Tax-ben policy panel



	MZ_2018 (base)	MZ_2019	Difference to base	MZ_2020	Difference to base
Government revenue through taxes, SSC and indirect taxes	**92,610.67**	**96,557.16**	**3,946.49**	**95,339.99**	**2,729.32**
... direct taxes	37,769.63	40,188.18	2,418.55	39,250.23	1,480.61
... indirect taxes	42,453.53	43,070.01	616.48	43,973.34	1,519.81
... social security contributions (employer, employee and self-employed)	12,387.51	13,298.97	911.46	12,116.42	-271.10
Government expenditure on social transfers	**43,196.60**	**19,260.55**	**-23,936.04**	**21,309.73**	**-21,886.87**
... child benefits	0.00	0.00	0.00	0.00	0.00
... social assistance	34,189.70	10,046.39	-24,143.30	11,847.75	-22,341.94
... orphan/widow benefits	0.00	0.00	0.00	0.00	0.00
... disabled benefits	0.00	0.00	0.00	0.00	0.00
... unemployment benefits	0.00	0.00	0.00	0.00	0.00
... pension benefits	9,006.90	9,214.16	207.26	9,461.97	455.07

Finally, on each result panel, there are buttons (a disk icon, or two icons following the ***Default*** option) to enable the user to export the results. Often the option with a single disk icon is the most useful as it will export the tables on each of the panels to Excel.

9.3 Compulsory information

There are two sets of information that the ***Statistics Presenter*** requires in order to function correctly: (1) Income lists generated in the model, and (2) variables included in the input datasets.

Compulsory income lists and variables generated in the model

The first set of information consists of income lists generated and contained in the model. Some (although not most) of these income lists require alteration in certain circumstances. The subsection on 'Statistics Presenter income lists (ildef_stats_cc)' in Section 6.2 covers the compulsory income lists needed for the Statistics Presenter to function correctly. There are also two income lists, ***ils_tax*** and ***ils_pen***, that are part of the standard income list policy but also required by the Statistics Presenter. More information on is also available in the SOUTHMOD Modelling Conventions (pages 27–28).

While most of these income lists will never require amendment, some will, particularly if a new policy is introduced:

- Income list ***ils_tax*** (from the standard income list policy) will only require amendment if new direct taxes simulated.

- Income list ***ils_taxind*** will only require amendment if new indirect taxes simulated (not for instance if existing VAT or excise duties, included in the list, are amended).
- Income list ***ils_sic*** should not be amended. However, one of income lists ***ils_sicee***, ***ils_sicer*** or ***ils_sicer*** (from the standard income list policy) will require amendment if a new social insurance contribution are simulated.
- The mutually exclusive income lists ***ils_pen*** (from the standard income list policy), and those for different types of benefits *(**ils_bch**, **ils_bsa**, **ils_bsu**, **ils_bdi**, **ils_bun**,* and ***ils_bag**)* will need to be amended if a reform scenario is simulated that introduces a new benefit, or when a new benefit is introduced by the government.
    - It is important to note that these 7 lists are mutually exclusive. A simulated benefit can only occur in one of the lists. So, for example, if a disabled child benefit was simulated, a decision would need to be made whether to insert this into the child benefit income list or the disability benefit income list.
- Income lists ***ils_dispyx*** and ***ils_con*** are used to calculate inequality and poverty outcomes, and should not be amended by the user; new benefits, taxes or social contributions (that influence these income lists) should be made to the 'component' lists, as per above instructions. This also applies to income lists ***ils_dispyx_pf*** and ***ils_con_pf**,* which are used to calculate post-fiscal inequality and poverty outcomes.

Certain general guidelines should also be followed when amending income lists:

- One income list should include either (a) the benefit amount simulated in the model or (b) the variable on benefit amounts collected directly from the data, but never both together. The simulated amount is generally used if both are available.
- No variable should be double-counted in income lists. For instance, a benefit component cannot be included both alone and as a part of an aggregated variable.
- Income lists should include detailed incomes, benefits or taxes rather than their aggregates (here referring to variables, not other income lists). For example, assuming there are two unemployment benefits and both need to be included, it is better to have each component separately rather than their aggregate.

Additionally, an equivalence scale variable (***ses***, the equivalence scale in adult equivalents) must be modelled in policy ***ses_cc***, and poverty lines (***spl***, and ***splpf***, i.e. the post-fiscal line) must be modelled in policy ***spl_cc***. Both are needed for poverty and inequality measurement. See 'Defining poverty lines (spl_cc)' and 'Defining equivalence scales (ses_cc)' in Section 6.3 for modelling details.

Compulsory variables included in the input data

Second, certain information is generated when the baseline data is prepared and forms part of the data preparation process. Such information is included in the input data and will not need to be altered by the user. For the courtesy of the users, however, the required variables are listed in Table 9.1.

Table 9.1: Compulsory information in the input data required by the Statistics Presenter

Variable	Type
xhh	Household consumption in the base year, for calculating consumption-based poverty measures
dhh	Set to 1 for household head, and 0 for other members
xivot	Value of home-grown produce where available Added to income to estimate income-based poverty This will be set to 0 when not known or where it is desirable to estimate income properly without taking into account home-grown produce
Social benefits	Variables representing social benefits reported in data (where applicable and not simulated)
Direct taxes	Variables representing direct taxes reported in data (where applicable and not simulated)

10 Summary

The SOUTHMOD model bundle and relevant options can be summarized as follows:

1. The SOUTHMOD model bundle allows for accessing and working with several tax-benefit microsimulation models developed by UNU-WIDER, SASPRI and national partners. The models can be accessed by opening model bundle in the EUROMOD software, and then clicking the relevant country flag.
2. The central access point of each model is the main EUROMOD user interface, from where the content files can be accessed. The content files store the information that the model needs for its calculations and provide other tools and applications.
3. For each model, the content files contain information required for both the implementation of the framework of the tax-benefit model, and for the implementation of the particular policies which comprise the tax-benefit system. This information is mainly contained in ***policies*** displayed in the ***policy spine***.
4. ***Functions*** are used as building blocks of both definitional and tax-benefit policies. Functions are comprised of ***parameters***.
5. Each model is underpinned by one or more ***input datasets***, which represent the population in the respective country. These datasets come from national household budget surveys and include demographic and labour market information as well as information on incomes and expenditures.
6. A single input dataset (e.g. from 2021) can be used for several ***policy systems or years*** (e.g. 2021 and 2022) in case subsequent datasets are not available. An older dataset is used for simulating several policy years by uprating monetary values to the corresponding policy year. For each system, there is a dataset that is the 'best match', normally the one whose collection year is nearest to the policy year.
7. ***Tax-benefit policies*** include several policies that apply user-coded legislative rules to the model input data to calculate and simulate selected outputs. For instance, a policy on income tax simulates individual tax liabilities, while a pension benefit policy may simulate individual pension entitlements. Social insurance contributions by employers and employees are also modelled.
8. ***Definitional policies*** include those for applying adjustments to the input data (e.g. ***uprate_cc*** for uprating factors for monetary variables); and those for defining income lists (e.g. ***ilsdef_cc***), assessment units (***tudef_cc***), constants (***constdef_cc***), and model outputs (***output_std_cc)***.
9. Income lists, included in ***ilsdef_cc*** and other such policies, are aggregates of variables defining for example disposable income, taxable income, etc. Income lists are used in tax and benefit policies for the implementation of the respective tax or benefit.
10. Information relating to ***Country*** settings (e.g. names), ***System*** settings (e.g. currency) and the input ***Datasets*** is contained and modified within ***Country Tools***.

11. The ***variable description file*** (VarConfig.xml) in the model folder contains descriptions of all variables available in the bundle.

12. To run the model and analyse outputs:

    i. Enter the user interface and click on the button ***Run SOUTHMOD*** in the top left corner.
    ii. Select the flags for the country models you wish to run.
    iii. Check the boxes for the ***system-dataset*** combinations of choice. Also ensure that the path to the output file is correct.
    iv. Click on ***Run***.
    v. When the simulations have finished running use a text or statistical package such as STATA to explore the output files, or utilize the ***Statistics Presenter***.