SOUTHMOD

User manual

Mozambique

MOZMOD v2.0

Vanda Castelo and Finorio Castigo, Michael Noble, Gemma Wright and Christine Byaruhanga (SASPRI)

March 2018

República de Moçambique
Ministério da Economia e Finanças

University of Essex

UNITED NATIONS
UNIVERSITY
UNU-WIDER

Contents

Acknowledgements

United Nations University World Institute for Development Economics Research (UNU-WIDER) is thanked for funding the building of the model and the preparation of this manual. The manual was produced as part of SOUTHMOD, a major research project in which tax-benefit microsimulation models for selected developing countries in Africa (Ethiopia, Ghana, Mozambique, Tanzania, Zambia, Uganda) and also elsewhere (Ecuador and Viet Nam) are built in addition to those that already exist for South Africa and Namibia. SOUTHMOD is a collaboration between UNU-WIDER, the EUROMOD team at the Institute for Social and Economic Research (ISER) at the University of Essex, and Southern African Social Policy Research Insights (SASPRI).

1. Introduction

1.1 About MOZMOD

Tax-benefit microsimulation models, which combine representative household-level data on incomes and expenditures and detailed coding of tax and benefit legislation, have proven to be an extremely useful tool for policymakers and researchers alike. The models apply user-defined tax and benefit policy rules to micro-data on individuals and households and calculate the effects of these rules on household income. The effects of different policy scenarios on poverty, inequality, and government revenues can be analysed and compared.

Mozambique, like other developing countries, is now building up its social protection system 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.

Against this backdrop UNU-WIDER, the EUROMOD team at the Institute for Social and Economic Research (ISER) at the University of Essex, and Southern African Social Policy Research Insights (SASPRI) have launched SOUTHMOD, a major research project in which tax-benefit microsimulation models for selected developing countries in Africa (Ethiopia, Ghana, Mozambique, Tanzania, Zambia, Uganda) and also elsewhere (Ecuador and Viet Nam) are built in addition to those that already exist for South Africa and Namibia.

MOZMOD, the tax-benefit microsimulation model for Mozambique, has been developed in cooperation with the Ministry of Economy and Finance of Mozambique. MOZMOD is based on the Household Budget Survey (Inquérito ao Orçamento Familiar, IOF) 2008-09 and based on the IOF 2014-2015 allowing for representative results at the national level. Policies are simulated for 2015, 2016 and 2017. Given its expertise and long-term support to the Government of Mozambique since 2006 in establishing a Social Protection Floor at the national level, International Labour Organization (ILO) joined the MOZMOD initiative and supports the establishment of an inter-ministerial policy and technical users group for MOZMOD.

MOZMOD is a highly versatile yet easy to use tool for policymakers and researchers alike. Possible policy reform simulations in MOZMOD include, for example, a universal child benefit, a universal pension payment to the elderly or a youth unemployment benefit. With MOZMOD the number of beneficiaries and the total cost to the state budget can be simulated and characteristics of the prospective recipients analysed (such as: Are prospective recipients more likely to live in urban or rural areas? Do they work in the formal sector or not? What is the expected impact on poverty reduction? etc.), offering rich information to policymakers on effects of different tax and social protection policy proposals. MOZMOD also enables one to explore the effects on the government's budget (How much would such a policy cost?) including, for example, how tax rates could be increased to offset the additional expenditures on social protection or which type of tax reform benefits the vulnerable population more in terms of poverty and inequality reduction.

Microsimulation is a technique that involves taking household survey data and applying a set of policy rules to the data to calculate individual entitlement to benefits and/or liability for taxation. The resulting output at individual and household level can then be analysed to provide national data on, for example, impact of social benefits on poverty and inequality. The base model simulates the existing tax/social benefits arrangements within the country. However, the real strength of the model is that it enables hypothetical changes to benefits and/or the tax system to be simulated and the impact of such changes on the income distribution (including changes in poverty and inequality) to be assessed. Moreover, the expenditure on both the status quo and on any revisions social benefits can be assessed as can the revenue generated through personal taxation (both direct and indirect).[1]

The EUROMOD platform on which MOZMOD is based was built by Professor Holly Sutherland and colleagues at the University of Essex to simulate policies for the European Union countries.[2] EUROMOD has been built and developed over a 20-year period and now runs simulations for over 25 countries. The main benefits of EUROMOD which make it a particularly suitable basis for the Mozambican model are that: all the calculations are transparent and can be easily modified by the user, and the model is very flexible as it allows policies to be modified and almost any type of new policy to be created.

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

This manual was prepared with reference to MOZMOD version 2.0. However, it has been updated to be compatible with MOZMOD 2.0 and also be applicable to later versions such as version 2.1, 2.2 etc.

1.2 Introduction to this manual

This manual is designed as an introductory guide for new users and a reference for those already familiar with the basic operations of MOZMOD. The manual provides comprehensive instructions on using the model for the first time as well as more complex tasks such as building new policies.[3] The focus of the manual is on the technicalities of how to use the model MOZMOD, in practice, rather than a manual about the many processes that could be undertaken using the EUROMOD software more generally. A separate Country Report has been produced which describes each of the taxes and benefits that are included in MOZMOD, as well as a discussion of how the simulated results compare with external sources of data for validation purposes. A Data Requirement Document (DRD) has also been produced which describes each of the input variables that are used within MOZMOD. [i]

[1] See, for example Mitton et al., 2000 and Zaidi et al., 2009.
[2] Sutherland and Figari, 2013.
[3]Much of the material in this manual has been drawn from documentation prepared for the EUROMOD model and the authors are grateful to the EUROMOD team for granting their permission to use this material.

This manual is organized into four main interlinked sections. Section 1 provides an introduction and background to MOZMOD and the manual while section 2 introduces users or readers to the model and its interface. Section 3 presents MOZMOD in detail by describing how to interpret and use the content in the main content file which stores all the information the model needs for its policy simulations. Section 4 explains how to undertake a number of tasks in MOZMOD such as running the model, adding in new SYSTEMS (the rules necessary to simulate a particular tax-benefit system, e.g. rules for 2015, 2016 and 2017 or rules for a reform scenario), and implementing policy reforms.

Throughout the manual special EUROMOD terms are printed in blue capitals, e.g. INCOMELIST, TAX UNIT, PARAMETER. Filenames, tab names, policy names, names of menu items, etc. are printed in italics, e.g. (file) *mz.xml,* (tab) *Display,* (policy) *uprate.mz,* (menu item) *Save Country*.

The following boxes are also used:

Boxes on technical details are marked with a gearwheel. These boxes provide additional information to the main text which is not crucial to understanding the main operations of the model.

Boxes with a warning sign contain information that will help you avoid some of the more common mistakes that can be made when running the model. All users should pay special attention to text in these boxes.

Boxes with a notepad provide a summary of key points and can be used as a quick reminder or reference.

More detail on all aspects of EUROMOD can be found in the EUROMOD help which can be accessed from the *Help & Info* tab within MOZMOD. For general information about the EUROMOD microsimulation model and related research please refer to https://www.iser.essex.ac.uk/euromod.

2. Getting started with MOZMOD

2.1 A brief description of MOZMOD

MOZMOD consists of a software file and several content files. The software file includes the user interface, the executable and the integrated help menu. The content files include the country xml files, as well as various other tools and applications. The software and content files can be updated separately from each other allowing greater flexibility.

The information the model needs for its calculations is stored in the main content files (*mz.xml* and *mz_DataConfig.xml*). These files contain both the information for the implementation of the framework of the tax-benefit model and for the implementation of the particular policies that make up the tax-benefit system. However, these files are not accessed directly by the user. All user input is via the user interface. Within the user interface information is mainly written into POLICIES, and all POLICIES – whether relating to the framework of the model or to the tax-benefit policies being modelled – are directly embedded in the POLICY SPINE so that the whole system can be displayed all at once in a single workspace.

The POLICIES are made up of FUNCTIONS which are the building blocks for implementing a country's tax-benefit system. Each POLICY is described by one or more such FUNCTIONS. Each FUNCTION comprises a number of PARAMETERS which represent a particular element of the POLICY functionality. Usually more than one FUNCTION is used to calculate a tax or benefit and FUNCTIONS can interact with each other.

MOZMOD is running from the main window of the '*countries*' tab of the user interface. MOZMOD draws on data stored in text files and returns the output to a text file.

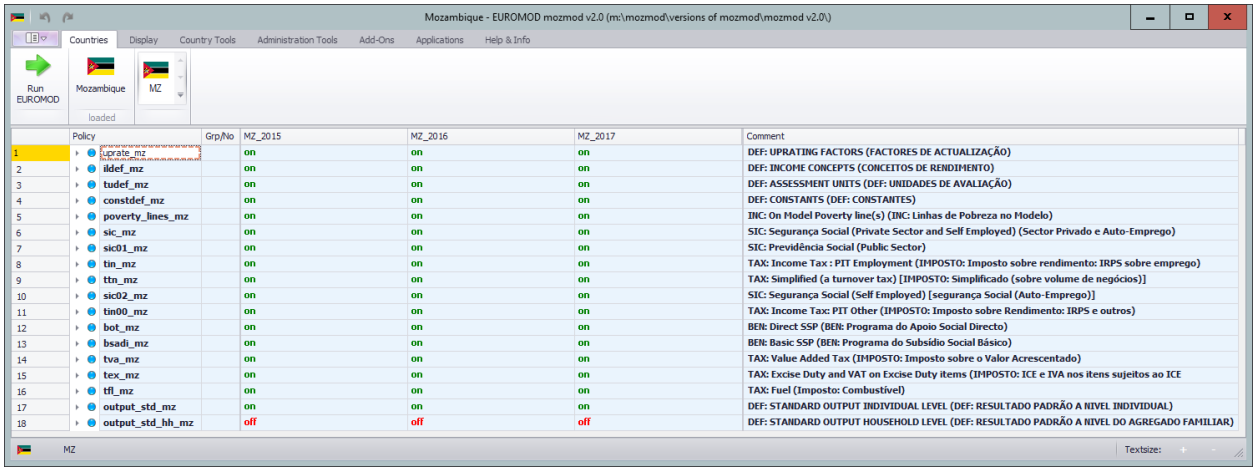
MOZMOD can be summarised as follows:

- DATA – in text format – supplied with the model (but new datasets can be added and existing ones amended)
- MODEL PROGRAM –stores all the model parameters and allows the user to make changes and run simulations
- OUTPUT – in text format – can be analysed using a statistics package

2.2 The MOZMOD user interface

Once the software is launched, the main window of the user interface can be accessed (see Figure 2.1).

Figure 2.1: The MOZMOD user interface

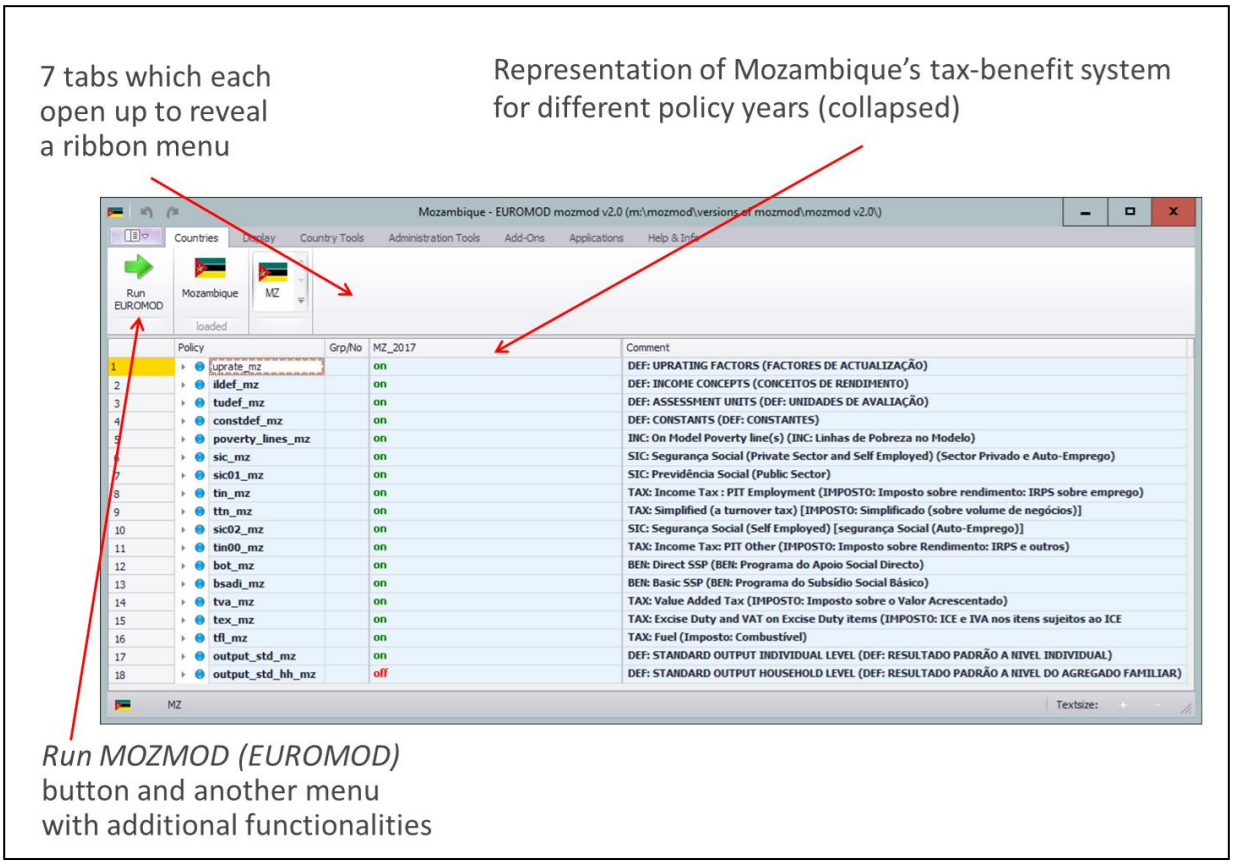


The user interface has seven tabs – *Countries*, *Display*, *Country Tools*, *Administration Tools*, *Add-ons*, *Applications*, *Help & Info* – which each open up to reveal a ribbon menu with a number of functionalities. In addition, there is a *Run MOZMOD*[4] button to the far left of the main window. The main menu directly above the *Run MOZMOD* button contains additional functionalities. Many of these components are described in subsequent sections of this manual.

In order to access the model, click on the Mozambican flag and this opens the main MOZMOD workspace. In EUROMOD there would be a number of country flags visible but in MOZMOD there is just the flag for Mozambique. The main part of the window displays the representation of Mozambique'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. The MOZMOD user interface showing POLICIES in collapsed form and the SPINE is shown in Figure 2.2.

[4] Currently this button is labelled *Run EUROMOD*.

Figure 2.2: The MOZMOD user interface



3. MOZMOD in detail

3.1 Introduction to policies

The first step in microsimulation is to collect data on the incomes and expenditures of individuals in a representative survey of households. The second step is to have a series of policy rules which 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 (a) one or more dataset(s) (i.e. survey data collected in different years) and (b) one or more SYSTEM(S) (i.e. tax-benefit rules for different policy years).

Ideally, to calculate taxes and benefits for the year 2015, for example, you would use the 2015 policy rules together with data referring to the year 2015. 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').

It is EUROMOD good practice to set the best match flag only for BASELINES. The EUROMOD Basic Concepts section of the EUROMOD Help (accessed from the *Help & Info* tab) states that BASELINE is the term used for a SYSTEM-dataset combination which fulfils the best match criterion, and in addition, the SYSTEM must refer to an actual policy year and the SYSTEM-dataset combination must be the main or default implementation for the respective policy year.

MOZMOD v2.0 is underpinned by a micro-dataset constructed using the Household Budget Survey (Inquérito ao Orçamento Familiar, IOF) 2014-15 (which relates to a 2015 time point) and contains only three SYSTEMS which relate to 2015, 2016 and 2017. The 2014-15 dataset is currently used with this SYSTEM and the *uprate_mz* POLICY (see below) uprates the monetary values to 2016 and 2017 using the CPI (or other indices). See Section 3.4 for further information about SYSTEM-dataset combinations.

There are 18 POLICIES in MOZMOD v2.0, which can be grouped into definitional[5] and tax-benefit POLICIES. All 18 POLICIES are visible in the POLICY SPINE and each row of the SPINE represents one POLICY. The POLICIES are processed by the model in the order they appear in the SPINE: first the definitional POLICIES *uprate_mz, ildef_mz*, *tudef_mz* and *constdef_mz*, followed by social insurance contribution POLICIES sic_mz, sic01_mz, tax POLICIES *tin_mz, ttn_mz, then a further* social insurance contributions POLICIES *sic02_mz, and a tax* POLICY *tin00_mz,* followed by the benefit POLICIES *bot_mz, bsadi_mz, the indirect tax* POLICIES*, tva_mz,tex_mz and tfl_mz, tva_mz*, and finally the

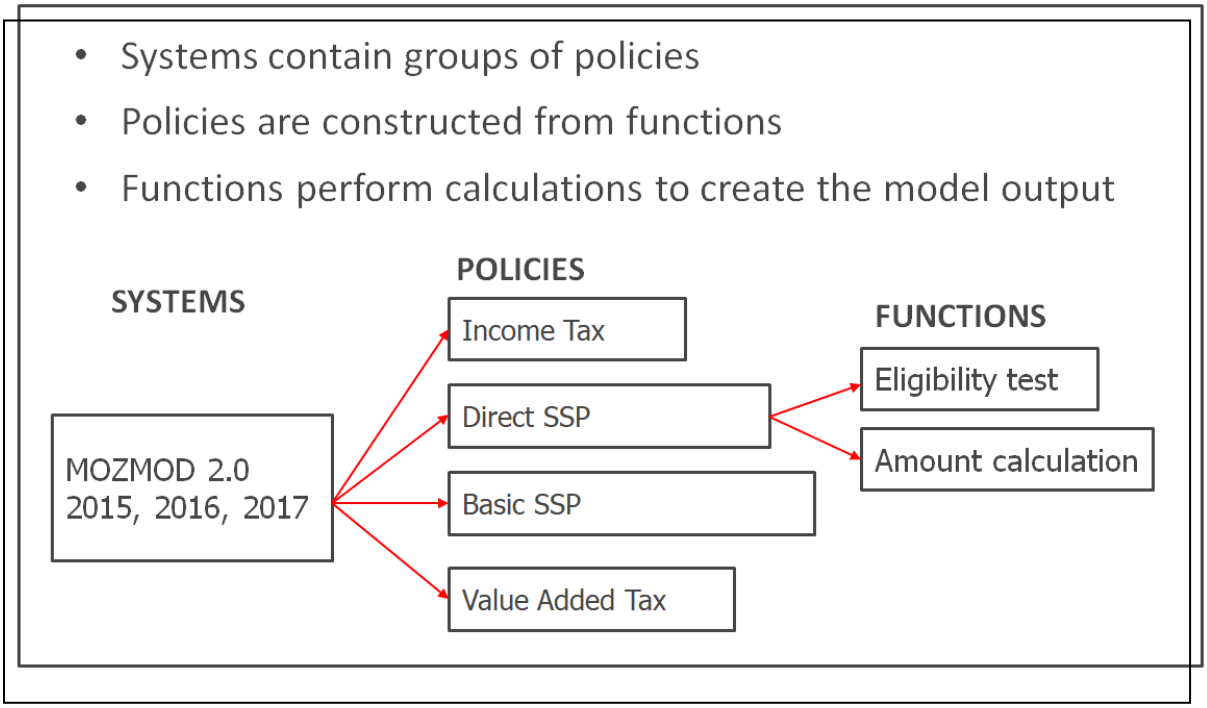
[5]Referred to as 'special policies' in the EUROMOD Basic Concepts section of the EUROMOD Help (accessed from the *Help & Info* tab).

definitional POLICIES *output_std_mz and output_std_hh_mz*. In other words, first, monetary data variables are uprated to the year to which the tax-benefit system refers; second, definitions of income concepts and assessment units are specified; third, CONSTANTS are defined[6]; fourth, poverty lines are specified; fifth, social insurance contributions for the private and public sectors are modelled; sixth, employed persons income and turnover taxes are modelled; seventh, social insurance contributions for the self-employed are modelled; eighth, other income taxes are modelled; ninth, benefits are modelled; tenth, indirect taxes are modelled; and finally, the results are outputted. 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.

All POLICIES, whether definitional or tax-benefit, have the same general structure. The following figure (Figure 3.1) shows this general structure:

Figure 3.1: The relationship between SYSTEMS, POLICIES and FUNCTIONS

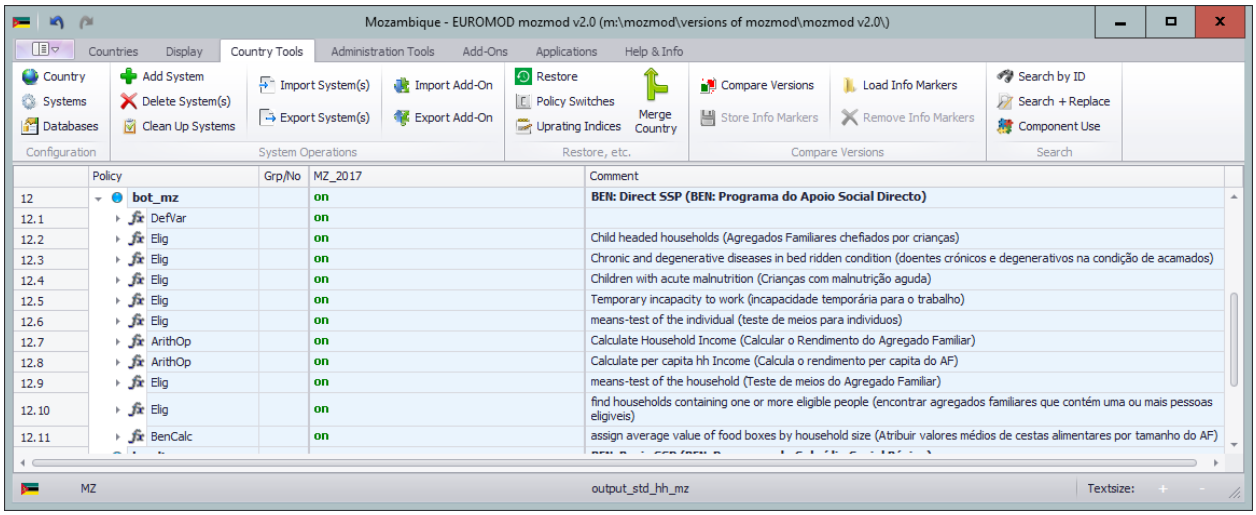


The structure is now described briefly using the Direct Social Support Programme (DSSP) as an example. The POLICY *bot_mz* models DSSP and is the twelfth policy in the model.

Figure 3.2 shows the POLICY for DSSP (b*ot_mz*). Each POLICY consists of a header displaying the name of the POLICY (*bot_mz* in the example) and a 'switch' defining whether the POLICY is activated or not in the SYSTEM (i.e. *MZ_2015*). This facility to switch off a POLICY may, for example, be used if a reform scenario is implemented where the POLICY is not required.

[6]CONSTANTS are explained in the next section.

Figure 3.2: The POLICY *bot_mz*



The POLICY is composed of FUNCTIONS. By right-clicking on the blue dot next to the POLICY name and selecting *Expand All Functions* you can view all the FUNCTIONS of the POLICY. 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*. In figure 3.3 below, three such FUNCTIONS are expanded as an illustration.

Each FUNCTION is a self-contained building block that has its own PARAMETERS and represents a particular component of the POLICY functionality. The purpose of using FUNCTIONS as building blocks of the model is to provide a general structure and standardised language to describe policy instruments.

FUNCTIONS can be classified in three categories: policy (for implementing tax-benefit policies), system (for implementing the framework of the model) and special. Eleven FUNCTIONS are used in MOZMOD, three are policy FUNCTIONS (*Elig, ArithOp, BenCalc*) and eight are system FUNCTIONS (*Uprate, DefVar, DefInput, DefIL, IlVarop, DefTU, DefConst, and DefOutput*). There are no special FUNCTIONS in MOZMOD. 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).

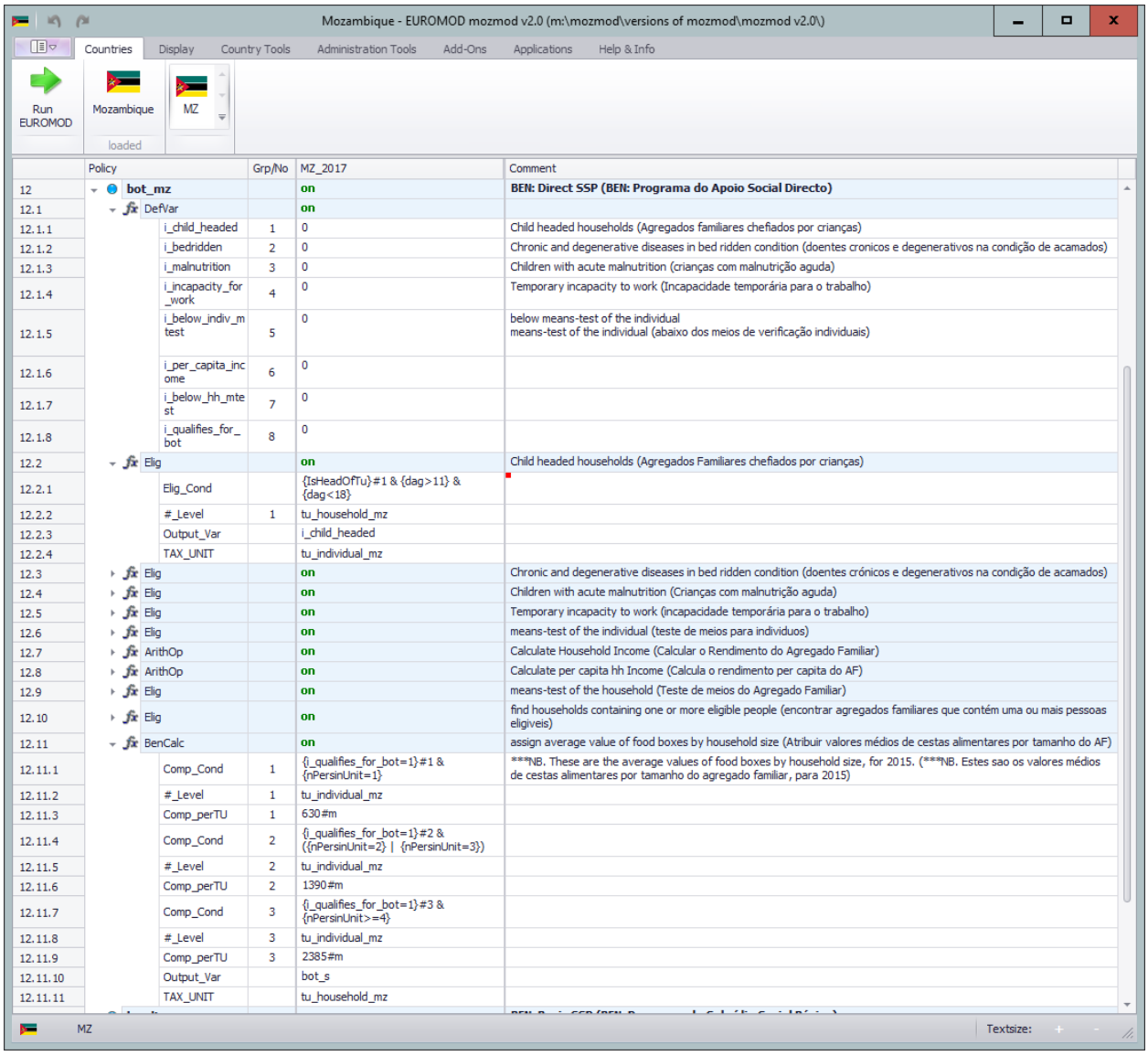
The DSSP example is a fairly typical benefit POLICY that can be implemented. In the *bot_mz* POLICY there are eleven FUNCTIONS.

Each FUNCTION consists of a header displaying the name of the FUNCTION (*BenCalc* in the last example) and a 'switch' defining whether the FUNCTION is activated or not (the FUNCTION is switched on in the example). This facility to switch off a FUNCTION may, for example, be used if a reform scenario is implemented where, in a policy with more than one FUNCTION, one of the FUNCTIONS is not required. So, in the case of the *bot_mz* policy, a reform situation might be to consider the impact of removing the individual level means test. In such case the relevant *Elig*

FUNCTION could be switched off (however further amendments would also be required to the policy to remove reference to the intermediate variable generated by this FUNCTION in subsequent FUNCTIONs in this particular POLICY).

MOZMOD is told how to calculate a particular POLICY by setting the PARAMETERS of a FUNCTION to appropriate values. Many PARAMETERS appear within multiple FUNCTIONS, while some are specific to particular FUNCTIONS. Most of the policy FUNCTIONS and some of the system and special FUNCTIONS provide COMMON PARAMETERS. There are compulsory and optional PARAMETERS, for example the PARAMETER *TAX_UNIT* must be included in all policy FUNCTIONS otherwise MOZMOD will issue an error message. PARAMETER VALUES can take a number of different forms, for example yes/no values, amounts, variables and formulae. These will be introduced as necessary in subsequent sections. The PARAMETER VALUES may change for each SYSTEM.

Figure 3.3 The POLICY bot_mz with some of the FUNCTIONS expanded



	Policy		Grp/No	MZ_2017	Comment
12	bot_mz			on	**BEN: Direct SSP (BEN: Programa do Apoio Social Directo)**
12.1	DefVar			on	
12.1.1		i_child_headed	1	0	Child headed households (Agregados familiares chefiados por crianças)
12.1.2		i_bedridden	2	0	Chronic and degenerative diseases in bed ridden condition (doentes cronicos e degenerativos na condição de acamados)
12.1.3		i_malnutrition	3	0	Children with acute malnutrition (crianças com malnutrição aguda)
12.1.4		i_incapacity_for_work	4	0	Temporary incapacity to work (Incapacidade temporária para o trabalho)
12.1.5		i_below_indiv_mtest	5	0	below means-test of the individual means-test of the individual (abaixo dos meios de verificação individuais)
12.1.6		i_per_capita_income	6	0	
12.1.7		i_below_hh_mtest	7	0	
12.1.8		i_qualifies_for_bot	8	0	
12.2	Elig			on	Child headed households (Agregados Familiares chefiados por crianças)
12.2.1		Elig_Cond		{IsHeadOfTu}#1 & {dag>11} & {dag<18}	
12.2.2		#_Level	1	tu_household_mz	
12.2.3		Output_Var		i_child_headed	
12.2.4		TAX_UNIT		tu_individual_mz	
12.3	Elig			on	Chronic and degenerative diseases in bed ridden condition (doentes crónicos e degenerativos na condição de acamados)
12.4	Elig			on	Children with acute malnutrition (Crianças com malnutrição aguda)
12.5	Elig			on	Temporary incapacity to work (incapacidade temporária para o trabalho)
12.6	Elig			on	means-test of the individual (teste de meios para individuos)
12.7	ArithOp			on	Calculate Household Income (Calcular o Rendimento do Agregado Familiar)
12.8	ArithOp			on	Calculate per capita hh Income (Calcula o rendimento per capita do AF)
12.9	Elig			on	means-test of the household (Teste de meios do Agregado Familiar)
12.10	Elig			on	find households containing one or more eligible people (encontrar agregados familiares que contém uma ou mais pessoas eligiveis)
12.11	BenCalc			on	assign average value of food boxes by household size (Atribuir valores médios de cestas alimentares por tamanho do AF)
12.11.1		Comp_Cond	1	{i_qualifies_for_bot=1}#1 & {nPersinUnit=1}	***NB. These are the average values of food boxes by household size, for 2015. (***NB. Estes sao os valores médios de cestas alimentares por tamanho do agregado familiar, para 2015)
12.11.2		#_Level	1	tu_individual_mz	
12.11.3		Comp_perTU	1	630#m	
12.11.4		Comp_Cond	2	{i_qualifies_for_bot=1}#2 & ({nPersinUnit=2} | {nPersinUnit=3})	
12.11.5		#_Level	2	tu_individual_mz	
12.11.6		Comp_perTU	2	1390#m	
12.11.7		Comp_Cond	3	{i_qualifies_for_bot=1}#3 & {nPersinUnit>=4}	
12.11.8		#_Level	3	tu_individual_mz	
12.11.9		Comp_perTU	3	2385#m	
12.11.10		Output_Var		bot_s	
12.11.11		TAX_UNIT		tu_household_mz	

Figure 3.3 shows *bot_mz* with some of the FUNCTIONs expanded. In the final column, *Comment*, where each POLICY is named (the name starts with either *DEF* for definitional, *TAX* for tax, *BEN* for

benefit, or *SIC* for social insurance contributions), each FUNCTION can also be described, and each of the PARAMETERS can be explained if necessary.

3.2 Definitional policies

Uprate_mz

Overview: Datasets are usually used for simulating several policy years, by uprating monetary values to the corresponding year. The POLICY *uprate_mz* contains information for uprating monetary variables in the dataset.

Figure 3.4: The POLICY *uprate_mz*

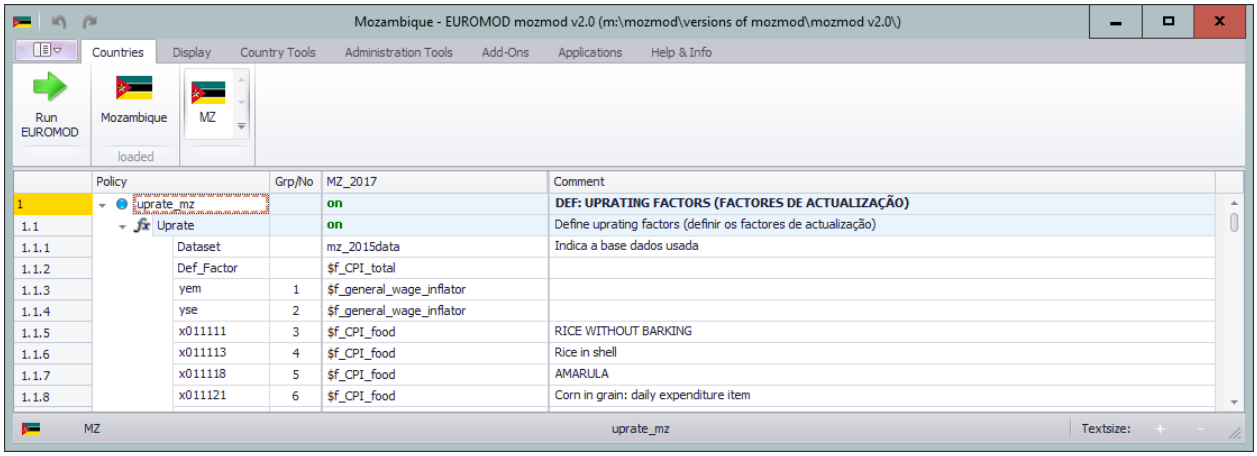
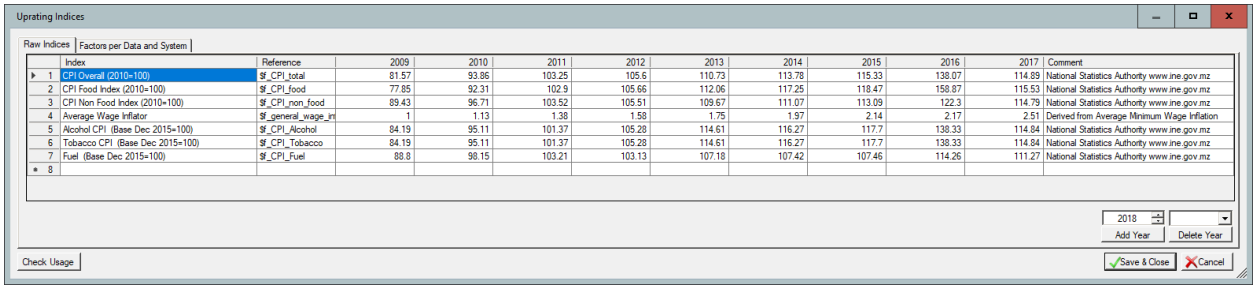


Figure 3.4 shows the POLICY *uprate_mz*. The FUNCTION *Uprate* allows for the uprating of monetary dataset variables to the price level of a policy year. NB only some of the expenditure items are shown that are present in the uprating policy.

The PARAMETER *dataset* specifies the name of a dataset for which the uprating settings apply.

The default uprating factor is specified in *def_factor*. This is the CPI (supplied by the National Statistics Authority). It is expressed as *$f_CPI_total*. Additionally, there are specially calculated inflators for employment income (both earned income and self-employed income). The numerical values of the uprating factors are specified in the 'uprating indices' dialogue which is called from the tab 'Country Tools' and then by pressing the 'uprating indices' button. This is shown in figure 3.5

Figure 3.5 Uprating Indices



This dialogue defines the uprating factors, gives a reference name and describes the source. 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.

Ildef_mz

The POLICY *ildef_mz* contains definitions of INCOMELISTS. Technically an INCOMELIST is the aggregate of several variables, which are added or subtracted to build the aggregate. The most common applications of this concept are income definitions, for example disposable income or taxable income. INCOMELISTS are used in tax-benefit POLICIES for the implementation of the respective tax or benefit.

INCOMELISTS are an important concept. In a single country tax-benefit model such as MOZMOD, definitions of income may simply be hard-wired with a single variable created from its components in the underpinning dataset. However, in a multi-country tax-benefit model such as EUROMOD it is important to retain transparency. INCOMELISTS achieve this and are therefore retained within MOZMOD.

Figure 3.6 shows the POLICY *ildef_mz*. The FUNCTION *DefIL* defines each of the INCOMELISTS. There are a number of such FUNCTIONS in *ildef_mz* but only three have been expanded in Figure 3.6: the FUNCTION for defining the INCOMELIST for taxable income not derived from paid employment where turnover tax is not applicable (*il_taxeabley01* - used in income tax policy *tin00_mz*), original income (*ils_origy*) and disposable income (*ils_dispy*). Names of INCOMELISTS start either with *ils_* or *il_*, mainly to distinguish them from ordinary variables. INCOMELISTS starting with *il_* are 'normal' INCOMELISTS (which are model specific), while INCOMELISTS starting with *ils_* are 'standard' INCOMELISTS which exist in all countries' models and have a particular definition.

The income list *ils_dispy* describes one of the most important EUROMOD concepts: standard disposable income. In general the following components make up disposable income in EUROMOD:

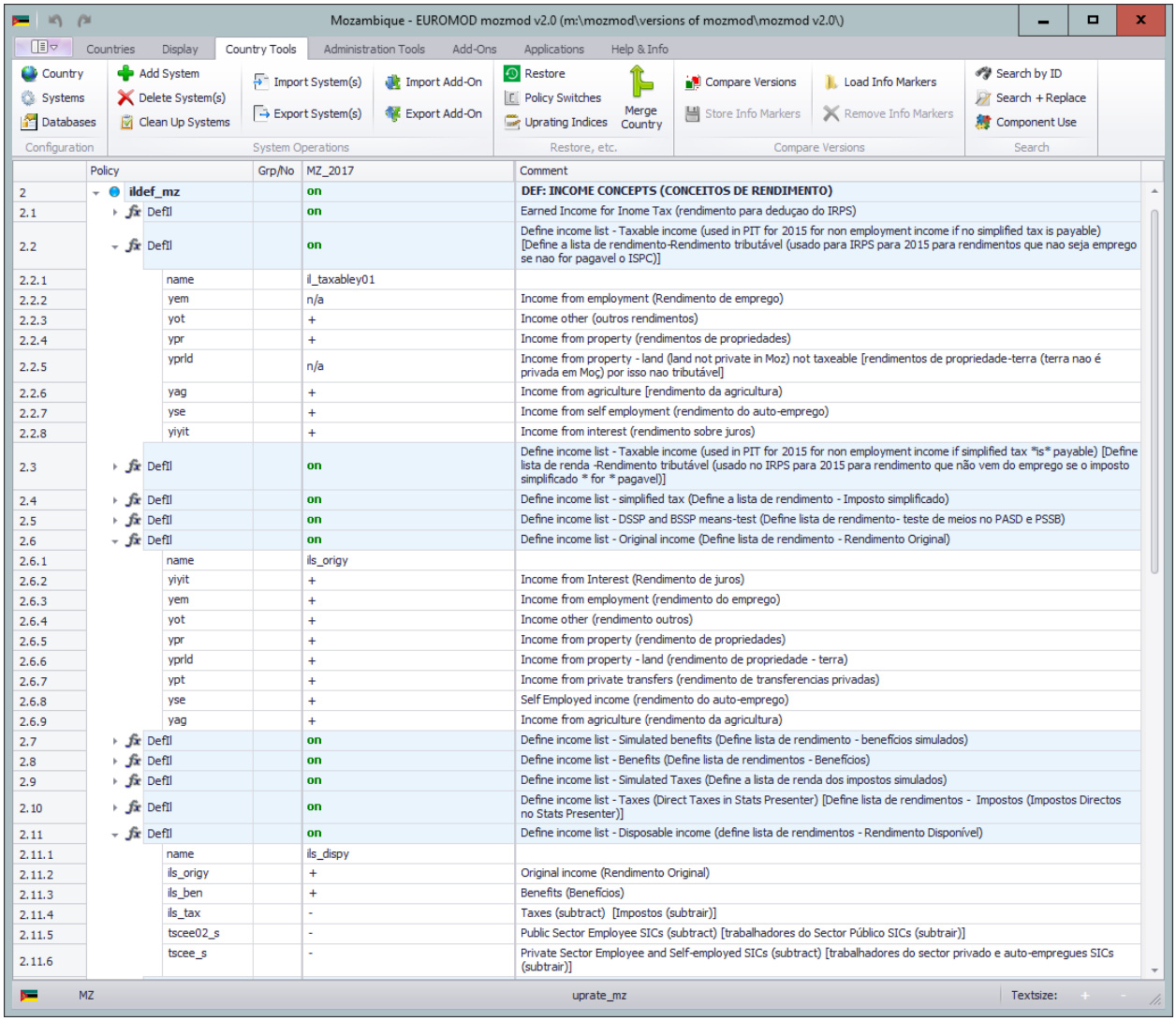
- original income (essentially employment and self-employment income; capital, property and investment income; private pensions and transfers)
- plus benefits (cash transfers, that is unemployment benefits, public pensions, family benefits, social transfers, other (country specific) cash transfers)
- minus direct taxes (income tax, turnover tax, and other (country specific) direct taxes)
- minus social insurance contributions (paid by employees and the self-employed)

This is a fairly standard definition of disposable income but it could easily be modified if a slightly different definition was required, by either amending the existing INCOMELIST or adding a new INCOMELIST.

It can be seen in Figure 3.6 that different income components, listed by their variable name (e.g. *ils_origy, ils_ben, ils_tax*.), are either added (+) or subtracted (-) to create *ils_dispy*. In MOZMOD *ils_dispy* is not used within any POLICY, but rather is an important concept for the analysis stage. Nine INCOMELISTS that are used within POLICIES in MOZMOD are *il_taxabley, il_taxabley01, il_taxabley02, il_simplified_tax, il_dsa , il_exp_vat01 il_exp_vat02 il_exp_vat03 il_exp_vat04 and*

il_exp_vat05. The INCOMELISTs *il_taxabley01*, *il_taxabley02, il_simplified_tax* define taxable income and are used in policies for direct taxes (*tin_mz, ttn_mz and tin00_mz*). The INCOMELIST *il_dsa* defines the income for the means tests in the BSSP and DSSP. The INCOMELISTs *il_exp_vat01 il_exp_vat02 il_exp_vat03 il_exp_vat04 and il_exp_vat05* list expenditure items subject to VAT. The INCOMELIST *il_exp_vat01* deals with standard rated items whereas the other INCOMELISTs deal with other items that are subject to different VAT rates. In fact, *il_exp_vat01* lists all expenditure items imported by the *expenditure_mz* POLICY and puts a + sign against items subject to standard rate VAT and an 'n/a' against items that are exempt from VAT, carry a different VAT rate, or are zero rated. N.B. items that are VAT exempt, carry a different VAT rate or are zero rated should **not** have a '-' sign alongside them as this would mean they are subtracted from the total INCOMELIST and this would be incorrect.

Figure 3.6: The POLICY *ildef_mz*



In principle any POLICY other than *uprate_mz* may contain INCOMELIST definitions. Indeed this was the case for the purpose of creating an INCOMELIST for uprating in the POLICIES expenditure_mz and expenditure_excise_mz. However, for reasons of transparency, most INCOMELISTS are defined centrally in the POLICY *ildef_mz*. This rule may be disregarded if a particular INCOMELIST is just used temporarily in one POLICY. In any case, an INCOMELIST, once defined, is available for all subsequent FUNCTIONS and POLICIES.

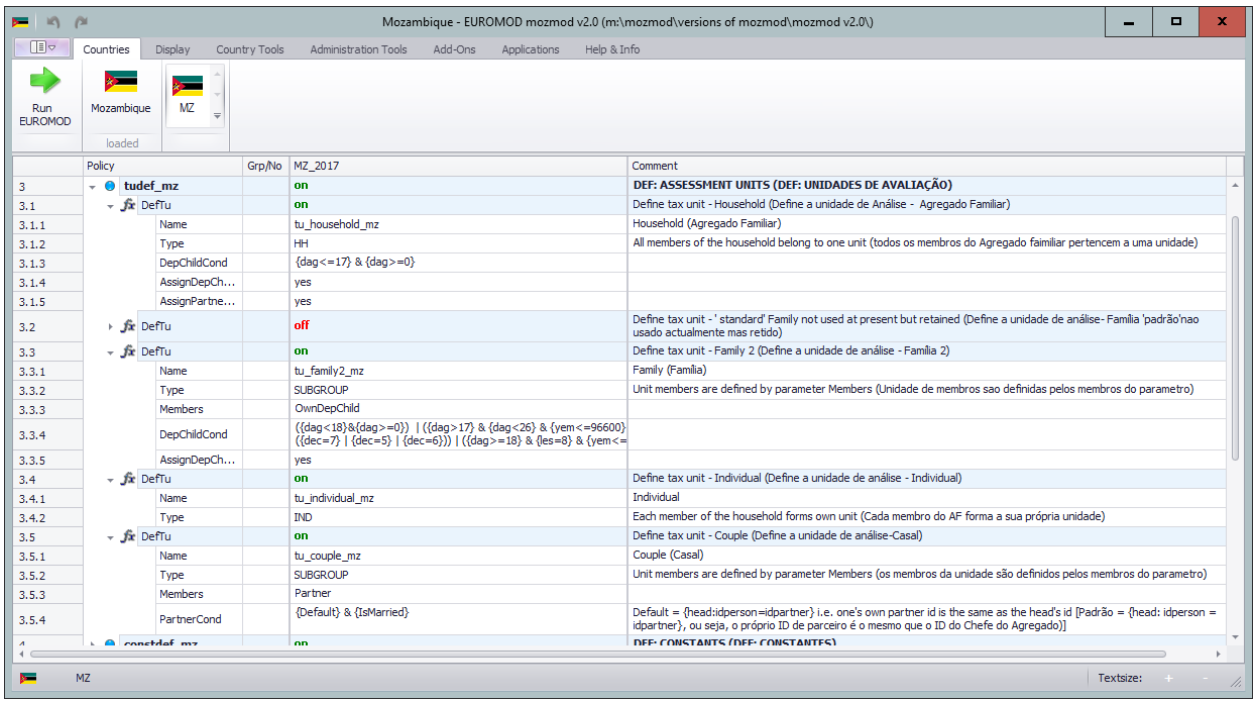
Tudef_mz

Overview: The POLICY *tudef_mz* contains definitions of TAX UNITS or assessment units. Many taxes and especially benefits do not concern single individuals, but refer to larger units, most commonly households. TAX UNITS specify which household member belongs to the different assessment units and who is a child, etc. TAX UNITS are used in tax-benefit POLICIES for the implementation of the respective tax or benefit.

As with INCOMELISTS, in a single country tax-benefit model, the definition of, for example, a dependent child may simply be hard-coded. In contrast, a multi-country tax-benefit model has to deal with numerous and possibly very different assessment unit definitions and therefore must allow for flexible specification. TAX UNITS allow this and are therefore retained within MOZMOD.

Figure 3.7 shows the POLICY *tudef_mz*. The FUNCTION *DefTu* defines the assessment units or TAX UNITS.

Figure 3.7: The POLICY *tudef_mz*



There are four assessment unit definitions in MOZMOD, and they are called *tu_individual_mz*, *tu_household_mz, tu_family2_mz, and tu_couple_mz* (as indicated by the PARAMETER *Name*). The assessment unit *tu_family_mz* is defined but is switched off as no current policy in MOZMOD uses it. The PARAMETER *Type* has three possible values: *HH*, *IND* and *SUBGROUP*. *HH* refers to household type units, which means that all members of the household belong to the same unit. *IND* denotes individual type units, which means that each member of the household forms its own unit. *SUBGROUP* means that the household may be split into several units of different size.

The simplest form of definition is for the TAX UNIT *tu_individual_mz*. This definition is used, for example, for personal income tax, where tax is calculated for each person.

The expression *IsMarried* is used in the PARAMETER *PartnerCond* in the definition of the TAX UNIT *tu_couple_mz*. Such an expression is referred to in EUROMOD terminology as a QUERY. QUERIES are used extensively in EUROMOD, effectively as shorthand for determining the answers to particular questions (e.g. *IsMarried* answers the question 'Is the person married?' and saves each time having to specify all the variables and codes needed to provide the answer). The full list of QUERIES can be found in the EUROMOD Functions section of the EUROMOD Help (accessed from the *Help & Info* tab). The result of a QUERY is either yes or no (e.g. for the QUERY *IsMarried)*, or some numeric (monetary or non-monetary) value (e.g. for the QUERIES *GetPartnerIncome* and *nDepChildrenInTu* respectively). There are no PARAMETERS that only take QUERIES as their values; rather QUERIES are usually used within formulae (discussed above) and conditions (e.g. the PARAMETER *DepChildCond* used in the *tu_household_mz* TAX UNIT definition).

Negation: The operator ! before an expression indicates negation. The expression *![IsParent}* specifies that the person should not be a parent. Had the expression been *{IsParent}* without the ! operator, this would have specified that the person should be a parent. Operators will be discussed in more detail in Section 3.3.

The most complex form of definition is for the TAX UNIT *tu_family2_mz*, where *Type* is set to *SUBGROUP*, meaning that is a subset of the household. Which household member belongs to which unit primarily depends on the PARAMETER *Members*. This is set to *OwnDepChild*. 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, and if there are several equally rich and equally old persons, the one with the lowest number for the variable *idperson*. Knowing this, the PARAMETER *Members* can be interpreted as follows: a unit consists of the head and their own dependent children. The *DepChildCond* PARAMETER is particularly important in MOZMOD as it defines dependent children in a very specific way. Normally one would think of dependent children as being those children under the age of 18 but in this case children over the age of 18 are, in certain circumstances, brought into the definition of "family". This is because this particular tax unit is used in two of the income tax policies where there are special rules relating to additional tax payable where the amount is calculated using information on the individual's income tax band and the number of his/her dependents as defined in this TAX UNIT as being within the taxpayers "family".

In MOZMOD there is another tax unit relating to family. This is the TAX UNIT *tu_family_mz*. This is actually not used at present by the model but is retained (but switched off) as it prescribes a definition of the family which is common internationally. That is a family comprised of a person plus his or her spouse plus their children under the age of 18 (dependent children). In this case *Type* is set to *SUBGROUP*, meaning that the TAX UNIT is a subset of the household. Which household member belongs to this unit primarily depends on the PARAMETER *Members*. This is set to *Partner* & *OwnDepChild*. The PARAMETERS *DepChildCond*, *AssignDepChOfDependents* and *AssignPartnerOfDependents* further specify the TAX UNIT (and are the same as for the TAX UNIT *tu_household_mz*). The PARAMETER *DepChildCond* determines who is treated as a dependent child, whereby !*{IsParent}* specifies that they should not themselves be a parent and *{dag<=17}* specifies that they should be aged under 18. The PARAMETERS *AssignDepChOfDependents* and *AssignPartnerOfDependents* mean that dependent children or partners respectively of dependent unit members (i.e. persons who are not the head or partner of the unit) are assigned to the unit.

In order to build the family unit, first the head of the household is determined and her/his (potential) partner and their (potential) own dependent children are assigned to her/his unit. If there are any persons left in the household who have not yet been assigned, a head is determined amongst them and her/his (potential) partner and their (potential) own dependent children are assigned to her/his unit. This process is continued until all persons in the household are assigned to a unit.

In principle any POLICY other than *uprate_mz* may contain assessment unit definitions. However, for reasons of transparency, assessment units are usually defined centrally in the POLICY *tudef_mz*. This rule may be disregarded if a particular assessment unit is just used temporarily in one special POLICY. Anyway, an assessment unit, once defined, is available for all subsequent FUNCTIONS and POLICIES.

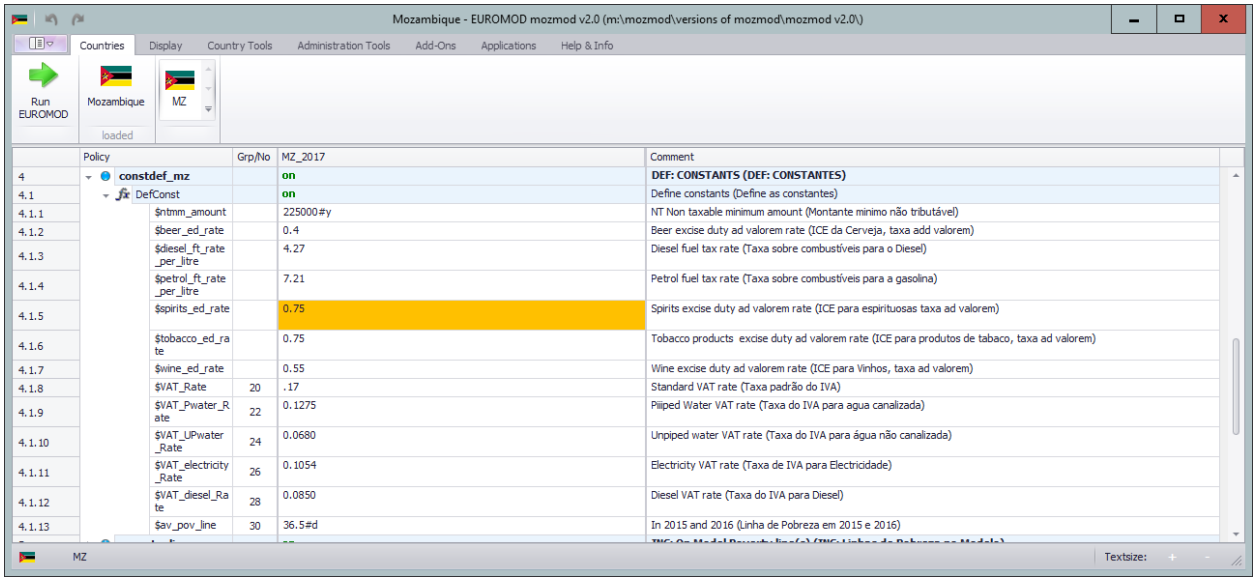
Constdef_mz

Overview: The POLICY *constdef_mz* contains definitions of CONSTANTS. A CONSTANT is simply a way of storing a number (usually a monetary amount) for use in one or more tax-benefit POLICIES. Because the CONSTANTS defined reside in one place, this POLICY is a particularly good place to define the social benefit amounts that are uprated each year. However, some users of the model prefer constants to be placed within the policy to which they refer. This is perfectly acceptable unless the constant is used in more than one policy in which case it should be defined in a separate policy so as to be accessible to more than one policy.

Figure 3.8 shows the POLICY *constdef_mz*. The FUNCTION *DefConst* defines the CONSTANTS. The name of the CONSTANT is defined in the *Policy* column (the first character of a CONSTANT's name should always be $) and the value of the CONSTANT is defined in the respective SYSTEM column. 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. In MOZMOD v2.0 CONSTANTS are defined and relate to tax thresholds, tax rates etc. For example the standard rate of

VAT in the VAT POLICY is assigned to the CONSTANT *$VATt_rate* and given a value of 0.17 in 2015 reflecting the standard VAT rate of that year (17%). Having all the tax/benefit thresholds and amounts in one place makes the task of updating the model for a subsequent year much easier.

Figure 3.8: The POLICY *constdef_mz*



In principle the *DefConst* FUNCTION could appear at the beginning of the policy to which the constant relates. CONSTANTs have not currently been implemented in this way but may be implemented in this way in future versions of the model if users find this more helpful.

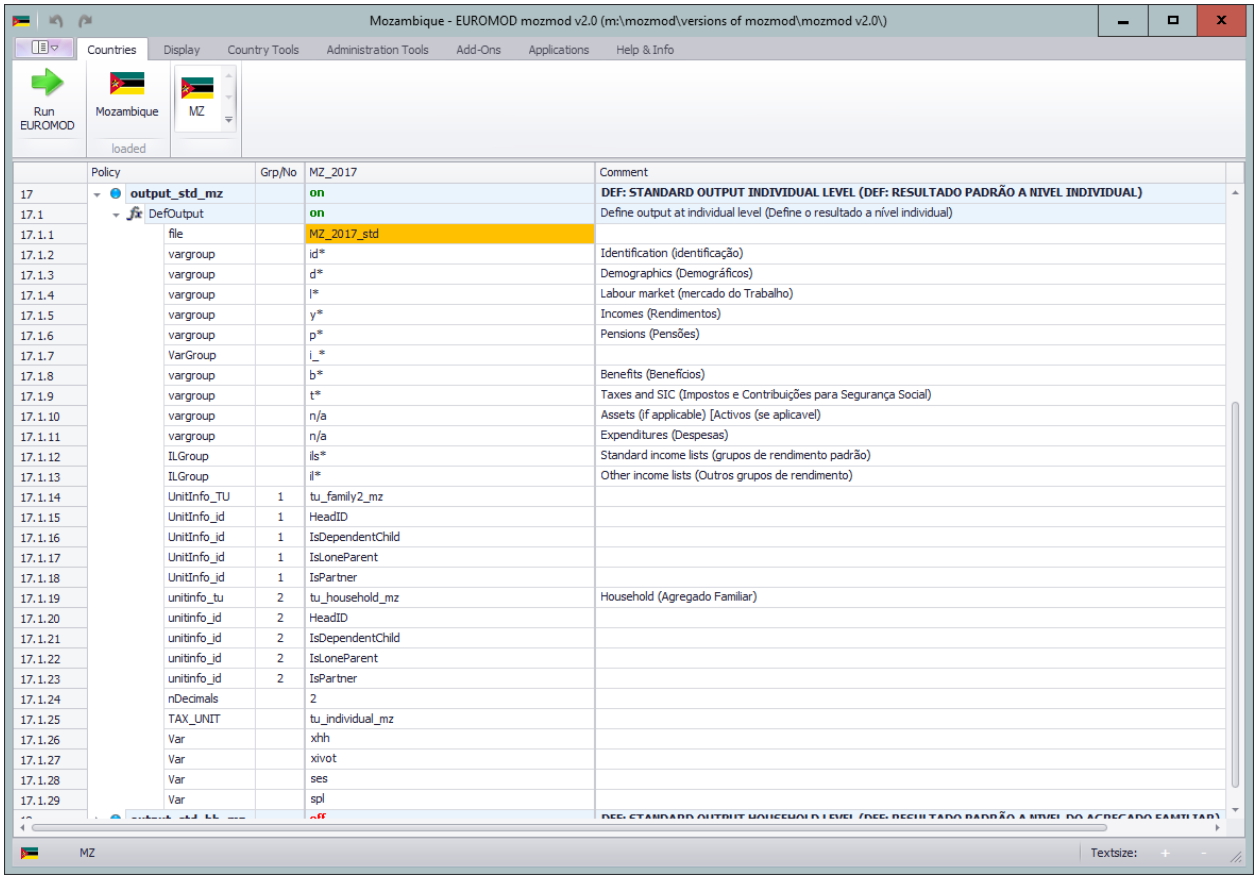
Output_std_mz

Overview: The POLICY *output_std_mz* contains the specification of the output file. Figure 3.9 shows the POLICY *output_std_mz*. The FUNCTION *DefOutput* specifies the output.

The name of the output file is determined by the PARAMETER *file*. STANDARD OUTPUT for the 2015 SYSTEM is written to a text file named *MZ_2015_std.txt*. The PARAMETER *vargroup* allows for a group of variables to be output indicated by the initial letter (S) followed by the*symbol. The asterisk is a 'wild card' and thus for example, *vargroup b** will output all variables in the input data set as well as any simulated variables that begin with the letter b (i.e. the benefit variables both original and simulated). However, it is also possible to output individual variables by using the PARAMETER *var* and specifying the name of the variable. Just as wild cards can be used to output groups of variables this also is the case with income lists so, for example, *ILgroup* ils* will output the value of all the standard INCOMELISTS. The PARAMETER *nDecimals* determines the number of decimal places for monetary variables (any amounts with more decimal places are rounded). The PARAMETER *TAX_UNIT* defines the level of output. In MOZMOD it is set to an individual assessment unit (*tu_individual_mz*), which means that the output contains one row for each individual. If the PARAMETER is set to some larger unit (e.g. a household), there would be one row for each unit,

where monetary variables refer to the sum over all unit members and non-monetary values refer to the head of the unit.

Figure 3.9: The POLICY *output_std_mz*



There is also the option to use the PARAMETER *defIL*, which stands for 'definition INCOMELIST', or in other words, the set of variables that are included within a particular INCOMELIST will be outputted. So, for example, specifying the PARAMETER *DefIL* as *ils_dispy* would mean that the output file would contain all variables included in standard disposable income. Using the PARAMETER *ILGroup* and specifying as *ils_dispy* instead results in only the overall value for standard disposable income being outputted.

3.3 Tax-benefit policies

Tax-benefit POLICIES usually contain the description of one tax or benefit, where this description is made up of FUNCTIONS. Examples of some FUNCTIONS are given below.

- *Elig* determines eligibility/liability for benefits/taxes.
- *BenCalc* calculates the benefit/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 (tax) schedules[7].
- *Allocate* allows (re)allocating amounts (incomes, benefits, taxes) between members of assessment units.[8]
- *Min and Max* are simple minimum and maximum calculators.[9]

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 social benefit POLICIES follow the same general structure, using the FUNCTION *BenCalc* to calculate eligibility for, and the amount of, the grant. Alternatively, the FUNCTION *Elig* can be used to calculate eligibility for the grant and the FUNCTION *ArithOp* can be used to calculate the amount of grant for eligible individuals.

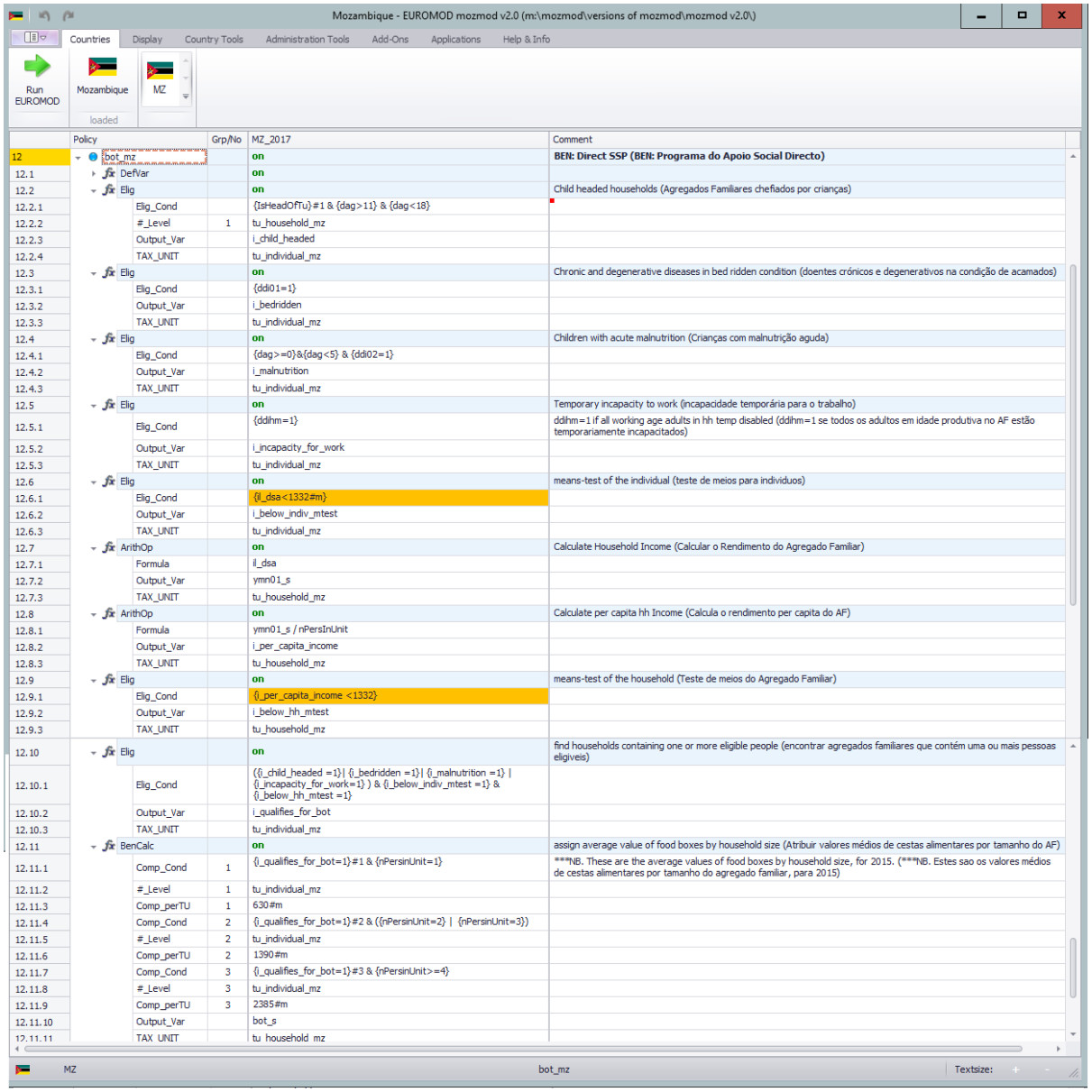
Tax-benefit POLICIES are described in general terms below using the DSSP as an example. The DSSP POLICY *bot_mz* was given as an example in Section 3.1 and is explained in greater detail here. It is shown in Figure 3.10. All these functions are expanded with the exception of the DefVar function which was shown expanded in figure 3.3.

The first, *DefVar* defines intermediate variables required in the modelling of the POLICY. These intermediate variables are mainly eligibility flags set by the following *Elig* FUNCTIONS. As far as possible these intermediate variables are given meaningful names. There are many different groups of people eligible to DSSP (see *Elig* FUNCTIONs 12.2 to 12.6 in figure 3.10). These include child headed households (12.2), bedridden people with degenerative illnesses (12.3), Children suffering from acute malnutrition (12.4) and all adults in the household temporarily disabled (12.5). The *Elig* FUNCTION at 12.6 applies the means test to the individual in question. The two *ArithOp* functions (12.7 and 12.8) calculate total household income and per capita income. This is necessary as in addition to the individual eligibility criteria there is also the criterion that the per capita household income is below a certain threshold. The final *BenCalc* FUNCTION calculates the value of the food box based on the numbers of people in the household. The *BenCalc* FUNCTION is one of the most frequently used FUNCTIONS in MOZMOD.

An alternative – but less streamlined – way of modelling POLICIES that use the *BenCalc* FUNCTION is to use a combination of *Elig* and *ArithOp* FUNCTIONS instead of *BenCalc*.

[7]This FUNCTION is not currently used in MOZMOD; further details can be found in the EUROMOD Functions section of the EUROMOD Help (accessed from the *Help & Info* tab).
[8]This FUNCTION is not currently used in MOZMOD; further details can be found in the EUROMOD Functions section of the EUROMOD Help (accessed from the *Help & Info* tab).
[9] This FUNCTION is not currently used in MOZMOD; further details can be found in the EUROMOD Functions section of the EUROMOD Help (accessed from the *Help & Info* tab).

Figure 3.10: The POLICY *bot_mz*



This can be illustrated by using the following hypothetical policy. For example, assume that a new policy is implemented which provides for a social benefit of 600 Mt per month to each person aged 60 or over. Let us call this POLICY *boa_mz*. This is named for the output variable boa_s (which indicates a benefit 'b', for old age 'oa', which is simulated '_s'). The monthly amount is put into a constant *$boa_amnt*.

Figure 3.11: The POLICY *boa_mz* implemented using *BenCalc*

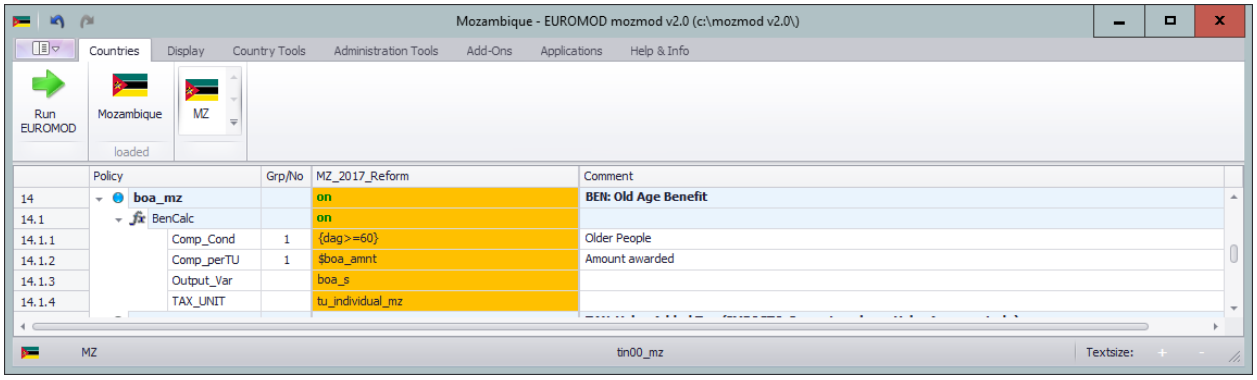
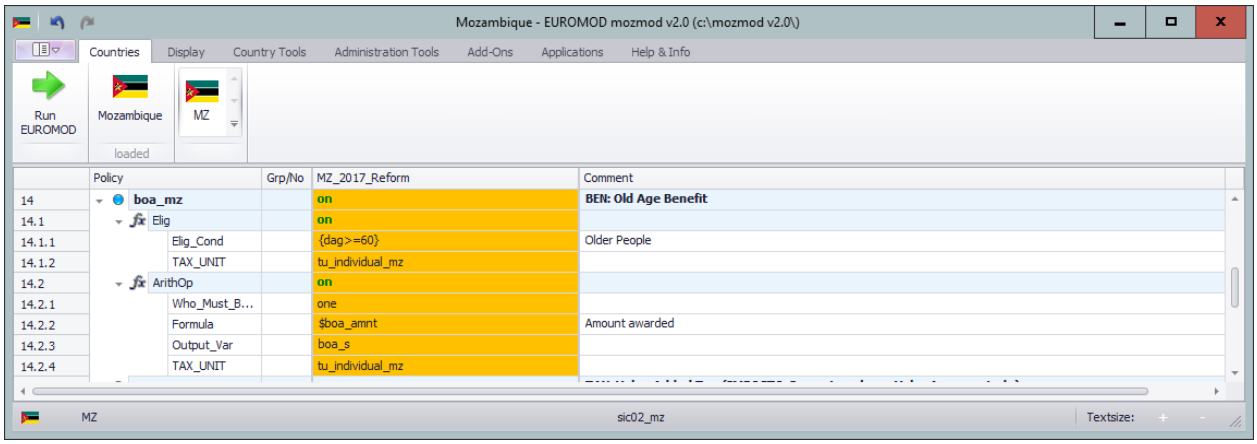


Figure 3.11 shows the hypothetical policy implemented using *BenCalc*. As can be seen the *Comp_Cond* defines eligibility as people aged greater than or equal to 60 {dag>=60}. The *Comp_perTU* is set to the CONSTANT *$boa_amnt* which represents the monthly amount of the benefit. The output variable is *boa_s*. N.B. there are two functions which are turned off. These represent the alternative way of modelling the policy and are illustrated below.

The following screenshot shows how this same POLICY could be (less elegantly) designed using two FUNCTIONs – *Elig* and *ArithOp*

Figure 3.12: An alternative POLICY *boa_mz*



The FUNCTION *Elig* has a PARAMETER *elig_cond* which is exactly the same as the PARAMETER *Comp_Cond* in the *BenCalc* in the implementation of *boa_mz* above and contains the same expression. If the result of the FUNCTION *Elig* is true then a system variable *sel_s* will be set to 1. This is then picked up in the *ArithOp* FUNCTION by use of the PARAMETER *Who_Must_Be_Eliigible*. In this case the PARAMETER is set to one. The *formula* PARAMETER contains the amount and is equivalent to *Comp_perTU* in the *BenCalc.* Finally, the amount is placed in the *Output_var boa_s.*

Certain syntax rules have to be followed when writing the PARAMETER Comp_Cond or elig_cond:

First and foremost each condition must be included within curly brackets {..}.

A single condition {…} has either one component, e.g. a yes/no QUERY (e.g. {IsParent}) or two components separated by a comparison operator > (greater than), < (less than), >= (greater than or equal to), <= (less than or equal to), = (equal to) or != (not equal to) (e.g. {ddi=1}).

There can be conditions which must be fulfilled e.g. {dag<16} and conditions which must not be fulfilled e.g. !{IsMarried}.

Conditions are combined by the logical operators & (and) and | (or) and parentheses can be used to group conditions.

The PARAMETER *who_must_be_elig* can have the following values:

- *one_member* (or *one*): one member of the assessment unit must be eligible
- *one_adult*: one adult member of the assessment unit must be eligible
- *all_members* (or *all* or *taxunit*): all members of the assessment unit must be eligible
- *all_adults*: all adult members of the assessment unit must be eligible
- *nobody*: calculations are carried out for each assessment unit (the default)

Further details on the PARAMETER *who_must_be_elig* can be found in the EUROMOD Functions section of the EUROMOD Help (accessed from the *Help & Info* tab).

When using the *Elig* FUNCTION it is not actually necessary to specify the PARAMETER *output_var* as there is a default output variable called *sel_s* (s=system, el=eligibility, s=simulated) that can be used if an output variable is not specified. This is the only FUNCTION where there is a default value for the PARAMETER *output_var*. This also means that it is not necessary to specify the PARAMETER *elig_var* in the *ArithOp* FUNCTION.

However, for transparency it can be helpful to specify an output variable in the *Elig* FUNCTION. In all other FUNCTIONS the PARAMETER *output_var* is compulsory as there is not a default output variable. The PARAMETER *output_var* is referred to as a COMMON PARAMETER within EUROMOD, in other words the PARAMETER is found in more than one FUNCTION.

The PARAMETER *formula* in the FUNCTION *ArithOp* allows for the following operations:

- addition: operator +
- subtraction: operator –
- multiplication: operator *
- division: operator /
- raising to a power: operator ^, e.g. *2 ^ 3* (result: 8)
- percentage: operator %, e.g. *yem*3%* (result: yem*(3/100))
- reminder of division: operator \, e.g. 22\5 (result: 2)
- minimum and maximum: operators <min> and <max>, e.g. *10 <min> 15* (result: 10)
- absolute value: operator <abs>(), e.g. *<abs>(-22)* (result: 22), *<abs>(50-70)* (result: 20)
- negation: operator !(), e.g. *!(IsMarried)*, *!(17)* (result: 0), *!(0)* (result: 1)

to be used with the following operands:

- numeric values, e.g. *10*, *0.3*, *-25*
- numeric values with a period, e.g. *12000#m*, *1000#y #i* as place holders for numeric *values specified by fo*otnote parameters *variable*s, e.g. *yem*
- INCOMELISTS, e.*g. ils_dispy*
- *queries, e*.g. *IsUnemployed*
- random numbers, *rand*

Order of operation rules:

- ^, <min>, <max>, <abs>, (), !(), %
- before multiplicative operations */ \
- before additive operations + –

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

The remainder of this section comprises points of particular relevance to MOZMOD, but again it is recommended that the user refers to the EUROMOD Functions section of the EUROMOD Help (accessed from the *Help & Info* tab) for a more comprehensive introduction to the FUNCTIONS.

Common parameters

COMMON PARAMETERS are provided by many of the FUNCTIONS and can be classified into four categories:

- COMMON PARAMETERS affecting output. An example already encountered is *output_var* which is provided by all tax-benefit POLICY FUNCTIONS. Other options are *output_add_var* (with *output_var* any existing value is overwritten, but with *output_add_var* the result is added to any existing value of the output variable) and *result_var* (allows a second output variable, generally used in combination with *output_add_var*). These are provided by all FUNCTIONS that have the PARAMETER *output_var* (except *Elig* where it is not meaningful to add to a yes/no variable).

- COMMON PARAMETERS affecting eligibility. Examples already encountered are *who_must_be_elig* and *elig_var*. These are provided by all tax-benefit POLICY FUNCTIONS.
- COMMON PARAMETERS limiting results. There are three PARAMETERS in this category: *lowlim* (for setting a lower limit), *uplim* (for setting an upper limit) and *threshold* (to define a threshold). These are provided by all tax-benefit POLICY FUNCTIONS.
- The COMMON PARAMETER *TAX_UNIT*. As we have seen, this allows for the definition of an assessment unit to which a FUNCTION refers and is compulsory for all tax-benefit POLICY FUNCTIONS.

There are also features that control whether a FUNCTION is processed, including the switch (for turning off FUNCTIONS - 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). These are provided by all POLICY FUNCTIONS.

Further detail on COMMON PARAMETERS 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

There is an issue of interpretation if PARAMETERS taking variables, INCOMELISTS and queries as their values are used with assessment units comprising more than one person.

The following rules of interpretation apply:

- PARAMETERS relating to monetary variables and INCOMELISTS 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.
- PARAMETERS relating to non-monetary variables and individual level QUERIES are interpreted at the level of the individual, if a condition, or at the level of the head of the assessment unit for all other PARAMETERS.
- PARAMETERS relating to non-individual level QUERIES are interpreted in varying ways.

It is possible to change the assessment unit generally used by the FUNCTION for single variables, INCOMELISTS or QUERIES.

Footnotes

There are PARAMETERS which serve to further specify other PARAMETERS. These are referred to as footnote PARAMETERS (or FOOTNOTES). They can be easily identified by names starting with the character #with an integer number, *i*, given in the Grp/No column, e.g. *#_amount1* and *#_level 1*. FOOTNOTES are applicable with several FUNCTIONS, specifically all FUNCTIONS providing formula and/or condition PARAMETERS.

The four most commonly used types of FOOTNOTE are:

1. Limits – it is sometimes necessary to set limits (lower, upper, threshold) to FUNCTION results and single operands.
2. Amounts – it is sometimes more transparent to indicate amounts outside the formula (particularly in a complex formula and/or implementation of several policy years).

3. Assessment units – it is possible to change the FUNCTION's assessment unit (indicated by the PARAMETER *TAX_UNIT*) for a single operand.
4. QUERIES – a few queries need further specification (e.g. *nDepChildrenInTaxunit* counts the number of dependent children in the assessment unit and 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. Only *#m* (monthly – no conversion) and *#y* (yearly – divided by 12) are used in MOZMOD, but other options are 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/grant amounts in monthly terms, income tax thresholds in annual terms).

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:

- Condition: one FUNCTION (usually *Elig*) evaluates a condition and a subsequent FUNCTION operates on the basis of the result of this evaluation.
- Input: one FUNCTION calculates some result, which is used as an input by a subsequent FUNCTION.
- Addition: one FUNCTION calculates a part of a POLICY and a subsequent FUNCTION calculates another part of the POLICY and therefore needs to add to the first part.
- Replacement (actually not a real interaction): 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.

3.4 General settings

The following is a list of some of the key actions which can be undertaken from the *Country Tools* tab:

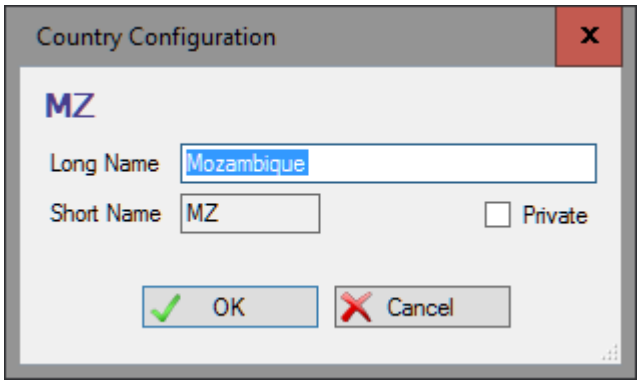
- configuring country settings (e.g. name, short name), SYSTEM settings (e.g. currency) and the input datasets which can be used for simulating the country's tax benefit system;
- adding or deleting SYSTEMS;
- searching – standard search and replace and help with finding errors and assessing if and where certain EUROMOD components (e.g. variables) are used in the respective country's PARAMETERS (see Working with EUROMOD - Searching section of the EUROMOD Help, accessed from the *Help & Info* tab);
- formatting tools, e.g. highlighting with colours and setting bookmarks (see Working with EUROMOD –Formatting section of the EUROMOD Help, accessed from the *Help & Info* tab).

The first of these actions is described in this section. Adding or deleting SYSTEMS is dealt with in Section 4.2.

There are three buttons to the left hand side of the *Country Tools* ribbon: *Country*, *Systems* and *Databases*.

Country

Figure 3.13: The *Country Configuration* dialog

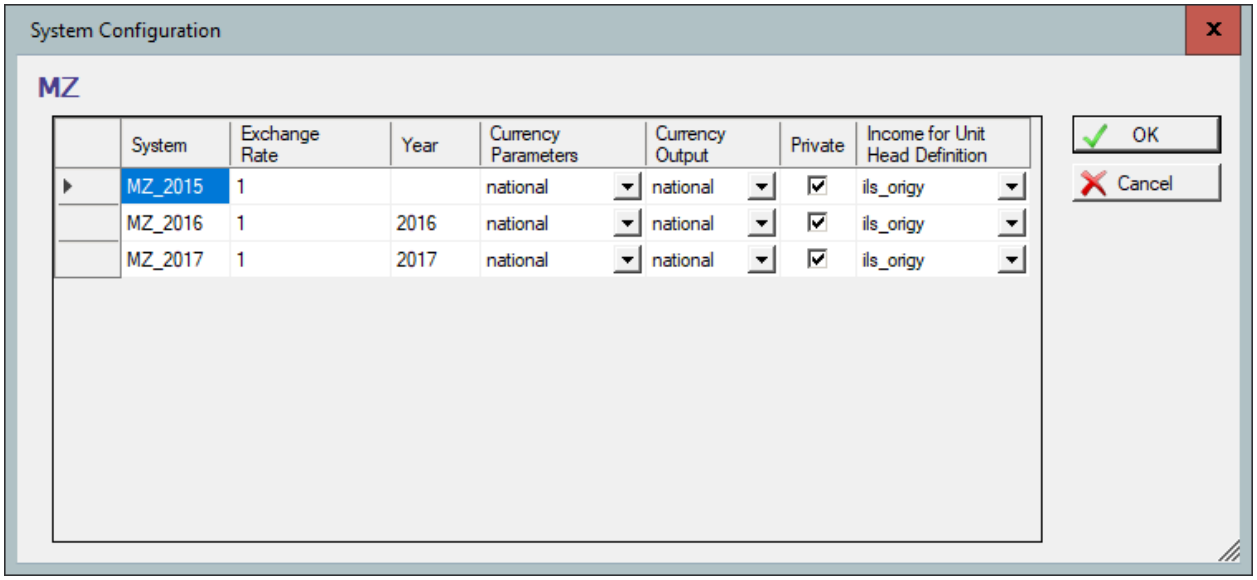


In the *Country Configuration* dialog the *Long Name* of the country can be changed, however the *Short Name* of the country cannot be changed by the user as it needs to correspond to the filenames of the country XML files.

Systems

All systems and their settings are listed and can be changed in the *System Configuration* dialog (see Figure 3.14). The settings include:

Figure 3.14: The *System Configuration* dialog



Exchange Rate: Indicates the rate used by the model to convert national currency into Euro or vice versa (i.e. Euro amounts are multiplied by the rate to get amounts in national currency). This setting is not relevant in MOZMOD and so is set to 1.

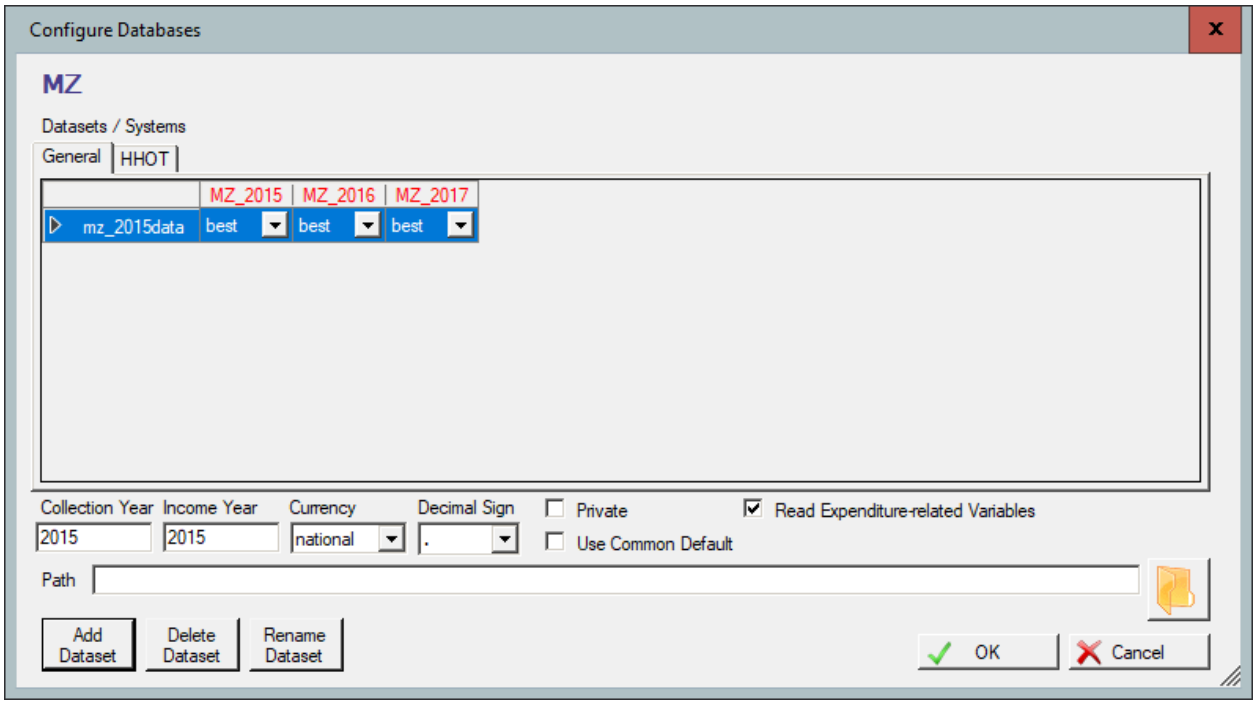
Currency Parameters: Indicates the currency used for monetary PARAMETER values. It is set to *national* in MOZMOD, which means the national currency is used (the alternative in EUROMOD would be *euro* but this is not relevant for Mozambique).

Currency Output: Indicates the currency used within the output file (e.g. *MZ_2015_std.txt*), which again is set to *national* (the alternative in EUROMOD would be *euro*).

Income for Unit Head Definition: Indicates which INCOMELIST (from the dropdown list) is to be used as a default for determining who the head of the assessment unit is. The default setting is *ils_origy*.

Databases

Figure 3.15: The *Configure Databases* dialog



The concept of SYSTEM-dataset combinations was introduced in Section 3.1. There are various options relating to databases:

Assigning datasets to systems

The upper part of the *Configure Databases* dialog (see Figure 3.12) 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 (as described in Section 3.1).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 a 'best match' (as described in Section 3.1), and *n/a* means that the SYSTEM cannot be run with the dataset.[10] In MOZMODv2.0 there is only one dataset and so this is the 'best match' for the single SYSTEM *MZ_2015*.[11]

Adding, removing or renaming a dataset

To add a dataset click the *Add Dataset* button, which opens a file search dialog allowing for the search of a text file containing data suitable to run (one or more of) the SYSTEMS.

To delete a dataset, select it and click 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 of course; the removal concerns only the ability to use the dataset within MOZMOD.

To rename a dataset, select it and click the *Rename Dataset* button, which opens a textbox where the new name can be entered.

Settings of the 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 in which currency data are stored. This is set to *national* in MOZMOD (*euro* would be an option in EUROMOD).
Decimal Sign: Indicates whether data uses point (.) or comma (,) as the decimal sign.
Path: Indicates a specific path to locate the dataset. Usually it is left empty, to instruct the model to locate the dataset at the default path.
Use Common Default: Checking this option means that any variable not in the data, but used by the SYSTEM, is set to zero (i.e. no error message is issued).

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

For further information see the Working with EUROMOD - Changing Countries' Settings section of the EUROMOD Help (accessed from the *Help & Info* tab).

[10] Note that, if a system is copied (see Section 4.2), the datasets assigned to the original system are automatically also assigned to the copied system. Use the *Configure Databases* dialog to change this, if necessary.
[11] Though as mentioned above it is good practice to set the best match flag only for BASELINES.

3.5 Variables

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

Variables follow specific naming conventions. All variables generated by the model end with _*s* for simulated, whereas variables in the input data do not have such an ending (e.g. *bch* is a child benefit taken from the data i.e. 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:
b = benefit (e.g. *bch*: b=benefit, ch=child)
t = tax (e.g. *tin*: t=tax, in=income)
p =pension (e.g. *poa*: p=pension, oa=old age)
d = demographic (e.g. *dag*: d=demographic, ag=age)
l = labour market (e.g. *les*: l=labour market, es=economic status)
y = income (e.g. *yem*: y=income, em=employment)
a = assets (e.g. *afc*: a=assets, fc=financial capital)
x = expenditure (e.g. *xcc*: x=expenditure, cc=child care)
k = in kind (e.g. *ked*: k=in kind, ed=education)

Variables are stored in the file *VarConfig.xml*. The file contains the names and properties of all variables available in the model. Some of the properties help the model to distinguish if 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 and columns which allow for a special description for the country in question. The file also stores the acronyms of which the variable names are composed.

The MOZMOD user interface provides a tool for administering the information stored in the VARIABLE DESCRIPTION FILE. To access this tool click on the button *Variables* in the *Administration Tools* tab.

To the left hand side is a list of all available variables. For each variable there is a checkbox indicating whether the variable is monetary (checked) or not (not checked). There is 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, organised into three levels. The first level is the type as described above (i.e. benefit, tax, income, ...). The second level subdivides types into different categories and 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 (accessed from the *Help & Info* tab):

- Adding variables (see also Section 4.4)
- Changing the name of a variable
- Changing the monetary state of a variable
- Changing the country specific descriptions of a variable
- Deleting variables
- Filtering variables
- Sorting variables
- Searching variables
- Adding acronyms

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 as their use impedes harmonisation of variables across different SOUTHMOD models.

In fact, there are just two general uses of such variables within MOZMOD. The first is for temporary or intermediate variables within policies which are generally not output to the final output file (except for the purpose of debugging). These all begin with i_ (i for intermediate). The rest of the name is as meaningful as possible.

The other exception is VAT/Excise expenditure variables which all begin with an x and quantity variables for excise duties which begin with q. These are then followed by the COICOP code for the item in question. The main reason that these are not introduced in the usual way via the Variables Tool is because there are large numbers of these within the model.

MOZMOD can be summarised as follows:

1. MOZMOD's central access point is the main user interface. From here the content files can be accessed. These store the information the model needs for its calculations and provide other tools and applications.
2. MOZMOD's 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.
3. FUNCTIONS are used as building blocks of both definitional and tax-benefit POLICIES. FUNCTIONS are comprised of PARAMETERS.
4. The definitional POLICIES include *uprate_mz*, *ildef_mz*, *tudef_mz* and *constdef_mz*.
5. *Uprate_mz* contains the uprating factors for monetary variables.
6. *Ildef_mz* contains definitions of INCOMELISTS (i.e. aggregates of variables defining for example disposable income, taxable income, etc.). INCOMELISTS are used in tax/benefit POLICIES for the implementation of the respective tax or benefit.
7. *Tudef_mz* contains definitions of assessment units, which are also used in tax-benefit POLICIES for the implementation of the respective tax or benefit.
8. *Constdef_mz* contains definitions of CONSTANTS used in tax-benefit POLICIES.
9. There is also a definitional POLICY *output_std_mz* which contains the specification of the output from the model.
10. Information relating to country settings (e.g. name, short name), SYSTEM settings (e.g. currency) and the input datasets is contained and modified within *Country Tools*.
11. The VARIABLE DESCRIPTION FILE contains descriptions of all variables available in the model.

4. Tasks in MOZMOD

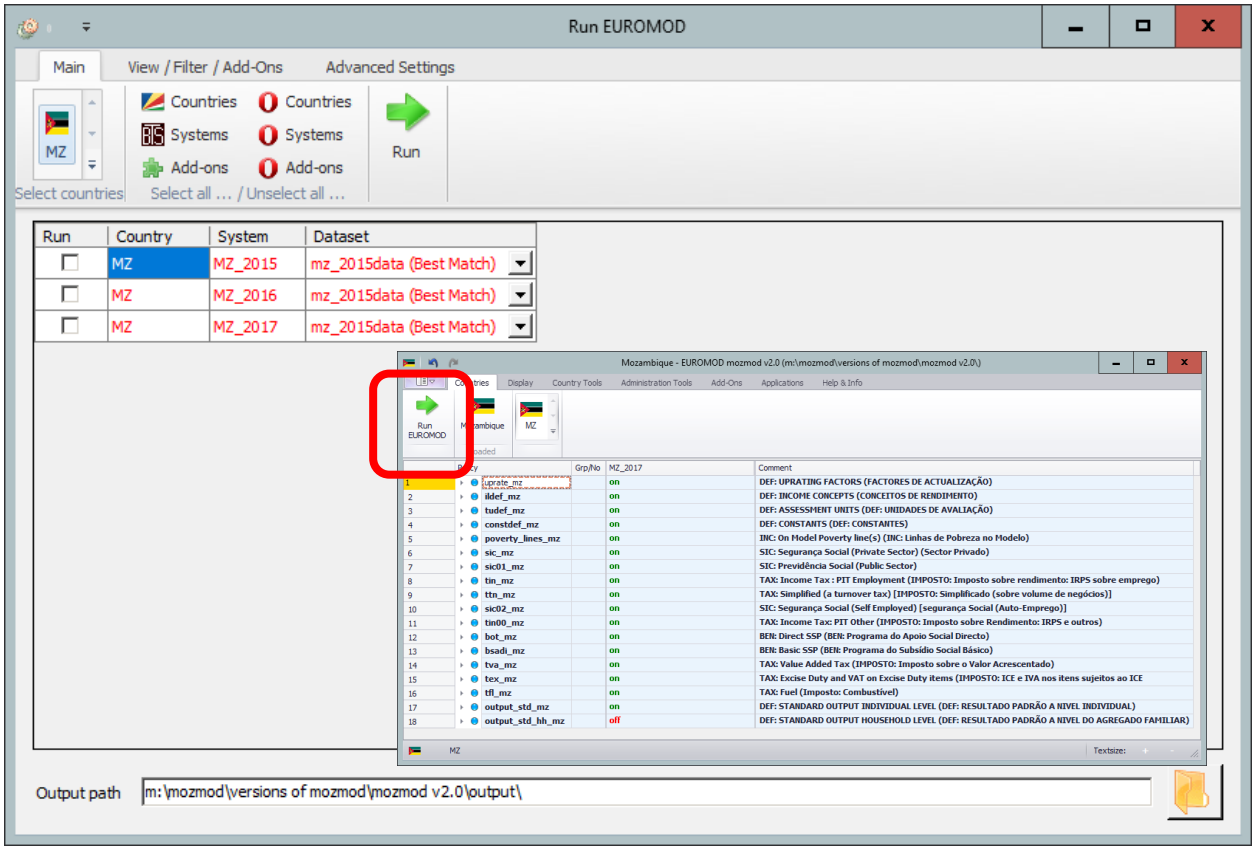
4.1 Running MOZMOD

The next step, having implemented each of the POLICIES and configured the settings in MOZMOD, is to run the model to simulate the tax and benefit policies.

As we have seen, MOZMOD output is based on two inputs: (a) household micro-data and (b) rules on how to calculate taxes and benefits stored in the content file. Using these two information sources, the model calculates all taxes and benefits that have been implemented. The calculations are carried out for each individual or household in the dataset and the result is written to a micro-output file. The output is at the individual level (unless specified otherwise).

To activate the *Run MOZMOD* dialog (see Figure 4.1) click on the *Run MOZMOD* button in the top left corner of the user interface (see inset of Figure 4.1).[12]

Figure 4.1: The *Run MOZMOD* dialog



[12] Currently these both say *Run EUROMOD*.

The main part of the dialog is a list of SYSTEMS which are ready to run. MOZMOD only covers Mozambique and so the *Country* column only shows SYSTEMS for Mozambique (*MZ*) but it has the capability (though not the content) to cover other countries as well. To select a SYSTEM for running check the box to the right of the SYSTEM. The list provides a drop down box for each SYSTEM which 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. As we have seen, there is just one SYSTEM (*MZ_2015 MZ_2016, MZ_2017*) and just one dataset (*mz_2015data*) in MOZMOD at the moment.

The buttons to the left of the *Run* button facilitate the selection of more than one country and/or SYSTEM.

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 installing MOZMOD (or when opening a project – see Section 4.6). 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 MOZMOD issues an error message.

Once the required selections have been made, click on the *Run* button to start the simulation process. A window appears providing information about the progress of the run and allowing for some manipulation. All SYSTEM-dataset combinations selected for running are listed and their status is shown: *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 the processing speed of the computer, on the size of the dataset (i.e. how many households must be processed), and on the extensiveness of the SYSTEM (i.e. the number and complexity of implemented taxes and benefits).

There are three buttons for each run. The *Stop* button enables the user to abort the run. 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. 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 button - its heading indicates what is currently displayed. If any warnings or errors are issued, an error log file is generated, named *yyyymmddhhmm_errlog.txt* (e.g. *201511011530_errlog.txt*).

The information window will stay open until it is closed by the user, even if all runs are finalised, 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 minimise button.

When MOZMOD has finished its calculations, the output is stored as one or more text files at the storage place defined in the field *Output path*. An output text file is produced: the STANDARD OUTPUT. 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 be quickly viewed in Excel, by using the in-built tool *Open Output File* (accessed from the *Applications* tab). It is possible to obtain a range of summary statistics by using the STATISTICS PRESENTER built-in tool without needing to import the output dataset into a statistical analysis package. This is described in more detail in section 4.5 below.

The model produces a header file relating to the output file with 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. The name of this file is *yyyymmddhhmm_EMHeader.txt* (e.g. *201511011530_EMHeader.txt*).

How to produce output:

1. Enter the user interface and click on the button *Run MOZMOD* in the top left corner.
2. Check the box for the SYSTEM-dataset combination of choice. Also ensure that the path to the output file is correct.
3. Click on *Run*.
4. When the simulations have finished running use your favourite text or statistical package to explore the output files.

SYSTEMS will need to be added for future years, for example the policy rules for 2016 will need to be incorporated as a new SYSTEM. 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.

4.2 Adding a new system in MOZMOD

There is a tool in MOZMOD that allows you to add a new SYSTEM. There are two ways to access this tool: either 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*. This opens a dialog which asks for the new SYSTEM's name (e.g. a SYSTEM for 2018 could be called *MZ_2018*). 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. In addition, the user will be prompted for the SYSTEM year – this will usually be either the year of the existing SYSTEM if a reform SYSTEM is being prepared or the new year if a new year is being added. Clicking *OK* adds the new SYSTEM.

New SYSTEMS are initially always a copy of an existing SYSTEM, as it is very likely that the new SYSTEM can use an already implemented SYSTEM as a template. If the new SYSTEM is added by right clicking on the header of an existing SYSTEM, the respective SYSTEM serves as the base SYSTEM. If the new SYSTEM is added via the *Add System* button, a dialog (*Select Base System*) opens allowing for the selection of the base SYSTEM.

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 datasets as in the base SYSTEM. There is just one small difference between the 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 (e.g. the output file for *MZ_2015* is called *MZ_2015_std.txt*, but in a new SYSTEM, say *MZ_2016*, the name of the output file would be *MZ_2016_std.txt*).

The user interface allows the differences between a base and a derived SYSTEM to be highlighted, using background and/or text colour.
This is usually very useful to highlight the changes introduced by a reform (see below). In some of the screenshots in this manual where we have incorporated changes (see e.g. Figures 3.11 and 3.12) these are shown in orange as this is the default highlighting background colour if *Automatic Conditional Formatting* is turned on to indicate changes from the base SYSTEM.

Changes to PARAMETERS (e.g. inputting the 2016 values) can now be made in the new SYSTEM.

Deleting a SYSTEM can similarly be undertaken in two ways: either click the *Delete System(s)* button in the *Country Tools* tab, which opens a dialog (*Select Systems*) where the SYSTEM to be deleted can be selected, or right click the SYSTEM's header and select *Delete System*.

Note the warning that SYSTEM(S) 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.

Conditional formatting can be used to highlight the differences between SYSTEMS. Under the *Display* tab you can choose *Automatic Conditional Formatting* to highlight the differences between the newest SYSTEM and the previous SYSTEM. Alternatively you can specify your own *Conditional Formatting*.

4.3 Implementing a policy reform in MOZMOD

In order to implement a policy reform, it is necessary to remember that a SYSTEM is a collection of tax and benefit policies that apply to a particular point in time, for example the *MZ_2017* SYSTEM records the tax and benefit rules in existence in 2017.

Understanding how to correctly implement a policy reform is a crucial part of using MOZMOD. There are two ways of doing this but only one way which is regarded as good modelling practice. The process of implementing a reform should begin with adding a new SYSTEM to MOZMOD (as described above). This could be called *MZ_2017_reform*, for example, and can be made to copy an existing SYSTEM. Having done this all the changes can be made in the reform SYSTEM and the PARAMETERS in both the existing SYSTEM and the reform SYSTEM will be preserved so that it is easy to see the differences between them and simulations can easily be run for both SYSTEMS.

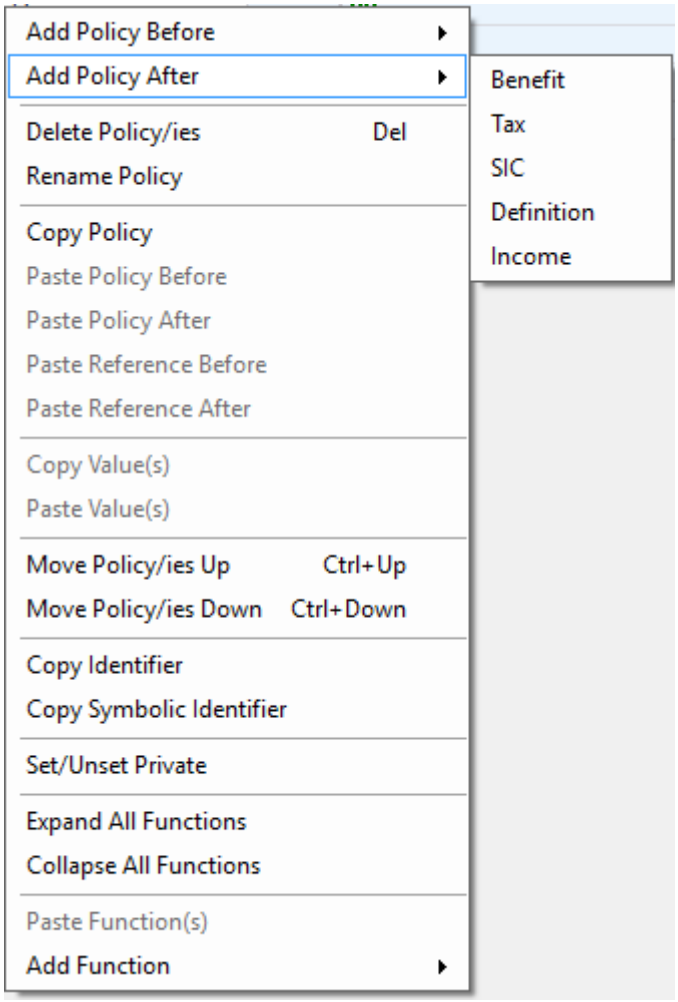
This is preferable to making changes to particular policies in an existing SYSTEM, which although possible is definitely not recommended. The existing SYSTEM is a record of the POLICY PARAMETERS at a particular time point. If it is amended then this record is lost and it would not be possible to run simulations on the existing SYSTEM without undoing the changes made in the reform scenario.

The exact process of implementing a policy reform will then depend on whether 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 criteria) or the introduction of a new policy. The possible options are too many and varied to explain in detail here, so instead the main operations that will be required when implementing a policy reform are described next.

Adding a policy

To add a POLICY right click on the name of the POLICY either before or after which you want to insert the new POLICY. This opens the POLICY's context menu, where you select the menu item *Add Policy Before* or *Add Policy After* (see Figure 4.2). A sub menu opens where the type of the new POLICY can be chosen (*Benefit*, *Tax*, ...). Click 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 *_mz*. If this is not the case the user interface asks whether it should add this ending for you. You may answer this question with *No* to use a non-standard policy name, it is however recommended to answer with *Yes*.

Figure 4.2: The *Add Policy* tool

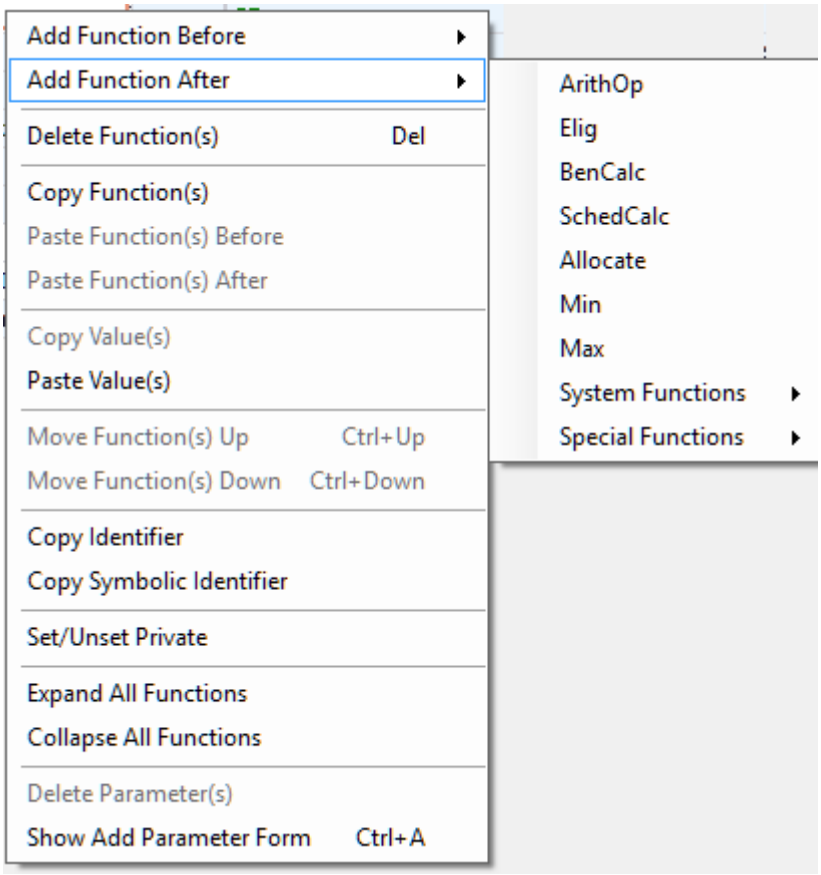


Adding a function

To add a FUNCTION to a POLICY right click on the name of the FUNCTION either before or after which you want to insert the new FUNCTION. This opens the FUNCTION's context menu, where you select the menu item *Add Function Before* or *Add Function After*. A sub menu opens where the FUNCTION to be added can be chosen (see Figure 4.3).

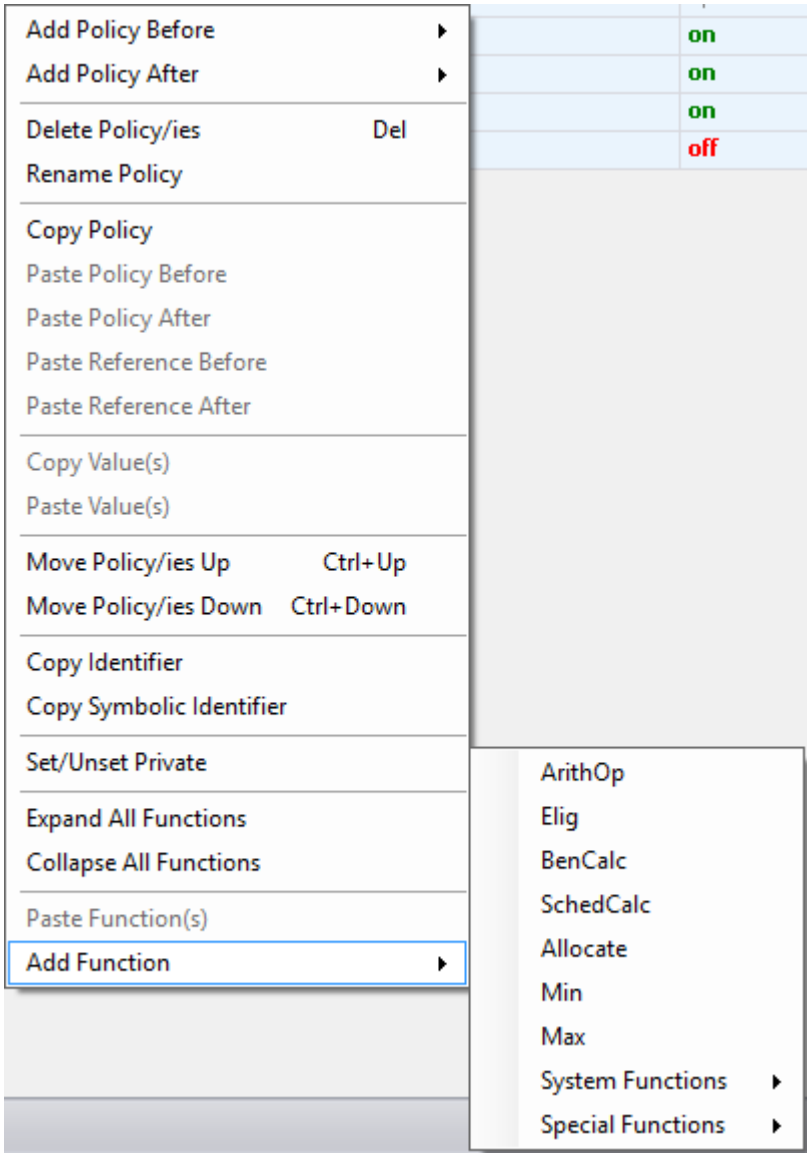
This sub menu is slightly different depending on the POLICY that the new FUNCTION will be part of. Click the respective FUNCTION to add the FUNCTION itself as well as the compulsory PARAMETERS of this FUNCTION (e.g. for most FUNCTIONS, *TAX_UNIT* and *Output_var*).

Figure 4.3: The *Add Function* tool accessed from FUNCTION's context menu (right click on preceding or succeeding FUNCTION)



Alternatively, the POLICY context menu offers the menu item *Add Function* (see Figure 4.4). This adds the FUNCTION as the very last FUNCTION of the POLICY.

Figure 4.4: The *Add Function* tool accessed from POLICY's context menu (right click on POLICY name)



Adding a parameter

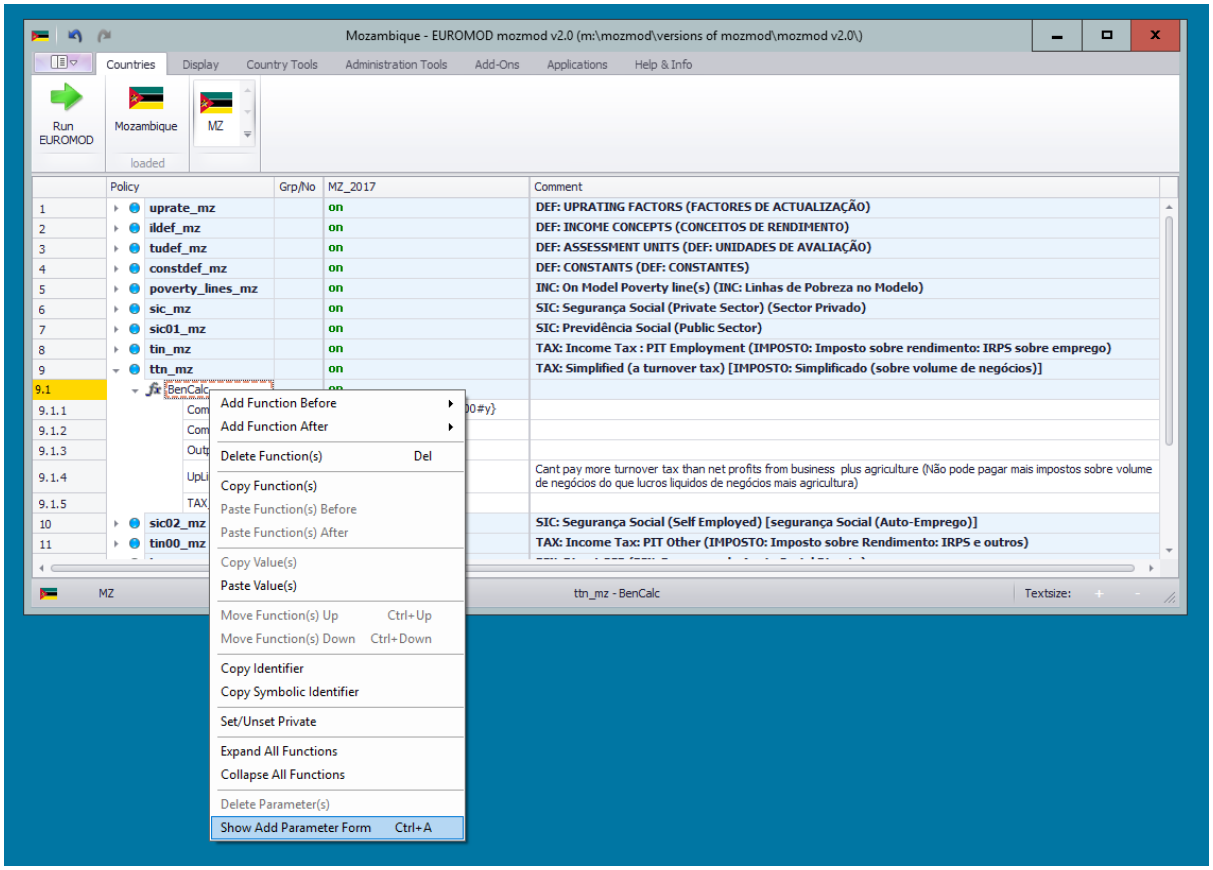
To add a PARAMETER to a FUNCTION right click the respective FUNCTION to open its context menu and select the menu item *Show Add Parameter Form* (see Figure 4.5).

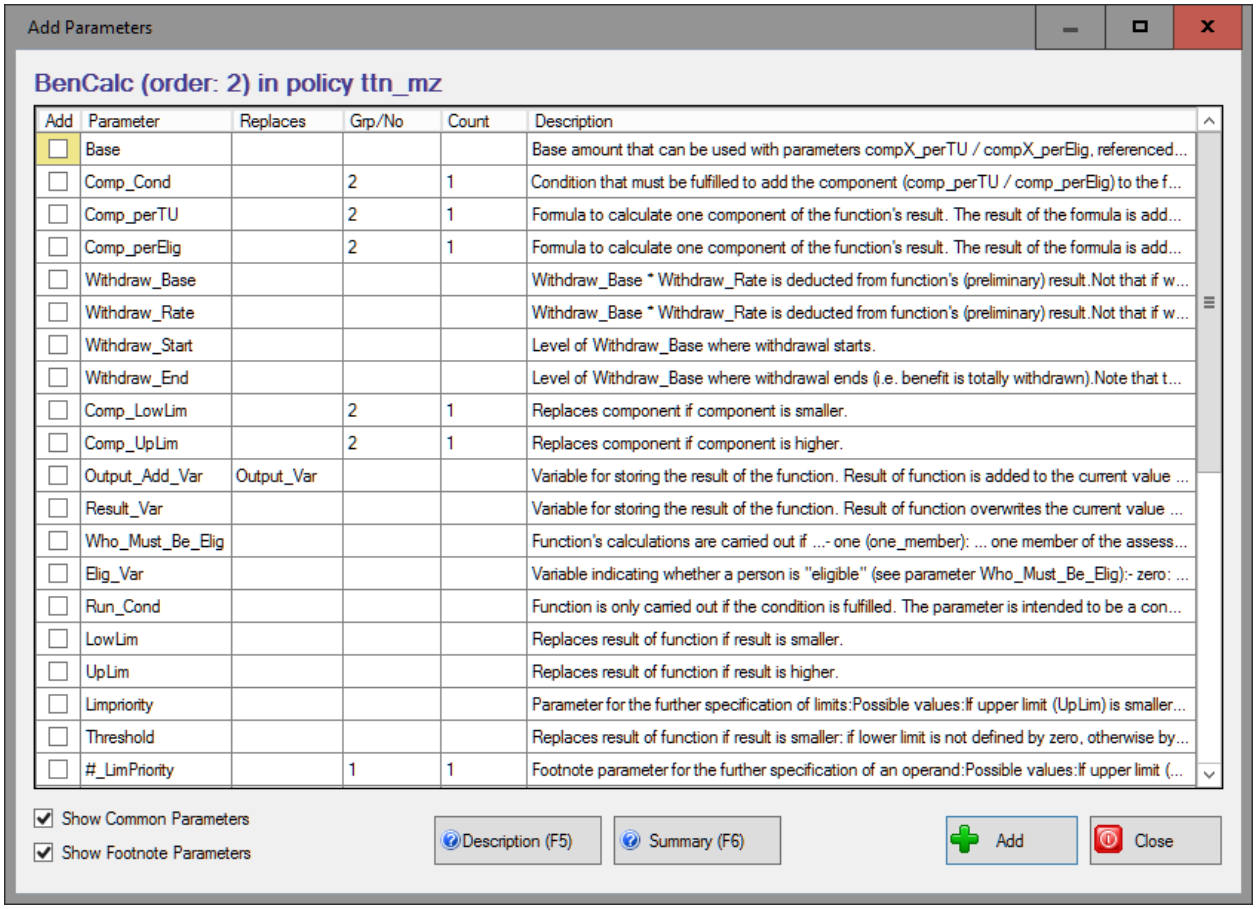
The form shows all PARAMETERS that can be added to the FUNCTION (in the column *Parameter*) with a description (in the column *Description*). Compulsory PARAMETERS of the FUNCTION are not listed (e.g. the PARAMETER *TAX_UNIT*). Exceptions to this rule are 'special' PARAMETERS such as those which can be added more than once (e.g. the PARAMETER *Var* of the FUNCTION *DefOutput* or the PARAMETER *Comp_Cond* of the FUNCTION *BenCalc*) and those which have 'aliases' (e.g. the PARAMETER *Output_Var* with its alias *Output_Add_Var*). The form observes what the user is doing and adapts its content respectively.

In brief, to add one or more PARAMETERS to the FUNCTION select them by ticking the corresponding check boxes. Then click the *Add* button (the button with the green plus). It is possible to add PARAMETERS in bulk (where more than one incidence is allowed e.g. the PARAMETER *Var* of the FUNCTION *DefOutput*). 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).

Note that if in the main view a PARAMETER is selected, new PARAMETERS are added after this PARAMETER, whereas if a FUNCTION is selected (as in Figure 4.3), new PARAMETERS are added at the end of the FUNCTION.

Figure 4.5: The *Add Parameter Form*





Having introduced a new policy, it is necessary to amend certain INCOMELISTS. If it is a new benefit, the INCOMELIST *ils_bensim* will need to be amended to include the new benefit output variable. If it is a new direct tax policy, the new output variable must be added to the INCOMELIST *ils_taxsim*. In addition certain INCOMELISTS will need amendment to ensure that STATISTICS PRESENTER takes the new policy into account for the summary statistics (see 4.5 below).

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).

To make the process of implementing a reform more manageable, sometimes it is helpful to concentrate only on this SYSTEM and maybe its base SYSTEM. To facilitate this, SYSTEMS can be hidden. Right click on a SYSTEM's header and then move the mouse over the menu item *Move to Hidden Systems Box ...* to reveal various sub menu items where there are options for hiding and un-hiding 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 by 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.

4.4 Adding new variables to MOZMOD

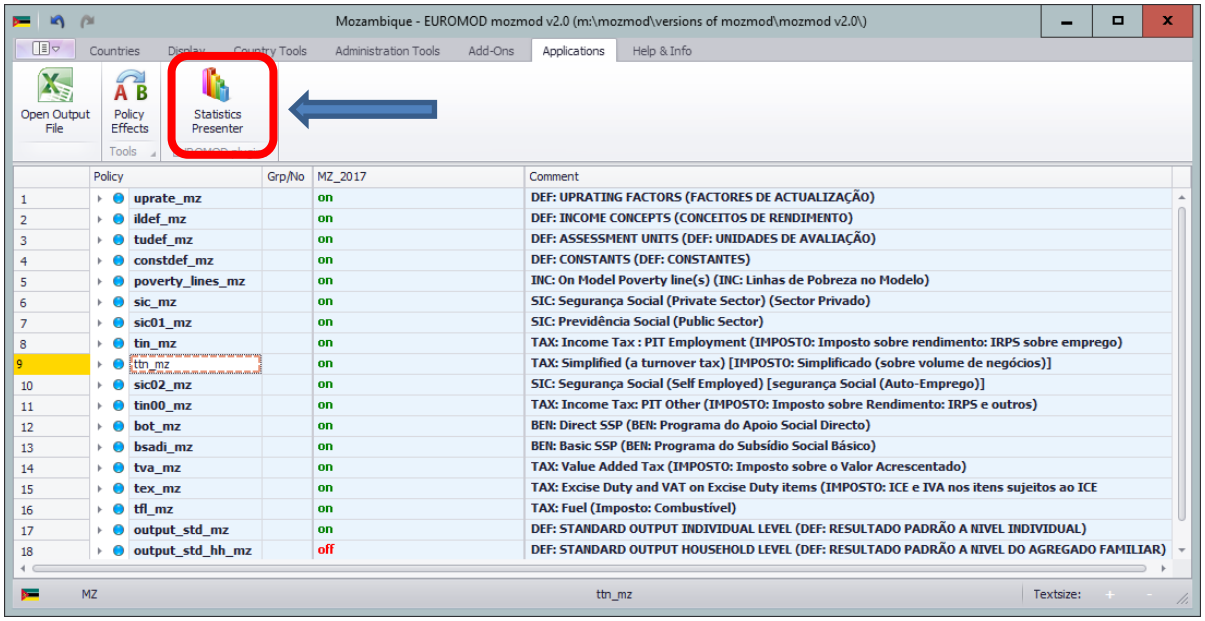
As part of implementing a reform or a new SYSTEM where there has been a change to a policy, it may be necessary to incorporate new variables. As described above, the VARIABLE DESCRIPTION FILE is accessed via the button *Variables* in the *Administration Tools* tab. In order to add a variable click on the button *Add Variable* in the top left hand corner of the *Variables* tab or alternatively press the keys *Alt* and *V* simultaneously. This adds an empty row to the list of variables. Initially the row is added below the selected row. 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.

The variable name can be entered by typing directly into the cell of the *Name* column, using the listed acronyms. If the required acronym is not in the list it can be added using the buttons in the *Acronyms* tab (see the Working with EUROMOD – Administration of EUROMOD variables section of the EUROMOD Help, accessed from the *Help & Info* tab). The *Monetary* box is checked by default but can be unchecked. As discussed above, the *Automatic Label* is automatically generated from the acronyms used in the name of the variable, and cannot be edited. A Mozambique specific description of the variable can be added by typing directly into the *Description* cell. Remember to save the file before closing.

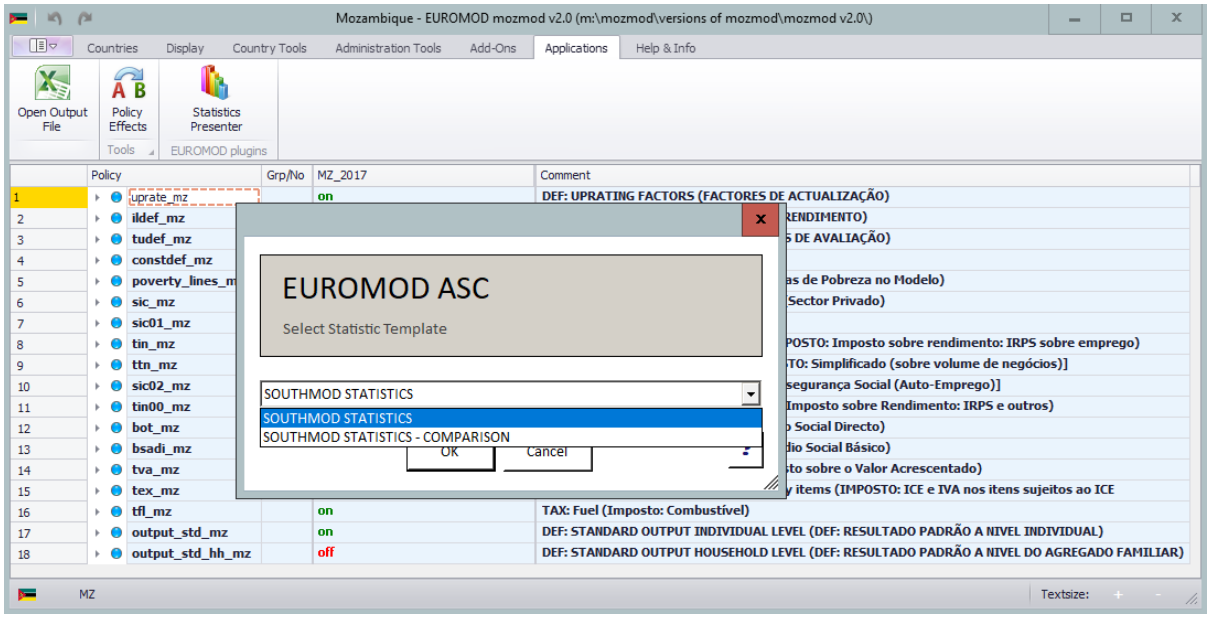
4.5 Using the statistics presenter

The STATISTICS PRESENTER is an extremely powerful (and flexible) tool within MOZMOD. It allows immediate analysis of the output data. At the present time the STATISTICS PRESENTER calculates the costs of benefits and amount of taxes simulated. It produces estimates of both consumption poverty and inequality and income poverty and inequality taking into account simulated taxes and social transfers. It also allows comparisons between two systems.

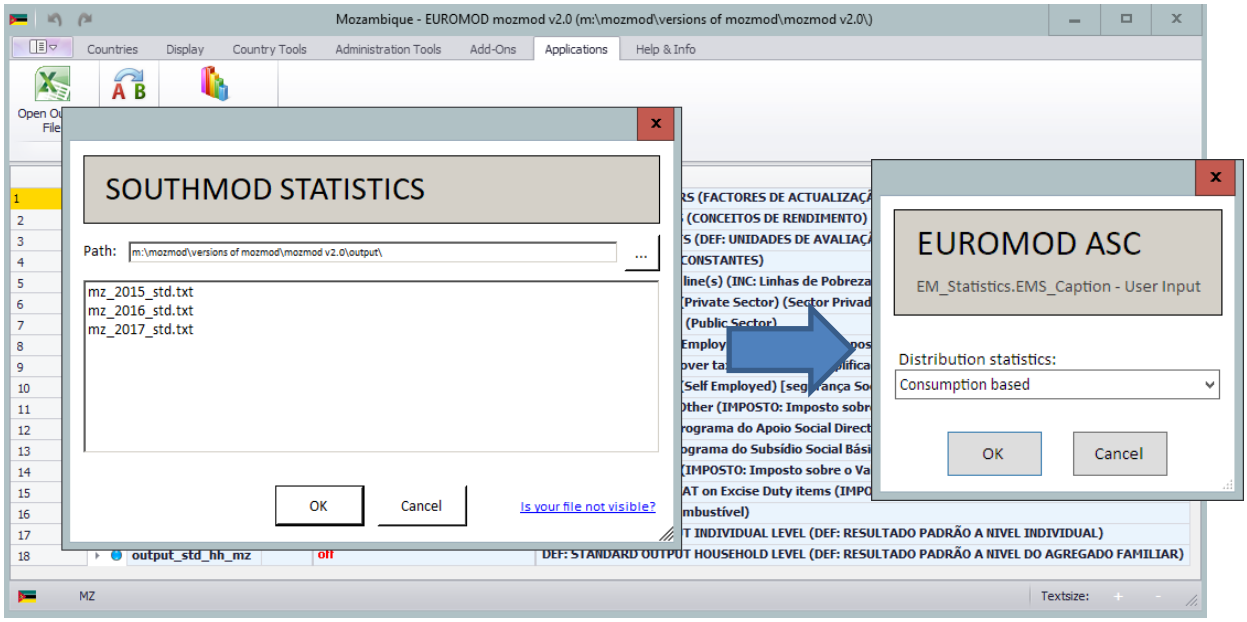
The STATISTICS PRESENTER is accessed from the Applications menu by clicking on the Statistics Presenter button displayed:



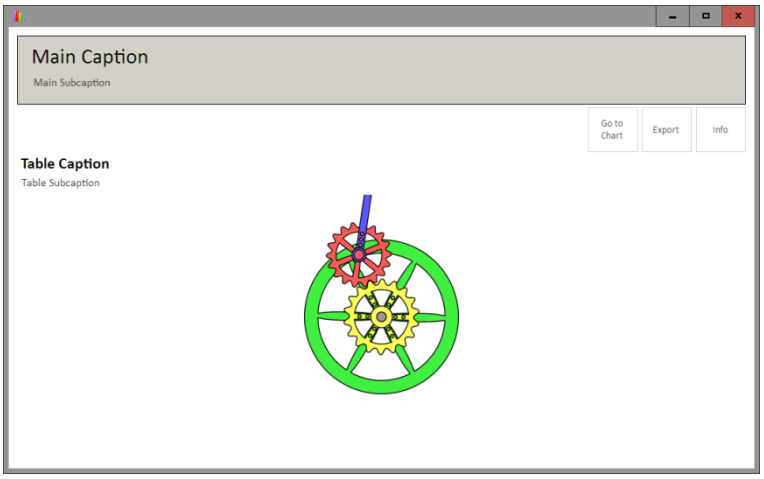
This then reveals the following dialogue:



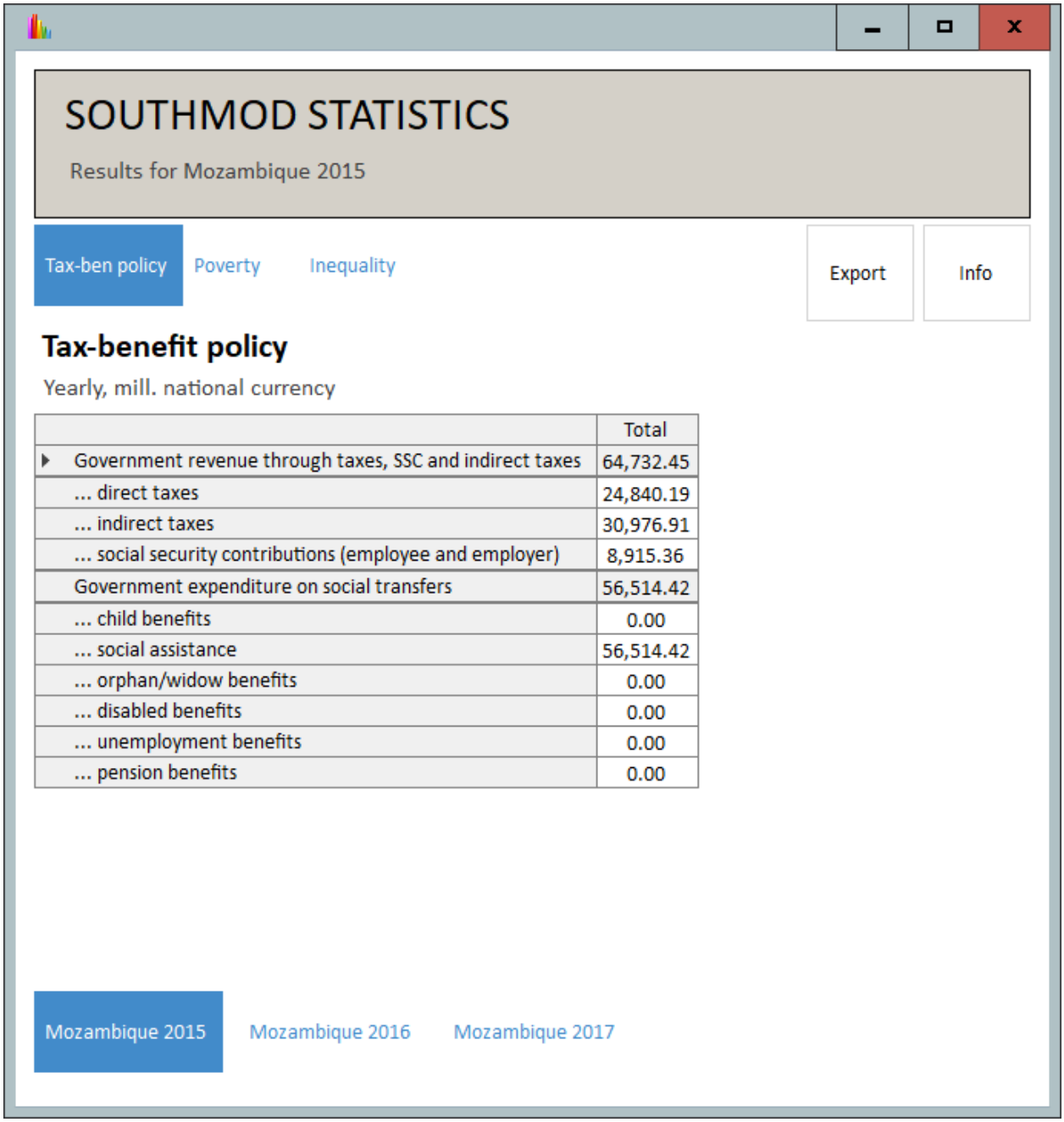
At this point the user is asked to choose between two templates: SOUTHMOD STATISTICS or SOUTHMOD STATISTICS COMPARISON (NB Other Templates may be added in due course). Choosing SOUTHMOD STATISTICS will reveal a list of possible output files to be analysed. One or more may be selected. To select more than one use the control key and mouse click to select the desired output files. Pressing OK will give another dialogue which enables the user to select whether poverty and inequality statistics are consumption based or income based.



After the user has selected either consumption or income based poverty/inequality the STATISTICS PRESENTER will begin to process the results and will display the following gear wheel whilst it is doing so:



When the calculations are complete the following screen is showed first:



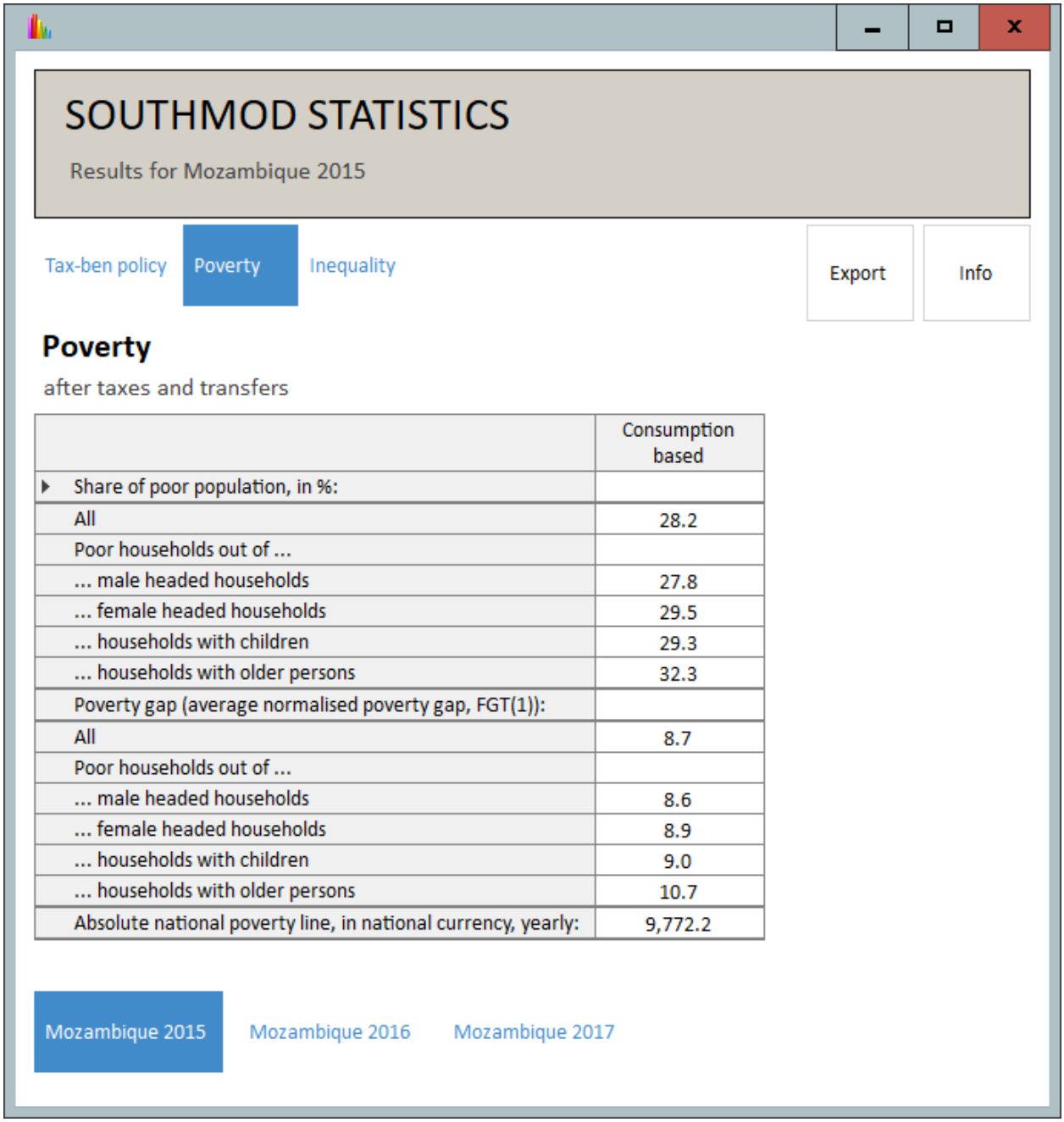
This is referred to as the TAX-BEN POLICY PANEL and presents information on modelled annual government revenue through taxes (direct and indirect) and social security contributions. It also displays annual government expenditure on various kinds of social transfer. What is displayed under different headings is controlled through INCOMELISTS. These will be discussed later in this section.

The POVERTY PANEL (see next page) and the INEQUALITY PANEL are accessed using the tabs to the right of the TAX-BEN POLICY tab (see screenshots below). The POVERTY PANEL shows both the percentage of the population in poverty (also shown broken down by the following types of household: male headed households; female headed households; households with children and households with older persons). The POVERTY PANEL also displays the average normalised 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 the model and can be modified as needed. This poverty POLICY will be discussed later in this section.

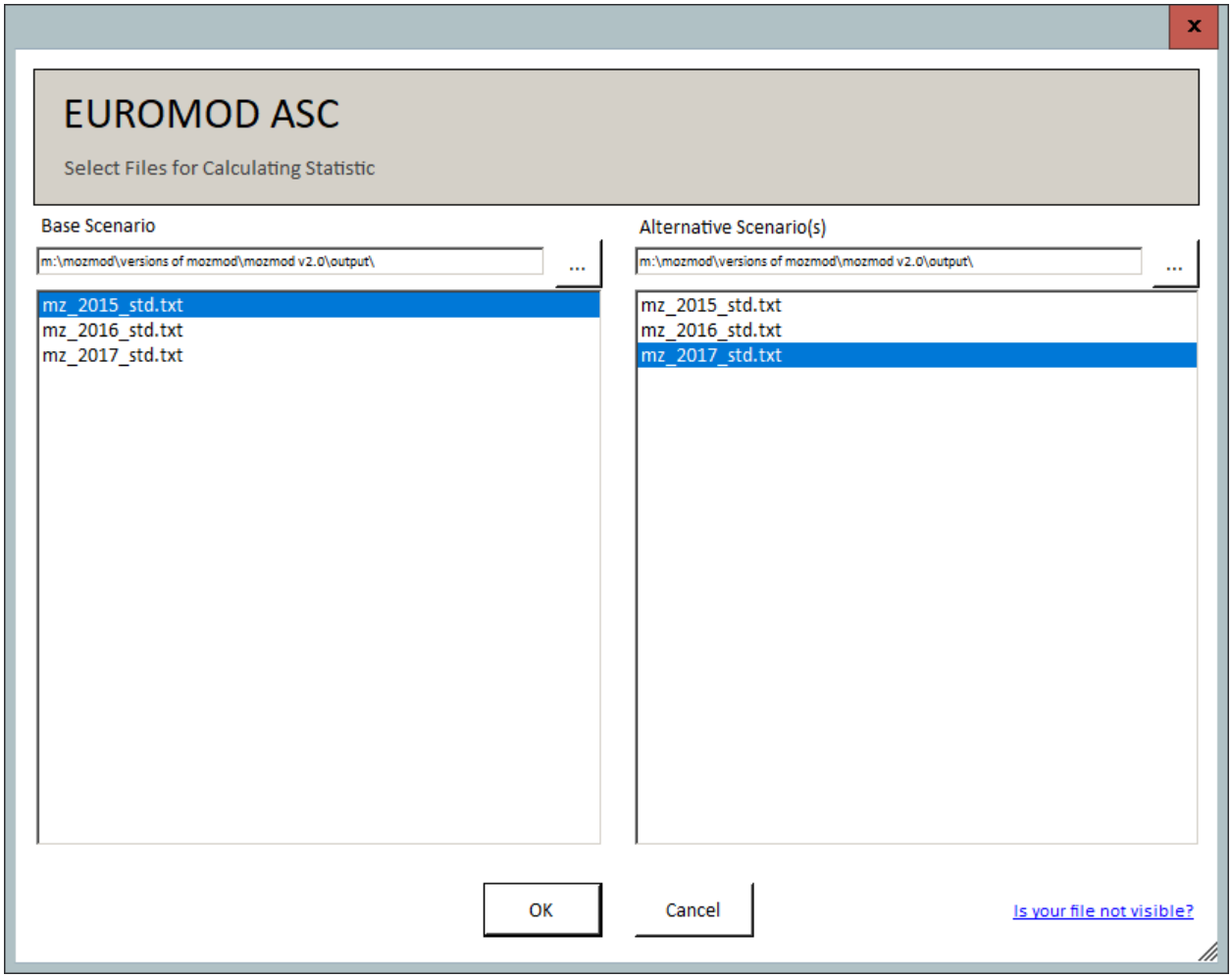
In addition to the GINI coefficient, the INEQUALITY PANEL also displays the P80/P20 ratio measure of inequality (the ratio of the income of those at the eightieth percentile of the distribution compared to the income of those at the twentieth percentile). The income at each quintile of the income distribution is also shown.

As we have selected more than one output file then the tabs at the bottom of the screen show the results for each output file selected.



The initial dialog also allowed users to select If this option is selected then the user is presented with the dialogue shown below. In this example the base output file selected is *mz_2015_std.txt* (the selection is undertaken by holding down the control key and clicking on the base scenario required) and the comparison output file selected is *mz_2017_std.txt* (also selected by holding down the control key and clicking the output file required). After this dialogue the user is presented with the choice of whether to proceed with consumption based or income based estimates and the final output shows the difference between the base system and the comparator system.

This option is particularly useful when examining the impact of policy reforms.



The output for all three panels takes the form of a comparison between the base system (i.e. the output file selected in the leftmost part of the dialogue box shown above) and the comparator system.

Before turning to the model requirements to enable the STATISTICS PRESENTER to work correctly it should be noted that on each panel, there is a button to enable the user to Export the results. Various options are presented. Perhaps '*ALL*' is most useful as it will export to Excel the tables on each of the panels.

Compulsory information required by the STATISTICS PRESENTER

There are two sets of information that the STATISTICS PRESENTER requires in order to function correctly. Certain information is generated when the baseline data is prepared and forms part of the data preparation process: this will not need to be altered by the user and is recorded here only for information purposes. The second set of information is contained in the model: **some of these will require alteration in certain circumstances** but again most will not require alteration.

Data requirements for STATISTICS PRESENTER - generated during the data preparation stage and not requiring alteration by the user

The following variables are generated during the data preparation step and **will not require alteration:**

Variable *xhh* - household consumption as used by the National Statistics Authority (www.ine.gov.mz) for calculating consumption poverty measures in the base year (2015).
Variable *ses* – Equivalence scale in Mozambique for poverty and inequality measurement (per capita).
Variable *dhh* – set to 1 for household head, 0 for other members
Variables representing social benefits reported in data (where applicable).
Variables representing direct taxes reported in data (where applicable).
Variable *xivot* - value of home-grown produce where available to add to income to estimate income poverty. This will be set to 0 when not known or where it is desirable to estimate income property without taking into account home-grown produce.

Requirements for STATISTICS PRESENTER – to be included in the model

Apart from new variables generated within the data preparation stage, the STATISTICS PRESENTER requires certain information mainly in the form of INCOMELISTS to generate the requisite output. As with the variables generated during the data preparation stage many will never require amendment. However, some will require amendment particularly if a new policy is introduced. Where this is the case, this will be clearly indicated in the following sections. The sections indicate the information required for each of the three panels in the STATISTICS PRESENTER output and are grouped accordingly.

i) The INCOMELISTS required for the TAX-BEN POLICY PANEL

The first three INCOMELISTS represent the revenue from taxation reported in the first part of the table in the TAX-BEN POLICY PANEL. **They will only require amendment if new taxes or Social Security contributions** are simulated.

ils_tax: contains simulated direct taxes (income tax and presumptive tax)
ils_taxind: contains simulated indirect taxes (such as VAT and excise taxes)
ils_sic: contains simulated social security contributions (employee and employer)

The second group of six INCOMELISTS contain the benefits simulated in the model. They are required for the second part of the table in the TAX-BEN POLICY PANEL - the costs of social benefits. **It is important to note that these six lists are mutually exclusive:** a simulated benefit can only occur

in one of the lists. So, for example, if a disabled child benefit were simulated, a decision would need to be made whether to insert this into the child benefit income list or the disability benefit income list. **Amendments to these lists will need to be made any time that a reform scenario is introduced that introduces a new benefit, or when a new benefit is introduced by government.**

ils_bch: contains all simulated child-related benefits
ils_bsa: contains all simulated social assistance-related benefits
ils_bsu: contains all simulated orphan and widowhood-related benefits
ils_bdi: contains all simulated disability-related benefits
ils_bun: contains all simulated unemployment-related benefits
ils_pen: contains all simulated pension benefits

ii) The INCOMELISTS and other requirements for the POVERTY PANEL and the INEQUALITY PANEL

There are a number of income lists required for the POVERTY PANEL and the INEQUALITY PANELs to work correctly.

For the consumption based measures the following INCOMELISTS are essential. **However, in general, they will not need to be altered.** Any alterations will have been made to component income lists **e.g. if a new benefit is simulated it will have already been added to *ils_bensim* just as any new tax simulated will have been inserted within *ils_taxsim*** (see section 4.3 above). This is good practice and should have been carried out whether or not the STATISTICS PRESENTER is used.

The INCOMELISTS for consumption poverty are as follows:

Ils_tistn -this is composed of the following items recorded or imputed in the underpinning data: employee social security, turnover tax and any other direct tax.

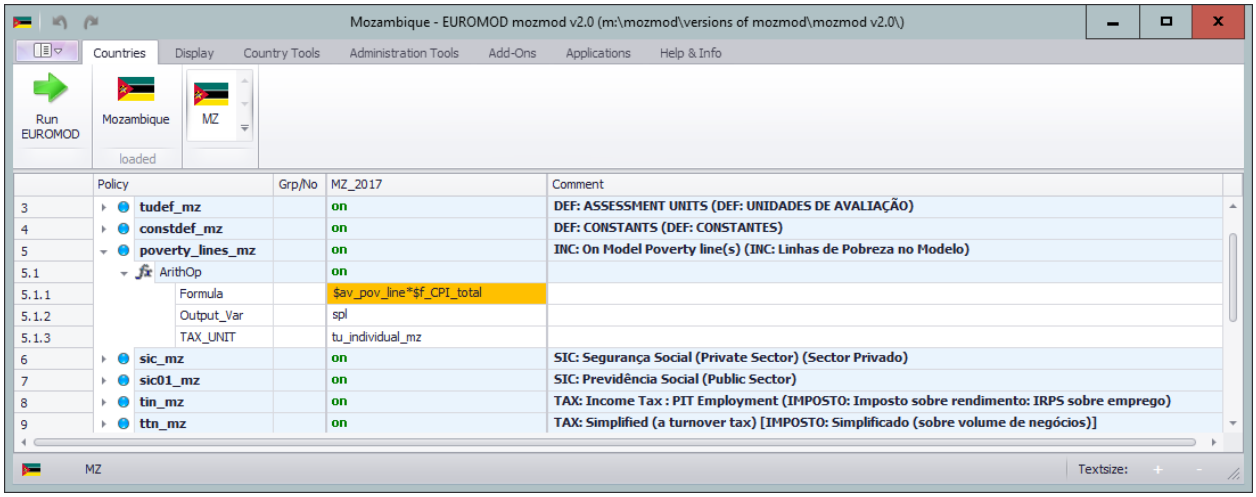
Ils_bendata – these are social benefits as reported in data.

The key Income list supporting consumption-based poverty is *ils_xhh_s* – this comprises the following components which are either added (+) or deducted(-): *xhh* (+), *ils_sicee* (-), *ils_tistn* (+) and *ils_bendata* (-)

There is only one INCOMELIST used for income-based measures of poverty and inequality. This is *Ils_dispy2*. It is comprised of the standard INCOMELIST of *Ils_dispy* (see Section 3.2 above) together with the variable *xivot* (which may or may not contain any value – see above).

In addition to the income lists there is a POLICY – *poverty_lines_mz* which specifies the poverty line to be used. The POLICY is a simple one comprising a single *Arithop* FUNCTION. This FUNCTION calculates the average poverty line.

The key PARAMETER is the formula which takes the constant for the average poverty line and multiplies it by the overall CPI factor *$f_CPI_total*. The output variable is *spl* and this is fed into the STATISTICS PRESENTER. The following screenshot illustrates the policy:

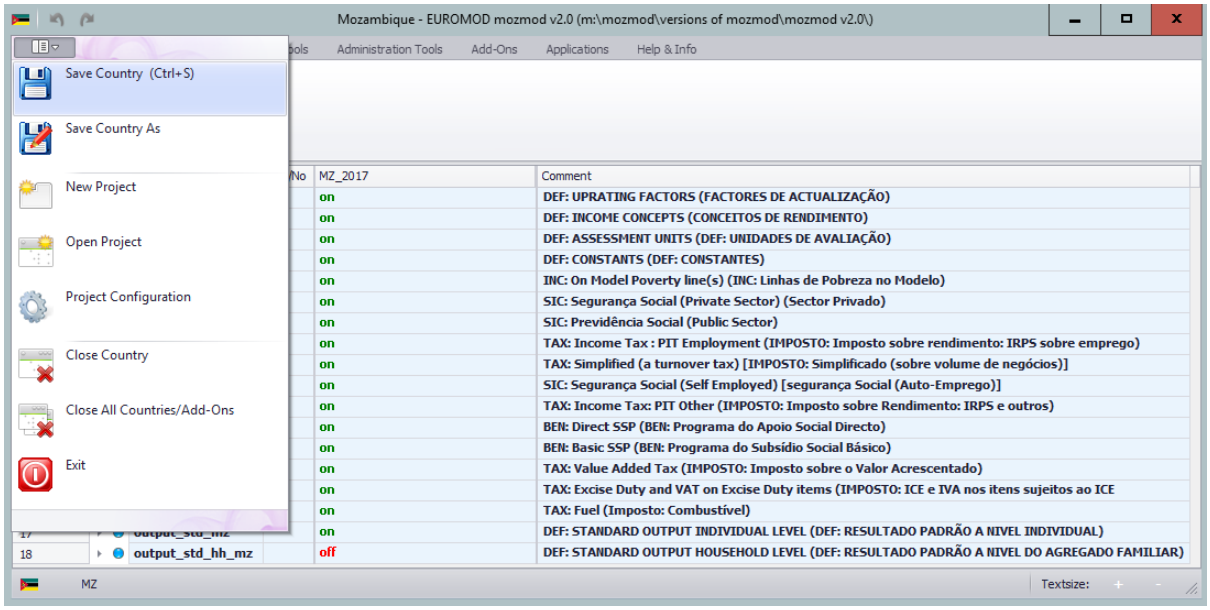


4.6 Other tasks

Saving changes

To save your changes open the main menu (above the *Run MOZMOD* button) and select the menu item *Save Country* (see Figure 4.6). Alternatively press *Ctrl-S*.

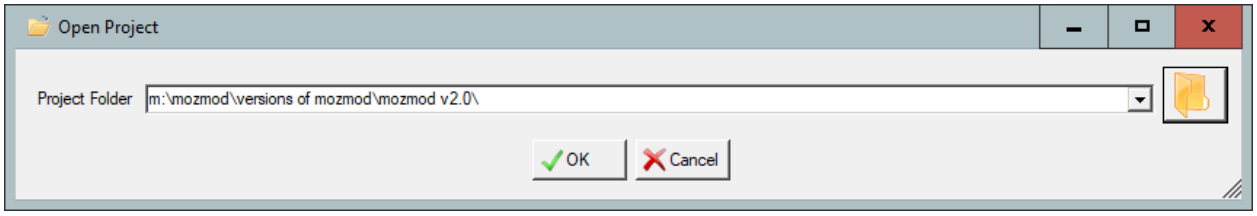
Figure 4.6: Saving changes



Opening a new project

The main menu also has an option to *Open Project*. This opens a dialog which allows the content displayed by the user interface to be changed, as well as the default input and output paths (see Figure 4.7). This may be required if, for example, you wish to create a version of MOZMOD for testing purposes, or a new version of MOZMOD is issued and you wish to switch to using the new version.

Figure 4.7: The *Open Project* dialog



In the dialog box you are asked to provide the locations of the EUROMOD (i.e. MOZMOD) Folder (e.g. C:\MOZMODv2.0).

References

Mitton, L., Sutherland, H. and Weeks, M. (Eds.) (2000) *Microsimulation Modelling for Policy Analysis,* Cambridge: University Press.

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

Zaidi, A. Harding, A. and Williamson, P. (Eds.) (2009) *New Frontiers in Microsimulation Modelling*, Vienna: Ashgate.