EUROMOD training course - exercises

Fiscal Policy Analysis Unit
Joint Research Centre
European Commission

Course based on
EUROMOD I6.0++, software v.3.6.19
May 2024

© European Union, Institute for Social and Economic Research, University of Essex, 2021-2024

The reuse policy of the European Commission is implemented by the Commission Decision 2011/833/EU of 12 December 2011 on the reuse of Commission documents (OJ L 330, 14.12.2011, p. 39). Except otherwise noted, the reuse of this document is authorised under the Creative Commons Attribution 4.0 International (CC BY 4.0) licence. This means that reuse is allowed provided appropriate credit is given and any changes are indicated. For any use or reproduction of photos or other material that is not owned by the EU, permission must be sought directly from the copyright holders.

The JRC is deeply indebted to colleagues from ISER - University of Essex who ran EUROMOD training until 2020. A number of exercises in this manual were originally developed by them and have been successively updated by the JRC.

Contents

Exercise 1: Running EUROMOD; Using the Statistics Presenter 4

Exercise 2: Implementing a reform to the model of Finland; Using the Statistics Presenter to analyse its impact on the income distribution 13

Exercise 3a: Reforming Child Allowance in Estonia 19

Exercise 3b: Reforming Child benefit in Greece 27

Exercise 4: Introducing a supplement, withdrawn with earnings, to the Child Allowance in Estonia . 31

Exercise 5a: Introducing a zero tax band to the flat income tax in Bulgaria 36

Exercise 5b: Reforming income tax in EL (using functions *SchedCalc* and *DefConst*) 40

Exercise 6: Introducing a benefit cap in Denmark 47

Exercise 7: Reforming PT Child Benefit by narrowing its age condition 53

Exercise 8: Introducing the Belgian social insurance contributions for pensioners & survival pensioners in Portugal 58

Exercise 9: Reforming the health tax in Denmark 63

Exercise 10: New means-tested income support for families with children in education in Croatia 66

Exercise 11: Introducing an In-Work Benefit in Cyprus 70

Exercise 12: Revenue-neutral reform in Simpleland 74

Exercise 13: Implementing an EU child benefit using an AddOn 80

Exercise 14: Adding a new dataset for Austria, specifying uprating factors and setting switches 88

Exercise 15: Modifying uprating factors in Austria to account for differential wage evolution 94

Exercise 16: Introducing a benefit for secondary education in Simpleland 101

Exercise 1: Running EUROMOD; Using the Statistics Presenter

The aim of this exercise is to learn how to run EUROMOD, produce micro-output and learn how to use the Statistics Presenter tool with Default and MultiSystem options. The exercise also introduces some basic options such as (un)hiding systems and expanding policies.

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.

Description:

The **Statistics Presenter Tool** provided with EUROMOD allows computing a range of commonly used indicators and statistics for analysing EUROMOD micro-output. Amongst others, it computes basic poverty and inequality indices for the overall population and for selected groups; distribution of household income, taxes and benefits by income group; information on households by income group; and government revenue and expenditure.

The computations are based on EUROMOD standard micro-output, which comprises variables for disposable income and its components (original income, benefits, taxes and social insurance contributions).

Therefore, the first step is to produce this micro-output for the countries you would like to analyse.

Solution and further information

Task 1: Producing output micro-data for Austria for 2019

Run EUROMOD in order to produce micro-data output for Austria. The model can be run for one country at the time or the user can select several countries (and systems) for running all of them at once.

- In the Countries tab on the ribbon bar press the "Run EUROMOD" button. This will bring up a dialogue box.
- In the dialogue box, select Austria by pressing on the AT flag (if not already selected). This will display the available systems for Austria (AT) (see Figure 1.1).
- Select the AT_2019 system by clicking the corresponding boxes.
- For this system, select *AT_training_data* as the dataset which EUROMOD should use (see Figure 1.2).
- Check (and if necessary, modify) the output path where EUROMOD will be writing the output files.
- Click on the Run button in the dialogue box.

Figure 1.1: EUROMOD ribbon – all countries displayed

Figure 1.2: Run EUROMOD dialogue, selecting Austria 2019 system and "AT_training_data" for dataset

To see the Run or the Error logs (if there are any) click on the corresponding buttons.

Figure 1.3: Running AT_2019 system with AT_training_data dataset – in progress

Figure 1.4: Running AT_2019 system with AT_training_data dataset – finished with no errors

Well done! You have just run EUROMOD for the first time!

What happened is that EUROMOD simulated this country and year's policy system using 'AT_training_data' as the input data. We will come back to this and have a look at the outputs of the run in Task 3. Let's try to do the same now for multiple countries and years.

Note: If you are stuck, a good place to look for help, is the detailed 'Help' section within the EUROMOD software. It is located in the ribbon menu under 'Help & Info' > Help.

Task 2: Running 2020 systems for Austria, Finland, Italy and Spain

Click on the 'Run EUROMOD' button. Click on the flags for Austria (AT), Finland (FI), Italy (IT) and Spain (ES). A list of all available systems will appear (Figure 1.5).

Figure 1.5: EUROMOD run dialogue – selected countries: Austria, Finland, Italy and Spain

Click on the "View/Filter/Add-Ons" tab and using the 'Filters system' field filter the 2020 systems for these countries (Figure 1.6).

Figure 1.6: Advanced Run settings – filtering systems

Then type '*_training' in the 'Filter Datasets' field to select the country-specific 'training_data' as the dataset (Figure 1.7). Make sure all boxes on the left of the systems are selected (ticked) (Figure 1.8) to ensure that they will be included in the EUROMOD run.

Figure 1.7: Advanced Run settings – filtering datasets and systems

Figure 1.8: Advanced Run settings – selecting systems to run

Click on the 'Main' tab to return to the main 'Run EUROMOD' window, then click on 'Run' button to run the systems (Figure 1.9).

Figure 1.9: Running the selected systems

The run window will appear as before and all systems should run without any problems (status: finished).

Figure 1.10: EUROMOD Run window – all systems and data combinations finished without errors

Task 3: Using the Statistics Presenter with the Default option

Let's now explore the outputs of the different runs that we set up in tasks 1 and 2. For each run, EUROMOD creates an output file, which includes the simulated values in addition to non-simulated variables. These are stored in the Output folder within your project folder.

An easy way to explore the output files is to use the Statistics Presenter ('Applications'> 'Statistics Presenter'):

- To access the Statistics Presenter, click on the Applications tab. Click on the EUROMOD Statistics button (Figure 1.11)
- Click on the Statistics Presenter button to launch the tool (Figure 1.12)

Figure 1.11 Opening the Statistics Presenter

Figure 1.12 Opening the Statistics Presenter

There are different types of options. Select the *Default* option and click ok (Figure 1.13)

Figure 1.13: Statistics Presenter: generating a Default report view

In the window that will appear (Figure 1.14), use Ctrl (select multiple individual elements) or Shift (select a range of multiple elements) to select the relevant EUROMOD output files that you would like to use and click OK. Note that the path where the output files are stored can be changed, if needed and that the order you select the files will be reflected on the order the systems will appear in the report.

Let's first compare the 2019 and 2020 Austrian systems like below:

- click on *at_2019_std.txt* first
- holding CTRL, click on *at_2020_std.txt* next

Figure 1.14: Statistics Presenter, Default report view: selecting systems to output

Click on OK.

A number of tables, as shown in Figure 1.15, will be generated, for each of the two systems. You can export the statistics in an excel file if you would like to further edit the tables, by clicking on the "disk" icon.

Figure 1.15: Statistics Presenter. Default report view; exporting to external file

Go through the different systems tabs (at_2019, at_2020) and different tables (top tab) to see the different types of statistics.

Alternatively, you can use the Statistics Presenter with the "MultiSystem" option to analyse the output micro-data. The next task explains how.

Task 4: Using the Statistics Presenter with the MultiSystem option

Open the Statistics Presenter and select the *MultiSystem* option (Figure 1.16). For this task, we will analyse the output micro-data for 2020 for Austria, Finland, Italy and Spain.

Select the output files for the four countries as in Figure 1.17. The output report should look like the ones in Figure 1.18 and Figure 1.19. Note that the different systems are now displayed next to each other (for easy comparison).

Figure 1.16: Statistics Presenter, Multisystem report

Figure 1.17: Statistics Presenter, Multisystem report: Selecting system output files

Figure 1.18: Statistics Presenter. Multiple system report view – fiscal overview

Summary Statistics - Multiple Systems

Summary Statistics - Multiple Systems

Results for AT_2019, AT_2020, ES_2020, FI_2020, IT_2020

1. Fiscal Overview | 2. Distributional | 3. Inequality and Poverty | Metadata

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	AT_2019	AT_2020	ES_2020	FI_2020	IT_2020
Total market incomes	**133,094.74**	**133,094.74**	**360,172.06**	**70,015.29**	**579,017.60**
... income from (self) employment	133,094.74	133,094.74	360,172.06	70,015.29	579,017.60
... other sources	0.00	0.00	0.00	0.00	0.00
Government revenue through taxes and social insurance contributions	**84,986.48**	**82,922.92**	**204,135.92**	**39,239.87**	**399,579.13**
... direct taxes	24,465.89	22,466.06	61,191.04	20,328.27	175,338.53
... employee social insurance contributions	20,912.31	21,014.07	22,695.44	6,206.02	46,452.33
... self-employed social insurance contributions	3,027.87	2,921.47	8,929.29	1,136.85	20,771.79
... other social insurance contributions	2,577.64	2,577.64	4,457.34	0.00	0.00
... employer social insurance contributions (not part of disposable income)	34,002.76	33,943.67	106,862.81	11,568.73	157,016.47
Credited social insurance contributions (not part of disposable income)	**2,011.59**	**2,011.59**	**22,381.65**	**0.00**	**0.00**
Government expenditure on social transfers	**69,871.51**	**71,208.31**	**215,013.07**	**44,131.32**	**320,733.71**
by target group					
... unemployment benefits	8,654.53	9,066.07	56,902.57	4,830.91	50.77
... family and education benefits	4,687.10	5,323.72	1,196.72	1,641.23	7,376.55
... social assistance and housing benefits	5,986.33	6,274.98	6,964.18	7,123.33	24,005.12
... pensions, health and disability benefits	50,543.55	50,543.55	149,949.60	30,535.88	289,301.26
... firms	0.00	0.00	0.00	0.00	0.00
by benefit design					
... means-tested non-pension benefits	6,306.47	6,595.11	7,492.06	7,446.50	33,747.65
... non-means-tested non-pension benefits	13,021.50	14,069.65	57,571.41	6,148.97	334.15
... pensions	50,543.55	50,543.55	149,949.60	30,535.88	286,651.89
... firms subsidies	0.00	0.00	0.00	0.00	0.00

Figure 1.19: Statistics Presenter. Multiple system report view – inequality and poverty

Summary Statistics - Multiple Systems

Summary Statistics - Multiple Systems

Results for AT_2019, AT_2020, ES_2020, FI_2020, IT_2020

1. Fiscal Overview | 2. Distributional | 3. Inequality and Poverty | Metadata

3. Inequality and Poverty

3.1 Basic Inequality Indices

	Gini AT_2019	Gini AT_2020	Gini ES_2020	Gini FI_2020	Gini IT_2020	S80/S20 AT_2019	S80/S20 AT_2020	S80/S20 ES_2020	S80/S20 FI_2020	S80/S20 IT_2020
Original Income	0.5083	0.5083	0.4800	0.5306	0.5108	6.9100	6.9096	4.3555	6.3518	4.4034
Original Income after Taxes/SIC	0.5122	0.5093	0.4883	0.5225	0.5173	5.0863	5.0087	3.8879	5.0477	3.1154
Original Income incl. Public Pensions after Taxes/SIC	0.2901	0.2883	0.3164	0.3078	0.2781	5.0427	4.9903	6.2717	6.3999	5.5045
Disposable Income	0.2191	0.2145	0.2727	0.2060	0.2304	3.0431	2.9662	4.6215	2.6851	3.3754

3.2.a Basic Poverty Indices

	Value for AT_2019	Value for AT_2020	Value for ES_2020	Value for FI_2020	Value for IT_2020
At-risk-of-poverty rate	16.47 %	14.48 %	20.89 %	5.09 %	18.01 %
Median at-risk-of-poverty gap	10.16 %	11.00 %	36.36 %	6.57 %	15.91 %
Fixed Poverty line (monthly)	1,256.70	1,285.02	837.00	1,116.79	891.67

3.2.b At-risk-of-poverty rates by household type

	Rate for AT_2019	Rate for AT_2020	Rate for ES_2020	Rate for FI_2020	Rate for IT_2020
One adult < 65, no children	39.59 %	39.59 %	44.23 %	0.00 %	52.39 %
- Female adult	39.59 %	39.59 %	44.23 %	0.00 %	52.39 %

Exercise 2: Implementing a reform to the model of Finland; Using the Statistics Presenter to analyse its impact on the income distribution

The aim of this exercise is to learn how to create (add) a new reform system and how to use the Statistics Presenter – Baseline/Reform option – to analyse the impact on the income distribution of the reform.

Steps:

1. Create a new system in Finland based on the 2021 system and name it FI_2021_E2;

2. Increase the Child Benefit amount for the first child from €94.88 to €100 per month;

3. Run EUROMOD to produce micro-outputs for these two systems (2021 and 2021reform);

4. Analyse the results using the Statistics Presenter with Baseline/Reform option.

Description:

You are asked to implement a simple reform to FI_2021 policy system. You will increase the Child Benefit amount for the first child from €94.88 per month to 100€ per month.

In more detail, you will make a copy of the FI 2021 system and call it FI_2021_E2. Then, you will modify the reform system FI_2021_E2 as follows:

- Go to the Child Benefit policy (policy name *bch_fi* in row 40) and open the functions DefConst (row 40.1) and BenCalc (row 40.3)
- Change the parameter value of $bch1 (row 40.1.1) from #94.88 (which is equal to €94.88 per month) to *100#m* (€100 per month).
- Save your changes.

When you are done with your changes to the reform system, run the model for the baseline system FI_2021 and reform system FI_2021_E2. Use the *Statistics Presenter* – Baseline/Reform option – to analyse the distributional effect of the reform.

Hints:

- The variable for the simulated child benefit is *bch_s* (b: benefit; ch: child; _s: simulated)
- The suffix *#m* indicates that the benefit amount is paid on a monthly basis. By default, EUROMOD treats all values of parameters and variables as monthly amounts. The suffix #m is therefore not strictly required and could be omitted. Defining the parameter *$bch1* as *100*, instead of *100#m*, would therefore lead to exactly the same results.
- You can also assign amounts on a weekly basis (with suffix #w) or on a yearly basis (suffix #y). In such case, you need to specify the suffix.

Solution and further information

Task 1: Creating a new system

Open the FI model, by clicking on the Finnish country flag. To work more easily with the FI_2021 system, hide the rest of the systems, by right-clicking on the system name (FI_2021), selecting "*move to hidden system box*" and then selecting "*all systems but selected*" (see Figure 2.1).

Figure 2.1: Hiding a system

Add a new system. There are two ways of doing this, either:

1) right-click on the system heading FI_2021 and select the option 'Copy/Paste System' or
2) click on the button *Add System* in the Country Tools tab.

Figure 2.2: Adding a reform system using the copy/paste option

Let's use option 1). In the window that appears (Figure 2.3), specify a name for the reform system (type e.g. *FI_2021_E2*). The new systems should appear on the right of the FI_2021 system (Figure 2.4). Note that columns can be expanded (by dragging the lines) if you cannot see all the contents.

Figure 2.3: Giving a name to the reform system

Figure 2.4: Baseline and new reform system

	Policy	Gr...	FI_2021	FI_2021_E2	Comment
1	SetDefault_fi		on	on	DEF: SET DEFAULT VALUES
2	uprate_fi		on	on	DEF: UPRATING FACTORS
3	ConstDef_fi		on	on	DEF: CONSTANTS
4	ilsdef_fi		on	on	DEF: STANDARD INCOME CONCEPTS
5	ilsUDBdef_fi		on	on	DEF: UDB INCOME CONCEPTS
6	InitVars_fi		on	on	DEF: Initialization of
7	IlDef_fi		on	on	DEF: NON-STANDARD INCOME CONCEPTS
8	Random_fi		on	on	DEF: Random assignment
9	TransLMA_fi		off	off	DEF: Modelling labour market transitions (DO NOT SWITCH ON; ONLY WORKS WITH THE LMA
10	tudef_fi		on	on	DEF: ASSESSMENT UNITS
11	yem_fi		switch	switch	DEF: Minimum wage
12	neg_fi		on	on	DEF: Recode negative self-employment income to zero
13	ysecomp_fi		on	on	BEN: compensation scheme Covid-19 for the self-employed (ONLY WORKING WITH LMA ADD-ON)

Task 2: Implementing the reform

You need to increase the Child Benefit amount for the first child from €94.88 per month to 100€ per month. To do that, we will need to edit the 'Child Benefit' policy on the country spine.

The Child Benefit is implemented in the policy called *bch_fi*. The amount for the first child is defined in the constant *$bch1* in the function *DefConst* (row 40.1.1). The constant *$bch1* is then used in the function BenCalc (row 40.3.17), which calculates the benefit amounts per number of children. If you hover your mouse over the *$bch1* constant in row 40.3.17 you can see its values ("$bch1 = 94.88#m" in Figure 2.5).

Figure 2.5: Hovering over the $CBFirst constant to see the current value

40	bch_fi		on	on	BEN: Child benefit
40.1	fx DefConst		on	on	Constants for Child Care Benefits
40.1.1	$bch1		94.88#m	94.88#m	Child benefit for the 1st child
40.1.2	$bch2		104.84#m	104.84#m	Child benefit for the 2nd child
40.1.3	$bch3		133.79#m	133.79#m	Child benefit for the 3rd child
40.1.4	$bch4		163.24#m	163.24#m	Child benefit for the 4th child
40.1.5	$bch5		182.69#m	182.69#m	Child benefit for the 5th child & subsequent
40.1.6	$bchsup		63.30#m	63.30#m	Child benefit supplement for a lone parent
40.2	fx Elig		on	on	Child benefit: eligibility
40.3	fx BenCalc		on	on	Child benefit: amount
40.3.1	who_m...		all	all	
40.3.2	comp_c...	1	IsNtoMchild#1=1	IsNtoMchild#1=1	
40.3.3	#_N	1	1	1	
40.3.4	#_M	1	1	1	
40.3.5	comp_c...	2	IsNtoMchild#2=1	IsNtoMchild#2=1	
40.3.6	#_N	2	2	2	
40.3.7	#_M	2	2	2	
40.3.8	comp_c...	3	IsNtoMchild#3=1	IsNtoMchild#3=1	
40.3.9	#_N	3	3	3	
40.3.10	#_M	3	3	3	
40.3.11	comp_c...	4	IsNtoMchild#4=1	IsNtoMchild#4=1	
40.3.12	#_N	4	4	4	
40.3.13	#_M	4	4	4	
40.3.14	comp_c...	5	IsNtoMchild#5=1	IsNtoMchild#5=1	
40.3.15	#_N	5	5	5	
40.3.16	#_M	5	99	99	
40.3.17	comp_p...	1	$bch1	$bch1	Amount for the 1st child
40.3.18	comp_p...	2	$bch2	$bch2	Amount for the 2nd child
40.3.19	comp_p...	3	$bch3	$bch3	Amount for the 3rd child
40.3.20	comp_p...	4	$bch4	$bch4	Amount for the 4th child
40.3.21	comp_p...	5	$bch5	$bch5	Amount for the 5th & subsequent children
40.3.22	output_...		bch_s	bch_s	Child benefit (without lone parent supplement)
40.3.23	TAX_U...		tu_family17_fi	tu_family17_fi	

In the reform system FI_2021_E2, change the parameter value of *$bch1* from 94.88#m to *100#m* (€100 per month).

Figure 2.6: New benefit amount for the first child

bch_fi		on	on	**BEN: Child benefit**
fx DefConst		on	on	Constants for Child Care Benefits
$bch1		94.88#m	100#m	Child benefit for the 1st child
$bch2		104.84#m	104.84#m	Child benefit for the 2nd child
$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

Save your changes (Ctrl+S).

Task 3: Produce and analyse the micro-data output

Run the model to produce micro-data output for the baseline FI_2021 and reform FI_2021_E2 systems using FI_*training_data* (see exercise 1 if you cannot remember how to do this) (Figure 2.7). Use the Statistics Presenter – option Baseline/Reform (Figure 2.8) – to produce a report (Figure 2.9) and analyse differences in government revenues and income poverty due to the changes in the Child Benefit (Figure 2.10).

Figure 2.7: Producing micro-data output for the baseline and reform systems

Figure 2.8: Opening Statistics Presenter – Baseline/Reform option

Figure 2.9: Selecting the relevant output files with the Statistics Presenter

Figure 2.10: Statistics Presenter. Baseline/Reform report view – mean household income by decile

2.1.a Mean household income (monthly) by decile groups

	FI_2021 (base)	FI_2021_E2	Difference to base	% Difference to base
Decile 1	1,248.48	1,248.82	0.33	0.03 %
Decile 2	1,716.76	1,717.48	0.72	0.04 %
Decile 3	2,016.92	2,017.61	0.69	0.03 %
Decile 4	2,529.37	2,530.41	1.03	0.04 %
Decile 5	3,032.09	3,033.39	1.31	0.04 %
Decile 6	3,378.66	3,379.97	1.30	0.04 %
Decile 7	3,722.91	3,724.11	1.20	0.03 %
Decile 8	4,285.10	4,286.31	1.20	0.03 %
Decile 9	5,015.46	5,016.59	1.13	0.02 %
Decile 10	7,914.23	7,915.16	0.93	0.01 %
All	**3,283.44**	**3,284.38**	**0.94**	**0.03 %**
Poor	1,296.92	1,296.46	-0.46	-0.04 %

Exercise 3a: Reforming Child Allowance in Estonia

The aim of this exercise is to introduce functions Elig and ArithOp. It also introduces some basic options such as renaming and deleting systems, search and replace and queries.

Steps:

1. Create a new system in Estonia based on the 2012 system and name it EE_2012_E3;

2. 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;

3. Run EUROMOD to produce micro-outputs for these two systems (2012 and 2012_E3);

4. Analyse the results.

Description:

The aim of this exercise is to perform a reform of the Child Allowance (CA) benefit in Estonia in 2012. In that year, the CA was a monthly universal non-means-tested benefit paid to families with children below an age limit. The amount per child was two times the Child Allowance Rate (CAR equal to 9.59 EUR), resulting in 19.18 EUR per month for the first and second child, and 6 x CAR for the third and any consequent child.

Your task is to add a means-tested supplement to the universal CA using functions *Elig* and *ArithOp*.

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*.

Solution and further information

Task 1: Creating a new system

When our task is to create (add) a new system, we should first think about which existing system the new system should be based on. In the case where we need to create a system for a new year, it is most likely that we will use the previous year as a starting point. If we want to change/modify an existing system year (but show it and analyse it as a separately), we will use that original system as a base. How do we do that?

Our first step is to open Estonia (country acronym EE) *(see exercise 1 for more information on how to load a country system)*. Hide all other systems except for EE_2012. If we go to the tab *Country Tools* in the ribbon bar, we can use the option *Add System* (Figure 3a.1) to add a new system based on an existing base system. Click on the 'Add System'. You will get a dialog box with base systems for the country that we are working with (i.e. currently active), in this case Estonia. As you can see, we are only allowed to choose one of them. In our case, it will be the 2012 system (Figure 3a.2).

Figure 3a.1: Country Tools menu

Figure 3a.2: Selecting the Base system

Rename the new system 'EE_2012_E3' (Figure 3a.3). Since we do not need other systems, we will hide all but these two (EE_2012 and EE_2012_E3).

Figure 3a.3: Renaming the System

You could also do the same by going to the system that we want to copy/paste/delete, right-click on it. This provides several options (Figure 3a.4). You can rename the system using the option *Rename System*.

Figure 3a.4: Creating a copy of the existing system and renaming it

Task 2: Implementing the CA reform

After we have created a new system, our next task in implementing the reform is to find the Child Allowance policy in the policy spine. The easiest way to do this is to use command search and focus on the column *Comment* (Figure 3a.5). This should bring up the Child Allowance policy **bch00_ee.**

Figure 3a.5: Locating the Child Allowance policy

To see how it is calculated, we should use the expand option (click on the little triangle on the left of the policy) – Figure 3a.6. Functions with ***n/a*** (= not applicable) are not used for the calculations so we can collapse them. The Child Allowance amount is EUR 19.18 per month (twice the Child Allowance Rate of EUR 9.59) for the first and second child and EUR 57.54 per month (six times the Child

Allowance Rate) for the third and any other child. These amounts are defined as constants in the policy *ConstDef_ee*. You can hover over the constant names to see their values.

Figure 3a.6: Exploring the bch00_ee (Child Allowance) policy

Policy	Grp/No	EE_2012	EE_2012_E3	Comment
bch00_ee		**on**	**on**	**BEN: child allowance (lapsetoetus)**
BenCalc		**on**	**on**	monthly benefit
comp_cond	1	IsNtoMchild#1	IsNtoMchild#1	1) for the first child
#_N	1	1	1	
#_M	1	1	1	
comp_perElig	1	$CB_Ch1	$CB_Ch1	
comp_cond	2	IsNtoMchild#2	IsNtoMchild#2	2) for the second child
#_N	2	2	2	
#_M	2	2	2	
comp_perElig	2	$CB_Ch2	$CB_Ch2	
Comp_Cond	3	IsNtoMchild#3	IsNtoMchild#3	3) for the third and each subsequent child
#_N	3	3	3	
#_M	3	99	99	
Comp_perElig	3	$CB_Ch3plus	$CB_Ch3plus	
output_var		bch00_s	bch00_s	
TAX_UNIT		tu_CBfamily_ee	tu_CBfamily_ee	children up to 16 years and 17-19 years if studying at basic, upper secondary or vocational school (põhik. baasil). From 2023, children until 19. If studying, then including 19.

Defining the Eligibility conditions

In order to give a supplement to families who have at least two children and satisfy the income test, we need to introduce an eligibility function (*Elig*) to define the families entitled to receive the supplement. Right-click on the function or a parameter name in the *BenCalc* function in *bch00_ee* to activate the context menu. Select 'Add Function After' and select function *Elig*. Note that via a 'hover over' tooltip, EUROMOD can provide a short explanation of the function (Figure 3a.7).

Figure 3a.7: Adding a new Elig function after the first BenCalc

This creates the new function with its two compulsory parameters - *Elig_Cond* and *TAX_UNIT* - with default values *n/a*. We should set our new function as *on* for the reform system (EE_2012_E3). The next very important thing is to choose the appropriate tax unit. In our case it will be *tu_CBfamily_ee (tax unit applicable to the child benefit family)* so we should click on the grey arrow and select it from the drop-down list (Figure 3a.8).

Figure 3a.8: Selecting the tu_CBfamily_ee tax unit

Policy		Grp/No	EE_2012	EE_2012_E3	Comment
	comp_cond	2	IsNtoMchild#2	IsNtoMchild#2	2) for the second child
	#_N	2	2	2	
	#_M	2	2	2	
	comp_perElig	2	$CB_Ch2	$CB_Ch2	
	Comp_Cond	3	IsNtoMchild#3	IsNtoMchild#3	3) for the third and each subsequent child
	#_N	3	3	3	
	#_M	3	99	99	
	Comp_perElig	3	$CB_Ch3plus	$CB_Ch3plus	
	output_var		bch00_s	bch00_s	
	TAX_UNIT		tu_CBfamily_ee	tu_CBfamily_ee	children up to 16 years and 17-19 years if studying at basic, upper secondary or vocational school (põhik. baasil). From 2023, children until 19. If studying, then including 19.
fx	Elig		n/a	on	
	Elig_Cond		n/a	n/a	
	TAX_UNIT		n/a	n/a	
fx	DefVar		n/a	tu_CBfamily_ee	define temporary variables
fx	BenCalc		n/a	tu_bma_ee tu_bchba_ee tu_tscerfamily_ee	increase from 1 July (2007, 2013) and non means-tested part for bcha00 (2013-2017)
fx	ArithOp		n/a	tu_ITmarried_ee tu_ITfamily_ee	1) policy as of 30th June
fx	ArithOp		switch	n/a	2) full year average

The next step is to create the eligibility condition. The supplement should be restricted to families i) with at least two children and ii) with earnings below 200 EUR per month. First, instead of generating a variable which contains the number of dependent children in each family, we can use the already-defined query *nDepChildrenInTu*. Queries contain the result of ready-made calculations: go to tab *Help&Info* and search for *Queries* for a description (Figure 3a.9). For a description of the query *nDepChildrenInTu* see Figure 3a.10. Second, the variable for earnings is called *yem*.

Figure 3a.9: EUROMOD Help for queries

Figure 3a.10 EUROMOD Help entry for nDepChildrenInTu Query

nDepChildrenInTu#x	Returns the number of dependent children in the assessment unit who fulfil dag >= parameter #_AgeMin and dag <= parameter #_AgeMax. For being counted as dependent child the query IsDepChild must apply.	#_AgeMin; optional #_AgeMax; optional	nDepChInTu nDepChildrenInTaxunit nDepChInTaxunit

So, we should type in *Elig_Cond*: yem < 200#m & nDepChildrenInTU >= 2

(*#m* = per month). By doing this we have formulated a means test based on earnings below 200 EUR per month and number of dependent children equal to at least 2. Note that the means test is at the family level, i.e. the unit of assessment is tu_CBfamily_ee (Figure 3a.11).

Figure 3a.11: The modified new Elig function

fx Elig		n/a	on
	Elig_Cond	n/a	yem<200#m & nDepParentsInTu>=2
	TAX_UNIT	n/a	tu_CBfamily_ee

Calculating the benefit amount: The other part of our task is to calculate the benefit amount of 1800 EUR per month. So, we need an arithmetic function *ArithOp after* the function *Elig*:

- Right-click on function *Elig => Add Function After => ArithOp*. This creates the new function with its three compulsory parameters: *Formula*, *Output_var* and *TAX_UNIT* with default values *n/a*.
- Set the function to *on* for the reform system (EE_2012_E3).
- Set the value of the *Output_Var* to the simulated benefit bch00_s.
- Define the tax unit: it will be the same as in the previous function - tu_CBfamily_ee.
- Set the value of the parameter *Formula* from n/a to 1800#m (1,800EUR per month - the amount of the supplement).

The important thing to remember is that we need to assign the benefit only to families who satisfy the eligibility condition. This is done by creating an interaction between the eligibility function (*Elig*) and the arithmetical function (*ArithOp*) by using the parameter *Who_Must_Be_Elig*. How do we do that?

Point the cursor at the function *ArithOp*, then right-click on the function or a parameter name in *ArithOp* and select *Show Add Parameter Form*; alternatively use the shortcut Ctrl+A (Figure 3a.12).

Figure 3a.12: Showing the Add Parameter Form

In the *Add Parameters* list we need to select the parameter *Who_Must_Be_Elig* and click the *Add* button (Figure 3a.13). You can drag the new parameter in order to have it at the beginning of the function. If EUROMOD warns you with "Please note that the action will have effect on the hidden systems as well", click ok. Set its value to *one* or any of the other values (all, all_adults, one_adult) – all will give the same result as the eligibility condition is at the family unit and takes, in this case, the same value for each member of the unit (Figure 3a.14).

Figure 3a.13: Inserting the who_must_be_elig parameter

Add Parameters

ArithOp (order: 4) in policy bch00_ee

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.

☑ Show Common Parameters
☑ Show Footnote Parameters

Description (F5) | Summary (F6) | Add | Close

Figure 3a.14: Adding value "one" for the who_must_be_elig parameter

bch00_ee		on	on	**BEN: child allowance (lapsetoetus)**
BenCalc		on	on	monthly benefit
Elig		n/a	on	
Elig_Cond		n/a	yem<200#m & nDepChildrenInTU>=2	
TAX_UNIT		n/a	tu_CBfamily_ee	
ArithOp		n/a	on	
Formula		n/a	1800#m	
Who_Must_...		n/a	one	
Output_Var		n/a	bch00_s	
TAX_UNIT		n/a	tu_CBfamily_ee	

Next, we will need to replace parameter *Output_Var* with parameter *Output_Add_Var*, in order to add the supplement to (rather than replace the value of) the benefit calculated with the first *BenCalc* function. To do this, simply left-click on the parameter *Output_Var* and you will be asked if you would like to replace it (Figure 3a.15). Your reform is now ready to run as in Figure 3a.16.

Figure 3a.15: Replacing parameter Output_Var with parameter Output_Add_Var

ArithOp		n/a	on	
Formula		n/a	1800#m	
Who_Must_...		n/a	one	
Output_Var		n/a	bch00_s	
TAX_UNIT		n/a	tu_CBfamily_ee	
DefVar				define temporary variables
BenCalc				increase from 1 July (2007, 2013) and non means-tested part for bcha00 (2013-2017)
ArithOp				1) policy as of 30th June
ArithOp				2) full year average
bchlg_ee				**BEN: allowance for families with 3+ children (kolme- ja enamalapselise pere toetus)**

EUROMOD - Request

Replace 'Output_Var' by 'Output_Add_Var'?

OK | Cancel

Figure 3a.16: The complete implementation

fx Elig		n/a	on	eligibility for supplement
	Elig_Cond	n/a	yem<200#m & nDepChildrenInTU>=2	
	TAX_UNIT	n/a	tu_CBfamily_ee	earnings should be less than EUR 200 per month and there should be at least 2 children in the family
fx ArithOp		n/a	on	supplement
	Formula	n/a	1800#m	amount of EUR 1800 per month
	Who_Must_...	n/a	one	
	Output_Add...	n/a	bch00_s	add amount to the results of the previous function
	TAX_UNIT	n/a	tu_CBfamily_ee	

Save your changes.

Task 3: Producing the micro-output

Run EUROMOD for the baseline EE_2012 and reform EE_2012_E3 systems using EE_*training_data*.

Task 4: Analyse the results

Use the *Statistics Presenter* (Baseline/Reform option) to analyse the fiscal and distributional impact of the reform.

Figure 3a.17: Budgetary and distributional impact

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	EE_2012 (base) (annual)	EE_2012_E3 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**8,728.13**	**8,728.13**	**0.00**	**0.00 %**
... income from (self) employment	8,728.13	8,728.13	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**5,133.70**	**5,133.70**	**0.00**	**0.00 %**
... direct taxes	1,694.55	1,694.55	0.00	0.00 %
... employee social insurance contributions	403.67	403.67	0.00	0.00 %
... self-employed social insurance contributions	109.55	109.55	0.00	0.00 %
... other social insurance contributions	0.00	0.00	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	2,925.93	2,925.93	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**64.23**	**64.23**	**0.00**	**0.00 %**
Government expenditure on social transfers	**2,376.26**	**2,511.65**	**135.39**	**5.70 %**
by target group				
... unemployment benefits	254.74	254.74	0.00	0.00 %
... family and education benefits	87.84	230.16	142.32	162.02 %
... social assistance and housing benefits	97.94	91.01	-6.93	-7.08 %
... pensions, health and disability benefits	1,935.74	1,935.74	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	97.94	91.01	-6.93	-7.08 %
... non-means-tested non-pension benefits	342.58	484.90	142.32	41.54 %
... pensions	1,935.74	1,935.74	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

	Poverty Risk for EE_2012 (base)	Poverty Risk for EE_2012_E3	Difference to base
At-risk-of-poverty rate	21.26 %	19.72 %	-1.54pp
Median at-risk-of-poverty gap	35.04 %	32.29 %	-2.75pp
Fixed Poverty line (monthly)	487.79		

Exercise 3b: Reforming Child benefit in Greece

The aim of this exercise is to get familiar with functions DefVar, Elig and ArithOp.

Steps:

1. Create a new system in Greece based on the 2021 system and name it EL_2021_E3;

2. Add a supplement to the existing child benefit (bch_s). The supplement is given to recipient families with at least one child aged up to 6. Its aim is to increase the child benefit amount to 300 EUR per month (and not to reduce it, if it is higher than this amount);

3. Run EUROMOD to produce micro-outputs for these two systems (EL_2021 and EL_2021_E3);

4. Analyse the results with the statistics presenter.

Description:

The aim of this exercise is to perform a reform of the child benefit (bch_el) in Greece in 2021. In that year, the benefit was means-tested, paid to families with children below an age limit. The basic amount for the first and the second child was 70 EUR/month, and for the third and beyond 140 EUR/month.

Your task is to add a supplement to this benefit, using functions *DefVar*, *Elig* and *ArithOp*. The supplement is given to recipient families with at least one child up to age of 6, and increases the child benefit amount to 300 EUR per month. It does not reduce it, if it is higher than this amount.

Hints:

- Use DefVar to define a temporary variable for this top up. You can name it i_bch.
- Use the same tax/assessment unit for your calculations as the one already used in this policy, i.e. *tu_bch_el*.

Solution and further information

Task 1: Creating a new system

Our first step is to open Greece (country acronym EL). Hide all other systems except for EL_2021. We then go to this system and right-click on it. This provides several options (see Figure 3a.4). You can rename the system to *EL_2021_E3* using the option *Rename System*.

Task 2: Implementing the child benefit reform

The next task to implement the reform in the new system is to find the Child benefit policy in the spine. The easiest way to do this is to use the search option (Ctrl+F), i.e. search for **bch_el.**

Now we need to define the intermediate variable for the benefit supplement. We do that by adding a new *DefVar* function and include a parameter for the intermediate variable definition.
Adding the new function: right-click on the last function of the policy (policy 49.12) to activate the context menu. Select 'Add Function After' and then select the system function *DefVar*. Set the function as *on* for the reform system (EL_2021_E3).
Adding parameter to define the intermediate variable: point the cursor at the new function, right-click on it and select *Show Add Parameter Form*. In the *Add Parameters* list select the parameter *[Placeholder]* and click the *Add* button. Name the intermediate variable i_bch and set its original value equal to zero.

Figure 3b.1: Defining the intermediate variable

	bch_el		**on**	**on**	**BEN: child benefit (επίδομα παιδιού)**
fx	BenCalc		on	on	equivalence scale for children
fx	BenCalc		on	on	equivalence scale for household head
fx	BenCalc		on	on	equivalence scale for non household head
fx	BenCalc		on	on	additional equivalence scale for lone parents
fx	ArithOp		on	on	define household equivalent weight
fx	ArithOp		on	on	define household equivalent income
fx	BenCalc		on	on	full amount
fx	BenCalc		on	on	reduction ratios
fx	ArithOp		on	on	apply reduction to benefit
fx	ArithOp		n/a	n/a	addiitonal instalment due to inflation
fx	BenCalc		switch	switch	non take-up adjustment
fx	DefVar		n/a	on	define intermediate variable
	i_bch	1	n/a	0	child benefit top up

To give the supplement to recipient families who have at least one child up to the age of 6, we need to introduce an eligibility function (*Elig*). Right-click on the *DefVar* function to activate the context menu again. Select 'Add Function After' and select function *Elig*. This creates the new function with its two compulsory parameters - *Elig_Cond* and *TAX_UNIT* - with default values *n/a*. Set our new function as *on* for the reform system.

The next step is to choose the appropriate tax unit. In our case it will be *tu_bch_el* (the tax unit applicable to the child benefit policy) so in the *TAX_UNIT* field, click on the grey arrow and select it from the drop-down list (similar to Figure 3a.8 in the previous exercise). To create the eligibility condition we fill in the Elig_Cond field with a specific expression able to capture eligible tax units. The supplement should be restricted to families i) already receiving the child benefit and ii) with at least one child aged up to 6. The expression to be typed in *Elig_Cond*: bch_s > 0 & dag <= 6.

Please keep in mind that conditions related to monetary variables are assessed at the tax unit level, whereas conditions related to non-monetary variables (such as age) are assessed at the individual level.

Figure 3b.2: The modified new *Elig* function

fx	DefVar		n/a	on	define intermediate variable
	i_bch	1	n/a	0	child benefit top up
fx	Elig		n/a	on	eligibility conditions for top up
	Elig_Cond		n/a	bch_s>0 & dag<=6	
	TAX_UNIT		n/a	tu_bch_el	

Our next task is to calculate the supplement amount. We do it with the arithmetic function *ArithOp* which is placed just after the function *Elig*. Right-click on function *Elig => Add Function After => ArithOp*. This creates the new function with its three compulsory parameters: *Formula*, *Output_var* and *TAX_UNIT* with default values *n/a*.

- Set the function to *on* for the reform system (EL_2021_E3)
- Set the value of the parameter *Formula* from n/a to 300#m – bch_s
- Set the value of the *Output_Var* to the simulated benefit bch_s
- Define the tax unit: it will be the same as in the previous function - tu_bch_el

Remember that we need to assign the top up only to families who satisfy the eligibility condition. This is done by creating an interaction between the eligibility function (*Elig*) and the arithmetical function (*ArithOp*) by using the parameter *Who_Must_Be_Elig*. Point the cursor at the function *ArithOp*, then right-click on the function or a parameter name in *ArithOp* and select *Show Add Parameter Form*. In the *Add Parameters* list we need to select the parameter *Who_Must_Be_Elig* and click the *Add* button. You can drag the new parameter in order to have it at the beginning of the function. Set its value to *one*, as we need at least one member of a child benefit-recipient family complying with the age criterion.

As far as the top up amount is concerned, our current implementation (300#m – bch_s) does not guarantee that the original child benefit is not reduced by the top up (as bch_s can be higher than 300#m). To do so, we need to add a new parameter to the *ArithOp* function: *LowLim*. We do that the same we introduced the *Who_Must_Be_Elig* parameter, and set it equal to zero.

Moreover, as we would like to have the (intermediate) child benefit top up variable included in our output file, and not just the final child benefit (this is very useful for checking purposes), we need to add another parameter to the function, namely *Result_Var*, and set it equal to i_bch.

Finally, we will need to replace parameter *Output_Var* with parameter *Output_Add_Var*, in order to add the supplement to the already calculated child benefit (and not replace its original value with the one of the top up). To do this, simply left-click on the parameter *Output_Var* and you will be asked if you would like to replace it. Your reform is now ready to run.

Figure 3b.3: The modified new *ArithOp* function

fx	DefVar		n/a	on	define intermediate variable
	i_bch	1	n/a	0	child benefit top up
fx	Elig		n/a	on	eligibility conditions for top up
	Elig_Cond		n/a	bch_s>0 & dag<=6	
	TAX_UNIT		n/a	tu_bch_el	
fx	ArithOp		n/a	on	top up amount
	Who_Must_...		n/a	one	
	Formula		n/a	300#m - bch_s	
	LowLim		n/a	0	
	Result_Var		n/a	i_bch	
	Output_Ad...		n/a	bch_s	
	TAX_UNIT		n/a	tu_bch_el	

Task 3: Producing the micro-output

Save your changes and run EUROMOD for the baseline EL_2021 and reform EL_2021_E3 systems using *training_data*.

Task 4: Analyse the results

Use the *Statistics Presenter* (Baseline/Reform option) to analyse the fiscal and distributional impact of the reform.

Figure 3b.4: Budgetary and distributional impact

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	EL_2021 (base) (annual)	EL_2021_E3 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**38,367.24**	**38,367.24**	**0.00**	**0.00 %**
... income from (self) employment	38,367.24	38,367.24	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**22,365.79**	**22,365.79**	**0.00**	**0.00 %**
... direct taxes	3,928.10	3,928.10	0.00	0.00 %
... employee social insurance contributions	5,417.45	5,417.45	0.00	0.00 %
... self-employed social insurance contributions	2,938.17	2,938.17	0.00	0.00 %
... other social insurance contributions	1,434.14	1,434.14	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	8,647.93	8,647.93	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**0.00**	**0.00**	**0.00**	**0.00 %**
Government expenditure on social transfers	**33,907.88**	**34,651.56**	**743.68**	**2.19 %**
by target group				
... unemployment benefits	5,267.97	5,267.97	0.00	0.00 %
... family and education benefits	1,062.07	1,822.43	760.36	71.59 %
... social assistance and housing benefits	3,409.29	3,392.61	-16.68	-0.49 %
... pensions, health and disability benefits	24,168.55	24,168.55	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	4,315.21	5,058.88	743.68	17.23 %
... non-means-tested non-pension benefits	5,277.90	5,277.90	0.00	0.00 %
... pensions	24,314.77	24,314.77	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

	Poverty Risk for EL_2021 (base)	Poverty Risk for EL_2021_E3	Difference to base
At-risk-of-poverty rate	23.49 %	22.49 %	-1.00pp
Median at-risk-of-poverty gap	51.77 %	51.81 %	0.05pp
Fixed Poverty line (monthly)	502.83		

Exercise 4: Introducing a supplement, withdrawn with earnings, to the Child Allowance in Estonia

The aim of this exercise is to learn how to use the function BenCalc.

Steps:

1. Create a new system in Estonia based on the 2021 system and name it EE_2021_E4;

2. Introduce a supplement for large families to the Child Allowance, which is withdrawn with earnings;

3. Run EUROMOD to produce micro-outputs for the baseline and reform systems;

4. Analyse the distributive impact of this reform.

Description:

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 for the whole family 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.

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*.

Solution and further information

Task 1: Creating a new system

Load the Estonian model. Hide all systems except for EE_2021. Add a new reform system based on EE_2021 and call it EE_2021_E4.

Task 2: Introduce a supplement to the Child Benefit

First, to introduce the supplement, we need a function which defines **who is entitled to it and how much the supplement amounts to**. We will use *BenCalc* (benefit calculator) function to help us with that, as it allows for modelling a wide range of policy instruments, particularly benefits.

- Locate the Child Benefit policy (bch00_ee) on the policy spine, right-click on the first function *BenCalc* and select from the option *Add Function After* the function *BenCalc*, as shown in Figure 4.1. This creates the new function with its compulsory parameters *Comp_Cond*, *Comp_perTU*, *output_var* and *TAX_UNIT* (with default values n/a). Note that the parameters *Comp_Cond* and *Comp_PerTU* are already grouped together into one component (column "Grp/No"), so you do not need to do anything. Set the function to **'on'** for the reform system (EE_2021_E4).

Figure 4.1: Adding a new BenCalc function after the current function in policy bch00_ee

- For the system we are reforming (EE_2021_E4), change the value of the parameter *Comp_Cond* to *nDepChildrenInTU#x>=2*. Define the age limits there. Thus, the eligibility condition says that there should be at least 2 dependent children below 18 in the benefit unit.
- Change the value of the parameter *Comp_perTU* to *30#*w i.e. €30 per week.
- Change the parameter value *Output_Var* from *n/a* to the simulated benefit *bch00_s*. You also need to use the parameter *Output_Add_Var* instead of *Output_Var*, to add the supplement to (rather than replace the value of) the benefit calculated with the first *BenCalc* function. To do

this, simply click on the parameter *Output_Var* and you will be asked if you would like to replace it.

- Set the value of the parameter *TAX_UNIT* from *n/a* to *tu_CBfamily_ee* (i.e. the relevant benefit unit or the family used to calculate the Child Benefit).

Your reformed system should now look like Figure 4.2.

Figure 4.2: The modified function reform system

bch00_ee		**on**	**on**	**BEN: child allowance (lapsetoetus)**
BenCalc		**on**	**on**	monthly benefit
comp_cond	1	IsNtoMchild#1	IsNtoMchild#1	1) for the first child
#_N	1	1	1	
#_M	1	1	1	
comp_perElig	1	$CB_Ch1	$CB_Ch1	
comp_cond	2	IsNtoMchild#2	IsNtoMchild#2	2) for the second child
#_N	2	2	2	
#_M	2	2	2	
comp_perElig	2	$CB_Ch2	$CB_Ch2	
Comp_Cond	3	IsNtoMchild#3	IsNtoMchild#3	3) for the third and each subsequent child
#_N	3	3	3	
#_M	3	99	99	
Comp_perElig	3	$CB_Ch3plus	$CB_Ch3plus	
output_var		bch00_s	bch00_s	
TAX_UNIT		tu_CBfamily_ee	tu_CBfamily_ee	children up to 16 years and 17-19 years if studying at basic, upper secondary or vocational school (põhik. baasil). From 2023, children until 19. If studying, then including 19.
BenCalc		n/a	**on**	top up
Comp_Cond	1	n/a	nDepChildrenInTu#1>=2	
Comp_perTU	1	n/a	30#w	
#_AgeMin	1	n/a	0	
#_AgeMax	1	n/a	18	
Output_Add_Var		n/a	bch00_s	
TAX_UNIT		n/a	tu_CBfamily_ee	

Second, we need to **calculate the benefit withdrawal**:

- Right-click on the function or a parameter name of the new *BenCalc* function and select *Show Add Parameter* Form. In the Add Parameters list select the parameters *Withdraw_Base*, *Withdraw_Start*, *Withdraw_Rate* and click the *Add* button, as shown in Figure 4.3. You can drag the parameters within the *BenCalc* function to change their position, e.g. after *Comp_perTU*. Set *Withdraw_Base* equal to *yem* (variable for earnings), *Withdraw_Start* to *10000#y* and *Withdraw_Rate* to *0.1*, as shown in Figure 4.4.

Figure 4.3: adding BenCalc parameters *Withdraw_Base, Withdraw_Start, Withdraw_Rate*

Add Parameters

BenCalc (order: 3) in policy bch00_ee

Add	Parameter	Replaces	Grp/No	Count	Default	Description
☐	Base					Base amount that can be used with parameters compX_perT...
☑	Withdraw_Base					Withdraw_Base * Withdraw_Rate is deducted from function's ...
☑	Withdraw_Rate				0	Withdraw_Base * Withdraw_Rate is deducted from function's ...
☑	Withdraw_Start				0	Level of Withdraw_Base where withdrawal starts.
☐	Withdraw_End				1.79769313486232E+308	Level of Withdraw_Base where withdrawal ends (i.e. benefit i...
☐	Comp_Cond		2	1		Condition that must be fulfilled to add the component (comp_p...
☐	Comp_perTU		2	1		Formula to calculate one component of the function's result. T...
☐	Comp_perElig		2	1		Formula to calculate one component of the function's result. T...
☐	Comp_LowLim		2	1	-1.79769313486232E+308	Replaces component if component is smaller.
☐	Comp_UpLim		2	1	1.79769313486232E+308	Replaces component if component is higher.
☐	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...

☑ Show Common Parameters
☑ Show Footnote Parameters

Description (F5) | Summary (F6) | Add | Close

Figure 4.4: The completed modified function reform system

fx BenCalc		n/a	on	top up
Comp_Cond	1	n/a	nDepChildrenInTu#1>=2	
Comp_perTU	1	n/a	30#w	
#_AgeMin	1	n/a	0	
#_AgeMax	1	n/a	18	
Withdraw_Base		n/a	yem	
Withdraw_Rate		n/a	0.1	
Withdraw_Start		n/a	10000#y	
Output_Add_Var		n/a	bch00_s	
TAX_UNIT		n/a	tu_CBfamily_ee	

Finally, save your changes.

Task 3: Producing the micro-output

Run EUROMOD for the baseline EE_2021 and reform EE_2021_E4 systems with EE_*training_data* as the dataset.

Task 4: Analyse the results

Use the *Statistics Presenter* – option Baseline/Reform – to analyse the fiscal and poverty impact of the reform. Figure 4.5 shows the budgetary impact.

Figure 4.5: Budgetary impact

Summary Statistics - Baseline vs Reform(s)

Results for Estonia: EE_2021 vs EE_2021_E4

1. Fiscal | 2. Distributional | 3. Inequality and Poverty | Metadata

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	EE_2021 (base) (annual)	EE_2021_E4 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**8,728.13**	**8,728.13**	**0.00**	**0.00 %**
... income from (self) employment	8,728.13	8,728.13	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**4,684.25**	**4,684.25**	**0.00**	**0.00 %**
... direct taxes	1,215.36	1,215.36	0.00	0.00 %
... employee social insurance contributions	302.74	302.74	0.00	0.00 %
... self-employed social insurance contributions	181.51	181.51	0.00	0.00 %
... other social insurance contributions	0.00	0.00	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	2,984.63	2,984.63	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**107.62**	**107.62**	**0.00**	**0.00 %**
Government expenditure on social transfers	**2,751.85**	**2,802.66**	**50.81**	**1.85 %**
by target group				
... unemployment benefits	271.28	271.28	0.00	0.00 %
... family and education benefits	318.25	381.94	63.69	20.01 %
... social assistance and housing benefits	226.58	213.69	-12.89	-5.69 %
... pensions, health and disability benefits	1,935.74	1,935.74	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	226.58	213.69	-12.89	-5.69 %
... non-means-tested non-pension benefits	589.53	653.22	63.69	10.80 %
... pensions	1,935.74	1,935.74	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

Exercise 5a: Introducing a zero tax band to the flat income tax in Bulgaria

The aim of this exercise is to learn how to use the function SchedCalc.

Steps:

1. Create a new system in Bulgaria based on the 2021 system and name it BG_2021_E5;

2. Introduce a zero tax band to the flat income tax and modify the tax rate;

3. Run EUROMOD to produce micro-outputs for the baseline and reform systems;

4. Analyse the distributive impact of this reform.

Description:

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%.

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.

Solution and further information

Task 1: Creating a new system

Open the Bulgarian model. Hide all systems except for BG_2021. Add a new reform system based on BG_2021 and call it BG_2021_E5 and locate the Income Tax policy.

Task 2: Modify the income tax

The income tax is implemented in the policy *tin_bg*. Since 2008, Bulgaria has had a flat tax of 10%, levied on taxable income (variable *tintb_s*) from the first BGN earned. The flat tax is calculated in the second *SchedCalc* function towards the end of the policy (policy spine row 38.13). The income tax parameters are defined as constants in the first function *DefConst* (policy spine row 38.1).

To implement the reform:

- First, increase the tax rate – constant *$flat_rate* – from 10% to 15% in the *DefConst* function in the reform system *BG_2021_E5*.
- Second, introduce a zero tax band of BGN 3,200 per year, by defining a new constant called *$flat_zeroband* in the *DefConst* function – to do so add a new *[Placeholder]* parameter and type the name for the constant in the *Policy* column. Type 3200#y in as value for the constant in the reform BG_2021_E5, as shown in Figure 5a.1.

Figure 5a.1: Introducing a zero tax band ($flat_zeroband) and new flat rate ($flat_rate)

Policy	Grp/No	BG_2021	BG_2021_E5	Comment
tin_bg		on	on	**TAX: income tax (данък общ доход)**
DefConst		on	on	Parameters used in the income tax policy - defined as constants
$tintach1		4500#y	4500#y	Child tax deduction amount for the first child in 2007 and again since 2015
$tintach2		9000#y	9000#y	Child tax deduction amount for the second child in 2007 and again since 2015
$tintach3		13500#y	13500#y	Child tax deduction amount for the all other children in 2007 and again since 2015
$tintapv		0.1	0.1	Tax deduction for private pension contributions - rate
$tintart		0.1	0.1	Tax deduction for income from rent - rate
$tintase		0.25	0.25	Tax deduction for freelancers
$tintadi		7920#y	7920#y	Tax deduction for disability - applied on the tax base (not taxable income!)
$band1_threshold		n/a	n/a	Zero tax band
$band2_rate		n/a	n/a	Rate of second band
$band2_threshold		n/a	n/a	Second tax threshold
$band3_rate		n/a	n/a	Rate of third band
$band3_amt		n/a	n/a	Second lump-sum tax
$band3_threshold		n/a	n/a	Third tax threshold
$band4_rate		n/a	n/a	Rate of fourth band in 2001 / Top marginal tax rate in 2007 - 2008
$band4_amt		n/a	n/a	Third lump-sum tax
$flat_rate		0.1	0.15	Flat tax rate from 2008 onwards
$flat_zeroband	1	n/a	3200#y	Zero tax band

- Third, modify the calculation of the flat tax in the function *SchedCalc*, by adding a new parameter *Band_LowLim* with a group number 1. In the reform BG_2021_E5, type in the new constant *$flat_zeroband* against *Band_LowLim*. Thus, the tax rate of 15% is levied on taxable income equal or above the zero tax band of BGN 3,200 per year. See Figure 5a.2.

Figure 5a.2: Modifying the calculaton on *SchedCalc*, including the new *$flat_zeroband*

SchedCalc		on	on	since 2008: flat income tax
base		tintb_s	tintb_s	tax base
Band_LowLim	1	n/a	$flat_zeroband	Zero tax band
band_rate	1	$flat_rate	$flat_rate	
output_var		tin_s	tin_s	income tax
TAX_UNIT		tu_individual_bg	tu_individual_bg	

Task 3: Producing the micro-output

Run EUROMOD for the baseline BG_2021 and reform BG_2021_E5 systems with BG_*training_data* as dataset.

Task 4: Analyse the results

Analyse the impact on the fiscal budget and the income distribution of the reform, you can use the *Statistics Presenter* with Baseline/Reform option. Figure 5a.3 show some selected results.

Figure 5a.3: Selected results

Summary Statistics - Baseline vs Reform(s)

Results for Bulgaria: BG_2021 vs BG_2021_E5

1. Fiscal | 2. Distributional | 3. Inequality and Poverty | Metadata

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	BG_2021 (base) (annual)	BG_2021_E5 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**36,662.08**	**36,662.08**	**0.00**	**0.00 %**
... income from (self) employment	36,662.08	36,662.08	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**14,504.18**	**14,553.23**	**49.05**	**0.34 %**
... direct taxes	2,522.80	2,571.85	49.05	1.94 %
... employee social insurance contributions	4,741.74	4,741.74	0.00	0.00 %

1. Fiscal | 2. Distributional | 3. Inequality and Poverty | Metadata

2. Distributional

2.1.a Mean household income (monthly) by decile groups

	BG_2021 (base)	BG_2021_E5	Difference to base	% Difference to base
Decile 1	145.24	146.15	0.91	0.63 %
Decile 2	156.09	156.23	0.14	0.09 %
Decile 3	351.96	361.27	9.31	2.65 %
Decile 4	534.29	540.16	5.87	1.10 %
Decile 5	650.81	661.23	10.42	1.60 %
Decile 6	932.34	947.64	15.30	1.64 %
Decile 7	1,237.86	1,246.24	8.38	0.68 %
Decile 8	1,504.20	1,504.07	-0.13	-0.01 %
Decile 9	1,688.45	1,680.73	-7.72	-0.46 %
Decile 10	2,176.47	2,140.95	-35.52	-1.63 %
All	**862.37**	**861.50**	**-0.87**	**-0.10 %**
Poor	186.18	186.74	0.56	0.30 %

3. Inequality and Poverty

3.1 Basic Inequality Indices

	Gini BG_2021	S80/S20 BG_2021	Gini BG_2021_E5	Diff. Gini BG_2021_E5/BG_2021	S80/S20 BG_2021_E5	Diff. S80/S20 BG_2021_E5/BG_2021
Original Income	0.5218	80.6769	0.5218	0.0000	81.1520	0.4751
Original Income after Taxes/SIC	0.5159	77.4254	0.5112	-0.0048	72.3284	-5.0969
Original Income incl. Public Pensions after Taxes/SIC	0.5159	77.4254	0.5112	-0.0048	72.3284	-5.0969
Disposable Income	0.4163	7.2816	0.4116	-0.0047	7.1918	-0.0898

3.2.a Basic Poverty Indices

	Poverty Risk for BG_2021 (base)	Poverty Risk for BG_2021_E5	Difference to base
At-risk-of-poverty rate	35.64 %	35.52 %	-0.12pp
Median at-risk-of-poverty gap	43.84 %	43.84 %	0.00pp
Fixed Poverty line (monthly)	260.55		

Exercise 5b: Reforming income tax in EL (using functions *SchedCalc* and *DefConst*)

The aim of this exercise is to learn how to modify and introduce new tax bands, using functions SchedCalc and DefConst.

Steps:

1. Create a new system based on the EL_2020 system and name it EL_2020_E5
2. Define the new tax band of 50,000 per year as a constant in policy tin00_el and add a new tax rate (48%) → function DefConst used to define the income tax parameters
3. Add the new tax band and modify the tax rates in the two SchedCalc functions used to calculate the income tax for employment, pension, self-employment & farming income
4. Run the 2020 system to produce micro-data outputs
5. Analyse the distributional impact of the reform using training_data

Description:

Income taxation is graduated, with progressively higher marginal tax rates applying to higher income brackets.

The table below summarises the income tax bands for tax year 2020:

Taxable income	Rate of tax
0 – 10,000	9%
10,001 – 20,000	22%
20,001 – 30,000	28%
30,001 – 40,000	36%
Over 40,000	44%

You are asked to modify the part of the income tax schedule. This change has to be applied both to the general tax base and to the farming income (which is taxed separately).

Keep the income tax schedule the same but on incomes above 50,000 euro apply a tax rate of 48%.

Analyse the impact of the tax reform on the government budget and income distribution.

Hints:

- Analyse the structure of the personal income tax in EL focusing on the policy (tin00_el)
- Perform a reform in EL that involves modifying the income tax, using the SchedCalc and DefConst functions
- Income tax band values and tax rates are defined with a DefConst function (see constants $tin_rate* and $tin_thres*). Define the new tax band as a constant (e.g. $tin_thres_ref) which has a value of 50,000 euro per year.
- Add parameters Band_Rate and Band_UpLim in each of the SchedCalc functions used to calculate the income tax on the general tax base (tintb_s) and on farming income (tinag_s)

Solution and further information

Task 1: Creating a new system

Our first task is to open EL and create a new system using the 2020 system as a base. Give the name EL_2020_E5 to the new system. Hide all systems but these two - 2020 and 2020_E5 (see exercise 1 for explanation).

Task 2: Analysing the current policy

Access the Income Tax policy in the EL (*tin00_el*). Consider year 2020. Hide other years (Figure 5b.1).

Figure 5b.1: Locating the Income Tax policy (*tin00_el*)

- Explore how personal income tax is calculated (policy *tin00_el*)

Further information can be found in the Country Report (see https://euromod-web.jrc.ec.europa.eu/using-euromod/country-reports/latest).

Locate the *SchedCalc* (Schedule Calculator) functions, used for the calculation of the income tax, by checking the comment column and the implementation. Expand it by clicking on the arrow on the left hand side of the function.

There are two SchedCalc functions: one for the general income tax and one for farming income tax. The general income tax base is defined by *tintb_s* (figure 5b.2), while the farming income tax base is defined by *tintbag_s* (figure 5b.3). For details of the components of the income list *tintb_s* and *tintbag_s*, see policy tin00_el (you can also use the search option (Ctrl+F) to locate these variables in the model, e.g. search for tinbt_s).

Figure 5b.2: income tax calculation in 2020, general income tax

SchedCalc		on	on	general tax schedule
who_must_...		n/a	n/a	
base		tintb_s	tintb_s	
band_uplim	1	$tin_thres1	$tin_thres1	
band_uplim	2	$tin_thres2	$tin_thres2	
band_uplim	3	$tin_thres3	$tin_thres3	
band_uplim	4	$tin_thres4	$tin_thres4	
band_uplim	5	n/a	n/a	
band_uplim	6	n/a	n/a	
band_uplim	7	n/a	n/a	
band_uplim	8	n/a	n/a	
band_rate	1	$tin_rate1	$tin_rate1	
band_rate	2	$tin_rate2	$tin_rate2	
band_rate	3	$tin_rate3	$tin_rate3	
band_rate	4	$tin_rate4	$tin_rate4	
band_rate	5	$tin_rate5	$tin_rate5	
band_rate	6	n/a	n/a	
band_rate	7	n/a	n/a	
band_rate	8	n/a	n/a	
band_rate	9	n/a	n/a	
output_var		tin00_s	tin00_s	
TAX_UNIT		tu_individual_el	tu_individual_el	

Figure 5b.3: income tax calculation in 2020, farming income tax

SchedCalc		on	on	tax schedule for farming income
base		tintbag_s	tintbag_s	taxed separately
band_uplim	1	$tin_thres1	$tin_thres1	
band_uplim	2	$tin_thres2	$tin_thres2	
band_uplim	3	$tin_thres3	$tin_thres3	
Band_UpLim	4	$tin_thres4	$tin_thres4	
band_rate	1	$tin_rate1	$tin_rate1	
band_rate	2	$tin_rate2	$tin_rate2	
band_rate	3	$tin_rate3	$tin_rate3	
band_rate	4	$tin_rate4	$tin_rate4	
Band_Rate	5	$tin_rate5	$tin_rate5	
LowLim		0	0	
Result_Var		tinag_s	tinag_s	
Output_Ad...		tin00_s	tin00_s	
TAX_UNIT		tu_individual_el	tu_individual_el	

The income tax bands and the tax rates are defined as constants – ($tin_thres1, $tin_thres2, $tin_thres3, $tin_thres4, $tin_rate1, $tin_rate2, $tin_rate3, $tin_rate4, $tin_rate5) – using a *DefConst* function in policy *tin00_el* (see Figure 5b.4) (you can also use the search option (Ctrl+F) to locate the constants in the model):

Figure 5b.4: Income tax constants in tin00_el

Policy	Grp/No	EL_2020	EL_2020_E5	Comment
tin00_el		on	on	TAX: personal income tax (φόρος εισοδήματος)
DefConst		on	on	define constants
$tc_rate		0.1	0.1	rate for tax credits
$yiyit_rate		0.15	0.15	rate for interest income
$tax_extra		10000#y	10000#y	threshold for extra taxable income
$tin_rate1		0.09	0.09	rate 1 for tin00_s
$tin_rate2		0.22	0.22	rate 2 for tin00_s
$tin_rate3		0.28	0.28	rate 3 for tin00_s
$tin_rate4		0.36	0.36	rate 4 for tin00_s
$tin_rate5		0.44	0.44	rate 5 for tin00_s
$tin_thres1		10000#y	10000#y	threshold 1 for tin00_s
$tin_thres2		20000#y	20000#y	threshold 2 for tin00_s
$tin_thres3		30000#y	30000#y	threshold 3 for tin00_s
$tin_thres4		40000#y	40000#y	threshold 4 for tin00_s
$tinag_rate		n/a	n/a	rate for tinag_s
$tinpr_rate1		0.15	0.15	rate 1 for tinpr_s
$tinpr_rate2		0.35	0.35	rate 2 for tinpr_s
$tinpr_rate3		0.45	0.45	rate 3 for tinpr_s
$tinpr_thres1		12000#y	12000#y	threshold 1 for tinpr_s
$tinpr_thres2		35000#y	35000#y	threshold 2 for tinpr_s

To find out the tax band values, you can also hover with the cursor over the constant names within the SchedCalc functions (see Figure 5b.5).

Figure 5b.5: Checking constant values by moving the cursor over the constant names

Policy	Grp/No	EL_2020	EL_2020_E5	Comment
Elig		n/a	n/a	general tax schedule: eligibility
SchedCalc		on	on	general tax schedule
who_must_...		n/a	n/a	
base		tintb_s	tintb_s	
band_uplim	1	$tin_thres1	$tin_thres1	
band_uplim	2	$tin_thres2	$tin_thres2	
band_uplim	3	$tin_thres3	$tin_thres3	
band_uplim	4	$tin_thres4	$tin_thres4	
band_uplim	5	n/a	n/a	
band_uplim	6	n/a	n/a	$tin_thres4 = 40000#y
band_uplim	7	n/a	n/a	
band_uplim	8	n/a	n/a	
band_rate	1	$tin_rate1	$tin_rate1	
band_rate	2	$tin_rate2	$tin_rate2	
band_rate	3	$tin_rate3	$tin_rate3	
band_rate	4	$tin_rate4	$tin_rate4	
band_rate	5	$tin_rate5	$tin_rate5	
band_rate	6	n/a	n/a	
band_rate	7	n/a	n/a	
band_rate	8	n/a	n/a	
band_rate	9	n/a	n/a	
output_var		tin00_s	tin00_s	
TAX_UNIT		tu_individual_el	tu_individual_el	

Task 3: Reforming the policy

In the reform system EL_2020_E5, add a new constant (i.e. parameter "placeholder") in the DefConst function where the income tax parameters are defined. The new constant will contain the value of the new income tax band – call it e.g. $tin_thres_ref. This constant should take the value of 50,000 euro per year which would be the default constant value and apply on all individuals. Note the constant should not have any group number (see column Grp/No) as no condition should apply on it (you may need to delete the group number if it has been automatically added).

To define the value of the tax rate, add a new constant (i.e. parameter "placeholder") with a new name (e.g. $tin_rate_ref), a value of 0.48. Once you are done, your implementation should look as in Figure 5b.6.

Figure 5b.6: The new income tax band and tax rate defined as a constant

tin00_el		on	on	TAX: personal income tax (φόρος εισοδήματος)
fx DefConst		on	on	define constants
$tc_rate		0.1	0.1	rate for tax credits
$yiyit_rate		0.15	0.15	rate for interest income
$tax_extra		10000#y	10000#y	threshold for extra taxable income
$tin_rate1		0.09	0.09	rate 1 for tin00_s
$tin_rate2		0.22	0.22	rate 2 for tin00_s
$tin_rate3		0.28	0.28	rate 3 for tin00_s
$tin_rate4		0.36	0.36	rate 4 for tin00_s
$tin_rate5		0.44	0.44	rate 5 for tin00_s
$tin_rate_ref		n/a	0.48	
$tin_thres1		10000#y	10000#y	threshold 1 for tin00_s
$tin_thres2		20000#y	20000#y	threshold 2 for tin00_s
$tin_thres3		30000#y	30000#y	threshold 3 for tin00_s
$tin_thres4		40000#y	40000#y	threshold 4 for tin00_s
$tin_thres_ref		n/a	50000#y	
$tinag_rate		n/a	n/a	rate for tinag_s
$tinpr_rate1		0.15	0.15	rate 1 for tinpr_s
$tinpr_rate2		0.35	0.35	rate 2 for tinpr_s
$tinpr_rate3		0.45	0.45	rate 3 for tinpr_s
$tinpr_thres1		12000#y	12000#y	threshold 1 for tinpr_s
$tinpr_thres2		35000#y	35000#y	threshold 2 for tinpr_s

To modify the income tax schedule of the general income tax, you can add the upper limit ($tin_thres_ref 50,000 euro) for tax rate 0.44 (group 5). Additionally, you have to add the new tax rate ($tin_rate_ref, 48%) for income above 50,000 euro. See figure 5b.7.

Figure 5b.7: Modified income tax schedule for the general income tax

fx SchedCalc		on	on	general tax schedule
who_must_be_elig		n/a	n/a	
base		tintb_s	tintb_s	
band_uplim	1	$tin_thres1	$tin_thres1	
band_uplim	2	$tin_thres2	$tin_thres2	
band_uplim	3	$tin_thres3	$tin_thres3	
band_uplim	4	$tin_thres4	$tin_thres4	
band_uplim	5	n/a	$tin_thres_ref	
band_uplim	6	n/a	n/a	
band_uplim	7	n/a	n/a	
band_uplim	8	n/a	n/a	
band_rate	1	$tin_rate1	$tin_rate1	
band_rate	2	$tin_rate2	$tin_rate2	
band_rate	3	$tin_rate3	$tin_rate3	
band_rate	4	$tin_rate4	$tin_rate4	
band_rate	5	$tin_rate5	$tin_rate5	
band_rate	6	n/a	$tin_rate_ref	
band_rate	7	n/a	n/a	
band_rate	8	n/a	n/a	
band_rate	9	n/a	n/a	
output_var		tin00_s	tin00_s	
TAX_UNIT		tu_individual_el	tu_individual_el	

To modify the income tax schedule of the farming income tax, add parameters Band_Rate and Band_UpLim in the SchedCalc function used to calculate the income tax. You may need to adjust the group number of the parameters (see column Grp/No), so they apply on the correct parts of the tax schedule (see Figures 5b.8 and 5b.9).

Figure 5b.8: Adding new parameters to the *SchedCalc* function

Add Parameters

SchedCalc (order: 28) in policy tin00_el

Add	Parameter	Replaces	Grp/No	Count	Default	Description
☐	Do_Average_Rates					If set to yes the rate of the highest band reached by Base is...
☐	Quotient					If defined Base is divided by the quotient before the schedu...
☐	BaseThreshold					If Base is smaller result is set to zero.
☐	Round_Base					If defined Base is rounded to nearest whole number if set to...
☐	Simple_Prog					If set to yes the same rate/amount is applied on all income. ...
☑	Band_Rate		6	1		Rate to apply on band.
☐	Band_Amount		6	1		Amount to add for band.
☐	Band_LowLim		6	1		Lower limit of band.
☑	Band_UpLim		6	1		Upper limit of band.
☐	Output_Var	Output_Add_Var				Variable for storing the result of the function. Result of functi...
☐	Who_Must_Be_Elig					Function's calculations are carried out if ...- one (one_memb...
☐	Elig_Var				sel_s	Variable indicating whether a person is 'eligible' (see parame...
☐	Run_Cond					Function is only carried out if the condition is fulfilled. The p...
☐	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 ...
☐	Limpriority				n/a	Parameter for the further specification of an operand:Possibl...
☐	Round_Down					Result is rounded down to nearest whole number if set to 1,...
☐	Round_Up					Result is rounded up to nearest whole number if set to 1, to ...
☐	Round_to					Result is rounded to nearest whole number if set to 1, to ne...
☐	#_LowLim		1	1	-1.79769313486232E+308	Footnote parameter for the further specification of an opera...

☑ Show Common Parameters
☑ Show Footnote Parameters

Description (F5) | Summary (F6) | Add | Close

Figure 5b.9: Modified income tax schedule for the farming income tax

fx SchedCalc		on	on	tax schedule for farming income
base		tintbag_s	tintbag_s	taxed separately
band_uplim	1	$tin_thres1	$tin_thres1	
band_uplim	2	$tin_thres2	$tin_thres2	
band_uplim	3	$tin_thres3	$tin_thres3	
Band_UpLim	4	$tin_thres4	$tin_thres4	
Band_UpLim	5	n/a	$tin_thres_ref	
band_rate	1	$tin_rate1	$tin_rate1	
band_rate	2	$tin_rate2	$tin_rate2	
band_rate	3	$tin_rate3	$tin_rate3	
band_rate	4	$tin_rate4	$tin_rate4	
Band_Rate	5	$tin_rate5	$tin_rate5	
Band_Rate	6	n/a	$tin_rate_ref	
LowLim		0	0	
Result_Var		tinag_s	tinag_s	
Output_Add_Var		tin00_s	tin00_s	
TAX_UNIT		tu_individual_el	tu_individual_el	

Task 4: Produce and analyse micro-data output

Run EUROMOD to produce micro-output for the baseline (EL_2020) and new system (EL_2020_E5) using the EL_*training_data* dataset. Then use the Statistics Presenter Tool (EUROMOD Statistics – baseline/reform) to analyse the fiscal and distributional impact of the reform.

***If you need any additional information regarding any of the things mentioned above, you can easily access the EUROMOD Help & Info (the last tab in the ribbon bar), click on the Help and type the word(s) in the search tab and/or you can continue with exercises.

Figure 5b.19: Budgetary impact of reform

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	EL_2020 (base) (annual)	EL_2020_E5 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**38,367.24**	**38,367.24**	**0.00**	**0.00 %**
... income from (self) employment	38,367.24	38,367.24	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**23,608.07**	**23,608.07**	**0.00**	**0.00 %**
... direct taxes	4,019.35	4,019.35	0.00	0.00 %
... employee social insurance contributions	5,881.66	5,881.66	0.00	0.00 %
... self-employed social insurance contributions	2,938.17	2,938.17	0.00	0.00 %
... other social insurance contributions	1,434.14	1,434.14	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	9,334.73	9,334.73	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**0.00**	**0.00**	**0.00**	**0.00 %**
Government expenditure on social transfers	**34,572.94**	**34,572.94**	**0.00**	**0.00 %**
by target group				
... unemployment benefits	5,267.97	5,267.97	0.00	0.00 %
... family and education benefits	1,066.19	1,066.19	0.00	0.00 %
... social assistance and housing benefits	3,369.70	3,369.70	0.00	0.00 %
... pensions, health and disability benefits	24,869.08	24,869.08	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	4,289.67	4,289.67	0.00	0.00 %
... non-means-tested non-pension benefits	5,267.97	5,267.97	0.00	0.00 %
... pensions	25,015.30	25,015.30	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

Exercise 6: Introducing a benefit cap in Denmark

The aim of this exercise is to learn how to introduce a new policy, create a new income list and use the functions BenCalc and DefIL.

Steps:

1. Create a new system in Denmark based on the 2016 system and name it "DK_2016_E6";

2. Create a new policy to limit the total amount of means tested benefits received by households in Denmark;

3. Run EUROMOD to produce micro-outputs for these two systems (DK_2016 and DK_2016_E6);

4. Analyse the budgetary and distributive impact of this reform.

Description:

The aim of this exercise is the creation of a new policy that, from 2016, will include a limit on the total amount of means-tested benefit that people can receive in Denmark – a benefit cap. The benefits that will be affected by the cap are *Social Assistance Benefit* (bsa_s), *Housing Benefit* (bho01_s), *Housing Grant* (bho02_s) and *Green Check* (bhtuc_s).

Some working-age families, because of their special circumstances, will not be affected by the benefit cap. This is the case if anyone in the household qualifies for *Disability Benefit* (pdi), *Unemployment Benefit* (bunct_s) and *Survivor's benefit* (psu).

The maximum level of benefits is set to 15,000 Dkr per month for households with 4 or more dependent children and at 10,000 Dkr per month for households with by 3 or fewer dependent children.

Your task is to introduce this new policy – **ben_cap** using functions *DefIl*, *Elig* and *BenCalc* keeping in mind that this policy is a cap on other benefits, i.e. all these benefits need to be calculated first in order to set the cap

Hints:

- Use two *DefIL* functions to define 1) an income list for the benefits that will be capped and 2) another income list for the benefits that prevent the cap being applied.
- Using a *BenCalc* function, calculate the benefit cap as the benefit amount in excess of the cap. Use the optional parameter *LowLim*, equal to 0, to apply a minimum amount on the cap of 0 (i.e. to avoid negative values).
- Save the result of the benefit cap in the variable *brd_s* (b:benefit, rd_reduction, _s: simulated).
- Subtract the benefit cap from the sum of all means-tested benefits, i.e. income list *ils_benmt* in the policy *ILsDef_dk*.
- The household in the Danish model is defined as the tax unit *tu_household_dk*.

Solution and further information

Task 1: Creating a new system

Open Denmark, hide all systems except for DK_2016 and create a new system using the 2016 system as a base. Give the name "DK_2016_E6" to the new system.

Task 2: Implementing a new policy

Before implementing a new policy, we should first analyse the tax-benefit system in Denmark – what are the policies, how are they constructed, how are they connected, which income list(s) do they use, which benefits will be affected by the cap, which tax units are used, what are the output variables, etc.

Based on the previous analysis, we can see that the new policy will be a cap on the *Social Assistance Benefit* (bsa_s), *Housing Benefit* (bho01_s), *Housing Grant* (bho02_s) and *Green Check* (bhtuc_s), i.e. all these benefits need to be calculated first in order to set the cap.

Creating a new policy in the spine

As we can see from the policy spine, the *green check policy* (bhtuc_dk) is the last of the four benefits considered for the cap. Right-clicking on this policy, will give a list of different options. Choose *Add Policy After* and since we are creating a new benefit, choose *Benefit* (Figure 6.1).

Figure 6.1: Adding a new policy

When we choose the option to create a new benefit, we will be asked to give it a name. Naming new variables, components, policies, etc. should follow the *EUROMOD Modelling Conventions* document. Based on those rules, we will name our benefit cap policy **bencap_dk** (Figure 6.2) and set it as *on* instead of *n/a* in our reform system (DK_2016_E6).

When we create a new policy, function, etc. we should also make a comment (in column *Comment*) so that we have a record what this policy adjustment is for. In our case, we can simply type: 'BEN: *Benefit cap'*.

Figure 6.2: Switching the new policy on and adding a name

Generating income lists

The next step is to generate two new income lists:

- one containing benefits to be capped (we could name it e.g. *il_bencap*)
- one containing benefits that prevent the cap to be applied (we could name it e.g. *il_bencap_dis*).

Income list *il_bencap* should include all the benefits that will be affected by the cap – bsa_s, bho01_s, bho02_s and bhtuc_s.

To define the new income lists, right-click on the *bencap_dk* policy. Select function *DefIl* under *System Functions* (Figure 6.3).

Figure 6.3: Adding an income list function

A parameter *Name* will be created automatically when we add function *DefIl*. Set *DefIl* to *on* instead of *n/a* and name the income list il_bencap in the reform system (DK_2016_E6). Add an explanation in the *Comment* column. Then add all the previously mentioned benefits that will be affected by the cap. To do this, right-click on *Name*, and choose *Show Add Parameter Form*.

Click on [*Placeholder*] and type "4" in field *Count* (Figure 6.4). This will add 4 empty placeholder rows.

Figure 6.4: Adding new placeholders

Add	Parameter	Replaces	Grp/No	Count	Default	Description
☐	Warn_If_NonMonetary				True	If yes, a warning is issued if any component is non-monetary.
☑	[Placeholder]			4		[Placeholder] stands for the name of a component (variable or incomelist) of t...
☐	RegExp_Def		1	1		
☐	RegExp_Factor		1	1		
☐	Run_Cond					Function is only carried out if the condition is fulfilled. The parameter is intend...
☐	#_DataBasename		1	1		Parameter of query IsUsedDatabase.
☐	#_VariableName		1	1		Parameter of query IsDataVariable.

The next step is to rename the fields to include all the benefits that will be affected by the cap – bsa_s, bho01_s, bho02_s and bhtuc_s the and type sign "+" as the il_bencap should be the sum of their values (Figure 6.5).

Figure 6.5: Add benefits to the income list

bencap_dk		n/a	on	BEN: Benefit cap
fx DefIl		n/a	on	Benefits affected by the cap
Name		n/a	il_bencap	
bsa_s		n/a	+	
bho01_s		n/a	+	
bho02_s		n/a	+	
bhtuc_s		n/a	+	

Repeat the same steps to define the other income list - il_bencap_dis – which will include all benefits that prevent the cap being applied (pdi, psu, bunct_s) (Figure 6.6).

Figure 6.6: Adding a second income list

bencap_dk		n/a	on	BEN: Benefit cap
fx DefIl		n/a	on	Benefits affected by the cap
Name		n/a	il_bencap	
bsa_s		n/a	+	
bho01_s		n/a	+	
bho02_s		n/a	+	
bhtuc_s		n/a	+	
fx DefIl		n/a	on	Benefits preventing the cap
Name		n/a	il_bencap_dis	
pdi		n/a	+	
psu		n/a	+	
bunct_s		n/a	+	

Defining the eligibility condition for the cap

Now we need to define the eligibility condition for the cap: no one in the household should receive any of the benefits listed in the income list *il_bencap_dis* (i.e. these benefits should be zero), by using the function *Elig*. The relevant tax unit is tu_household_dk (Figure 6.7).

Figure 6.7: Adding an Elig function

fx Elig		n/a	on	
Elig_Cond		n/a	il_bencap_dis=0	
TAX_UNIT		n/a	tu_household_dk	

Calculating the benefit reduction for each affected household

After setting the eligibility condition, we need to calculate the benefit reduction for each affected household by using function *BenCalc*.

The maximum level of benefits needs to be set to 15,000 Dkr per month for households with 4 or more dependent children and at 10,000 Dkr per month for households with by 3 or less dependent children. This function will have, based on the previous information, two conditions – households with 4 or more and households with less than 4 dependent children.

To do that we add a new B*enCalc* function after the *Elig* function.

When we add function *BenCalc*, we will automatically get four components – *Comp_Cond*, *Comp_perTU*, *Output_Var* and *TAX_UNIT*. Components *Comp_Cond* and *Comp_perTU* are connected by the number from column *Grp/No* so that we know which condition is related to which value. Since we have to create two conditions, we will have to add new pair of components - *Comp_Cond* and *Comp_perTU* by using *Show Add Parameter Form* and connect them with group number "2".

Also, we can change the order of the components inside the policy simply by pressing the cursor, dragging the component to the place where we want it to be and releasing it when we see the blue arrow.

Let's change BenCalc now to implement the reform:

- The first condition refers to households with 4 or more dependent children. Set the value of the first *Comp_Cond* to *nDepChildrenInTu>=4*, where the variable *nDepChildrenInTu* is a query. The

benefit cap equals the benefit amount in excess of 15,000 Dkr per month. Thus, set the value of the first *Comp_perTU* to *il_bencap-15000#m*.

- The second condition refers to households with less than 4 children. Set the value of the second *Comp_Cond* to *nDepChildrenInTu<4*. The benefit cap equals the benefit amount in excess of 10,000 Dkr per month. Thus, set the value of the second *Comp_perTU* to *il_bencap-10000#m*.
- It is very important to see if there should be some minimum/maximum values. In our case, we do not want to have negative values, so we will set up the minimum to be zero. In *Show Add Parameter Form* choose component *LowLim* (low limit) and this limit will apply on the result of the function. If our limit is different for different components, choose *#_LowLim* and connect it using the same group number in column *Grp/No*.
- To link the *Elig* and *BenCalc* function, use the parameter *Who_Must_Be_Elig* from *Show Add Parameter Form* box. All members need to fulfil the eligibility condition, so set the value of the parameter to *all*.
- Use the household as the tax unit - tu_household_dk - and call the output variable brd_s (Figure 6.8).

Figure 6.8: Final implementation of the benefit cap

bencap_dk		n/a	on	**BEN: Benefit cap**
fx DefIl		n/a	on	Benefits affected by the cap
Name		n/a	il_bencap	
bsa_s		n/a	+	
bho01_s		n/a	+	
bho02_s		n/a	+	
bhtuc_s		n/a	+	
fx DefIl		n/a	on	Benefits preventing the cap
Name		n/a	il_bencap_dis	
pdi		n/a	+	
psu		n/a	+	
bunct_s		n/a	+	
fx Elig		n/a	on	Eligibility for the cap
Elig_Cond		n/a	il_bencap_dis=0	
TAX_UNIT		n/a	tu_household_dk	
fx BenCalc		n/a	on	Calculation of the benefit cap
Who_Must_...		n/a	all	all members
Comp_Cond	1	n/a	nDepChildrenInTu>=4	Households with 4 or more children
Comp_perTU	1	n/a	il_bencap - 15000#m	
Comp_Cond	2	n/a	nDepChildrenInTu<4	Households with less than 4 children
Comp_perTU	2	n/a	il_bencap - 10000#m	
LowLim		n/a	0	no negative values
Output_Var		n/a	brd_s	
TAX_UNIT		n/a	tu_household_dk	

Finally, before running the model, we need to subtract variable brd_s from the sum of all means-tested benefits – income list *ils_benmt* in policy *ILsDef_dk* – which in turn is part of household disposable income (income list ils_dispy) (Figure 6.9).

Figure 6.9: Adding the benefit cap to the means-tested benefits income list

IlsDef_dk		on	on	DEF: STANDARD INCOME CONCEPTS
DefIl		on	on	earnings
DefIl		on	on	original income
DefIl		on	on	original and replacement income
DefIl		on	on	simulated benefits
DefIl		on	on	public pensions
DefIl		on	on	non means-tested benefits
DefIl		on	on	means-tested benefits
	name	ils_benmt	ils_benmt	
	bsa_s	+	+	social assistance benefit (Kontanthjælp; Aktivering af kontanthjælpsmodtagere og flygtninge; Integrationsydelse til flygtninge)
	bsaot	+	+	other means-tested social assistance benefits
	bho01_s	+	+	housing benefit (Boligsikring)
	bho02_s	+	+	housing grant (Boligydelse)
	bhtuc_s	+	+	green check (Grøn check)
	bfachnm_s	+	+	child family grant (Børnefamilieydelse)
	bsard_s	n/a	n/a	benefit ceiling
	bfachxp_s	n/a	n/a	Extra child benefit
	bfach00_s	n/a	n/a	
	bsaxp_s	n/a	n/a	Social assistance inflation compensation (Kontanthjælp)
	brd_s	n/a	-	Amount in excess of the cap

Task 3: Producing the micro-output

Run EUROMOD for the baseline DK_2016 and reform DK_2016_E6 systems using DK_*training_data* as the dataset.

Task 4: Analyse the results

Use the *Statistics Presenter Tool* (Baseline/Reform option) to analyse the budgetary and distributional impact of the benefit cap. Figure 6.10 shows distributional results by household type.

Hint: A benefit cap was implemented in reality in 2017. Have a look at the actual benefit ceiling in the model.

Figure 6.10: Distributional results by household type

2.2.b Mean equivalised income (monthly) by household type

	DK_2016 (base)	DK_2016_E6	Difference to base	% Difference to base
One adult < 65, no children	14,878.38	13,910.11	-968.26	-6.51 %
- Female adult	14,878.38	13,910.12	-968.26	-6.51 %
- Male adult	14,878.38	13,910.11	-968.26	-6.51 %
One adult >= 65, no children	13,884.70	13,884.70	0.00	0.00 %
- Female adult	13,884.70	13,884.70	0.00	0.00 %
- Male adult	13,884.70	13,884.70	0.00	0.00 %
One adult with children	14,360.90	13,479.11	-881.78	-6.14 %
- Female adult	14,361.26	13,479.49	-881.77	-6.14 %
- Male adult	14,359.40	13,477.58	-881.82	-6.14 %
Two adults < 65, no children	22,763.62	22,560.13	-203.49	-0.89 %
Two adults, at least one >= 65, no children	13,251.43	13,251.43	0.00	0.00 %
Two adults with one child	18,376.21	17,917.55	-458.66	-2.50 %
Two adults with two children	17,230.35	16,944.50	-285.85	-1.66 %
Two adults with three or more children	16,484.62	16,172.29	-312.33	-1.89 %
Three or more adults, no children	17,260.43	16,959.76	-300.67	-1.74 %
Three or more adults, with children	14,795.12	13,915.37	-879.74	-5.95 %
All	**16,913.21**	**16,534.47**	**-378.75**	**-2.24 %**

Exercise 7: Reforming PT Child Benefit by narrowing its age condition

The aim of this exercise is to learn how to use the function DefTU.

Steps:

1. Create a new system in Portugal based on 2020 system and name it PT_2020_E7;

2. Reform the tax unit used to calculate the Family Benefit so that families with children aged 14+ no longer receive this benefit;

3. Run EUROMOD to produce micro-outputs for these two systems (2020 and 2020_E7);

4. Analyse the results.

Description:

The aim of this exercise is to limit the provision of the Child benefit in 2020 only to families with children aged up to 13, i.e. families with children aged 14+ will lose the benefit in the reform scenario. The idea of making the payments age-related is being studied as part of a government shake-up.

Currently the benefit is paid up to the age of 16 or until 24 if a child is in full-time education. The Government poverty advisers have suggested that there should be a lower age limit for the child. Extra money could be given in the child's early years, and then taken away by the age of 14.

Your task is to implement the reform by creating and using a new Tax Unit.

Solution and further information

Task 1: Creating a new system

Our first task is to open the PT model and create a new system using the 2020 system as a base. Give the name PT_2020_E7 to the new system.

Task 2: Implementing a new policy

Before implementing a new policy, we should first identify and analyse the policy where the 'Child Benefit' is implemented in the PT for system year 2020 - how the benefit works, which tax unit is used for the calculation, how is the tax unit defined, etc.

Our next step is, to create a new Tax Unit: Add a *DefTu* function after the function which defines tu_bch_fa_pt.

The new tax unit should only identify children under 14 as dependents. The easiest way to do this is by using the relevant tax unit (tu_bch_fa_pt) in the policy *TuDef_PT* as a starting point. Name the new tax unit as **tu_buref_pt** (Figure 7.1) and add the appropriate parameters and their values as shown. Remember to set the new *DefTu* to 'on'.

Figure 7.1: Defining a new Tax Unit

bch_pt		on	on	**BEN: Family benefit (Abono de família para crianças e jovens)**
DefConst		on	on	Defines the benefit amounts for every bracket
DefVar		on	on	Intermediate variables
DefTu		on	on	tu_bch_fa_pt: tu for child benefit
Name		tu_bch_fa_pt	tu_bch_fa_pt	tu for child benefit
Type		SUBGROUP	SUBGROUP	
Members		Partner & OwnDepChild & LooseDepChild & DepParent	Partner & OwnDepChild & LooseDepChild & DepParent	
DepChildCond		!IsParent & idpartner=0 & ils_earns#1 = 0 & (dag<=16 | (dag>=17 & dag<=18 & dec>=2) | (dag>=19 & dag<=20 & dec>=3) | (dag>=21 & dag<=24 & dec>=6) | (dag>=16 & dag<=24 & ddi>0))	!IsParent & idpartner=0 & ils_earns#1 = 0 & (dag<=16 | (dag>=17 & dag<=18 & dec>=2) | (dag>=19 & dag<=20 & dec>=3) | (dag>=21 & dag<=24 & dec>=6) | (dag>=16 & dag<=24 & ddi>0))	Not above 16. It may extend until 24 (not above...) under certain conditions:; a) 17 or 18: need to attend primary education (1st to 6th grade) or greater;; b) 19 or 20: need to attend secondary education (7th to 12th grade) or greater;; c) Until 24: need to attend superior school or greater;; d) Also until 24 if disabled children and receiving disability allowances (not simulated).;
#_level	1	tu_individual_pt	tu_individual_pt	
AssignDepC...		yes	yes	Make sure that dependet children who are parents are not separated from their children
LoneParentC...		Default & !IsMarried	Default & !IsMarried	
DefTu		n/a	on	Reform: Benefit Unit
Name		n/a	tu_buref_pt	
Type		n/a	SUBGROUP	
Members		n/a	Partner & OwnDepChild & LooseDepChild	
DepChildCond		n/a	!IsParent & idpartner=0 & ils_earns#1=0 & dag<14	
#_Level	1	n/a	tu_individual_pt	
AssignDepC...		n/a	yes	
LoneParentC...		n/a	Default & !IsMarried	

Now, in the reform system, we need to modify the tax unit used for the calculation of the child benefit by using the new one. Locate the *bch_pt* policy and modify the *TAX_UNIT* as in Figure 7.3. Except for the TaxUnits, nothing else should be changed in the policy (Figure 7.2).

Figure 7.2: Using the new tax unit for the reform

fx	ArithOp		on	on	Equivalised tested income
	formula		il_bch / (nDepChildrenInTu + 1)	il_bch / (nDepChildrenInTu + 1)	
	output_var		i_bch_eqinc	i_bch_eqinc	equivalised income
	TAX_UNIT		tu_bch_fa_pt	tu_buref_pt	tu_bch_fa_pt: tu for child benefit
fx	BenCalc		on	on	Income brackets
	comp_cond	1	i_bch_eqinc <= ($SSI*14/12 * 0.5)	i_bch_eqinc <= ($SSI*14/12 * 0.5)	1) < 50% of minimum wage/SSI (14 pays per year)
	comp_perTU	1	1	1	
	comp_cond	2	i_bch_eqinc > ($SSI*14/12 * 0.5) & i_bch_eqinc <= ($SSI*14/12 * 1)	i_bch_eqinc > ($SSI*14/12 * 0.5) & i_bch_eqinc <= ($SSI*14/12 * 1)	2) <100% of minimum wage/SSI (14 pays per year)
	comp_perTU	2	2	2	
	comp_cond	3	i_bch_eqinc > ($SSI*14/12 * 1) & i_bch_eqinc <= ($SSI*14/12 * 1.5)	i_bch_eqinc > ($SSI*14/12 * 1) & i_bch_eqinc <= ($SSI*14/12 * 1.5)	3) <150% of minimum wage/SSI (14 pays per year)
	comp_perTU	3	3	3	
	comp_cond	4	i_bch_eqinc > ($SSI*14/12 * 1.5) & i_bch_eqinc <= ($SSI*14/12 * 2.5)	i_bch_eqinc > ($SSI*14/12 * 1.5) & i_bch_eqinc <= ($SSI*14/12 * 2.5)	4) <250% of minimum wage/SSI (14 pays per year)
	comp_perTU	4	4	4	
	comp_cond	5	i_bch_eqinc > ($SSI*14/12 * 2.5) & i_bch_eqinc <= ($SSI*14/12 * 5)	i_bch_eqinc > ($SSI*14/12 * 2.5) & i_bch_eqinc <= ($SSI*14/12 * 5)	5) <500% of minimum wage/SSI (14 pays per year)
	comp_perTU	5	5	5	
	output_var		i_bch_bracket	i_bch_bracket	income test bracket
	TAX_UNIT		tu_bch_fa_pt	tu_buref_pt	tu for child benefit
fx	BenCalc		on	on	No. of children aged 1 or 2
	comp_cond	1	(dag#1=1|dag#1=2) & IsDepChild	(dag#1=1|dag#1=2) & IsDepChild	assigns the # of children aged 1 or 2 to the tax unit
	comp_perElig	1	1	1	
	#_level	1	tu_individual_pt	tu_individual_pt	
	output_var		n_bch_children12	n_bch_children12	
	TAX_UNIT		tu_bch_fa_pt	tu_buref_pt	
fx	BenCalc		on	on	Benefit for children aged 0
	comp_cond	1	i_bch_bracket = 1 & dag#1 < 1	i_bch_bracket = 1 & dag#1 < 1	1st income bracket ; child < 1 years-old
	comp_perElig	1	$bch_ben_a1	$bch_ben_a1	
	comp_cond	2	i_bch_bracket = 2 & dag#1 < 1	i_bch_bracket = 2 & dag#1 < 1	2nd income bracket ; child < 1 years-old
	comp_perElig	2	$bch_ben_a2	$bch_ben_a2	
	comp_cond	3	i_bch_bracket = 3 & dag#1 < 1	i_bch_bracket = 3 & dag#1 < 1	3rd income bracket ; child < 1 years-old
	comp_perElig	3	$bch_ben_a3	$bch_ben_a3	
	comp_cond	4	i_bch_bracket = 4 & dag#1 < 1	i_bch_bracket = 4 & dag#1 < 1	4th income bracket ; child < 1 years-old
	comp_perElig	4	$bch_ben_a4	$bch_ben_a4	
	comp_cond	5	n/a	n/a	5th income bracket ; child < 1 years-old
	comp_perElig	5	n/a	n/a	
	#_level	1	tu_individual_pt	tu_individual_pt	
	output_var		bch_s	bch_s	
	TAX_UNIT		tu_bch_fa_pt	tu_buref_pt	
fx	BenCalc		on	on	Benefit for children aged 1 or more (1 or

Task 3: Producing the micro-output

Run EUROMOD for the baseline PT_2020 and reform PT_2020_E7 systems using the PT_*training_data* dataset.

Task 4: Analyse the results

Use the *Statistics Presenter* (Baseline/Reform option) to analyse the distributional impact of the reform.

Figure 7.3: Statistics Presenter Results, distributional effect

2.2.a Mean equivalised income (monthly) by decile groups

	PT_2020 (base)	PT_2020_E7	Difference to base	% Difference to base
Decile 1	399.62	384.21	-15.40	-3.85 %
Decile 2	608.76	599.11	-9.65	-1.59 %
Decile 3	759.22	755.76	-3.47	-0.46 %
Decile 4	880.24	878.56	-1.68	-0.19 %
Decile 5	1,004.59	1,004.79	0.21	0.02 %
Decile 6	1,130.88	1,130.60	-0.29	-0.03 %
Decile 7	1,268.71	1,269.27	0.56	0.04 %
Decile 8	1,431.13	1,431.53	0.39	0.03 %
Decile 9	1,631.78	1,631.85	0.06	0.00 %
Decile 10	1,988.83	1,988.83	0.00	0.00 %
All	**1,107.84**	**1,104.90**	**-2.95**	**-0.27 %**
Poor	464.13	453.40	-10.72	-2.31 %

Figure 7.4: Statistics Presenter Results, budgetary effects

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	PT_2020 (base) (annual)	PT_2020_E7 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**70,807.13**	**70,807.13**	**0.00**	**0.00 %**
... income from (self) employment	70,807.13	70,807.13	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**33,860.71**	**33,860.71**	**0.00**	**0.00 %**
... direct taxes	10,203.86	10,203.86	0.00	0.00 %
... employee social insurance contributions	7,255.79	7,255.79	0.00	0.00 %
... self-employed social insurance contributions	735.45	735.45	0.00	0.00 %
... other social insurance contributions	0.00	0.00	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	15,665.62	15,665.62	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**0.00**	**0.00**	**0.00**	**0.00 %**
Government expenditure on social transfers	**34,887.06**	**34,692.07**	**-194.99**	**-0.56 %**
by target group				
... unemployment benefits	6,312.04	6,312.04	0.00	0.00 %
... family and education benefits	759.30	564.31	-194.99	-25.68 %
... social assistance and housing benefits	438.91	438.91	0.00	0.00 %
... pensions, health and disability benefits	27,376.80	27,376.80	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	1,198.21	1,003.22	-194.99	-16.27 %
... non-means-tested non-pension benefits	6,312.04	6,312.04	0.00	0.00 %
... pensions	27,376.80	27,376.80	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

Exercise 8: Introducing the Belgian social insurance contributions for pensioners & survival pensioners in Portugal

The aim of this exercise is to learn how to implement a policy swap between two countries.

Steps:

1. Create a new system in Portugal based on the 2019 system and name it PT_2019_E8;

2. Simulate what the impact of introducing a new social insurance contribution for pensioners & survival pensioners in Portugal would be, following the same structure as the one that exists in Belgium;

3. Run EUROMOD to produce micro-outputs for these two systems (PT_2019 and PT_2019_E8);

4. Analyse the results.

Description:

The high cost of nursing homes in Portugal has led the government to consider introducing a new social insurance contribution with the same structure as the one that exists in Belgium. This will then be used to help individuals to pay for the cost of care in old age. Perform this policy swap exercise by replicating, in the 2019 Portuguese system, the social insurance policy for pensioners and survival pensioners of Belgium in place in 2019.

Hints:

The aim of this exercise is to create a social insurance contribution – **tscpe_pt -** for 2019 which will have the same structure as the one in the Belgium model **- tscpe_be**.

- You will need to replace the tax unit **tu_family_be** used in Belgium with the unit **tu_bch_fa_pt** used in Portugal and also make an equivalent change for **tu_individual_be**.
- Instead of the income list **il_pension** used in Belgium you can use the income list **ils_pen** defined for Portugal.
- Remember to include the new contribution in one of the standard income lists (e.g. in the income list for "Other Social Insurance Contributions" **ils_sicot**).

Solution and further information

Task 1: Creating a new system

Our first task is to open the Portuguese model and create a new system using the 2019 system as a base. Give the name PT_2019_E8 to the new system.

Then, we should open Belgium as well without making any changes.

Task 2: Implementing a new policy

Before implementing a new policy, we should first analyse the tax-benefit system in both countries – Belgium and Portugal. Find policy **tscpe_be** in the Belgium model and see how the social insurance contribution is created – where it is in the policy spine, how it is connected with other policies, what its elements, tax units, etc. are.

After familiarising yourselves with the different policy spines, start by implementing this contribution in the Portuguese system simply by copying the Belgian *tscpe_be* policy by right- clicking on it and using the option *Copy Policy* from the context (Figure 8.1).

Figure 8.1: Copying the policy in the Belgium model

Based on the analyses of both models (Belgium and Portugal), we know that our new social contribution must be created between *Self-employed social insurance contribution (repetition of policy with order 23)* - **tscse_pt** - and *Unemployment insurance (Subsídio de desemprego)* - **bunct_pt** (Figure 8.2).

Figure 8.2: Pasting the policy to the Portuguese model

When we chose the option *Paste Policy After* we will be asked to give a (new) name to the policy. In our case, we want to keep the same name, but we will need to change the country acronym - **tscpe_pt** (Figure 8.3).

Figure 8.3: Changing the name to the model specific country acronym

When we use *paste* we should do it in a way to link the BE_2019 and PT_2019_E8 systems. After clicking *ok*, we will get a small window where we will see how the systems are connected. If the systems are connected in a way that we want, we should choose option *ok* (Figure 8.4).

Figure 8.4: Assign systems

If they are not, then we should point our cursor to the Origin cell and with a left-click, we will get a window with all available systems. Simply by ticking the system which we want, we can create a combination which is the most suitable for what we are trying to do.

In our case, we are only interested in the PT_2019_E8 system, so we will keep this new policy *on* only for this system, and for all other systems we will set it as *n/a (all components)* (Figure 8.5). Also, we might wish to modify the *Comment* column, in case it contains information related to the Belgium model and data.

Figure 8.5: Setting the policy to n/a for the baseline 209 system

Policy	Grp/No	PT_2019	PT_2019_E8	Comment
yempb_pt		off	off	INC: Public wages cuts
poacm_pt		off	off	BEN: Minimum pension (Pensoes mínimas)
pcuts_pt		n/a	n/a	INC: Pensions cuts and solidarity contributions
neg_pt		on	on	DEF: recode negative self-employment income to zero
yemcomp_pt		n/a	n/a	BEN: Wage compensation scheme Covid-19
tscee_pt		on	on	SIC: Employee social insurance contributions
tscer_pt		on	on	SIC: Employer social insurance contribution
tscse_pt		on	on	SIC: Self-employed social insurance contribution
ysecomp_pt		n/a	n/a	BEN: Self-employment income compensation scheme Covid-19
tscse_pt		on	on	SIC: Self-employed social insurance contribution (repetition of policy with order 20)
tscpe_pt		on	on	SIC: pensioners contributions to health and disability insurance and solidarity contribution
bunct_pt		off / on / toggle / n/a / n/a (all components)	on	BEN: Unemployment insurance (Subsídio de desemprego) PART-SIMULATED
bunnc_pt			on	BEN: Unemployment assistance (subsidio social de desemprego) PART-SIMULATED

Then, work as follows:

- Make all systems visible. Make sure that the policy is set as n/a for all components for all systems except for PT_2019_E8. Then, you can either set it as n/a for each system individually or to avoid repeating this process for all systems, use hot-key ***Alt+s*** (spread). This option will copy the values to all cells in an observed row. Remember to set it to *on* for PT_2019_E8.

- Next, make all the necessary adjustments in tax units and income lists to make sure this new policy runs and affects disposable income in the correct way. The easiest way to do so is by using the function *Search and Replace* which you can find in the tab *Country Tools* in the ribbon bar – (Figure 8.6).

Figure 8.6: Searching for Belgium related information

What do we need to *find and replace*?

- After analysing both models, we know that the main thing to do is to change tax units. Instead of the tax unit family in Belgium - **tu_family_be** - we should use the unit already defined in Portugal - **tu_bch_fa_pt** (Figure 8.7).
- The next tax unit that needs to be changed is **tu_individual_be**. Instead of this tax unit, we should use **tu_individual_pt**.
- Then, instead of the income list **il_pension** in Belgium you should use an income list already defined in Portugal - **ils_pen**.

Figure 8.7: Replacing Belgium related information

The new calculated variable will only affect disposable income if it is included in the relevant standardized income lists. In our case it should be included in *Other Social Insurance Contributions* - **ils_sicot** (Figure 8.8).

Figure 8.8: Adding the new social insurance contribution to the correct income list

fx DefIl	on	on	other SIC
Name	ils_sicot	ils_sicot	
txcpe_s	+	+	extraordinary solidarity contribution on pensions
tscpe_s	n/a	+	

Task 3: Producing the micro-output

Run EUROMOD –for the baseline PT_2019 and reform PT_2019_E8 systems using PT_*training_data* as the dataset.

Task 4: Analyse the results

Use the *Statistics Presenter Tool* (Baseline/Reform option) to analyse the budgetary and distributional impact of the policy swap. Figure 8.9 shows the budgetary impact.

Figure 8.9: Budgetary impact

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	PT_2019 (base) (annual)	PT_2019_E8 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**70,807.13**	**70,807.13**	**0.00**	**0.00 %**
... income from (self) employment	70,807.13	70,807.13	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**33,858.77**	**34,262.97**	**404.20**	**1.19 %**
... direct taxes	10,201.91	10,201.91	0.00	0.00 %
... employee social insurance contributions	7,255.79	7,255.79	0.00	0.00 %
... self-employed social insurance contributions	735.45	735.45	0.00	0.00 %
... other social insurance contributions	0.00	404.20	404.20	N/A
... employer social insurance contributions (not part of disposable income)	15,665.62	15,665.62	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**0.00**	**0.00**	**0.00**	**0.00 %**
Government expenditure on social transfers	**34,752.82**	**34,752.82**	**0.00**	**0.00 %**
by target group				
... unemployment benefits	6,312.04	6,312.04	0.00	0.00 %
... family and education benefits	728.73	728.73	0.00	0.00 %
... social assistance and housing benefits	443.55	443.55	0.00	0.00 %
... pensions, health and disability benefits	27,268.48	27,268.48	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	1,172.28	1,172.28	0.00	0.00 %
... non-means-tested non-pension benefits	6,312.04	6,312.04	0.00	0.00 %
... pensions	27,268.48	27,268.48	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

Exercise 9: Reforming the health tax in Denmark

The aim of this exercise is to learn how to use the function SchedCalc.

Steps:

1. Create a new system in Denmark based on 2018 system and name it DK_2018_E9;

2. Modify the Health Tax in order to make the tax allowance non-transferable between partners and increase the progressivity of the tax adding a 10% marginal tax rate for income above 300,000 Dkr per year

3. Run EUROMOD to produce micro-outputs for these two systems (2018 and 2018_E9);

4. Analyse the results.

Description:

In the 2018 system, the Danish Health Tax consists of a 1% contribution paid by each individual. The tax base consists of the "Taxable Income" and the tax schedule includes a tax allowance equal to the General Personal Allowance. The unused individual allowance can be transferred between spouses.

The aim of this exercise is implementation and analysis of the distributional effect of a reform consisting of making the unused allowance non-transferable between spouses and including a 10% marginal tax rate for income above 300,000 Dkr per year.

The first step should be the analysis of the structure of the Health Tax in Denmark, focusing on transferability of the tax allowance between partners and tax schedule.

Solution and further information

Task 1: Creating a new system

Our first task is to open Denmark and create a new system using the 2018 system as a base. Give the name dk_2018_E9 to the new system.

Task 2: Implementing a new policy

Before implementing a new policy, we should first identify and analyse the policy where the 'Health Tax' is implemented in Denmark for system year 2018, with a special focus on the function(s) where the unused part of the allowance is transferred between spouses.

We should observe what the tax base is, where in the policy the unused tax allowance is calculated, where it is transferred, how the tax liability is computed, etc.

Our next step is to modify the policy in order to make the allowance non-transferable between spouses. The way to end the transfer of the unused allowance is simply by switching off the relevant function in the thl_dk policy (Figure 9.1).

Figure 9.1: Abolishing the transfer of the partner's allowance

thl_dk		**on**	**on**	**TAX: Health Contribution (Sundhedsbidrag)**
DefVar		on	on	
ArithOp		on	on	
formula		il_tin	il_tin	
output_var		ttbhl_s	ttbhl_s	
TAX_UNIT		tu_individual_dk	tu_individual_dk	
ArithOp		on	off	Transferable unused personal allowance calculation
formula		GenPersAllowance-ttbhl_s	GenPersAllowance-ttbhl_s	
output_var		i_thl	i_thl	Unused allowance amount
uplim		GenPersAllowance	GenPersAllowance	
TAX_UNIT		tu_individual_dk	tu_individual_dk	
BenCalc		on	off	Tax base after the transfer of partner's unused allowance
comp_cond	1	i_thl<0 & IsMarried & GetPartnerIncome#1>0	i_thl<0 & IsMarried & GetPartnerIncome#1>0	If the partner does not use all the personal allowance...
comp_perTU	1	(GetPartnerIncome#1)*(-1)	(GetPartnerIncome#1)*(-1)	The tax base is reduced accordingly
#_income	1	i_thl	i_thl	
output_add_...		ttbhl_s	ttbhl_s	
TAX_UNIT		tu_individual_dk	tu_individual_dk	
SchedCalc		on	on	Health Contribution
base		ttbhl_s	ttbhl_s	The tax base is the taxable income
band_lowlim	1	GenPersAllowance	GenPersAllowance	General Personal Allowance
band_rate	1	$thl	$thl	
output_var		thl_s	thl_s	
TAX_UNIT		tu_individual_dk	tu_individual_dk	

The second part of our task is to include a 10% marginal tax rate for income above 300,000 Dkr per year. How do we do that?

One of the most used functions in EUROMOD (mainly for progressive taxes) is *SchedCalc*. As we can see from the observed policy, this function usually contains a tax base (*base*), tax bands (*band_upLim*/*band_lowlim*) and a tax rate (*band_rate*). Also, for fixed amounts, instead of rates we need to use *band_amount* and the component *simple_prog* is used to apply the highest marginal tax rate reached by base on the whole income.

In our case, we need to modify this function by adding new components so as to create the lower limit from which the new rate will be applied. By using *Add Parameter Form*, we can add additional components *band_lowlim* and *band_rate*, thus defining a new tax band (Figure 9.2).

Save your changes.

Figure 9.2: Adding the new band and rate

thl_dk		**on**	**on**	**TAX: Health Contribution (Sundhedsbidrag)**
DefVar		on	on	
ArithOp		on	on	
ArithOp		on	off	Transferable unused personal allowance calculation
BenCalc		on	off	Tax base after the transfer of partner's unused allowance
SchedCalc		on	on	Health Contribution
base		ttbhl_s	ttbhl_s	The tax base is the taxable income
band_lowlim	1	GenPersAllowance	GenPersAllowance	General Personal Allowance
band_rate	1	$thl	$thl	
Band_LowLim	2	n/a	300000#y	
Band_Rate	2	n/a	0.10	
output_var		thl_s	thl_s	
TAX_UNIT		tu_individual_dk	tu_individual_dk	

Task 3: Producing the micro-output

Run EUROMOD for the baseline DK_2018 and reform DK_2018_E9 systems using *DK_training_data* as the dataset.

Task 4: Analyse the results

Use the *Statistics Presenter* (Baseline/Reform option) to analyse the fiscal and distributional impact of the reform. Figure 9.3 shows the impact on inequality indices.

Figure 9.3: Impact on inequality

3.1 Basic Inequality Indices

	Gini DK_2018	S80/S20 DK_2018	Gini DK_2018_E9	Diff. Gini DK_2018_E9/DK_2018	S80/S20 DK_2018_E9	Diff. S80/S20 DK_2018_E9/DK_2018
Original Income	0.5394	7.4679	0.5394	0.0000	7.4724	0.0044
Original Income after Taxes/SIC	0.5481	7.4413	0.5437	-0.0044	7.0937	-0.3476
Original Income incl. Public Pensions after Taxes/SIC	0.3630	6.7240	0.3538	-0.0092	6.4059	-0.3181
Disposable Income	0.2204	3.0035	0.2093	-0.0111	2.8674	-0.1361

Exercise 10: New means-tested income support for families with children in education in Croatia

The aim of this exercise is to refresh existing knowledge about introducing new policies – benefits and taxes.

Steps:

1. Create a new system in Croatia based on the 2021 system and name it HR_2021_E10.

2. Implement the 'Income support to families with children in education'.

3. Run EUROMOD to produce micro-outputs for these two systems (2021 and 2021_E10).

4. Analyse the results.

Description:

The aim of this exercise is to create a new means-tested benefit for 2021 – **bched_hr -** which will be given to families with children in education (any level, not just the compulsory primary and lower secondary levels): 380 HRK per month per entitled dependent child.

The tax unit used to assess the means test will be that of the personal income tax, i.e. **tu_tin_hr**. Consistently, the definition of dependent children will be also the one used in this tax unit.

Educational level is stored in the variable **dec**. A value of 0 indicates that the person is not in education and a value higher than 0 indicates that the person is in education.

The household incomes used in the income test are earnings and public pensions. The income threshold is set to 60,000 HRK per year, increased by 3,800 HRK per year per dependent child.

The financing of this reform will come from introducing an 'Extraordinary contribution' (**txctx_hr**) at the end of the policy spine, where individuals with annual disposable incomes (**ils_dispy**) exceeding 100,000 HRK will pay an extraordinary contribution equal to 1.5% of disposable income.

Solution and further information

Task 1: Creating a new system

Our first task is to open Croatia and create new system using the 2021 system as a base. Give the name HR_2021_E10 to the new system.

Task 2: Implementing new policies

Before implementing a new policy, we should first analyse the tax-benefit system in Croatia – what the relevant policies are, how they are constructed, how they are connected, which income list(s) they use, which child benefits exist, which tax units are used, what variables are outputted, etc.

Introducing a new benefit – bched_hr

We should create a new benefit - **bched_hr -** after the *Compensation for electricity costs* (bhout_hr) which is the last benefit simulated in 2021.

For the simulation of this benefit, you will have to create an income list (the household incomes used in means testing are earnings and public pensions) which will be used for the income test and name it **il_bched** (Figure 10.1).

Figure 10.1: Adding the new policy and first function

bched_hr		n/a	on	**BEN: means-tested income support for families with children**
fx DefIl		n/a	on	household incomes used in means-testing
Name		n/a	il_bched	
ils_earns		n/a	+	
ils_pen		n/a	+	

In order to implement the means testing condition, you need to calculate the income threshold. As we said, the income threshold is set to 60,000 HRK per year, increased by 3,800 HRK per year per dependent child. The tax unit which will be used is **tu_tin_hr**. The function *ArithOp* will help you do that. However, you need to use a temporary variable (e.g. *i_bched*) to store the calculation, which needs to be initialised with a *DefVar* function, before *ArithOp*. See both functions in Figure 10.2.

Figure 10.2: Adding the income test

bched_hr		n/a	on	**BEN: means-tested income support for families with children**
fx DefIl		n/a	on	household incomes used in means-testing
fx DefVar		n/a	on	intermediate variable
i_bched	1	n/a	0	
fx ArithOp		n/a	on	income threshold
Formula		n/a	60000#y + nDepChildrenInTu*3800#y	60,000 HRK per year, increased by 3800 per dependent child
Output_Var		n/a	i_bched	
TAX_UNIT		n/a	tu_tin_hr	like for the personal inocme tax

The next step is to implement the eligibility conditions and set the value of the benefit. This benefit should be given to the families with children in any level of education. The amount is 380 HRK per month per entitled dependent child. All this can be done in a single *BenCalc* function (Figure 10.3).

Figure 10.3: Calculate the benefit amount

bched_hr		n/a	on	**BEN: means-tested income support for families with children**
DefIl		n/a	on	household incomes used in means-testing
DefVar		n/a	on	intermediate variable
ArithOp		n/a	on	income threshold
BenCalc		n/a	on	benefit amount
Comp_Cond	1	n/a	IsDepChild & dec! =0 & il_bched<i_bched	children in compulsory education with family taxable income below the threshold
Comp_perElig	1	n/a	380#m	… amount per eligible dependent child
Output_Var		n/a	bched_s	
TAX_UNIT		n/a	tu_tin_hr	

The new calculated variable bched_s will only affect disposable income if is included in the relevant standardized income lists. In our case it should be included in means-tested benefits (**ils_benmt**) for the reformed system (HR_2021_E10).

Introduce a new tax – txcxt_hr

The second part of this exercise is the introduction of a new tax ('Extraordinary contribution', **txcxt_hr**), where individuals with annual disposable incomes over 100,000 HRK will pay an extraordinary contribution equal to 1.5% of disposable income.

You should create a new policy by using the same steps as for the introduction of a new benefit, doing it right after the newly-created benefit. The reason for this is that our condition to apply the new tax includes disposable income. As you already know, disposable income includes all benefits, so we need to create and simulate all of them before simulating the new tax (Figure 10.4).

This tax should be also added to the list of simulated taxes (**ils_taxsim**), as a way to make it part of disposable income. As you may have realised, there is a circular reference here, since the extraordinary contribution is part of ils_dispy (through ils_taxsim), but ils_dispy is also used to compute the extraordinary contribution. EUROMOD solves this issue by giving value zero to all variables that hasn't been created yet when the income list is used, but will issue a warning. In order to avoid this warning, we may initialise the variable immediately before, as we do in row 34.1.

However, at the end of the spine all income lists are recomputed, so the extraordinary contribution will be finally deducted from ils_dispy, and this is the value that will be part of the output microdata.

Figure 10.4: Adding the new extraordinary contribution policy

txcxt_hr		n/a	on	TAX: Extraordinary contribution
ArithOp		n/a	on	initialise result variable as this itself is part of disposable income
Formula		n/a	0	
Output_Var		n/a	txcxt_s	
TAX_UNIT		n/a	tu_individual_hr	
BenCalc		n/a	on	tax
Comp_Cond	1	n/a	ils_dispy>100000#y	individuals with annual disposable incomes over 100,000 HRK only
Comp_perTU	1	n/a	ils_dispy*1.5%	… pay a contribution of 1.5%
Output_Var		n/a	txcxt_s	
TAX_UNIT		n/a	tu_individual_hr	

Task 3: Producing the micro-output

Run EUROMOD for the baseline HR_2021 and reform HR_2021_E10 systems using HR_*training_data*.

Task 4: Analyse the results

Use the *Statistics Presenter* (Baseline/Reform) to analyse the distributional impact of the reform.

Figure 10.5: Distributional impact

2. Distributional

2.1.a Mean household income (monthly) by decile groups

	HR_2021 (base)	HR_2021_E10	Difference to base	% Difference to base
Decile 1	319.29	349.38	30.09	9.42 %
Decile 2	660.60	685.69	25.09	3.80 %
Decile 3	749.67	751.48	1.82	0.24 %
Decile 4	1,235.98	1,239.59	3.62	0.29 %
Decile 5	1,233.96	1,236.53	2.58	0.21 %
Decile 6	1,536.79	1,537.63	0.84	0.05 %
Decile 7	1,653.54	1,653.54	0.00	0.00 %
Decile 8	1,796.30	1,796.30	0.00	0.00 %
Decile 9	1,916.78	1,916.78	0.00	0.00 %
Decile 10	2,398.25	2,398.25	0.00	0.00 %
All	**1,233.85**	**1,241.68**	**7.84**	**0.64 %**
Poor	469.36	474.87	5.51	1.17 %

3.2.a Basic Poverty Indices

	Poverty Risk for HR_2021 (base)	Poverty Risk for HR_2021_E10	Difference to base
At-risk-of-poverty rate	19.77 %	18.92 %	-0.85pp
Median at-risk-of-poverty gap	33.15 %	31.86 %	-1.29pp
Fixed Poverty line (monthly)	428.77		

Exercise 11: Introducing an In-Work Benefit in Cyprus

The aim of this exercise is to recap your knowledge on introducing new policies – benefits and taxes.

Steps:

1. Create a new system in Cyprus based on the 2020 system and name it CY_2020_E11;

2. Introduce a new means-tested benefit for families with children under 16 years of age, where partners together need to earn at least 30,000 euros per year; use DefTU, Elig and BenCalc functions for that.

3. Run EUROMOD to produce micro-outputs for these two systems (CY_2020 and CY_2020_E11);

4. Analyse the results.

Description:

Cyprus is introducing a new 'In Work Benefit' for parents (**bwkmt_cy)**. The idea is to give a benefit to families with dependent children if both parents or lone parents are low or medium income earners. A child is dependent if it is younger than 16 (below compulsory school age). In order to be eligible, each parent needs to be liable to income tax (**tin_s>0**). The benefit amount depends on the joint earnings (**ils_earns**) of the parents and the number of dependent children. Parents with joint earnings:

- of €30,000 and below €40,000 per year are entitled to receive €600 per child per year;
- of €40,000 and below €50,000 per year are entitled to receive €500 per child per year;
- of €50,000 and below €60,000 per year are entitled to receive €400 per child per year;

The threshold is divided by 2 for lone parents. In addition, this benefit is not restrictive, i.e. receiving this benefit does not affect the amounts of other benefits.

Your task is to implement the new benefit by using functions **DefTU**, **Elig** and **BenCalc**. Remember to include the new benefit in the relevant income lists (**ils_bensim** and **ils_benmt**).

Solution and further information

Task 1: Creating a new system

Our first task is to open Cyprus and create a new system using the 2020 system as a base. Give the name CY_2020_E11 to the new system.

Task 2: Implementing a new policy

Before implementing a new policy, we should analyse the tax-benefit system in Cyprus.

One of the ways of implementing this benefit is to create a new Tax Unit and name it **tu_inwork_cy.** Eligible families are those with children under 16 years of age.

To get an idea of how to create a new Tax Unit you could have a look how other tax units are defined under the policy **tudef_cy**. The first step is to add a function *DefTu* by pointing the cursor at new policy, right-clicking and, among *System Functions*, selecting the one that we need (Figure 11.1).

Figure 11.1: Adding a new Tax Unit

By default, we only get two components - *Type* and *Name*. As we can see from the other tax units, most of them have the component *Members*. Our entitled family should have two parents or a lone parent who is not married with dependent children in the household (both own and loose children).

Also, when we create a new tax unit, we need to specify its type. In our case it will be a *SUBGROUP*. New components could be created/added by using the *Add Parameter Form*.

Another way of doing this is simply to copy-paste an existing tax unit - which should be similar to ours - and then modify it, e.g. **tu_bch_cy** (Figure 11.2). Remember to set the new *tu_inwork_cy* tax unit to 'on'.

Figure 11.2: Specifying the new Tax Unit

After defining the tax unit, our next step should be to create the new benefit. Based on the information that the new benefit is not restrictive, we create it at the end of the policy spine to make sure that it does not enter the means test in other benefits and we name it **bwkmt_cy**.

The next step is to implement the eligibility conditions and amounts using functions *Elig* and *BenCalc*. Following the Modelling Convention rules, name the output variable as **bwkmt_s**. Please note that we need to do this for couple parents and for lone parent separately (Figure 11.3). Do not forget to use "Output_Add_Var" in the second BenCalc, and set the policy and its functions to on for the reform system.

Figure 11.3: Modelling the new in-work benefit

bwkmt_cy		n/a	on	BEN: In-work benefit for parents
fx Elig		n/a	on	Eligibility for couples
Elig_Cond		n/a	tin_s#1>0 & IsWithPartner#1	
#_Level	1	n/a	tu_individual_cy	
TAX_UNIT		n/a	tu_inwork_cy	
fx BenCalc		n/a	on	
Who_Must_...		n/a	all_adults	Both parents must be eligible
Comp_Cond	1	n/a	ils_earns>=30000#y & ils_earns<40000#y	Earnings between
Comp_perTU	1	n/a	nDepChildrenInTu*600#y	... each child receives 600 per year
Comp_Cond	2	n/a	ils_earns>=40000#y & ils_earns<50000#y	Earnings between
Comp_perTU	2	n/a	nDepChildrenInTu*500#y	... each child receives 500 per year
Comp_Cond	3	n/a	ils_earns>=50000#y & ils_earns<60000#y	Earnings between
Comp_perTU	3	n/a	nDepChildrenInTu*400#y	... each child receives 400 per year
Output_Var		n/a	bwkmt_s	
TAX_UNIT		n/a	tu_inwork_cy	
fx Elig		n/a	on	Eligibility for lone parents
Elig_Cond		n/a	tin_s#1>0 & IsLoneParent#1	
#_Level	1	n/a	tu_individual_cy	
TAX_UNIT		n/a	tu_inwork_cy	
fx BenCalc		n/a	on	
Who_Must_...		n/a	all_adults	Both parents must be eligible
Comp_Cond	1	n/a	ils_earns>=15000#y & ils_earns<20000#y	Earnings between
Comp_perTU	1	n/a	nDepChildrenInTu*600#y	... each child receives 600 per year
Comp_Cond	2	n/a	ils_earns>=20000#y & ils_earns<25000#y	Earnings between
Comp_perTU	2	n/a	nDepChildrenInTu*500#y	... each child receives 500 per year
Comp_Cond	3	n/a	ils_earns>=25000#y & ils_earns<30000#y	Earnings between
Comp_perTU	3	n/a	nDepChildrenInTu*400#y	... each child receives 400 per year
Output_Var		n/a	bwkmt_s	
TAX_UNIT		n/a	tu_inwork_cy	

The new benefit is a means-tested benefit. Before running the model, we need to add this benefit to the appropriate income lists - **ils_bensim**, **ils_benmt** in our new system (Figure 11.4).

Figure 11.4: Adding the new benefit to income lists

fx DefIl		on	on	means-tested benefits
name		ils_benmt	ils_benmt	
bsa_s		n/a	n/a	social assistance benefit (δημόσιο)
bsamm_s		+	+	guaranteed minimum income benefit (ελάχιστο εγγυημένο εισόδημα)
bsaoa_s		+	+	low pension benefit
bsals_s		+	+	Easter Beneft (ΙΠασχαλινό Επίδομα)
bch01_s		n/a	n/a	child benefit - supplementary part (επίδομα τέκνου)
bho		+	+	housing benefits
bedet_s		+	+	student grant (φοιτητική χορηγία)
bchlp_s		+	+	single parent child allowance (επίδομα μονογονιού)
bch_s		+	+	child benefit (επίδομα τέκνου)
bwkmt_s		n/a	+	in-work benefit

Task 3: Producing the micro-output

Run EUROMOD for the baseline cy_2020 and reform cy_2020_E11 systems using CY*_training_data*.

Task 4: Analyse the results

Use the *Statistics Presenter* (baseline/reform option) to analyse the budgetary and distributional impact of the reform. Figure 11.5 shows the budgetary impact.

Figure 11.5: Budgetary impact

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	CY_2020 (base) (annual)	CY_2020_E11 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**8,132.03**	**8,132.03**	**0.00**	**0.00 %**
... income from (self) employment	8,132.03	8,132.03	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**2,496.57**	**2,496.57**	**0.00**	**0.00 %**
... direct taxes	388.19	388.19	0.00	0.00 %
... employee social insurance contributions	828.13	828.13	0.00	0.00 %
... self-employed social insurance contributions	103.39	103.39	0.00	0.00 %
... other social insurance contributions	50.01	50.01	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	1,126.86	1,126.86	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**866.81**	**866.81**	**0.00**	**0.00 %**
Government expenditure on social transfers	**2,534.73**	**2,535.54**	**0.81**	**0.03 %**
by target group				
... unemployment benefits	0.63	0.63	0.00	0.00 %
... family and education benefits	126.82	126.82	0.00	0.00 %
... social assistance and housing benefits	520.01	520.01	0.00	0.00 %
... pensions, health and disability benefits	1,887.26	1,887.26	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	643.89	644.70	0.81	0.13 %
... non-means-tested non-pension benefits	3.57	3.57	0.00	0.00 %
... pensions	1,887.26	1,887.26	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

Exercise 12: Revenue-neutral reform in Simpleland

The aim of this exercise is to learn about special functions such as Loop, ChangeParam and Totals, and create a revenue-neutral reform.

Steps:

1. Create a new system in Simpleland based on the SL_1996 system and name it SL_1996_E12;

2. Set up a function Loop in the beginning of this block. Then, add relevant steps to the end of the loop:

- calculation of total (weighted) disposable income in the baseline and in each iteration,
- calculation of the difference of the two,
- adjustment of income tax rate,
- changing relevant policy parameters;

3. Run EUROMOD to produce micro-outputs for these two systems (SL_1996 and SL_1996_E12);

4. Analyse the results.

Description:

Simpleland is an additional hypothetical tax-benefit system, whose purpose is to make learning and teaching EUROMOD easier. This has also served as a "sketch" for the New Member States' prototype models. SL provides a simple version of policies like social insurance contributions (distinguishing employer, employee and self-employed), income tax (single flat rate with a general allowance), universal child benefit and means-tested social assistance and is based on a synthetic database of hypothetical households.

A revenue neutral policy presents changes in the tax laws that result in no change in the amount of revenue coming into the government's coffers. For example, a provision may require individuals to pay less tax, but corporations will pay correspondingly more taxes. The aim of this exercise is to implement a budget neutral reform which doubles the child benefit amount in SL and funds it by raising the income tax rate, and to analyse its distributional effects.

Budget neutrality also takes into account interactions with the social assistance, i.e. the aggregate disposable income should remain the same. The new income tax rate is obtained through a loop which repeats the calculations until converging on a solution with required precision – total (weighted) disposable income should differ by less than 0.01% from its baseline value.

Solution and further information

Task 1: Creating new system

Our first task is to open Simpleland (SL in the list of countries) and create a new system using the SL_1996 system as a base. Give the name SL_1996_E12 to the new system.

Task 2: Implementing a new policy

The looping functions allow repeating part (or all) of the tax-benefit calculations. As an example, for calculating marginal tax rates at least part of a policy needs to be calculated twice, once for original income and once for marginally increased income. Function *Loop* allows for such a loop over a group of policies.

The loop is carried out until the number of scheduled iterations is reached and/or the break condition is fulfilled. As certain calculations may depend on the current iteration a variable called *loopcount_loopid* is provided. If for example the identifier (parameter *loop_id*) of the loop for the marginal tax rate calculations is *mtr*, the variable *loopcount_mtr* will take a value of 1 in the first loop and 2 in the second.

This allows for a respective condition to increase income in the second loop. Moreover, if certain policies within the loop should not be repeated, they can be switched off after the first iteration, by using *run_cond*: *loopcount_loopid=1*.

Before implementing a new policy, we should first find relevant existing policies (child benefit, income tax) in the spine and parameters of interest (benefit amount, tax rate). Then we should see which policies need to be run repeatedly to achieve revenue neutrality in terms of (aggregate) disposable incomes. Simpleland has a child benefit (equal to 200 units per month) and a flat income tax with a 20% marginal tax rate.

Our next step is to add a new *Definition* policy, before the child benefit policy (bch_sl), to define the iterative loop and name it **loopdef_sl**. In this policy we need to define two new (intermediate) constants (Add function>System Functions>DefConst):

- adjusted income tax rate (**rate_pit**), and
- initial value of the child benefit (**amt_bch**).

The initial value for the income tax rate and child benefit should be the same as the existing rate and amount in the base system (Figure 12.1).

Figure 12.1: Defining the iterative loop

8		loopdef_sl		n/a	on	DEF: Constants needed for the loop
8.1	fx	DefConst		n/a	on	
8.1.1		amt_bch	1	n/a	200#m	
8.1.2		rate_pit	2	n/a	0.2	
9		bch_sl		on	on	BEN: Child Benefit

Then we should replace the current values of the child benefit and the income tax rate with the newly created constants (Figure 12.2).

Figure 12.2: Adding the new constants

9	bch_sl		on	on	BEN: Child Benefit
9.1	ArithOp		on	on	benefit calculation
9.1.1	formula		nDepChildrenInTU*amount#1	nDepChildrenInTU*amount#1	
9.1.2	#_amount	1	200#m	amt_bch	
9.1.3	output_var		bch_s	bch_s	
9.1.4	TAX_UNIT		tu_sben_family_sl	tu_sben_family_sl	There are age limits set for dep. children in TU.
10	tin_sl		on	on	TAX: Income Tax
10.1	SchedCalc		on	on	
10.1.1	base		il_TaxableY	il_TaxableY	
10.1.2	band_lowlim	1	1000#m	1000#m	i.e. general tax exemption/allowance
10.1.3	band_rate	1	0.2	rate_pit	
10.1.4	output_var		tin_s	tin_s	
10.1.5	TAX_UNIT		tu_individual_sl	tu_individual_sl	

The next step is to add another new policy, before the output policy (output_std_sl), to perform additional tasks at the end of each iteration. Name this function **loopend_sl**.

In this policy, we should first use function *Totals* to calculate total disposable income (**agg=ils_dispy**), pre- and post-reform. This requires you to add two new *Totals* functions (Add function>Special Functions>Totals). By default these new functions only include the parameter *TAX_UNIT*. We need to add the following four additional parameters*:*

- *Run_Cond*: to define whether the function is carried out for the pre-reform scenario
  - **(Run_Cond=loopcount_revneutral=1)** for the pre-reform scenario
  - **(Run_Cond=loopcount_revneutral>1)** for the post-reform scenario
- *Agg*: to define the income list
- *Use_Weights*: to calculate totals using household weights
- *Varname_Sum*; storing results in **$pre_ils_dispy** and **$post_ils_dispy**

Figure 12.3: Calculating disposable income pre and post reform

12	loopend_sl		n/a	on	DEF: End the loop
12.1	Totals		n/a	on	Calculate pre-reform aggregate disposable income
12.1.1	Run_Cond		n/a	loopcount_revneutral=1	
12.1.2	Agg		n/a	ils_dispy	
12.1.3	Varname_Sum		n/a	pre	
12.1.4	Use_Weights		n/a	yes	
12.1.5	TAX_UNIT		n/a	tu_individual_sl	
12.2	Totals		n/a	on	Calculate post-reform aggregate disposable income
12.2.1	Run_Cond		n/a	loopcount_revneutral>1	
12.2.2	Agg		n/a	ils_dispy	
12.2.3	Varname_Sum		n/a	post	
12.2.4	Use_Weights		n/a	yes	
12.2.5	TAX_UNIT		n/a	tu_individual_sl	

Then add a function *Loop* (Add function>Special Functions>Loop) within the **loopend_sl** policy to specify which policies are included in the loop - *First_pol*, *Last_pol* - and what is the exit condition – *BreakCond* (Figure 12.4). However, when we add this function, we will only get *Loop_Id* as the default parameter and we will need to add the other parameters: *First_Pol*, *Last_Pol*, *BreakCond.*

The break condition needs to ensure that the loop does not stop on the first run when the $pre and $post disposable income variables are equal:

loopcount_revneutral>1

and that the calculation is carried out until the difference between the pre-reform and post-reform disposable income is less than 0.01%:

abs(((post_ils_dispy-pre_ils_dispy) / pre_ils_dispy) < 0.01%))

At the same time, we restrict the number of loops to 100 to avoid never ending loops:

loopcount_revneutral = 100

The full break condition is as follows:

(loopcount_revneutral=100) | ((loopcount_revneutral>1) & abs(((post_ils_dispy-pre_ils_dispy)/pre_ils_dispy)<0.01%))

Figure 12.4: Setting the loop parameters

12	**loopend_sl**		n/a	on	**DEF: End the loop**
12.1	Totals		n/a	on	Calculate pre-reform aggregate disposable income
12.2	Totals		n/a	on	Calculate post-reform aggregate disposable income
12.3	Loop		n/a	on	Define loop
12.3.1		Loop_Id	n/a	revneutral	ID
12.3.2		First_Pol	n/a	bch_sl	First policy
12.3.3		Last_Pol	n/a	loopend_sl	Last policy
12.3.4		BreakCond	n/a	(loopcount_revneutral=100) | ((loopcount_revneutral>1) & abs(((post_ils_dispy-pre_ils_dispy)/pre_ils_dispy)<0.01%))	Break condition

Our next step is to adjust model parameters for the child benefit amount and the income tax rate (Figure 12.5). To do that, we first add two *ArithOp* functions within the **loopend_sl** policy. In the first one we set the value of the child benefit (**$amt_bch**) to the desired amount. In the second function we adjust the income tax rate **$rate_pit** depending on how much the total disposable income differs from the baseline disposable income. For example according to this formula:

new rate = current rate * (current disposable income / baseline disposable income)

Figure 12.5: Adjusting the tax rate and child benefit amount

12	**loopend_sl**		n/a	on	**DEF: End the loop**
12.1	Totals		n/a	on	Calculate pre-reform aggregate disposable income
12.2	Totals		n/a	on	Calculate post-reform aggregate disposable income
12.3	Loop		n/a	on	Define loop
12.4	ArithOp		n/a	on	Adjust tax rate
12.4.1		Run_Cond	n/a	loopcount_revneutral>1	
12.4.2		Formula	n/a	rate_pit*(post_ils_dispy/pre_ils_dispy)	
12.4.3		Output_Var	n/a	rate_pit	
12.4.4		TAX_UNIT	n/a	tu_individual_sl	
12.5	ArithOp		n/a	on	Adjust child benefit amount
12.5.1		Run_Cond	n/a	loopcount_revneutral=1	
12.5.2		Formula	n/a	400#m	
12.5.3		Output_Var	n/a	amt_bch	
12.5.4		TAX_UNIT	n/a	tu_individual_sl	

Finally, we should include the variables in the **output_std_sl** policy to show the income tax rate after the convergence and the number of iterations performed (Figure 12.6).

Figure 12.6: Adding the new variables to the output

13	output_std_sl		on	on
13.1	DefOutput		on	on
13.1.1	file		sl_demo_std	SL_demo_E12_std
13.1.2	Var		n/a	loopcount_revneutral
13.1.3	Var		n/a	amt_bch
13.1.4	Var		n/a	rate_pit
13.1.5	Var		n/a	pre_ils_dispy
13.1.6	Var		n/a	post_ils_dispy

Task 3: Producing the micro-output

Run EUROMOD in order to produce micro-output for the initial and new system - SL_1996 and SL_1996_E12 - using dataset '**SL_training_data**'. Also see what the new income tax rate is and how many iterations were needed.

Task 4: Analyse the results

Use the *Statistics Presenter tool* to analyse differences in benefits and disposable income due to the changes.

In the micro data output file (Figure 12.7) we can see that the new tax rate is 25% and 41 iterations were needed to get to the result. As requested, the difference between pre-reform and post-reform disposable income is less than 0.01%. In Figure 12.8 we can see the summary statistics showing the revenue-neutrality of the reform and in Figure 12.9 how poverty and inequality has been affected.

Figure 12.7: Output file

loopcount	$amt_bch	$rate_pit	pre_ils_dispy	post_ils_dispy	idhh	idperson	idmother	idfather	idpartner	dag
51	400	0.27	797151444.4	797228241.7	1	101	0	0	0	
51	400	0.27	797151444.4	797228241.7	2	201	0	0	0	
51	400	0.27	797151444.4	797228241.7	3	301	0	0	0	

Figure 12.8: Fiscal impact

Market Incomes and Government Revenue & Expenditure

Yearly, mill., currency as defined in EM output

	SL_1996 (base) (annual)	SL_1996_E12 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**8,132.03**	**8,132.03**	**0.00**	**0.00 %**
... income from (self) employment	8,132.03	8,132.03	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**1,961.74**	**2,339.50**	**377.76**	**19.26 %**
... direct taxes	755.08	1,132.84	377.76	50.03 %
... employee social insurance contributions	378.14	378.14	0.00	0.00 %
... self-employed social insurance contributions	72.23	72.23	0.00	0.00 %
... other social insurance contributions	0.00	0.00	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	756.28	756.28	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**0.00**	**0.00**	**0.00**	**0.00 %**
Government expenditure on social transfers	**2,639.25**	**3,017.93**	**378.68**	**14.35 %**

Figure 12.9: Poverty and inequality impact

3.1 Basic Inequality Indices

	Gini SL_1996	S80/S20 SL_1996	Gini SL_1996_E12	Diff. Gini SL_1996_E12/SL_1996	S80/S20 SL_1996_E12	Diff. S80/S20 SL_1996_E12/SL_1996
Original Income	0.4886	6.1584	0.4886	0.0000	5.8178	-0.3405
Original Income after Taxes/SIC	0.4786	5.6932	0.4802	0.0016	5.3532	-0.3399
Original Income incl. Public Pensions after Taxes/SIC	0.3221	7.2719	0.3282	0.0061	7.0199	-0.2520
Disposable Income	0.2616	3.9080	0.2505	-0.0111	3.7216	-0.1864

3.2.a Basic Poverty Indices

	Poverty Risk for SL_1996 (base)	Poverty Risk for SL_1996_E12	Difference to base
At-risk-of-poverty rate	22.07 %	19.99 %	-2.08pp
Median at-risk-of-poverty gap	19.88 %	18.99 %	-0.89pp
Fixed Poverty line (monthly)	832.91		

Exercise 13: Implementing an EU child benefit using an AddOn

The aim of this exercise is to learn how to build, export and modify an AddOn which will implement a common child benefit scheme across EU countries.

Steps:

1. Create a new system in Simpleland based on SL_1996 system and name it SL_1996_E13;
2. Implement the new child benefit. Call the new policy bcheu and the new benefit bchot_s. Place the new policy before the output policies;
3. Technical step: create a new output policy and switch off the existing output policy using a change parameter function. Make sure that the new output file terminates with _std;
4. Analyse the distributional impact of the benefit in Simpleland;
5. Export the reform system as an AddOn;
6. Analyse the AddOn;
7. Modify the AddOn such that it can be used with the chosen countries;
8. Analyse the results

Description:

Italy and Netherlands have been chosen as pilot countries for the introduction of an EU child benefit aimed at alleviating child poverty in low-income households. The scheme will have common rules across countries and, if successful, will then be extended to the remaining EU countries. The benefit rules specify that beneficiaries of the EU benefit will be children aged 14 or younger living in households in which the sum of market incomes (ils_origy) and pensions (ils_pen) amounts to less than 24,000 Euros per year. Each child in eligible households is entitled to a benefit of 3,000 Euros per year. Analyse the impact of the benefit on child poverty.

Although it would be possible to implement the reform independently in each country, the reform is suited to be implemented through an AddOn given the common features of the scheme across member states. AddOns cannot be built from scratch and users have two possibilities. The first consists in modifying an existing add-on. The second is to implement the changes in a country of choice, and then to export these changes as an add-on. As a further step, the newly created AddOn will have to be modified to be able to run it on other countries. In this exercise we follow the second possibility and create an AddOn starting from Simpleland.

Solution and further information

Task 1: Creating a new system

Our first task is to open Simpleland and create a new system using the SL_1996 system as a base. Give the name SL_1996_E13 to the new system.

Task 2: Implementing a new policy

Our next step is to add a new policy containing the implementation of the EU child benefit. We will place the new policy before **Output_std_sl**. We'll be asked to change the name of the policy bcheu in bcheu_sl, because we are working on Simpleland. In this case we should press "No" because our aim is to create an AddOn to be used in other countries as well.

The policy can be implemented by defining an income list which contains the relevant incomes for the means-tested benefit and calculating benefit entitlement through a function *BenCalc*. Assign the name *bchot_s* to the EU child benefit. Remember to calculate the benefit for each eligible child and not for the Tax Unit in total (Comp_perElig).

Figure 13.1: New EU child benefit

bcheu		n/a	on	**BEN: EU child benefit**
fx DefIl		n/a	on	Income list for means test
Name		n/a	il_bcheu	
ils_origy		n/a	+	
ils_pen		n/a	+	
fx BenCalc		n/a	on	Benefit amount
Comp_Cond	1	n/a	dag<=14 & il_bcheu<24000#y	Children aged 14 or younger and household income below threshold
Comp_perElig	1	n/a	3000#y	... per eligible child in the tax unit
Output_Var		n/a	bchot_s	
TAX_UNIT		n/a	tu_household_sl	

We should also remember to include the newly created benefit among the components of the standard income lists *ils_benmt*. This step allows EUROMOD to take into account the EU child benefit when computing disposable income.

Task 3: Technical step – create a new output policy

Copy and paste the existing output policy **output_std_sl.** Paste it before **Output_std_sl**. Again if you are asked, select 'No' as we would like this policy to be applicable across different countries.

- Call the new policy output_bcheu, access it and modify the function Def_Output by changing the value of the parameter file in sl_bcheu_std.
- As a second step, introduce a ChangeParam function to switch the original output policy off. The reason behind the technical step will be explained in the remainder of the solutions.
- Finally, make sure to set all functions to n/a in the original 'SL_1996' system.

Figure 13.2: New output policy

bcheu		n/a	on	BEN: EU child benefit
output_bcheu		n/a	on	DEF: STANDARD OUTPUT INDIVIDUAL LEVEL
fx ChangeParam		n/a	on	switch standard output off
Param_Id	1	n/a	output_std_sl	
Param_New...	1	n/a	off	
fx DefOutput		n/a	on	new output file
file		n/a	SL_bcheu_std	
Var		n/a	n/a	
Var		n/a	n/a	
Var		n/a	n/a	
Var		n/a	n/a	
Var		n/a	n/a	
VarGroup		n/a	id*	Identifiers
VarGroup		n/a	d*	Demographic

Task 4: Run the model and analyse the effect of the reform on Simpleland

Using the Statistics Presenter, create statistics using the base/reform option to analyse the distributional effect of the reform using 'SL_training_data' as the dataset. Your outputs should look like below.

Figure 13.3: Fiscal table

	SL_1996 (base) (annual)	SL_1996_E13 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**8,132.03**	**8,132.03**	**0.00**	**0.00 %**
... income from (self) employment	8,132.03	8,132.03	0.00	0.00 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**1,961.74**	**1,961.74**	**0.00**	**0.00 %**
... direct taxes	755.08	755.08	0.00	0.00 %
... employee social insurance contributions	378.14	378.14	0.00	0.00 %
... self-employed social insurance contributions	72.23	72.23	0.00	0.00 %
... other social insurance contributions	0.00	0.00	0.00	0.00 %
... employer social insurance contributions (not part of disposable income)	756.28	756.28	0.00	0.00 %
Credited social insurance contributions (not part of disposable income)	**0.00**	**0.00**	**0.00**	**0.00 %**
Government expenditure on social transfers	**2,639.25**	**2,763.12**	**123.87**	**4.69 %**
by target group				
... unemployment benefits	0.00	0.00	0.00	0.00 %
... family and education benefits	0.00	0.00	0.00	0.00 %
... social assistance and housing benefits	0.00	0.00	0.00	0.00 %
... pensions, health and disability benefits	0.00	0.00	0.00	0.00 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	332.59	456.46	123.87	37.24 %
... non-means-tested non-pension benefits	419.39	419.39	0.00	0.00 %
... pensions	1,887.26	1,887.26	0.00	0.00 %
... firms subsidies	0.00	0.00	0.00	0.00 %

Figure 13.4: Poverty and inequality table

3.1 Basic Inequality Indices

	Gini SL_1996	S80/S20 SL_1996	Gini SL_1996_E13	Diff. Gini SL_1996_E13/SL_1996	S80/S20 SL_1996_E13	Diff. S80/S20 SL_1996_E13/SL_1996
Original Income	0.4886	6.1584	0.4886	0.0000	5.4912	-0.6671
Original Income after Taxes/SIC	0.4786	5.6932	0.4786	0.0000	5.1161	-0.5770
Original Income incl. Public Pensions after Taxes/SIC	0.3221	7.2719	0.3221	0.0000	6.1903	-1.0816
Disposable Income	0.2616	3.9080	0.2518	-0.0098	3.6399	-0.2680

3.2.a Basic Poverty Indices

	Poverty Risk for SL_1996 (base)	Poverty Risk for SL_1996_E13	Difference to base
At-risk-of-poverty rate	22.07 %	16.68 %	-5.39pp
Median at-risk-of-poverty gap	19.88 %	20.11 %	0.24pp
Fixed Poverty line (monthly)	832.91		

Task 5: Export the policy reform as an add-on

We can do that using the Export Add-On button which is placed in the tab Country tools tab under System Operations. From here you need to:

- fill a long and a short name for our Add-On
- select the Add-On and base systems from the list.
- By ticking 'Use symbolic identifiers', references to changes of specific functions and parameters will be done using the symbolic identifier rather than the cell identifier.
- By ticking 'Use country placeholder', references to Simpleland will be replaced by the placeholder =cc=, making the process of adaption of the AddOn to other countries easier.

Figure 13.5: Exporting Add-On

Except to delete the system SL_1996_E13 and to move it to the Add-on instead. The original system is stored in the backup folder and can be restored under Country Tools → Restore.

Click on the Ribbon menu item 'Add-Ons', the bcheu icon should be there (Figure 13.6). Click on it to open it.

Figure 13.6: BCHEU Add-On

Task 6: Analyse the AddOn

The exported AddOn is composed of four policies, namely ao_control_bcheu, Implementation_bcheu; bcheu and Output_bcheu.

The policy ao_control_bcheu contains the following functions:

- AddOn_Applic: specifies that the AddOn runs for the system SL_1996
- AddOn_Func: contains a unique identifier referring to a function that is created by the AddOn and which will be added before the first function (#1) of the policy Uprate. This function is actually not needed and can be deleted.
- The function AddOn_Par modifies the income list called ils_benmt by including the newly simulated benefit. This function needs to be rewritten in more general terms to make sure that the Add-On can also be used for other countries. Thus, instead of using a unique identifier, change the functions to 'IlsDef_=cc=_#ils_benmt. =cc= refers to a placeholder that will be replaced by the country acronym.
- The two functions AddOn_Pol specify the names of the policies that will be added and their position in the spine. While the second function already works for all countries, the first needs to be revised to make sure that the new policy 'bcheu' is included before the output policy instead of after the bsa_sl policy given that this policy only exists in Simpleland.
    - Thus, click on 'Insert_After_Pol' and change it to 'Insert_Before_Pol'
    - Change the policy name to 'output_std_=cc=

Figure 13.7: Control policy of Add-On

Policy	Grp/No	bcheu	Comment
ao_control_bcheu		**on**	
fx AddOn_Applic		**on**	Systems the Add-on can be run with
Sys		SL_1996	
fx AddOn_Par		**on**	Add new benefit to income list
Insert_Func		IlsDef_=cc=_#ils_benmt	
bchot_s		+	
fx AddOn_Pol		**on**	Add bcheu policy before output policy
Pol_Name		bcheu	
Insert_Before_Pol		output_std_=cc=	
fx AddOn_Pol		**on**	Add new output policy after new benefit policy
Pol_Name		output_bcheu	
Insert_After_Pol		bcheu	
Implementation_bcheu		**on**	
bcheu		**on**	**BEN: EU child benefit**
output_bcheu		**on**	**DEF: STANDARD OUTPUT INDIVIDUAL LEVEL**

The policy called Implementation_bcheu contains a function change parameter. This parameter changes the name of the output file. However, we have already specified a new output file name in the output_bcheu policy as well as a ChangeParam function to switch off the standard output. Thus, this policy can be dropped from the AddOn. As mentioned above the related function AddOn_Func of ao_control_bcheu in 1.2 can be dropped from the add-on.

The policies bcheu and output_bcheu contain the implementation of the benefit and of the output policy.

Task 7: Modify the AddOn such that it can be used with other countries

The AddOn now needs to be modified such that it will be possible to run it on the 2020 systems of the Netherlands and Italy.

- The first step is to add additional systems to the AddOn_Applic function within the ao_control_bcheu policy
- Change the tax unit to the generic name 'tu_household_=cc='
- Change the Param_Id to 'output_std_=cc='
- Change the output file name to '=sys=_bcheu_std'. 'Sys' stands for the system that the Add-On is run with. And the tax unit of the output file to 'tu_individual_=cc=

Figure 13.8: Adjusting the Add-On to be run with other countries

Policy	Grp/No	bcheu	Comment
ao_control_bcheu		**on**	
AddOn_Applic		**on**	Systems the Add-on can be run with
Sys		SL_1996	
Sys		IT_2020	
Sys		NL_2020	
AddOn_Par		**on**	Add new benefit to income list
AddOn_Pol		**on**	Add bcheu policy before output policy
AddOn_Pol		**on**	Add new output policy after new benefit policy
bcheu		**on**	**BEN: EU child benefit**
DefIl		**on**	Income list for means test
BenCalc		**on**	Benefit amount
Comp_Cond	1	dag<=14 & il_bcheu<24000#y	Children aged 14 or younger and household income below threshold
Comp_perElig	1	3000#y	… per eligible child in the tax unit
Output_Var		bchot_s	
TAX_UNIT		tu_household_=cc=	
output_bcheu		**on**	**DEF: STANDARD OUTPUT INDIVIDUAL LEVEL**
ChangeParam		**on**	switch standard output off
Param_Id	1	output_std_=cc=	
Param_NewVal	1	off	
DefOutput		**on**	new output file
file		=sys=_bcheu_std	
Var		n/a	
Var		n/a	
Var		n/a	
Var		n/a	
Var		n/a	
VarGroup		id*	Identifiers
VarGroup		d*	Demographic
VarGroup		l*	Labour
VarGroup		a*	Assets
VarGroup		k*	In-kind
VarGroup		p*	Pensions
VarGroup		b*	Benefits
VarGroup		y*	Income
VarGroup		t*	Taxes
VarGroup		x*	Expenses
VarGroup		i*	Intermediate variables
VarGroup		s*	Temporary variables
ILGroup		ils_*	Standard income lists
ILGroup		il_*	Non-standard income lists
nDecimals		2	
TAX_UNIT		tu_individual_=cc=	

Task 8: Analyse the results

We can now run the Italian and Dutch model with the Add-On. In the Run menu, click on View/Filter/Add-Ons > Add-Ons and select the new Add-On 'bcheu'. Then select the systems for Italy (IT) and Netherlands (NL), filter them for year 2020 and run with their country specific training dataset. Make sure you have ticked the box for the bcheu add on.

Figure 13.9: Run the model with the Add-On

Figure 13.10: Select the Add-on for the two systems

To analyse the results, we can again use the statistics presenter. In this case we would select multisystem option and then select the four output of the baselines and the reform scenarios. Please note that the name of the reform scenarios appear the same as the baselines, so is it important to remember the order with which you select the files (baseline first, with Add-on second). Another option is to export results to excel and to change the name of the reform scenarios.

Figure 13.11: Fiscal table

	NL_2020	NL_2020	IT_2020	IT_2020
Total market incomes	**265,907.79**	**265,907.79**	**579,006.03**	**579,006.03**
... income from (self) employment	265,907.79	265,907.79	579,006.03	579,006.03
... other sources	0.00	0.00	0.00	0.00
Government revenue through taxes and social insurance contributions	**130,315.18**	**130,315.18**	**400,594.31**	**400,594.31**
... direct taxes	39,590.68	39,590.68	176,358.17	176,358.17
... employee social insurance contributions	30,701.26	30,701.26	46,451.42	46,451.42
... self-employed social insurance contributions	0.00	0.00	20,771.35	20,771.35
... other social insurance contributions	23,365.21	23,365.21	0.00	0.00
... employer social insurance contributions (not part of disposable income)	36,658.35	36,658.35	157,013.38	157,013.38
Credited social insurance contributions (not part of disposable income)	**1,136.26**	**1,136.26**	**0.00**	**0.00**
Government expenditure on social transfers	**72,186.90**	**73,057.31**	**326,023.47**	**329,780.06**
by target group				
... unemployment benefits	6,694.74	6,694.74	50.77	50.77
... family and education benefits	4,592.52	4,592.52	6,844.03	6,844.03
... social assistance and housing benefits	10,264.28	10,264.28	29,824.62	29,824.62
... pensions, health and disability benefits	50,635.37	50,635.37	289,304.05	289,304.05
... firms	0.00	0.00	0.00	0.00
by benefit design				
... means-tested non-pension benefits	20,064.79	20,935.20	39,015.15	42,771.74
... non-means-tested non-pension benefits	9,372.82	9,372.82	334.30	334.30
... pensions	42,749.30	42,749.30	286,674.02	286,674.02
... firms subsidies	0.00	0.00	0.00	0.00

Figure 13.12: Poverty and inequality table

3.1 Basic Inequality Indices

	Gini NL_2020	Gini NL_2020	Gini IT_2020	Gini IT_2020	S80/S20 NL_2020	S80/S20 NL_2020	S80/S20 IT_2020	S80/S20 IT_2020
Original Income	0.5229	0.5229	0.5253	0.5253	19.7988	21.6820	4.3208	4.3283
Original Income after Taxes/SIC	0.5202	0.5202	0.5434	0.5434	20.9618	23.3902	2.8753	2.9244
Original Income incl. Public Pensions after Taxes/SIC	0.3561	0.3561	0.2925	0.2925	3.8831	4.0025	6.4931	6.1147
Disposable Income	0.2818	0.2778	0.2381	0.2315	2.6645	2.7438	3.5313	3.3297

3.2.a Basic Poverty Indices

	Value for NL_2020	Value for NL_2020	Value for IT_2020	Value for IT_2020
At-risk-of-poverty rate	8.07 %	6.40 %	19.42 %	18.03 %
Median at-risk-of-poverty gap	18.83 %	11.45 %	16.13 %	12.75 %
Fixed Poverty line (monthly)	1,019.40	1,019.40	893.94	893.94

Exercise 14: Adding a new dataset for Austria, specifying uprating factors and setting switches

The aim of this exercises is to learn how to add a new dataset to a country and how to specify uprating factors and switches.

Steps

1. Adding a new dataset and specifying its characteristics
2. Setting switches for the new dataset
3. Specifying uprating factors for the new dataset
4. Check whether there are dataset specific settings in policies
5. Run the 2023 system with the non-uprated and the uprated dataset and compare results

Description

The training datasets are not uprated in the country models. However, some functions and tools of EUROMOD make use of uprating factors. Thus, we will add an additional training dataset for Austria where we specify uprating factors. The EUROMOD input folder includes an additional training dataset for Austria (AT_training_data_2) which is a copy of AT_training_data.

When adding a new dataset, the software asks whether we want to specify switches for the new dataset which we will do. Switches – often also referred to as 'Extensions' – allow users to change specified default settings when running the model. For example: users can decide whether to run the model with benefit take-up adjustments or without.

Third, we need to make sure that the uprating functions included for the micro datasets is applied to the additional training dataset.

Finally, we need to check whether there are dataset specific settings in policies and adjust those for our new dataset. Tip: search for the parameter 'Run_Cond'.

Now the system can be run with the non-uprated and the uprated datasets.

Solutions and further information

Task 1: Adding a new dataset

Datasets are specified under ‘Country Tools’ and ‘Databases’. The menu includes two tabs, ‘General’ includes the microdata-based datasets as well as the training data, ‘HHOT’ includes specifications for datasets produced using the ‘Hypothetical Household Tool’ (HHoT).

Figure 14.1: Opening database specifications

We will add the new dataset under the ‘General’ Tab by clicking on ‘Add Dataset’. This opens the input folder of our EUROMOD project and shows all available datasets. Select the dataset called ‘AT_training_data_2’ and click ‘Open’.

Figure 14.2: Adding a new dataset

Figure 14.3: Selecting the new dataset

This adds the additional dataset to the databases available for Austria, for which we still need to configure all the settings:

- Specify for which policy years the dataset can be run. The training dataset should be available for all years. Thus, right-click on the name of the dataset and select 'set x for a all systems'.
- Collection and income year: the training datasets are generated in HHoT based on 2019 income levels and the distribution of characteristics in EU-SILC 2020. We have opted for not setting uprating factors to training datasets and thus, these fields are set to '-' for 'AT_training_data'. For this exercise we need to set the fields to 2019.
- Currency: no changes required
- Select 'Use Common Default': Checking this option means that any variable not existent in data, but used by the system, is set to zero, i.e. no error message is issued.

Figure 14.4: Specifications for new dataset

Task 2: Setting switches

A window pops up automatically asking whether we would like to define Policy Switches. Select 'Yes'. Switches – often also referred to as 'Extensions' – allow users to change specified default settings when

running the model. The default settings are specified in the 'Set Policy Switches' menu. These defaults are taken into account when running models without changing switches.

Different switches exist in different countries. Switches that are set to 'n/a' are not available in the country. This is the case for most switches in Austria. Switches that require action are:

- Parental leave benefit which should be switched off
- Minimum Wage Adjustments which should be switched off
- HHoT unemployment extension which should be switched off
- Labour market adjustments which should be switched off

To switch off the switch for a specific dataset for all policy years, right-click on the dataset name and select 'Set OFF for all systems'.

Figure 14.5: Setting switches

Task 3: Specifying uprating factors

Each country model has an uprating policy at the top of the policy spine. Typically, there is one function for SILC data and another one for training datasets and hypothetical data. The current uprating function for 'AT_training_data' uprates all monetary variable by 1. This is not what we want for our new dataset.

Figure 14.6: Current uprating factors for training data

uprate_at	on	on	on	on	DEF: UPRATING FACTORS
fx Uprate	on	on	on	on	Update SILC data
fx Uprate	on	on	on	on	Update training and hypo data
dataset	*training_data	*training_data	*training_data	*training_data	
Dataset	*_hhot	*_hhot	*_hhot	*_hhot	
def_factor	1	1	1	1	
WarnIfNoFa...	no	no	no	no	

Instead, we would like to uprate 'AT_training_data_2' using the uprating factors specified for SILC datasets. Thus, we need to make sure that the dataset is taken into account in the uprating function for SILC data. This can be done by adding a new dataset parameter to the uprating function.

Finally, add the dataset name to the new dataset parameter in all policy systems. You can spread the name from one system to all systems using the keys Alt + S.

Figure 14.7: New specification for uprating factors

uprate_at	on	on	on	on	DEF: UPRATING FACTORS
fx Uprate	on	on	on	on	Update SILC data
dataset	AT_20??_??	AT_20??_??	AT_20??_??	AT_20??_??	
Dataset	AT_training_data_2	AT_training_data_2	AT_training_data_2	AT_training_data_2	

Task 4: Check whether there are dataset specific settings in policies

The best way to do this is to search for the parameter 'Run_Cond' in the model. This parameter specifies whether a function should only be run for a specific dataset. Several such conditions are included in the tscee_at policy. They specify the current year that should be used to calculate the birth year of an

individual. One function specifies the year for training datasets and this is where the new training dataset needs to be added. This can be done by adding a new parameter to specify the dataset and to add this new parameter to the run condition.

Figure 14.8: Adding a new dataset new to the run condition

DefConst		on	on	on	on	training data
$PolicyYear		2007	2008	2009	2010	
Run_Cond		IsUsedDatabase#1 | IsUsedDatabase#2	IsUsedDatabase#1 | IsUsedDatabase#2	IsUsedDatabase#1 | IsUsedDatabase#2	IsUsedDatabase# IsUsedDatabase#:	
#_DataBase...	1	*training_data	*training_data	*training_data	*training_data	
#_DataBase...	2	AT_training_data_2	AT_training_data_2	AT_training_data_2	AT_training_data_	

Task 5: Run the 2023 system with the non-uprated and the uprated dataset and compare results

We want to run the same system with both datasets. However, this means that we need to make sure that the generated files have different names. This can be achieved by selecting 'do not pool system's datasets' in the 'Advanced Settings'. This saves the output files in two different folders within the Output folder. This is important to know when comparing the results in the Statistics Presenter Tool.

Figure 14.9: Running the same system with two datasets at the same time

Note: the model produces an error message warning users that four variables are uprated with default factor: 'WARNING: 2.1 uprate_at/Uprate (04d59373-087c-4958-862f-5e7a4a288583): variable(s) pyr, bac, bcc, bch is/are uprated with default factor (1)'. These are variables that are not used in the model and for which no uprating factor is defined. Thus, we can just ignore the message.

Figure 14.10: Fiscal results

	at_2023 (base) (annual)	at_2023 (annual)	Difference to base (annual)	% Difference to base
Total market incomes	**133,092.80**	**155,892.44**	**22,799.65**	**17.13 %**
... income from (self) employment	133,092.80	155,892.44	22,799.65	17.13 %
... other sources	0.00	0.00	0.00	0.00 %
Government revenue through taxes and social insurance contributions	**78,792.01**	**96,129.87**	**17,337.86**	**22.00 %**
... direct taxes	18,336.40	26,574.58	8,238.18	44.93 %
... employee social insurance contributions	21,214.33	24,609.57	3,395.24	16.00 %
... self-employed social insurance contributions	2,860.06	2,860.06	0.00	0.00 %
... other social insurance contributions	2,577.78	2,906.51	328.73	12.75 %
... employer social insurance contributions (not part of disposable income)	33,803.44	39,179.16	5,375.72	15.90 %
Credited social insurance contributions (not part of disposable income)	**2,011.70**	**2,268.19**	**256.49**	**12.75 %**
Government expenditure on social transfers	**75,206.71**	**81,686.15**	**6,479.44**	**8.62 %**
by target group				
... unemployment benefits	8,678.82	9,706.66	1,027.83	11.84 %
... family and education benefits	4,780.13	4,828.67	48.53	1.02 %
... social assistance and housing benefits	10,742.45	9,763.33	-979.12	-9.11 %
... pensions, health and disability benefits	51,005.25	57,387.46	6,382.22	12.51 %
... firms	0.00	0.00	0.00	0.00 %
by benefit design				
... means-tested non-pension benefits	10,552.93	9,510.67	-1,042.26	-9.88 %
... non-means-tested non-pension benefits	14,107.56	15,185.38	1,077.82	7.64 %
... pensions	50,546.29	56,990.06	6,443.77	12.75 %
... firms subsidies	0.00	0.00	0.00	0.00 %

Figure 14.11: Inequality and poverty results

3.1 Basic Inequality Indices

	Gini at_2023	S80/S20 at_2023	Gini at_2023	Diff. Gini at_2023/at_2023	S80/S20 at_2023	Diff. S80/S20 at_2023/at_2023
Original Income	0.4996	6.4444	0.4998	0.0002	7.2533	0.8090
Original Income after Taxes/SIC	0.5006	4.6131	0.5051	0.0044	5.2149	0.6018
Original Income incl. Public Pensions after Taxes/SIC	0.2992	5.1578	0.3027	0.0036	5.2300	0.0722
Disposable Income	0.2056	2.7553	0.2190	0.0135	3.0027	0.2474

3.2.a Basic Poverty Indices

	Poverty Risk for at_2023 (base)	Poverty Risk for at_2023	Difference to base
At-risk-of-poverty rate	9.76 %	8.10 %	-1.66pp
Median at-risk-of-poverty gap	8.67 %	8.67 %	0.00pp
Fixed Poverty line (monthly)	1,346.51		

Exercise 15: Modifying uprating factors in Austria to account for differential wage evolution

The aim of this exercise is to understand the concept of uprating factors and how to feed them with external information on the evolution of a variable.

Steps:

1. Create a copy of the AT 2021 system (e.g. AT_2021_E15).
2. Create two new time-series in the uprating indices table (tab Raw indices) to account for the differential evolution of "low" and "high" salaries between 2018 and 2021.
3. Check the differential evolution of the two time series (table Factors per data and system).
4. Modify the uprate_at policy to account for the new information.
5. Run EUROMOD to produce micro-outputs for these two systems (2021 and 2021_E15);
6. Analyse the results using the In-depth analysis tool

Description:

The average monthly salary in Austria has increased by from 2018 to 2021 by 9%. Now consider the hypothetical case that lower salaries have increased less than higher salaries. The table shows these hypothetical values:

	Average salary				
Monthly salary in 2018	2018	2019	2020	2021	Change in %
Average annual wage	33,221	34,167	35,072	36,283	9.2
"Low" salaries (up to average)	25,500	26,226	26,921	27,387	7.4
"High" salaries (above average)	51,000	52,452	53,842	56,763	11.3

We would like to take this into account in the uprating of earnings. This requires changes to the uprating factor table as well as the uprating factor policy:

- In "Uprating indices → Raw indices" add two rows for two new indices, $f_yemlow and $f_yemhigh, and paste the two time series. Bear in mind that for this exercise we don't care about what happened before 2018 and in-between 2018 and 2021, so you can leave those values blank (EUROMOD will fill them with missing values).
- In the new system AT_2021_E15 go to the Uprate function in the uprate_at policy and make sure that the new uprating factors are used.

Use the In-depth Analysis plugin to compare the new system with the original 2021 system.

- **Note**: although in most cases the solution proposed for this exercise works correctly, double uprating might occur for some observations whose income is very close to the threshold, due to the sequential application of the two factors. There are ways to circumvent this issue, but they lie beyond the scope of this training. In any case, the exercise fits the double purpose of (1) showing how to use factor conditions and (2) understand the distributional impact of using differential factors by income levels.

Solution and further information

Task 1: Creating a new system

We right click on AT_2021, select Copy/paste system, and name the new as AT_2021_E15.

Task 2: Create two new time-series in the uprating indices table

Our first task is to understand how the uprating indices table is constructed in EUROMOD. We open Austria, we go to Country tools - Uprating indices.

Figure 15.1: Uprating indices - Raw indices

	Index	Reference	2018	2019	2020	2021	2022
1	Harmonised Index of Consumer Prices	$HICP	105.41	106.98	108.47	111.46	121.07
2	CPI 2005 = 100 (as of June)	$f_cpi	127.4	129.4	130.9	134.6	146.3
3	Average annual wage	$f_yem	33221	34167	35072	36283	39437
4	Average annual wage, previous year	$f_yempv	32267	33221	34167	35072	36283
5	Average annual self employment income (break in series 2009-2010)	$f_yse	29708	30209	30209	30209	30209
6	Equ. disposable income (original silc), not uprated	$f_ydses	1	1	1	1	1
7	Annual average issue yields 2006 = 100	$f_yiy	1.277164374	1.285082793	1.29125119	1.29628707	1.301212961
8	Family benefits that are not uprated	$f_bfa	1	1	1	1	1
9	Family benefits (based on semi-orphan base rate at age 24)	$f_bfaam	594.4	609.85	631.8	653.91	673.53
10	Average daily rate of unemployment benefit	$f_bunct	32.13	32.81	33.12	35.2	35.89

In row 3 we see the index for salaries ($f_yem), expressed in annual EUR. This means that, if $f_yem is used in the model to uprate yem, earnings are uprated based on changes in average annual wages in Austria. From 2018 to 2021, the uprating factor changed from 33,221 to 36,283 which amounts to about 9.2%. This can be seen in the second tab called 'Factors per data and system', when selecting the EU-SILC-based 2019 input dataset in the dropdown list.

Figure 15.2: Uprating indices - Factors per data and system

Index	AT_2021	AT_2021_E15
$HICP	1.0574	1.0574
$f_cpi	1.0565	1.0565
$f_yem	1.0922	1.0922
$f_yempv	1.0869	1.0869
$f_yse	1.0169	1.0169
$f_ydses	1	1
$f_yiy	1.015	1.015
$f_bfa	1	1
$f_bfaam	1.1001	1.1001
$f_bunct	1.0955	1.0955

In this exercise, we want to apply different uprating factors to different earnings levels. Thus, we need to go back to the Raw indices table and add two new rows at the end, that we name $f_yemlow and $f_yemhigh. And copy the above table with the salaries over time into the respective columns. Then we save and close. No need to fill in values for the other years and just except the warning that EUROMOD produces.

Figure 15.3: Adding new uprating factors

	Index	Reference	2018	2019	2020	2021	2022
36	Average hourly wage of previous year, All activity sectors, units of national currency	$f_hourly_wage_pv	24.7237	25.4149	26.007	26.4784	27.207
37	Average hourly wage of following year, All activity sectors, units of national currency	$f_hourly_wage_dt	26.007	28.6431	28.4466	29.6295	30.8615
38	One	$f_one	1	1	1	1	1
39	Average of monthly remuneration for military service (Präsenzdienst) and alternative service (Zivildienst)	$f_yot01	362.61	362.61	362.61	362.61	362.61
40	Lower salaries	$f_yemlow	25500	26226	26921	27387	
41	Higher salaries	$f_yemhigh	51000	52452	53842	56763	
42							

Task 3: Check the differential evolution of the two time series

Now we go again to the second tab, select the 2019 dataset and see the impact that the use of the new time series would have. We can compare them with $f_yem, to see the differences between using this index or the two that we have just created: 1.0922 vs. 1.074 and 1.113. Thus, higher salaries are increasing above average while lower salaries are increasing below average.

Figure 15.4: Impact of the new indices

Index	AT_2021
$f_yem	1.0922
$f_yempv	1.0869
$f_yse	1.0169
$f_ydses	1
$f_yiy	1.015
$f_bfa	1
$f_bfaam	1.1001
$f_bunct	1.0955
$f_bunnc	1.1508
$f_xhcrt	1.0561
$f_bdi1	1.0331
$f_bdi2	1.0331
$f_bdi3	1.0332
$f_bdi4	1.0332
$f_bdi5	1.0334
$f_bdi6	1.0332
$f_bdi7	1.0333
$f_poa	1.0667
$f_bac	1.1001
$f_yemdt	1.1542
$f_hourly_wage_lindi_1	1.0299
$f_hourly_wage_lindi_2	1.0797
$f_hourly_wage_lindi_3	1.0702
$f_hourly_wage_lindi_4	1.0692
$f_hourly_wage_lindi_5	1.0777
$f_hourly_wage_lindi_6	1.06
$f_hourly_wage_lindi_7	1.0599
$f_hourly_wage_lindi_8	1.0775
$f_hourly_wage_lindi_9	1.073
$f_hourly_wage_lindi_10	1.0603
$f_hourly_wage_lindi_11	1.0678
$f_hourly_wage_lindi_12	1.0673
$f_hourly_wage	1.0705
$f_hourly_wage_pv	1.071
$f_hourly_wage_dt	1.1393
$f_one	1
$f_yot01	1
$f_yemlow	1.074
$f_yemhigh	1.113

Task 4. Modify the uprate_at policy

Until now we have just created two new series of uprating indices, but this won't have any impact on the results of the model unless we use them in the EUROMOD spine (as $f_yemlow and $f_yemhigh). We can do this in the policy uprate_at. We see that currently the main variable for salaries (yem) is uprated using different factors for different industries. The factor conditions allow us to differentiate the treatment of salaries below and above the average wage.

Instead, we would like to use different uprating factors for lower and higher salaries based on changes in annual earnings, i.e. $f_yemlow and $f_yemhigh.

Figure 15.5: Applying new uprating indices

Factor_Con...	19	lindi = 1	yem*12<33221	
yem	19	$f_hourly_wage_lindi_1	$f_yemlow	gross employee cash or near cash income - separately for Industry (NACE)
Factor_Con...	20	lindi = 2	yem*12>=33221	
yem	20	$f_hourly_wage_lindi_2	$f_yemhigh	
Factor_Con...	21	lindi = 3	n/a	
yem	21	$f_hourly_wage_lindi_3	n/a	
Factor_Con...	22	lindi = 4	n/a	
yem	22	$f_hourly_wage_lindi_4	n/a	
Factor_Con...	23	lindi = 5	n/a	
yem	23	$f_hourly_wage_lindi_5	n/a	
Factor_Con...	24	lindi = 6	n/a	
yem	24	$f_hourly_wage_lindi_6	n/a	
Factor_Con...	25	lindi = 7	n/a	
yem	25	$f_hourly_wage_lindi_7	n/a	
Factor_Con...	26	lindi = 8	n/a	
yem	26	$f_hourly_wage_lindi_8	n/a	
Factor_Con...	27	lindi = 9	n/a	
yem	27	$f_hourly_wage_lindi_9	n/a	
Factor_Con...	28	lindi = 10	n/a	
yem	28	$f_hourly_wage_lindi_10	n/a	
Factor_Con...	29	lindi = 11	n/a	
yem	29	$f_hourly_wage_lindi_11	n/a	
Factor_Con...	30	lindi = 12	n/a	
yem	30	$f_hourly_wage_lindi_12	n/a	
Factor_Con...	31	lindi < 1 | lindi > 12	n/a	
yem	31	$f_hourly_wage	n/a	

Task 5: Analyse the results

Run the two systems using 'AT_training_data_2' as this exercise requires the dataset to be uprated (see Exercise 14).

Figure 15.6: Running the two systems

Note: the model produces an error message warning users that four variables are uprated with default factor: 'WARNING: 2.1 uprate_at/Uprate (04d59373-087c-4958-862f-5e7a4a288583): variable(s) pyr, bac, bcc, bch is/are uprated with default factor (1)'. These are variables that are not used in the model and for which no uprating factor is defined. Thus, we can just ignore the message.

Use the In-depth Analysis tool to analyse the two output files. This tool is again under 'EUROMOD Statistics'. It allows you to specify what type of analysis you are interested. We will produce all tables for this exercise. You also need to select Baseline and Reform by clicking on 'Select baselines & reforms' and selecting the two AT 2021 systems. Finally, click the run button to produce the tables.

Figure 15.7: In-depth Analysis specification

Be aware that there are no differences between the tax-benefit policies applied, so all the differences come from the different evolution of salaries assumed from 2018 to 2021. Hence, the results should not be interpreted as the result of a reform, but rather as a sensitivity analysis of the uprating factors for employment incomes.

Figure 15.8: Fiscal results

1. Fiscal

1.1. Aggregate earnings, government revenue and expenditure (annual)

	Total Austria 2021 (Baseline)	Total Austria 2021 e15	Austria 2021 e15 Diff. w.r.t. Baseline	Austria 2021 e15 Diff. % of Baseline
+ employment income (yem)	128,561,727,207	132,054,932,989	3,493,205,782	2.7 %
+ self-employment income (yse)	10,201,413,236	10,201,413,236	0	0.0 %
+ COVID compensation paid by firm (yemmc_s)	0	0	0	0.0 %
Total earnings (ils_earns)	**138,763,140,443**	**142,256,346,225**	**3,493,205,782**	**2.5 %**

Figure 15.9: Inequality results

3.1. Inequality and redistributive effect of the tax-benefit system

	Austria 2021 (Baseline)	Austria 2021 e15	Austria 2021 e15 Diff. w.r.t. Baseline
Gini: A = original income	0.4995	0.5015	0.0021
Gini: B = A - taxes and social insurance contributions (EQ_INC23)	0.5077	0.5086	0.0010
Gini: C = B + pensions (EQ_INC22)	0.3006	0.3026	0.0020
Gini: D = C + other benefits (disposable income, EQ_INC20)	0.2192	0.2222	0.0030
Redistribution index = Gini first income - Gini last income	0.2802	0.2793	-0.0009
Social welfare index = mean eq. disp. income * (1-Gini last income)	1,796	1,810	13
Income quantile share ratio (S80/S20)	3.0343	3.0720	0.0377
Inter-decile ratio (D5/D1)	1.7608	1.7780	0.0172
Atkinson inequality index (inequality aversion parameter = 0.25)	0.0186	0.0191	0.0005
Mean log deviation	0.0000	0.0000	0.0000

Figure 15.10: Poverty results

3.3. At-risk-of-poverty rates for different poverty lines and definitions of income

	Austria 2021 (Baseline)	Austria 2021 e15	Austria 2021 e15 Diff. w.r.t. Baseline
Poverty line (annual) 40% of median baseline eq.disp.income	10,519.39	10,519.39	0.00
A = original income	33.2 %	33.2 %	0.0pp
B = A - taxes and social insurance contributions (EQ_INC23)	42.2 %	41.7 %	-0.4pp
C = B + pensions (EQ_INC22)	14.3 %	14.3 %	0.0pp
D = C + other benefits (disposable income, EQ_INC20)	0.0 %	0.0 %	0.0pp
Poverty line (annual) 50% of median baseline eq.disp.income	13,149.24	13,149.24	0.00
A = original income	37.0 %	37.0 %	0.0pp
B = A - taxes and social insurance contributions (EQ_INC23)	47.2 %	46.7 %	-0.5pp
C = B + pensions (EQ_INC22)	18.3 %	18.1 %	-0.2pp
D = C + other benefits (disposable income, EQ_INC20)	5.5 %	5.5 %	0.0pp
Poverty line (annual) 60% of median baseline eq.disp.income	15,779.09	15,779.09	0.00
A = original income	41.1 %	41.1 %	0.0pp
B = A - taxes and social insurance contributions (EQ_INC23)	53.6 %	52.5 %	-1.1pp
C = B + pensions (EQ_INC22)	24.1 %	23.7 %	-0.4pp
D = C + other benefits (disposable income, EQ_INC20)	15.0 %	14.9 %	-0.1pp

Exercise 16: Introducing a benefit for secondary education in Simpleland

The aim of this exercise is to implement a new benefit from scratch in Simpleland and analyse it with the in-depth analysis tool.

Description

Unlike real EU countries, Simpleland has a very simple tax-benefit system with a reduced number of taxes and benefits. A typical benefit that is missing is an education benefit, so we propose to include one. For learning purposes, we choose to implement a benefit only for students enrolled in secondary education, with the following features:

Entitlement	Households meeting the means test and with at least one person enrolled in secondary education
Means-test	Below 30,000 EUR/year of taxable income at household level
Benefit amount	100 EUR/year per person enrolled in secondary education
Interactions with the tax-benefit system	Non-taxable and not subject to SIC Part of the means test for social assistance

Implement this benefit in Simpleland and use the In-depth Analysis tool to analyse its impact.

Steps

Task 1. Check if the variable exists and add it if needed.

Task 2. Create a new system by copy-pasting the baseline.

Task 3. Create a new EUROMOD policy. Make sure that you place it correctly in the spine.

Task 4. Add the variable the relevant income lists

Task 5. Run baseline and reform and analyse the output with the In-depth Analysis tool

Hints

- The output variable should be named bedes_s (benefit | education | secondary | simulated). It doesn't exist in the variable list, so you have to create it there.
- The new policy should be placed after PIT and SIC and before social assistance.
- The variable has to be added to simulated benefits, means-tested benefits and the income list used for the social assistance means test. In real countries, it should be also added to the income lists for education benefits, so that they are correctly classified in the Statistics Presenter; these income lists don't exist in Simpleland, but they can be created.
- There are several ways of coding the benefit with the functions learnt so far.

Solution and further information

Task 1. Check if the variable exists and add it if needed.

The most appropriate name for the output variable is bedes_s (benefit | education | secondary | simulated). Go to Administration Tools - Variables and search it using the Search box. Once you confirm it doesn't exist, create it using the Add Variable button. You just have to type it and EUROMOD will fill the full name, since the acronyms already exist (Figure 16.1). Don't worry about the order, EUROMOD will reorder everything alphabetically once you close the Variable list.

Figure 16.1: Creating a new variable

Task 2. Create a new system

Like in previous exercises, create first a new system named. You can do this by right clicking on the heading of SL_1996 → Copy/paste system and name it SL_1996_E16.

Task 3. Create a new EUROMOD policy

The only restriction in terms of position of the policy is that it has to be placed before social assistance (bsa_sl), since the new benefit is part of the means test. Since it's not taxable by PIT of SIC, the position in relation to those policies is irrelevant, but for clarity we may place it immediately before bsa_sl (there is no doubt that it doesn't affect any other policy). In order to do this, right-click on bsa_sl → Add Policy Before → Benefit and name it bedes_sl.

Figure 16.2: Adding a new policy

The benefit can be easily simulated with the combination Elig + BenCalc, as shown in Figure 16.3.

Figure 16.3: Adding functions and parameters

10	bedes_sl		n/a	on	**BEN: Secondary education benefit**
10.1	fx Elig		n/a	on	Means test
10.1.1	Elig_Cond		n/a	il_TaxableY < 30000#y	Households with taxable income below 30,000 EUR/year
10.1.2	Result_Var		n/a	sin01_s	Saved as temporary variable for checks
10.1.3	TAX_UNIT		n/a	tu_household_sl	
10.2	fx BenCalc		n/a	on	Computation of benefit
10.2.1	Who_Must_Be_Elig		n/a	one	For households meeting the means test...
10.2.2	Comp_Cond	1	n/a	dec = 3 | dec = 4	.. for each person secondary education...
10.2.3	Comp_perElig	1	n/a	100#m	... 100 EUR/month
10.2.4	Output_Var		n/a	bedes_s	
10.2.5	TAX_UNIT		n/a	tu_household_sl	

Some comments:

- The income test relies on taxable income, which is already defined in an income list in the spine of Simpleland (il_TaxableY).
- ResultVar in 10.1.2 allow us to save the dummy of Elig in a temporary variable named sin01_s. Remember that for this specific type of variables you don't need to use the DefVar function before. In this case we only create the variable for testing purposes, in case we want to check the microdata.
- Note that the dummy will be 1 for all the individuals of the eligible households. This is why in 10.2 it's enough if one person is eligible (selecting All would have the same effect).
- The variable dec states the current education of status of the individual; 3 and 4 are the values that correspond to secondary school (you can check this information in the DRD codebook in your input folder).
- Note that we give 100 per eligible individual (Comp_perElig), not per household (for this we should use Com_perTU).

Task 4. Add the variable to the relevant income lists

The variable bedes_s has to be added to the following income lists: simulated benefits (ils_bensim), means-tested benefits (ils_benmt) and the income list used for the social assistance means test (il_bsa). The standard lists (ils_*) can be found in the policy ILsDef_sl, and the non-standard (il_bsa) in ILDef_sl. You can add the variable by adding a placeholder with the Add parameter form option in the corresponding function, either right-clicking or pressing CTRL+A (see Figure 16.4).

Figure 16.4: Adding a parameter to a DefIl function

In real countries, it should be also added to the income lists for education benefits, so that they are correctly classified in the Statistics Presenter; these income lists don't exist in Simpleland, but they can be created.

Task 5. Run baseline and reform and analyse the output with the In-depth Analysis tool

Run both baseline and reform scenarios. Once done, go to Applications → EUROMOD Statistics → In-depth Analysis. Since this tool allows for full customisation, we take advantage of it by going to the Advanced settings menu and selecting the indicators we want to customise. An example of customisation of 2. Distributional settings is shown in Figure 16.6. Once customised, you click on ok and select also the tables you want to be computed. If you want to save this customisation, you can do it by clicking on File → Save (Figure 16.7).

Then you just have to Select baselines (sl_1996_std.txt) and reforms (sl_1996_e16_std.txt) by clicking on the corresponding button, and click on Run. Figure 16.8 shows (partially) one of the tabs of the output. You can save the output by clicking on the top-right button.

Figure 16.6: Customising the distributional tables

Figure 16.7: Saving the customisation

Figure 16.8: Checking the results

In-depth Analysis

1. Fiscal | 2. Distributional by deciles of equivalised ils_dispy | 2. Distributional by HH Type | 2. Distributional by values of les | 2. Distributional by values of dgn | 2. Distributional by Age group | 3. In

2. Distributional

by values of hh_type

2.1. Individuals receiving the secondary education benefit

bedes_s

	Total SimpleLand 1996 (Baseline)	Total SimpleLand 1996 e16	Share SimpleLand 1996 (Baseline)	Share SimpleLand 1996 e16	Total SimpleLand 1996 e16 Diff. w.r.t. Baseline	Share SimpleLand 1996 e16 Diff. w.r.t. Baseline
One adult < 65, no children	0	0	0.0 %	0.0 %	0	0.0pp
- Female adult	0	0	0.0 %	0.0 %	0	0.0pp
- Male adult	0	0	0.0 %	0.0 %	0	0.0pp
One adult >= 65, no children	0	0	0.0 %	0.0 %	0	0.0pp
- Female adult	0	0	0.0 %	0.0 %	0	0.0pp
- Male adult	0	0	0.0 %	0.0 %	0	0.0pp
One adult with children	0	5,058	0.0 %	26.5 %	5,058	26.5pp
- Female adult	0	4,465	0.0 %	26.4 %	4,465	26.4pp
- Male adult	0	593	0.0 %	27.6 %	593	27.6pp
Two adults < 65, no children	0	0	0.0 %	0.0 %	0	0.0pp
Two adults, at least one >= 65, no children	0	0	0.0 %	0.0 %	0	0.0pp
Two adults with one child	0	8,432	0.0 %	10.3 %	8,432	10.3pp
Two adults with two children	0	8,996	0.0 %	7.2 %	8,996	7.2pp
Two adults with three or more children	0	2,791	0.0 %	5.5 %	2,791	5.5pp
Three or more adults, no children	0	0	0.0 %	0.0 %	0	0.0pp
Three or more adults with children	0	18,894	0.0 %	14.3 %	18,894	14.3pp
All	**0**	**44,170**	**0.0 %**	**5.0 %**	**44,170**	**5.0pp**