

Macroeconomic Model Data Base 3.1 - User Guide

This user guide describes how to install and use the Macroeconomic Model Data Base, version 3.1 (hereafter the Modelbase or MMB). Section 1 deals with the installation and the software requirements. Section 2 introduces the menu of the MMB 3.1 and describes how to run the software and conduct comparison exercises employing the models and options contained in the Modelbase. Section 3 explains the structure of the key files that govern the simulations carried out by the MMB 3.1. Lastly, section 4 lays out the structure of the model files. These are usual Dynare files, which have been amended with extra lines and commands to suit the MMB. For the time being we exclude instructions on how to alter the MMB by adding models or policy rules, which are planned to be included in the next release.

This guide is platform independent and can be applied to Windows, Linux and macOS operating systems.

1 Installation and software requirements

The MMB 3.1 can be downloaded from macromodelbase.com/download. The version for windows is `mmb-electron-win.exe`, for macOS it is `mmb-electron-mac.dmg`, and for Linux directly download the source code from the release on github at <https://github.com/IMFS-MMB/mmb-gui-electron/tags>.¹ The Windows version and the Mac version of the file autoinstall the MMB on your computer and opens the redesigned frontend of the MMB.

The files for carrying out the simulations of the models are written in MATLAB, so either some version of MATLAB or a recent version of its freeware clone, OCTAVE, must be installed on your computer. In the case of Matlab, one also needs the *Optimization Toolbox* as well as the *Statistics Toolbox* in order to be able to run all models in the Modelbase.² For model solution the program utilizes DYNARE, which can be downloaded free of charge from the web.³ Under Windows, double-clicking on the downloaded DYNARE exe-file opens a set of steps that guide you through the installation. Under macOS, locate the downloaded pkg-file in Finder, and Control-click the icon to select Open from the menu, thus creating an exception for the app to be installed. The installer will then guide you through the installation. To install Linux version of Dynare please follow the instructions on the Dynare Wiki⁴.

Compatibility

REVISE THIS SECTION

We have tested the MMB 3.0 with DYNARE 4.5.6 and 4.5.7. Earlier versions may work but have not been tested.

¹Linux users will have to build from source using `npm`. Find more info on our github page.

²For the time being there are some models, which cannot be simulated with Octave. The list of models contains: NK_GK11 and EA_Q14. The problems with the latter model exist only with Octave 4.4.0.

³<http://www.dynare.org>

⁴The DYNARE Wiki install guide for Ubuntu and Debian can be found at <http://www.dynare.org/DynareWiki/InstallOnDebianOrUbuntu>

On Windows, DYNARE 4.5.6 is compatible with OCTAVE 4.4.0, whereas DYNARE 4.5.7 is compatible with OCTAVE 4.4.1. Both Dynare versions are compatible with MATLAB R2007b and later. For macOS, the compatibility between DYNARE and MATLAB is the same. However, at the time of this release, the highest OCTAVE-supported version of DYNARE is 4.5.6 (compatible with OCTAVE 4.4.0 on macOS).

Further steps before running comparisons

When using MATLAB, one has to add the DYNARE path to MATLAB. In order to do so, open MATLAB and choose *Set path* from the *File* menu. Use the option *Add folder* and browse to the directory where you have installed DYNARE. The DYNARE subfolder that has to be added is called *MATLAB*. When using OCTAVE, one adds the DYNARE path by opening the command window of OCTAVE (it should be open automatically) and typing "*addpath directoryofdynare*". If you don't change the installation directory of OCTAVE you should type "*addpath C:\dynare\4.5.x\matlab*", where x completes the version of dynare.

Before running simulations with the MMB, you need to specify whether you want to run the simulations in MATLAB or OCTAVE. In order to do so, click on 'Menu' on the upper-right corner of the MMB and then on 'Settings', as shown in the Figure 'Settings'. It opens a window, in which you have the option either to let the program scan for versions of MATLAB and OCTAVE installed on your computer, or to search manually. If the scan finds more than one version of MATLAB or OCTAVE, you can choose from the list of programs in this subwindow by clicking on the small arrows on the right side. Note that the scan only looks in common directories as scanning the whole system would take too long. If the scan does not find the desired installation, click on 'Find manually', select whether you want to add MATLAB or OCTAVE to the list of executables and browse to the folder of the installation. Select the executable file which opens the program and click on 'Open'. In case of MATLAB you should browse to "...*\matlab\version\bin\matlab.exe*". Accordingly, for OCTAVE this is "...*\octave\version\bin\octave-cli.exe*". If you want to remove an entry from the selection menu, select it and click on 'Remove selected'.

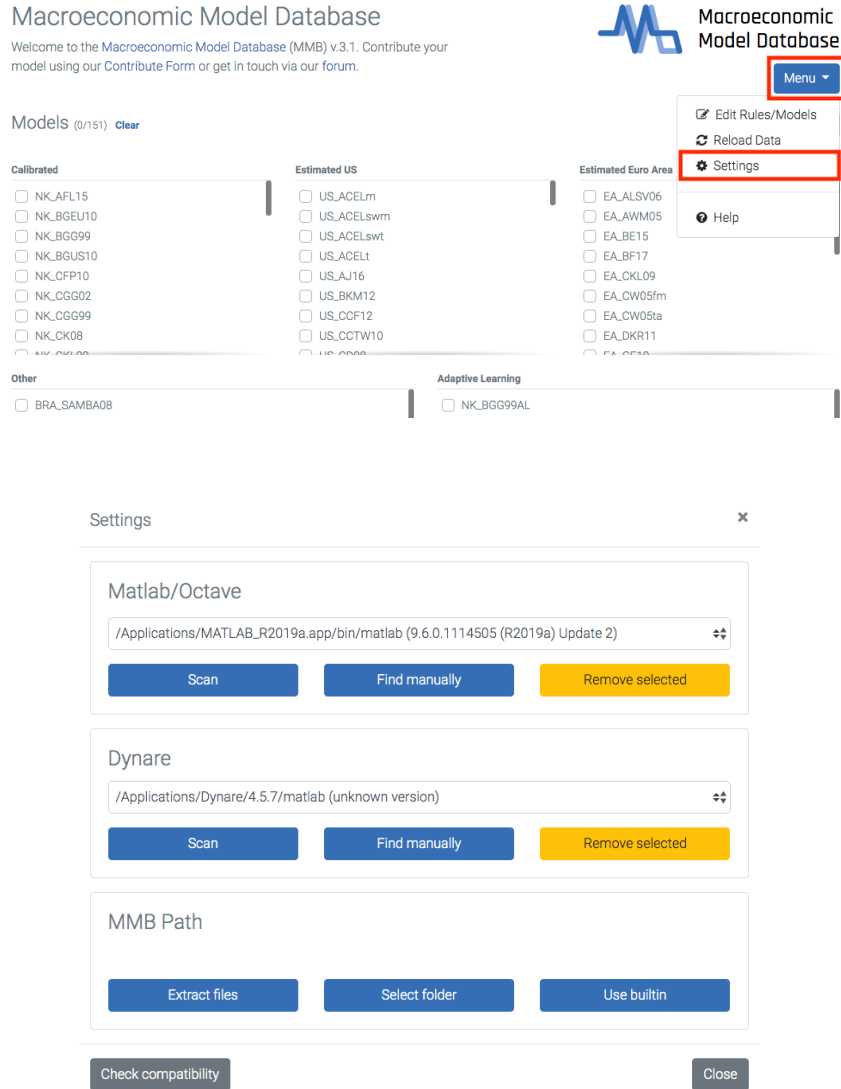
You can follow the same steps to select or delete a version of DYNARE on which you want to run the simulations.

Finally, the 'Settings' menu also allows you to set a path of the MMB. You can either select a folder by clicking directly into the path field and extract the files shipped with the MMB into this folder by clicking on 'Extract files'. You can also click on 'Select folder' to choose a folder in which you already have installed the MMB. This option can be used if you have installed more than one version of the MMB, e.g. an original one and one in which you changed or added models or policy rules. Thirdly, you can click on 'Use builtin' to reset the folder and use the default MMB files shipped with user interface you are using.

To check whether your installations of MATLAB/OCTAVE and DYNARE are compatible with the MMB, click on 'Check compatibility' in the lower left corner. This check can take up to 30 seconds.

You can close the 'Settings' window by clicking on 'Close' in the lower right corner or on the cross in the upper right corner.

Figure 1: SETTINGS



2 The Modelbase: Models, Rules and Options

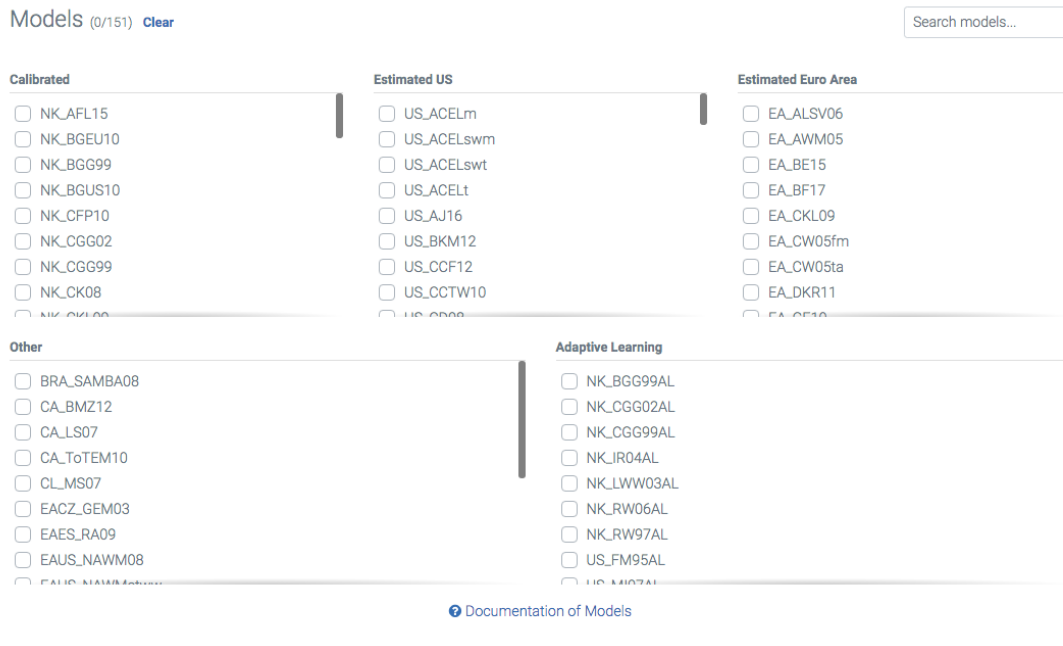
This section introduces the menu of models, policy rules and options, which can be selected in the MMB 3.1.

Models

For the time being there are some models, which cannot be simulated with Octave. The list of models contains: NK_GK11 and EA_Q14. The problems with the latter model exist only with Octave 4.4.0. We will address these issues in the next release.

The user can select from the number of models displayed in the frontend, and from the number of policy rules. In contrast to earlier versions, the MMB 3.1 simultaneously allows for the selection of more than one policy rule and more than one model (the earlier versions either only allowed for one model, many rules or one rule, many models). So, the user can select policy rules, shocks and variables, which are common in all selected models or remove common variables, which are automatically selected.

Figure 2: MODELS MENU



The models are sorted in columns, the first of which contains models that have been calibrated to match a closed economy (NK_XXX). The second column lists models that have been estimated on US data (US_XXX). The third column lists models that have been estimated on Euro area data (EA_XXX). The column 'Other' contains models that have been either calibrated or estimated on multi-country data, or have been estimated on other countries, such as 'CA_BMZ12', which has been estimated on Canadian data, or 'EAUS_NAWM08', which has been estimated on Euro area and US data in a two-economy-setting. The last column contains models in which agents form their expectations via adaptive learning (XXXAL).

Hovering with the mouse over the models in the list and over the common policy rules displays some basic information such as the title, author and academic reference of the article, in which the model was used. To search for models in the MMB, one can also use the search line on top of the menu. Here one can search for the name of the author or words that appear in the title of the articles, in which the models were used as well as journals or year of publication. For instance, searching for Galí, yields the results shown in Figure 3.

Figure 3: MODELS SEARCH: GALÍ

Models (0/128 Selected) [Clear](#)

galí

Calibrated **Estimated US** **Estimated Euro Area** **Other** **Adaptive Learning**

- ☐ NK_BGEU10
- ☐ NK_BGUS10
- ☐ NK_CGG02
- ☐ NK_CGG99
- ☐ NK_GLSV07
- ☐ NK_GM05
- ☐ NK_GM16

- ☐ NK_CGG02AL
- ☐ NK_CGG99AL

In addition, the user can download a complete list of all models currently used in the MMB on macromodelbase.com/downloads. Lastly, short descriptions of the models and their features are available on the same page. The menu of the MMB contains a direct link to the model descriptions below the model selection menu as shown in Figure 4.

Figure 4: MODEL DOCUMENTATION

Other **Adaptive Learning**

- ☐ BRA_SAMBA08
- ☐ CA_BMZ12
- ☐ CA_LS07
- ☐ CA_ToTEM10
- ☐ CL_MS07
- ☐ EACZ_GEM03
- ☐ EAES_RA09
- ☐ EAUS_NAWM08

- ☐ NK_BGG99AL
- ☐ NK_CGG02AL
- ☐ NK_CGG99AL
- ☐ NK_IR04AL
- ☐ NK_LWW03AL
- ☐ NK_RW06AL
- ☐ NK_RW97AL
- ☐ US_FM95AL

[Documentation of Models](#)

Monetary policy rules

Currently, we consider nine monetary policy rules that are taken from Taylor (1993), Levin, Wieland and Williams (2003), Smets and Wouters (2007), among others. Next to these common rules that can (in principle) be used for all models, the selection menu also features the options model specific rule and user-specific rule. The model specific rules are taken from the original articles, in which the models were used. Not all models can be simulated with their model-specific rule, as all interest rate rules that can be used in the modelbase, have to be expressed in terms of the common variables. These common variables are the quarterly output gap, quarterly output, the year-on-year rate of inflation and the policy interest rate in annual terms. As some of the interest rate rules used in the literature feature responses to financial indicators, exchange rates, etc. these rules cannot be included in the modelbase. Therefore only 93 of the 128 models included in the MMB 3.0 feature a model-specific rule.

Figure 5: POLICY RULES MENU

Policy Rules (0/9) [Clear](#)

- ☐ User specified rule (edit)
- ☐ Model specific rule
- ☐ Christiano et al. (2005)
- ☐ Christiano et al. (2014)
- ☐ Coenen et al. (2012)
- ☐ Gerdesmeier & Roffia (2004)
- ☐ Levin et al. (2003)
- ☐ Orphanides & Wieland (2008)
- ☐ Orphanides & Wieland (2012)

[Documentation of Policy Rules](#)

Shocks (0/2) [Select all](#) [Clear](#)

- ☐ Monetary Policy Shock
- ☐ Fiscal Policy Shock
- ☐ Model Specific Shocks

Variables (4/4) [Select all](#) [Clear](#)

- ☒ Inflation
- ☒ Interest
- ☒ Output
- ☒ Output Gap
- ☐ Model Specific Variables

Options

- ☐ Plot autocorrelation functions
- ☐ Plot variances

Horizon: 20

Gain: 0.01

[Select states](#)

[Compare](#)

[Need help?](#)

The user specific rule allows the user to directly determine the feedback coefficients in the interest rate rule. In order to do so, the user has to click on ‘(edit)’ next to the entry ‘User specific rule’ as shown in figure 5. Then, the subwindow shown in figure 6 opens. In this example, the coefficients for current and lagged inflation rates, as well as for the current output gap are chosen such as to mimic the pre-programmed Taylor rule. The time indices refer to quarters. Click on ‘OK’ to set the policy rule or on ‘Cancel’ to discard changes.

The user can edit the coefficients of the responses of the policy rates to the leads and lags of the variables in this submenu. However, it is not guaranteed that the Blanchard-Kahn conditions will hold and a determined equilibrium is obtained in the solution of the models with this rule.

Figure 6: MODEL DOCUMENTATION

User Specified Rule
×

	interest	inflation	output gap	output
t		0.375	0.5	0
t-1	0	0.375	0	0
t-2	0	0.375	0	0
t-3	0	0.375	0	0
t-4	0	0	0	0
t+1		0	0	0
t+2		0	0	0
t+3		0	0	0
t+4		0	0	0

Cancel
OK

In general, when a model (or a pre-specified rule) is chosen that is not compatible with a policy rule (or a model) in the way that a simulation of this model-rule-combination does not render the rational expectation equilibrium locally unique, the respective policy rule (or model) is faded out in the menu and cannot be selected for the comparison exercise. Unselecting the model (or rule), makes the rule (or model) again accessible for selection.

Output options

Having chosen the models and a policy rule, the user can make some non-exclusive choices regarding the exercise outcomes to be displayed. The user can decide whether to see the unconditional variances and plot autocorrelation functions of the common variables, both of which are computed using theoretical moments of the solution for each variable. Also the user can opt for plotting impulse response functions of the common variables and specify the horizon for the analysis that is set to twenty periods as a default.

As default the common MMB variables (quarterly output gap, quarterly output, the year-on-year rate of inflation and the policy interest rate in annual terms) are selected. If only one model is chosen, all other variables as defined in the mod-file are also available. As a new feature in the MMB 3.1, some more variables which are featured in many of the models, but not in all of them (among others consumption, investment and capital) can be selected for comparison if available in all selected models.

One can choose impulse responses to a unit monetary policy shock (one percent point increase in the monetary policy shock), and/or to a unit fiscal policy shock (one percent increase in GDP share of government expenditures). Note that all models of the Modelbase have a monetary policy shock, but a

significant number of them do not have a fiscal policy shock. If this is the case, the impulse responses to a fiscal policy shock will not be available. Again, if only one model is selected, all model-specific shocks as defined in the mod-file are available for selection. Alongside the new comparable variables, also a number of shocks appearing in many models are now available for comparison (e.g. Technology shock, Preference shock).

When you select an adaptive learning model, you can also set the gain parameter and select the states by clicking on ‘Select states’. In the window that pops up you can select the states individually for all adaptive learning models you have selected for the comparison.

Lastly, for certain models, the unconditional variances are not defined and the autocorrelation functions do not exist. This is the case for some models which feature unit roots. The presence of unit roots prevents a calculation of unconditional moments of some or all variables. Nonetheless, for these models IRFs can still be generated.

Menu

In the upper right corner of the MMB GUI you can find the button ‘Menu’. When clicking the button, a drop-down menu opens, where you can click four buttons.

Firstly, ‘Edit Rules/Models’ opens a sub-window with all models included in the MMB. Here you can manually find the model you want to edit. After clicking on a model name, you can see its two relevant files, namely the .json and .mod file. The .json file contains basic information about the model, like paper title, author and journal, but it also contains information about the model capabilities, i.e. if the model is capable of producing the common variables (output, inflation, interest rate, output gap) and common shock (monetary and fiscal policy shock). As an example, *fiscal_shock": false* means that the model does not contain a fiscal policy shock. You can also see if unconditional variances are possible for this model and if it is an adaptive learning model. Furthermore, under *rules* you can see which policy rule is eligible for the model you want to edit, while under *msr* you can see and change the coefficients for the model specific policy rule. In the section below, you can see the name and the text of each shock and below that of each variable contained in the model. In case you want to change the name of a shock or variable appearing in the main-window, please change its text to the desired text. Hint: The text is case sensitive, so if you want to change the name of a variable for many models, be consistent with the text. The other file appearing in for the selected model is the .mod file. This file consists of a preamble defining all variables, shocks and parameters. It is followed by the parameter initialization and the model equations, which define the sticky price and wage economy and the flex economy, as well as shocks. In the case of non-zero steady state values of variables, there usually also exists a separate *modelname_steadystate.m* file, which defines the steady state of the model. In the upper right corner of the sub-window you can save the changes of the current file. It is important to keep in mind that you need to save the file before switching to another file because then unsaved changes are gone. Also, saving the changes does not apply them. You need to reload the data. At any time you can close the sub-window by clicking the ‘Close’ button in the lower right corner of the sub-window.

The second button in the drop-down menu is ‘Reload Data’ and applies the saved changes made to the files, i.e. renaming a shock or variable.

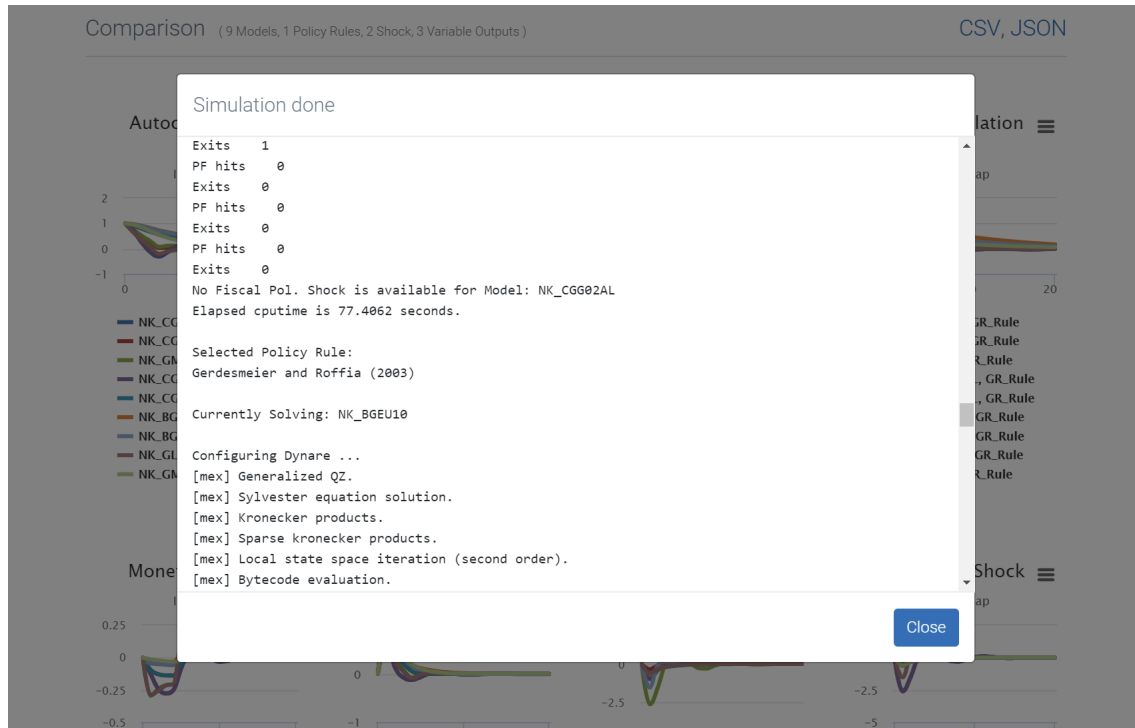
The third button ‘Settings’ is explained in section *Installation and software requirements*.

The last button ‘Help’ opens a help window.

Running a comparison exercise

Once you have selected a number of models, rules and output options, you can simply run the comparison exercise by clicking on the button ‘Compare’. When the simulations run the command window from MATLAB/OCTAVE will be embedded and you see the running output (see, Figure 7). Once the simulations are finished you can close the window by clicking on ‘Close’.

Figure 7: MODEL SIMULATION



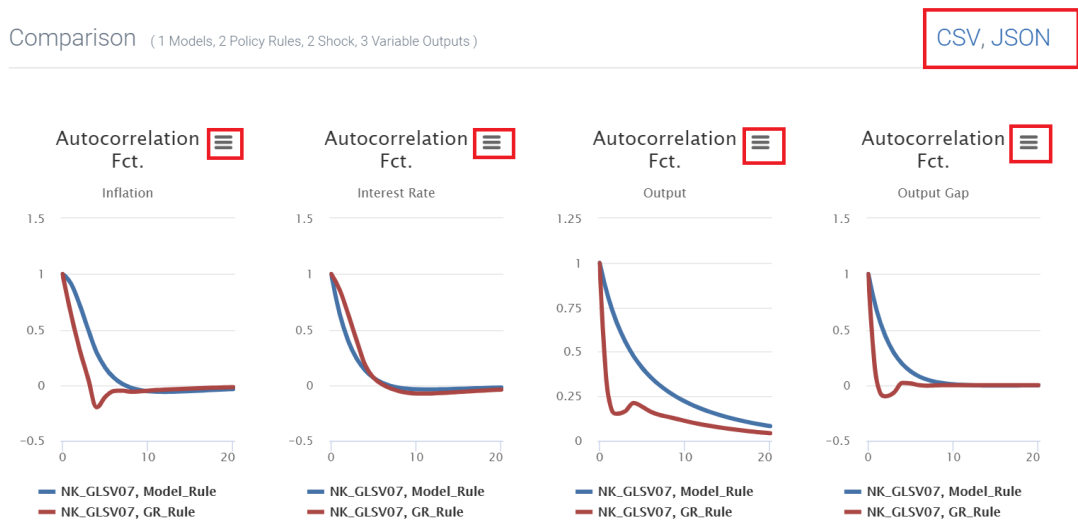
As a result, the selected Impulse response functions and autocorrelation functions are plotted, and the variances are displayed in a table at the bottom of the frontend. There are different options available to display the results (see figure 8). First, the user can choose whether results should be displayed in charts of variables, models or policy rules. This can be selected by clicking on ‘Group Data’ and then the respective option. The default is to group outputs by variable, i.e. the charts represent variables and the graphs in the charts represent model/policy rule-combinations. Further, the user can select the ‘maximum number of columns per row’, which might be useful to increase readability or clarity depending on the screen used. Finally, the user can choose which graphs are displayed in the charts by clicking on the respective entries of the caption.

Figure 8: DISPLAY OPTIONS



The results of the comparison exercise can be exported and saved in two ways. Either, one exports all the data, by clicking on CSV or JSON above the figures to receive a batch export in the respective file format, or one can export the figures one by one, by clicking on the menu on the upper right corner of each figure. Both options are marked with red boxes in Figure 9.

Figure 9: EXPORT OPTIONS



3 Structure of the Modelbase

This section describes the key folders, in which the models, options, and rules are stored and briefly sketches the structure of the key files, which are used in the execution of the modelbase.

Key Folders

Your installation of the MMB 3.1 contains a subdirectory 'resources/app/dist/electron/static/mmci-cli', which contains the .m-files and .mod-files for the models, rules and options being used in the comparison exercises. The subfolder *MODELS* contains a specific folder for every model included in the MMB. The specific folders contain a DYNARE mod-file in which the particular model is specified together with related MATLAB files, some of which are created by DYNARE, as well as a json-file, which is needed to link information displayed in the user interface to the corresponding mod-file.

Some Key files

The files discussed in this subsections are stored in the folder *MMB_OPTIONS*.

- **CMB_MMB.m**

This file receives the information on the selection of models rules and options that the user determines in the frontend of the modelbase. It checks, which operation platform is used, whether MATLAB or OCTAVE is employed and which version of Dynare is being used. Then it adds all other folders from the directory 'mmci-cli' to the path of MATLAB or OCTAVE. Then it loads relevant settings from *MMB_settings.m* and defines key variables and names as well as a blank structure for the JSON, which will contain the simulation results as they are passed on to the frontend. The core of the file is a large loop over each selected model (index 'epsilon') and each selected policy rule (index 'i'). In each run of the loop, it sets the coefficients of the policy rule, solves the model in dynare, stores the results in the structure *Modelbase.mat* and passes the demanded output of the simulation on to the file *Modelbase.json*. The files *Modelbase.mat* and *Modelbase.json* are available in the same folder even after the simulations.

- **MMB_settings.m**

Among others this file contains the list of model names in the vector 'names', it sorts the models into categories calibrated, estimated on US data, etc. and stores for each model the dimension of the shocks in the original model ('variablenum'). Furthermore it contains the vectors with the rulenames and specifies which models have a model-specific rule, and which have not. In the next part, it defines the coefficients of the common policy rules. Lastly, it defines some options for the adaptive learning models.

- **MSR_COEFFS.m**

This file contains the list of coefficients for all model specific rules.

Some key files

In earlier versions, the MMB had the subfolder *MMB_OPTIONS* containing specific MATLAB files related to the usage of the Modelbase for policy analysis and comparison, as well as explanatory notes for models and policy rules. The folder *ALTOOLS* contained scripts for the use of models with

adaptive learning. These folders have been removed. The information is mostly contained in the .json file in the models folder.

The files discussed here are stored in the folder MODELS and are a brief introduction. For further information read section *Structure of the model files*.

- **modelname.mod**

This file contains the model variables, shocks and parameters as well as the model equations including the flex economy, the sticky price and wage economy and shocks. If the model is not linear, then sometimes the file also contains an *initval* block, which provides guess values for non-linear solvers and steady state computations.

- **modelname.json**

This file contains information about the capabilities of the model. It defines, which of the policy rules, the four common variables and two common shocks are eligible for this model. It also states if unconditional variances can be calculated. Additionally, the file contains general information about the model (e.g. paper, author,...). The model specific policy rule is included if it is available and also contains the exact parameter values. It is also stated if the model includes adaptive learning and if so it shows which variables are forwards, states_short and states_long. For all models, the name and text of all shocks and variables can be seen and changed in this file.

- **optional: modelname_steadystate.m**

In case of non-zero steady states, this file is included in the model specific folder and contains the steady state equations or values.

4 Structure of the model files

Mod files

The model files are written in the syntax of DYNARE and have a common structure. As an example we take the simple New-Keynesian model by Rotemberg and Woodford (1997) to explain the structure of the mod-files, its model specific parts and the common model data base blocks. The current example is based on the DYNARE 4.5.6 version of the Modelbase. The mod-file is shown in **Figure 10** and **Figure 11**. However, the explanations apply to all models. In the following, the two main parts of a mod-file, the preamble and the model block, are described step by step.

Figure 10: STRUCTURE OF THE MODEL FILES: THE PREAMBLE

```

1 // Model: NK_RW97
2
3 var pi y ynat rnat i x u g g_
4 //*****
5 // Modelbase Variables                                     /**
6     interest inflation inflationq outputgap output fispol;    /**
7 //*****
8
9 varexo u_
10 //*****
11 // Modelbase Shocks                                       /**
12     interest_ fiscal_;                                       /**
13 //*****
14
15 parameters
16 //*****
17 // Modelbase Parameters                                   /**
18                                     /**
19     cofintintb1 cofintintb2 ... coffispol                    /**
20 //*****
21     beta sigma alpha theta omega kappa rhog rhog stdinflation_ stdfiscal_;
22
23 beta = 1/(1+0.035/4); // 0.9913
24 sigma= 6.25;
25 alpha= 0.66;
26 theta= 7.66;
27 omega= 0.47;
28 kappa= (((1-alpha)*(1-alpha*beta))/alpha)*(((1/sigma)+omega)/(1+omega*theta));
29 rhog=0;
30 stdinflation_=0.154;
31 rhog= 0.8;
32 stdfiscal_=1.524;
33
34 //*****
35 // Specification of Modelbase Parameters                 /**
36                                     /**
37 // Load Modelbase Monetary Policy Parameters            /**
38 thispath = cd; cd('..');
39 load policy_param.mat;
40 for i=1:33
41     deep_parameter_name = M_.param_names(i,:);
42     eval(['M_.params(i) = ' deep_parameter_name ' ;'])
43 end
44 cd(thispath);
45 // Definition of Discretionary Fiscal Policy Parameter    /**
46 coffispol = 1;                                           /**
47 //*****

```

Figure 11: STRUCTURE OF THE MODEL FILES: THE MODEL BLOCK

```

49 model(linear);
50
51 //*****
52 // Definition of Modelbase Variables in Terms of Original Model Variables /**
53
54 interest = i*4; /**
55 inflation = (1/4)*(4*pi+4*pi(-1)+4*pi(-2)+4*pi(-3)); /**
56 inflationq = pi*4; /**
57 outputgap = x; /**
58 output = y; /**
59 fispol = g_; /**
60 //*****
61
62 //*****
63 // Policy Rule /**
64 /**
65 // Monetary Policy /**
66 /**
67 interest = cofintintb1*interest(-1) /**
68           + cofintintb2*interest(-2) /**
69           ... /**
70           + cofintoutpf4*output(+4) /**
71           + std_r_*interest_; /**
72 /**
73 // Discretionary Government Spending /**
74 /**
75 fispol = coffispol*fiscal_; /**
76 //*****
77
78 // Original Model Code:
79
80 pi = beta * pi(+1) + kappa*x + u;
81 u = rhou*u(-1) + u_;
82 x = x(+1) - sigma * (i - pi(+1) - rnat) ;
83 rnat = sigma^(-1)*((g-ynat) - (g(+1)-ynat(+1)));
84 ynat = sigma^(-1)*g / (sigma^(-1)+omega);
85 x = y-ynat;
86 g = rhog*g(-1) + g_;
87 // i=hipi*pi + phix*x;
88 end;
89
90 shocks;
91 var fiscal_ = 1.524^2;
92 var u_ = 0.154^2;
93 end;
94
95 //stoch_simul (irf = 0, ar=100, noprint);

```

Part 1: The preamble

- Each model file begins with some information about the model. This should include the title, the authors, the publication etc. In front of this description you will find the symbols `//`, which denote a comment in DYNARE.
- The file then starts with the initialization of the model variables. In our example shown in **Figure 10** the model-specific endogenous variables are listed in line 3 after the keyword `var`: `pi`, `y`, `ynat`, `rnat`, `i`, `x`, `u`, `g` and `g_`. The latter in fact represents an exogenous government spending shock, however it has to be initialized as endogenous variable for reasons that will be explained below. It follows a Modelbase block in lines 5 to 8 in which the common variables are introduced. In general, Modelbase blocks are separated through `//*****` symbols from the rest of the file.
- Following the keyword `varexo` in line 10 the exogenous variables are initialized. In our example this is `u_`, a cost push shock as well as the common interest rate shock, `interest_` and the common fiscal policy shock, `fiscal_` in line 13. Note that in some models with no treatment of government spending, the latter Modelbase shock may be left out.
- Following the keyword `parameters` in line 16, the Modelbase parameters in the Modelbase block are initialized. In **Figure ??** line 20 we have, for brevity reasons, only included three policy parameters. In the actual mod-files there are many more leads and lags. These are the parameters of the general monetary policy function, except for the last one, `coffispol`, which enters the common discretionary government spending equation.
- Then the model-specific parameters are initialized in line 22.
- Afterwards numerical values are assigned to the model-specific parameters in lines 24 to 33.
- Finally a block called *Specification of Modelbase Parameters* is added. First in lines 38 to 46 the numeric values of the parameters of the selected monetary policy rule are loaded. They are contained in the file `policy_param.mat` in the subfolder `MODELS`. For models in which the original shocks are expressed in percent/100 the parameter `std_r_` has to be reset to 100 after the parameter-loading command. In our example this would have to be done in line 45. However, the shocks in this model are already expressed in percentage terms. Secondly, the discretionary fiscal policy parameter `coffispol` is defined as a function of the model-specific parameters in order to obtain a government spending shock of one percent of GDP. The exact implementation of the common fiscal policy shock will be described below. In our example no adjustment is needed and hence `coffispol` is set equal to one.

Part 2: The model block

- The model block starts in line 52 of **Figure ??** as indicated by the keyword `model` followed by `linear`, which tells DYNARE that the equations are already linearized and thus reduces computing time.
- In the Modelbase block going from lines 54 to 63 the common variables are defined in terms of the original model variables. The variable `interest` denotes the annualized short-term interest rate, `inflation` is annual inflation, `inflationq` represents annualized quarterly inflation, `outputgap`

and *output* denote the output gap and output, respectively. The common variable *fispol* represents discretionary fiscal policy. It is set equal to the model-specific government spending shock variable, which in the case of our example is *g_*. Note again, that this model-specific shock has to be initialized as an endogenous variable. This allows us to keep the original model equation for government spending unchanged.

- It follows the common *Policy Rule* block. In lines 70 to 74 the common monetary policy rule is specified. Again for reasons of brevity we have not displayed the complete general policy rule in **Figure ??**. Below in line 78, the common equation for discretionary government spending is specified.
- The original model equations are then specified in lines 83 to 90. Note that the model-specific monetary policy rule is commented out because the common policy rule is introduced. On the contrary, the government spending equation in line 89 has remained unchanged. The model section ends in line 91 with the required keyword *end*.
- Finally the variance covariance matrix is specified in lines 95 and 96 between the keywords *shocks* and *end*. Importantly, the variance of the original model-specific government spending shock has been assigned to the common fiscal policy shock variable *fiscal_*. Hence, the common shock *fiscal_* affects the fiscal policy variable *fispol* through the common discretionary government spending expression in line 78 which is set equal to the model-specific government spending shock *g_* in line 62.
- The *stoch_simul* command in line 99 is commented out. Alternatively one can also delete this command.

Json files

The json files have a common syntax. As take the same example as above to explain the structure of the json files. It can be seen in figure 12.

Every json file starts with a curved bracket. All objects need to be specified in " ". Firstly, it is specified what the json file includes, here it is a model, followed by the name of the model (*NK_RW97*). Here, the first letters determine which type of model it is (e.g. New Keynesian model: NK, Euro Area: EA, United States: US, or others), followed by a '_'. The next letters are set according to the last names of the authors (here: RW stands for Rotemberg and Woodford). The name is completed by the year of publication (97 stands for 1997, while 03 or 13 stands for 2003/2013). Optionally, AL is added to the model in case the model includes adaptive learning.

The first block you need to specify are the "*capabilities*" (lines 4-22). It defines, which of the four common variables and two common shocks are eligible for this model. If they are included, the value is set to *true*, else it is set to *false*. After that "*rules*" contains all policy rules, which are available for this model. Note that the set of available rules is written in square brackets. All unavailable policy rules appear in gray in the GUI and can not be chosen. It also states if unconditional variances can be calculated (line 21). So for *NK_RW97*, all common variables, shocks as well as unconditional variances are available. Rules 3-7 and 9-11 can be chosen.

The next block, "*description*" (lines 23-36), contains general information about the model, like reference, paper title, journal and year published, replicants name, keywords, description, category and authors. Again all sets are written in square brackets. This includes key words and authors. General information is shown when holding the mouse over a model. It is important to specify the

category of the model because it specifies in which model block the model appears in the GUI. Type *"Calibrated model"* for calibrated models, *"Estimated euro area model"* for EA models, *"Estimated US model"* for US models, *"Estimated other-country model"* for other countries or *"Adaptive learning model"* for adaptive learning models. Note that these entries are case sensitive.

Figure 12: STRUCTURE OF THE JSON FILES

```

1  {
2    "$schema": "model",
3    "name": "NK_RW97",
4    "capabilities": {
5      "fiscal_shock": true,
6      "inflation": true,
7      "interest": true,
8      "mp_shock": true,
9      "output": true,
10     "outputgap": true,
11     "rules": [
12       3,
13       4,
14       5,
15       6,
16       7,
17       9,
18       10,
19       11
20     ],
21     "unconditional_variances": true
22   },
23   "description": {
24     "ac_ref": "Rotemberg & Woodford (1997)",
25     "paper_title": "An Optimization-Based Econometric Framework for the Evaluation of Monetary Policy",
26     "journal": "NBER Macroeconomics Annual 12, pp. 297-346",
27     "replicants_name": "S. Schmidt",
28     "pub_date": "1997",
29     "keywords": [],
30     "description": "",
31     "category": "Calibrated model",
32     "authors": [
33       "Rotemberg, Julio",
34       "Woodford, Michael"
35     ]
36   },
37   "msr": null,
38   "variableldim": 1,
39   "al": false,
40   "shocks": [
41     {
42       "name": "fiscal_",
43       "text": "Fiscal Policy Shock"
44     },
45     {
46       "name": "interest_",
47       "text": "Monetary Policy Shock"
48     },
49     {
50       "name": "u_",
51       "text": "Price markup shock"
52     }
53   ],
54   "al_info": null,
55   "variables": [
56     {
57       "name": "fispol",
58       "text": "fispol"
59     },
60     {
61       "name": "g",
62       "text": "g"
63     }
64   ]
65 }

```

Next, there is a block for the model specific policy rule. The New Keynesian model by Rotemberg and Woodford does not have a model specific policy rule, so this entry is set to *null*. It is the policy rule used in the paper. Consider NK_CGG02AL as an example for msr and adaptive learning. Figure 13 (left) shows the syntax for the msr entry. As it is a set, you need to specify the policy rule coefficients

inside of square brackets.

Figure 13: STRUCTURE OF THE JSON FILES: MSR (LEFT) AND AL (RIGHT)

```

36  "msr": [
37    "0",
38    "0",
39    "0",
40    "0",
41    "1.5",
42    "0",
43    "0",
44    "0",
45    "0",
46    "0",
47    "0",
48    "0",
49    "0",
50    "0",
51    "0",
52    "0",
53    "0",
54    "0",
55    "0",
56    "0",
57    "0",
58    "0",
59    "0",
60    "0",
61    "0",
62    "0",
63    "0",
64    "0",
65    "0",
66    "0",
67    "0",
68    "1",
69    "0.25"
70  ],

71  "variabledim": 1,
72  "al": true,
73  "al_info": {
74    "forwards": [
75      "inflationq",
76      "infstar",
77      "ytildestar",
78      "ytilde",
79      "infl",
80      "ybar",
81      "ystar",
82      "ybarstar",
83      "y"
84    ],
85    "states_long": [
86      "astar",
87      "a",
88      "inflationq",
89      "inflationql",
90      "inflationql2",
91      "infstar",
92      "infstarl",
93      "infstarl2",
94      "rstar",
95      "interest"
96    ],
97    "states_short": [
98      "astar",
99      "a",
100     "inflationq",
101     "infstar",
102     "rstar",
103     "interest"
104   ]
105 }

```

Afterwards, the dimension of the shock is specified using *"variabledim"*. It is set to 1 if the shock is in percent, so the number 1 next to the graph means 1%, and 2 if it is in percent/100. In this case the number 1 next to the graph would mean 0.01%. However, MMB 3.1 takes this into account and standardizes it to %.

The next block is *"al"*, in which adaptive learning properties are specified. If the model does not include this feature, then the entry is set to *false*. Otherwise, it is set to *true*. Figure 13 (right) shows the structure of the code in the case of adaptive learning. In this case *"al_info"* is added, which contains three features, namely *forwards*, *states_short* and *states_long*. All of these features are a set of variables.

Lastly, there are blocks for *"shocks"* and *"variables"*, in which all shocks and variables of the model are listed. As the two objects are lists, the user needs to use square brackets. Every entry in the list is in curved brackets and separated by a comma. An entry contains a name and a text. The name is the same as in the mod file, while the text appears in the GUI. Note that the text is case sensitive.

References

- Levin, A., Wieland, V., Williams, J.C., 2003. The performance of forecast-based monetary policy rules under model uncertainty. *The American Economic Review* 93(3), 622–645.
- Rotemberg, J.J., Woodford, M., 1997. An optimization-based econometric framework for the evaluation of monetary policy. *NBER Macroeconomics Annual* 12, 297–346.
- Smets, F., Wouters, R., 2007. Shocks and frictions in US business cycles: A bayesian DSGE approach. *The American Economic Review* 97(3), 586–606.
- Taylor, J.B., 1993. Discretion versus policy rules in practice. *Carnegie-Rochester Conference Series on Public Policy* 39, 195–214.