EUROMOD training course

Fiscal Policy Analysis Unit

Joint Research Centre

European Commission

Online course based on

EUROMOD I4.0+, software v. 3.4.10

26-28 April 2022

TRAINING EUROMOD

Joint Research Centre

Acknowledgement

This EUROMOD training is indebted to colleagues from ISER - University of Essex who made it happen until 2020. These slides and exercises were originally developed by them and are regularly updated by the JRC from 2021 onwards.

Course structure

Mix of lecture and hands-on exercises

Extra exercises offered to do after the course

Please ask questions

Tell us what you would like to do with the model

Course outline

- **Session 1:** Introduction to microsimulation models and EUROMOD.
- **Session 2:** EUROMOD language: policies, functions and parameters.
- **Session 3:** Policy functions *Elig* & *ArithOp*. EUROMOD handling errors.
- **Session 4:** Policy functions *BenCalc* & *SchedCalc*. Defining constants and income lists.
- **Session 5:** Policy function *Allocate*. Defining tax units.
- **Session 6:** Variable types and system functions *DefVar* & *DefOutput*. Uprating indices.

Session 1

Introduction to microsimulation models and EUROMOD

In this session, you will learn about

- Microsimulation models: what they are, how they work and why they are useful
- What is EUROMOD, the European Commission (EC) microsimulation tool
- Some applications of EUROMOD
- How EUROMOD works: Input microdata, Model (tax-benefit policy code), Software
- Exercise 1: Running the model and using the Statistics Presenter for analysis

What are microsimulation models

- Encompass modelling techniques that:
  - Operate at the level of individual units (e.g. persons, households, vehicles, firms)
  - Apply rules to simulate changes in state or behaviour of these units
  - Estimate distributional outcomes after applying these rules at the micro level

- Focus on different areas, e.g.:
  - Traffic and transportation
  - Demand for health care
  - Spatial planning
  - **Tax-benefit (TB) policies**

Types of TB microsimulation models

- **Static:**
  - Pure policy effect → "morning-after" effect

- Behavioural:
  - Policy effect accounts for behavioural responses
    - Change in preferences estimated using microeconomic models (e.g. labour supply models)

- Dynamic:
  - People's characteristics are adjusted over time in response to natural processes and probabilities of relevant events
    - e.g. fertility, death, marriage/divorce, labour market status

Static TB microsimulation models

- Unit: households and persons living in the households
- Rules: tax-benefit policies, mostly:
  - Cash transfers
    - Means-tested benefits
    - Non-means-tested benefits
    - Pensions
  - Social insurance contributions (SIC)
  - Direct taxes: Income tax, including tax allowances and tax credits, capital gains tax
- Outcomes: income distribution, measures of income inequality and poverty, (net) budgetary cost of policy changes, gainers and losers from policy changes, indicators of work incentives... and changes to them

Why TB microsimulation models

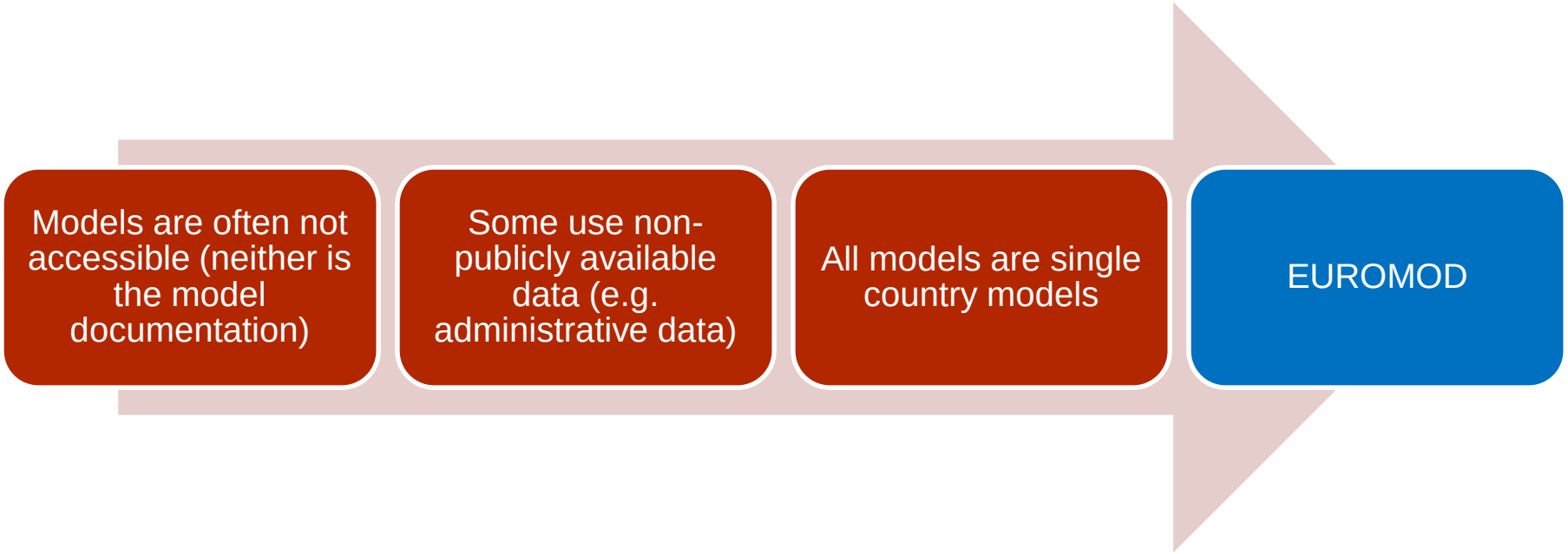
- Account for interactions between parts of the TB system
  - E.g. old-age pension is taxable, so higher pensions mean more income tax revenue
  - E.g. means-tested benefits depend on after-tax earnings, so a rise in income tax rate means higher spend on benefits
- Allow for "ceteris paribus" type of analysis, looking into the impact of changes in one policy holding everything else constant (useful for "understanding" as well as "monitoring")
- Bring added value to survey/register microdata, by providing information which is otherwise not (publicly) available
- Produce more up-to-date results for simulated variables, as data collection and release take time

Outcomes of TB microsimul. models

- Models enable analysis of the impact of changes in a policy:
  - On the mean of an outcome across several units, as with regression techniques
  - For one specific individual unit, as with OECD-style standard family type calculations
  - On the whole distribution of an outcome across units (distributional analysis)
  - On the government's budget (fiscal analysis)
  - On work incentives

Why EUROMOD

Models are built and maintained by researchers (for academic purposes) or ministries and civil service (for impact assessment of policies)



General features of EUROMOD

- The static TB microsimulation model for the EU-27
- Open-access, flexible, free and transparent
- No hardwiring, everything is parameterised
- (relatively) Easy to simulate structural reforms
- Extensive and growing documentation and community of users
- Uses flexible and user friendly interface
- Provides consistent and meaningful cross-country comparisons

Key users of EUROMOD

Academic research in (e.g.) public economics & quantitative social policy	
International policy organisations	European Commission policy Directorates OECD, IMF, World Bank, UNICEF (hands-on & commissioned work)
National governments and public institutions	SK, MT, LT, LV, EE, EL, RO ….
Civil society: EUROMOD is the engine for simplified web-based models	EUROMOD-JRC Interface - European Commission https://euromod-web.jrc.ec.europa.eu/euromod-jrc-interface SORESI - Austrian Ministry for Social Affairs http://soresi.sozialministerium.at/soresi
A platform for developing non-EU models	e.g. Africa (South Africa, Namibia, Mozambique, +4), Latin America (Ecuador, Colombia, Mexico…), Russia, Serbia ...

Key uses of EUROMOD

- EUROMOD can be used to analyse the impact of actual, proposed or hypothetical TB policy changes

- It can also be used as a tool in other type of analysis by adjusting the input data set:
  - How would a rise in earnings inequality affect the distribution of income?
  - How does demographic change affect government revenues?

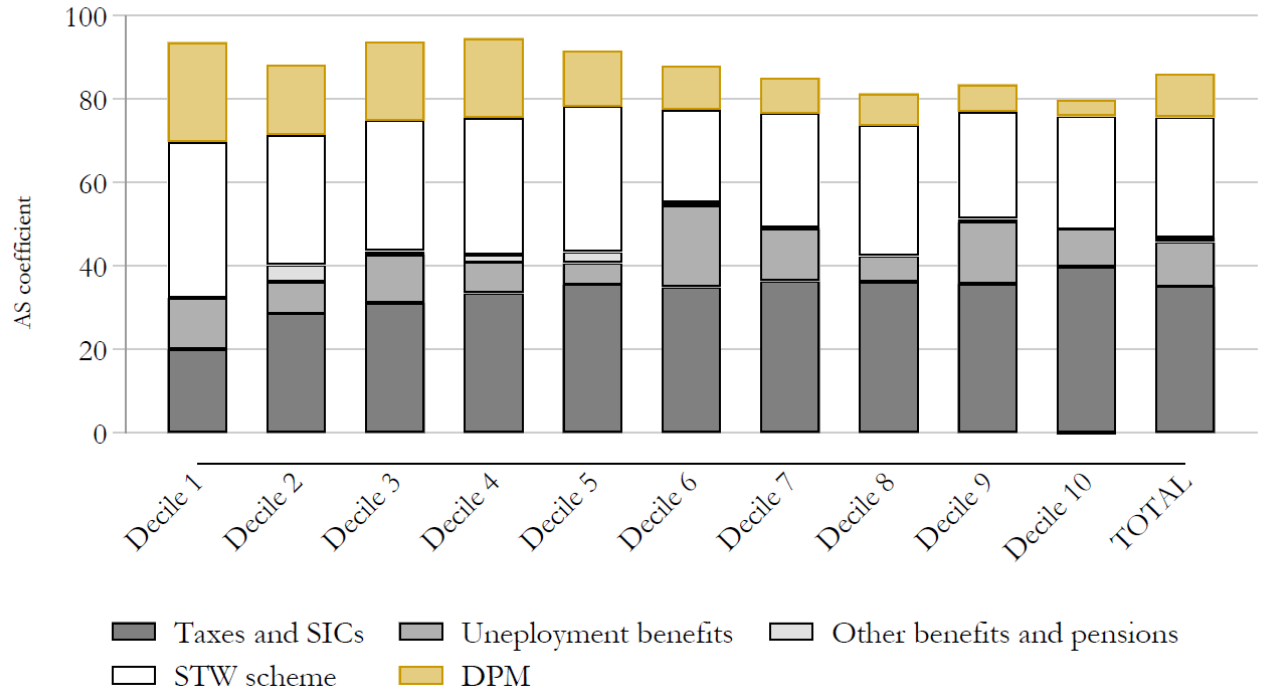
- Or by doing further analysis using the output data in combination with other models:
  - How does labour supply respond to a change in tax credits?

Additional uses of EUROMOD

- Complex reforms (e.g. revenue-neutral packages, policy swaps)
- Counterfactual ("what if") scenarios (e.g. stress test for unemployment shocks)
- Design regional/national/EU-wide policy reforms
- Extending policy scope with additional microdata
  - Indirect taxes (HBS data)
  - Wealth and property taxes (HFCS data)
  - Tax evasion and non-take-up simulation/calibration (administrative data)
- Link to macro models, providing a micro perspective within a macro type of analysis (e.g. links with CGE and DSGE models)
- Hypothetical Household Tool (HHoT), tool for creating households and individuals with user-defined characteristics

Example 1: existing policies

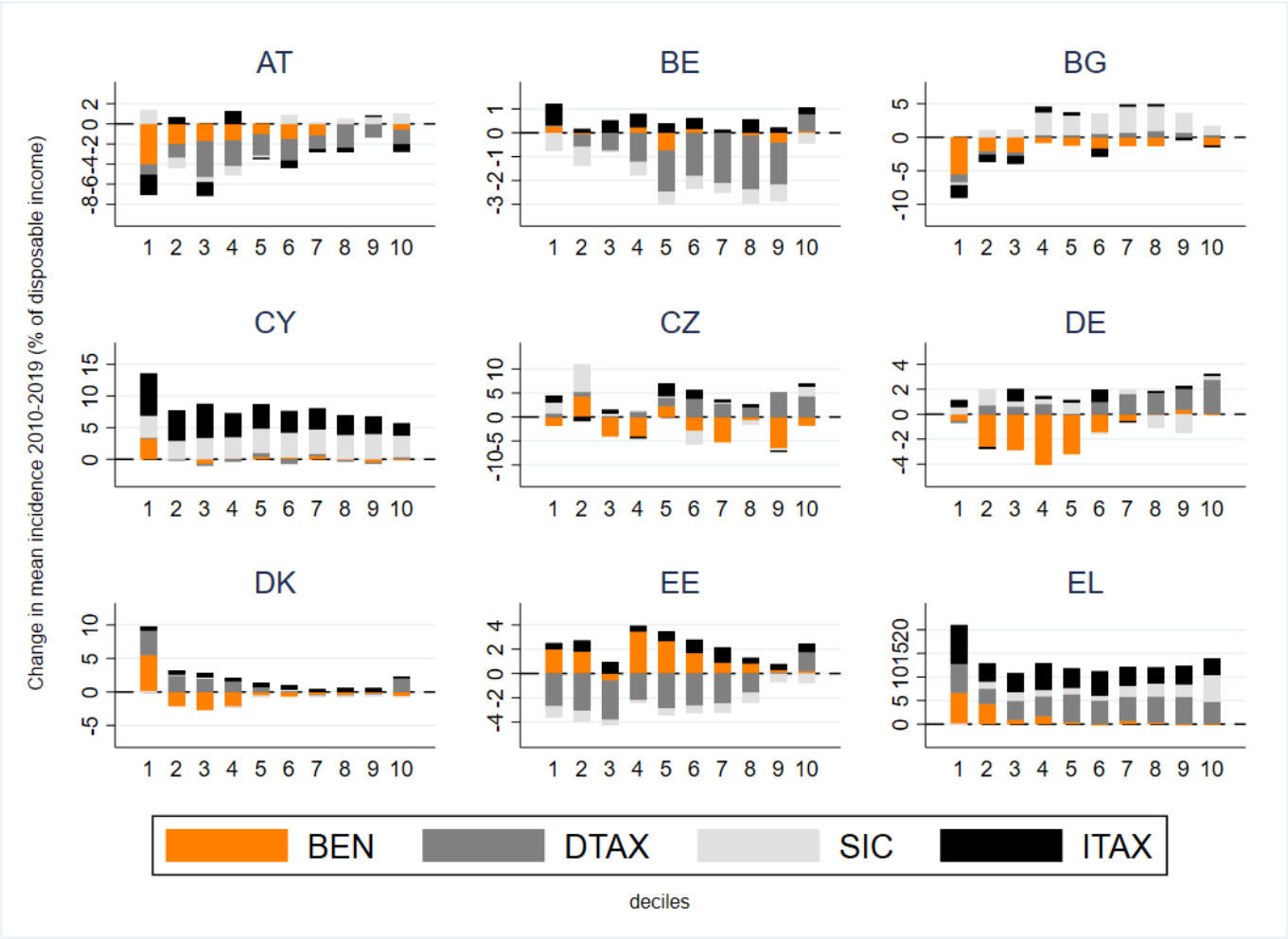
- Income stabilisers during the COVID-19 crisis in Germany



Source: Christl, De Poli, Hufkens, Peichl, and Ricci (2022), *International Tax and Public Finance.*

Example 2: actual policy changes

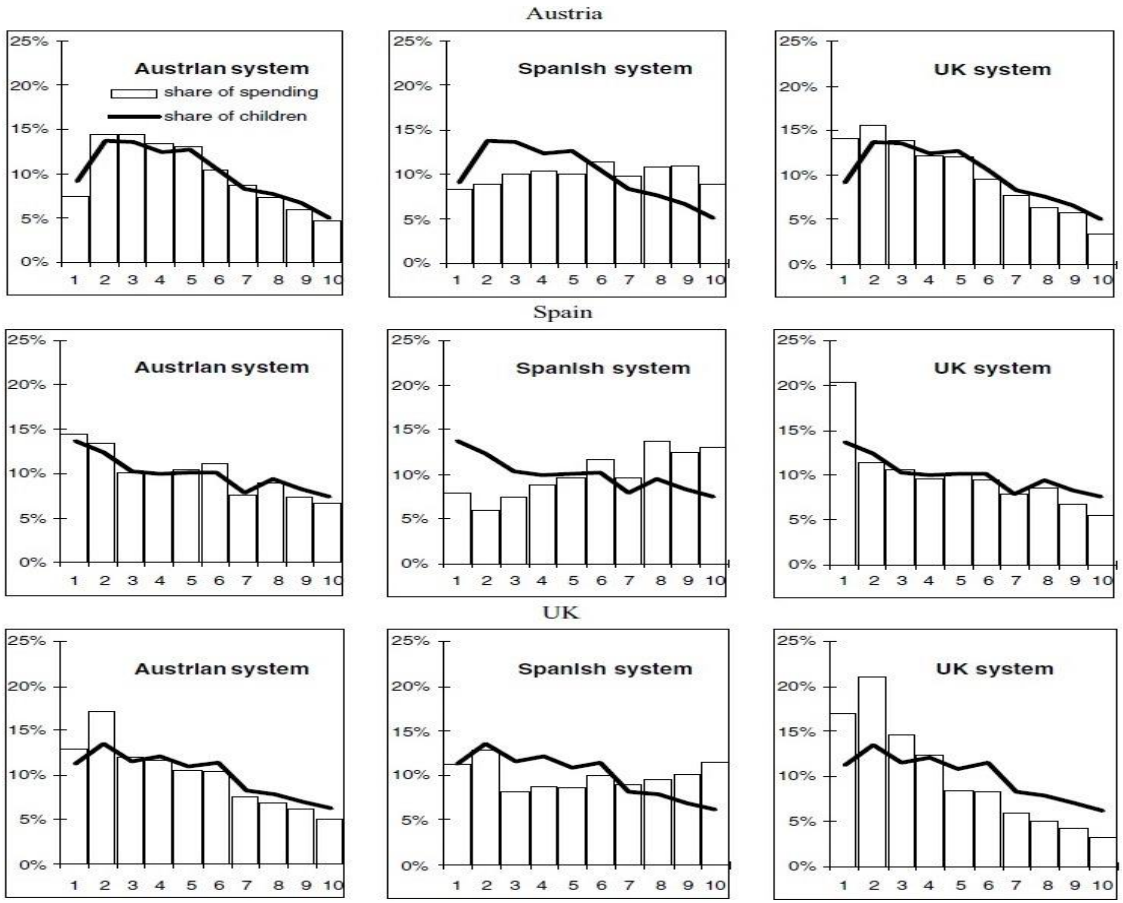
- The distributive impact of fiscal policy in the pre-COVID decade, 2010-2019 (selected countries)



Source: Maier and Ricci (2022), *Working Paper*

Example 3: policy swaps

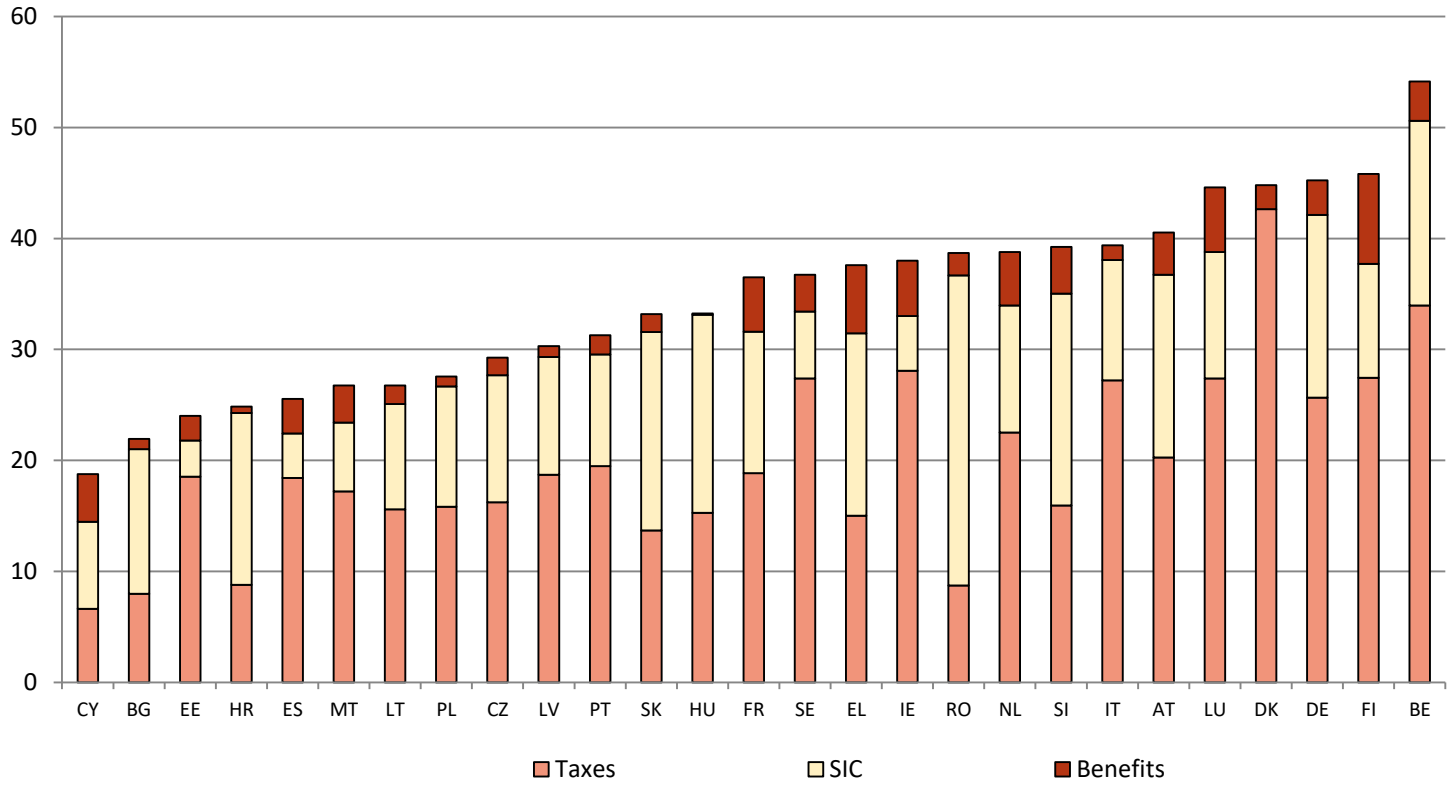
- Impact on household incomes of own and other countries' child-related policies



Source: Levy et al. (2007), *Journal of Social Policy*

Example 4: work incentives

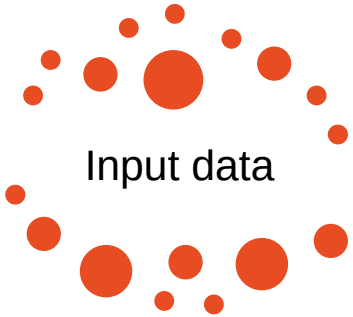
- Decomposing the mean Marginal Effective Tax Rate by type of policy in 2018



Source: Maier, Ricci et al (2022), *JRC Working Papers on Taxation and Structural Reforms No 1/2022*

Notes: METRs measure the incentive to work longer hours

How EUROMOD works



- on households (and individuals)
- representative of the population

Tax-benefit policy rules (Model)

- current or historical systems
- hypothetical systems

Calculation of benefit entitlements & tax liabilities (Software)

- for every household
- under multiple tax-benefit systems

Analysis

- household incomes
- fiscal impact
- work incentives

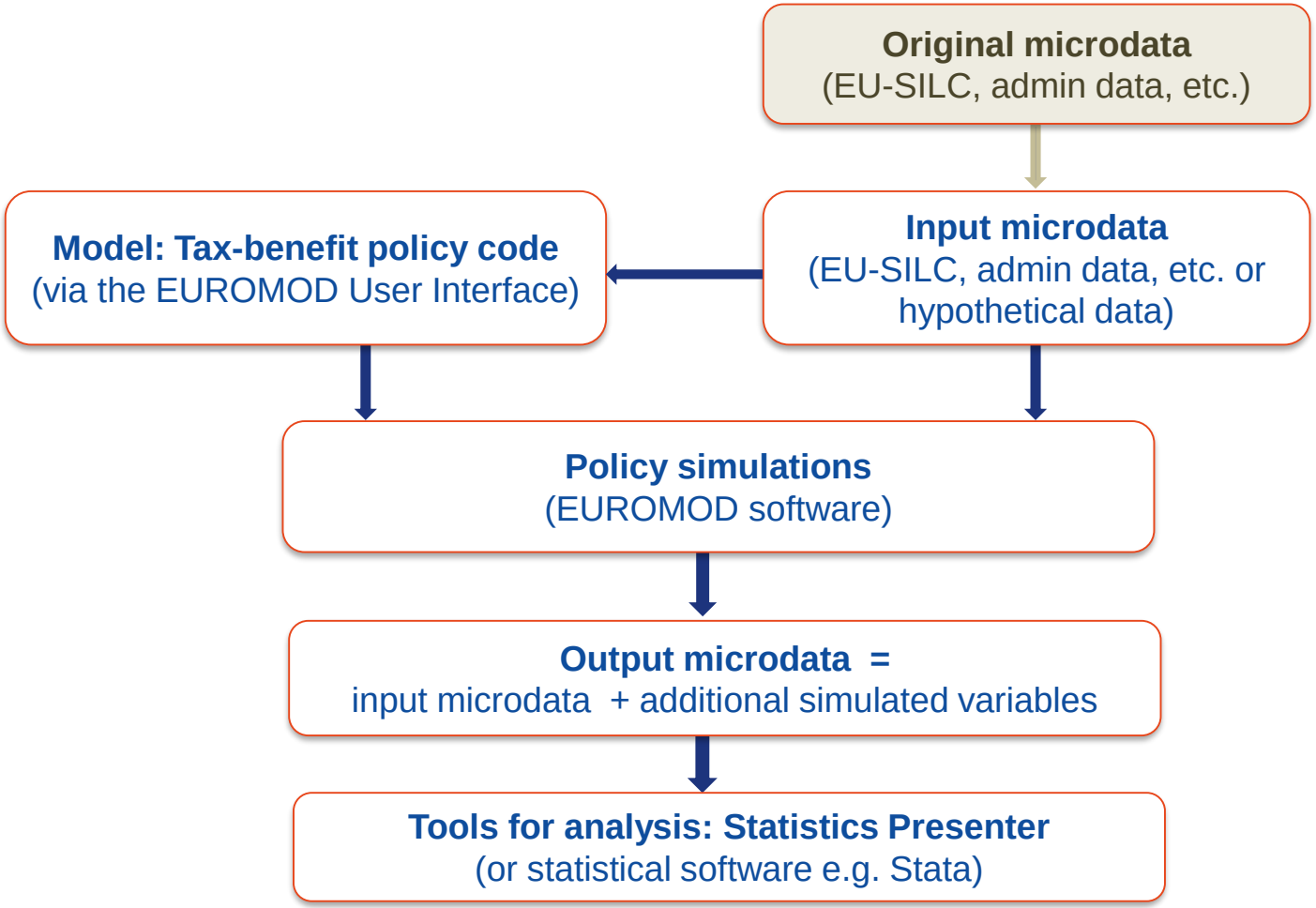
Access

- Input data → upon request
- Model → open source and downloadable for free
- Software → open source code and end-user licenced executable, downloadable for free

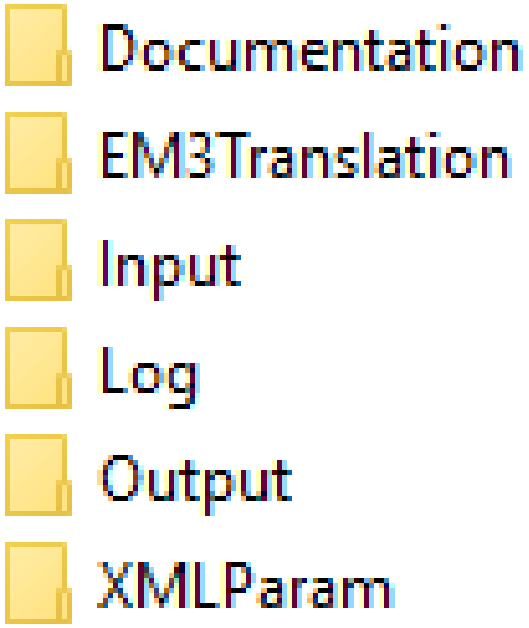
Account for:

- interactions between policies
- interactions between policies and household characteristics
- heterogeneity of household characteristics

EUROMOD workflow



EUROMOD project folder



→ Help and documentation in pdf

→ [folder for internal use of EUROMOD]

→ Default folder to store input data

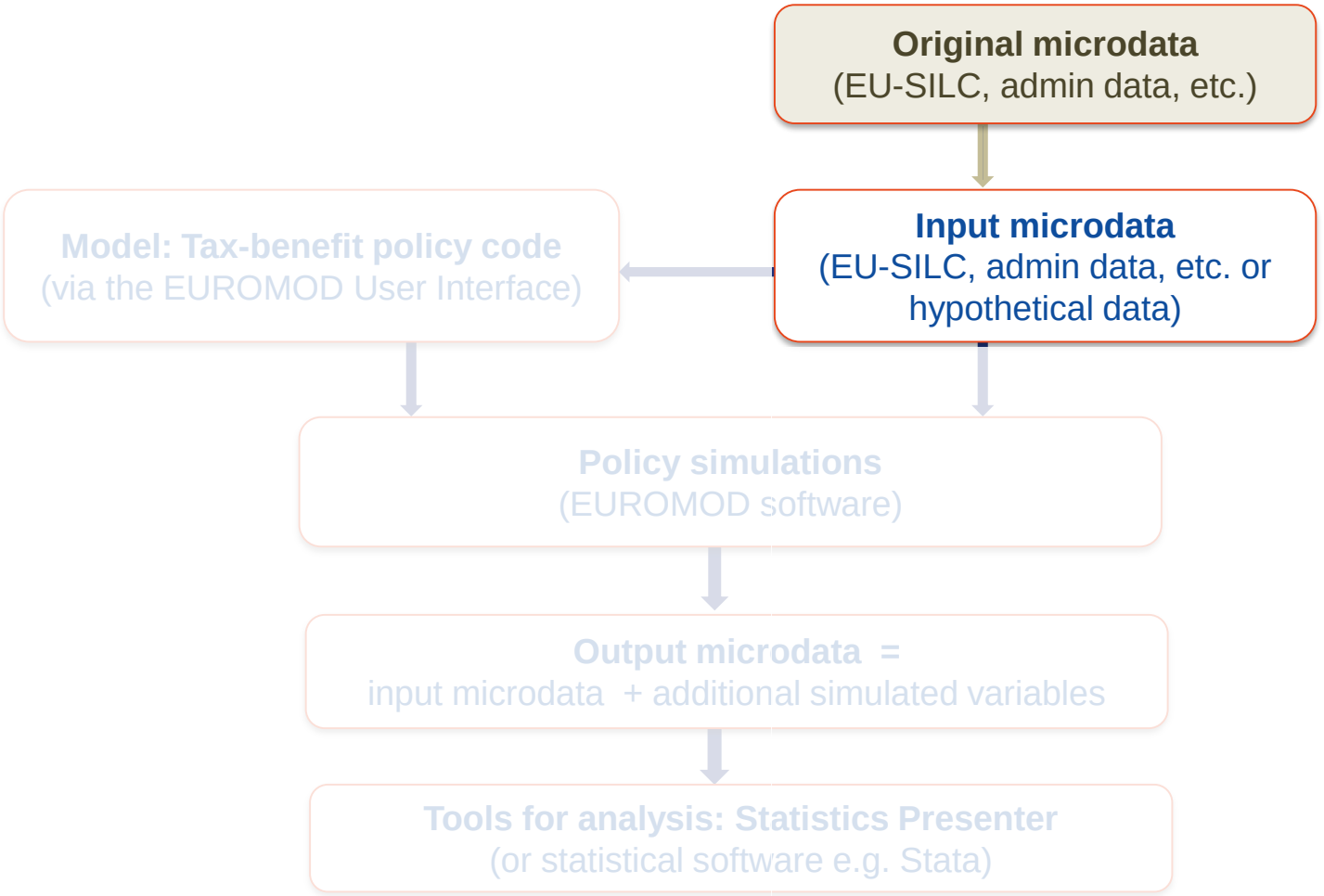
→ EUROMOD log with all versions

→ Default folder to save output data

→ Country files, where policies are coded!

EUROMOD in this course

- Input microdata
  - Training data
- Model (tax-benefit code):
  - Version I4.0+
  - Models for all EU member states: policy years (systems) for 2005/07-2021 for most countries and 2011-2021 for Croatia
- EUROMOD Software version: 3.4.10



EUROMOD
INPUT MICRODATA

Sources of input data

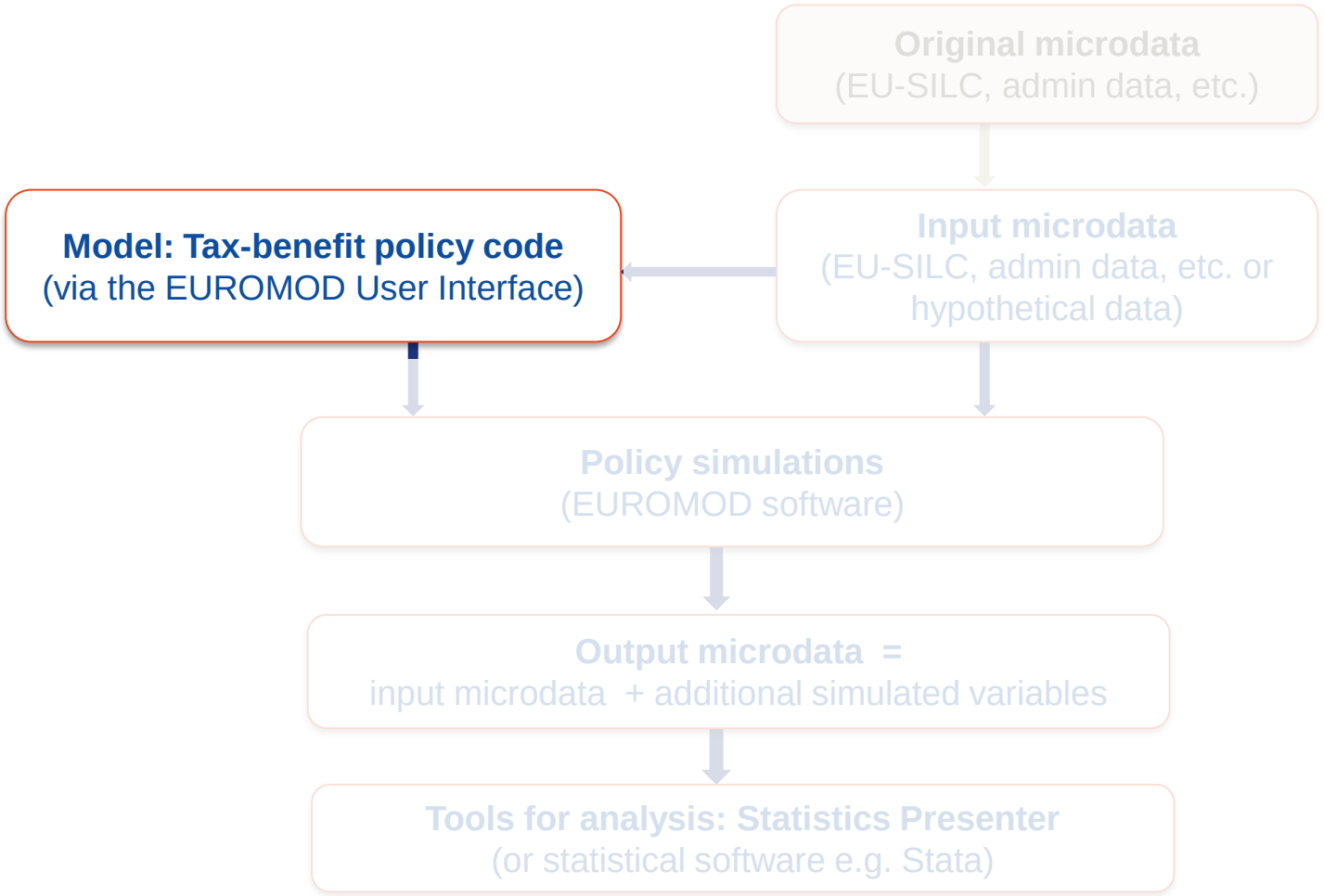
- Default datasets: based on **household survey microdata from EU-SILC;**
- **Any other income source can be adapted to be used in EUROMOD** (e.g. administrative data);
- **EUROMOD's Hypothetical Household Tool (HHoT)** allows to generate input files with model households
  - Households and individuals with user-defined characteristics
  - Abstracts from complexities of real data - focus on specific household types (Gasior & Recchia 2019).

Characteristics of input data

- Variables: demographic, labour, income, assets, expenditure
- Gross incomes, at the individual level
- Monetary variables recoded to (average) monthly basis
- No missing values

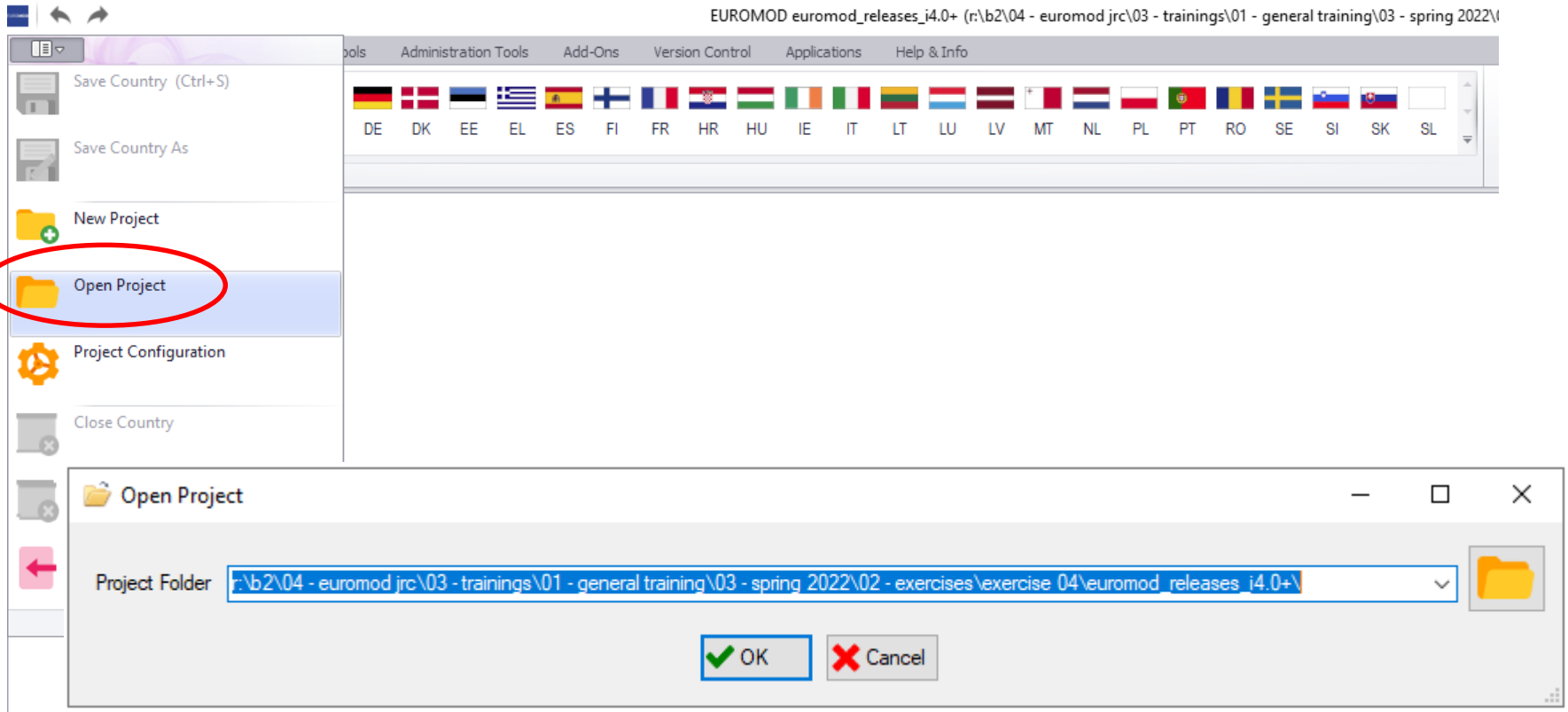
A typical input dataset

	identifiers					age	wage	self-employment income	pension
	idhh	idperson	idpartner	idmother	idfather	dag	yem	yse	poa
1	1	101	102	0	0	65	0	0	2429.72
2	1	102	101	0	0	60	0	0	0
3	1	103	0	102	101	30	0	0	0
4	1	104	0	102	101	28	157.42	0	0
5	2	201	202	0	0	29	1075.35	0	0
6	2	202	201	0	0	25	0	951.36	0
7	2	203	0	202	201	3	0	0	0
8	2	204	0	0	0	2	0	0	0
9	3	301	302	0	0	72	0	0	1959.98
10	3	302	301	0	0	59	0	0	0
15	5	501	0	0	0	86	0	0	1853.7



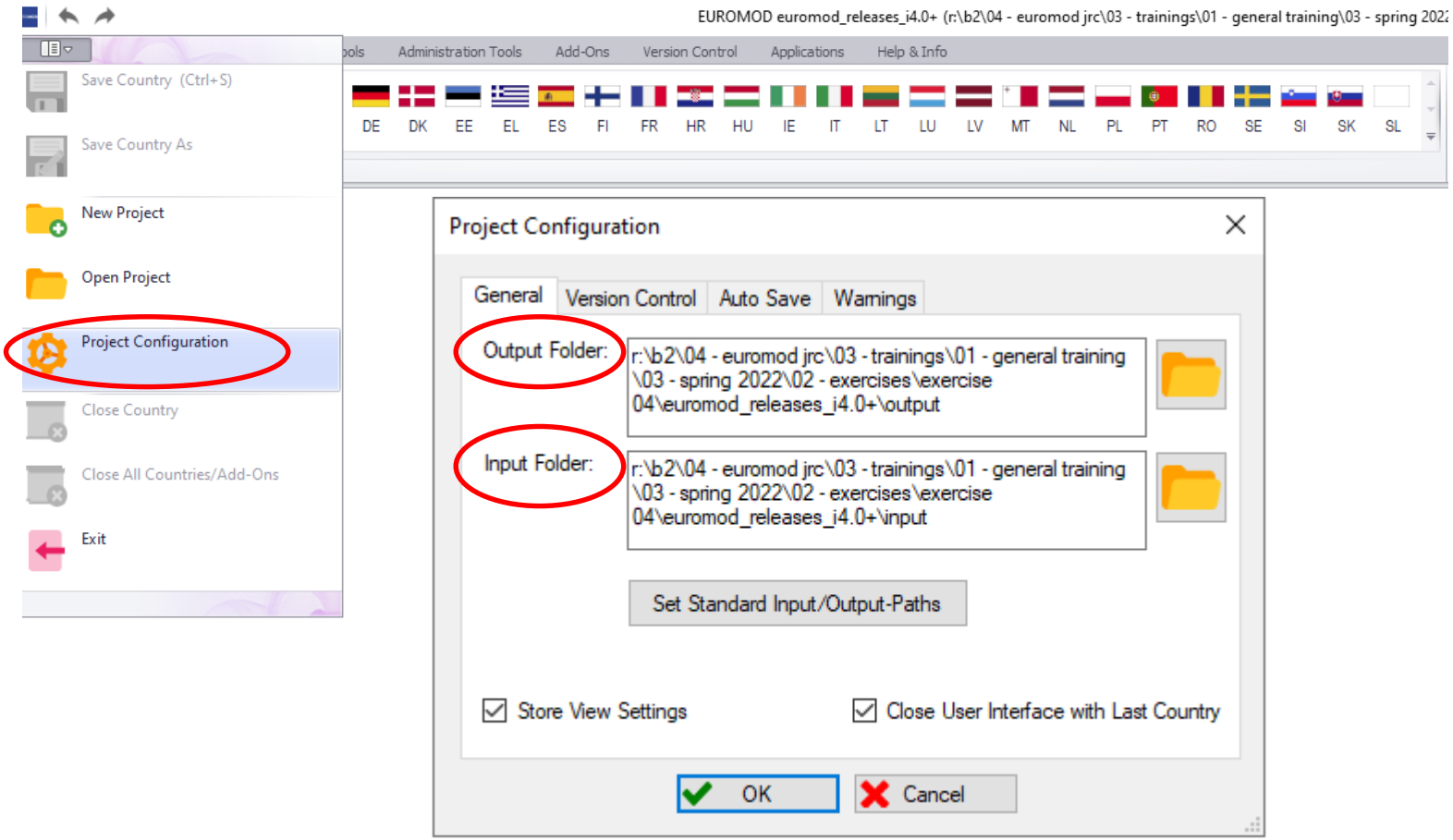
EUROMOD VIA THE USER INTERFACE (UI)

Opening a project



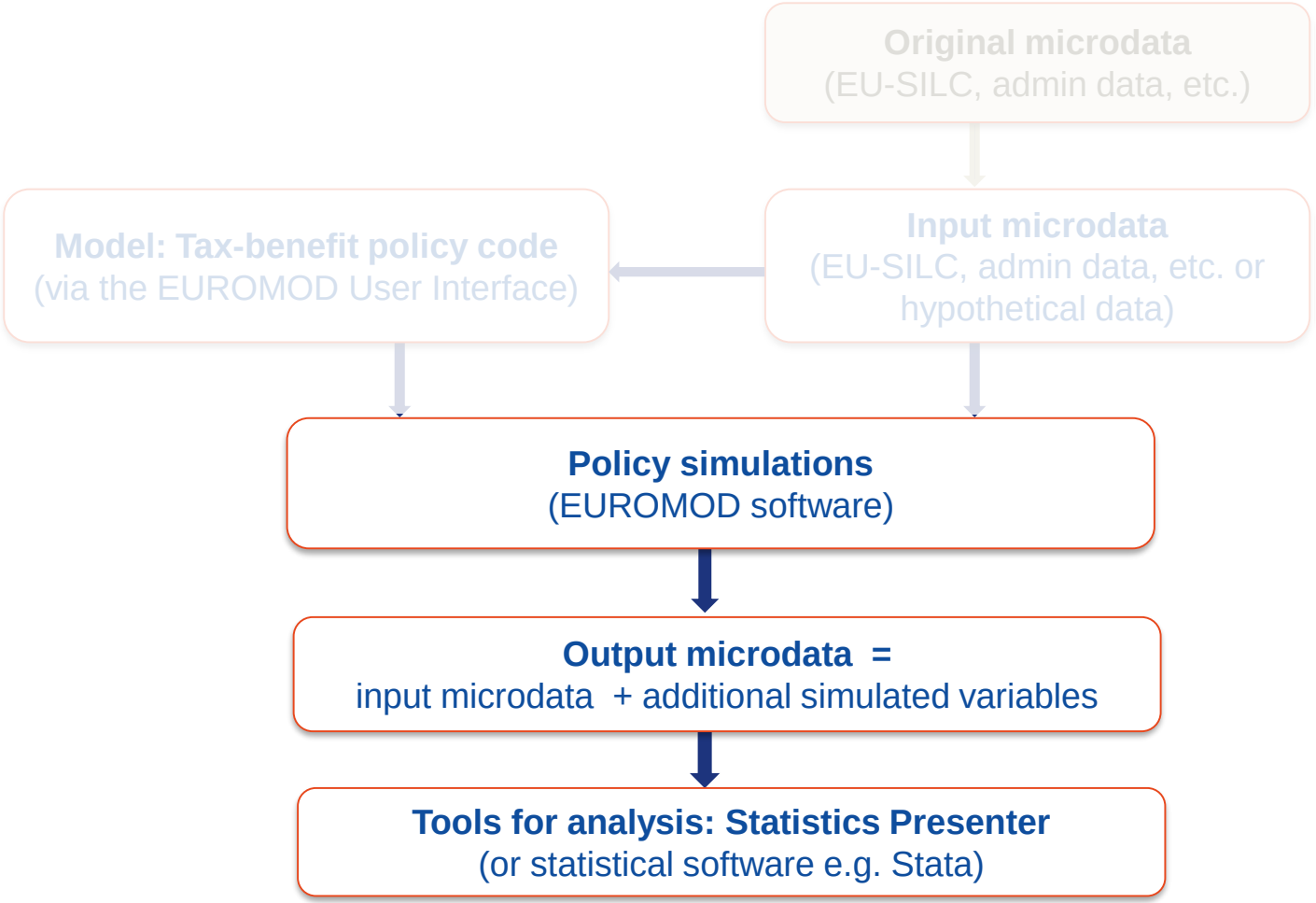
European Commission

Configuring the output/input folders



Selecting and running a country

European Commission



RUNNING EUROMOD, PRODUCING OUTPUT DATA & ANALYSIS

Running the model

- Two ways of Running the model:
  - by clicking on button *Run EUROMOD*
  - by clicking on a country flag and then clicking on the button *Run EUROMOD*

- We can run the model (simultaneously) for:

i) one country and one or more systems

ii) several countries (and systems)

iii) all countries and systems and for all data

Analysing the output data

- Running the model produces a txt file containing the output data. This will include all the input data variables plus the variables simulated by the model (with subscript _s)

- The file is stored in the Output folder and for the baseline model it is called cc_year_std (e.g. pt_2006_std)

- This file can be imported to any statistical software for analysis. Additionally, the EUROMOD tools *Statistics Presenter* and *In-depth analysis* can be used to produce several indicators (more on this in the next slides)

A typical output dataset

	identifiers					age	wage	self-employment income	pension	personal income tax
	idhh	idperson	idpartner	idmother	idfather	dag	yem	yse	poa	tin_s
1	1	101	102	0	0	65	0	0	2429.72	176.58
2	1	102	101	0	0	60	0	0	0	0
3	1	103	0	102	101	30	0	0	0	0
4	1	104	0	102	101	28	157.42	0	0	0
5	2	201	202	0	0	29	1075.35	0	0	4.32
6	2	202	201	0	0	25	0	951.36	0	36.94
7	2	203	0	202	201	3	0	0	0	0
8	2	204	0	0	0	2	0	0	0	0
9	3	301	302	0	0	72	0	0	1959.98	82.22
10	3	302	301	0	0	59	0	0	0	0
15	5	501	0	0	0	86	0	0	1853.7	101.49

EUROMOD tools for analysing output data

Two plugins to obtain summary statistics from EUROMOD ouput microdata:

Statistics Presenter

- Descriptive statistics of income and population
- Predefined tables for budgetary and distributional impact

In-depth Analysis

- Customised tables for budgetary and distributional impact of reforms

The Statistics presenter

- Four templates:
  1. **Default** - several scenarios, independent tables
  2. **Multisytem** - several scenarios, tables in parallel
  3. **Baseline/reform** - one baseline and several reforms, comparative tables with deciles and poverty lines fixed in the baseline
  4. **Variable** - descriptive statistics for a single variable

- Results shown on screen and exportable to Excel
- The statistics presenter is the most simple tool of result analysis in EUROMOD. This is what we will be using during this training.

In-depth Analysis plugin

- Three categories:
  1. **Fiscal** (4 tables) → full disaggregation of taxes and benefits: totals and taxpayers/beneficiaries for each
  2. **Distributional** (8 tables) → totals, means, taxpayers/beneficiaries and winners/losers, by selected breakdown variable(s) and target population
  3. **Inequality and poverty** (6 tables) → inequality, progressivity, redistribution and AROP rates and gaps

- Results shown on screen and exportable to Excel.
- The In-depth Analysis a more advance tool of result analysis which allows to customize analyses and tables.

Statistical and income concepts (1)

Disposable income

Market Income

+ Salaries
+ Self-employment income
+ Investment income
+ Property income
+ etc.

Taxes and SIC

- Personal Income Tax
- Employee Social Insurance Contributions
- etc.

Social transfers

+ Pensions
+ Family benefits
+ Minimum Income Schemes
+ etc.

Statistical and income concepts (2)

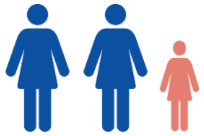
Equivalence scales

- Larger household size → higher income needs

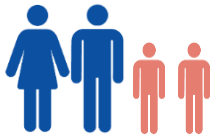
  but
- Sharing expenses and assets → needs do not double when size doubles

= 1

= 1 + 0.5 = 1.5



= 1 + 0.5 + 0.3 = 1.8



= 1 + 0.5 + 0.3 + 0.3 = 2.1

OECD-modified equivalence scale
First adult = 1
Each additional member ≥14 = 0.5
Each additional member <14 = 0.3

Statistical and income concepts (3)

Equivalised disposable income

= 1

DI = 10,000

EDI = 10,000/1 = 10,000

= 1 + 0.5 + 0.3 = 1.8

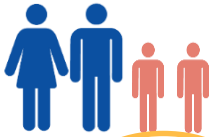
DI = 18,000

EDI = 18,000/1.8 = 10,000

= 1 + 0.5 = 1.5

DI = 30,000

EDI = 30,000/1.5 = 20,000



= 1 + 0.5 + 0.3 + 0.3 = 2.1

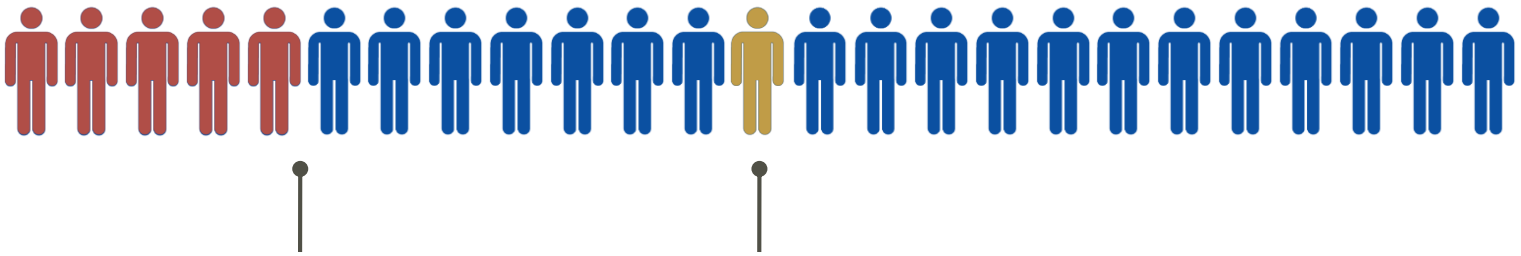
DI = 31,500

EDI = 31,500/2.1 = 15,000

Statistical and income concepts (4)

At-risk-of-poverty rate

Equivalised disposable income (in ascending order)



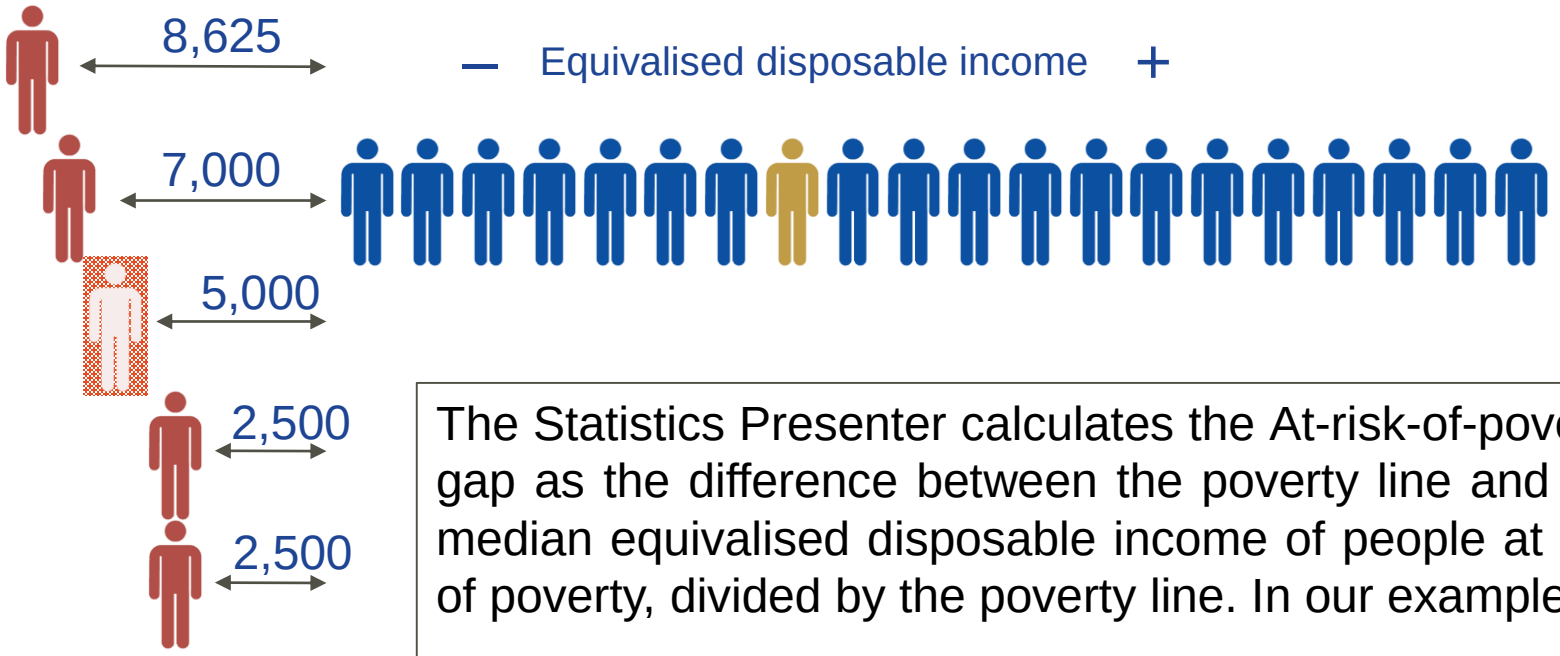
60% median = 8,625
(poverty line)

Median = 14,375

AROP rate = 5/25 = 20%

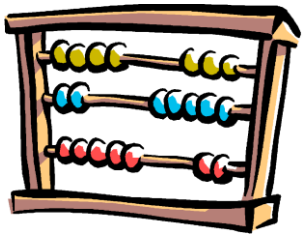
The Statistics Presenter: concepts (6)

At-risk-of-poverty gap



The Statistics Presenter calculates the At-risk-of-poverty gap as the difference between the poverty line and the median equivalised disposable income of people at risk of poverty, divided by the poverty line. In our example:

$$\text{AROP gap} = \frac{8{,}625 - 5{,}000}{8{,}625} = 42\%$$

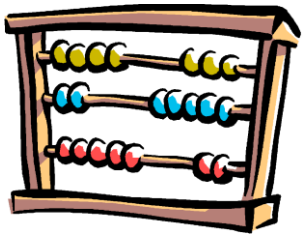


Exercise 1

Run EUROMOD and use the Statistics Presenter to analyse the results.

Run the policy systems for Austria (systems: 2019 and 2020) as well as for Finland, Italy and Spain (system: 2020 only). Then use the Statistics Presenter Tool to analyse the results and make cross-country comparisons on inequality and poverty.

We will do the exercise together!

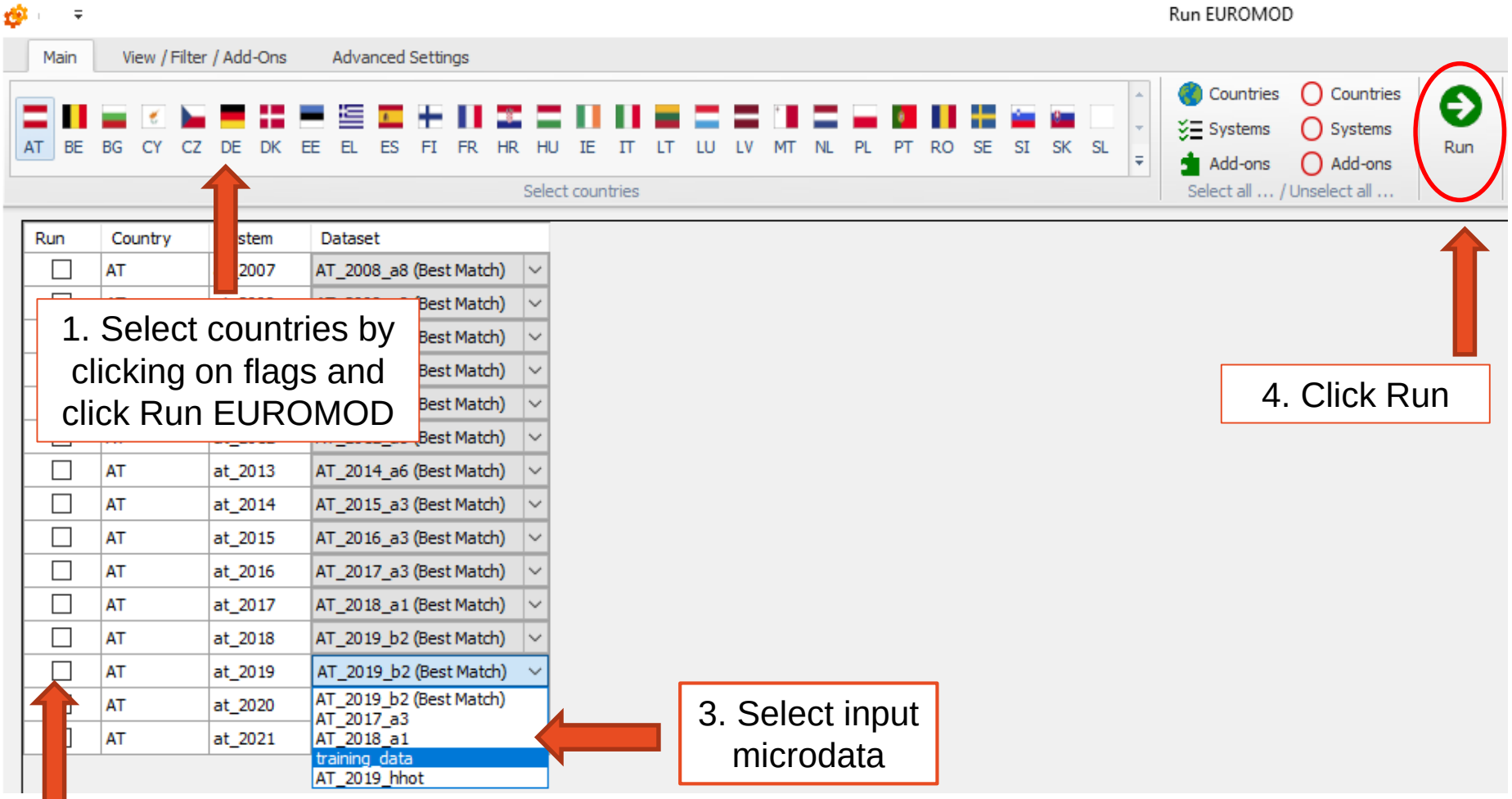


Exercise 1

Steps:

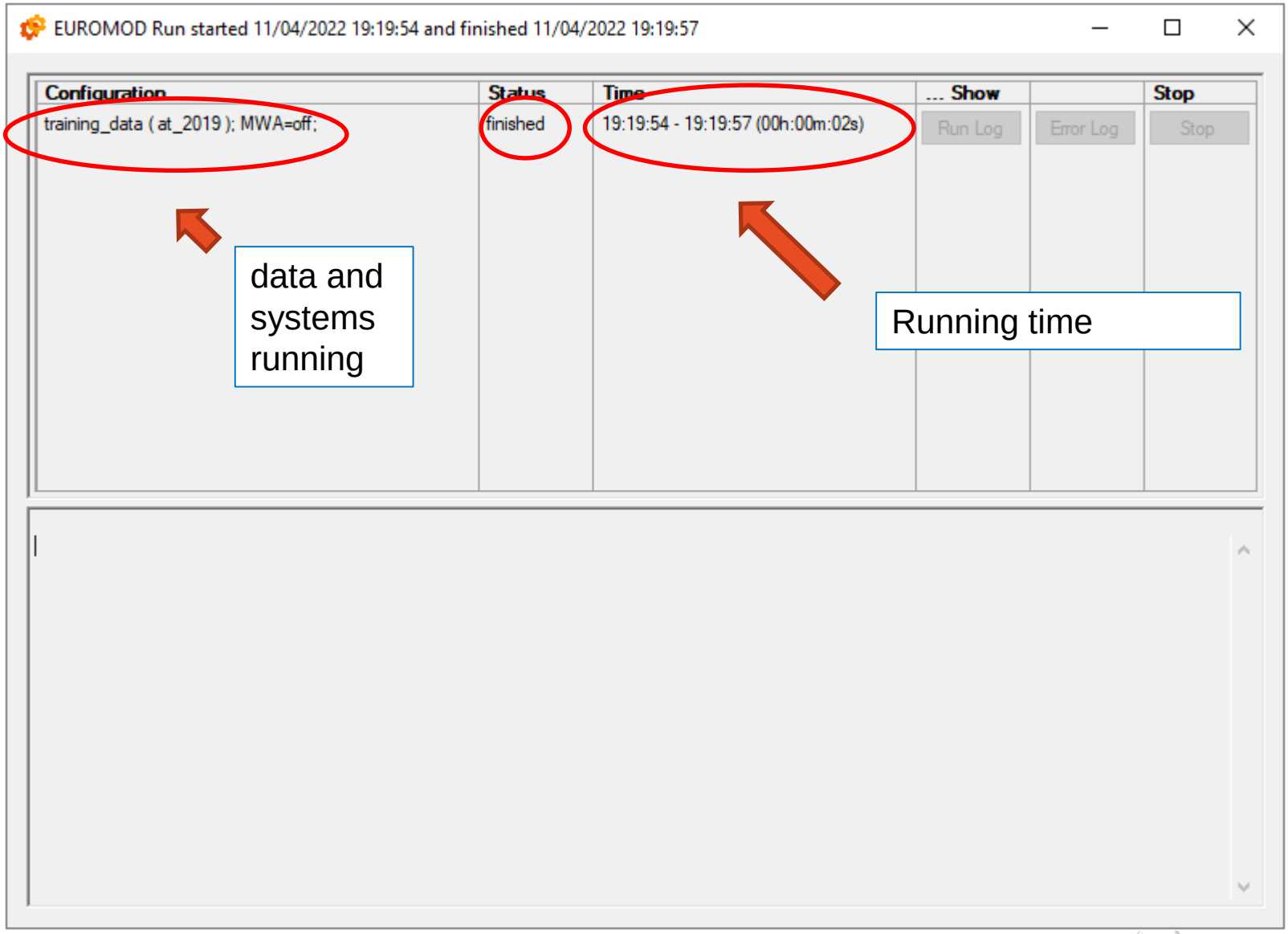
1. Run EUROMOD for Austria (system: 2019).
2. Run EUROMOD for Austria, Finland, Italy and Spain together (system: 2020).
3. Use the Statistics Presenter Tool – Default option – to produce results for the income distribution in Austria for 2019 and 2020 and analyse the results.
4. Use the Statistics Presenter Tool – MultiSystem option – to produce results for the income distribution for these four countries for 2020 and analyse the results.

Step 1: Run AT 2019 (I)



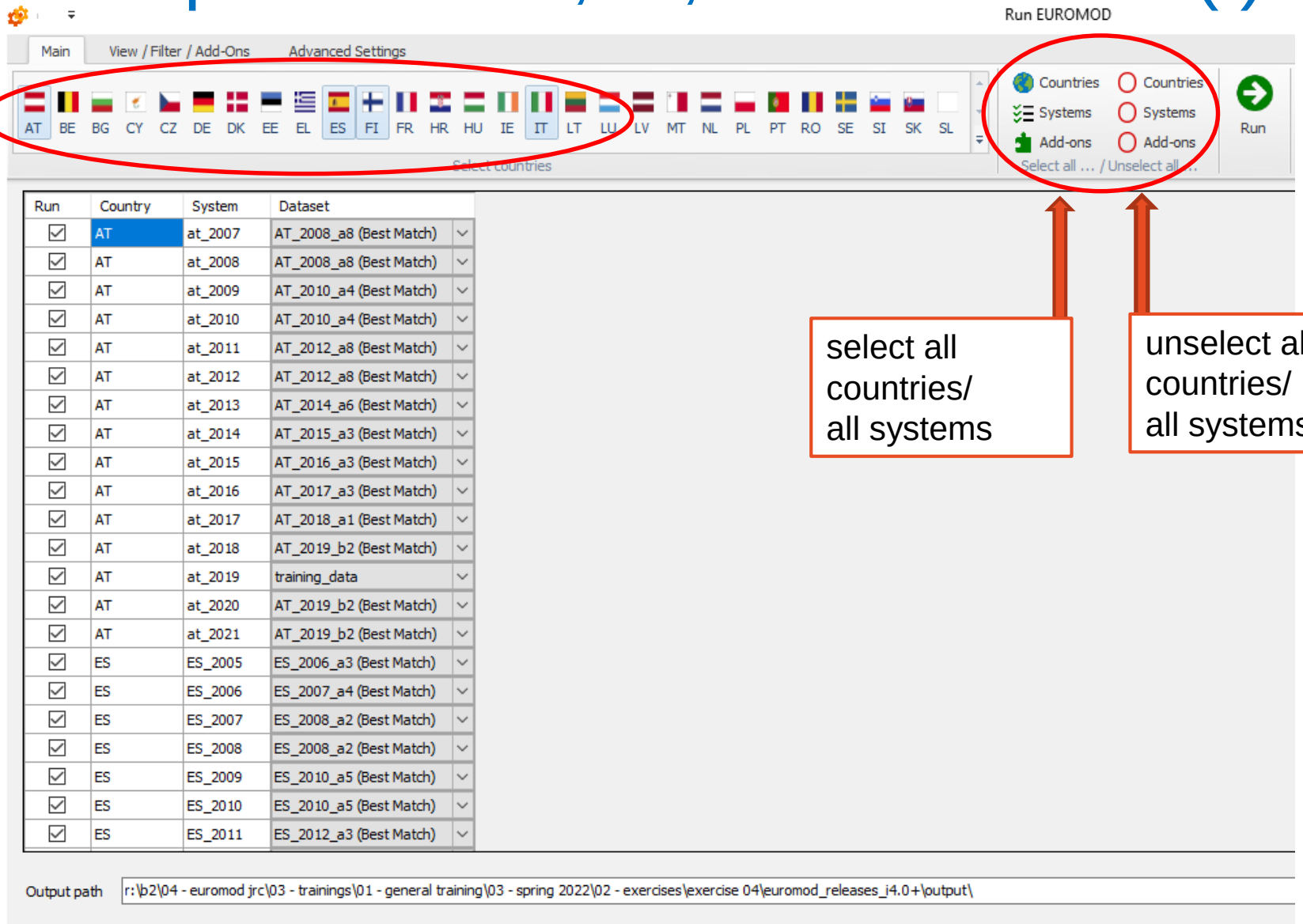
European Commission

Step 1: Run AT 2019 (II)

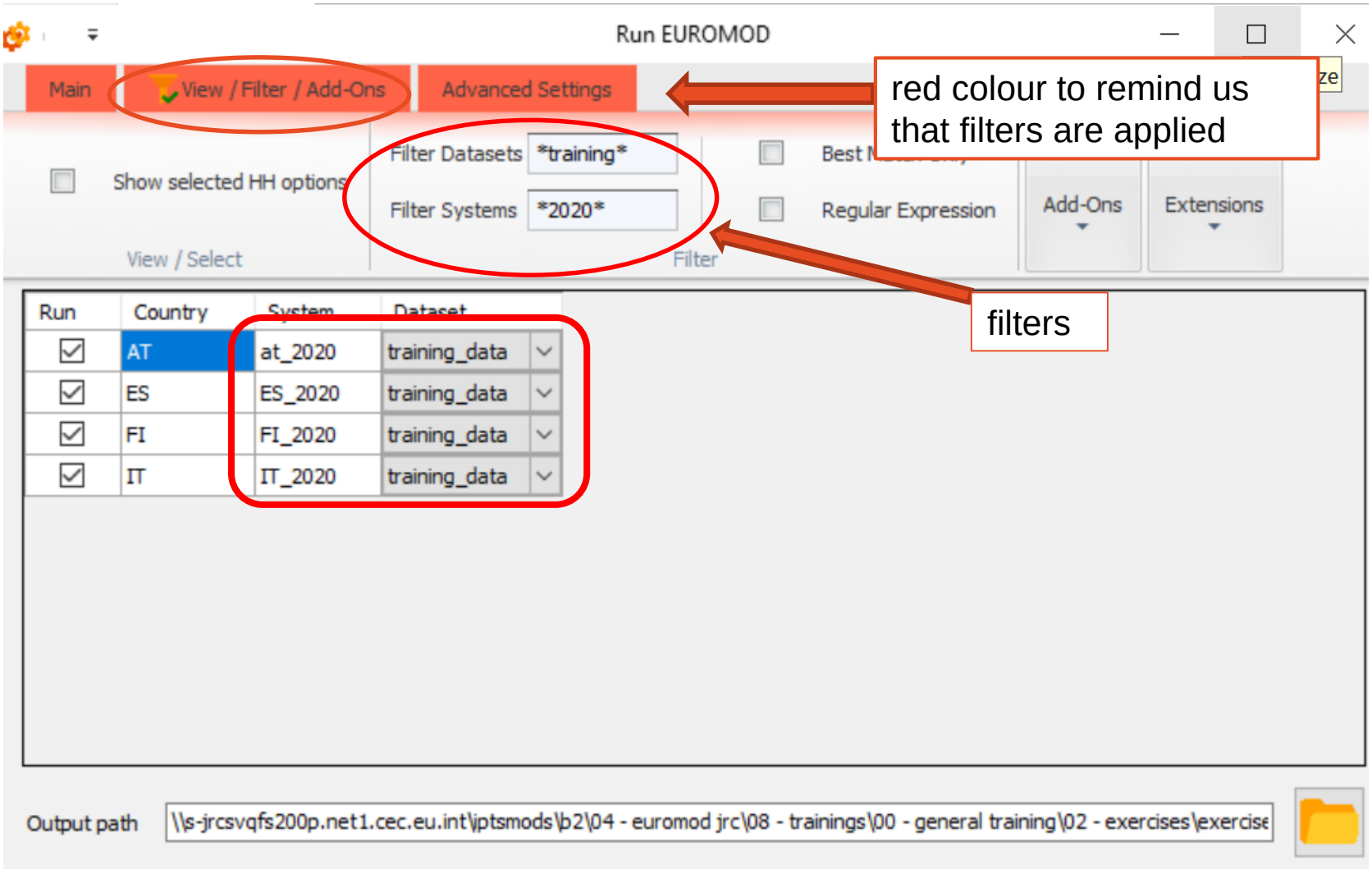


European Commission

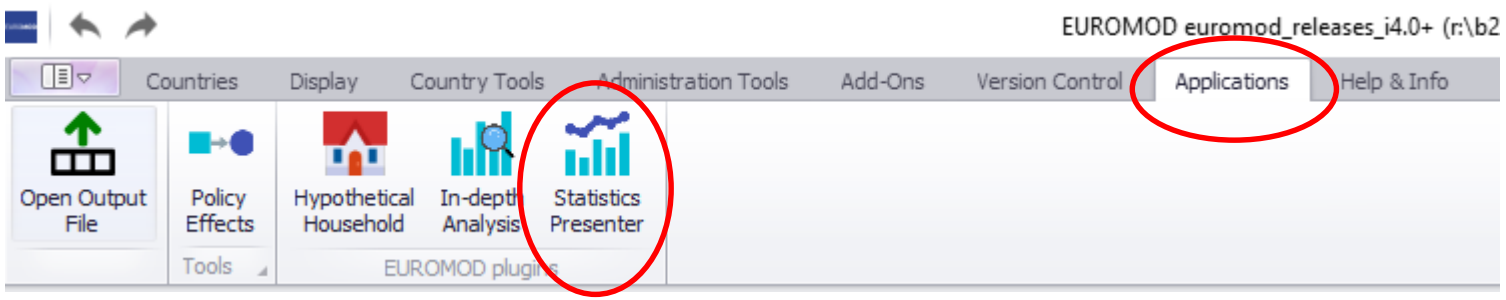
Step 2: Run AT, FI, IT & ES 2020 (I)

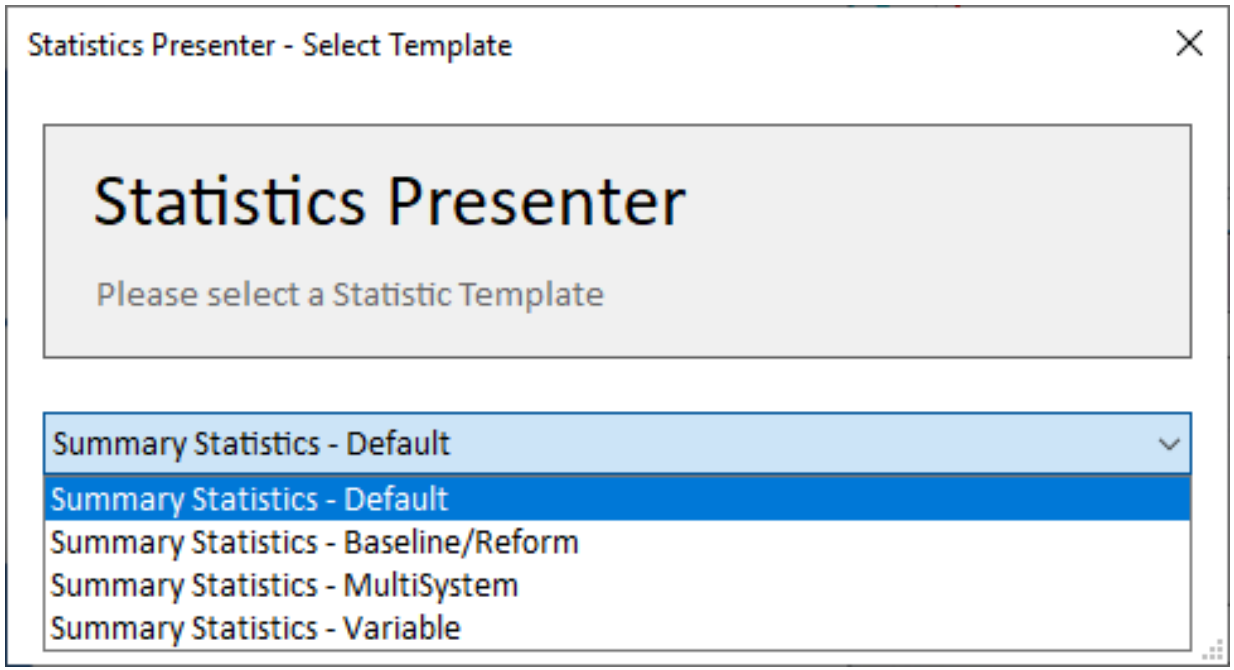


Step 2: Run AT, FI, IT & ES 2020 (II)

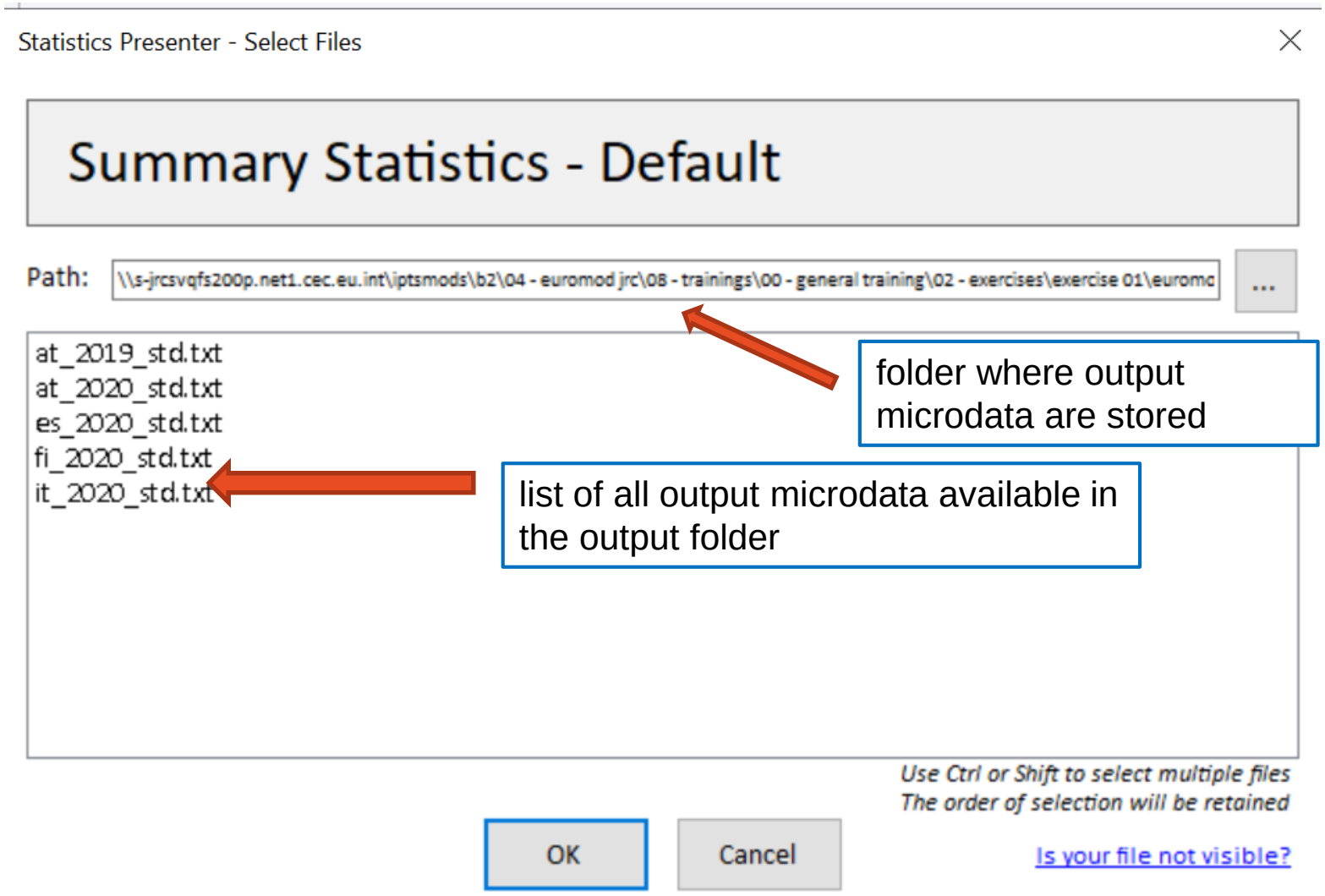


Step 3: Statistics Presenter Default (I)

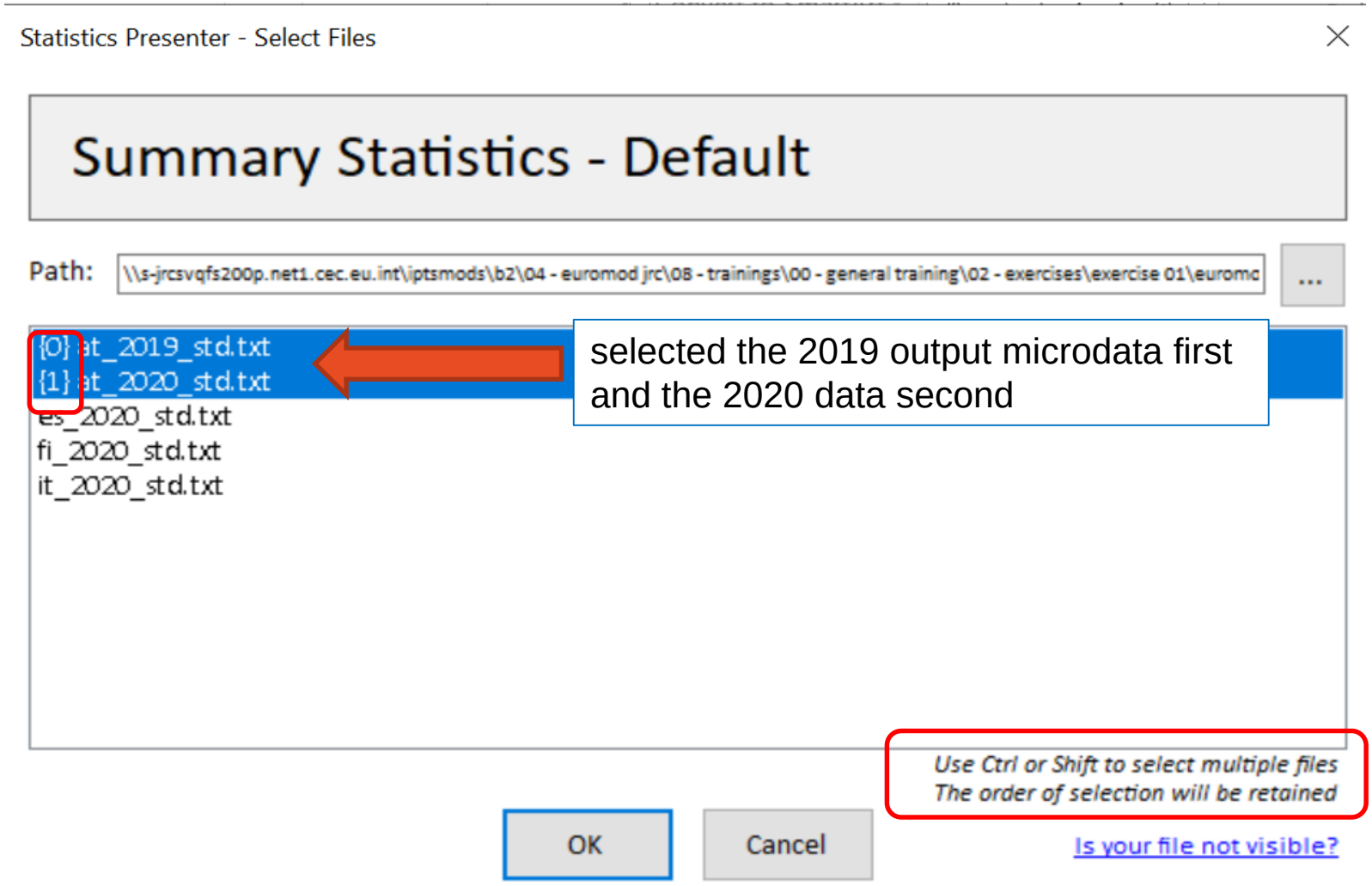




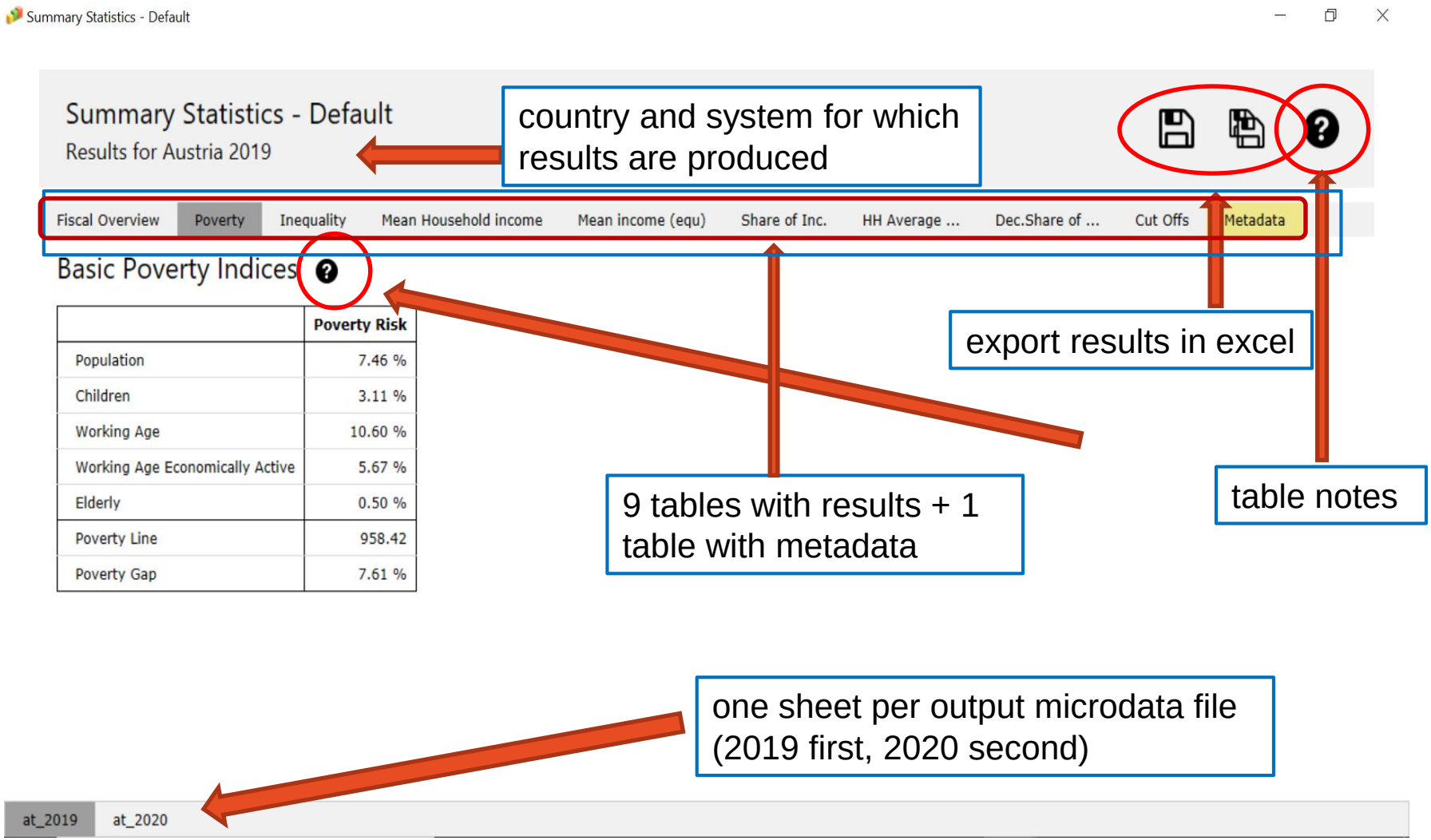
Step 3: Statistics Presenter Default (II)



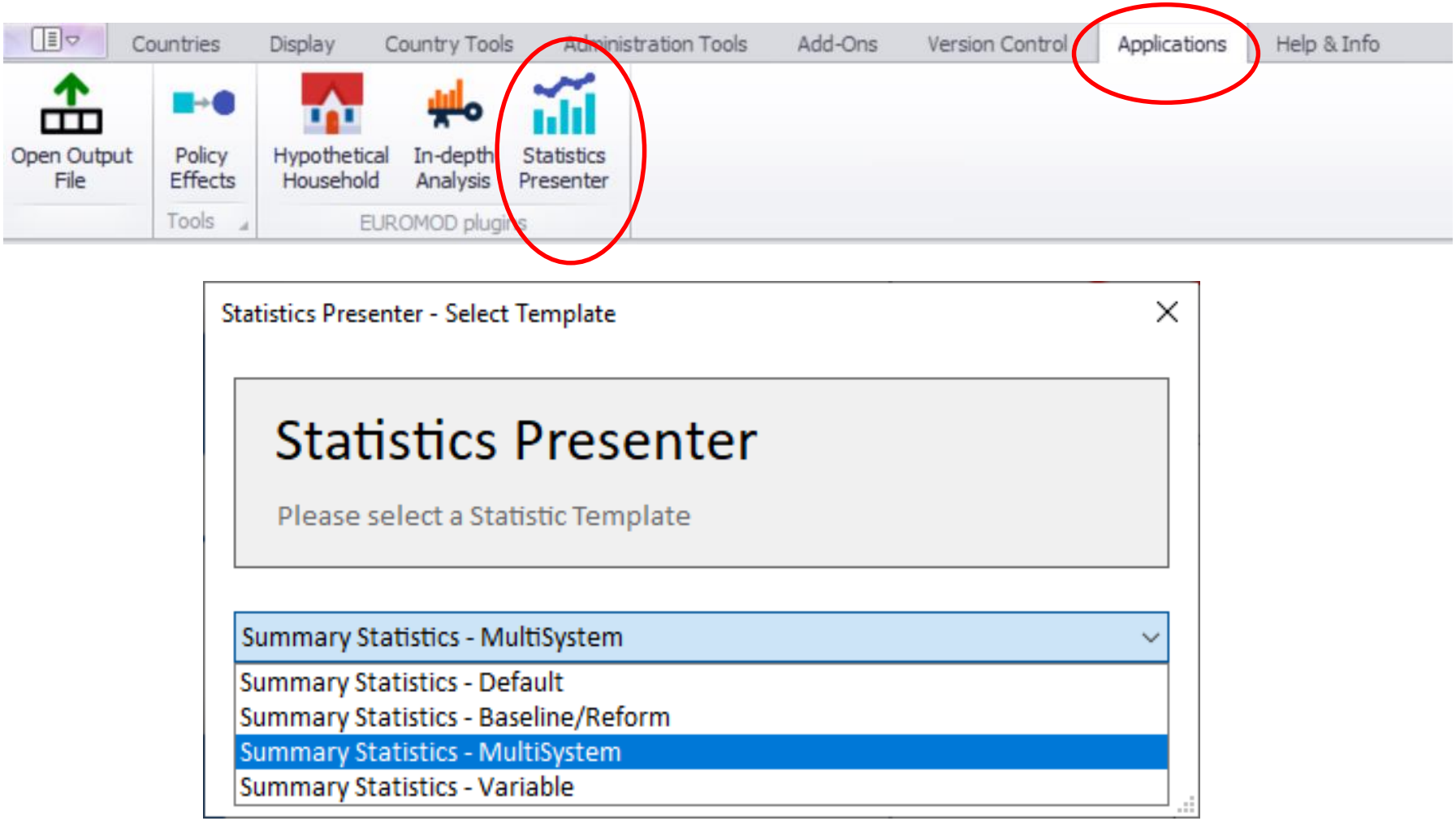
Step 3: Statistics Presenter Default (III)



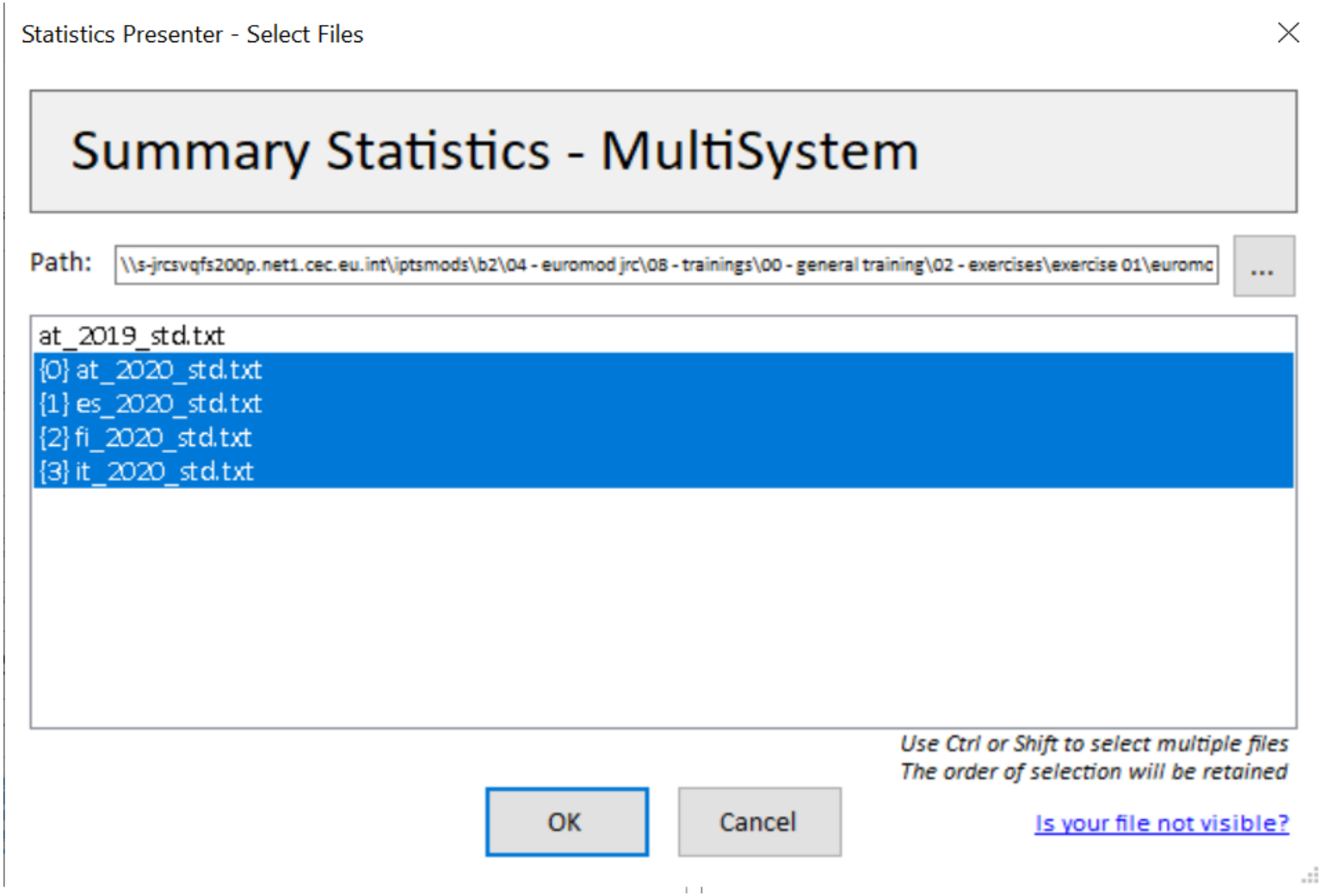
Step 3: Statistics Presenter Default (IV)



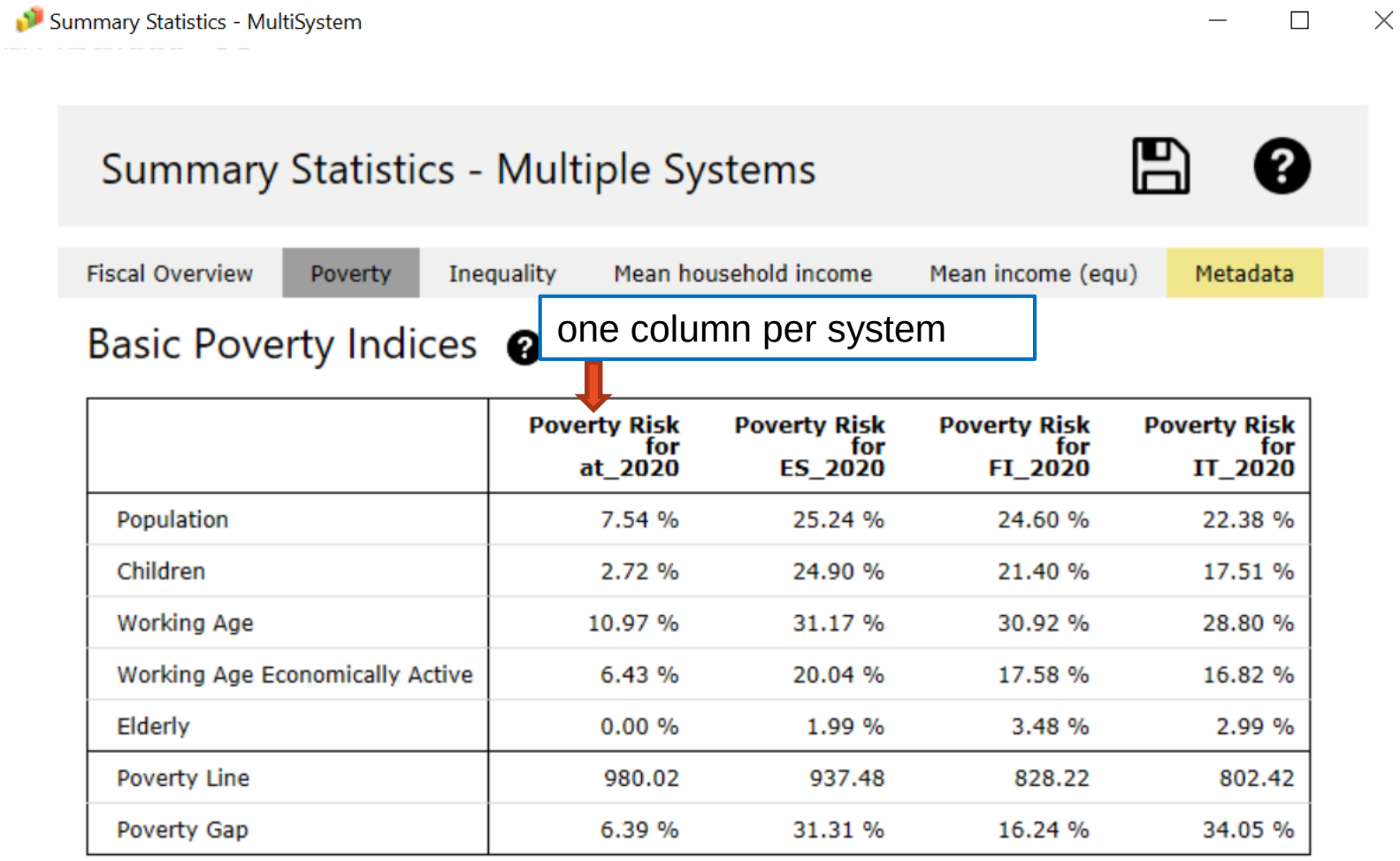
Step 4: Statistics Presenter MultiSystem (I)



Step 4: Statistics Presenter MultiSystem (II)



Step 4: Statistics Presenter MultiSystem (III)



	Poverty Risk for at_2020	Poverty Risk for ES_2020	Poverty Risk for FI_2020	Poverty Risk for IT_2020
Population	7.54 %	25.24 %	24.60 %	22.38 %
Children	2.72 %	24.90 %	21.40 %	17.51 %
Working Age	10.97 %	31.17 %	30.92 %	28.80 %
Working Age Economically Active	6.43 %	20.04 %	17.58 %	16.82 %
Elderly	0.00 %	1.99 %	3.48 %	2.99 %
Poverty Line	980.02	937.48	828.22	802.42
Poverty Gap	6.39 %	31.31 %	16.24 %	34.05 %

Summary of Exercise 1

- You learned how to:
  - Open the model
  - Run a single system or several systems at the same time
  - Analyse the output microdata with the Statistics Presenter, using the:
    - Default option
    - MultiSystem option

Q&A

Questions

TRAINING
EUROMOD

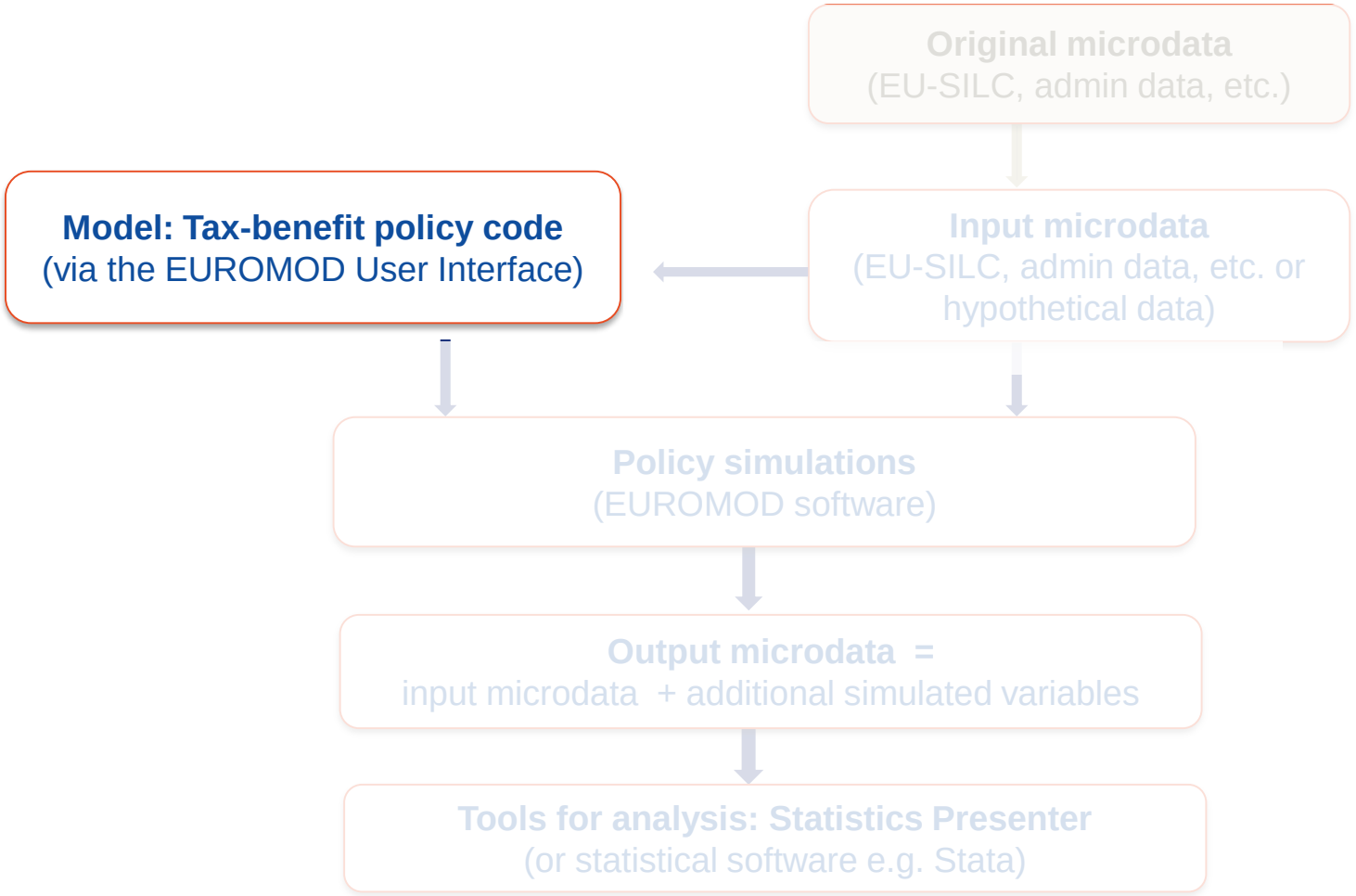
Session 2

EUROMOD language: policies, functions and parameters

European Commission

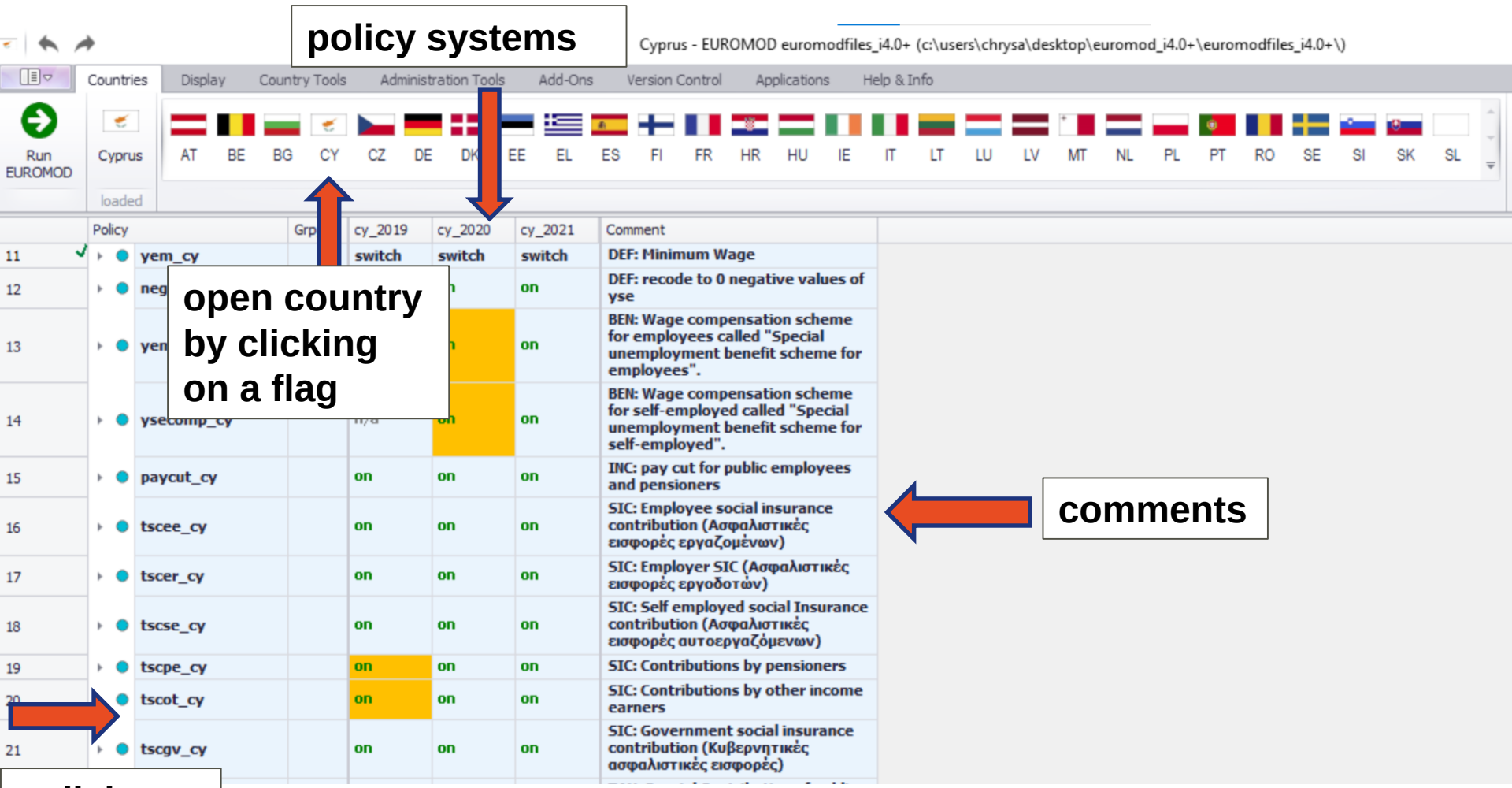
In this session, you will learn about

- The building blocks of EUROMOD tax-benefit microsimulation language:
  I. Policies
  II. Functions
  III. Parameters



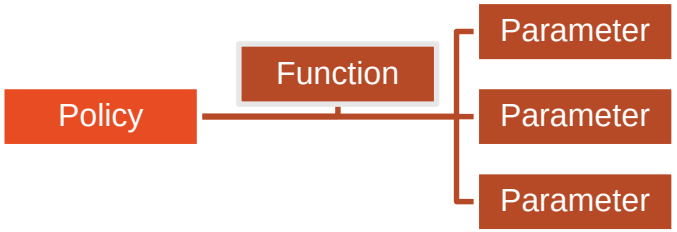
EUROMOD TAX-BENEFIT MICROSIMULATION LANGUAGE

Working environment



European Commission

Policies



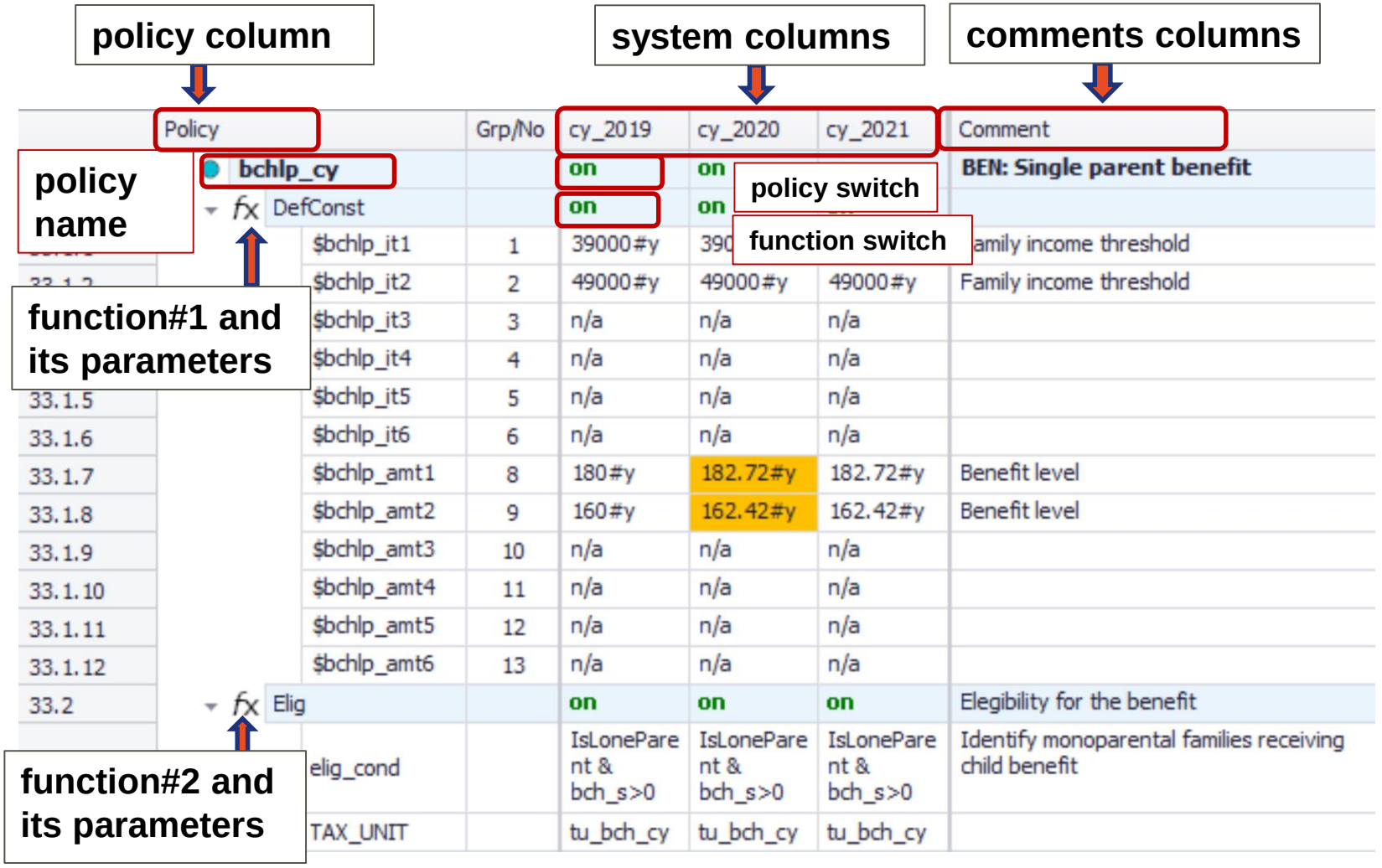
- a **block of functions** completes a policy simulation
- **policy names** end (usually) with the country acronym
  - policies can have any name – but good to follow naming conventions
- **comments** (in the comments column) explain what policies intend to simulate
- policies are run **in the order they appear**
  - the order of policies is called the **spine**

Structure of a policy

- **Policy** column includes policy name, function and parameter names
  - order of policies and functions reflects order of calculations
- **System** columns include the values of the tax-benefit parameters by policy year
- **Comments** column includes policy description

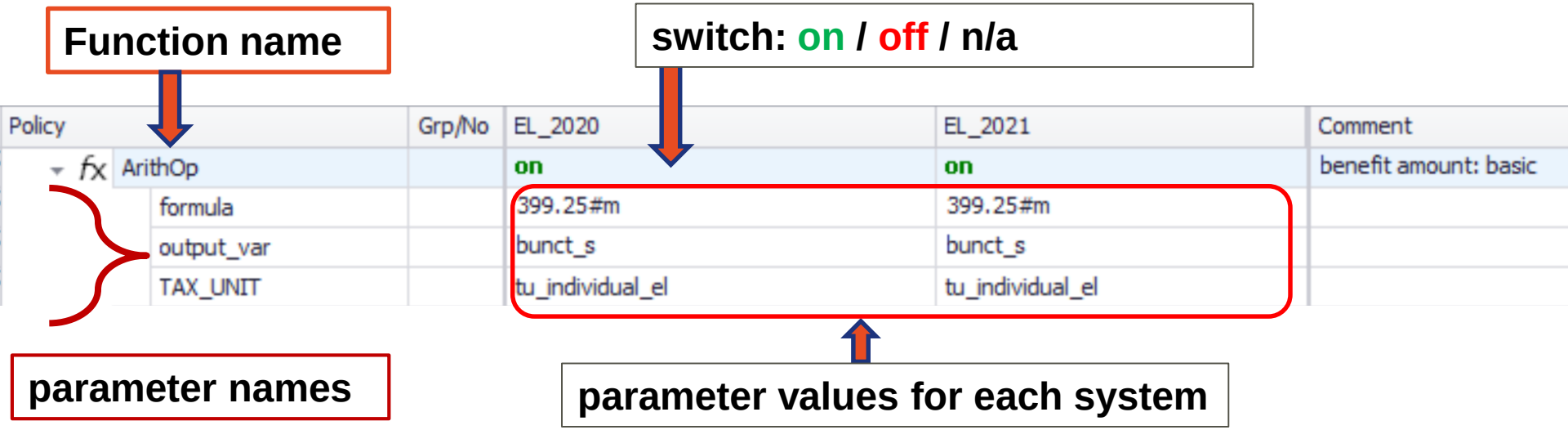
Policy column	**System columns**			**Comments column**
	System#1	**System#2**	**System#3**	
Policy#1 name	*policy switch*	*policy switch*	*policy switch*	Policy description
Function#1	*function switch*	*function switch*	*function switch*	Function description
Parameter#1	value	value	value	paremter description
Parameter#2	value	value	value	
Function#2	*function switch*	*function switch*	*function switch*	
Parameter#1	value	value	value	
Parameter#2	value	value	value	
Parameter#3	value	value	value	

Example of a policy

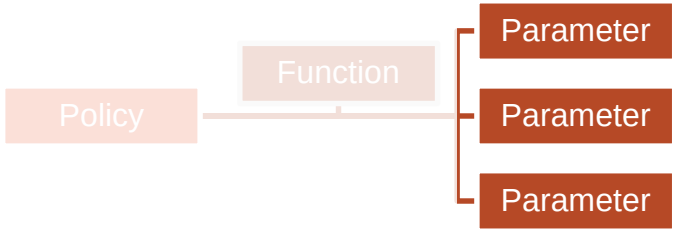


	Policy		Grp/No	cy_2019	cy_2020	cy_2021	Comment
	bchlp_cy			on	on		BEN: Single parent benefit
	fx DefConst			on	on		
		$bchlp_it1	1	39000#y	390		amily income threshold
		$bchlp_it2	2	49000#y	49000#y	49000#y	Family income threshold
		$bchlp_it3	3	n/a	n/a	n/a	
		$bchlp_it4	4	n/a	n/a	n/a	
33.1.5		$bchlp_it5	5	n/a	n/a	n/a	
33.1.6		$bchlp_it6	6	n/a	n/a	n/a	
33.1.7		$bchlp_amt1	8	180#y	182.72#y	182.72#y	Benefit level
33.1.8		$bchlp_amt2	9	160#y	162.42#y	162.42#y	Benefit level
33.1.9		$bchlp_amt3	10	n/a	n/a	n/a	
33.1.10		$bchlp_amt4	11	n/a	n/a	n/a	
33.1.11		$bchlp_amt5	12	n/a	n/a	n/a	
33.1.12		$bchlp_amt6	13	n/a	n/a	n/a	
33.2	fx Elig			on	on	on	Elegibility for the benefit
		elig_cond		IsLoneParent & bch_s>0	IsLoneParent & bch_s>0	IsLoneParent & bch_s>0	Identify monoparental families receiving child benefit
		TAX_UNIT		tu_bch_cy	tu_bch_cy	tu_bch_cy	

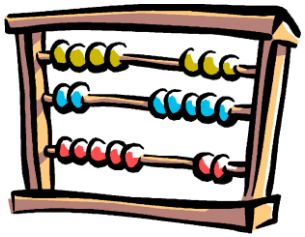
Structure of a function



Parameters



- Can be:
  - common to several functions or specific to one function
  - compulsory (i.e. error generated if not used) or optional
    - which parameters are compulsory/optional depends on the function
- Order of parameters in a function is not important
  - but order of functions in a policy is!
- Manipulated via context menu (right-click)
  - only relevant parameters for the given function are shown
- Drag & drop can be used

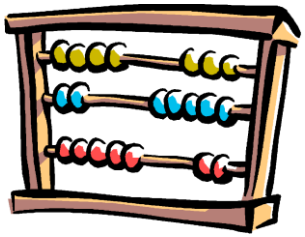


Exercise 2

Child benefit reform

Implement a Child benefit reform in Finland. Increase the benefit amount for the first child from €94.88 to €100 per month.

We will do the exercise together!

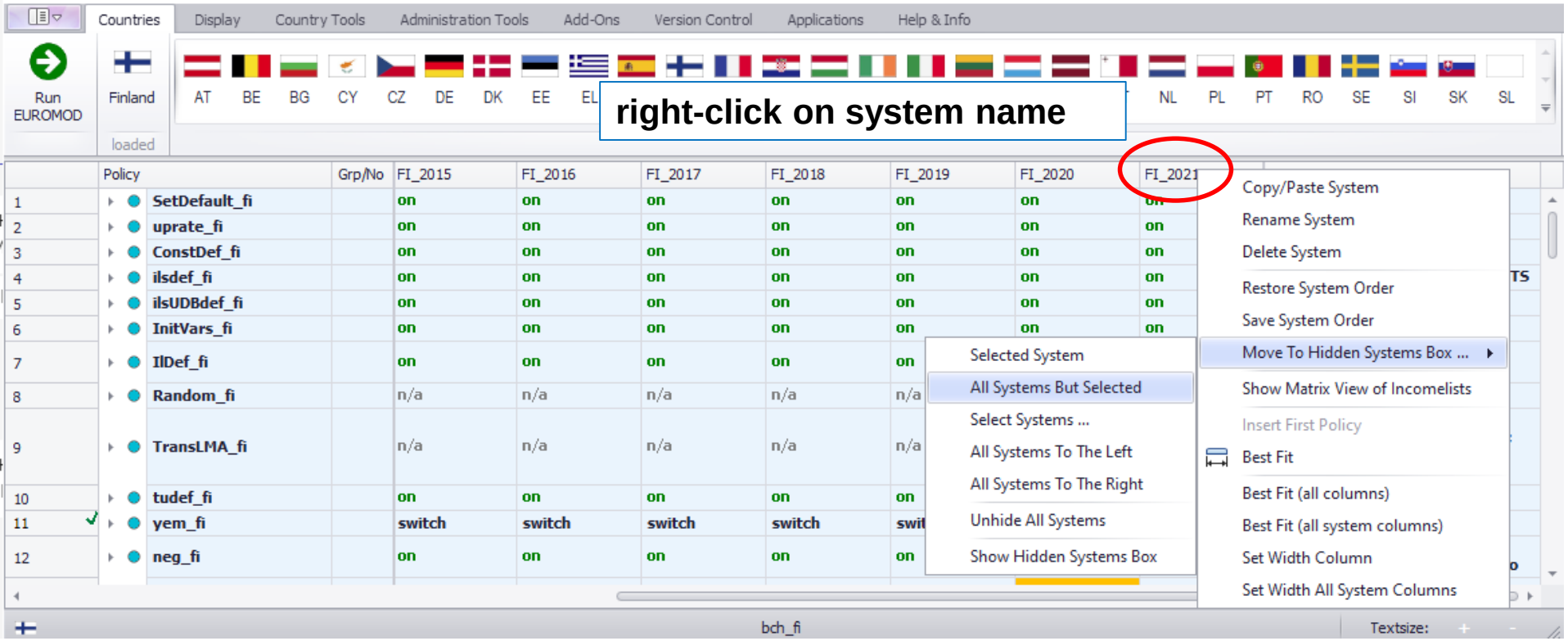


Exercise 2

Steps:

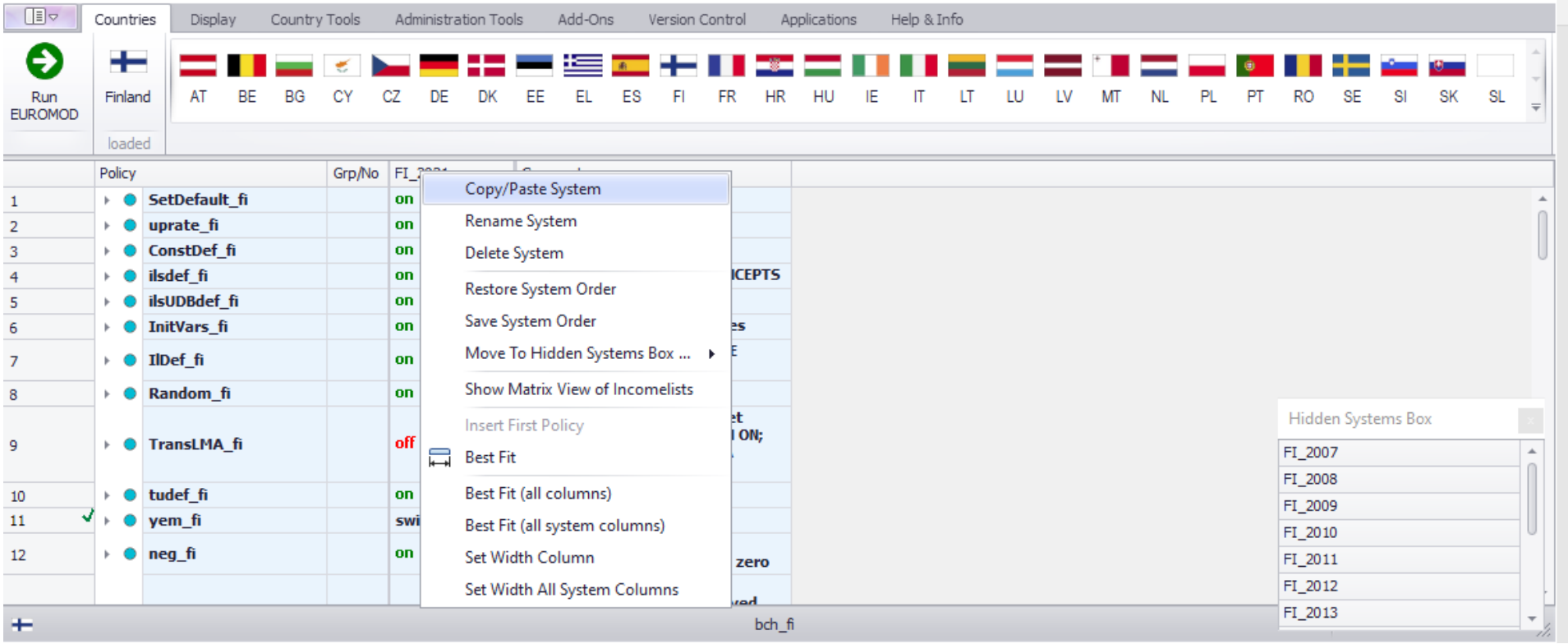
- You will make a copy of the FI 2021 system and call it FI_2021reform
- Then, you will modify the reform system FI_2021reform as follows:
  - Go to the Child Benefit policy (bch_fi) and open the functions DefConst and BenCalc
  - Change the parameter value of $bch1 from 94.88#m to 100#m
  - Save your changes
- Run the model and use the *Statistics Presenter* – Baseline/Reform option to analyse the distributional effect of the reform

Hiding systems



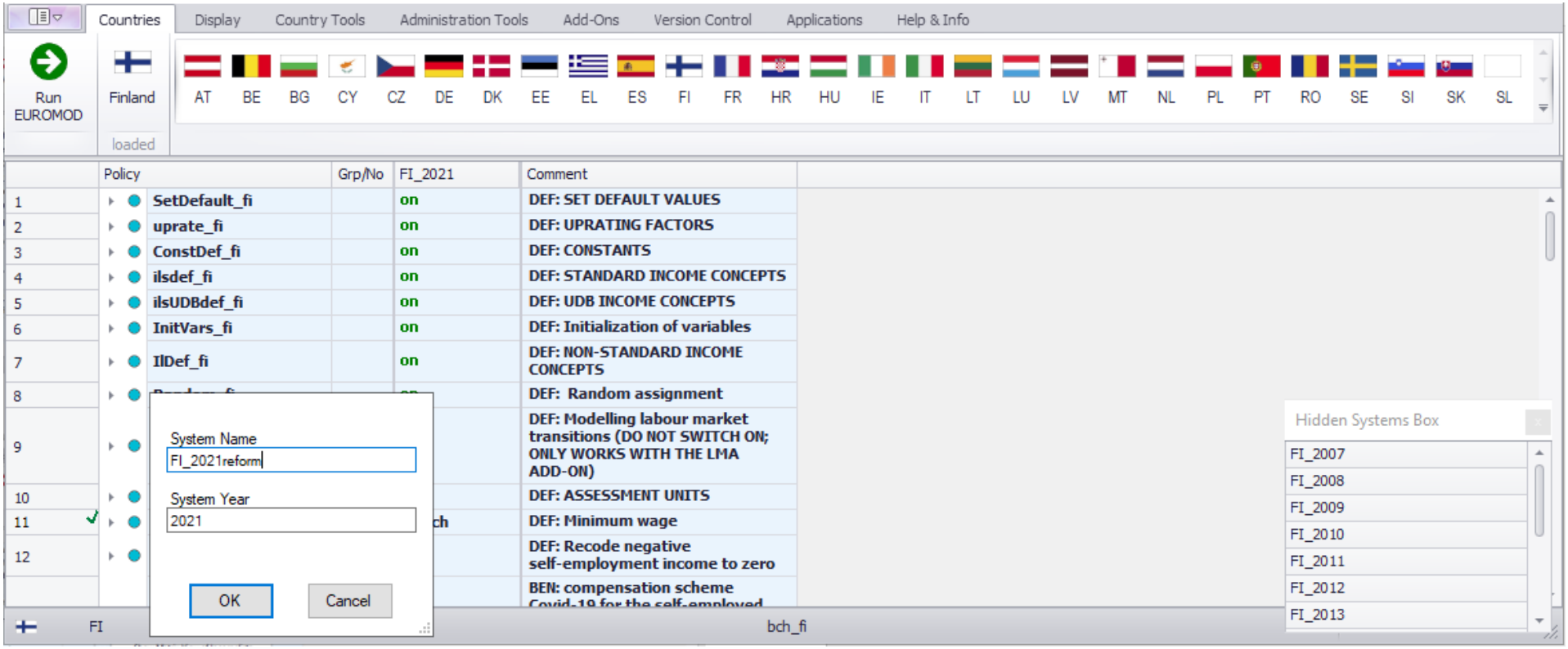
European Commission

Adding a reform system



European Commission

Adding a reform system (2)



European Commission

Implementing a reform (1)

Right click to expand policy

	Policy	Grp/No	FI_2021	FI_2021reform	Comment
41	**bch_fi**		on	on	BEN: Child benefit
41.1	DefConst		on	on	Constants for Child Care Benefits
41.2	Elig		on	on	Child benefit: eligibility
41.3	BenCalc		on	on	Child benefit: amount
41.4	Elig		on	on	Eligibility for Lone parent supplement
41.5	ArithOp		on	on	Child benefit: single parent supplement

Implementing a reform (2)

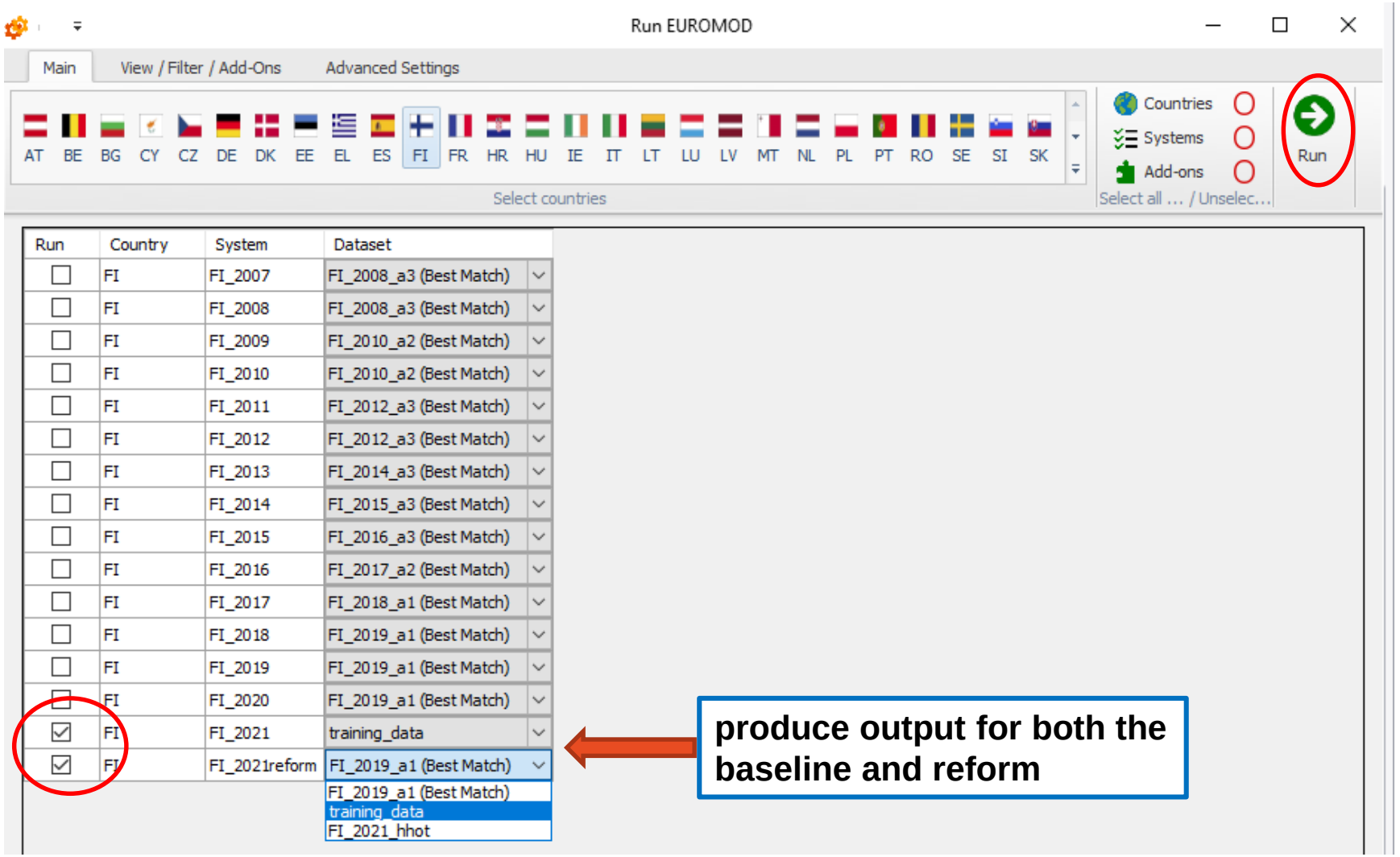
	Policy	Grp/No	FI_2021	FI_2021reform	Comment
41	**bch_fi**		on	on	BEN: Child benefit
41.1	DefConst		on	on	Constants for Child Care Benefits
41.1.1	$bch1		94.88#m	94.88#m	Child benefit for the 1st child
41.1.2	$bch2		104.84#m	104.84#m	Child benefit for the 2nd child
41.1.3	$bch3		133.79#m	133.79#m	Child benefit for the 3rd child
41.1.4	$bch4		163.24#m	163.24#m	Child benefit for the 4th child
41.1.5	$bch5		182.69#m	182.69#m	Child benefit for the 5th child & subsequent
41.1.6	$bchsup		63.30#m	63.30#m	Child benefit supplement for a lone parent
41.2	Elig		on	on	Child benefit: eligibility
41.3	BenCalc		on	on	Child benefit: amount
41.3.1	who_must_be_elig		all	all	
41.3.2	comp_cond	1	IsNtoMchild#1=1	IsNtoMchild#1=1	
41.3.3	#_N	1	1	1	
41.3.4	#_M	1	1	1	
41.3.5	comp_cond	2	IsNtoMchild#2=1	IsNtoMchild#2=1	
41.3.6	#_N	2	2	2	
41.3.7	#_M	2	2	2	
41.3.8	comp_cond	3	IsNtoMchild#3=1	IsNtoMchild#3=1	
41.3.9	#_N	3	3	3	
41.3.10	#_M	3	3	3	
41.3.11	comp_cond	4	IsNtoMchild#4=1	IsNtoMchild#4=1	
41.3.12	#_N	4	4	4	
41.3.13	#_M	4	4	4	
41.3.14	comp_cond	5	IsNtoMchild#5=1	IsNtoMchild#5=1	
41.3.15	#_N	5	5	5	
41.3.16	#_M	5	99	99	
41.3.17	comp_perElig	1	$bch1	$bch1	Amount for the 1st child
41.3.18	comp_perElig	2	$bch2	$bch2	Amount for the 2nd child
41.3.19	comp_perElig	3	$bch3	$bch3	Amount for the 3rd child
41.3.20	comp_perElig	4	$bch4	$bch4	Amount for the 4th child
41.3.21	comp_perElig	5	$bch5	$bch5	Amount for the 5th & subsequent children

Implementing a reform (3)

Type the new amount in the reform system

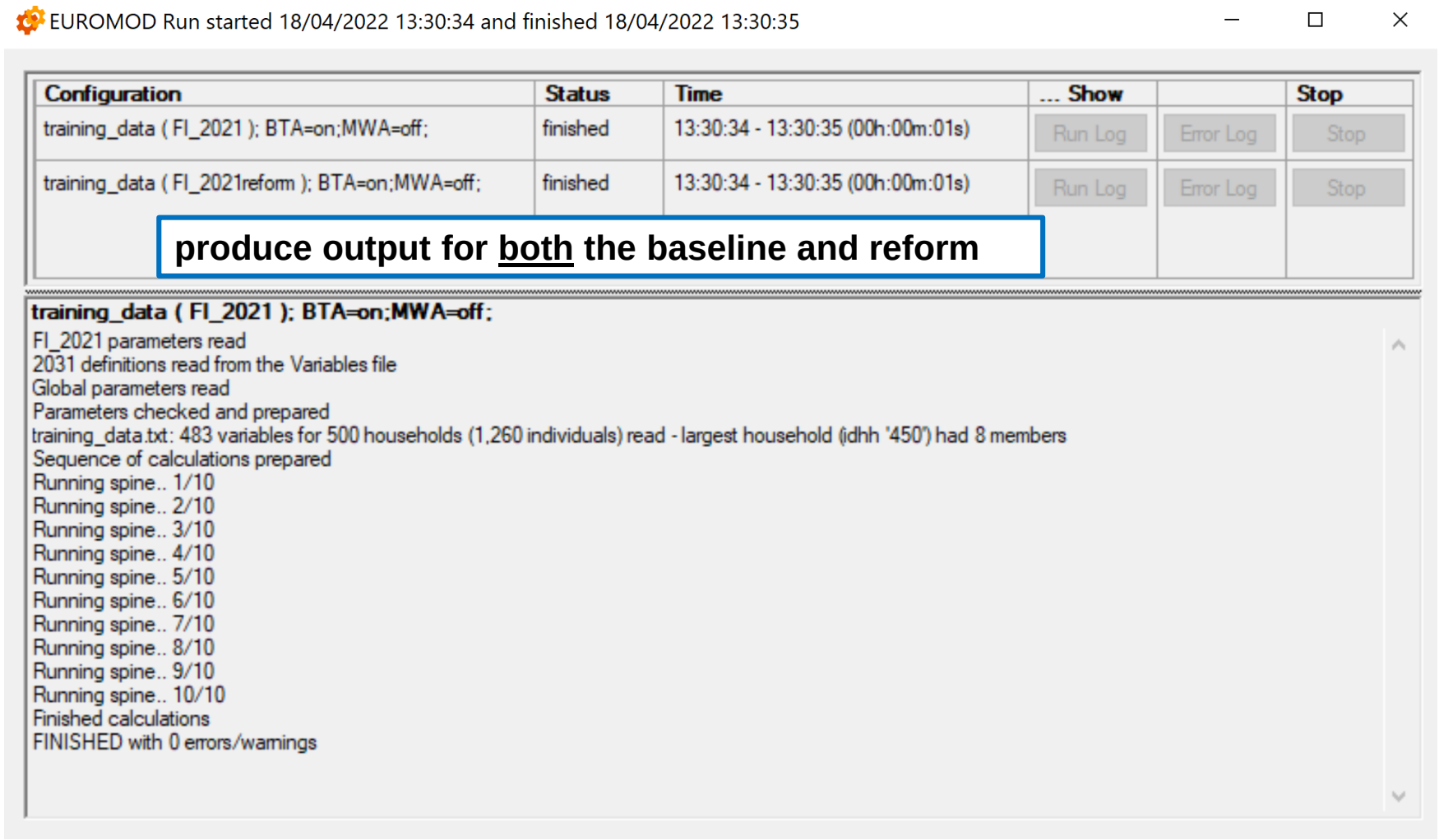
	Policy	Grp/No	FI_2021	FI_2021reform	Comment
41	**bch_fi**		on	on	BEN: Child benefit
41.1	DefConst		on	on	Constants for Child Care Benefits
41.1.1	$bch1		94.88#m	100#m	Child benefit for the 1st child
41.1.2	$bch2		104.84#m	104.84#m	Child benefit for the 2nd child
41.1.3	$bch3		133.79#m	133.79#m	Child benefit for the 3rd child
41.1.4	$bch4		163.24#m	163.24#m	Child benefit for the 4th child
41.1.5	$bch5		182.69#m	182.69#m	Child benefit for the 5th child & subsequent
41.1.6	$bchsup		63.30#m	63.30#m	Child benefit supplement for a lone parent

Producing output microdata (1)

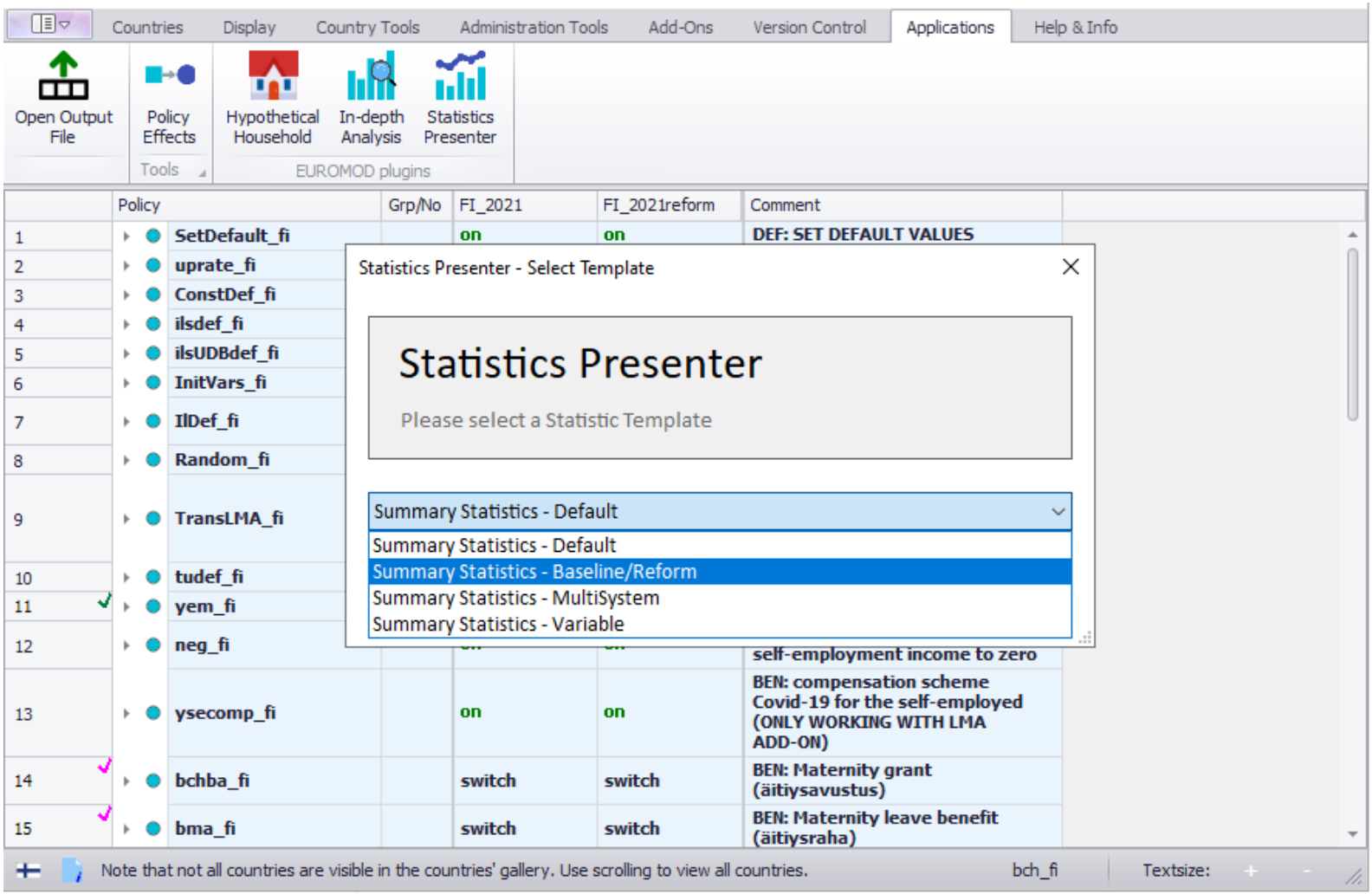


European Commission

Producing output microdata (2)

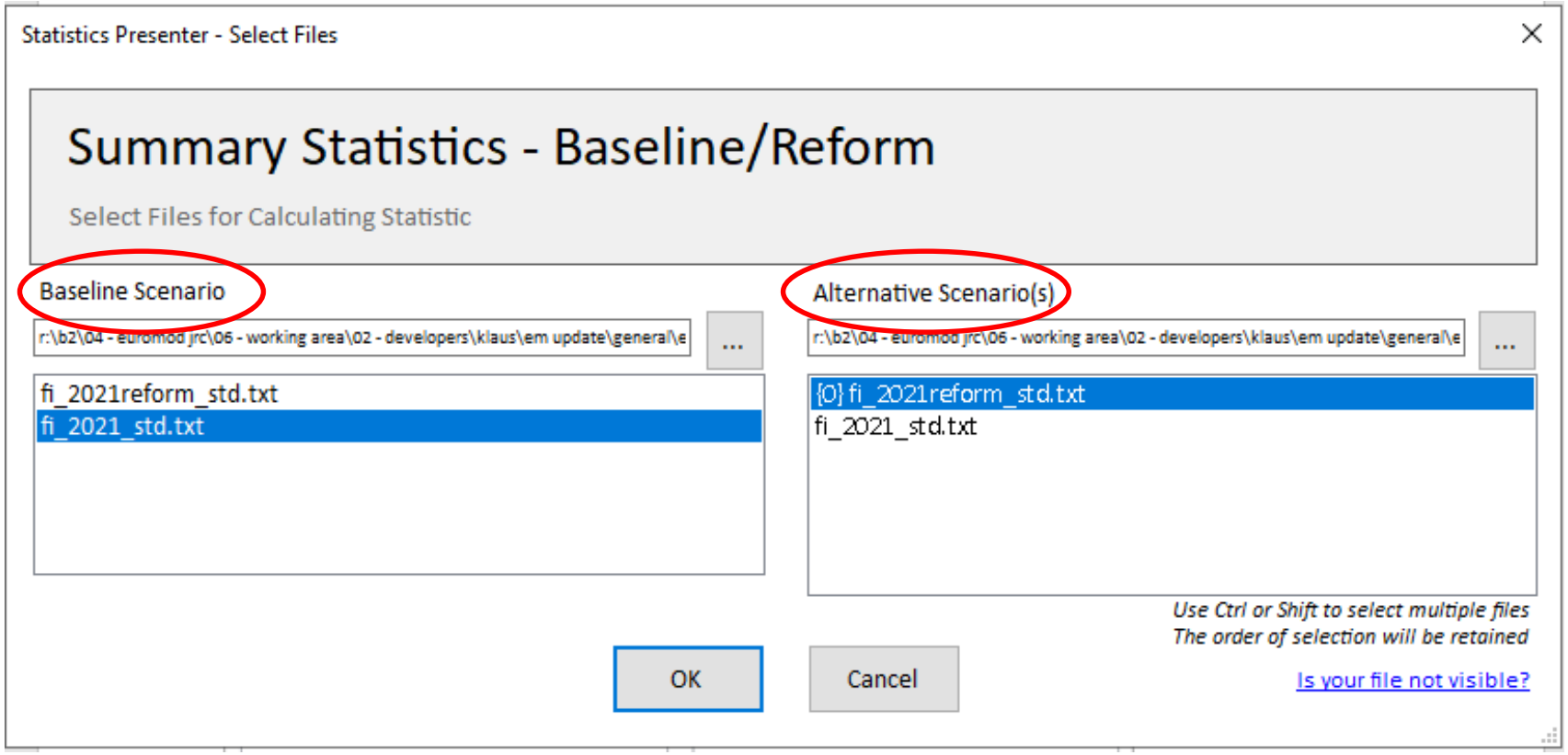


Opening the Statistics Presenter



European Commission

Statistics Presenter: Baseline/Reform (1)



Statistics Presenter: Baseline/Reform (2)

Table 2b: Mean household income by decile groups

	FI_2021 (base)	*FI_2021reform*	*Difference to base*
Decile 1	*729.68*	*730.04*	*0.36*
Decile 2	*1,485.36*	*1,486.61*	*1.25*
Decile 3	*1,745.62*	*1,747.42*	*1.80*
Decile 4	*1,835.96*	*1,837.32*	*1.36*
Decile 5	*2,060.86*	*2,062.53*	*1.67*
Decile 6	*2,526.84*	*2,528.73*	*1.89*
Decile 7	*2,544.38*	*2,546.12*	*1.74*
Decile 8	*3,123.74*	*3,124.63*	*0.89*
Decile 9	*3,766.25*	*3,767.50*	*1.25*
Decile 10	*4,862.14*	*4,864.08*	*1.95*
All	***2,437.55***	***2,438.91***	***1.37***
Poor	*1,095.62*	*1,096.42*	*0.79*

Summary: Exercise 2

- You learned:
  - how to add a reform system
  - about basic options such as (un)hiding systems, expanding a policy and conditional formatting
  - how to analyse the baseline and reform output microdata with the Statistics Presenter, using:
    - Baseline/Reform template

Q&A

Questions

	Policy	Grp/No	FI_2021	Comment
18	pmmtu_fi		on	BEN: Guarantee pension
19	tscee_fi		on	TAX: Employee social contributions
20	tscer_fi		on	TAX: Employer social contributions
21	tscse_fi		on	TAX: Self-employed contributions for entrepreneurs
22	tscfr_fi		on	TAX: Self-employed contributions for farmers
23	bed00_fi		on	BEN: Study Grant
24	bhosd_fi		n/a	BEN: Student housing supplement
25	bunnc_fi		on	BEN: Basic unemployment allowance (PART-SIMULATED)
26	bunct_fi		on	BEN: Earnings-related unemployment allowance (PART-SIMULATED)
27	bunmt_fi		on	BEN: Labour market subsidy (PART-SIMULATED)
28	bcc_fi		on	BEN: Child home care allowance
29	tiniy_fi		on	TAX: Capital income tax
30	tinna_fi		on	TAX: National income tax
31	tinmu_fi		on	TAX: Local income tax
32	tintcly_fi		on	TAX: Low income tax credit (LITC)
33	tintcch_fi		n/a	TAX: Child tax credit (CHTC)
34	tindc_fi		on	TAX: Income tax deficit credit
35	tindcsp_fi		on	TAX: Income tax special deficit credit

THE POLICY SPINE

The policy spine

- The policy spine:
  - The order of policies is called the **spine**
  - We will take a quick tour through the policies in the Estonian model
    - the models for all EU countries follow the same structure of the policy spine
  - See the EUROMOD Country Reports for more details

Policies in the spine

DEFINITIONS

- define certain parameters or concepts; or do some pre-policy calculations
- e.g. define assessment units; index (uprate) earnings if input microdata year lags behind policy year

TAX AND BENEFIT CALCULATIONS

- (almost) every tax and benefit implemented in a separate policy; policy name according to our naming convention
- e.g. bch00_ee (Child Allowance → b: benefit, ch: child, 00: main)

OUTPUT

- defines what variables to be included in the output microdata and assessment unit for the results
- e.g. include both simulated and non-simulated variables; output individual-level results

Definitions

	Policy	Grp/No	EE_2018	Comment
1	SetDefault_ee		on	DEF: DEFAULT VALUES FOR VARIABLES
2	Uprate_ee		on	DEF: UPRATING FACTORS
3	ConstDef_ee		on	DEF: CONSTANTS
4	ILSDef_ee		on	DEF: STANDARD INCOME CONCEPTS
5	ILSUDBDef_ee		on	DEF: STANDARD INCOME CONCEPTS (UDB related)
6	ILDef_ee		on	DEF: SPECIFIC INCOME CONCEPTS
7	random_ee		n/a	DEF: Random assignment for covid measure
8	random_pens_ee		n/a	DEF: Random assignment for pension contribution suspension
9	TransLMA_ee		n/a	DEF: Modelling labour market transitions (DO NOT SWITCH ON; ONLY WORKS WITH THE LMA ADD-ON)
10	PensCont_ee		n/a	DEF: Suspending pension payments: random allocation
11	TUDef_ee		on	DEF: ASSESSMENT UNITS
12	allocate_ee		on	DEF: reallocate benefits within household
13	BTA_ee		switch	SWITCH: Benefit Take-up Adjustments (ON=non take-up; OFF=full take up)
14	yem_ee		switch	DEF: minimum wage
15	neg_ee		on	DEF: recode negative incomes (e.g. self-employment income) to zero

Tax and benefit calculations

	Policy	Grp/No	EE_2018	Comment
16	yemcomp_ee		n/a	BEN: Wage compensation scheme COVID-19
17	tscer_ee		on	SIC: employer social insurance contributions
18	tscee_ee		on	SIC: employee social insurance contributions
19	tintaag_ee		on	TAX: allowance for self-employment income from agriculture (applicable to
20	tscse_ee		on	SIC: self-employed social insurance contributions
21	bmapr_ee		switch	BEN: Maternity benefit (sünnitushüvitis)
22	bmact_ee		switch	BEN: Parental benefit
23	bchba_ee		on	BEN: childbirth allowance
24	bch00_ee		on	BEN: child allowance
25	bchlg_ee		n/a	BEN: allowance for families with 3+ children (kolme- ja enamalapselise pere toetus)
26	bched_ee		n/a	BEN: school allowance
27	bcc00_ee		on	BEN: childcare allowance (lapsehooldustasu)
28	bcclg_ee		on	BEN: parental allowance for families with 7+ children / many children (seitsme- ja enamalapselise pere vanema toetus / lasterikka pere toetus)
29	bunct_ee		on	BEN: unemployment insurance benefit (töötuskindlustushüvitis)

	Policy	Grp/No	EE_2018	Comment
30	bunnc_ee		on	BEN: unemployment assistance benefit (töötu abiraha)
31	tscct_ee		on	SIC: credited social insurance contributions
32	tinwh_ee		on	TAX: withholding income tax (jooksvalt kinnipeetud
33	tin_ee		on	TAX: income tax (deklaratsioonijärgne ehk lõplik maksukohustus, final liability)
34	tinrf_ee		n/a	TAX: annual refund to low-paid employees (madala sissetulekuga töötava isiku iga-aastane tagasimakse)
35	bsa00_ee		on	BEN: subsistence benefit (incl. housing component) (toimetulekutoetus)
36	bsach_ee		n/a	BEN: needs-based family benefit (vajaduspõhine lapsetoetus)
37	bsape_ee		on	BEN: pensioner's living alone allowance (üksi elava pensionäri toetus)

Output

38	output_std_ee		on	DEF: STANDARD OUTPUT INDIVIDUAL LEVEL
39	output_std_hh_ee		off	DEF: STANDARD OUTPUT HOUSEHOLD LEVEL

Session 3

Policy functions Elig & ArithOp
EM handling errors

In this session, you will learn about

- Tax units
- Implementing eligibility conditions with function *Elig*
- Benefit/tax amount calculations with function *ArithOp*
- Function parameters: amounts, formulas and queries
- Handling errors in EUROMOD

fx Elig		on	Eligibility for Lone parent supplement
elig_cond		nDepChildrenInTU > 0 & IsLoneParent	
TAX_UNIT		tu_family17_fi	
fx ArithOp		on	Child benefit: single parent supplement
who_must_be_elig		one	
formula		nDepChildrenInTU * $bchsup	
output_add_var		bch_s	Child benefit (with lone parent supplement)
TAX_UNIT		tu_family17_fi	

THE POLICY FUNCTIONS
Elig & ArithOp

Calculating a benefit/tax

- What are the policy rules:
  - Who is entitled to the benefit/liable to the tax, i.e. the **assessment unit**:
    - e.g. the individual, family or household
  - What is the criteria, i.e. the **eligibility condition**
    - e.g. being a lone parent
  - What is the **benefit/tax amount**:
    - e.g. €50 per month or 20% of taxable income

- Write down the policy rule
  - Using EUROMOD **tax-benefit language**

Assessment units

Main assessment units (TAX_UNIT) defined in EUROMOD:

- HOUSEHOLD (e.g. tu_household_xx): all individuals of the household are in the same unit
- INDIVIDUAL (e.g. tu_individual_xx): each individual of the household forms its own unit
- FAMILY (e.g. tu_family_xx, tu_bu_xx): the nuclear family – e.g. the couple (cohabiting or married) or single adult plus any dependent children
  - The household may be split into several units of different size

Functions: *ArithOp*

- Arithmetical **calculator**
- Compulsory parameters: formula, output_var, TAX_UNIT
  - The parameter formula contains some calculations
  - The result is stored as output variable via parameter output_var
    - To add the answer to the result of a previous function, use output_add_var

Policy	System Name	Comment
ArithOp	on	Made-up example: child benefit
formula	20#w*nDepChildrenInTu	€20 per week for each dependent child in the assessment unit
output_var	bch_s	result saved in the variable bch_s (b: benefit, ch: child, _s: simulated)
TAX_UNIT	tu_bu_ee	assessment unit used for the calculations

- Now let's have a look at some key parameter values we can use for ***formula***
  - **amounts, formulas, queries**

Parameters: Amounts (#m)

- Monetary (numbers; use . for decimal) followed by their period:
  - *#m* for monthly (no conversion)
  - *#y* for yearly
  - *#q* for quarterly
  - *#w* for weekly
  - *#d* for daily
  - *#l* for labour day
  - *#s* for six day labour week
  - *#c* for capital (no conversion)
- Default is #m (monthly)

Policy	System Name	Comment
ArithOp	**on**	Made-up example: child benefit
formula	20#w*nDepChildrenInTu	€20 per week for each dependent child in the assessment unit
output_var	bch_s	result saved in the variable bch_s (b: benefit, ch: child, _s: simulated)
TAX_UNIT	tu_bu_ee	assessment unit used for the calculations

Parameters: Formulas

- Operations: *, /, +, -, ^, min(), max(), abs(), (), !(), %

Policy	System Name	Comment
ArithOp	on	Made-up example: child benefit
formula	(20#w - xed/2)*nDepChildrenInTu - bed_s	€20 per week minus half of educational expenses (xed), for each dependent child in the assessment unit; and deduct any education benefits (bed_s)
output_var	bch_s	result saved in the variable bch_s (b: benefit, ch: child, _s: simulated)
TAX_UNIT	tu_bu_ee	assessment unit used for the calculations

Parameters: Queries (1)

- Frequently used ready-made calculations
- The result of a query is:
  - yes (=1) or no (=0) (e.g. IsDepChild)
  - some (monetary or non monetary) value (e.g. nDepChildrenInTu)
- Well-documented in the Help file of the User Interface

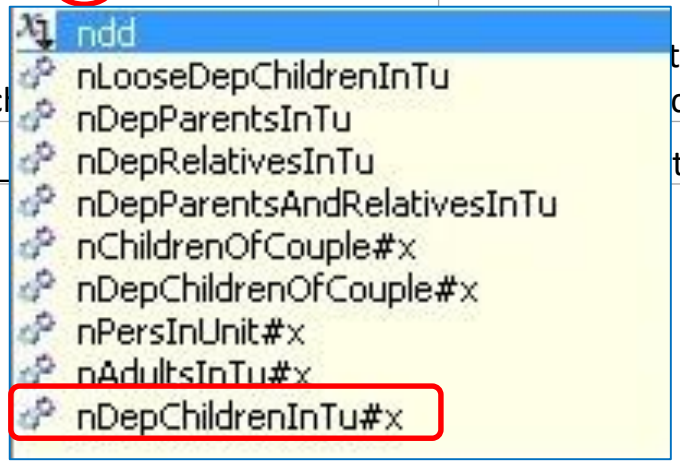
Query	Description	Parameters
IsDepChild	Returns 1 if a person is a 'dependent child', i.e. fulfils the DepChildCond of the assessment unit specification, 0 otherwise	
nDepChildrenInTu#x	Returns the number of dependent children in the assessment unit who fulfil dag >= parameter #_AgeMin and dag <= parameter #_AgeMax	#_AgeMin, #_AgeMax (optional)

Parameters: Queries (2)

- No need to learn the query names by heart

→ use “IntelliSense” feature to find and select the desired query

ArithOp	**on**	Made-up example: child benefit
formula	20#w*n	€20 per week for each dependent child in the assessment unit
output_var	bch	the variable bch_s child, _s: simulated)
TAX_UNIT	tu_	t used for the calculations



Functions: *Elig* (1)

- Used to implement **conditions**
- Compulsory parameters: elig_cond, TAX_UNIT
  - Condition is defined in parameter elig_cond
    - Creates a variable equal to 0 or 1 (by default sel_s)
    - Subsequent functions use this information via parameter Who_Must_Be_Elig

Functions: *Elig* (2)

Policy	System Name	Comment
Elig	**on**	Made-up example: condition to pay employee social insurance contributions (SIC)
elig_cond	!IsCivilServant	not a civil servant (!IsCivilServant)
TAX_UNIT	tu_individual_ee	assessment unit used for the calculations
ArithOp	**on**	Made-up example: pension contributions
formula	yem*0.08	8% of earnings (yem) for old-age pension SIC
output_var	tsceepi_s	result saved in the variable tsceepi_s (t: tax, sc: social contributions, ee: employee, pi: pension insurance, _s: simulated)
TAX_UNIT	tu_individual_ee	assessment unit used for the calculations

calculations in ArithOp are carried out for everyone with earnings, including civil servants

Functions: *Elig* (3)

Policy	System Name	Comment
Elig	**on**	Made-up example: condition to pay employee social insurance contributions (SIC)
elig_cond	!IsCivilServant	not a civil servant (!IsCivilServant)
TAX_UNIT	tu_individual_ee	assessment unit used for the calculations
ArithOp	**On**	Made-up example: pension contributions
Who_Must_Be_Elig	One	calculations carried out if at least one member of assessment unit fulfills condition from last Elig function
formula	yem*0.08	8% of earnings (yem) for old-age pension SIC
output_var	tsceepi_s	result saved in the variable tsceepi_s (t: tax, sc: social contributions, ee: employee, pi: pension insurance, _s: simulated)
TAX_UNIT	tu_individual_ee	assessment unit used for the calculations
ArithOp	**on**	Made-up example: unemployment contributions
Who_Must_Be_Elig	one	calculations carried out if at least one member of assessment unit fulfills condition from last Elig function
formula	yem*0.02	2% of earnings for unemployment SIC
output_var	tsceeui_s	result saved in the variable tsceeui_s (t: tax, sc: social contributions, ee: employee, ui: unemployment insurance, _s: simulated)
TAX_UNIT	tu_individual_ee	assessment unit used for the calculations

Parameters: Eligibility
(elig_cond; Who_Must_Be_Elig)

- Who_Must_Be_Elig: calculations of the function are carried out if…
  - **one**: one member of the assessment unit is eligible
  - **one_adult**: one adult member of the assessment unit is eligible
  - **all**: all members of the assessment unit are eligible
  - **all_adults**: all adult members of the assessment unit are eligible
  - **nobody**: calculations are carried out for each assessment unit (default)

- By default eligibility result is saved in the variable sel_s (can use other variable in parameter elig_var)
  - 0: person is not eligible
  - 1: person is eligible

Parameters: Eligibility

(elig_cond; Who_Must_Be_Elig)

Policy	System Name	Comment
Elig	on	Made-up example: eligibility condition for social assistance
elig_cond	dag>=80	individual should be of age (dag) of 80+ years
TAX_UNIT	tu_individual_ee	assessment unit is the INDIVIDUAL
ArithOp	On	Made-up example: social assistance amount
Who_Must_Be_Elig	?	who in the assessment unit must fulfil eligibility condition
formula	100#m	benefit amount is €100
output_var	bsa_s	result saved in the variable bsa_s (b: benefit, sa: social assistance, _s: simulated)
TAX_UNIT	tu_household_ee	assessment unit is the HOUSEHOLD

idhh	idperson	dag	sel_s	Will the calculations in ArithOp be carried out if: **who_must_be_elig=** one	one_adult	all	all_adults	nobody
1	11	80	1					
1	12	60	0	yes	yes	no	no	yes
1	13	40	0					
2	21	80	1					
2	22	6	0	yes	yes	no	yes	yes
3	31	80	1	yes	yes	yes	yes	yes
4	41	40	0					
4	42	40	0	no	no	no	no	yes

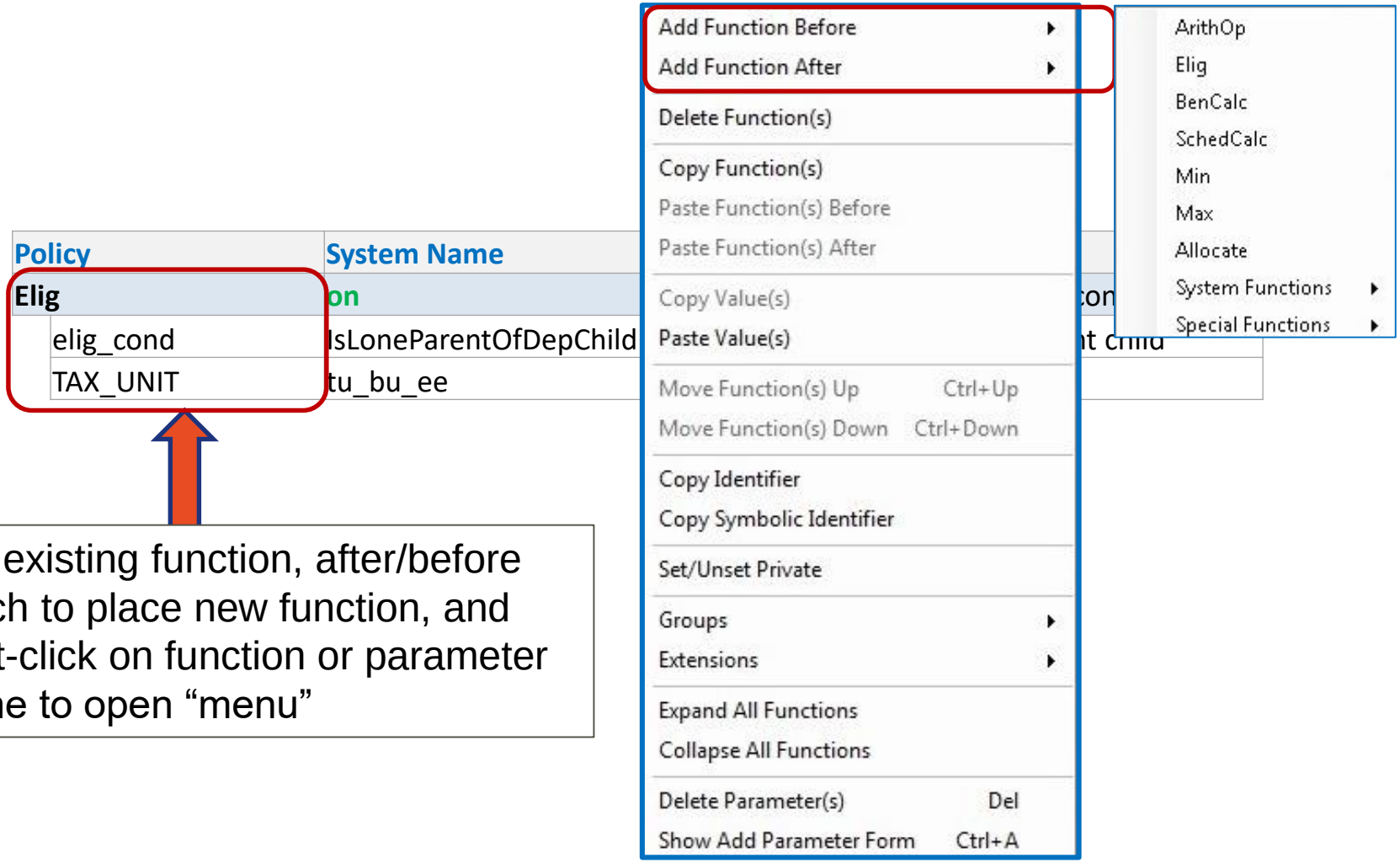
Parameters: Output

(output_var; output_add_var)

- Either output_var or output_add_var must be indicated!
- Exception: func_Elig → sel_s

Policy	System Name	Comment
ArithOp	**on**	Made-up example: child benefit
formula	20#w*nDepChildrenInTu	€20 per week for each dependent child in the assessment unit
output_var	bch_s	result saved in the variable bch_s (b: benefit, ch: child, _s: simulated)
TAX_UNIT	tu_bu_ee	
Elig	**on**	Made-up example: eligibility condition
elig_cond	IsLoneParentOfDepChild	if a lone parent of a dependent child
TAX_UNIT	tu_bu_ee	
ArithOp	**on**	Made-up example: child benefit supplement for lone parents
Who_Must_Be_Elig	one	
formula	15#w	€15 per week for the assessment unit
output_add_var	bch_s	add answer to the result of the previous function, saved in the variable bch_s
TAX_UNIT	tu_bu_ee	

User Interface: Adding a function

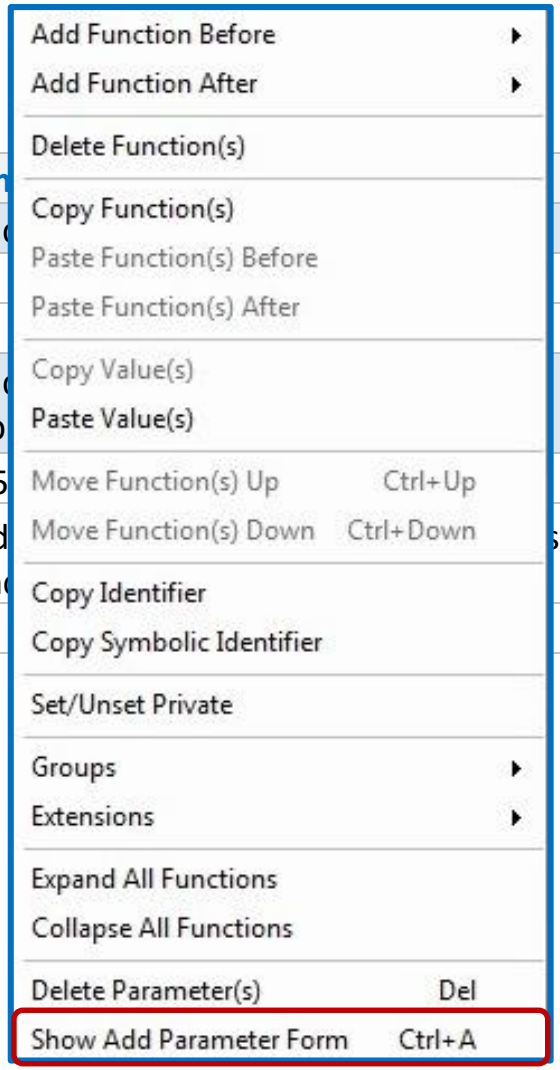


find existing function, after/before which to place new function, and right-click on function or parameter name to open “menu”

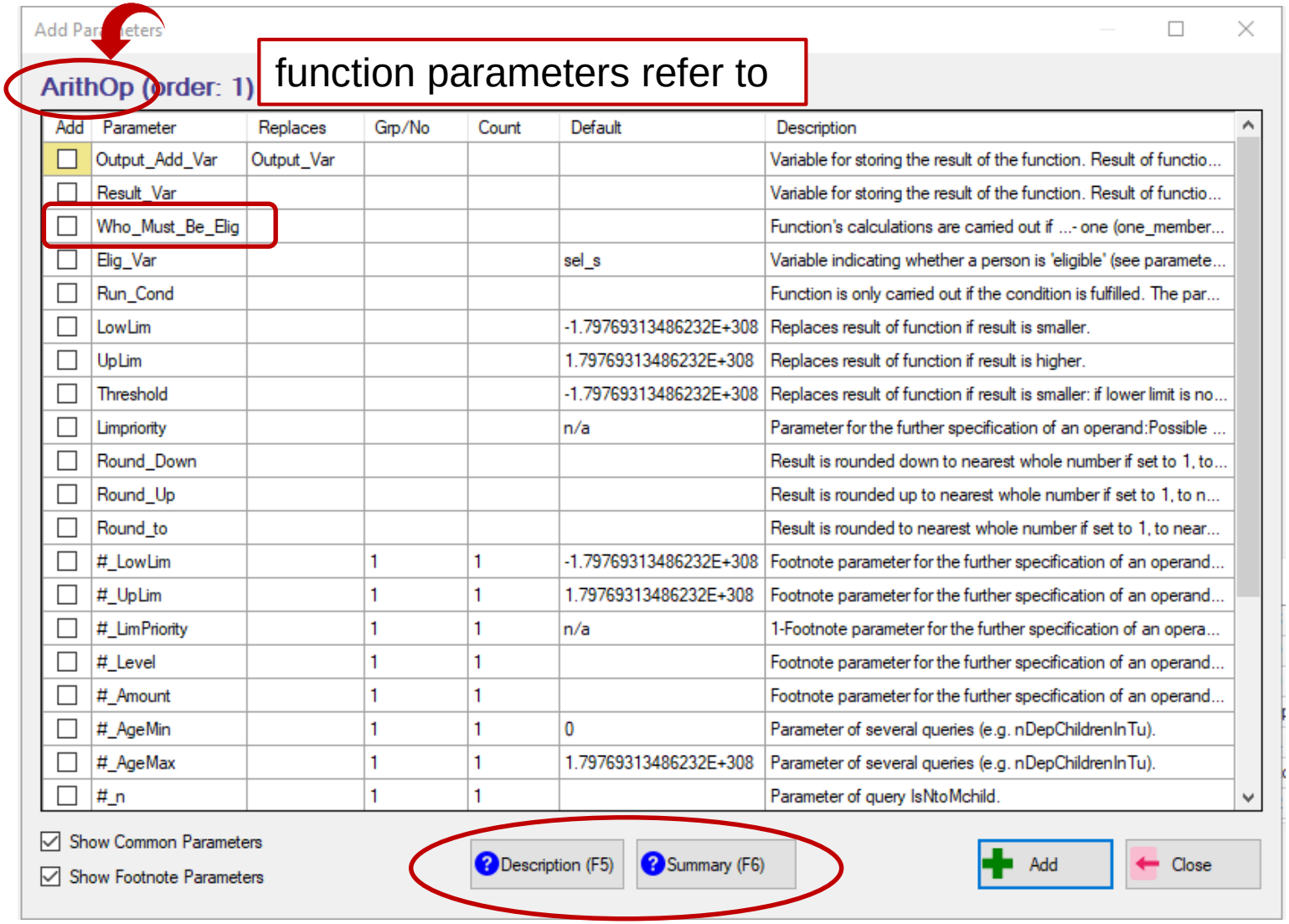
User Interface: Adding a parameter (1)

Policy	System Name	Con
Elig	**on**	Mac
elig_cond	IsLoneParentOfDepChild	if a
TAX_UNIT	tu_bu_ee	
ArithOp	**On**	Mac sup
formula	15#w	€15
output_add_var	bch_s	add fun
TAX_UNIT	tu_bu_ee	

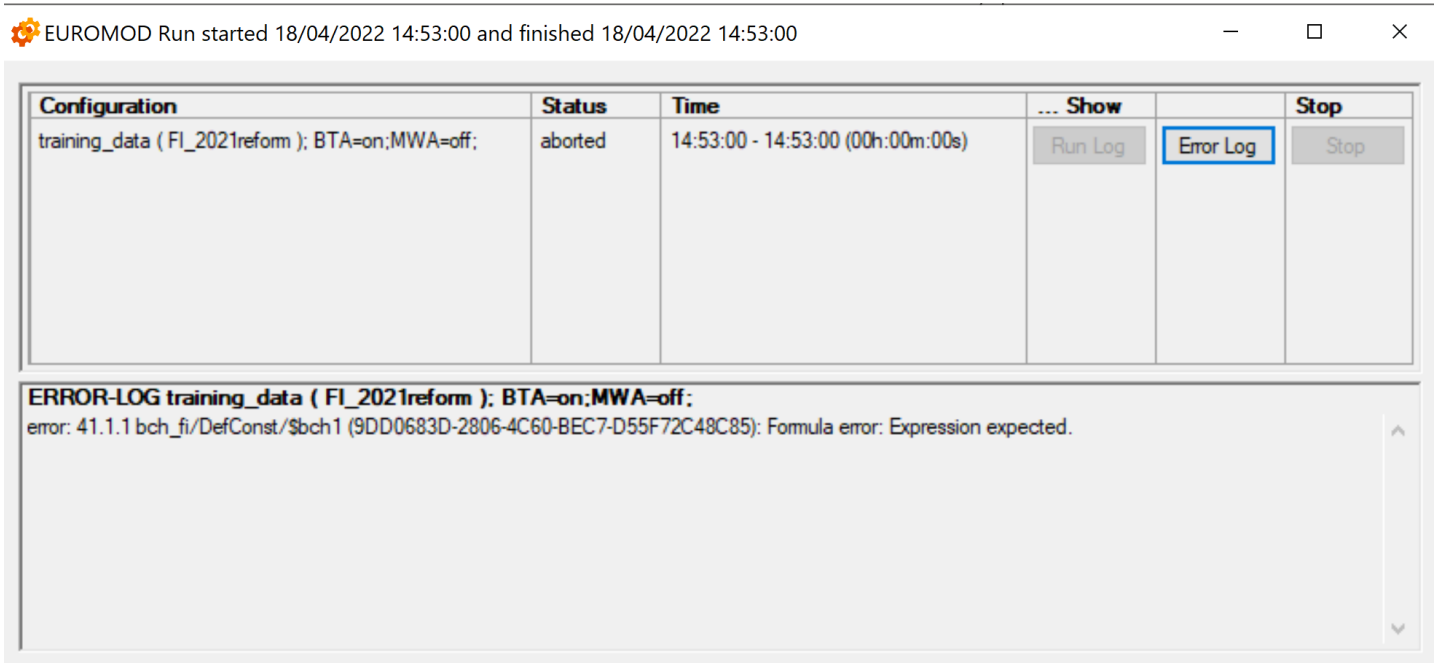
find relevant function and right-click on function or parameter name to open "menu"



User Interface: Adding a parameter (2)



Add	Parameter	Replaces	Grp/No	Count	Default	Description
☐	Output_Add_Var	Output_Var				Variable for storing the result of the function. Result of functio...
☐	Result_Var					Variable for storing the result of the function. Result of functio...
☐	Who_Must_Be_Elig					Function's calculations are carried out if ...- one (one_member...
☐	Elig_Var				sel_s	Variable indicating whether a person is 'eligible' (see paramete...
☐	Run_Cond					Function is only carried out if the condition is fulfilled. The par...
☐	LowLim				-1.79769313486232E+308	Replaces result of function if result is smaller.
☐	UpLim				1.79769313486232E+308	Replaces result of function if result is higher.
☐	Threshold				-1.79769313486232E+308	Replaces result of function if result is smaller: if lower limit is no...
☐	Limpriority				n/a	Parameter for the further specification of an operand:Possible ...
☐	Round_Down					Result is rounded down to nearest whole number if set to 1, to...
☐	Round_Up					Result is rounded up to nearest whole number if set to 1, to n...
☐	Round_to					Result is rounded to nearest whole number if set to 1, to near...
☐	#_LowLim		1	1	-1.79769313486232E+308	Footnote parameter for the further specification of an operand...
☐	#_UpLim		1	1	1.79769313486232E+308	Footnote parameter for the further specification of an operand...
☐	#_LimPriority		1	1	n/a	1-Footnote parameter for the further specification of an opera...
☐	#_Level		1	1		Footnote parameter for the further specification of an operand...
☐	#_Amount		1	1		Footnote parameter for the further specification of an operand...
☐	#_AgeMin		1	1	0	Parameter of several queries (e.g. nDepChildrenInTu).
☐	#_AgeMax		1	1	1.79769313486232E+308	Parameter of several queries (e.g. nDepChildrenInTu).
☐	#_n		1	1		Parameter of query IsNtoMchild.



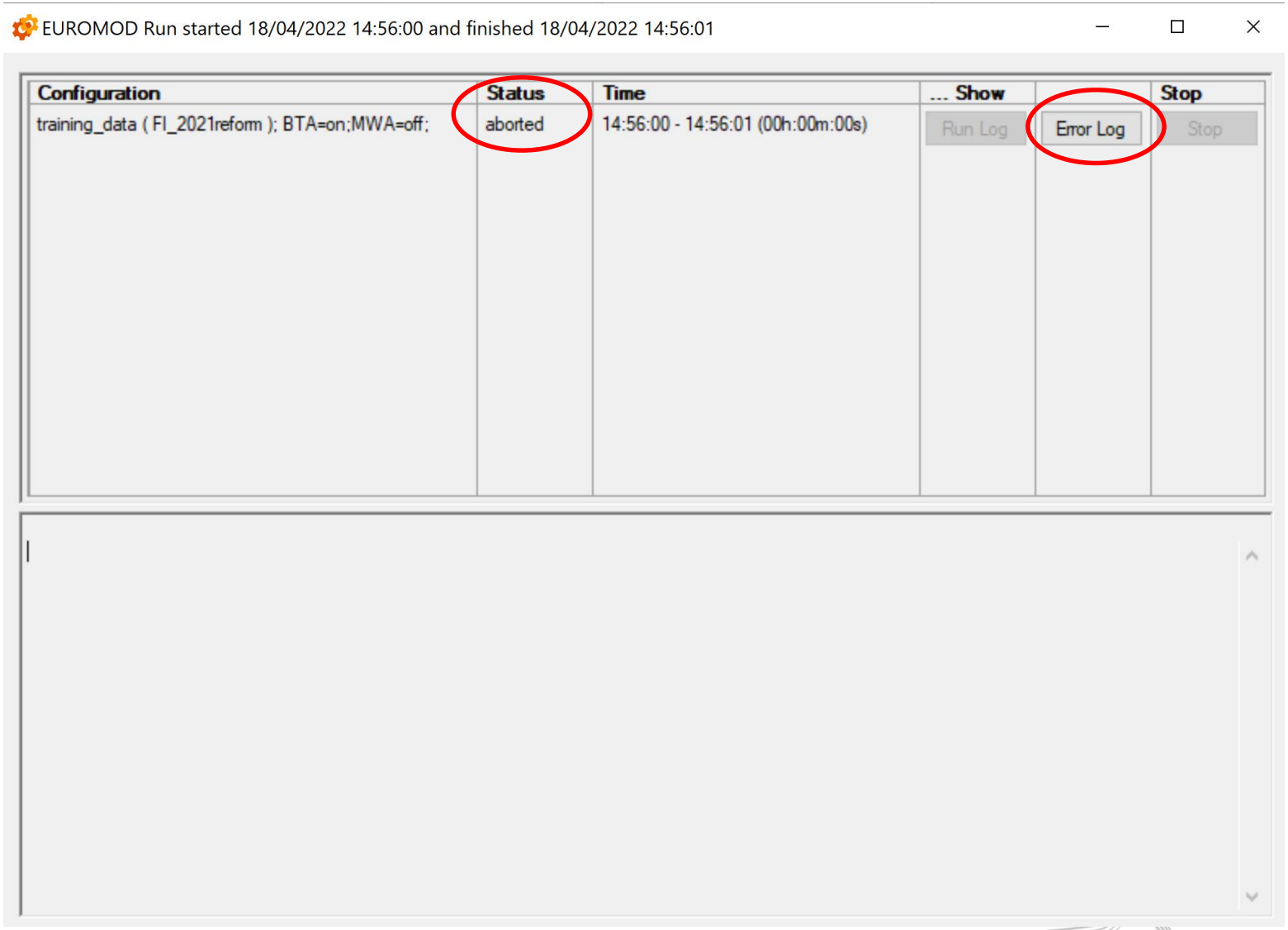
HANDLING ERRORS

Handling errors (1)

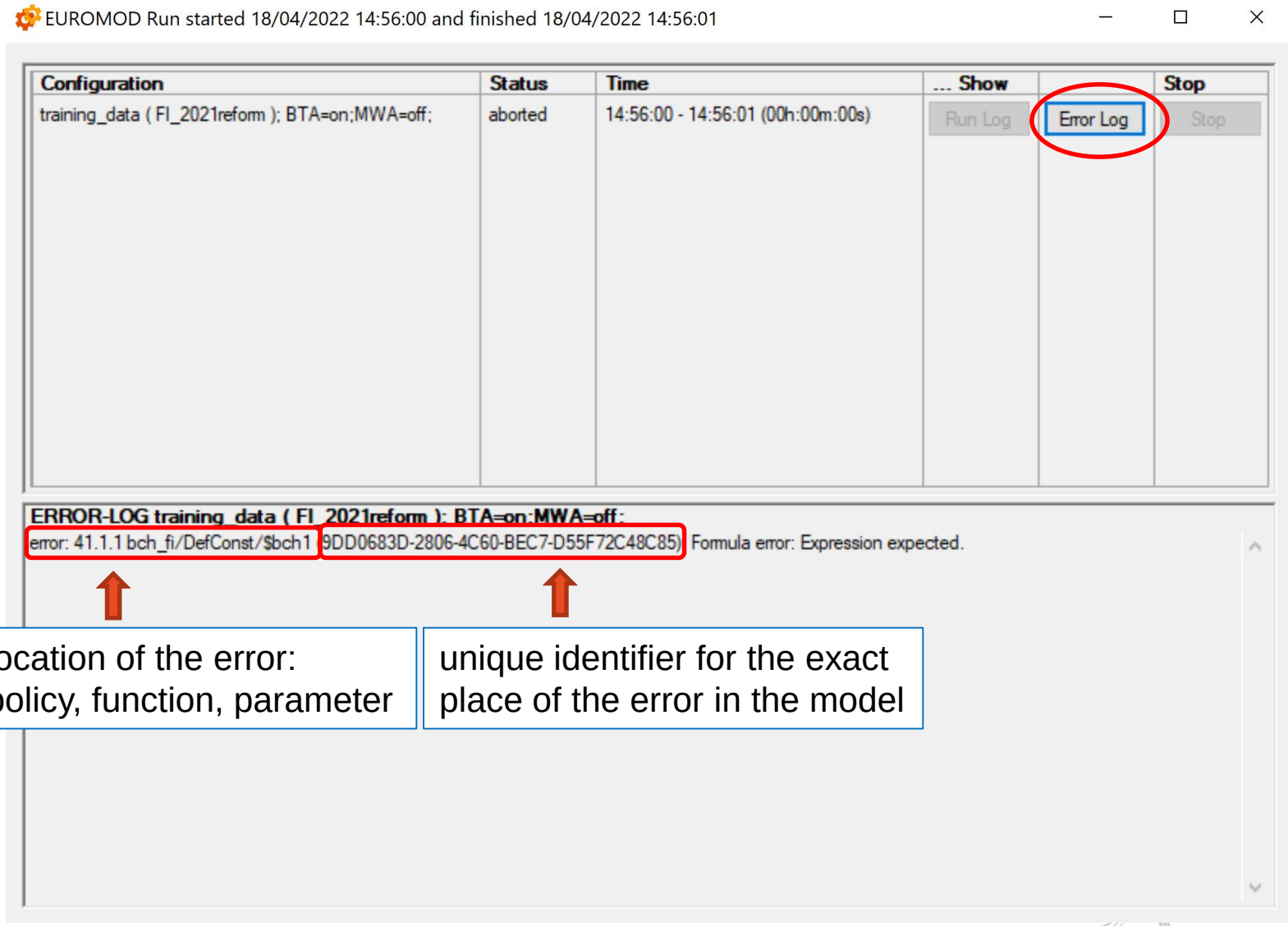
bch_fi		on	on	BEN: Child benefit
fx DefConst		on	on	Constants for Child Care Benefits
$bch1		94.88#m	100#	
$bch2		104.84#m	104.84#m	
$bch3		133.79#m	133.79#m	Child benefit for the 3rd child
$bch4		163.24#m	163.24#m	Child benefit for the 4th child
$bch5		182.69#m	182.69#m	Child benefit for the 5th child & subsequent
$bchsup		63.30#m	63.30#m	Child benefit supplement for a lone parent

typo in the syntax:
100# instead of 100#m

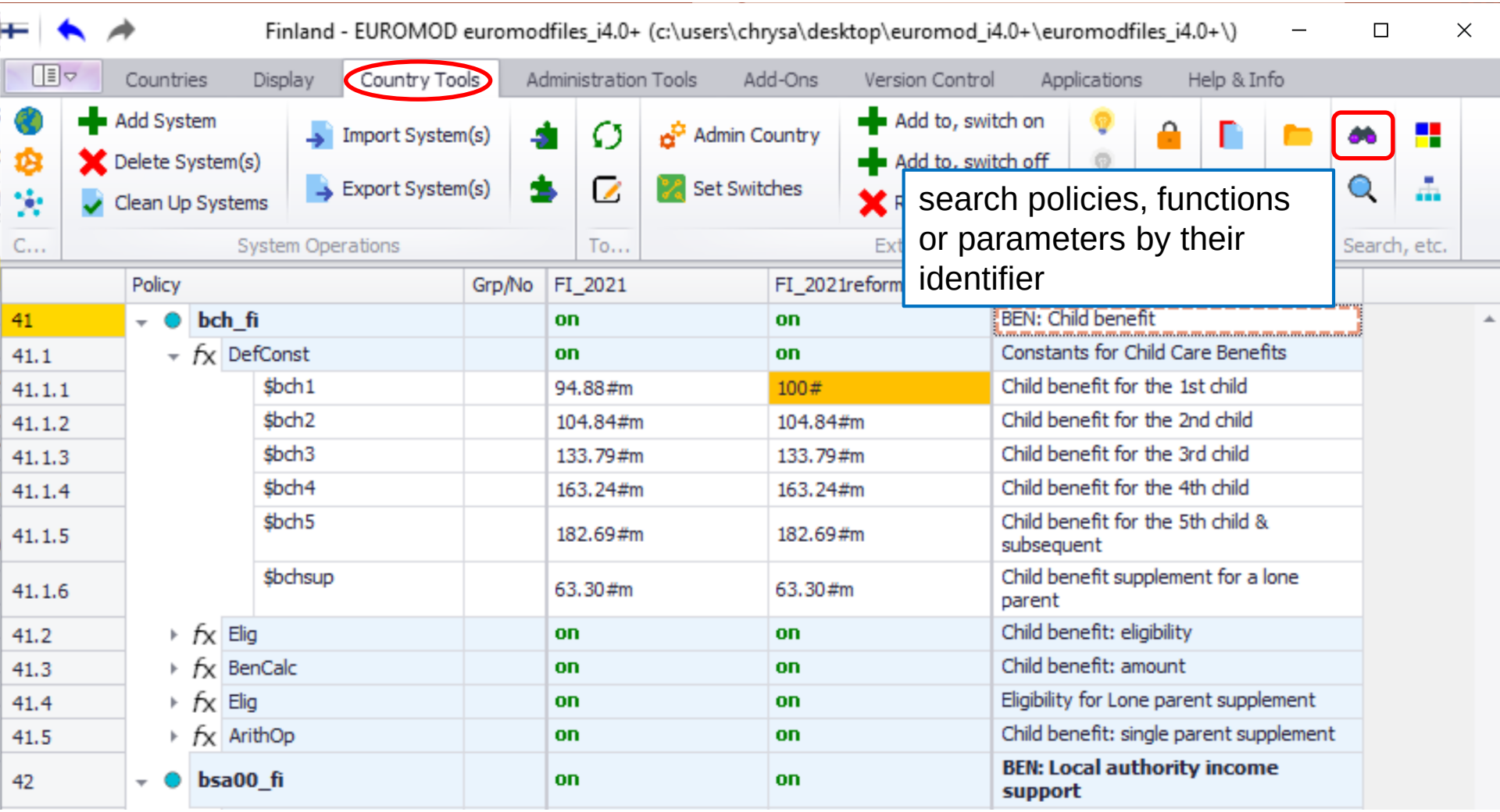
Handling errors (2)



Handling errors (3)

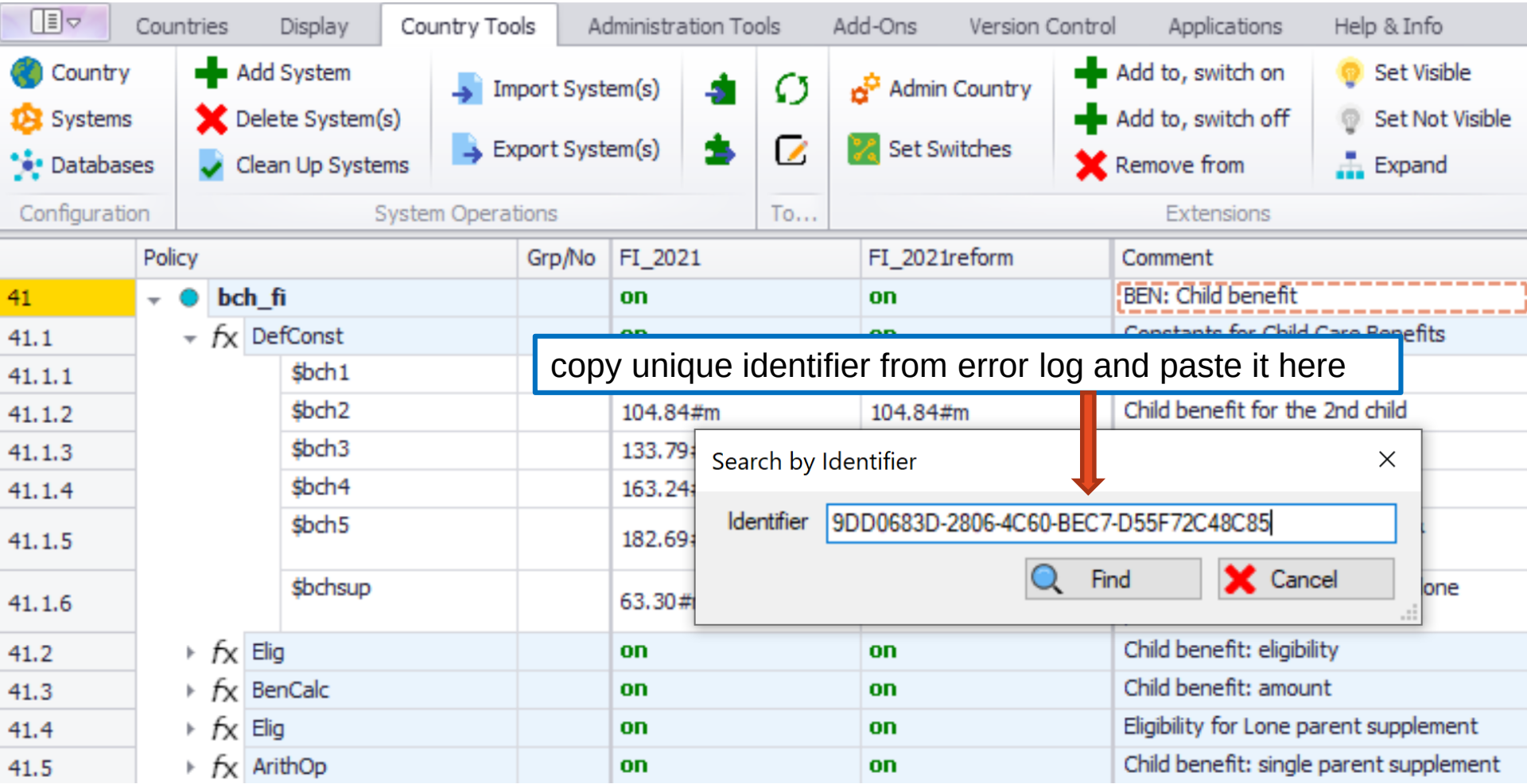


Handling errors (4)



	Policy	Grp/No	FI_2021	FI_2021reform	
41	**bch_fi**		**on**	**on**	BEN: Child benefit
41.1	DefConst		**on**	**on**	Constants for Child Care Benefits
41.1.1	$bch1		94.88#m	100#	Child benefit for the 1st child
41.1.2	$bch2		104.84#m	104.84#m	Child benefit for the 2nd child
41.1.3	$bch3		133.79#m	133.79#m	Child benefit for the 3rd child
41.1.4	$bch4		163.24#m	163.24#m	Child benefit for the 4th child
41.1.5	$bch5		182.69#m	182.69#m	Child benefit for the 5th child & subsequent
41.1.6	$bchsup		63.30#m	63.30#m	Child benefit supplement for a lone parent
41.2	Elig		**on**	**on**	Child benefit: eligibility
41.3	BenCalc		**on**	**on**	Child benefit: amount
41.4	Elig		**on**	**on**	Eligibility for Lone parent supplement
41.5	ArithOp		**on**	**on**	Child benefit: single parent supplement
42	**bsa00_fi**		**on**	**on**	**BEN: Local authority income support**

Handling errors (5)



European Commission

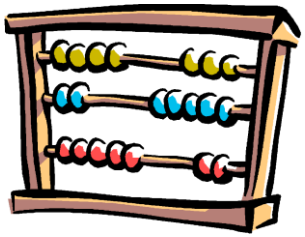
Handling errors (6)

- Output folder: error log file (text format)
- Same info as in the running dialog box
- Error logs contain time stamp of their creation
- Info about EUROMOD version, policy system where error occurred and dataset used

20220419102812_089fc7ee324d4740b15b10859c4ff22d_EUROMOD_Log - Notepad

File Edit Format View Help

```
===== EUROMOD LOG =====
Software-Version: 3.4.7
Project: euromodfiles_i4.0+
Start: 19 April 2022 10:28:11
End: 19 April 2022 10:28:12
Duration: 0.5839648s
Output-Path: c:\users\chrysa\desktop\euromod_i4.0+\euromodfiles_i4.0+\output\
-------------------------------------------------------------------------------------------------------------------------
===== RUN-LOG =====
Run-Id  Status  System  Database        Start   End     Duration        Currency        Exchangerate    BTA     MWA     Non-Default Ou
7028fa66-ab00-4cbf-95e8-bf83c3f1c0f2    aborted FI_2021reform   training_data   19 April 2022 10:28:11  19 April 2022 10:28:12  0.5839
-------------------------------------------------------------------------------------------------------------------------
===== ERROR-LOG =====
Run-Id: 7028fa66-ab00-4cbf-95e8-bf83c3f1c0f2
ERROR: 41.1.1 bch_fi/DefConst/$bch1 (9DD0683D-2806-4C60-BEC7-D55F72C48C85): Formula error: Expression expected.
```



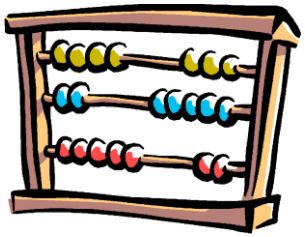
Exercise 3

Child allowance reform

Consider the effect of a Child Allowance (CA) reform in Estonia, in 2012. In that year, the CA was a monthly universal benefit. The amount per child was two times the Child Allowance Rate (CAR equal to 9.59 EUR), and 6 times the CAR for the third and any consequent child.

The reform adds a means-tested supplement to the universal CA for families with two children and with earnings below 200 EUR per month. The supplement is equal to 1800 EUR.

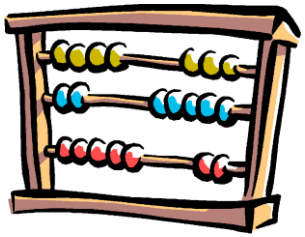
You will do this exercise on your own!



Exercise 3

Steps:

- Create a new system in Estonia based on the 2012 system and name it EE_2012_E2;
- Modify Child Allowance (CA) in way that access to the supplement should be restricted to families with at least two children and with earnings below 200 EUR per month. The supplement is equal to 1800 EUR per month;
- Run the model and use the *Statistics Presenter* – Baseline/Reform option to analyse the distributional effect of the reform.



Exercise 3

Hints:

- The variable for earnings is *yem* (y: market income; em: employment).
- Use the same variable name for your calculations as the one used in the *BenCalc* function in the same policy, i.e. *bch00_s* (b: benefit, ch: child, 00: main, _s: simulated).
- Use the same tax/assessment unit for your calculations as the one defined in the *BenCalc* function in the same policy, i.e. *tu_CBfamily_ee*.
- The number of dependent children in a tax unit is calculated by the query *nDepChildrenInTu*.
- Monthly amounts need to be indicated with the suffix *#m*.

Questions

Summary: Exercise 3

- You learned how to use Elig and ArithOp to simulate/reform a benefit, using
  - Who_Must_Be_Elig to link Elig and ArithOp
  - Queries & amount parameters
  - Parameter output_add_var

- You learned how to add a new function and parameters

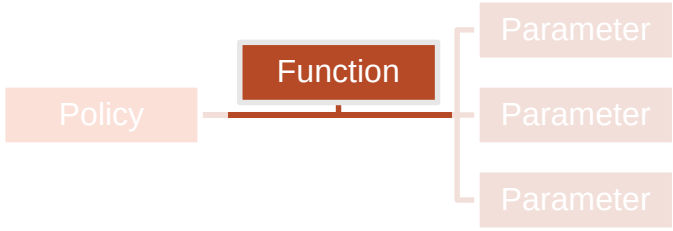
Session 4

Policy functions BenCalc & SchedCalc. Defining constants and income lists

In this session, you will learn about

- Combining the features of *Elig* & *ArithOp*, using policy function *BenCalc*
- Implementing a tax using policy function *SchedCalc*
- System functions to define
  - constants - *DefConst*
  - income lists - *DefIL*

Types of function



System functions

- functions used to define some general settings (e.g. to define income lists and tax units)
- **we will learn about *DefIL*, *DefConst*, *DefTU*, *DefVar*, *DefOutput*, *Uprate***

Policy functions

- functions used to implement tax-benefit policies (e.g. define eligibility, calculate amount etc.)
- **we already learned about *Elig* and *ArithOp***
- **we will also learn about *BenCalc*, *SchedCalc*, *Allocate***

Special functions

- more advanced functions that perform more complicated tasks (e.g. loops, change parameters at run-time etc.)
- **not covered in this course**

Policy		Grp/No	EE_2021
fx	BenCalc		on
	comp_cond	1	IsNtoMchild#1
	#_N	1	1
	#_M	1	1
	comp_perElig	1	$CB_Ch1
	comp_cond	2	IsNtoMchild#2
	#_N	2	2
	#_M	2	2
	comp_perElig	2	$CB_Ch2
	Comp_Cond	3	IsNtoMchild#3
	#_N	3	3
	#_M	3	99
	Comp_perElig	3	$CB_Ch3plus
	output_var		bch00_s
	TAX_UNIT		tu_CBfamily_ee

Policy function *BenCalc*

Function *BenCalc* (1)

- Very powerful function typically used to implement means-tested benefits
- Many benefits consist of several components, where a component is added if conditions are met by the assessment unit, and an income test, where certain types of income reduce benefit entitlement
- BenCalc allows these to be implemented/programmed easily
- It combines the functionalities of the functions Elig and ArithOp
- Compulsory parameters: Comp_Cond, Comp_PerTU/Comp_PerElig, output_var, TAX_UNIT

Function *BenCalc* (2)

- Comp_Cond used to define a condition
  - as *Elig_Cond* in function *Elig*
- Comp_perTU or Comp_perElig used to calculate a formula, as *Formula* in function *ArithOp*
  - Comp_perTU: amount is added once to the assessment unit
  - Comp_perElig: amount is added once for each individual fulfilling the condition in the assessment unit
- "groups" together Comp_Cond + Comp_perTU or Comp_perElig into one component
- can set upper and lower limits for each component: Comp_Lowlim, Comp_Uplim

Comp_perTU vs Comp_perElig

Policy	Grp/No	System Name	Comment
BenCalc		on	Made-up example: child benefit
Comp_Cond	1	nDepChildrenInTu>=1 & IsDepChild	if at least one dependent child in the assessment unit…
Comp_perTU	1	20#w	…benefit amount is €20 per week
output_var		bch_s	
TAX_UNIT		tu_CBfamily_ee	assessment unit used for the calculations

Policy	Grp/No	System Name	Comment
BenCalc		on	Made-up example: child benefit
Comp_Cond	1	nDepChildrenInTu>=1 & IsDepChild	if at least one dependent child in the assessment unit…
Comp_perElig	1	20#w	…benefit amount is €20 per week PER CHILD
output_var		bch_s	
TAX_UNIT		tu_CBfamily_ee	assessment unit used for the calculations

Function *BenCalc* (3)

- Ex. A lone parent family with two children, one with a disability, will receive €55#w

Policy	Grp/No	System Name	Comment
BenCalc		on	Made-up example: child benefit
Comp_Cond	1	nDepChildrenInTu>=1 & IsDepChild	if at least one dependent child in the assessment unit…
Comp_perElig	1	20#w	…benefit amount is €20 per week per child
Comp_Cond	2	IsDepChild & IsDisabled	if a child with a disability in the assessment unit…
Comp_perElig	2	10#w	…€10 per week for each child with a disability
Comp_Cond	3	IsLoneParentOfDepChild	if a lone parent in the assessment unit…
Comp_perTU	3	5#w	…€5 per week for the assessment unit
output_var		bch_s	
TAX_UNIT		tu_CBfamily_ee	assessment unit used for the calculations

Query IsNtoMchild#x

- IsNtoMchild#x is useful to implement child benefit which varies by *n* children
  - Returns 1 if a person belongs to the n to m oldest dependent children of the assessment unit, 0 otherwise.
  - n and m are defined by *footnote* parameters #_N and #_M.

- footnote parameters are referred to by #

- they are grouped together with main parameter they refer to

Policy	Grp/No	System Name	Comment
BenCalc		on	Made-up example: child benefit
Comp_Cond	1	IsNtoMchild#1	if the dependent child is…
#_N	1	1	…the first (oldest) child
#_M	1	1	
Comp_perElig	1	20#w	benefit amount is €20 per week for the first (oldest) child
Comp_Cond	2	IsNtoMchild#2	if the dependent child is…
#_N	2	2	…the second or any other child
#_M	2	99	
Comp_perElig	2	10#w	benefit amount is €10 per week for any other child
output_var		bch_s	
TAX_UNIT		tu_CBfamily_ee	assessment unit used for the calculations

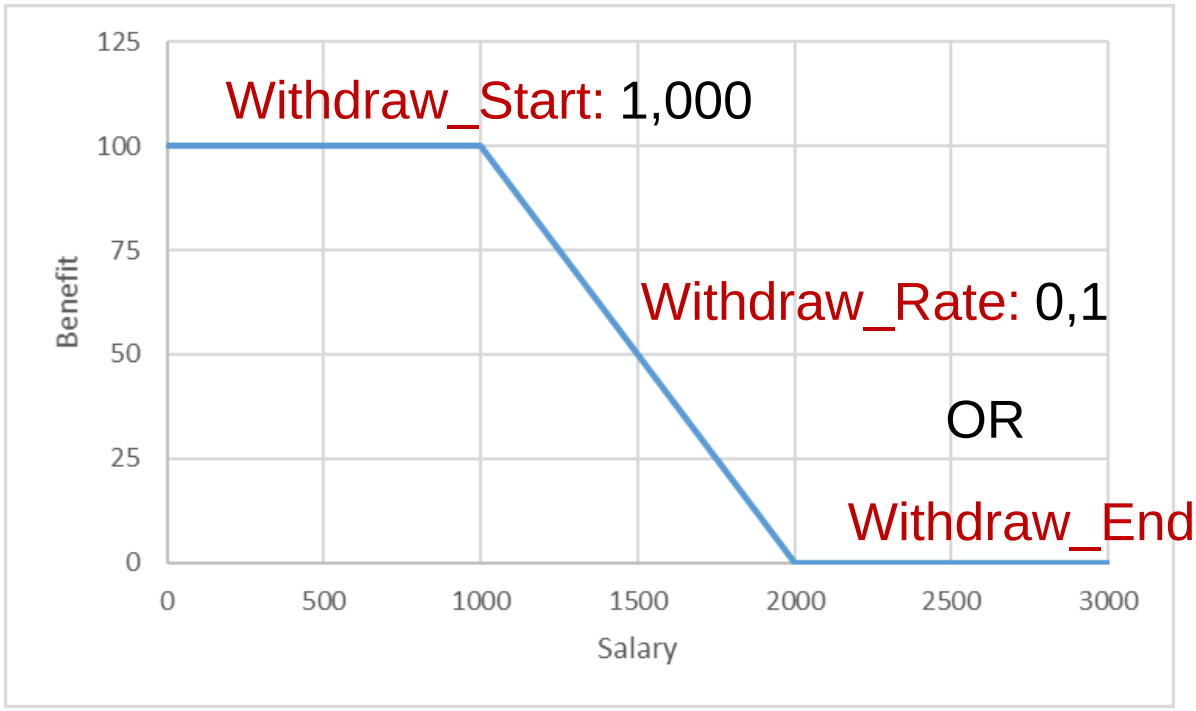
Function *BenCalc* (4)

- Withdraw parameters: subtract something from the calculated sum of components
  - Withdraw_Base: what is being subtracted (ex: variable, income list)
  - Withdraw_Rate: what percentage of the base is being subtracted
  - Withdraw_Start: level of the base where withdrawal starts
  - Withdraw_End: level of the base where sum of the components-base*rate is 0
- Negative result automatically set to 0
- Result=max(Sum of components-max(BASE-START, 0)*RATE, 0)
- *_Rate* and *_End* cannot be used simultaneously
  - If Withdraw_End is specified:
    - RATE=(sum of components)/(END-START)

Function *BenCalc* (5)

- Ex. Benefit for lone parents: 100 EUR/month benefit up to 1,000 EUR/month salary; for each additional EUR of salary, 0.1 EUR of the benefit are withdrawn

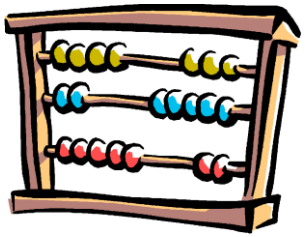
Withdraw_Base: salary



Function *BenCalc* (6)

Policy	Grp/No	System Name	Comment
BenCalc		on	Made-up example: child benefit
Comp_Cond	1	IsLoneParentOfDepChild	if there is a lone parent in the assessment unit…
Comp_perTu	1	100#m	…benefit amount is €100 per month as a general rule
Withdraw_Base		yem	If salary…
Withdraw_Start		1000#m	… is above €100 per month…
Withdraw_Rate		0.1	… €0.1 are withdrawn for each additional €1 earned above €1,000 per month
output_var		bch_s	
TAX_UNIT		tu_CBfamily_ee	assessment unit used for the calculations

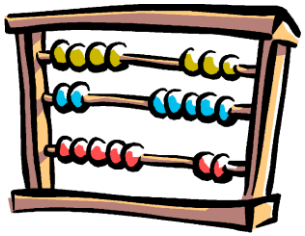
- the calculations done with the withdraw parameters apply to the sum of all components
- withdraw parameters do not need to be grouped



Exercise 4

Introducing a supplement, withdrawn with earnings, to the Child Benefit in Estonia

Perform a reform of the Child Allowance in Estonia in 2021. Your task is to **add** a supplement to the Child Allowance for families with at least 2 dependent children below the age of 18, using the function *BenCalc*. The supplement is equal to €30 per week but should be withdrawn with earnings: 10% should be withdrawn for each €1 that the family earns above €10,000 per year. Analyse the results after introducing the reform.

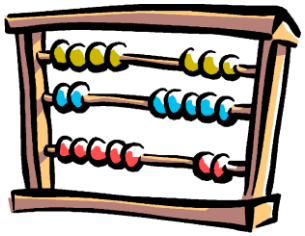


Exercise 4

individual study

Steps:

- Create a new system in the EE based on the 2021 system and name it EE_2021ref1.
- Introduce a supplement for large families to the Child Benefit, which is withdrawn with earnings.
- Run EUROMOD to produce micro-outputs for the baseline and reform systems.
- Analyse the distributive impact of this reform.



Exercise 4

individual study

Hints:

- To calculate the earnings withdrawal, use *BenCalc*'s optional parameters *Withdraw_Base*, *Withdraw_Start* and *Withdraw_Rate*.
- Use the same tax/assessment unit for your calculations as the one defined in the *BenCalc* function in the same policy, i.e. tu_CBfamily_ee (the family/benefit unit).
- Use the same variable name for your calculations as the one used in the *BenCalc* function in the same policy, i.e. bch00_s.
- The variable for earnings is called *yem*.

Parameter values and assessment unit: conditions vs. other parameters (1)

Level of Interpretation	... used in condition parameters	... used in other parameters
monetary variables and incomelists ...	assessment unit	assessment unit
non-monetary variables and individual level queries ...	individual	head of assessment unit
non individual level queries ...	consult description in section EUROMOD Functions - Queries	consult description in section EUROMOD Functions - Queries

Examples:

- monetary variables and income lists: yem, ils_origy
- non-monetary variables and individual level queries: dag, IsParent
- non-individual level queries: nDepChildren, nPersInUnit

Parameter values and assessment unit: conditions vs. other parameters (2)

Policy	System Name	Comment
Elig	**on**	Made-up example: condition to receive a housing benefit
elig_cond	**yem<30000#y & nPersonsInTU<=2 & dag>=60**	**household's earnings are less than €30,000 per year, there are max 2 persons in the household, and person's age is 60+**
TAX_UNIT	tu_household_ee	assessment unit is the household
ArithOp	**on**	Made-up example: housing benefit
Who_Must_Be_Elig	all	calculations carried out if each member of assessment unit fulfils condition from last Elig function
formula	**IsDisabled * (xhc – yem)**	**benefit funds expenditure for housing (xhc) if household head is disabled; benefit is reduced with household's earnings**
lowlim	0	min benefit amount is 0
output_var	bho_s	
TAX_UNIT	tu_household_ee	assessment unit is the household

fx SchedCalc		on	National Income tax schedule
Base		il_taxableY_nattax	Taxbase is income as defined in incomelist il_taxableY_nattax
Band_UpLim	1	15000#y	Income up to this annual amount is taxed ...
Band_Rate	1	0.23	... with this rate ...
Band_UpLim	2	28000#y	... income above the previous annual amount and up to this annual amount is taxed...
Band_Rate	2	0.27	... with this rate ...
Band_UpLim	3	55000#y	... income above the previous annual amount and up to this annual amount is taxed...
Band_Rate	3	0.38	... with this rate ...
Band_UpLim	4	75000#y	... income above the previous annual amount and up to this annual amount is taxed...
Band_Rate	4	0.41	... with this rate ...
Band_Rate	5	0.43	... income above the previous annual amount is taxed with this rate
output_var		tintsna_s	National Income tax schedule (i.e. National Income tax before tax credits)
lowlim		0	"This is to avoid negative results (i.e. if Taxbase is negative, the tax is set to 0)"
TAX_UNIT		tu_individual_it	

Policy function *SchedCalc*
System functions *DefConst, DefIL*

Function *SchedCalc* (1)

- Used (mainly) in progressive taxes to define a tax schedule:
  - Tax bands: Band_UpLim / Band_LowLim
  - Tax rate: Band_Rate
  - Tax base: Base
- Group together Band_Rate and Band_UpLim/ Band_LowLim
- Instead of rates, for fixed amounts use Band_Amount

Policy	Grp/No	System Name	Comment
SchedCalc		on	Made-up example: income tax
Base		tintb_s	income tax calculated based on taxable income (t: tax, in: income, tb: tax base, _s: simulated)
Band_Rate	1	0.2	first band rate: 20%
Band_LowLim	1	12500#y	first band rate applies on income above €12.5k per year…
Band_UpLim	1	50000#y	…and up to €50 per year
Band_Rate	2	0.4	second band rate: 40%
Band_UpLim	2	150000#y	second band rate applies on income above £50k and up to €150k per year
Band_Rate	3	0.45	third band rate: 45%; applies on income above €150k per year
output_var		tin_s	result saved in variable tin_s (t: tax, in: income, _s: simulated)
TAX_UNIT		tu_individual_it	assessment unit used for the calculations

Function *SchedCalc* (2)

- Quotient: split the base and apply the schedule separately
- Result = ((Base/Quotient)*Tax schedule)*Quotient

Policy	Grp/No	System Name	Comment
SchedCalc		on	Made-up example: income tax based on joint taxation
Base		tintb_s	income tax calculated based on taxable income (t: tax, in: income, tb: tax base, _s: simulated)
Band_Rate	1	0.2	first band rate: 20%
Band_LowLim	1	12500#y	first band rate applies on income above €12.5k per year…
Band_UpLim	1	50000#y	…and up to €50k per year
Band_Rate	2	0.4	second band rate: 40%
Band_UpLim	2	150000#y	second band rate applies on income above €50k and up to €150k per year
Band_Rate	3	0.45	third band rate: 45%; applies on income above €150k per year
Quotient		2	Base is divided by the quotient before the schedule is applied. Afterwards the result is multiplied by the quotient.
output_var		tin_s	result saved in variable tin_s (t: tax, in: income, _s: simulated)
TAX_UNIT		tu_couple_it	assessment unit used for the calculations

Function *SchedCalc* (3)

- Simple_Prog: apply on the whole base the highest marginal tax rate reached by it

Policy	Grp/No	System Name	Comment
SchedCalc		on	Made-up example: income tax
Base		tintb_s	income tax calculated based on taxable income (t: tax, in: income, tb: tax base, _s: simulated)
Band_Rate	1	0.2	first band rate: 20%
Band_LowLim	1	12500#y	first band rate applies on the whole income if income is between €12.5k per year…
Band_UpLim	1	50000#y	…and €50k per year
Band_Rate	2	0.4	second band rate: 40%
Band_UpLim	2	150000#y	second band rate applies on the whole income if income is between €50k and up to €150k per year
Band_Rate	3	0.45	third band rate: 45%; applies on the whole income if income is above €150k per year
Simple_Prog		yes	
output_var		tin_s	result saved in variable tin_s (t: tax, in: income, _s: simulated)
TAX_UNIT		tu_individual_uk	assessment unit used for the calculations

Function *DefConst* (1)

- Allows to define tax-benefit policy parameters as constants
  - e.g. monthly amounts, income thresholds, maximum amounts, tax rates
  - notated as *$x*
- Advantages:
  - over-time changes to tax-benefit parameters are visible at one glance
  - updating of tax-benefit parameters is easier
  - the same constant can be used in several policies (e.g. minimum wage)
- Constants are usually defined in the policy ConstDef_cc (if they are used in more than one policy) or at the beginning of a policy which calculates a tax/benefit

Function *DefConst* (2)

Policy		Grp/No	RO_2021	RO_2020	RO_2019	Comment
ConstDef_ro			on	on	on	**DEF: CONSTANTS**
fx DefConst			on	on	on	
	$FlatIncTax	1	0.10	0.10	0.10	
	const_mone...	1	no	no	no	
	$lhw	2	40	40	40	Standard number of working hours per week
	efConst		on	on	on	Average gross wage
	$AvGrossWage	1	5380#m	5429#m	5163#m	"The provisional average gross wage used issued by the National Institute of Statistics"; ;

user-defined constant names $x in "policy" column

changes to constant values highlighted

Function *DefIl* (1)

- Income lists are aggregates of several components
  - standard, used for distributional analysis (e.g. disposable income) or fiscal overview (e.g. taxes), compulsory for all countries
  - non-standard, used for policy implementation (e.g. taxable income), optional and country-specific

- System function DefIL allows to define income lists
  - standard, defined in the policy IlsDef_xx
    - prefix ils_
    - used by the Statistics Presenter!
  - non-standard, defined in the policy IlDef_xx or in policies where they are used
    - prefix il_

Function *DefII* (2)

- Aggregates:
  - variables
  - pre-defined income lists
  - fixed amounts
  - constants
- Operations:
  - +
  - -
  - shares (e.g. 0.8 means that the corresponding variable is multiplied by 0.8)
- Use
  - once defined, available for all subsequent functions and policies
  - behave like monetary variables
  - continuously updated throughout the spine

Function *DefIl* (3)

user-defined income list name in "system" column

- components of the income list
- some are being added, other subtracted

income list used to define another income list

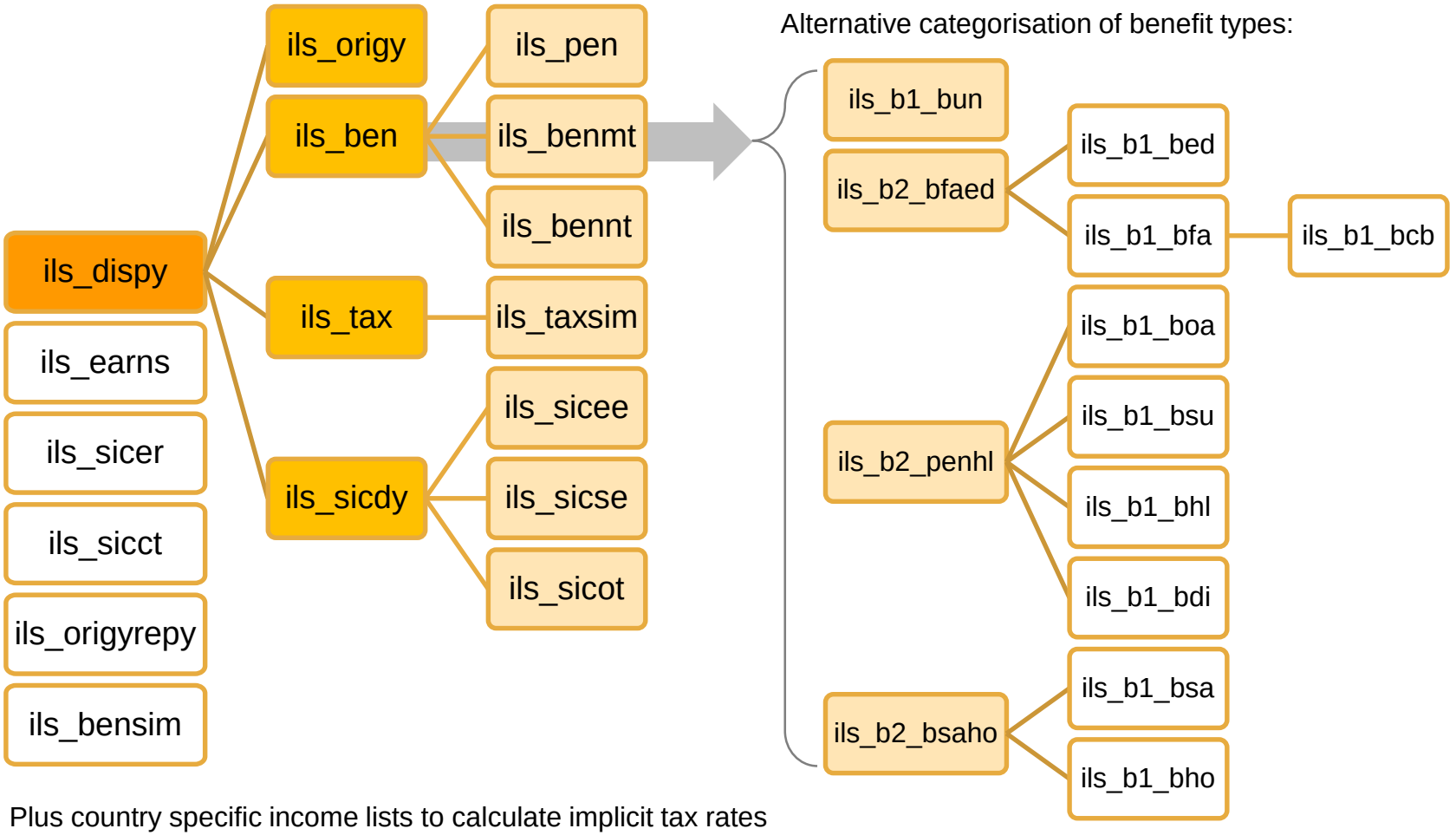
Policy	Grp/No	BG_2020	Comment
IlDef_bg		on	**DEF: NON-STANDARD INCOME CONCEPTS**
fx DefIl		on	Taxable income (before any tax exemptions/deductions) (il_taxableY)
name		il_taxableY	
yemtx		+	Taxable employment income
yemmc_s		+	Covid-19 in 2020: 60/40 wage compensation contribution paid by the employer
bwkmcee_s		+	Covid-19 in 2020: 60/40 wage compensation contribution paid by the state
ysetx		+	Taxable self-employment income
ypr		+	Income from property
yot		+	Income received by children
ils_sicee		-	employee social insurance contributions
ils_sicse		-	self-employed social insurance contributions

Standard income lists (1)

- Defined in every country
- Built in a comparable way to facilitate cross-national analysis
  - ils_earns: earnings
  - ils_origy: market incomes
  - ils_pen: public pensions
  - ils_bennt: non-means-tested benefits
  - ils_benmt: means-tested benefits
  - ils_ben: all benefits and public pensions
  - ils_tax: taxes
  - ils_sicee: employee SICs
  - ils_sicse: self-employed SICs
  - ils_sicdy: total SICs paid by individual
  - ils_sicer: employer SICs
  - ils_sicct: credited SICs
  - ils_dispy: disposable income
  - ils_bensim: simulated benefits
  - ils_taxsim: simulated taxes

Remember:
If you add a new policy that simulates a new benefit/tax, then you need to add the benefit/tax to the right income list!

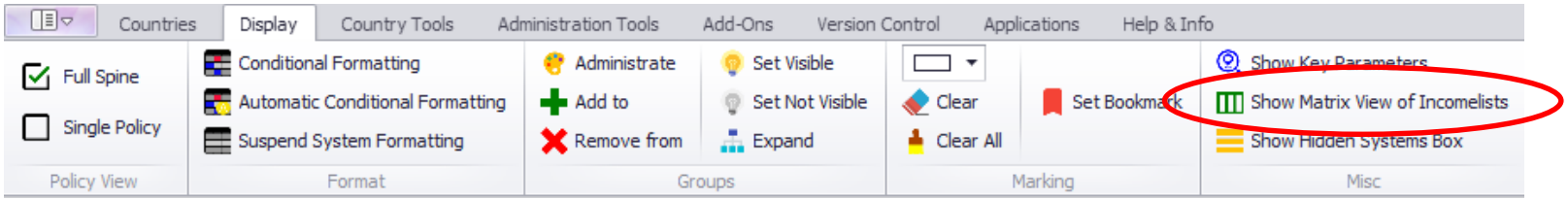
Standard income lists (2)



Plus country specific income lists to calculate implicit tax rates

Income lists - matrix view

- Summary of income list components



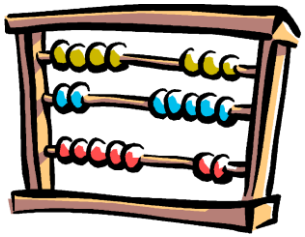
Matrix View of Incomelists for Bulgaria BG_2020

	il_bsaY	_extdispy	il_bchY	_taxable\	il_bsaht	il_pen	ils_eams	ils_origy	s_origrepy	ils_sicer	ils_sicct	ils_sicse	ils_sicee	ls_taxsim	ils_tax
yemtx	1	1	1	1	1		1	1	1						
bmaprct_s	1	1	1		1				1						
bmaycct_s	1	1	1		1				1						
bmaprnc_s	1	1			1										
pdinc	1	1	1		0.7225...	1									
pdiuc	1	1	1		0.7225...	1									
pdi00	1	1	1		0.7225...	1			1						
psu	1	1	1		0.7225...	1			1						
poa00	1	1	1		0.7225...	1			1						
poadi	1	1	1		0.7225...	1									
bhl	1	1	1		1				1						
bunct_s	1	1	1		1				1						
yot	1	1	1	1	1			1	1						
ypr	1	1	1	1	1			1	1						
ysetx	1	1	1	1	1		1	1	1						

Understanding constants and income lists

- We can now read and understand the code written in the *SchedCalc* function

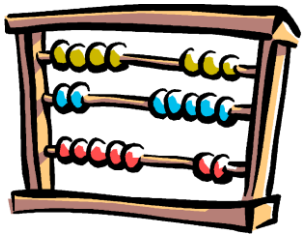
	Policy	Grp/No	ES_2020	Comment
37.27	fx Elig		on	General tax base higher than national tax allowance
37.27.1	Elig_Cond		tintbit_s-il_tintcit > 0	
37.27.2	TAX_UNIT		tu_individual_es	
37.28	fx SchedCalc		on	National tax schedule on general income (cuota 1 general estatall)
37.28.1	Who_Must_Be_Elig		all	
37.28.2	base		tintbit_s	tax base
37.28.3	band_uplim	1	$tin_ts_lt1	
37.28.4	band_uplim	2	$tin_ts_lt2	
37.28.5	band_uplim	3	$tin_ts_lt3	
37.28.6	band_uplim	4	$tin_ts_lt4	
37.28.7	band_uplim	5	$tin_ts_lt5	
37.28.8	band_uplim	6	$tin_ts_lt6	
37.28.9	band_rate	1	$tin_ts_rt1	
37.28.10	band_rate	2	$tin_ts_rt2	
37.28.11	band_rate	3	$tin_ts_rt3	
37.28.12	band_rate	4	$tin_ts_rt4	
37.28.13	band_rate	5	$tin_ts_rt5	
37.28.14	band_rate	6	$tin_ts_rt6	
37.28.15	band_rate	7	$tin_ts_rt7	
37.28.16	output_var		i_tiningt	gross income tax individual taxation
37.28.17	TAX_UNIT		tu_individual_es	tu: individual



Exercise 5a

Introducing a zero tax band to the flat income tax in Bulgaria

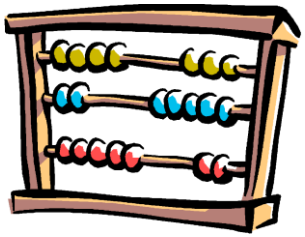
Currently, taxable income in Bulgaria is taxed at 10% from the first BGN earned. You are asked to introduce a zero tax band of BGN 3,200 per year, i.e. individuals should pay tax only on taxable income above BGN 3,200 per year. Furthermore, you should raise the marginal rate of income tax from 10% to 15%.



Exercise 5a

Steps:

- Create a new system in Bulgaria based on the 2021 system and name it BG_2021_reform.
- Introduce a zero tax band to the flat income tax and modify the tax rate.
- Run EUROMOD to produce micro-outputs for the baseline and reform systems.
- Analyse the distributive impact of this reform.



Exercise 5a

Hints:

- Define the zero tax band as a constant in the *DefConst* function in the income tax policy (e.g. call it *$flat_zeroband*)
- The tax rate is defined as a constant (e.g. *$flat_rate*) so modify the value of that constant.
- Introduce the zero tax band to the *SchedCalc* function, where the flat tax is calculated (*tin_bg*) by adding/modifying parameters in a *SchedCalc* function.

Summary of exercises

You learned:

- How to use *BenCalc* to implement a benefit, using parameters Comp_Cond + Comp_perTU and Withdraw_Base + Withdraw_Start + Withdraw_Rate
- How to use *SchedCalc* to reform the income tax policy, using parameter Band_LowLim and a DefConst function

Questions

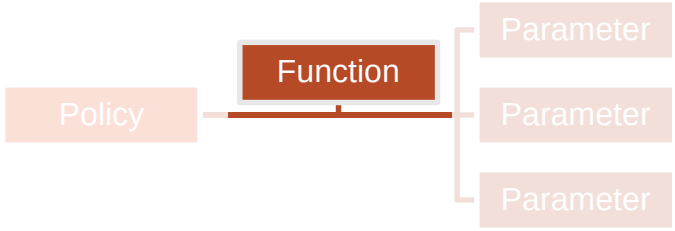
Session 5

Policy function Allocate. Defining tax units

In this session, you will learn about

- Allocating benefits and liabilities, using *Allocate*
- Parameters footnotes.
- System function to define tax units: *DefTU*

Types of functions



System functions

- functions used to define some general settings
- e.g. to define income lists and tax units
- **we already learned about DefIL, DefConst**
- **we will learn today about *DefTU***

Policy functions

- functions used to implement tax-benefit policies
- e.g. define eligibility, calculate amount etc
- **we already learned about *Elig* , *ArithOp* and *BenCalc, SchedCalc***
- **we will learn today about *Allocate***

Special functions

- more advanced functions that perform more complicated tasks
- e.g. loops, change parameters at run-time etc.
- **not covered in this course**

Policy	System Name	Comment
Allocate	on	Made-up example: Allocate Housing Benefit to the person responsible for housing costs and council tax
Share	bho_s	Housing Benefit to be reallocated between the members of the assessment unit (b: benefit, ho: housing, _s: simulated)
Share_Between	dhr=1	share the benefit between those responsible for the housing (dhr=1)
Share_All_IfNoElig	yes	if no one is responsible for the housing, share among all members
output_var	bho_s	save result in bho_s: result of Allocate function overwrites the current value of the variable
TAX_UNIT	tu_bu_lv	assessment unit used for the calculations

POLICY FUNCTION *Allocate*

Result of a function

- It is always assigned to the **head of the assessment unit**

- For all other members of the unit and for those in not eligible units (defined by who_must_be_elig) :
  - output_var is set to zero.
  - output_add_var not changed or set to 0 if undefined before
  - result_var is set to zero.

Function *Allocate* (1)

- Default: result is assigned to the head of the assessment unit
- Function *Allocate* is used to share/allocate the amount of a variable among specific members of the assessment unit
  - E.g. share the income tax from a joint income taxation among the partners in a couple → important for the simulation of benefits which depend on individual after-tax income
- Parameters of function *Allocate*
  - Share: which variable to split
    - amount to split first summed up across assessment unit members
  - Share_Between: condition parameter; who are the members 'participating' in the split
    - default is all members of the assessment unit
  - Share_Prop: in what proportion to split between the various qualifying members (i.e. those satisfying the Share_Between condition)
    - default is sharing in equal proportions

Function *Allocate (2)*

- Allocate reallocates amounts between members of assessment units (subject to conditions)

Policy	System Name	Comment
Allocate	on	Made-up example: Allocate Housing Benefit to the person responsible for housing costs and council tax
Share	bho_s	Housing Benefit to be reallocated between the members of the assessment unit (b: benefit, ho: housing, _s: simulated)
Share_Between	dhr=1	share the benefit between those responsible for the housing (dhr=1)
Share_All_IfNoElig	yes	if noone is responsible for the housing, share among all members
output_var	bho_s	save result in bho_s: result of Allocate function overwrites the current value of the variable
TAX_UNIT	tu_bu_lv	assessment unit used for the calculations

Function *Allocate (3)*

- Allocate reallocates amounts between members of assessment units (subject to conditions)

Policy	System Name	Comment
Allocate	on	Made-up example: Allocate the income tax - based on joint taxation - between the couple's partners
Share	tin_s	Income Tax to be reallocated between the partners of the couple (t: tax, in: income, _s: simulated)
Share_Prop	yem	share the income tax in proportion to the persons' earnings
Share_equ_Ifzero	yes	if noone has earnings, then income tax is shared equally between the couple's partners
Ignore_neg_prop	yes	if someome's earnings are negative, they are ignored (i.e. considered to be zero)
output_var	tin_s	save result in tin_s: result of Allocate function overwrites the current value of the variable
TAX_UNIT	tu_couple_lv	assessment unit used for the calculations

ASSESSMENT (TAX) UNITS & THE SYSTEM FUNCTION *DefTU*

Policy	System Name	Comment
DefTu	on	Made-up example: individual
Name	tu_individual_lv	
Type	IND	
DepChildCond	dag<16 | (dag<19 & IsInEducation & !IsMarried)	aged less than 16; or less than 19, in education and not married
DefTu	on	Made-up example: household
Name	tu_household_lv	
Type	HH	
DepChildCond	dag<16 | (dag<19 & IsInEducation & !IsMarried)	aged less than 16; or less than 19, in education and not married
DefTu	on	Made-up example: nuclear family
Name	tu_family_lv	
Type	SUBGROUP	
Members	Partner & OwnDepChild & LooseDepChild	head, head's partner, own and loose dependent children
DepChildCond	dag<16 | (dag<19 & IsInEducation & !IsMarried)	aged less than 16; or less than 19, in education and not married

Assessment/Tax Unit

- Unit: group of household members to be considered together
- Defined with System function *DefTU* usually in policy *TUDef_xx*
- Defined the first time used by the model
  - (although can be updated – see *UpdateTU* in extra slides)
- Naming convention: tu_yyyy_xx
- Parameter Type in *DefTU* defines the composition of the tax unit
  - HH: all individuals of the household are in the same unit.
  - IND: each individual of the household forms its own unit.
  - SUBGROUP: individuals determined by parameter *Members* form a unit. The household may be split into several units of different size.

Type of assessment/tax unit

Policy	System Name	Comment
DefTu	**on**	Made-up example: individual
Name	tu_individual_lv	
Type	IND	
DepChildCond	dag<16 | (dag<19 & IsInEducation & !IsMarried)	aged less than 16; or less than 19, in education and not married
DefTu	**on**	Made-up example: household
Name	tu_household_lv	
Type	HH	
DepChildCond	dag<16 | (dag<19 & IsInEducation & !IsMarried)	aged less than 16; or less than 19, in education and not married
DefTu	**on**	Made-up example: nuclear family
Name	tu_family_lv	
Type	SUBGROUP	
Members	Partner & OwnDepChild	head, head's partner, own dependent children
DepChildCond	dag<16 | (dag<19 & IsInEducation & !IsMarried)	aged less than 16; or less than 19, in education and not married

Assessment/tax units: examples

description	idhh	idperson	idpartner	idmother	idfather	dag	Household	Individual	Family (i.e. Subgroup)
typical family	1	101	102	0	0	30	A1	A1	A1
	1	102	101	0	0	28	A1	B1	A1
	1	103	0	102	101	3	A1	C1	A1
	1	104	0	102	101	1	A1	D1	A1
couple without children	2	201	202	0	0	56	A2	A2	A2
	2	202	201	0	0	55	A2	B2	A2
lone parent	3	301	0	0	0	35	A3	A3	A3
	3	302	0	301	0	6	A3	B3	A3
single	4	401	0	0	0	25	A4	A4	A4
two singles living together	5	501	0	0	0	22	A5	A5	A5
	5	502	0	0	0	23	A5	B5	B5
large family	6	601	602	606	0	48	A6	A6	A6
	6	602	601	0	0	45	A6	B6	A6
	6	603	0	602	601	20	A6	C6	B6
	6	604	0	602	601	15	A6	D6	A6
	6	605	0	602	601	10	A6	E6	A6
	6	606	0	0	0	70	A6	F6	C6

Head of the tax unit

- The head of a tax unit is by default
  - the richest member (System Configuration: Income for Head Definition)
    - the oldest
      - with the smallest personal id (i.e. *idperson)*
- Additional, optional specifications for the head:

Query	Value type	Default	Description
HeadDefInc	variable or incomelist	ils_origy	Incomelist used for determining who is the richest person in the assessment unit, see parameter ExtHeadCond.
ExtHeadCond	condition	!IsDepChild	Condition further defining the head of the assessment unit.
StopIfNoHeadFound	yes/no	no	If set to yes: an error is issued if ExtHeadCond rules out all household members. If set to no: no error issued, instead ExtHeadCond is dropped for affected households.
NoChildIfHead	yes/no	no	If set to yes (possible) child status is removed if person is the Head of the assessment unit.
NoChildIfPartner	yes/no	no	If set to yes (possible) child status is removed if person is Partner as defined by parameter PartnerCond.

Members of the tax unit (1)

- If tax unit type is SUBGROUP, parameter Members defines which hh members form a unit
- Relations are often defined with respect to the *head* of the unit
  - e.g. Partner, OwnChild, DepParent
- The status of each member is defined by a xCond parameter
  - e.g. PartnerCond, OwnChildCond, DepParentCond

Members of the tax unit (2)

Query	Default	Description
Members	n/a	Specifies which members of the household form a unit, if parameter Type is set to SUBGROUP, e.g. Partner & OwnChild.
PartnerCond	head:idperson=idpartner	Condition defining who is a partner, i.e. the partner of the head.
DepChildCond	0, i.e. nobody is a child	Condition defining who is a dependent child. The parameter equals 0, i.e. nobody is child, if not defined or set to n/a. However, setting the parameter to ***Default*** is the same as typing ***!IsParent & idpartner=0***.
OwnChildCond	head:idperson=idmother | head:idperson=idfather | partner:idperson=idmother | partner:idperson=idfather	Condition defining who is an own child: either the head or the head's partner is the mother/father of the child.
OwnDepChildCond	IsOwnChild & IsDepChild	Condition defining who is an own dependent child.
LooseDepChildCond	idmother=0 & idfather=0 & IsDepChild	Condition defining who is a loose dependent child.
DepParentCond	head:idmother=idperson | head:idfather=idperson | partner:idmother=idperson | partner:idfather=idperson	Condition defining who is a dependent parent, i.e. the parent of the head or the head's partner.
DepRelativeCond	0, i.e. nobody is a dependent relative	Condition defining who is a dependent relative.
LoneParentCond	IsParentOfDepChild & idpartner=0	Condition defining who is a lone parent.

Examples: Members of the tax unit (1)

- head: subsequent variable refers to the head
- partner: subsequent variable refers to the partner of the head
- Default setting can be overwritten or combined with further specifications

Policy	System Name	Comment
DefTu	**on**	Made-up example: couple with dependent parents
Name	tu_couple1_lv	
Type	SUBGROUP	
Members	Partner & DepParent	head, head's partner and dependent parents
DepParentCond	head:idmother=idperson | head:idfather=idperson	overwriting the default condition: only include parents of the head
DefTu	**on**	Made-up example: couple
Name	tu_couple2_lv	
Type	SUBGROUP	
Members	Partner	head and head's partner
PartnerCond	Default & IsMarried	default condition (i.e. head's partner) + partner is married

Examples: Members of the tax unit (2)

- DepChildCond: determines who is the dependent child
- OwnDepChild: “a son or daughter” (see OwnChildCond), who is a dependent child
- LooseDepChild: “someone, who is a dependent child, but doesn’t cohabit with parent/s”

Policy	System Name	Comment
DefTu	on	Made-up example: nuclear family
Name	tu_family1_lv	
Type	SUBGROUP	
Members	Partner & OwnDepChild	head, head's partner and **own** dependent children
DepChildCond	dag<=15	definition of the dependent child: aged 15 or less

idhh	idperson	idpartner	idmother	idfather	dag	ils_origy	IsHead	IsPartner	IsDepChild	IsOwnDepChild	IsLooseDepChild	tax unit
1	101	102	0	0	44	2000	no	yes	no	no	no	A
1	102	101	0	0	45	2100	yes	no	no	no	no	A
1	103	0	102	101	14	400	no	no	**yes**	**yes**	**no**	A
1	104	0	0	0	12	0	yes	no	**yes**	**no**	**yes**	**B**

Examples: Members of the tax unit (3)

- DepChildCond: determines who is the dependent child
- OwnDepChild: “a son or daughter” (see OwnChildCond), who is a dependent child
- LooseDepChild: “someone, who is a dependent child, but doesn’t cohabit with parent/s”

Policy	System Name	Comment
DefTu	on	Made-up example: nuclear family
Name	tu_family2_lv	
Type	SUBGROUP	
Members	Partner & OwnDepChild & LooseDepChild	head, head's partner, **own and loose** dependent children
DepChildCond	dag<=15	definition of the dependent child: aged 15 or less

idhh	idperson	idpartner	idmother	idfather	dag	ils_origy	IsHead	IsPartner	IsDepChild	IsOwnDepChild	IsLooseDepChild	tax unit
1	101	102	0	0	44	2000	no	yes	no	no	no	A
1	102	101	0	0	45	2100	yes	no	no	no	no	A
1	103	0	102	101	14	400	no	no	**yes**	**yes**	**no**	A
1	104	0	0	0	12	0	no	no	**yes**	**no**	**yes**	**A**

Avoiding splitting up families

Query	Value type	Default	Description
AssignDepChOfDependents	yes/no	no	If set to yes dependent children of dependent unit members (i.e. persons who are not Head or Partner of the head) are assigned to the unit.
AssignPartnerOfDependents	yes/no	no	If set to yes partners of dependent unit members (i.e. persons who are not Head or Partner of the head) are assigned to the unit.

Using conditions which refer to income

- If the assessment unit is bigger than the individual, monetary variables or income lists are assessed at the level of the whole unit
- Use *footnote* parameters to specify an alternative assessment unit

Policy	Grp/No	System Name	Comment
DefTu		on	Made-up example: family
Name		tu_family_lv	
Type		SUBGROUP	
Members		Partner & OwnDepChild & LooseDepChild	
DepChildCond		dag<25 & yem#1 < 1000#m	aged less than 25 and with (individual) earnings of less than GBP 1,000 per month
#_Level	1	tu_individual_lv	earnings are assessed at the individual level

- Now let's have a look at *footnote* parameters

Parameters: *Footnotes*

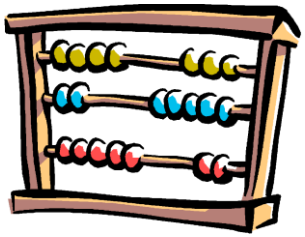
- They serve the further specification of other parameters.
- Identified by #i (i=number from 1 to....)
  - Limits
  - Amounts
  - Assessment units
  - Specification of queries

Policy	Grp/No	System Name	Comment
ArithOp		on	Made-up example: tax allowance for pensions
Formula		il_pensions#1 + (ils_earns#2 + GetPartnerIncome#3)*0.3	the person's pensions - up to 1,000 per month - plus 30% of the sum of the couple's earned income and the pension of the partner
#_UpLim	1	1000#m	upper limit on the person's pensions
#_Level	2	tu_couple_lv	alternative assessment unit
#_Income	3	poa	get the pension of the partner
output_var		tintape_s	
TAX_UNIT		tu_individual_lv	

The tax unit in the output file

Policy	Grp/ No	System Name
DefTu		on
Name		tu_family_lv
Type		SUBGROUP
Members		Partner & OwnDepChild & LooseDepChild & DepParent
DepChildCond		dag<25 & yem#1<8000#y
DepParentCond		Default & dag>65 & yem#1<8000#y
#_Level	1	tu_individual_lv
AssignDepChOfDependents		yes
AssignPartnerChOfDependents		yes
LoneParentCond		Default & !IsMarried

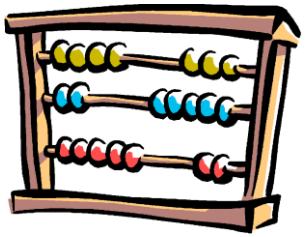
idhh	idperson	idpartner	idmother	idfather	dag	is_origy	yem	tu_family_lv_Headed	tu_family_lv_IsPartner	tu_family_lv_IsDepchild	tu_family_lv_IsDePparent	tu_family_lv_IsILoneParent
1	101	102	0	0	65	0	0	101	0	0	0	0
1	102	101	0	0	60	0	0	101	1	0	0	0
1	103	0	102	101	30	0	0	103	0	0	0	0
1	104	0	102	101	28	166	147	104	0	0	0	0
2	201	202	0	0	29	1,085	1,007	201	0	0	0	0
2	202	201	0	0	25	953	891	201	1	0	0	0
2	203	0	202	201	3	0	0	201	0	1	0	0
2	204	0	202	201	2	0	0	201	0	1	0	0
52	5,201	5,202	5,206	5,205	40	1,954	1,831	5,201	0	0	0	0
52	5,202	5,201	0	0	38	0	0	5,201	1	0	0	0
52	5,203	0	5,202	5,201	10	0	0	5,201	0	1	0	0
52	5,204	0	5,202	5,201	15	0	0	5,201	0	1	0	0
52	5,205	5,206	0	0	70	0	0	5,201	0	0	1	0
52	5,206	5,205	0	0	70	0	0	5,201	0	0	1	0
92	9,201	0	0	0	80	0	0	9,202	0	0	1	0
92	9,202	0	0	9,201	38	3,740	3,502	9,202	0	0	0	1
92	9,203	0	0	9,201	34	2,483	2,324	9,203	0	0	0	0
92	9,204	0	0	9,202	11	0	0	9,202	0	1	0	0



Exercise 7

Reforming PT Child Benefit by narrowing its age condition

Limit the provision of the Child benefit in the Portugal in 2020, to families with children aged up to 13, i.e. families with children aged 14+ will lose the benefit. Currently the benefit is paid up to the age of 16 or until 24 if a child is in full-time education.



Exercise 7

Steps:

- Open the PT 2020 system
- Use the function DefTU to create a new assessment unit *tu_buref_pt* which includes the reformed age condition for a dependent child
  - Make a copy of PT 2020 system and call it e.g. PT_2020refTU
  - You can copy/paste an existing tax unit (*tu_bch_fa_pt*) and modify it
  - Reform the Child Benefit so that families with children aged 14+ no longer receive this benefit
- Run the new system and use the Statistics Presenter to analyse the impact of the reform on inequality and poverty.

Questions

Summary: Exercise 7

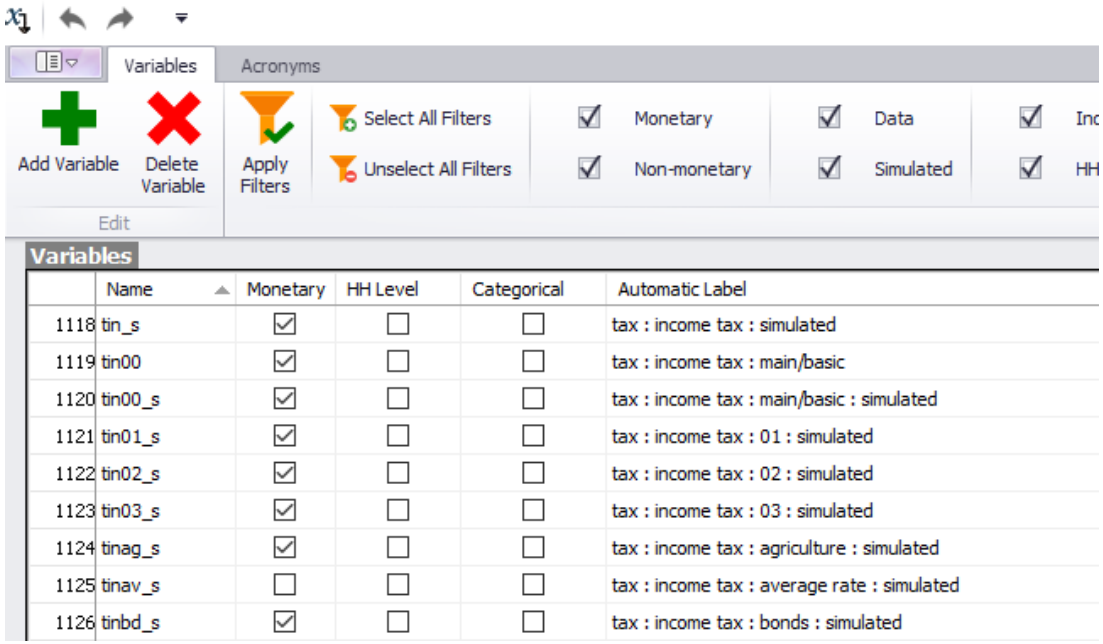
- You learned how to
  - Create a new assessment unit
  - Copy/paste a function
  - Modify those assessment units that are already defined into the model

Session 6

System functions DefVar & DefOutput. Variable types. Uprating indices

In this session, you will learn about

- EM variables: types, naming conventions, variable library and output
- System functions to define
  - variables: *Defvar*
  - output: *DefOutput*
- Uprating indices and function Uprate
- EUROMOD documentation and help



Variable types & system functions *DefVar* and *DefOutput*

Variable types

- **Standard** - following EUROMOD naming conventions
  - Household and individual characteristics
  - Incomes: market incomes, simulated and non-simulated benefits and taxes
  - Assets and expenditures
- **Intermediate**
  - Used to save the result of an intermediate calculation (e.g. an eligibility condition, number of twins in the tax unit, a subcomponent of a tax liability)

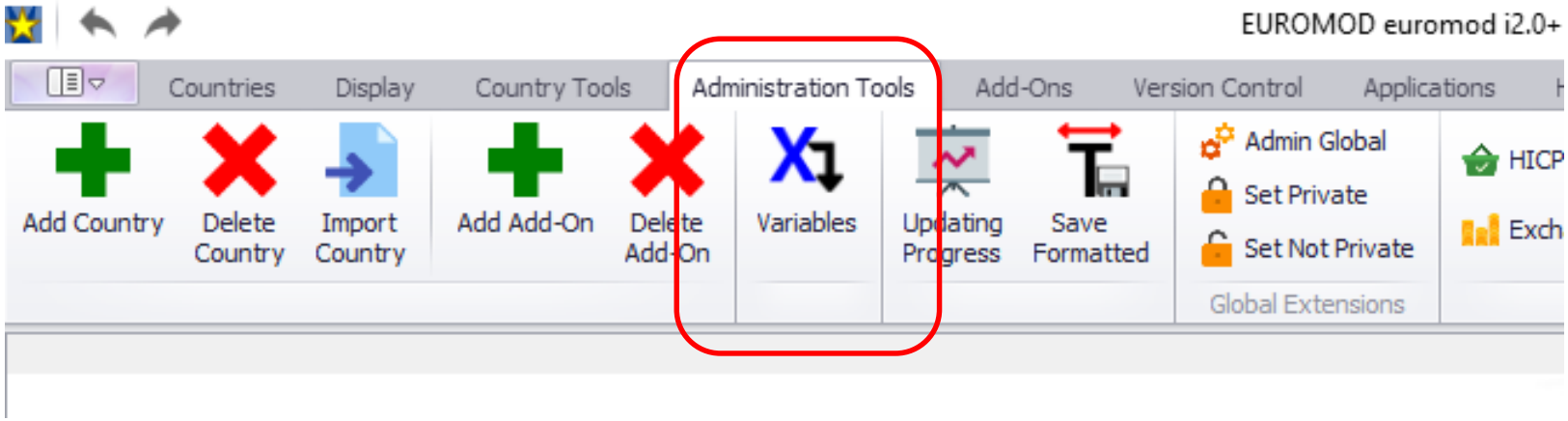
Variable naming conventions (1)

- Applicable to standard variables:
  - Included in the EUROMOD input microdata, or
  - Created in the EUROMOD spine for policy simulations and saved in the output microdata (*_s)
- The goal is to achieve:
  - Intuitive variable names
  - Harmonised variable names to allow for consistent cross-county comparisons

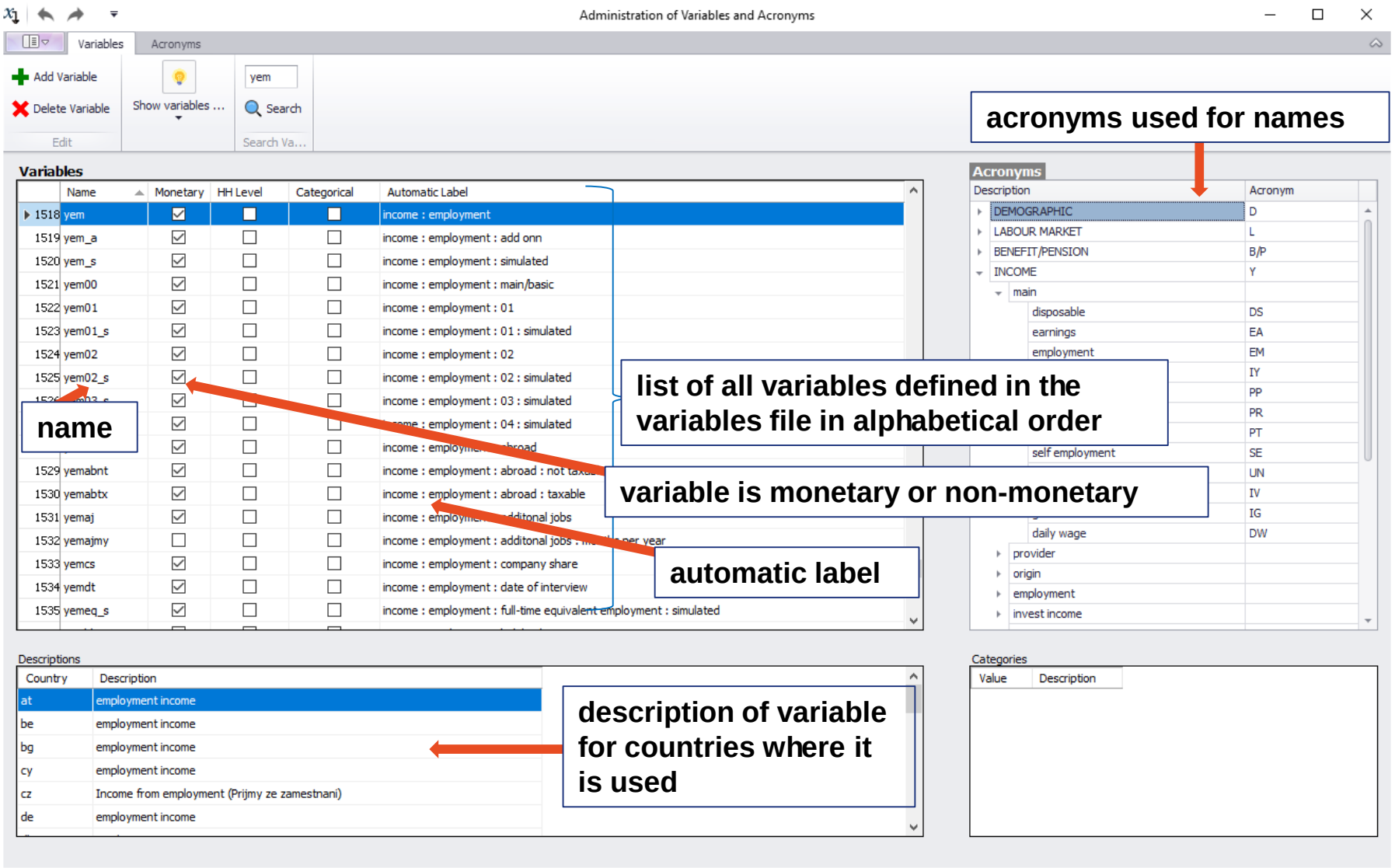
Variable naming conventions (2)

- Names are combination of acronyms: abb**[_s]
  - a = type of information
    - e.g. y=income, x=expenditure, d=demographic, l=labour
  - bb = specific for each type a, e.g.
    - e.g. y|em: employment income, y|se: self-employment income
  - ** = further bb's for additional information/detail, e.g.
    - e.g. y|em|xp: employment income, extra pay
  - _s for simulated variables
    - e.g. b|sa|rg_s: benefit, social assistance, regional, simulated
  - exception id*, e.g. idperson, idmother
- Acronyms and list of variables stored in a common variables library (VarConfig.xml)

Variables library (1)



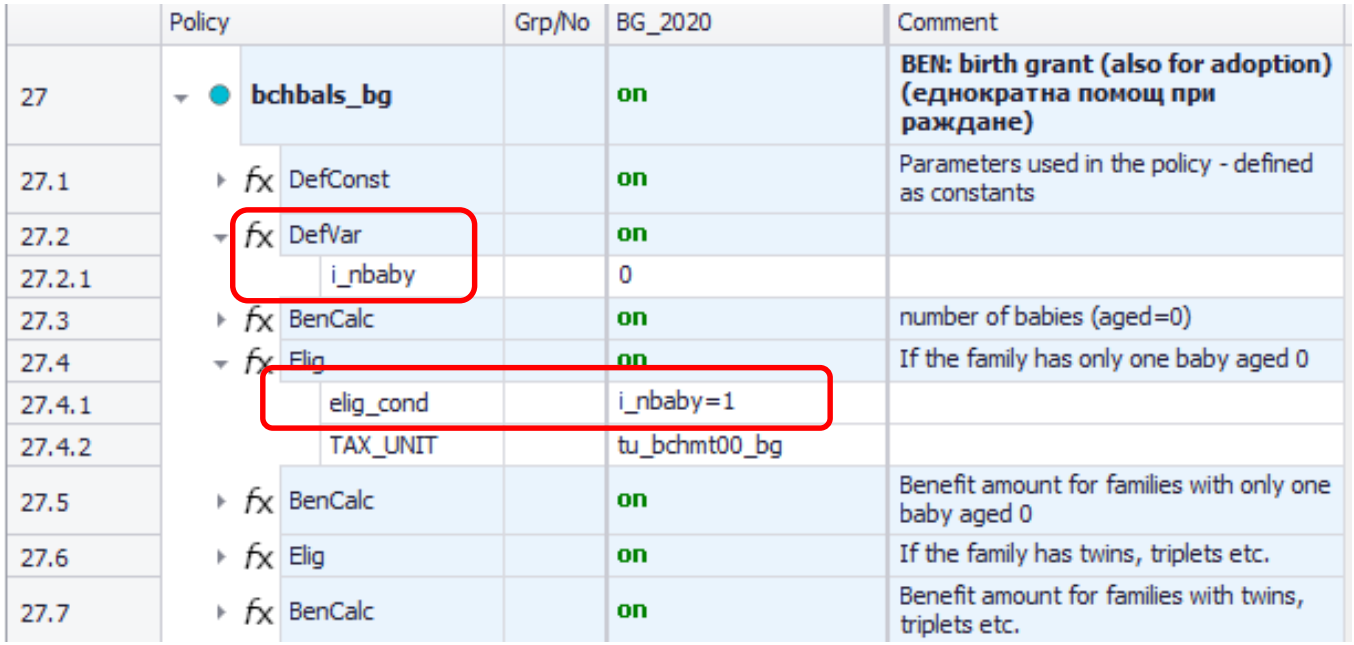
Variables library (2)



European Commission

System function *DefVar*

- To define intermediate (temporary) variables not included in the Variables library



	Policy	Grp/No	BG_2020	Comment
27	**bchbals_bg**		on	**BEN: birth grant (also for adoption) (еднократна помощ при раждане)**
27.1	DefConst		on	Parameters used in the policy - defined as constants
27.2	DefVar		on	
27.2.1	i_nbaby		0	
27.3	BenCalc		on	number of babies (aged=0)
27.4	Elig		on	If the family has only one baby aged 0
27.4.1	elig_cond		i_nbaby=1	
27.4.2	TAX_UNIT		tu_bchmt00_bg	
27.5	BenCalc		on	Benefit amount for families with only one baby aged 0
27.6	Elig		on	If the family has twins, triplets etc.
27.7	BenCalc		on	Benefit amount for families with twins, triplets etc.

- Not needed for pre-defined system variables sin00_s - sin50_s

Variables: summary

	Used in the spine	Variable exists in the variables list	Acronyms exist in the variables list	Action
Standard variables	yes	yes	yes	Use the variable directly
	no	yes	yes	Use the variable directly
	no	no	yes	Create the variable in the variables list
	no	no	no	Create the acronyms and the variable in the variables list (rarely needed)
Intermediate variables	yes	n/a	n/a	Use the variable directly
	no	n/a	n/a	Create the variable with DefVar (not needed for sin??_s)

EUROMOD output microdata

- Content manipulated in policy output_std_cc
  - Controls level at which info is outputted (e.g. individual, family or household)
  - Which variables to be included in the output
- Variables usually included:
  - All variables present in the input microdata file
  - Simulated variables (i.e. simulated taxes and benefits)
  - Standardised income lists (e.g. all benefits, all taxes)
  - (optional) non-standard income lists
  - (optional) intermediate variables
  - (optional) tax/assessment unit identification info

System function *DefOutput*

- Determines the content of the output file

	Policy		Grp/No	RO_2018	RO_2019	RO_2020	Comment
35	output_std_ro			on	on	on	DEF: STANDARD OUTPUT INDIVIDUAL LEVEL
35.1	fx DefOutput			on	on	on	
35.1.1		file		RO_2018_std	RO_2019_std	RO_2020_std	
35.1.2		vargroup		id*	id*	id*	ID variables
35.1.3		vargroup		d*	d*	d*	Demographic variables
35.1.4		vargroup		l*	l*	l*	Labour market variables
35.1.5		vargroup		y*			Market income variables
35.1.6		vargroup		p*			Public pensions variables
				b*			Benefit variables
				tin*	tin*	tin*	Tax related variables
				tsc*	tsc*	tsc*	Social contribution related variables
				tpr*	tpr*	tpr*	
35.1.11		vargroup		a*	a*	a*	Asset variables
35.1.12		vargroup		x*	x*	x*	Expenditure variables
35.1.13		VarGroup		k*	k*	k*	
35.1.14		ilgroup		il_*	il_*	il_*	Non-standardized income lists
35.1.15		ilgroup		ils_*	ils_*	ils_*	Standardized income lists
35.1.16		nDecimals		5	5	5	
35.1.17		TAX_UNIT		tu_individual_ro	tu_individual_ro	tu_individual_r	
35.1.18		UnitInfo_TU	1	tu_bsa_ro	tu_bsa_ro	tu_bsa_ro	
		UnitInfo_Id	1	HeadID	HeadID	HeadID	
		UnitInfo_TU	2	tu_bchmt_ro	tu_bchmt_ro	tu_bchmt_ro	
		UnitInfo_Id	2	HeadID	HeadID	HeadID	
		UnitInfo_TU	3	tu_family_ro	tu_family_ro	tu_family_ro	
		UnitInfo_Id	3	HeadID	HeadID	HeadID	
		UnitInfo_TU	4	tu_bcc_ro	tu_bcc_ro	tu_bcc_ro	
35.1.25		UnitInfo_Id	4	HeadID	HeadID	HeadID	

names of output files: xx_year_std

variables included in the output file

TAX_UNIT: level of aggregation

Tax assessment unit identification info

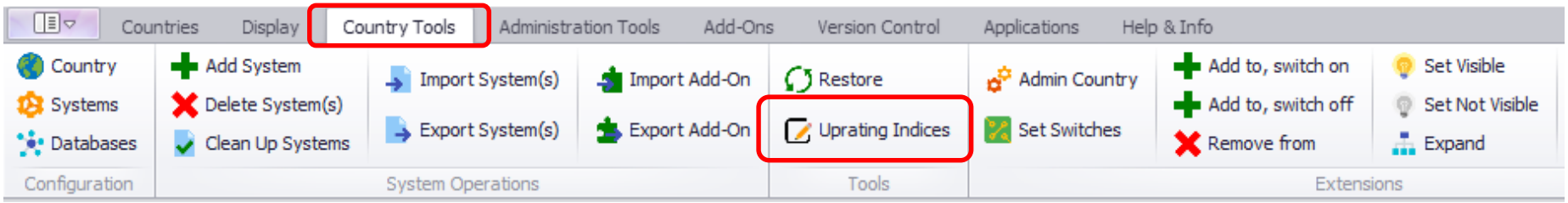
	Policy	Grp/No	EE_2018	EE_2019	EE_2020	Comment
1	**SetDefault_ee**		on	on	on	**DEF: DEFAULT VALUES FOR VARIABLES**
2	**Uprate_ee**		on	on	on	**DEF: UPRATING FACTORS**
2.1	fx Uprate		on	on	on	apply uprating factors
2.1.1	Dataset		EE_20??_??	EE_20??_??	EE_20??_??	all EE datasets (except HHoT datasets)
2.1.2	Dataset		EE_20??_???	EE_20??_???	EE_20??_???	
2.1.3	afc		$f_cpi	$f_cpi	$f_cpi	
2.1.4	afcbd		$f_cpi	$f_cpi	$f_cpi	
2.1.5	afcsa		$f_cpi	$f_cpi	$f_cpi	
2.1.6	afcsh		$f_cpi	$f_cpi	$f_cpi	
2.1.7	bedet		$f_cpi	$f_cpi	$f_cpi	
2.1.8	bedot		$f_cpi	$f_cpi	$f_cpi	
2.1.9	bsa00		$f_cpi	$f_cpi	$f_cpi	
2.1.10	bsach		0	0	0	
2.1.11	bsals		$f_cpi	$f_cpi	$f_cpi	

Uprating indices & system function *Uprate*

Why uprate incomes?

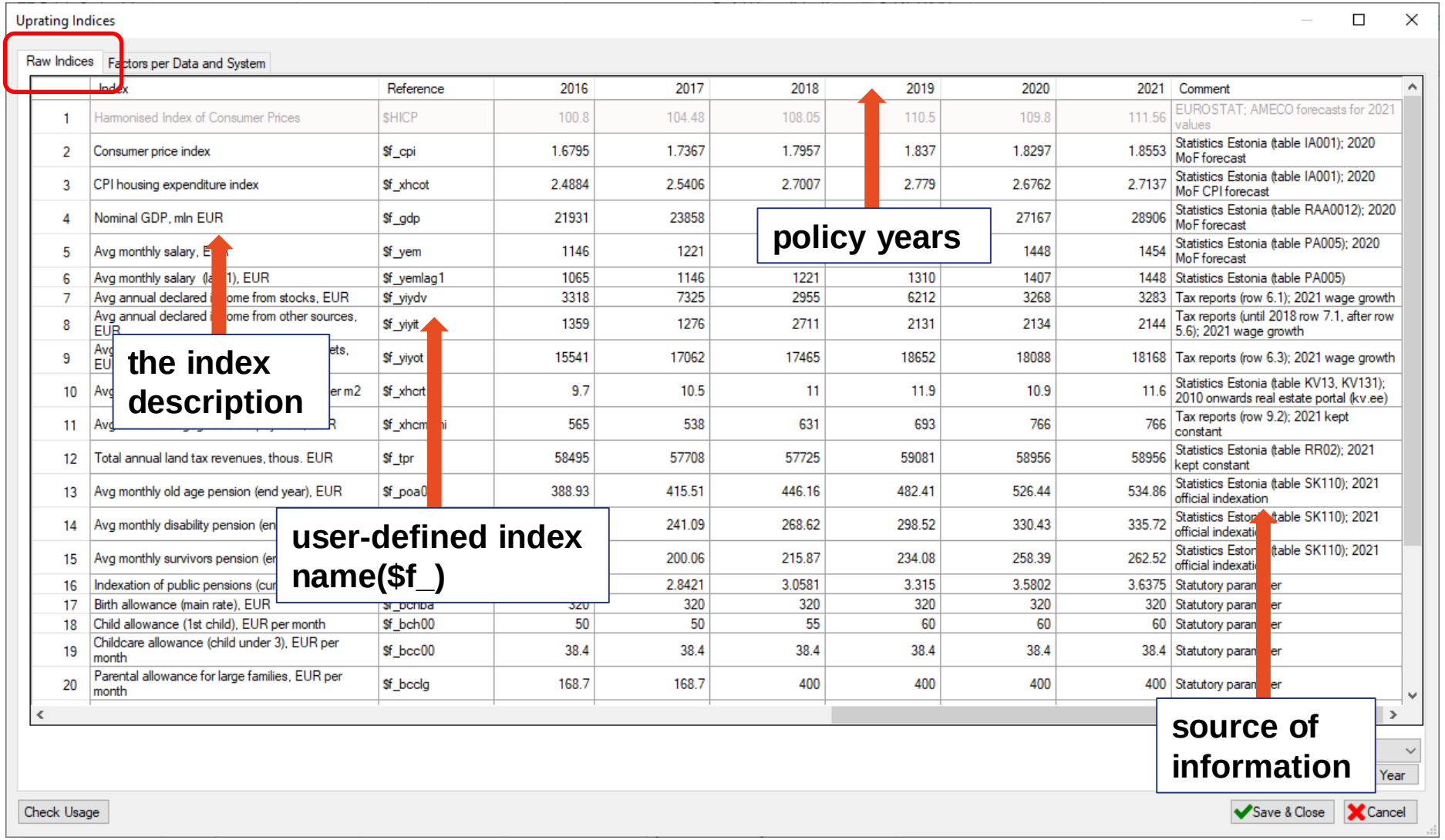
- Income reference period of input microdata and policy system may not match…
  - e.g. data availability issue: survey data become available usually with several years lag
- … so we need to adjust – i.e. *uprate* – the incomes in the input microdata to match the policy year
  - e.g. uprate 2017 input incomes to 2020 policy year
- We uprate incomes by source
  - define uprating indices in the *Uprating Indices* table
  - apply the uprating indices to the specific income variables from the input microdata, coded in the spine
  - they do not account for population changes between the data and policy year e.g. changes to the labour market

Uprating indices table



- Time-series information for the uprating indices (tab *Raw Indices*)
- EUROMOD calculates implicit uprating factors (tab *Factors per Data*)

Raw indices (time-series)



	Index	Reference	2016	2017	2018	2019	2020	2021	Comment
1	Harmonised Index of Consumer Prices	$HICP	100.8	104.48	108.05	110.5	109.8	111.56	EUROSTAT; AMECO forecasts for 2021 values
2	Consumer price index	$f_cpi	1.6795	1.7367	1.7957	1.837	1.8297	1.8553	Statistics Estonia (table IA001); 2020 MoF forecast
3	CPI housing expenditure index	$f_xhcot	2.4884	2.5406	2.7007	2.779	2.6762	2.7137	Statistics Estonia (table IA001); 2020 MoF CPI forecast
4	Nominal GDP, mln EUR	$f_gdp	21931	23858			27167	28906	Statistics Estonia (table RAA0012); 2020 MoF forecast
5	Avg monthly salary, E	$f_yem	1146	1221			1448	1454	Statistics Estonia (table PA005); 2020 MoF forecast
6	Avg monthly salary (la 1), EUR	$f_yemlag1	1065	1146	1221	1310	1407	1448	Statistics Estonia (table PA005)
7	Avg annual declared i ome from stocks, EUR	$f_yiydv	3318	7325	2955	6212	3268	3283	Tax reports (row 6.1); 2021 wage growth
8	Avg annual declared i ome from other sources, EUR	$f_yiyit	1359	1276	2711	2131	2134	2144	Tax reports (until 2018 row 7.1, after row 5.6); 2021 wage growth
9	Avg EU ets,	$f_yiyot	15541	17062	17465	18652	18088	18168	Tax reports (row 6.3); 2021 wage growth
10	Avg er m2	$f_xhcrt	9.7	10.5	11	11.9	10.9	11.6	Statistics Estonia (table KV13, KV131); 2010 onwards real estate portal (kv.ee)
11	Avg R	$f_xhcm i	565	538	631	693	766	766	Tax reports (row 9.2); 2021 kept constant
12	Total annual land tax revenues, thous. EUR	$f_tpr	58495	57708	57725	59081	58956	58956	Statistics Estonia (table RR02); 2021 kept constant
13	Avg monthly old age pension (end year), EUR	$f_poa0	388.93	415.51	446.16	482.41	526.44	534.86	Statistics Estonia (table SK110); 2021 official indexation
14	Avg monthly disability pension (en			241.09	268.62	298.52	330.43	335.72	Statistics Eston table SK110); 2021 official indexati
15	Avg monthly survivors pension (er			200.06	215.87	234.08	258.39	262.52	Statistics Eston table SK110); 2021 official indexati
16	Indexation of public pensions (cur			2.8421	3.0581	3.315	3.5802	3.6375	Statutory paran er
17	Birth allowance (main rate), EUR	$f_bchba	320	320	320	320	320	320	Statutory paran er
18	Child allowance (1st child), EUR per month	$f_bch00	50	50	55	60	60	60	Statutory paran er
19	Childcare allowance (child under 3), EUR per month	$f_bcc00	38.4	38.4	38.4	38.4	38.4	38.4	Statutory paran er
20	Parental allowance for large families, EUR per month	$f_bcclg	168.7	168.7	400	400	400	400	Statutory paran er

Factors per data and system

Uprating Indices

Raw Indices | Factors per Data and System

Dataset: EE_2019_c1 | Income Year: 2018 | Update

Index	EE_2014	EE_2015	EE_2016	EE_2017	EE_2018	EE_2019	EE_2020	EE_2021
$HICP	0.9248	0.9255	0.9329	0.967	1	1.0227	1.0162	1.0325
$f_cpi	0.9391	0.9344	0.9353	0.9671	1	1.023	1.0189	1.0332
$f_xhcot	0.9516	0.9373	0.9214	0.9407	1	1.029	0.9909	1.0048
$f_gdp	0.778	0.8012	0.8455	0.9198	1	1.0838	1.0474	1.1144
$f_yem	0.7672	0.813	0.8748	0.9321	1	1.074	1.1053	1.1099
$f_yemlag1	0.7772	0.8231	0.8722	0.9386	1	1.0729	1.1523	1.1859
$f_yiydv	1.2115	1.4406	1.1228	2.4788	1	2.1022	1.1059	1.111
$f_yiyit	0.9133	0.4666	0.5013	0.4707	1	0.7861	0.7872	0.7909
$f_yiyot	0.9641	0.9423	0.8898	0.9769	1	1.068	1.0357	1.0403
$f_xhcrt	0.7818	0.8636	0.8818	0.9545	1	1.0818	0.9909	1.0545
$f_xhcmomi	0.9683	0.916	0.8954	0.8526	1	1.0983	1.2139	1.2139
$f_tpr	1.0213	1.005	1.0133	0.9997	1	1.0235	1.0213	1.0213
$f_poa00	0.7805	0.8274	0.8717	0.9313	1	1.0812	1.1799	1.1988
$f_pdi	0.7435	0.7867	0.8295	0.8975	1	1.1113	1.2301	1.2498
$f_psu	0.7852	0.8333	0.8818	0.9268	1	1.0844	1.197	1.2161
$f_ipens	0.787	0.8366	0.8843	0.9294	1	1.084	1.1707	1.1895

- For the selected dataset, the implicit uprating factors for each system are shown, e.g.:
  - Dataset EE_2019_c1 → income referring to 2018
  - Prices ($f_cpi) increased by 3.32% between 2018 and 2021

System function *Uprate* (1)

- Defines which indices to apply on which income variables from the input microdata

	Policy	Grp/No	EE_2019	EE_2020	EE_2021	Comment
2	**Uprate_ee**		**on**	**on**	**on**	**DEF: UPRATING FACTORS**
2.1	fx Uprate		**on**	**on**	**on**	apply uprating factors
2.1.1	Dataset		EE_20??_??	EE_20??_??	EE_20??_??	all EE datasets (except HHoT datasets)
2.1.2	Dataset		EE_20??_???	EE_20??_???	EE_20??_???	
2.1.3	afc		$f_cpi	$f_cpi	$f_cpi	
2.1.4	afcbd		$f_cpi	$f_cpi	$f_cpi	
2.1.5	afcsa		$f_cpi	$f_cpi		
2.1.6	afcsh		$f_cpi	$f_cpi		
	bedet		$f_cpi	$f_cpi		
	bedot		$f_cpi	$f_cpi	$f_cpi	
	bsa00		$f_cpi	$f_cpi	$f_cpi	
	bsach		0	0	0	
	bsals		$f_cpi	$f_cpi	$f_cpi	
2.1.12	bsape		$f_bsape	$f_bsape	$f_bsape	
2.1.13	kivho		$f_cpi	$f_cpi	$f_cpi	
2.1.14	yds				1	used for validation only
2.1.15	ydses_o				1	used for validation only
2.1.16	yprro				$f_cpi	
2.1.17	yprho		$f_cpi	$f_cpi	$f_cpi	
2.1.18	xed		$f_cpi	$f_cpi	$f_cpi	
2.1.19	xcd		$f_cpi	$f_cpi	$f_cpi	
2.1.20	ypp02		$f_cpi	$f_cpi	$f_cpi	
2.1.21	ypp03		$f_cpi	$f_cpi	$f_cpi	
2.1.22	bchba		$f_bchba	$f_bchba	$f_bchba	
2.1.23	bch00		$f_bch00	$f_bch00	$f_bch00	

input data sets to which uprating applies

variables that are uprated

name of uprating factor

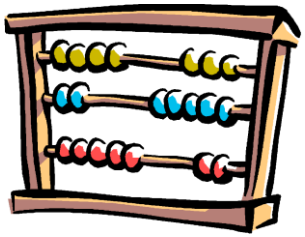
System function *Uprate* (2)

- Uprating the components of an aggregate variable

Uprate_bg		on
fx Uprate		on
dataset		BG_20??_??
yemtx		$f_yem
yemnr		$f_yem
AggVar_Name	1	yem
AggVar_Part	1	yemtx
AggVar_Part	1	yemnr
AggVar_Tolerance	1	1
ysetx		$f_yem
ysenr		$f_yem
AggVar_Name	2	yse
AggVar_Part	2	ysetx
AggVar_Part	2	ysenr
AggVar_Tolerance	2	1

- Using different uprating indices for different groups

uprate_el		on	DEF: UPRATING FACTORS
fx Uprate		on	apply uprating factors
dataset		EL_20??_??	all EL datasets
Factor_Con...	1	(lpmfc = 4)	.. workers in public enterprises
yem	1	$f_yem4	
Factor_Con...	2	(lpmfc = 5)	.. banking employees
yem	2	$f_yem5	
Factor_Con...	3	(lpmfc = 7)	.. civil servants
yem	3	$f_yem7	
Factor_Con...	4	(lpmfc = 1) | (lpmfc = 8) | (lpmfc = 9)	".. other private sector employees (IKA, liberal professions, seamen)"
yem	4	$f_yem189	
Factor_Con...	5	(lpmfc = 0) | (lpmfc = 2) | (lpmfc = 3) | (lpmfc = 6) | (lpmfc = -1)	.. other (e.g. self-employed)
yem	5	$f_yem	



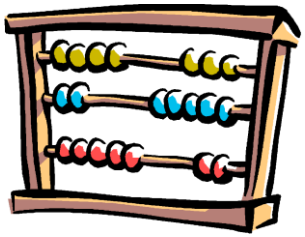
Exercise 14

Modifying uprating factors in Estonia to account for differential wage evolution

The average monthly salary in Estonia in 2018 was 1,310 EUR/month. Imagine that you have the following (hypothetical) information about the evolution of salaries between 2018 and 2021:

	Increase in relation to previous year		
Monthly salary in 2018	2019	2020	2021
Up to average	3%	3%	3%
Above average	4%	5%	6%

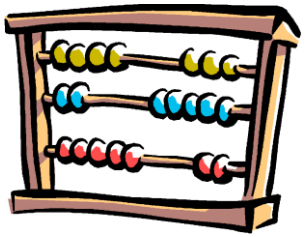
- Do the necessary changes in EUROMOD so that it reflects this different evolution of salaries, when running the 2021 system.
- Use the In-depth Analysis plugin to compare the new system with the original 2021 system [only with SILC-based input data]



Exercise 14

Steps:

- Create two new time-series in the uprating indices table (tab Raw indices) to account for the different changes in salaries between 2018 and 2021.
- Check that they correctly reflect the information in the table (tab Factors per data and system).
- Create a copy of the EE 2021 system (e.g. EE_2021_uprating).
- Modify the uprate_ee policy to account for the new information.
- Run EE_2021 and EE_2021_uprating in the In_depth Analysis plugin and analyse the fiscal and distributional impact [only with SILC-based input data]



Exercise 14

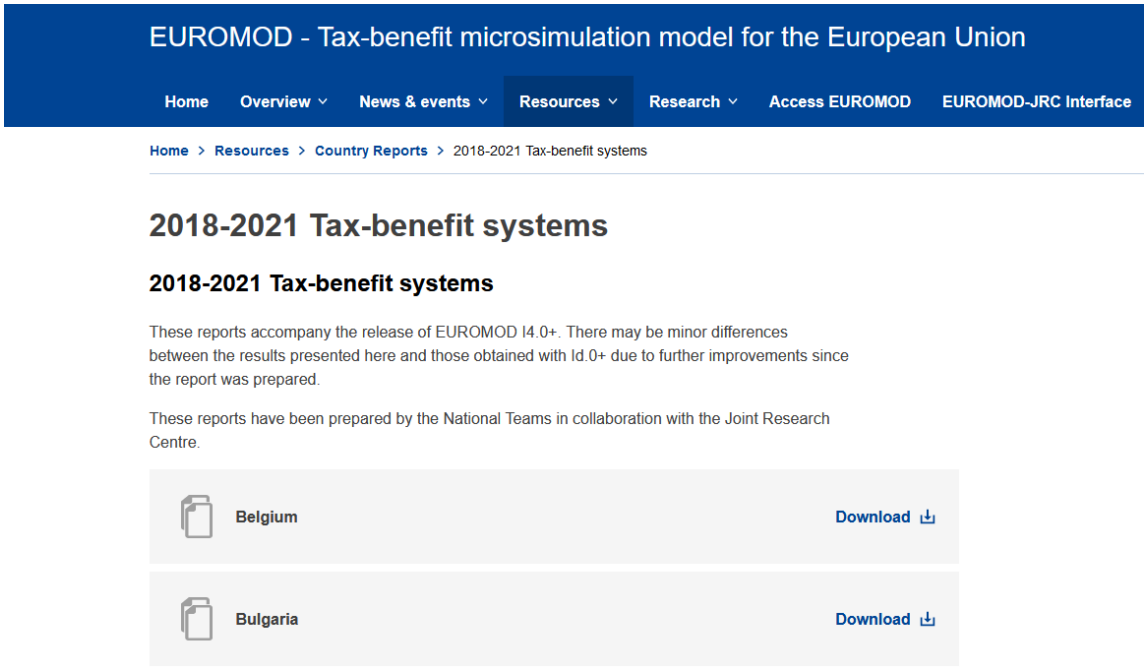
Hints:

- Before modifying any parameter in EUROMOD, compute in Excel index numbers, considering 2018 as the base (=100)
- In “Uprating indices → Raw indices” add two rows for two new indices, $f_yemlow and $f_yemhigh, and paste the results from Excel. Bear in mind that for this exercise we don’t care about what happened before 2018, so you can leave those values blank (EUROMOD will fill them with missing values).
- In the new system EE_2021_uprating go to the Uprate function in the uprate_ee policy and tell EUROMOD how to uprate yem00:
  - Using the new indices: you have to add two placeholders and two Factor_condition parameters.
  - Setting to n/a the original uprating factor used for yem.

Questions

Summary: Exercise 14

- You learned how to:
  - include new uprating factors
  - use them in the EUROMOD spine



Documentation

Documentation (1)

- Built-in help
  - EUROMOD terminology
  - Running EUROMOD and basic concepts
  - EUROMOD functions
  - Specific help for plugins, e.g. Statistics Presenter
- Data Requirement Document (DRD) - 1 per dataset
- Documentation folder:
  - EUROMOD built-in help saved in pdf
  - HHoT manual
  - Add-ons' documentation
  - What's new document
- Log folder:
  - EUROMOD version log

Documentation (2)

- Country report
  - Main document accompanying each country model
  - Provides information about:
    - tax-benefit system
    - modelling decisions and limitations
    - underlying data
    - accuracy of simulations
  - Each report covers the policy systems
    - from the income year of the latest available input data (2018 for I4.0+)
    - to the most recent policy year (2021 for I4.0+)

Documentation (3)

- Online documentation:
  - EUROMOD website: https://euromod-web.jrc.ec.europa.eu/
  - EUROMOD training material: https://euromod-web.jrc.ec.europa.eu/resources/training
  - Country Reports: https://euromod-web.jrc.ec.europa.eu/using-euromod/country-reports
  - Model documentation: https://euromod-web.jrc.ec.europa.eu/resources/model-documentation
  - Projects and publications using EUROMOD: https://euromod-web.jrc.ec.europa.eu/research

Keep in touch

https://euromod-web.jrc.ec.europa.eu/

https://euromod-web.jrc.ec.europa.eu/news-and-events/newsletters

JRC-EUROMOD@ec.europa.eu

Thank you

European Commission