

GGIG Graphical Interface Generator

User Guide

Wolfgang Britz, August 2010

- Version October 2014 -

The following user guide documents the outcome of a collaborative effort of University Bonn and the author. Larger parts of the Java code underlying GGIG had been developed over the years in the context of projects related to the [CAPRI](#) modelling system, which received considerably funds from the EU research framework programs. Following the general policy in CAPRI, the GGIG pre-compiled code can be used for other scientific projects as well without charge. The document comprises to a larger extent the content of earlier versions of the CAPRI user guide of which the GUI is now realized in GGIG.

The author would like to acknowledge the contribution of Alexander Gocht, vTI Braunschweig, to the CAPRI GUI coding efforts. All errors remain with the author.

Content

GGIG Graphical Interface Generator	1
User Guide.....	1
Content	3
Overview	8
An overview on the GUI.....	10
Initialization: General interface settings	11
GAMS and R related settings.....	11
SVN related settings.....	12
Case one: Exploiter and runner	12
Usage for installation purposes	15
Settings linked to the exploitation tools	17
Starting GAMS from GGIG.....	18
Viewing results: exploitation tools.....	19
Views as the basic concept for exploitation	19
Exploiting results.....	20
Selecting scenarios	21
The multi-dimensional viewer with pivoting and exporting possibilities.....	22
Pre-defined views	23
View selection.....	23
Navigating through views	24
Navigating in the outer dimensions of the viewport	24
Column and row selection.....	24
Predefined selection groups	26
Selection of the view type	26
Manually changing the pivot.....	26

Changing view options: fonts, number formatting and rounding, hiding empty cells, comparisons.....	28
Showing a histogram window	30
Working with tables	31
The toolbar	31
Tooltips for column and row headers.....	32
Drill-down	32
Clipboard export.....	32
Export to file.....	32
Sorting	33
Numerical filtering based on cell content	33
Changing the row height and column width with the mouse.....	33
Adding statistics	34
Outlier detection algorithms implemented	36
Working with graphics implemented	38
General handling of graphs	38
“Walking” through the data.....	41
Exporting the graphic to file.....	41
Exporting the graphic to clipboard.....	42
Bar charts.....	42
Line and point charts	43
Pie charts	45
Spider plots.....	46
Box and Whisker charts	47
Histograms	48
Deviation renderer.....	50
How to draw a line chart with mean / min / max etc. over a time series	51

Markov charts.....	53
Flow maps	54
Pie chart maps	56
Colored thematic maps.....	56
Changing the classification and the legend	57
Adding a histogram window to a map	58
Shrinking polygons according to UAA share	59
Area weighted classification	60
Excluding zeros from classification and removing small and large values	60
Classification method.....	60
Integration distribution information in the map window	63
Color table.....	63
Changing the way the legend is drawn.....	67
Copying the map to the clipboard or saving to disk.....	69
Changing the title of the map	69
Zooming in and out and navigating in the map.....	69
Getting data for specific polygons	70
Highlighting specific regions in the map	72
Updating the map	75
Adding region label to the map	75
Showing river and cities	76
Storing and re-loading your settings	77
Exporting the data underlying the map	78
Machine learning.....	89
Implementation in GGIG	91
Interaction between the GGIG GUI and WEKA	93

The WEKA GUI.....	95
Classification.....	95
Filtering.....	96
Attribute viewing and selection.....	97
Summary.....	98
References.....	98
Scenario editor.....	99
Meta data handling.....	100
Why meta data?.....	100
Technical concept.....	101
File menu.....	103
Settings menu.....	103
Utilities and GUI menu.....	103
Utilities: Batch execution.....	103
Format of the batch execution steering file.....	104
Header.....	104
Settings for tasks.....	105
Using the batch execution facility.....	107
The output from batch execution.....	108
Utilities: Generate GAMS documentation in HTML pages.....	110
Structure of the HTML pages.....	110
Tagged in-line comments.....	111
Refactoring Consequences for Gams Code.....	112
General overview.....	114
Example for a Symbol page.....	114
Example for a GamsSourceFile page.....	115

Example for a page for the a set	115
File list.....	116
Set element list	116
Utilities: Equation and variable viewer	117
Background and motivation	117
An overview on the viewer	117
Producing input for the view with GAMS	118
Includes	120
Loading symbols	121
Working with the equation and variable viewer	122
Utilities: Gdx-file(s) viewer	123
Utilities: Generating coordinate files for the exploitations tools from shapefiles.....	125
Analysis differences in GAMS based data using GGIG	126
Background	126
Comparing two data sets in GGIG, example from CAPRI	127
GGIG as GDXDIFF	128
Using the table definitions.....	130
Comparing two GDX files with GGIG	131
Index.....	131

Overview

The GAMS Graphical Interface Generator (GGIG) is a tool to generate a basic Graphical User Interface (GUI) for a GAMS or R project¹ with five main functionalities:

1. **Generation of user operable graphical controls from XML based definitions.** The XML file defines the project specific layout of the GUI. The user can then interact with the GUI to change the state of the controls. The state of each control component such as a checkbox can then be mapped to GAMS code (`$SETGLOBALS`, Set definitions, settings for parameters). It combines hence the basic functionality of a GUI generator and a rudimentary GAMS code generator.
2. **Generation of GAMS compatible meta data** from the state of the control which can be stored in GAMS GDX format and later accessed, so that scenario definitions are automatically stored along with results.
3. **Execution of a GAMS or R project while passing the state of the control to GAMS respectively R** as a include file.
4. **Exploitation of results from GAMS runs** by providing an interface to define the necessary interfacing definitions in text file to load results from a GAMS into the CAPRI exploitation tools.
5. **Access to a set of GAMS related utilities.** This include e.g. a viewer for GDX files, a utility to build a HTML based documentation of the GAMS code or a batch execution utility.

That guide is thought for users of GGIG generated interfaces. It will be typically be complemented with a user guide which is specific for the project, such as the CAPRI user interface documentation. The “GGIG programming guide” comprises the necessary information to set up interfaces based on GGIG.

The main parts of GGIG are graphically depicted below. At its core stands the GGIG Control generator, based on Java code. Based on a XML based definition file provided by the project, it generates a project specific GUI which can be operated by the user. The state of these controls such as numerical settings, on/off settings or n of m selection can be passed to

¹ The code can also be used from inside Java, but that feature is not discussed in the documentation.

GAMS by an automatically generated include file which also contains generated meta data documenting the state of the controls. The user can also execute GAMS from the GUI. The GUI can equally load numerical results and meta data in a specific GDX viewer. The latter supports “view definition”, i.e. pre-defined reports to exploit the results. The details of the different elements are discussed below.

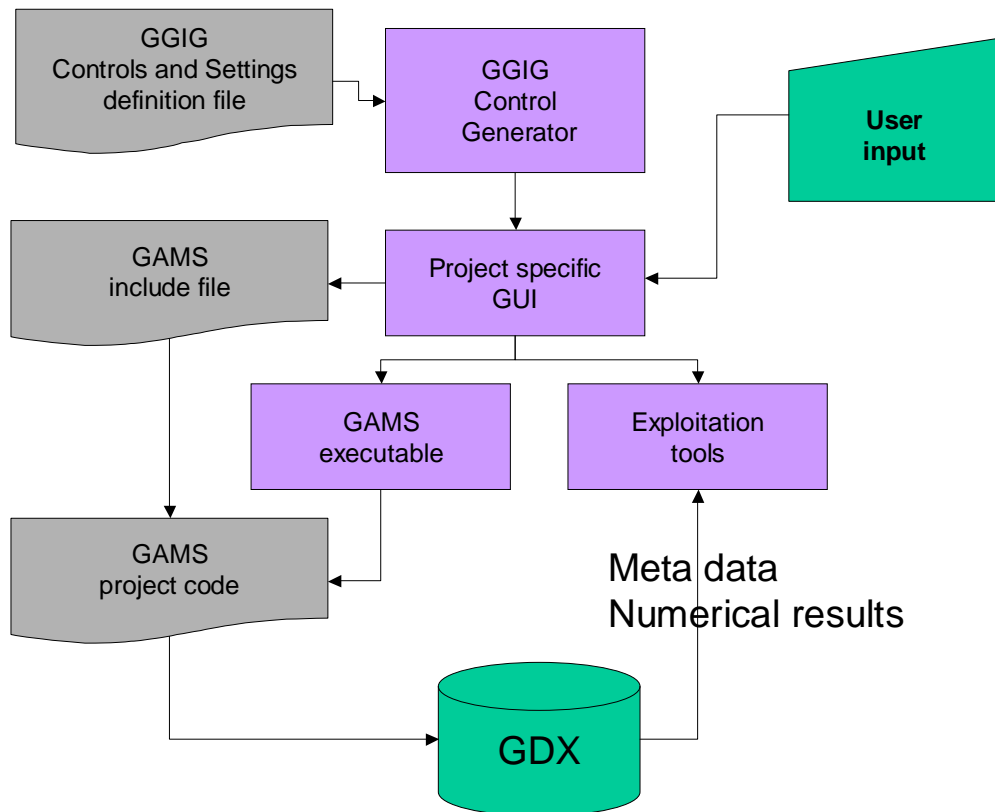
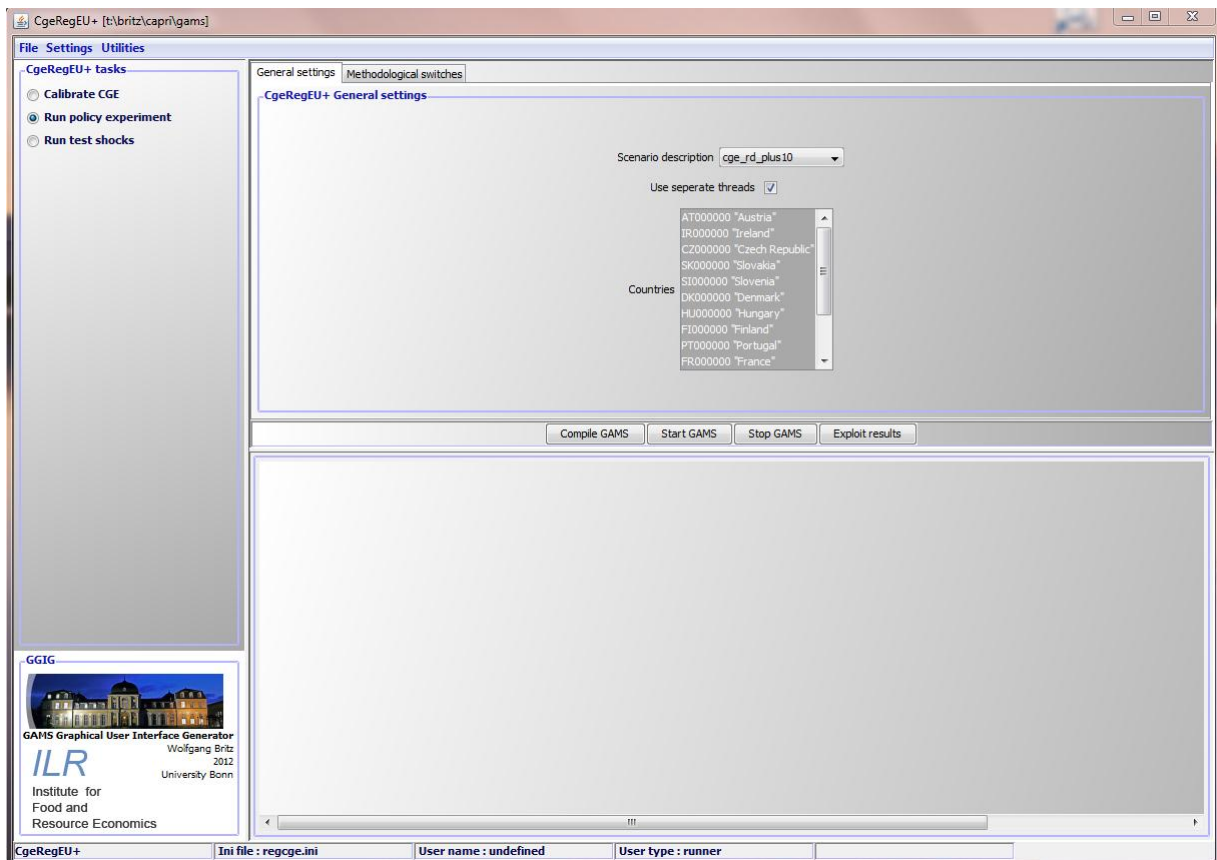


Diagram: Overview on information flow in GGIG

An overview on the GUI



As shown in the example above, the GUI consists a few elements:

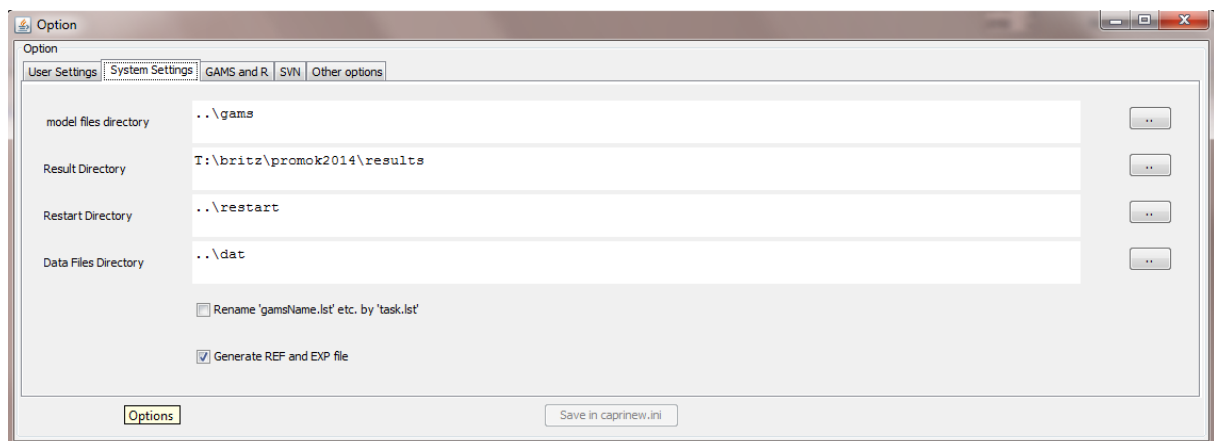
1. A **menu bar** which allows to change some settings (see the section on general interface settings)
2. A **workstep and task selection panel** on the left hand side where the user can select between different tasks belonging to the project.
3. A **right hand side panel** which either shows:
 - i. **The generated controls**, a button panel to start GAMS and a windows in which the message log from GAMS is shown
 - ii. **A panel to select data to view** and to start their exploitation
 - iii. **The exploitation tools**
4. A small window in the left lower corner which present a logo.

Before using a GGIG based interface, the users need to edit some project specific settings, see next chapter.

Initialization: General interface settings

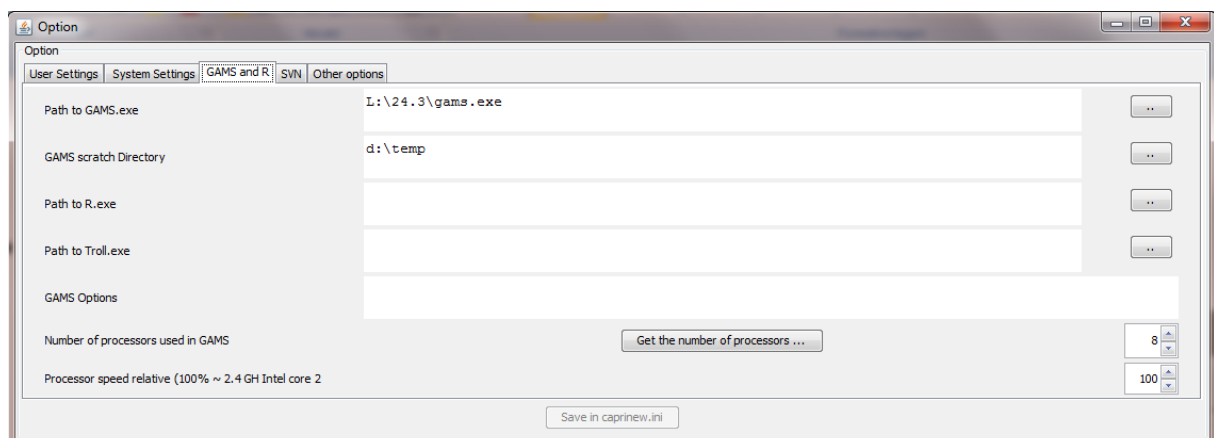
The interface has a few standard settings which can also be accessed via the “edit settings dialogue”. These are:

- Certain file locations:** the directory where GDX files for results are assumed to be stored (resDir) , and three directories which can be used to adjust the specific model application: the root of the GAMS file (workDir in GAMS), called modelDir, a directory for restart files and one for data files.

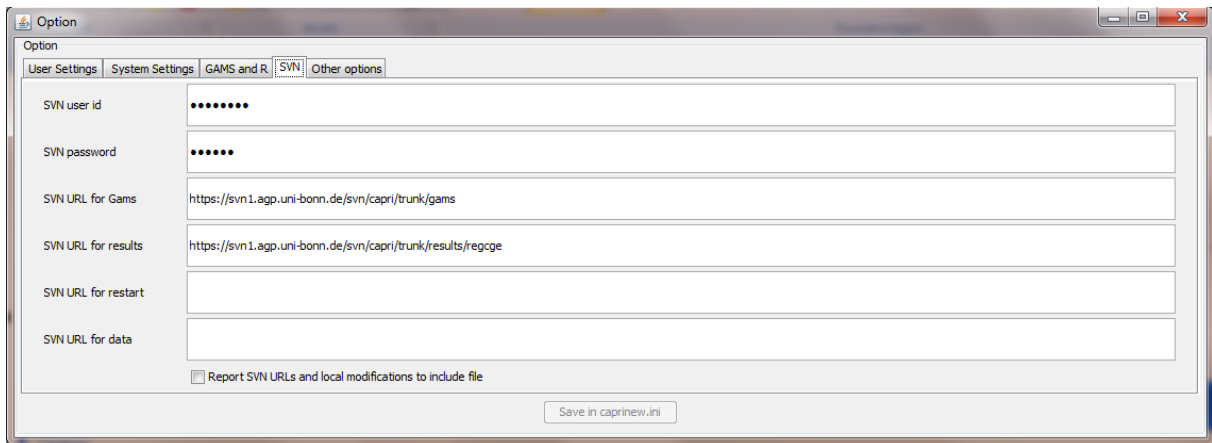


These file locations are passed to GAMS and can be used in the GAMS code to read / include files from the correct locations on disk. In order to make an initialization file portable, locations can be defined relative to the GUI directory.

GAMS and R related settings



SVN related settings



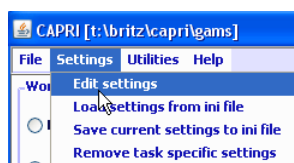
The SVN settings can be used to perform checkout and updates in cases where the model code with related data, restart files or result files is under versioning control on a SVN server. If the model is not under version control, the settings “svn=no” renders the tabbed plan invisible. The SVN settings are thus only optional.

Case one: Exploiter and runner

Entering the necessary information to link to the SVN server

An exploiter by definition only accesses GDY files from the result directory. He is not allowed to run GAMS programs, and thus does not need access to the GAMS source code, data and restart files read in by the different GAMS based working steps of CAPRI.

Accordingly, in order to work with SVN, only three pieces of information have to be entered



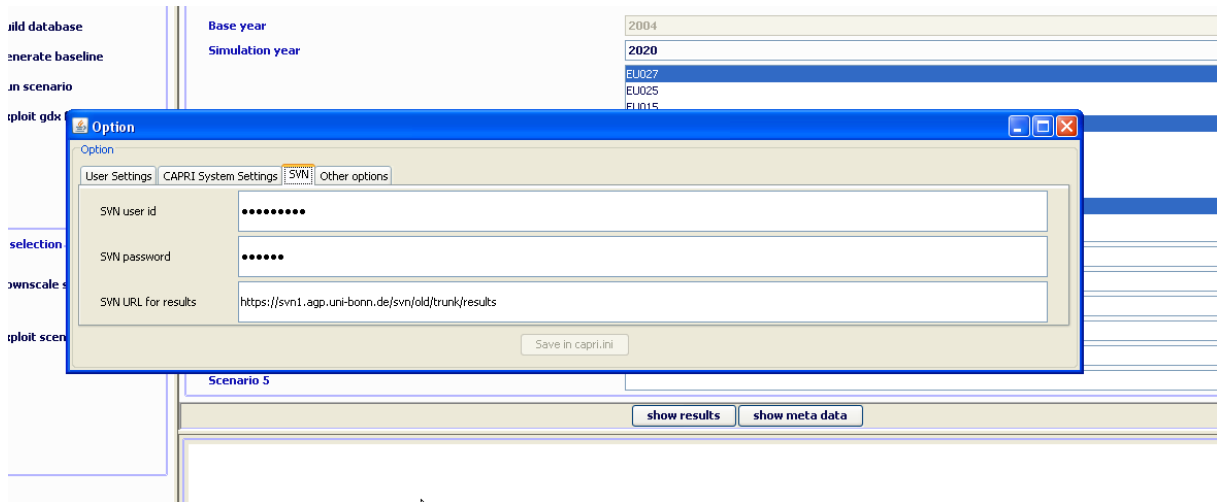
under “Settings / Edit Settings” in the SVN tab:

- The SVN user id
- The SVN password
- The url of the result directory

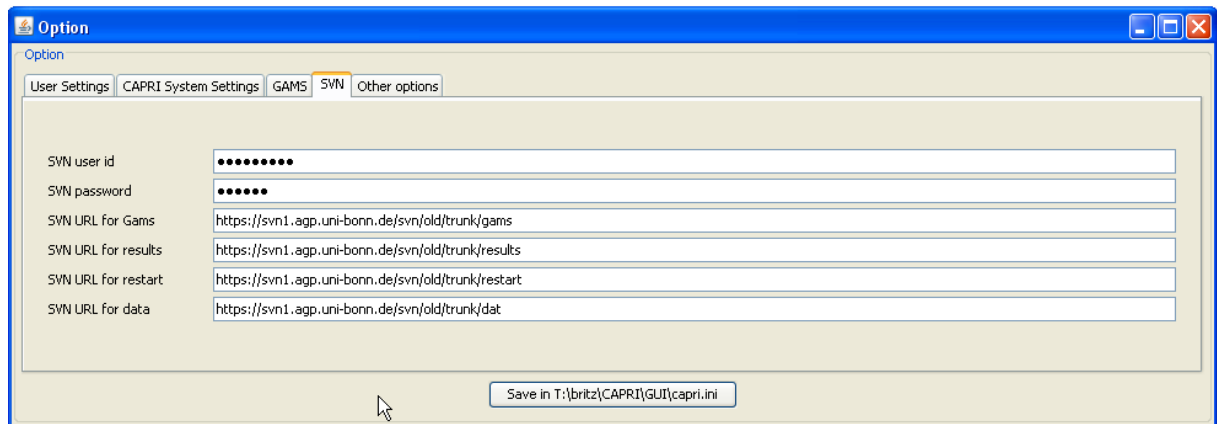
The first two fields are not visible, and the related entries in the ini file are encrypted. The last entry can be set to a specific branch relating e.g. to a training session. That allows for CAPRI “mini installations”. These mini installations do not need to be distributed as SVN installations as the SVN interface in the GUI will also allow to “checkout” over existing sub-directories and files. That ensures some additional safety regarding access information to

sensible branches of the server – a bystander cannot read the user id and password. But users should always place local copies of such branches including the directory from which the GUI is started on secured parts of their file system.

The local directory for the GUI is simply taken from the start directory of the GUI; whereas the SVN address for the GUI is stored in the “default.ini” file.



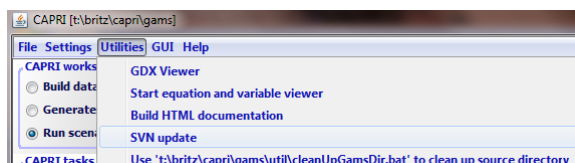
The runner can enter the additional SVN urls relating to the different sub-directories of a CAPRI installation. That should give some flexibility when working with branches on the server:



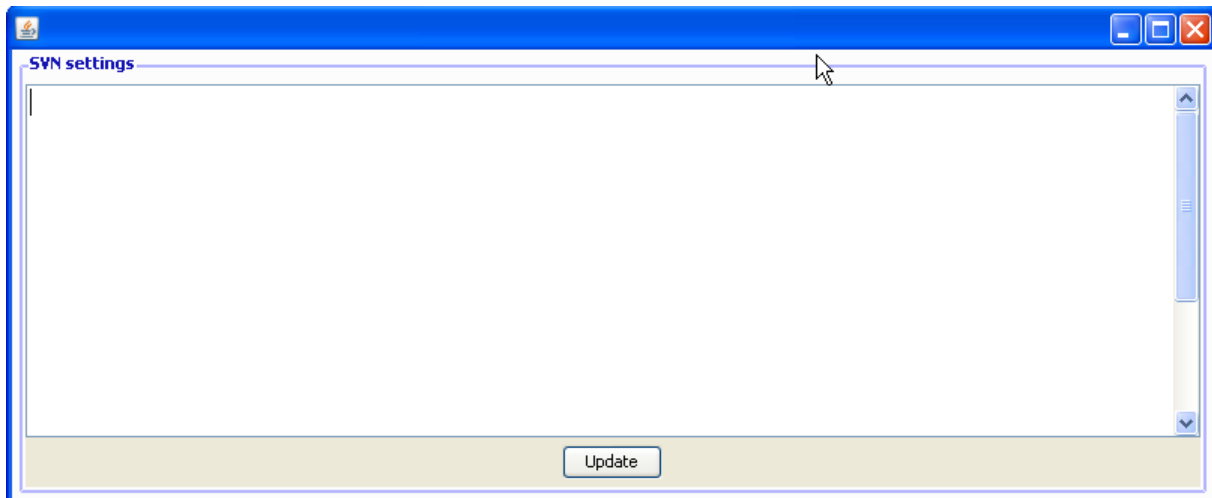
Performing an update

The second functionality for an exploiter (and runner) is to update all directories with the

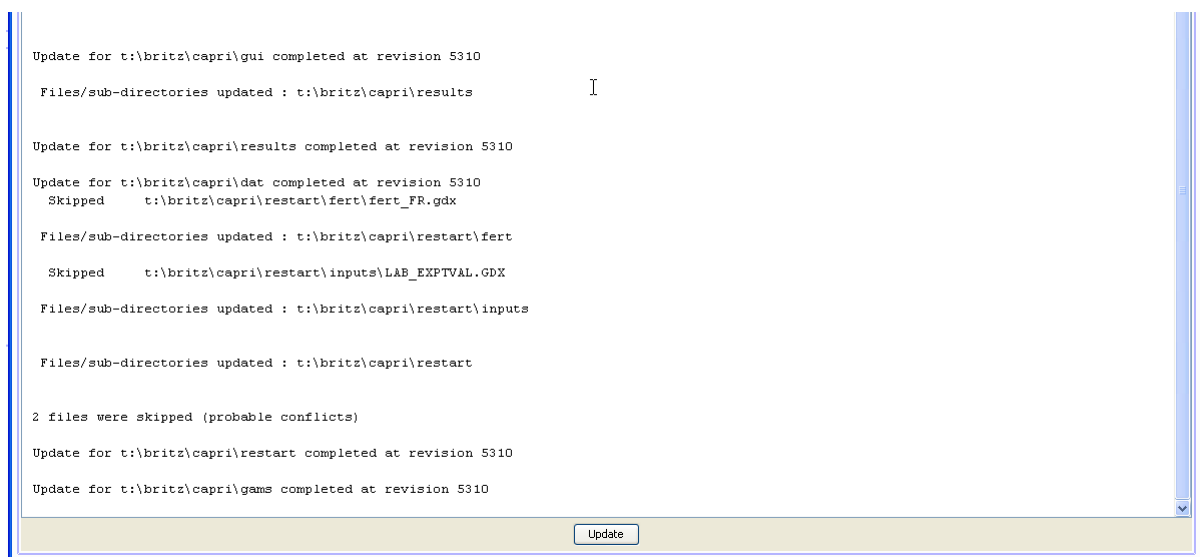
menu item “Utilities / SVN update”. An update will download updated versions of files into hidden directories, and, if the related files in the local working copy have not been modified, will also replace the local files.



Choosing that menu item will open a dialogue with just one button termed “update” and an area into which messages from the SVN updates / checkouts are reported:



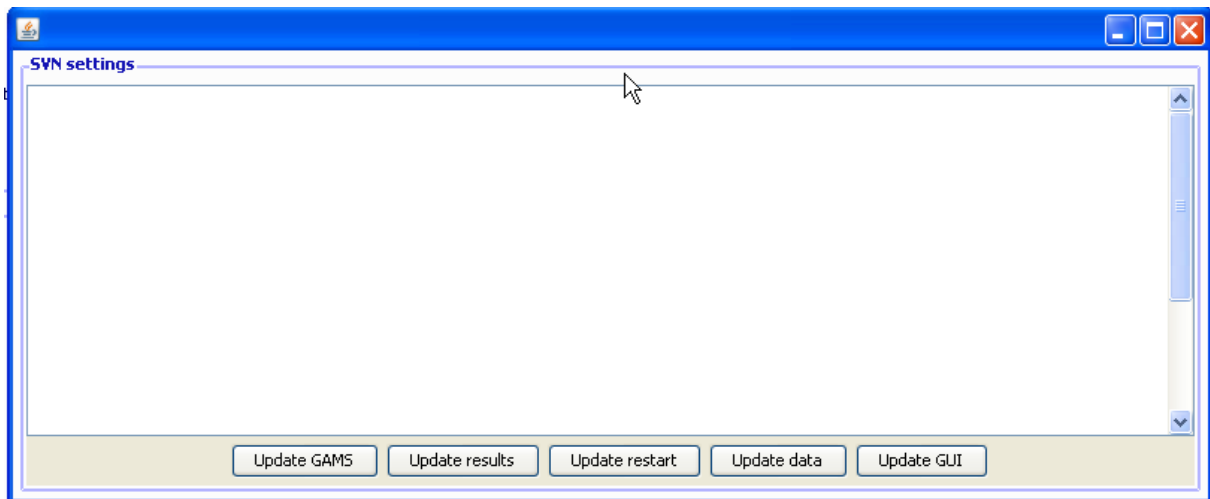
Pressing the “update” button will trigger an update. Possible conflicts, merges etc. are shown in the reporting area:



If the directory is not yet under version control, the GUI will perform a checkout instead, i.e. setting up the first installation of the hidden copies from the server. Before an update, a “clean-up” operation will remove any possible local locks related to earlier unsuccessful SVN operations. As long as an internet connection is available, that should ensure smooth updates in most cases and avoid some of the more tricky problems TortoiseSVN users might face.

Case two: Administrator

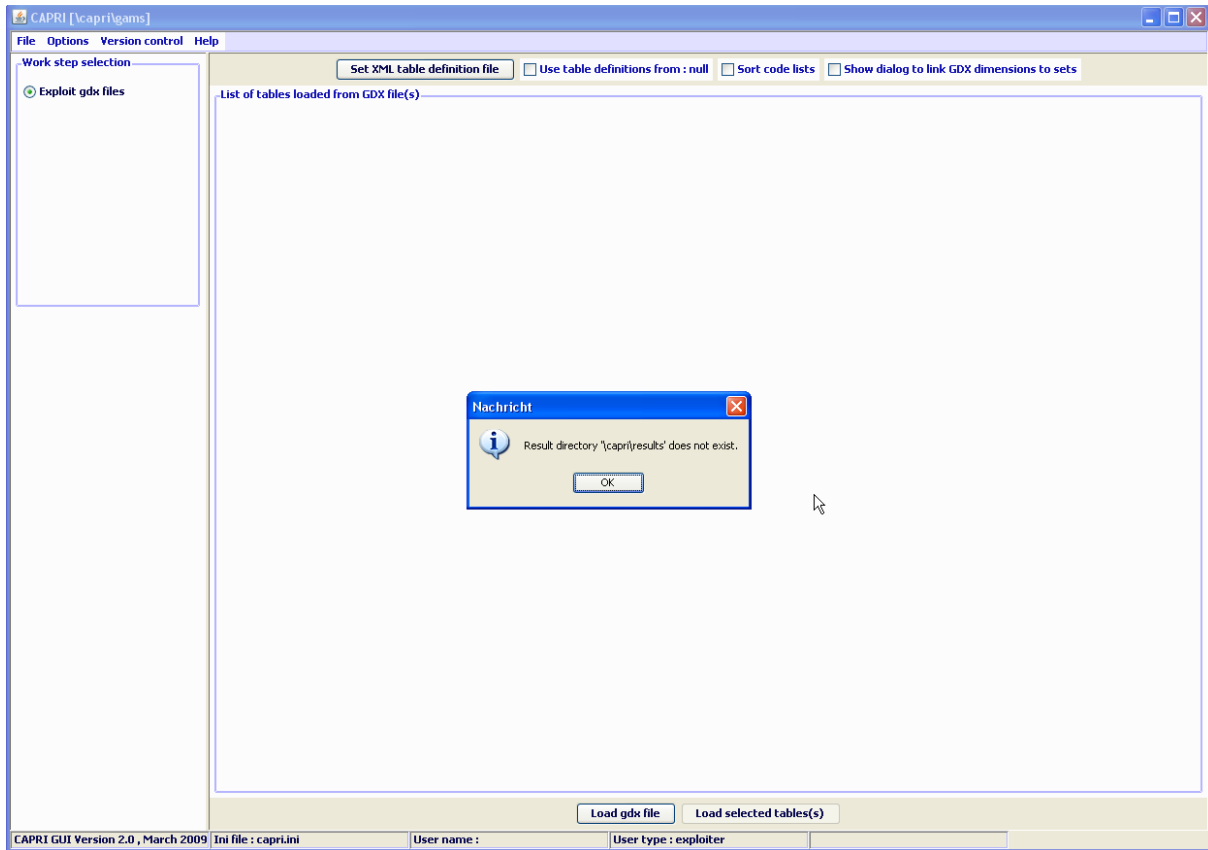
An administrator can enter the same SVN directories as a runner, but can trigger updates for the different parts separately:



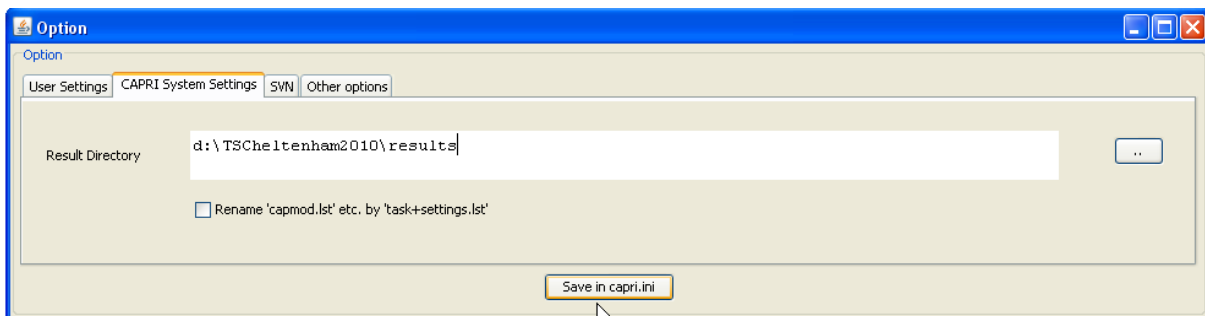
Usage for installation purposes

Since quite a while the CAPRI network discusses how installations specifically for training sessions can be organized more easily. The newly embedded SVN functionalities in the GUI should ease that task somewhat, specifically in cases where only exploitation functionalities are asked for.

The installation of CAPRI based on the new functionality is relatively straightforward. As before, a JAVA run time engine must be installed for the GUI to run. For an exploiter, only a minimum GUI installation (e.g. without the large geometries for the 1x 1 km layer) and the necessary results files to view can then be copied to a local directory. At first start, the user must then only enter where the results had been copied to (if the result files are not parallel to the GUI) and save the information to his new *CAPRI.INI* file.

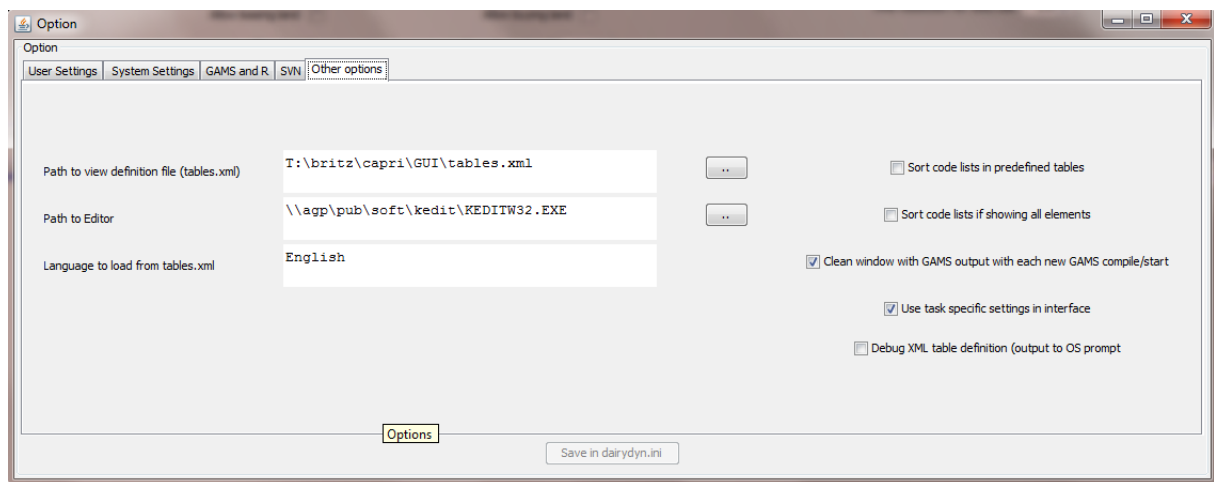


The interface is set-up such that only the results of those work steps are visible where result files are found. For a training session concentrating on analysing scenarios, only those result files can be distributed. An installation with four scenarios at NUTS2 level plus all the necessary GUI files will require under 100 MByte disk space.



Once the user has optionally entered the results directory, and stored it to the ini file, the user will face a rather clean interface which only allows to exploit existing scenarios and to exploit GDx files (also that option could be removed for exploiters).

Settings linked to the exploitation tools



The “Path to the view definition (tables.xml)” allows to load a XML which defines views into the results (see chapter “exploit results”).

Starting GAMS from GGIG

GGIG allows starting the GAMS project directly from the interface, either in compile or run mode. A break request can also be sent to GAMS (“stop GAMS”):

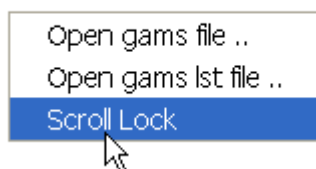


Once started, the GAMS project routes its output instead to the console (the DOS prompt) to the lower right part of the interface, such that the user can follow the progress:

```

--- .farm_constructor.gms(91) 3 Mb
--- exp_starter.gms(74) 3 Mb
--- .ini_herds.gms(19) 3 Mb
--- ..title.gms(30) 3 Mb
--- .ini_herds.gms(86) 3 Mb
--- exp_starter.gms(78) 3 Mb
--- .decl.gms(29) 3 Mb
--- exp_starter.gms(195) 3 Mb
--- .title.gms(30) 3 Mb
--- exp_starter.gms(202) 3 Mb
--- .title.gms(30) 3 Mb
--- exp_starter.gms(240) 3 Mb
--- .store_res.gms(232) 3 Mb
--- exp_starter.gms(323) 3 Mb
--- .title.gms(30) 3 Mb
--- exp_starter.gms(325) 3 Mb
--- .store_res.gms(232) 3 Mb
--- exp_starter.gms(344) 3 Mb
*** Status: Normal completion
--- Job exp_starter.gms Stop 12/01/10 21:06:06 elapsed 0:00:00.047
GAMS RC 0
  
```

The pane with the content can be scrolled by a right mouse click in the pane to open a popup menu. If an editor is added under “other options”, the GAMS and the listing file can be opened as well:



The pane can hence be “frozen” so that e.g. the status of a model solve can be inspected while the project continues to run. In order to successfully start a project, the ini file for GGIG must comprise the information where the GAMS executable can be found, but also where the GAMS code of the project to start is stored, see the discussion on settings above.

Viewing results: exploitation tools

The basic strategy of the GGIG exploitation tools roots in the [CAPRI exploitation tools](#), which require that all model results are stored into one GAMS parameter which can have up to 10 dimensions and saved to GDX container as a sparse matrix on disk. One or several GDX containers with results are then read from disk and merged.

An additional dimension can be added if several files are loaded, e.g. to compare scenarios or years. A specific XML dialect defines views (filters, pivots, view types) into the cube, and allows the user to load several result sets – typically from different scenarios – in parallel.

If no table definition file is present, GIGG offers a GDX viewer which some interesting possibilities not found in the standard GDX viewer (such as numerical sorting, statistics, selections). For details, see below.

Views as the basic concept for exploitation

The concept of the GGIG exploitation tools is centred on the idea of a view. Content wise, each view may be understood as showing one or several indicators relating to results of working steps defined in GGIG, e.g. environmental effects of farming, prices or market balances. Each view thus

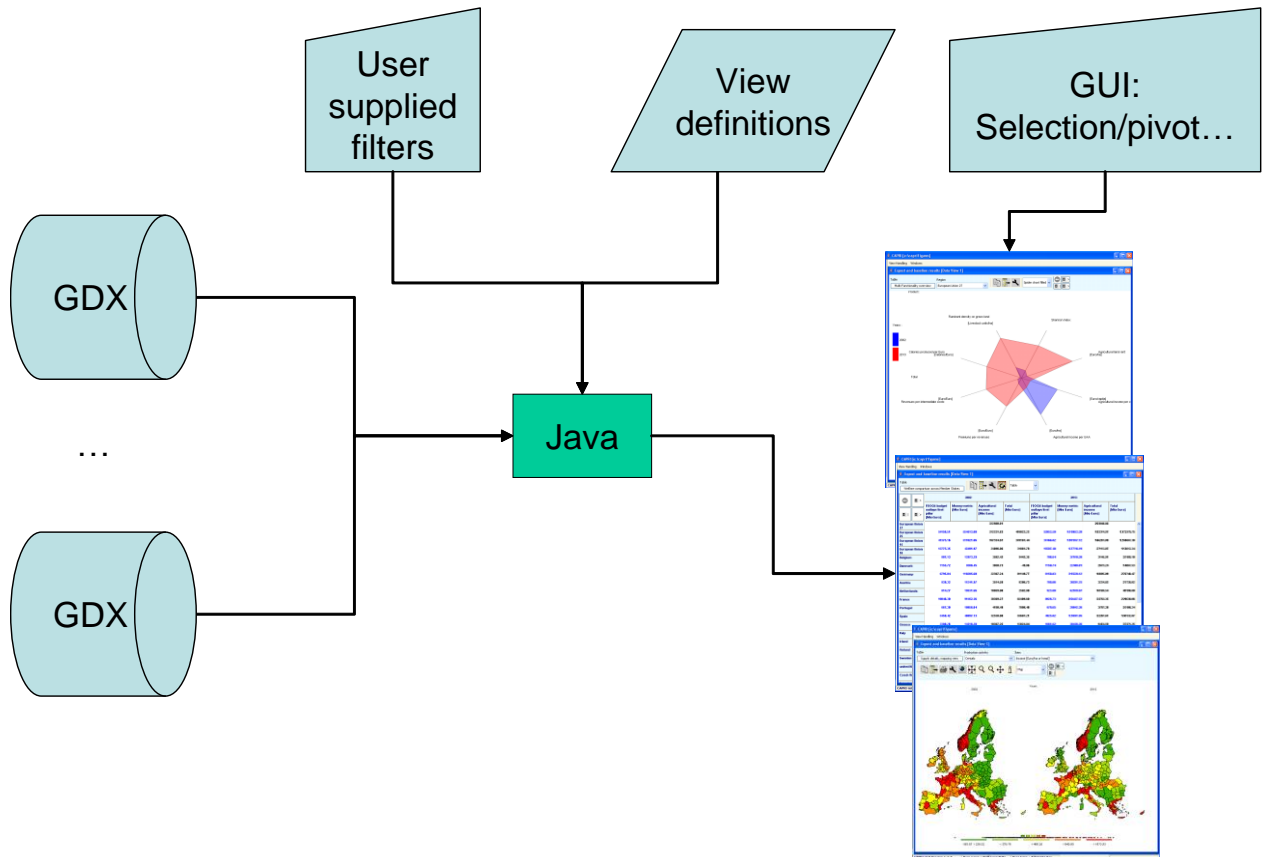
- extracts a certain collection of numerical values (filtering)
- labels them so that they carry information to the user (long texts, units)
- chooses a matching presentation – as a table, map or graphic
- and arranges them in a suitable way on screen.

The views can be linked to each others, allowing a WEB like navigation through the data cube. Views can be grouped to themes. The user may open several views in parallel, and she may change the views interactively according to her needs, e.g. switch from a map to a tabular presentation, or change the pivot of the table, sort the rows, add statistics, introduce comparisons etc.

Internally, each view is stored in a XML schema. Technically, a view can be understood as a combination of a pre-defined selection query, along with reporting information. The XML schema allows to attach long texts, units and tooltips to the items of a table, and thus to show meta-data information to the user. The XML schema does hence replace look up tables in a DBMS. It may equally store information regarding the pivoting, the view type (table, map,

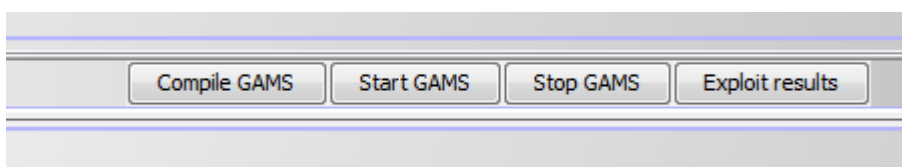
different graphic types), and for maps, classification, colour ramp and number of classes. The views can be grouped into logical entities, and are shown as a popup menu to the user.

Tabular views may feature column and row groups. Empty columns and rows can be hidden; tables can be sorted by column, with multiple sort columns supported. Numerical filter can be applied to columns.



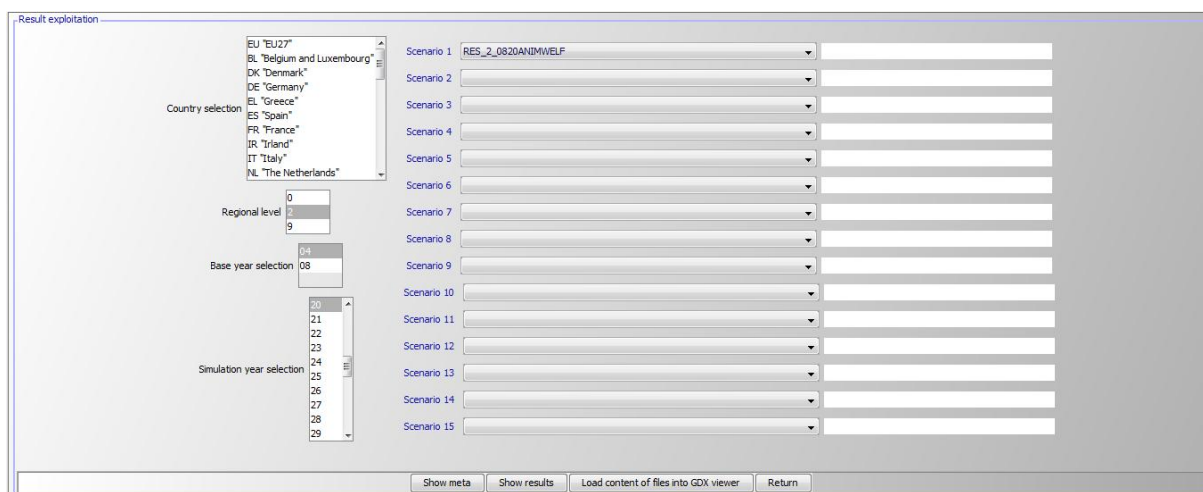
Exploiting results

For each work step, pressing the “Exploit results” button:



Which will load the exploit result exploitation panel.

Graph: The interface in "exploitation" mode

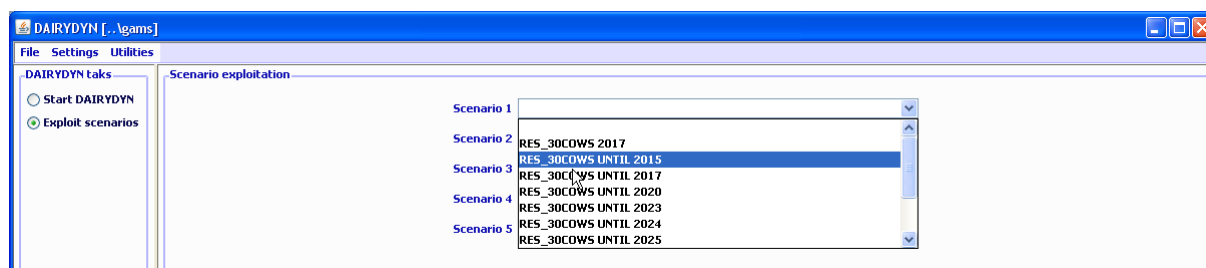


The right hand side comprises a set of drop-down boxes from which up to 15 different scenarios (or result files) can be selected. The first element in each box is empty.

The left hand side shows, depending on the work step, selection control for countries, the regional level, base year and simulation year. Operating these control filters out files from the disk shown in the drop-down boxes. In the example above, only results files for the base "04", simulation year "20" and the regional level 2 (= NUTS2) can be selected.

Selecting scenarios

When the user clicks on "Exploit scenarios" in the task selection panel, the drop boxes are shown on the right hand side. Each box comprises the list of GDx files found in the result directory registered for that task. The user can select in each box a file, or leave it empty. The text fields next to the boxes allow the user to replace the file name normally used as a description of the scenario by a user chosen text.



At the bottom of the panel, pressing the "show results" button will open the exploitation tools:

DAIRYDYN [..gams]

View Handling Windows

Herd summary, mean [0]

Year: mean Farm: RES_30COWS 2017

View type: Table

Cows

	Herd size [Heads]	Revenues [Euro/ha]	Variable costs (incl. concentrates) [Euro/ha]	Costs of concentrates [Euro/ha]	Gross margin [Euro/ha]	Labour [hours/ha]	Milk yield [liter/head and year]	Nu lac [lit an
actBased	red0	25.71	2980.48	451.46	201.29	2529.02	22.00	7.14
	red1	25.68	2973.75	449.12	199.42	2524.63	22.00	7.13
	red2	25.01	2971.81	448.56	199.08	2523.25	22.00	7.13
	red3	24.24	2975.55	449.85	200.11	2525.70	22.00	7.14
	red4	23.44	2980.59	451.59	201.50	2528.99	22.00	7.15
	red5	22.67	2985.35	453.17	202.68	2532.18	22.00	7.16
	red6	22.01	2986.51	453.38	202.64	2533.13	22.00	7.16
	red7	21.35	2987.75	453.57	202.55	2534.17	22.00	7.15
	red8	20.70	2989.09	453.73	202.35	2535.36	22.00	7.15
	red9	20.04	2990.52	453.89	202.15	2536.62	22.00	7.15
prodBas ed	red10	19.33	2992.08	454.18	202.11	2537.90	22.00	7.15
	red0	25.71	2980.48	451.46	201.29	2529.02	22.00	7.14
	red1	25.41	2940.22	437.50	190.10	2502.72	22.00	7.05
	red2	24.75	2938.59	437.07	189.89	2501.52	22.00	7.05
	red3	24.10	2937.43	436.64	189.51	2500.79	22.00	7.04
	red4	23.43	2938.74	437.02	189.74	2501.71	22.00	7.05
	red5	22.80	2937.10	436.33	189.04	2500.77	22.00	7.04
red6	22.13	2939.59	437.03	189.44	2502.56	22.00	7.04	

The full functionality is only available if a table definition file (see programmer guide) matching the structure of the parameters in the GDX file is provided.

The multi-dimensional viewer with pivoting and exporting possibilities

The results are – as mentioned above – loaded from a GDX container and comprise a GAMS a parameter with up to 10 dimensions. That cube of data is loaded in a spreadsheet like viewer with pivot-possibilities.² The user may switch between a tabular view of the data, or different types of graphs (line, bar, pie, spider) or maps. Scroll-down boxes allow the user to rotate through data dimension not shown in the view port columns and rows. Several data dimensions may be merged into one view port dimension. The user can use column and rows

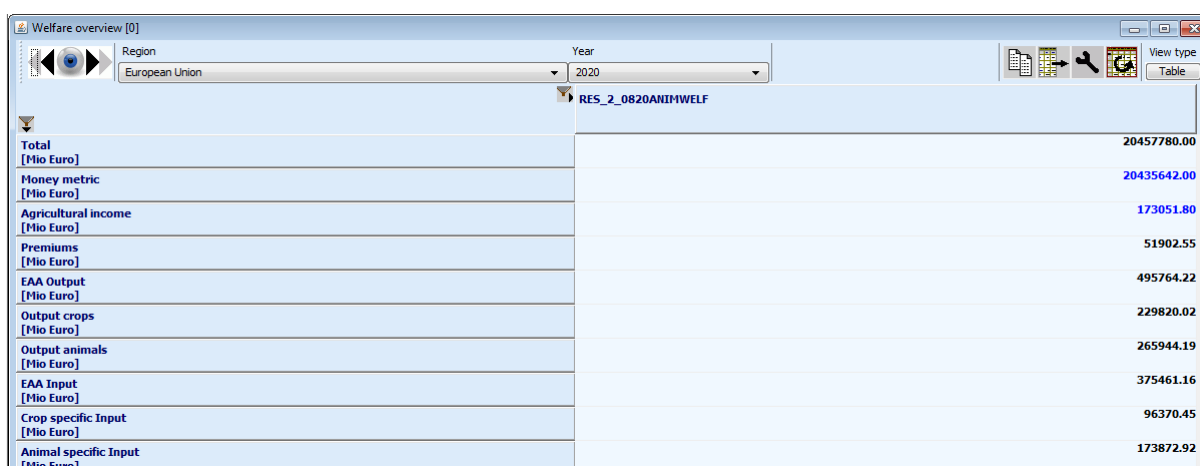
² Both the GDX container and the interface work in “sparse mode”, i.e. only non-zero values require disk or memory space. Introducing additional dimension has therefore limited impact on space requirements.

groups, and may apply selection to columns and rows as well as to columns and column groups. Rows carrying zero values may only be hidden. Rows may be sorted by size of the numerical values in one or several columns. The current table may be loaded into the clipboard. Alternatively, all or a selection of tables may be exported to an external file, in different formats (HTML, CSV, tab-separated, GAMS, fixed width tables). There are further possibilities available such as changing fonts or the number of decimals. The following chapters give details on these possibilities.

Pre-defined views

An XML file links pre-defined views to the result content of the tasks defined in GGIG. Each view defines selections in the different data dimensions, the view type (table, graph or map) and the pivot, plus some other information.

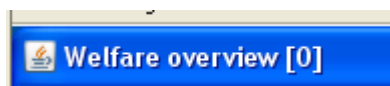
Graph: A pre-defined view




RES_2_0820ANIMWELF	
Total [Mio Euro]	20457780.00
Money metric [Mio Euro]	20435642.00
Agricultural income [Mio Euro]	173051.80
Premiums [Mio Euro]	51902.55
EAA Output [Mio Euro]	495764.22
Output crops [Mio Euro]	229820.02
Output animals [Mio Euro]	265944.19
EAA Input [Mio Euro]	375461.16
Crop specific Input [Mio Euro]	96370.45
Animal specific Input [Mio Euro]	173872.92

View selection

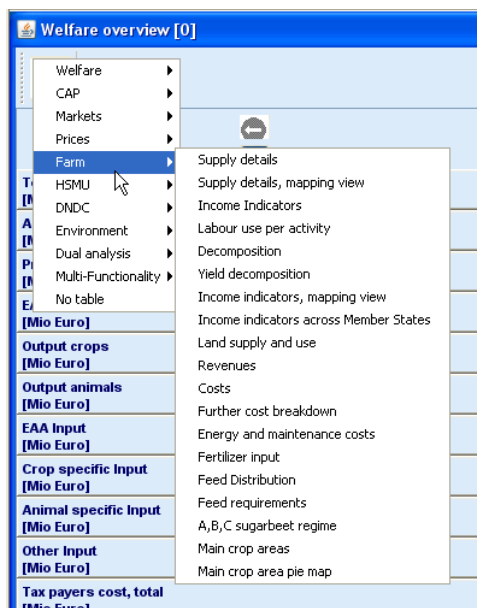
The *currently selected view* is shown as a description of the window title:



The number behind gives the internal order of the views as several views can be open in parallel.


The currently shown view in such as window can be changed by pressing the view  button. Pressing the button opens a pop-up menu to *select another view*. The available views will depend on the results you have loaded. The views are logically grouped under headings,

and moving the cursor on the heading will show the single views grouped under that heading. Some views will be opened as graphics (see chapter) or maps (see chapter).



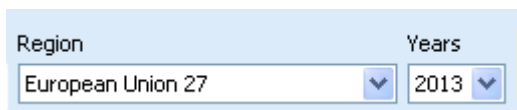
Navigating through views



The dark triangle to left and right of the view  button allow navigating through to the list of available views. The outer triangles in grey allow navigating through the previously visited views.

Navigating in the outer dimensions of the viewport

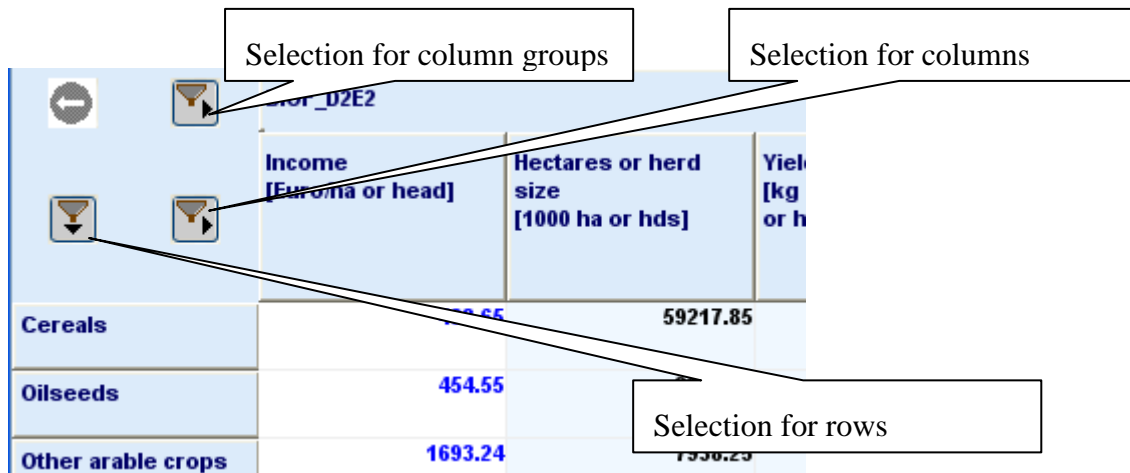
In many views, some data dimensions will not be shown in the columns and rows, but as drop-down boxes in the toolbar. Use the mouse to select within the boxes. You can also use the keyboard to search items by typing. An example for these controls is shown here.



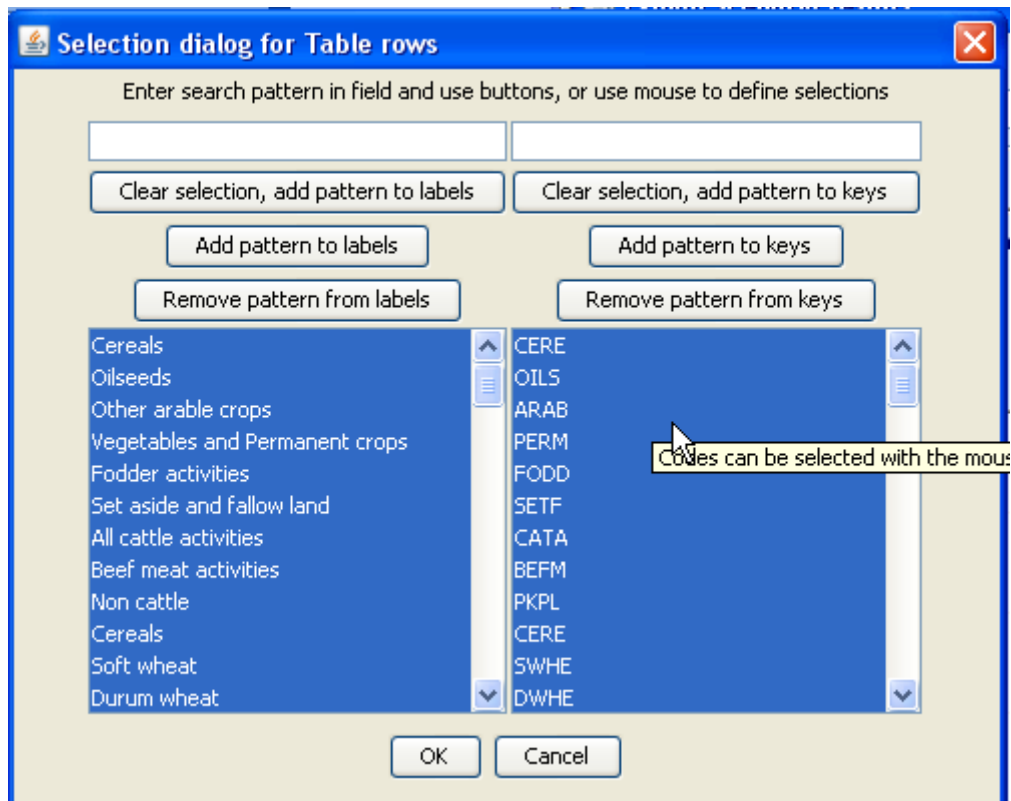
Note: If an outer dimension does only comprise one element, no drop-box list is shown

Column and row selection

Columns and rows can be hidden and included in the current view by using the buttons shown below.



Double-clicking the button will open a selection dialogue:




The selections can be done by mouse, following the convention of the operation systems.


Additionally, a selection string can be entered in the field above, with the following possibilities:

- "*" select all
- "C*" select all items starting with "C", "C???" will select a string starting with C followed by any 3 characters.

After entering the selection string in the text field, one of the three buttons must be right-

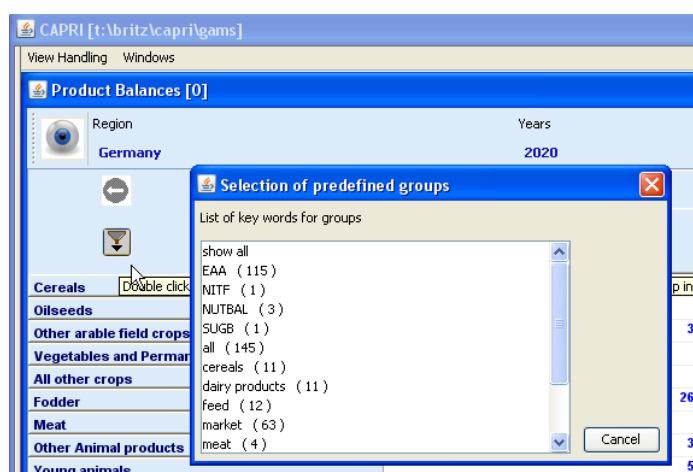
clicked. The  button will remove any selection and select

only those items which match the pattern entered in the text field.  will

keep the selection and add the matching items, whereas  will remove matching items from the selection.

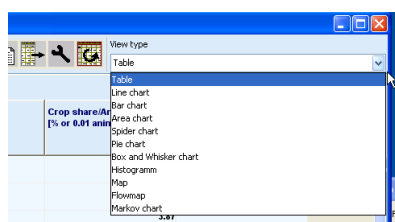
Predefined selection groups

For some tables, pre-defined selection groups for columns or rows are stored. When the mouse is moved over the selection button and rests there for some time and such groups exists, a dialogue will show as below where the groups can be selected.




Selection of the view type

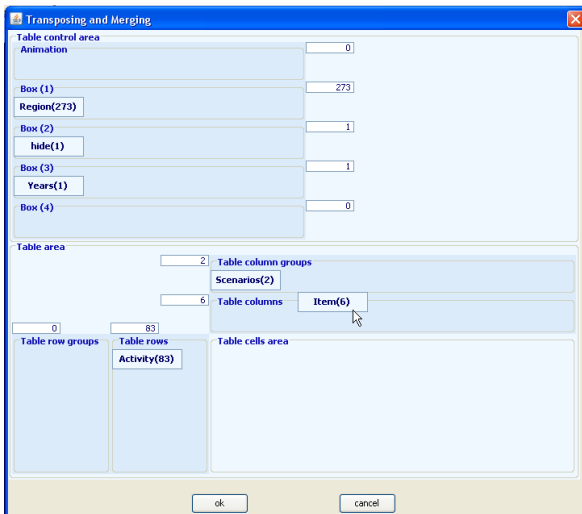
As discussed below, the data can be shown as tables, graphics or maps, to do so use the view selection drop-down box:





Manually changing the pivot

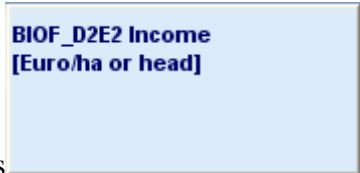
Normally, the predefined views will link the data dimension in an appropriate way to columns and rows. However, the user is free to change the pivot, to e.g. generate a cross-

sectional series. A dialog opens when left-clicking the  button to pivot the currently shown or selected part of the view:



The boxes  show the data dimension and their lengths. They can be dragged to the different viewport dimensions as shown in the screen shot above. Assigning several

dimensions to the columns  leads to “spanned”

dimensions . Alternatively, columns and rows can have row block:

	Cereals	Oilseeds	Other
BIOF_D2E2			
Income [Euro/ha or head]	438.65 -17.73%	454.55 -30.01%	
Hectares or herd size [1000 ha or hds]	59217.85 -2.88%	8335.37 -13.24%	
Yield [kg or 1/1000 head/ha or head]	4858.89 -3.58%	2271.59 -8.17%	
BIOF_D1HE1			
Income [Euro/ha or head]	533.21 0.00%	649.48 0.00%	
Hectares or herd size [1000 ha or hds]	60473.37 0.00%	9607.46 0.00%	
Yield [kg or 1/1000 head/ha or head]	5142.97 0.00%	2421.83 0.00%	

In combination with the selections for columns and rows, and column and row blocks, the view can be adjusted to the need of the user, e.g. to export the data in a specific ordering to an external file.


The pivot can alternatively be changed by mouse clicks in the text field above a selection box:

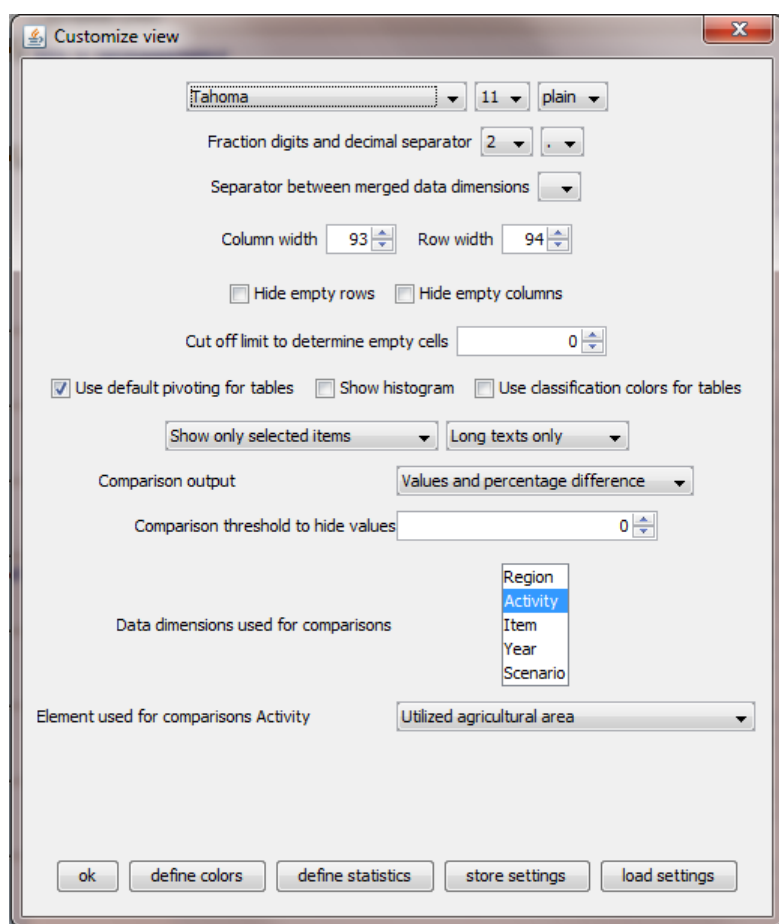
- A left mouse click puts the items from the selection box into the rows, while moving the items from the rows into a selection box. A double click generates row groups.

- A right mouse click puts the items from the selection box into the columns, while moving the items from the rows into a selection box. A double click generates column groups.

The row and columns can be switched by a right click on the “pivot button”.

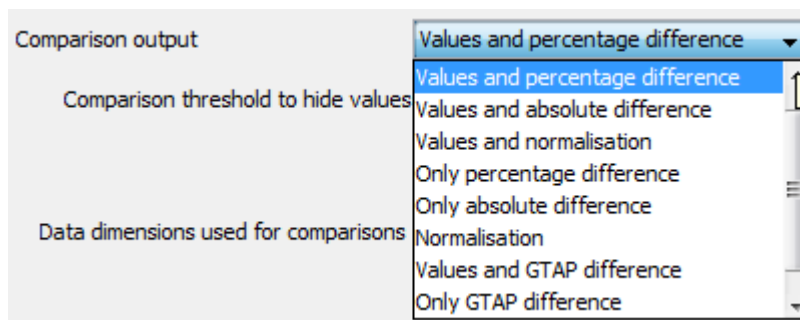
Changing view options: fonts, number formatting and rounding, hiding empty cells, comparisons

A dialog opens when pressing the  button to change various options of the current view:



- *Fonts*: set font family, size and style - affects tabular views.
- *Number formatting*: chose the number of digits and define the decimal separator. The tool supports rounding numbers before the decimal point by allowing for negative fraction digits. Choosing e.g. -1 will round all numbers to tens. The numbers shown in graphics or tables are based on the rounded results is applied.

- *Hide empty rows and hide empty columns* will suppress in the currently seen view, any columns and rows which would show only blank cells.
- *Cut off limit to determine empty cells* . In standard mode, the interface will treat zeros as missing values, and items will be shown as blanks. But the user might also enter a different value (any value, in absolute terms, below the threshold will be treated as if it was zero).
- *Use default pivoting for table*: That is the normal mode, where the pivot is defined by the table views. By clicking that off, the currently chosen pivot (from the current table or manually defined) will be kept even if a different table is chosen.
- *Show histogram*: A histogram is shown additionally to the current view as a separate window. The current window might however hide the histogram window, so that minimizing other windows might be required.
- *Use classification colors for tables*: Use the colours which would be used to colour the regions in a thematic map to colour the numbers shown in tables.
- *Use of short code and/or long texts*
- *Comparison output*: the exploitation tools can add different types of comparison output. They also affect what is shown in maps and graphics.



“Normalisation” means that the value is divided by the comparison points, allowing e.g. also to calculate shares. The “GTAP” difference is a compromise between a percentage and an absolute difference: it multiplies the difference in the logs with the difference (thanks to Rob McDougall from the GTAP team in Purdue for the proposal).

In tables, the “and” options will show two lines in each data cell, one with the observations, and one with the comparison output, as seen below.

BIOF_DZE2				
	Income [Euro/ha or head]	Hectares or herd size [1000 ha or hds]	Yield [kg or 1/1000 head/ha or head]	Supply [1000 t or 1000 animals]
Cereals	438.65 -17.73%	59217.85 -2.08%	4958.80 -3.58%	293649.34 -5.58%
Oilseeds	454.55 -30.01%	8335.37 -13.24%	2271.59 -5.17%	18934.54 -18.60%
Other arable crops	1693.24 -4.26%	7938.25 2.27%	32743.16 -2.99%	259923.47 -0.78%

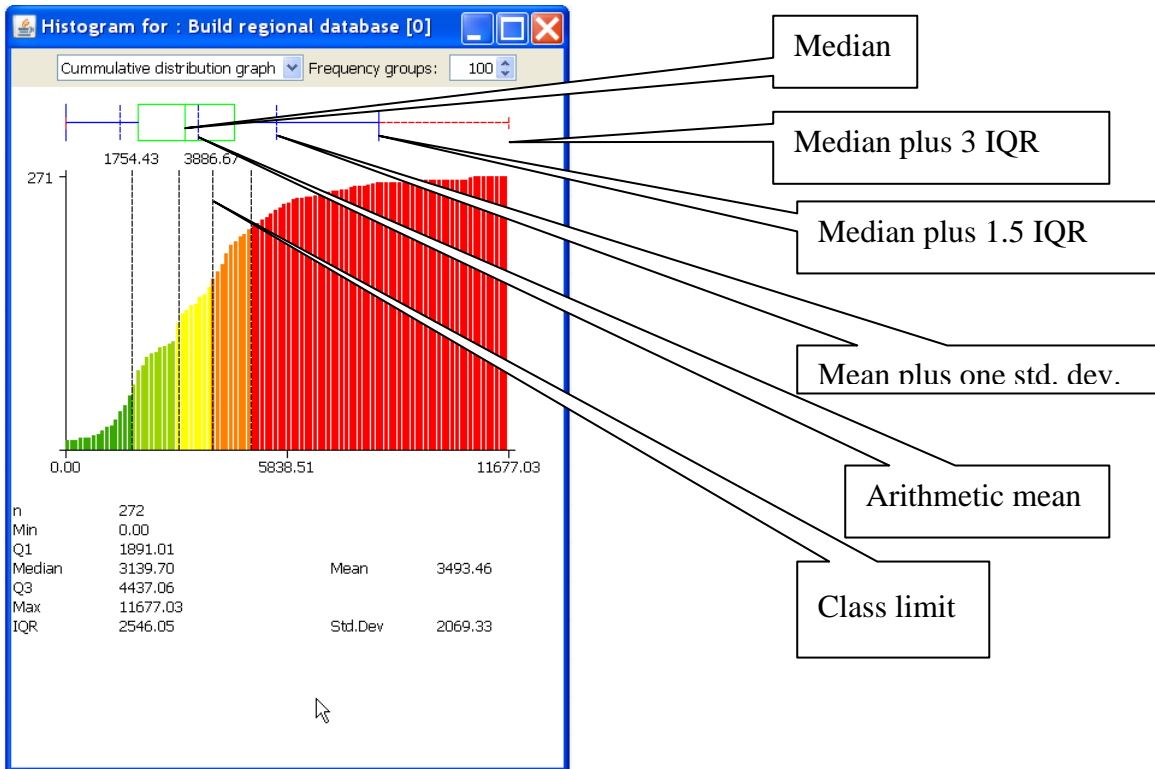
The “Data dimensions used for comparisons” allows to select one or several dimensions used for comparisons. A typical application is the comparison against a scenario. The use of more than one dimension allows e.g. to compare values against a specific year of the reference scenario. For each comparison dimension, a drop-down list allows to select the “Element used for comparisons” defined the comparison point. If statistics had been added to the a view, these can be used for comparisons as well, they can be found at the bottom of the selection list.

Showing a histogram window

The system offers different ways to retrieve information about the distribution. For maps and tables, the user can show an additional window with a box and whisker diagram, histogram and some descriptive statistics as shown below. The box and whisker diagram is defined as follows: the green box shows the first (Q1) to third quartile (Q3), so that the width of the box is equal to the so-called inner quartile range (IRQ). The blue “whiskers” are defined by Q1 minus 1.5 times IQR and Q3 plus 1.5 times IQR, but bounded by the minimum and maximum of the observations. In many applications, any value falling outside that range is classified as a mild outlier. The red dotted whiskers are at Q1 minus 3 times IQR and Q3 plus 3 times IQR, but bounded by the minimum and maximum of the observations. In many applications, any value falling outside that range is classified as a stronger outlier.

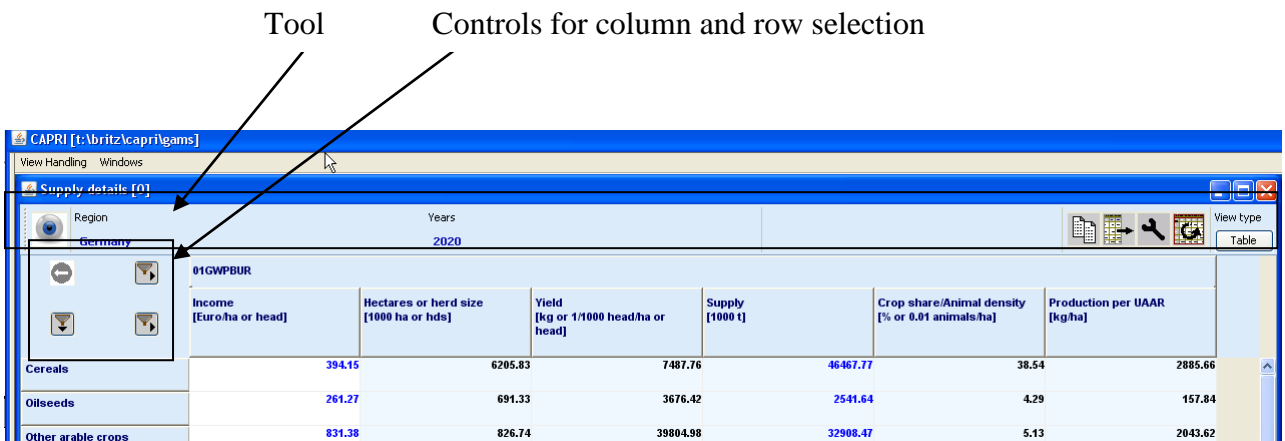
The user can restrict the plotted range as to exclude stronger outliers. If outliers are present, the red dotted whiskers at the tail with strong outliers are removed.

The blue dotted lines show the mean, and +/- one standard deviation around the mean. For a normal distribution, that would cover around 2/3 of the observations. The black dotted lines in the histogram show the class limits used for the colour model. The bottom reports some descriptive statistics. The technical implementation is set up according to the way maps are drawn: the population consists of all values in the rows and the columns of the table, and thus differs from the outlier control, which treats each column as a separate set of observations.

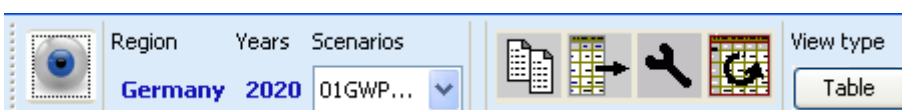


The colours are typically used to visualize the distribution in maps, but, as a second option, they can also be applied to the numerical values in tables. Alternatively, histograms and box and whisker diagrams can be drawn via the graphics.

Working with tables



The toolbar



Tooltips for column and row headers

For predefined tables, tooltips may be stored which give additional information on the columns and rows. They will appear when the mouse is moved over the respective column or row header.

Y				
Income [Euro/ha or head]	Hectares or herd size [1000 ha or hds]	Yield [kg or 1/1000 head/ha or head]	Supply [1000 t or 1000 animals]	Crop share [% or 0.01 €]
244.87	4484.26	6228.03	9244.00	
Premiums + Revenues - variable costs according to the definition of Economic Accounts for Agriculture, income available to farmers to pay for land, labour, capital and profits				
398.57	63.49	2945.76	187.02	
1171.26	140.57	39192.45	5509.33	
30713.87	14.42	785446.28	4116.05	

Drill-down


Some views comprise hyper-links to other tables. Numbers with hyperlinks are shown in blue

2515.60	
27	Double click to table : Supply details

, and a tooltip will appear when the mouse is moved over them.


Double-clicking in the cell will jump to the connected table.

Clipboard export

The content of the currently shown view can be copied to the clipboard by pressing the  button. Tables are placed as tab delimited text in the clipboard, so that they can be pasted into spreadsheets. Graphics and maps are placed as graphics in the clipboard and can be copied e.g. into word processing.

Note: If copying numbers from the clipboard to EXCEL, it might be necessary to change the delimiter.

Export to file

A dialog opens when pressing the  button to export the full dataset of the view to a file – not only the currently seen part. The action provoked by the button depends on the view type. In tabular view, in opposite to the clipboard export, the export file will scroll through the outer dimensions and will copy all stacked tables after each other into a file. Take the table below as an example. Clipboard export will export the data for Belgium and 1984. File export will export data for *all* regions and for *all* years, if the user does not apply filters in the export dialog. An example is discussed on page 78.

RAW		
	Unit value EAA [Euro t]	Quantity [1000 t]
Production value		
Cereals	2515.60	
Oilseeds	22.00	
Inputs		
Premiums		
Production value		
Cereals	2515.60	4137.40

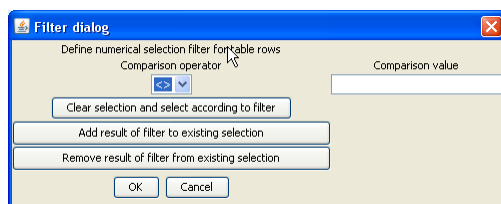
Sorting

The rows can be sorted by one or several columns by clicking with the left mouse button in the column headers. Adding additional sorting columns is achieved by pressing the “shift”-key and then using the mouse as explained before. A sorting symbol will show sort direction, and its size will show the sorting order.

Cereals	Oilseeds
60473.37	9607.46
0.00%	0.00%
59217.85	8335.37
-2.29%	-13.29%
5142.97	2421.03
0.00%	0.00%

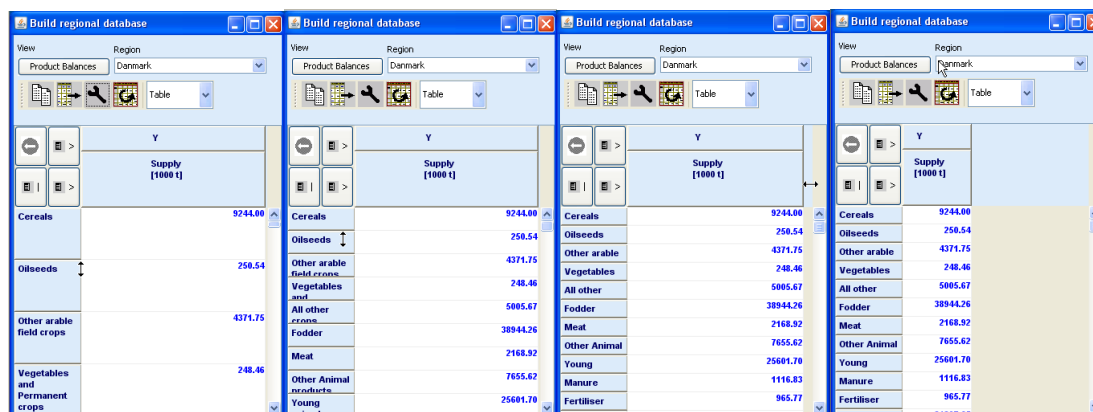
Numerical filtering based on cell content

Clicking with the right mouse button on one of the column headers will open the “filter dialog” which can be used to apply numerical filters to remove rows not matching the filter from the view.



Changing the row height and column width with the mouse

While dragging with the mouse the bottom of the first row header, the cell height of each row (the height of each row) is changed at the same time. But, the column width can be changed selectively per each desired column (if you change the width on one column, the widths of the other columns do not change). The column width can be changed in a similar way by dragging the right border of the column header. Alternatively, the size can be set in the “Changing view options” dialogue.



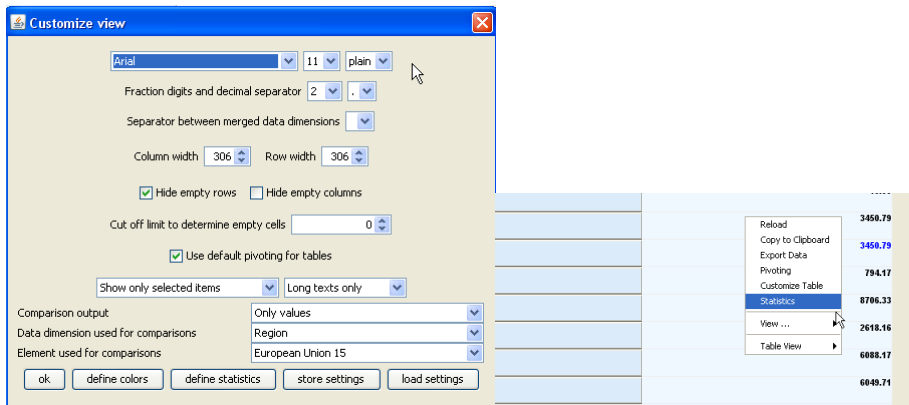
Adding statistics

The user may add different statistics as rows to the table as reported in the following table. The observations are assumed to be mapped into the rows of the current views. Zeros can be treated as missing values. The statistics summarize the observation separately for each column.

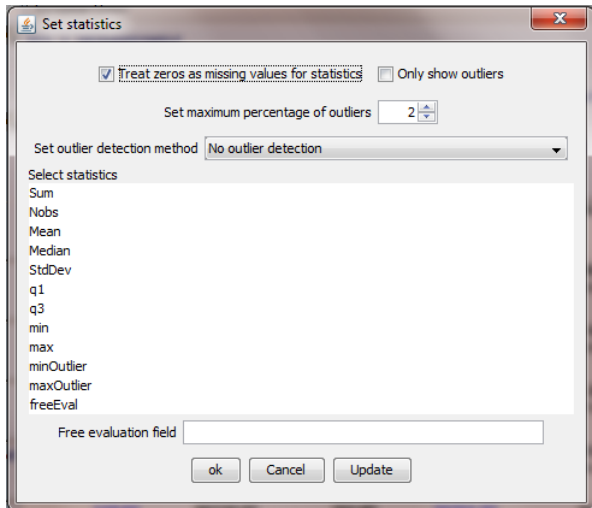
Statistics	Shortcut
Sum over observations	Sum
Number of observations	nObs
Arithmetic mean	Mean
Median	Median
Standard Deviation	StdDev
End value in first quartile	q1
First value in fourth quartile	q4
Minimum of the values	min
Maximum of the values	max
Minimum limit for outlier detection as defined from user settings	minOutlier

Maximum limit for outlier detection as defined from user settings	maxOutlier
Free chosen algebraic expression	freeEval

The above related options can be either found in the “customize dialogue” box, which opens by clicking the button on the toolbar, using the “define statistics” button, or by right clicking on any cell inside the table to open the popup menu, and choosing “Statistics”.

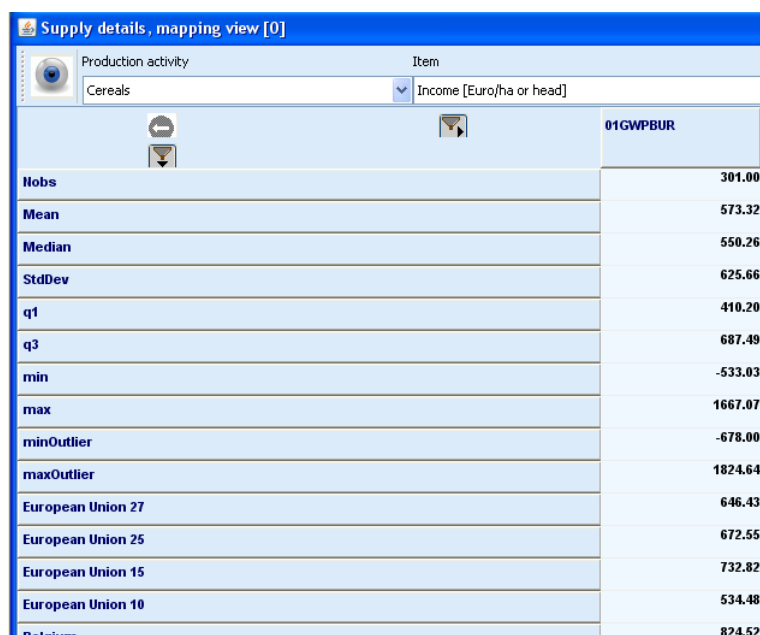


The dialog has the options as shown below, which in parts are dynamically changing depending on the detection algorithm.



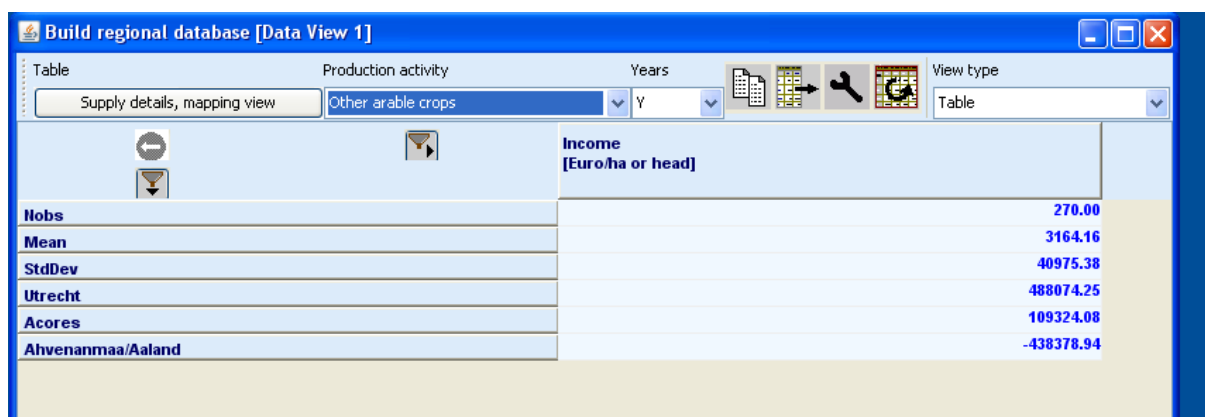
The free evaluation field follows the convention used for the definition of <eval> fields in table definition, see programming guide.

The selected statistic options will appear as first rows of the table:



	01GWPBUR
Nobs	301.00
Mean	573.32
Median	550.26
StdDev	625.66
q1	410.20
q3	687.49
min	-533.03
max	1667.07
minOutlier	-678.00
maxOutlier	1824.64
European Union 27	646.43
European Union 25	672.55
European Union 15	732.82
European Union 10	534.48
Belgium	824.52

Perhaps the most interesting option is to show only the outlier rows besides the statistics in the table, as illustrated below:



	Income [Euro/ha or head]
Nobs	270.00
Mean	3164.16
StdDev	40975.38
Utrecht	488074.25
Acores	109324.08
Ahvenanmaa/Aaland	-438378.94

Outlier detection algorithms implemented

The GUI offers currently the following ways to look up possible outliers. For all the methods, the user may additionally define a maximum percentage of observations show in which case only the largest or smallest outliers according to the outlier detection algorithm shown will be selected.

Standard deviation around the mean

The user can define the factor β before the standard deviation. Observations are marked as outliers when their distance to the arithmetic mean exceeds the value defined by the multiplication of the standard deviation σ and a user defined factor β :

$$\neg(\bar{x} + \beta_{\sigma} \cdot \sigma > x_i > \bar{x} - \beta_{\sigma} \cdot \sigma).$$

Large outliers can easily bias the result as they will change both the mean and the standard deviation of the observation sample. Further on, many time series in the CAPRI data base have by definition a lower limit of zero, so that the assumption of normally distributed data sets cannot hold. Therefore, other outlier detection methods are also implemented as discussed below. The dialog allows changing the factor β from its default of 2 which covers 95% of the values for normally distributed data.

Standard deviation of values normalized by median

The values are all divided by the median and the new series is classified as under the option discussed above. The main advantage of that method is the shift to a mid point which is less vulnerable to large outliers in the observations.

Standard deviation of trend line error

A regression is estimated by using the index position in the unsorted values as explanatory values. The resulting errors are then classified according to the first option discussed above. The typical application would be a table where consecutive time points – e.g. years – are shown along the rows.

Median and inner quartile range

Box-and-whisker charts, which are also supported by the graphics view, are using the median and quartile to visualize the distribution. They are also an easy and robust way to detect possible outliers. First, the so-called “inner quartile range (IQR)” is calculated as the difference in values between the beginning value of the first and the ending value of the third quartile. The IQR then consists of the 50% range of values around the median. The IQR is multiplied with a user defined factor β added to Q3 respectively subtracted from Q1 to define the lower and upper bound for regular values. The factor β default value is 1.5. The quartiles and the median are not affected by outliers at the tails of the distribution, allowing for a rather robust way to filter outliers:

$$\neg(Q3 + \beta_{IQR} \cdot IQR > x_i > Q1 - \beta_{IQR} \cdot IQR)$$

Conformity based on relation of distances

Here, the following formulae are used, taken from Last & Kandel (2001):

$$\mu_{i,l} = 2 / \left(1 + \exp \left(\frac{\beta \cdot n \cdot m \cdot (x_{i+1} - x_i)}{x_{i+m} - x_{i+1}} \right) \right)$$

$$\mu_{i,h} = 2 / \left(1 + \exp \left(\frac{\beta \cdot n \cdot m \cdot (x_i - x_{i-1})}{x_{i-1} - x_{i-m-1}} \right) \right)$$

They define “conformity” from below and above by comparing the distance from the current value to its neighbour in relation to the average distance for a predefined group size m . Before the formulae are applied, the values are sorted. In opposite to the outliers based on first and second moment, the method is also able to detect outliers in between “clusters” of values. Inside such a cluster, differences in distances between values are small, so that the relation between the distance to the next neighbour, and the average distance between the neighbour and its m -th neighbour is around unity. The big advantage of the approach is that it does neither assume a certain functional form for the distribution (as in the case of the mean/standard deviation approach), nor a uni-modal distribution as in the case of the IQR method, and it is rather easy to compute. It may be worth to continue with a literature research in the direction of similar outlier detection methods.

The factor β describes how distances between succeeding values are assessed. Outliers are defined when the maximum of the above and below conformity is above a predefined threshold α .

$$\neg(\max(u_{i,l}, u_{i,h}) > \alpha)$$

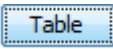
Last & Kandel have tested their algorithm for $\beta=0.001$, $\alpha=0.05$ and $m=10$. There seems to be a rich literature on that kind of “neighbourhood distance“, where outlier control based with different algorithms is analyzed in detail. The different parameter can be set by the user interface.

Reference: Last M. & Kandel M. (2001), Automated Detection of Outliers in Real-World Data, Proc. of the Second International Conference on Intelligent Technologies

Working with graphics implemented

The exploitation tools allow showing the current content of a tabular view as a graphic. Most of the graphic types are based on the JFreeChart library (see <http://www.jfree.org/jfreechart/>).

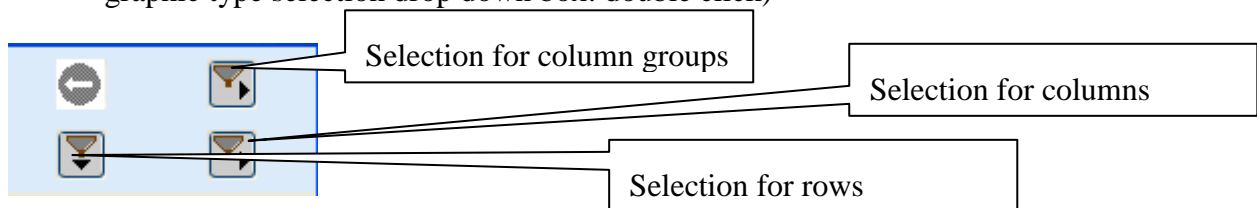
General handling of graphs

In the system, the selection of graphs is based under the  bottom in the tool bar and the following *graphic types* are currently supported:

- Bar charts
- Line charts
- Area chart
- Spider chart
- Pie chart
- Box and Whisker chart
- Histogram
- Markov chart

The *selection* of rows and columns shown in the graph can be set in three different ways, for all types of graphics:

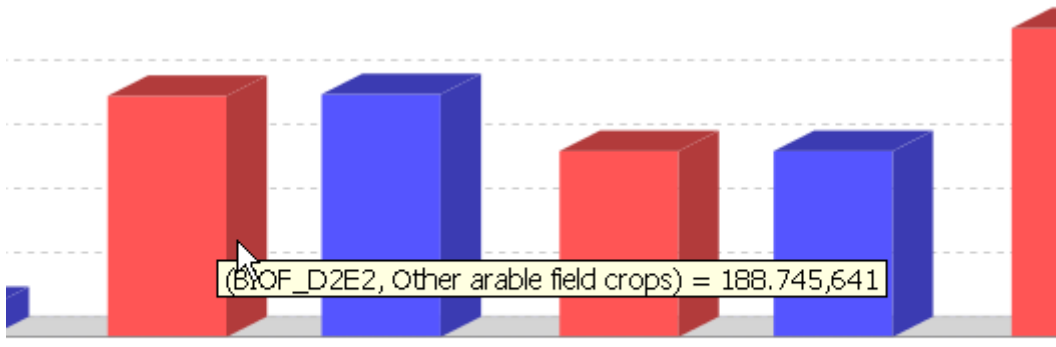
- Using the selection dialog (upper left corner of the table, or the buttons next to the graphic type selection drop down box: double click)



- Using those buttons in graphic mode: single clicks with the left mouse button will scroll down in the list, right mouse, single clicks will scroll up.
- Scrolling the table with the scroll bar to a specific position. The column/row in the upper left corner of the table will define the starting point for the graphic.

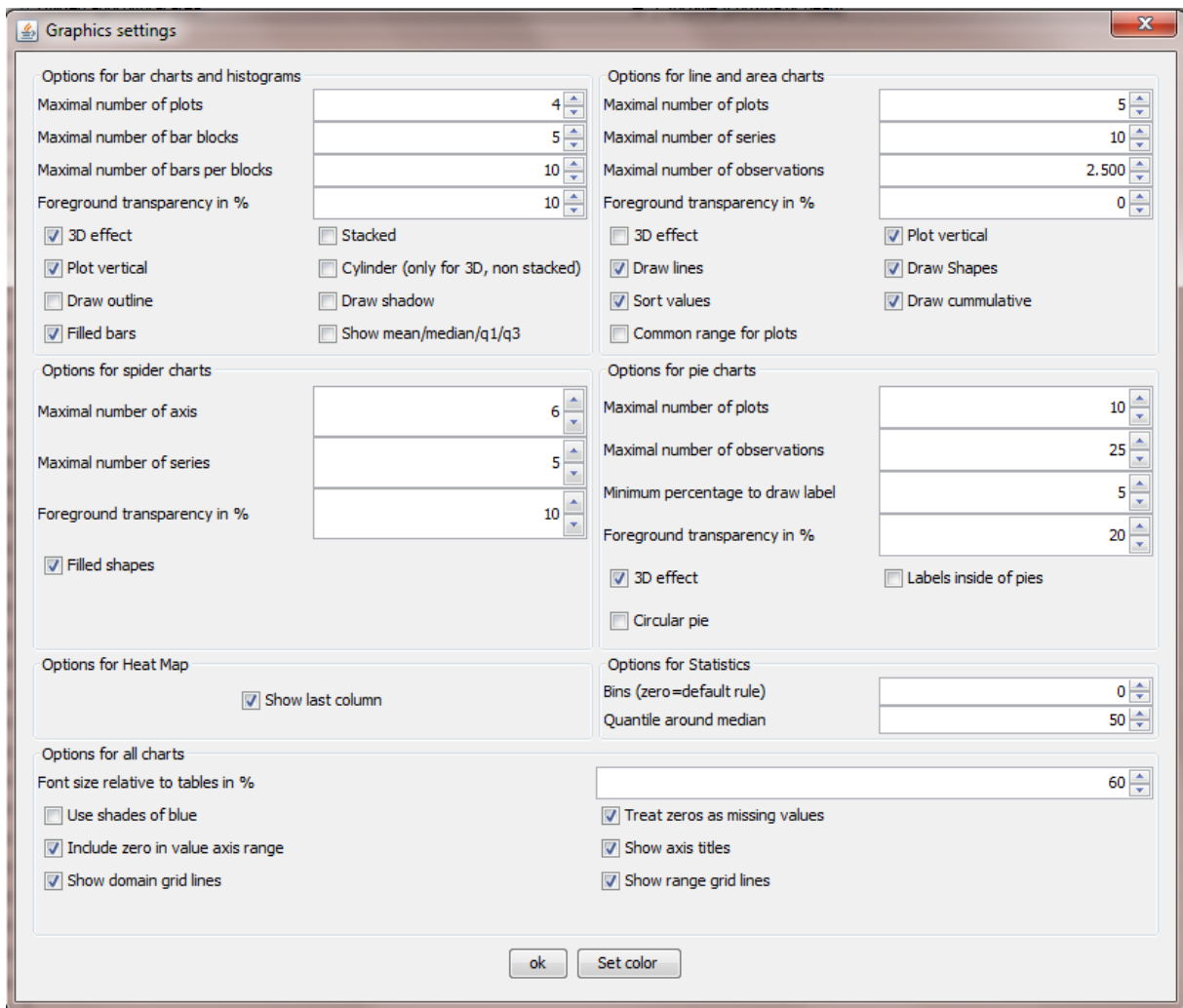
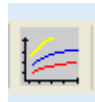
All types of graphics support *tooltips* to query the numerical values underlying the graphic.

The tooltips appear when moving the mouse on a graphic element linked to the value as e.g. a bar.



Perhaps an unexpected feature is the zooming in and out with the mouse. The graphs support, *saving* to the disk via a popup menu and *printing*. The popup menu also allows changing certain properties for the current graph temporarily. Some settings which will pertain can be

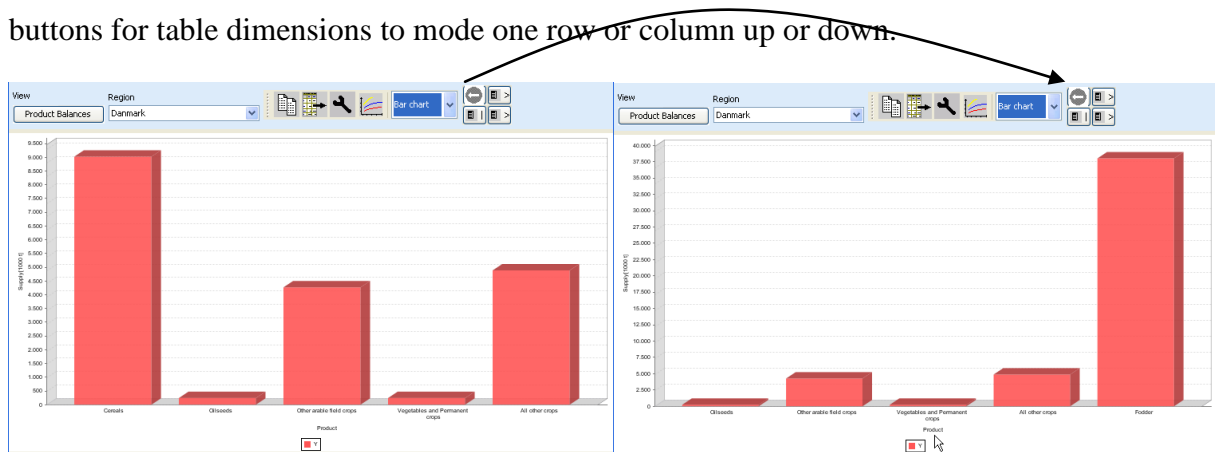
edited by opening the graphics option dialogue, press:



The chart type’s specific settings are discussed in more detail below. The general options should be self-explanatory, it is best to try them out interactively.

“Walking” through the data

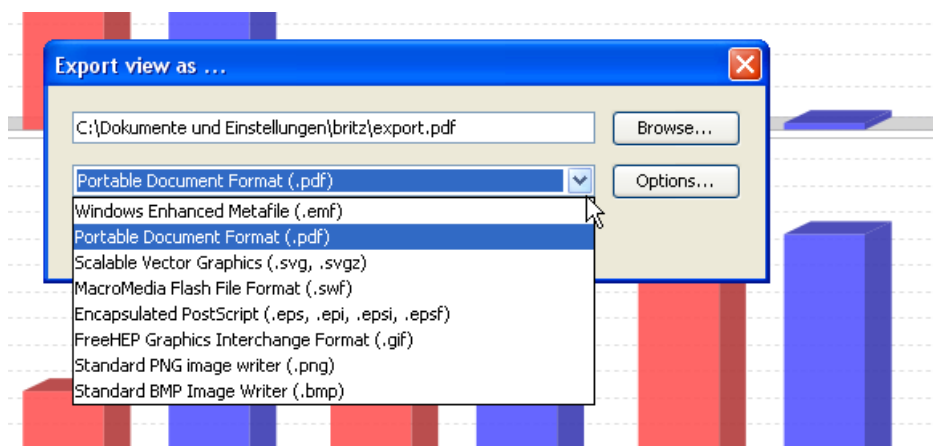
As the maximal numbers of elements shown is restricted (see above), typically not all columns and/or rows will be shown in a graph. The user basically has two possibilities to change the visible columns or rows. Firstly, columns and rows can be selected by the selection dialogues. Secondly, the user can click with the right or left mouse button on the buttons for table dimensions to move one row or column up or down.



Exporting the graphic to file




The graphics can be saved to file in different formats by pressing the export button. The following dialogue will appear which allows the user to define the file, and a range of different file formats. For MS Office users, the “Windows Enhanced Metafile” format is interesting, as it allows changing later the graphics manually, e.g. by adding new text.



Exporting the graphic to clipboard

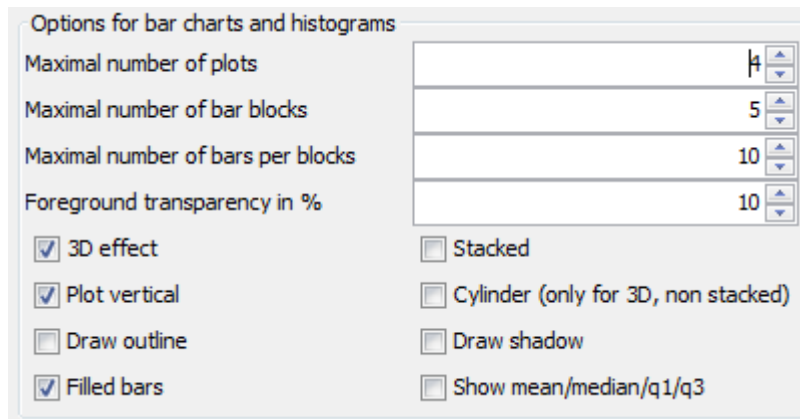
Alternatively, the graphic can be placed into the clipboard where it is stored as a bitmap or as


jpeg by double-clicking the “copy to clipboard”  button.

Bar charts

Bar charts treat the columns – typically the table items – as having different units and consequently assign an own plot with a value axis to each of them. The observations are taken from the table rows and define the domain, the horizontal axis. Each groups of bar columns present – typically the scenarios – receives its own colour. An example is given below.

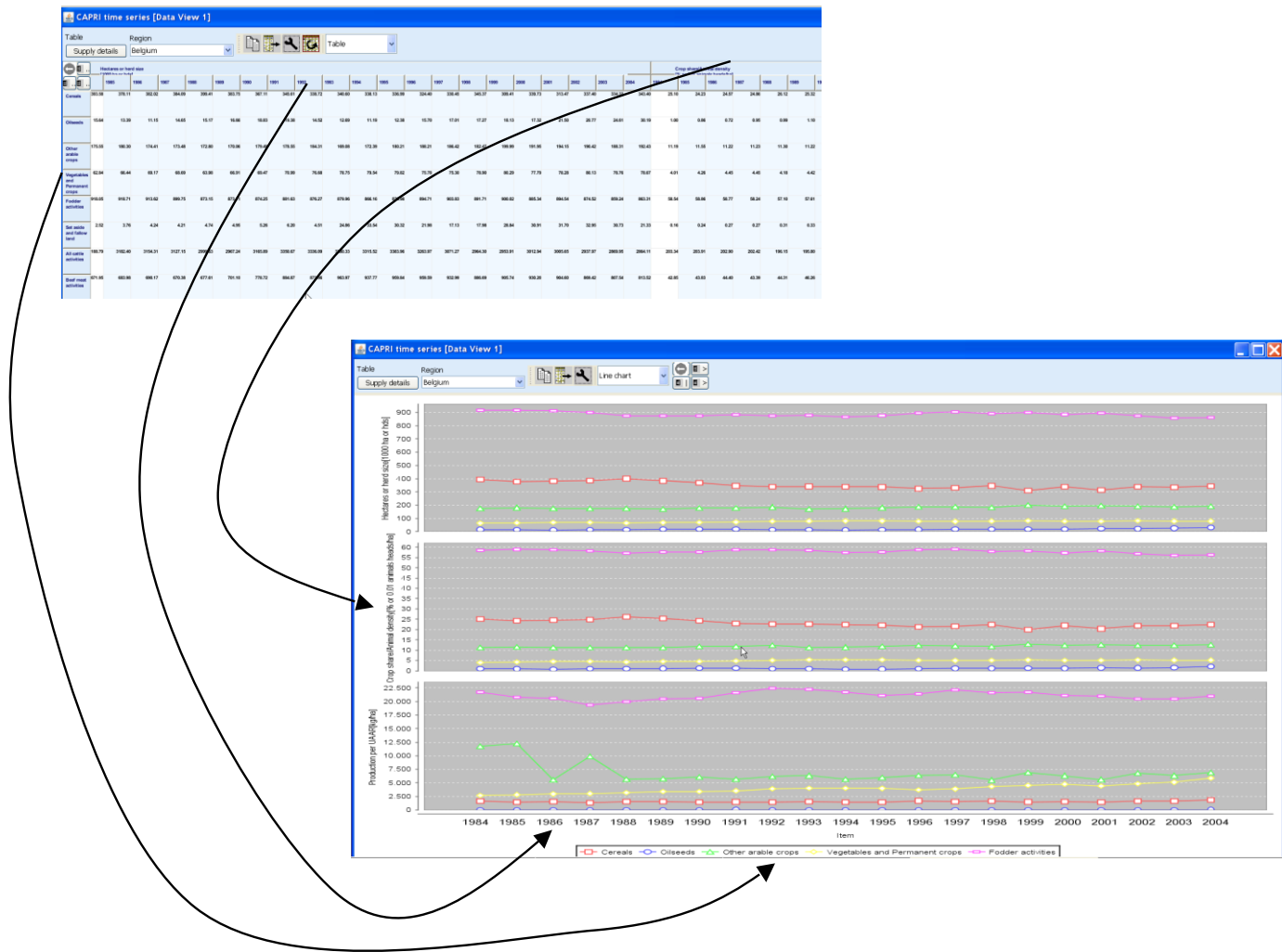




The user has a number of options for the bar charts. By pressing the  button in the toolbar, a dialog box including the section of “Options for bar charts” opens. The number of plots refers to the number of columns in the underlying tables, each column will receive an own plot with a matching value axis. The bar blocks refer to the rows, each bar block may comprise several bars taken from the column groups (typically scenarios). As seen above, it is also possible to generate stacked bars from the column groups, or to generate cylinders instead of cubes.

Line and point charts

Line and point charts assume that the columns of the table present some ordered sets e.g. years or iterations. There is currently a default of 25 such observations which can be increased by the user. The different series to plot are taken from the table rows. If different column groups are present, those receive their own plot with an own value axis.



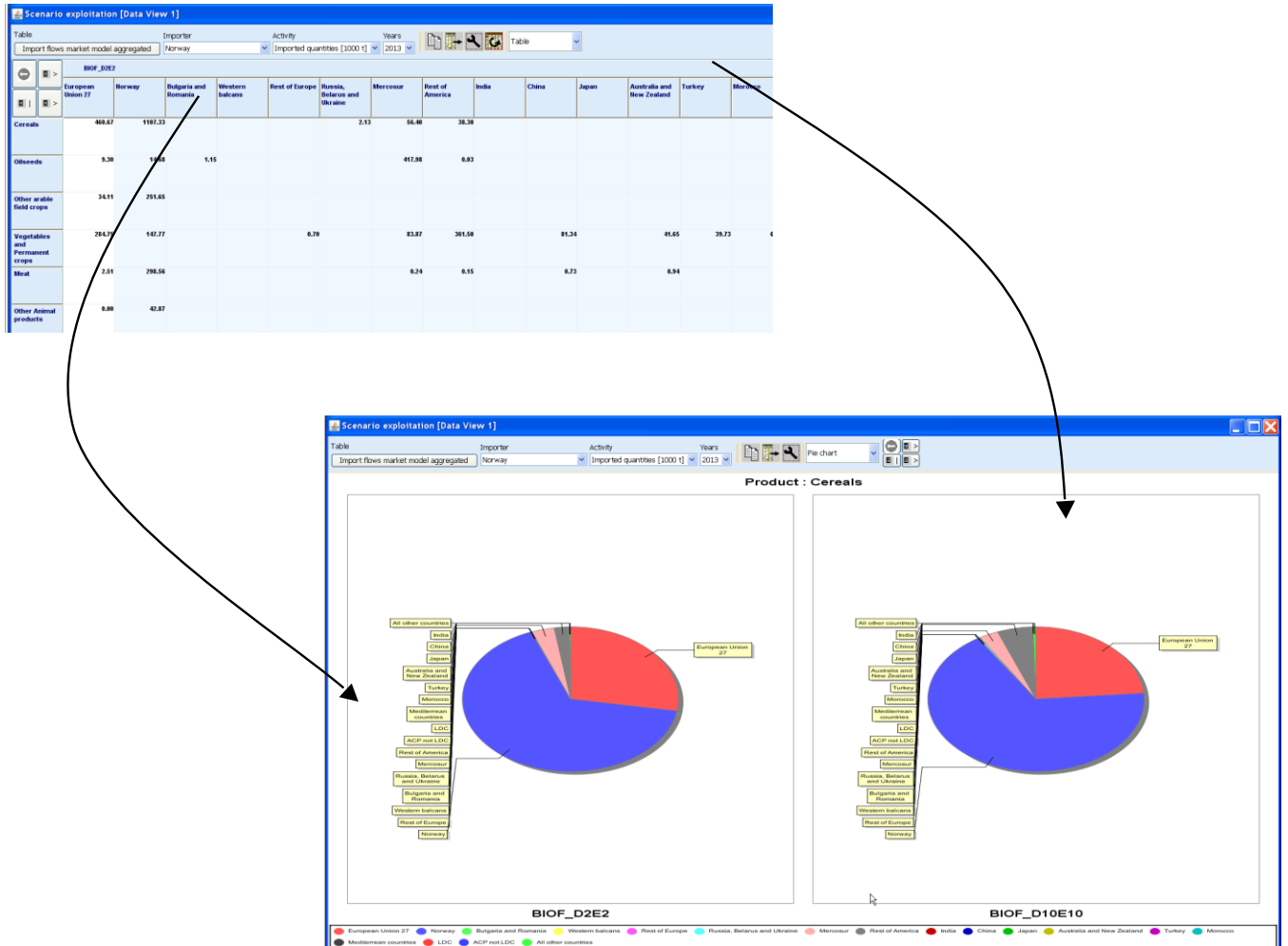
Options for line and area charts

- Maximal number of plots: 5
- Maximal number of series: 10
- Maximal number of observations: 2.500
- Foreground transparency in %: 0
- 3D effect
- Draw lines
- Sort values
- Common range for plots
- Plot vertical
- Draw Shapes
- Draw cummulative

The options for line and area charts are similar to the ones for bar charts. The number of plots refers to the column groups, the number of series to the rows of the table. Area charts are equivalent to stacked bars, i.e. the observations are added. The number of observations is linked to the columns.

Pie charts

Pie charts are useful to show shares on total as e.g. trade flows. The shares are calculated from the columns, whereas each column group – typically scenarios – receives its own pie. Only one row is allowed.



The user has the following options to modify the presentation of pie charts:

Options for pie charts

Maximal number of plots:

Maximal number of observations:

Minimum percentage to draw label:

Foreground transparency in %:

3D effect Labels inside of pies

Circular pie

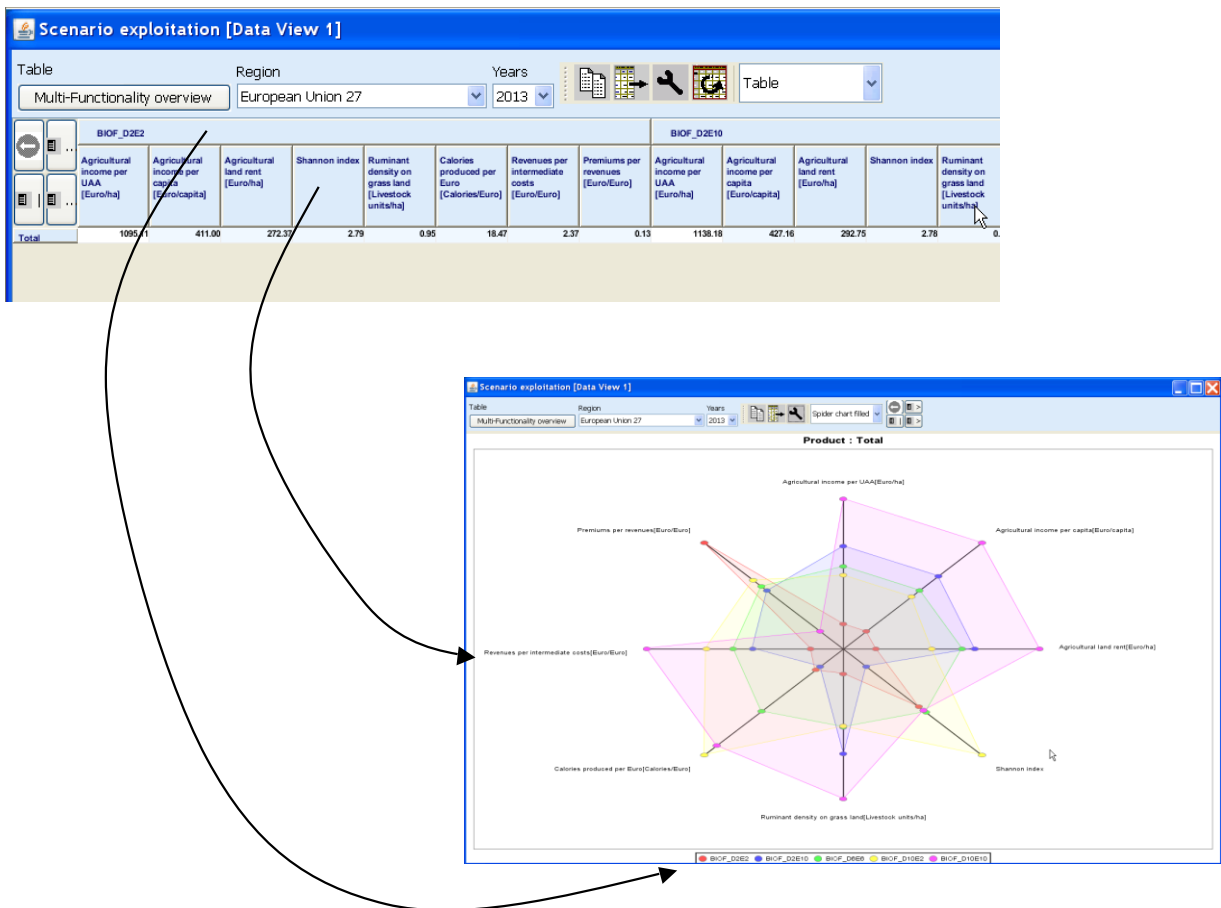
The maximum number of plots refers to the number of elements in the dimensions of the column group., The example above shows two plots. The number of observations defines the

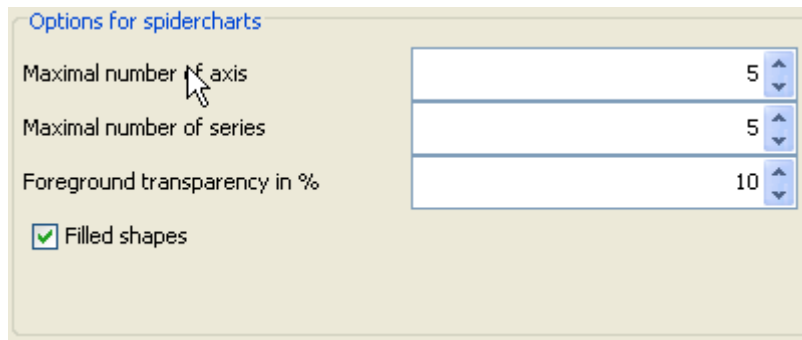
numbers of pies – if more columns are available, the cake will eventually give a wrong impression if not all values are used to define the sum and the shares.


The “minimum percentage to draw label” defines a lower cut-off limit, if a cake’s size is below the threshold, no label will be drawn. As shown in the example above, setting the threshold to 100% will erase the labels (see Pie chart maps for an example). It is also possible to place the labels in the pies, and not outside of the cake as shown in the example above.

Spider plots

Spider charts are useful to compare several dimensions simultaneously across a range of alternatives. It is assumed that the columns show the items of which each receives its own axis, whereas the column groups are the alternatives to compare. The axis are not ticked with numerical values, instead they are always scaled to cover the minimum and maximum found in any alternative.





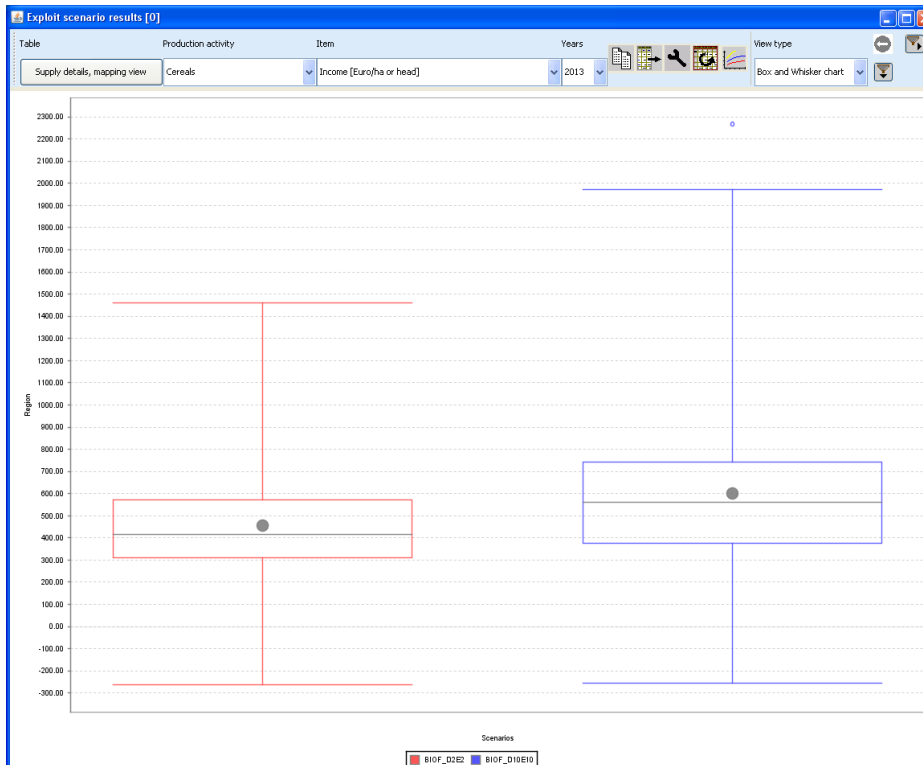
The options for spider charts which are found under the  button in the toolbar are rather limited. The user can determine how many axes – taken from the columns – are included in the diagram and the maximum number of series, which typically consist of scenarios.

Box and Whisker charts

In descriptive statistics, a box plot or boxplot (also known as a box-and-whisker diagram or plot) is a convenient way of graphically depicting groups of numerical data through their five-number summaries (the smallest observation (sample minimum), lower quartile (Q1), median (Q2), upper quartile (Q3), and largest observation (sample maximum)). A boxplot may also indicate which observations, if any, might be considered outliers (see also http://en.wikipedia.org/wiki/Box_plot).

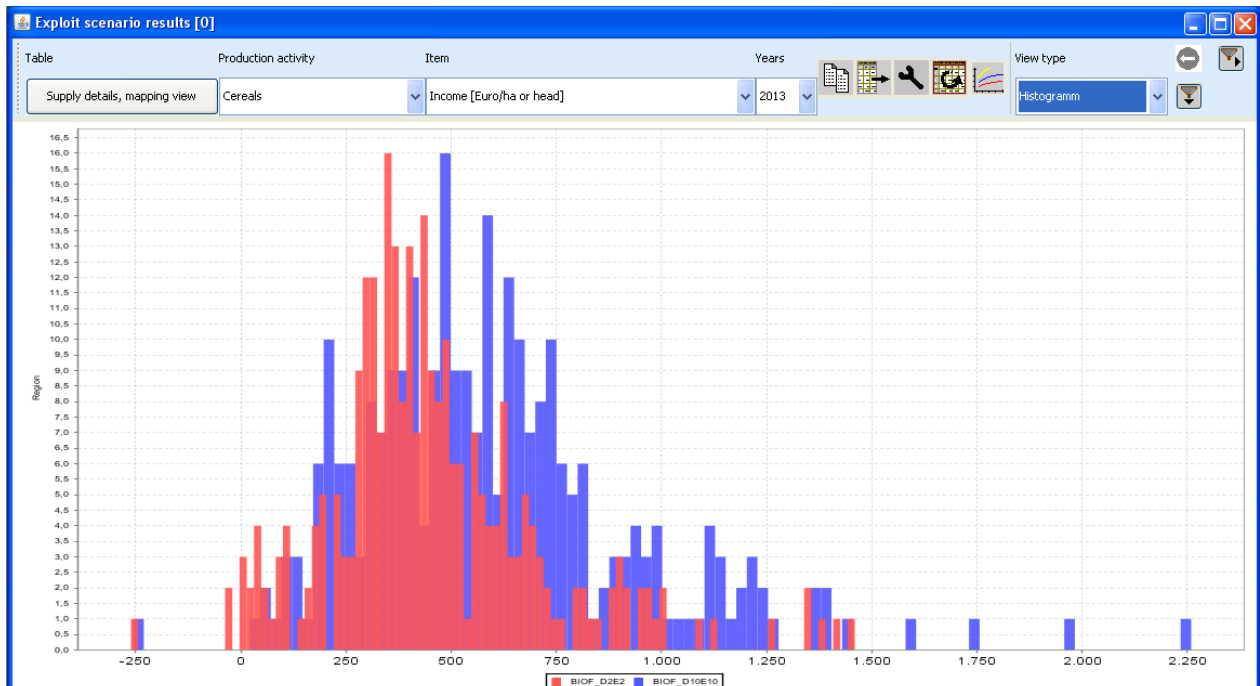
Boxplots can be useful to display differences between populations without making any assumptions of the underlying statistical distribution: they are non-parametric. The spacings between the different parts of the box help indicate the degree of dispersion (spread) and skewness in the data, and identify outliers. Boxplots can be drawn either horizontally or vertically (text so far from Wikipedia).

The box and whisker chart uses the rows as the observations, and generates an own graph per column. The box shows +/-25% of the observations around the median which is shown as a grey line, whereas the arithmetic mean is shown as a grey circle. The whiskers show the median +/- three times the inner quartile range. Mild outliers are drawn as dots and strong outliers are indicated by arrows. So far there are no specific options for that type of diagram.



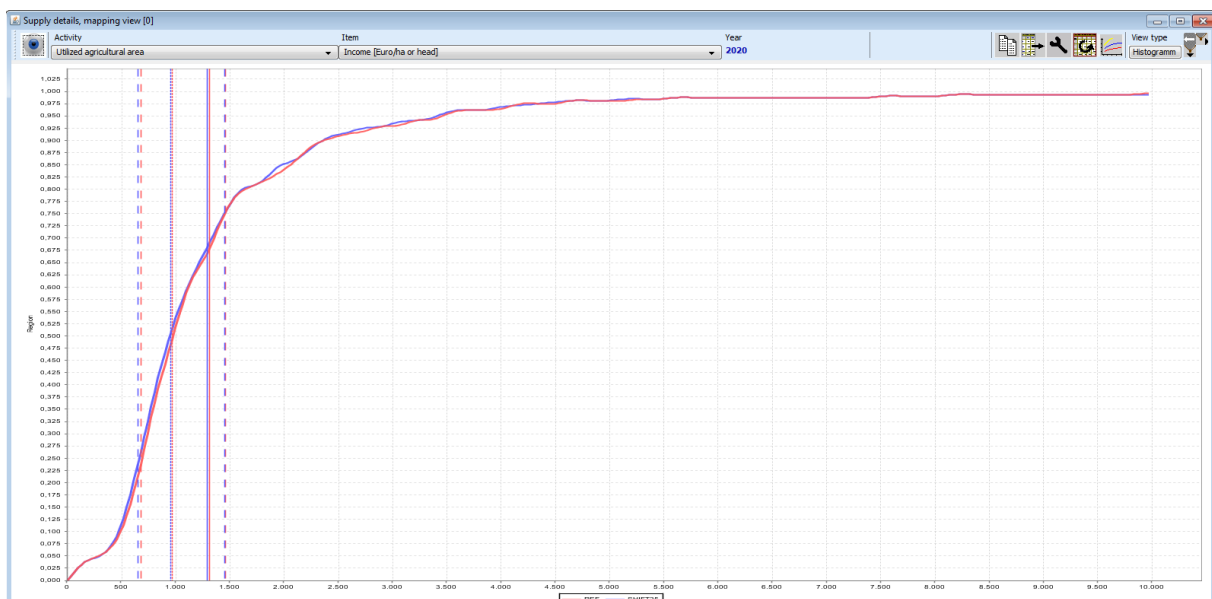
Histograms

As for whisker charts and statistics shown in tables, the observations are taken from rows, and different columns are charted individually. Please note that it is also possible to generate a separate Histogram window, but then, the observations refer to all columns simultaneously.



Some tips:

- If the data set comprises zeros which should be interpreted as **missing values**, check the box “Treat zeros as missing values”. Otherwise, the value axis will show a zero as the lower bound even if “Include zero in value axis range” is not ticked.
- The **number of bars** (the so-called bins) can be set with a spinner in the second lower panel in the graphic dialogue. The zero as the default value determines the number of bins automatically as the minimum of 100 or the number of observations divided by 5.
- It might be worth to increase with **transparency** of the bars to better capture overlapping parts of the distribution. It might also be worth to use unfilled bars.
- In order to draw a **continuous distribution** instead of bars, set “draw outline” and “Filled bars” to off, and use “Draw line” which generates a graph as seen below
- With “Show mean/median/q1/q3”, switch on the marker lines (mean: normal line; median: dotted; q1/q3: dashed) as seen in the graph below.

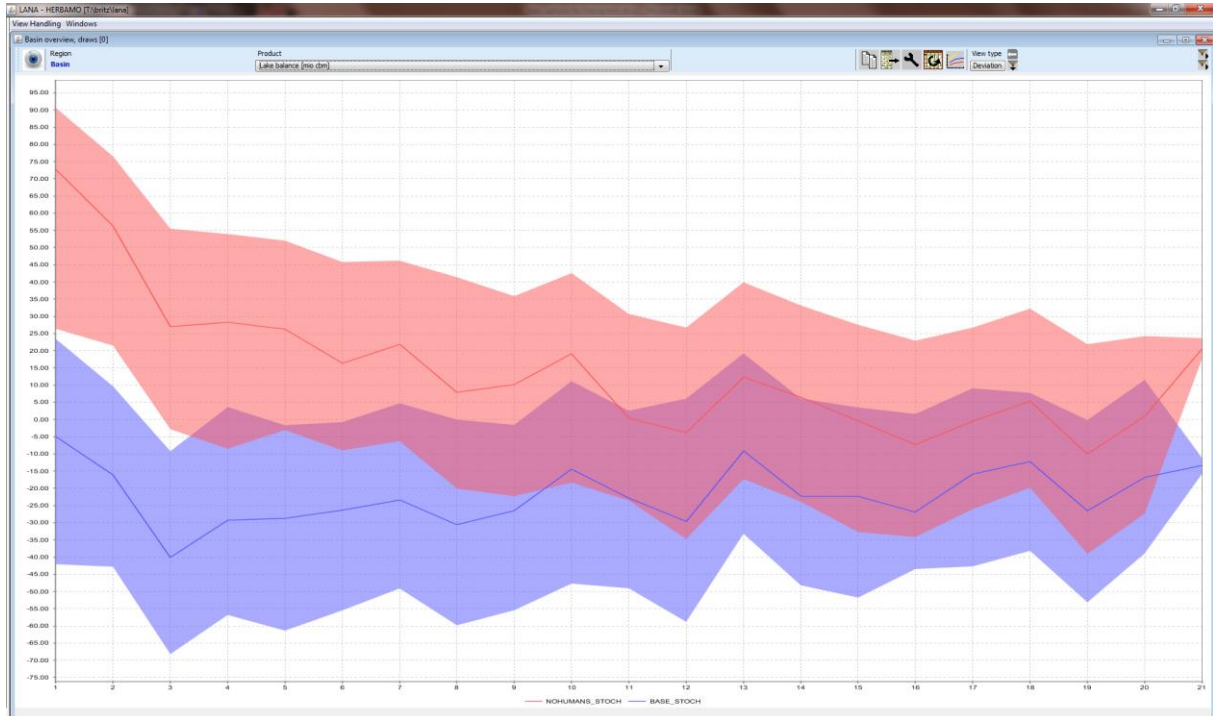


As the continuous distributions are drawn with a spline renderer, they can be quite nicely smoothed if the number of bins is decreased (in the example below from 1000 to 100 to 10 bins):



Deviation renderer

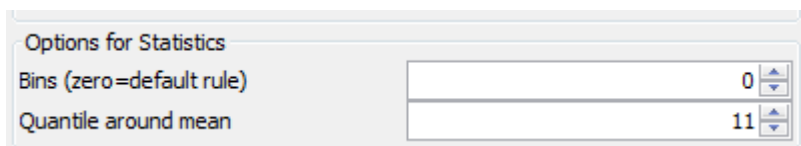
A deviation renderer shows the median of a series along with a symmetric quantile around it. That makes only sense if one has an ordered series, e.g. points in time.



In order to produce such as graph:

- Put the observations (e.g. years) in the columns
- Put the draws in the rows
- Put the scenarios in the column groups

The bandwidth of the graph can be changed with the “Quantile around mean” setting in the graphic dialogue, below the bandwidth is 22%: 11% above and 11% below the median.



The deviation renderer can be expanded to resemble a **contour plot** which shows the density inside the band by lines which combine point for each observation of equal cumulative probability, if the draw line shapes is switched on:



How to draw a line chart with mean / min / max etc. over a time series

Alternatively to a deviation renderer, one can also show more statistics as time series. In order to do so:

- Put the observations (i.e. the stochastic draws) into the rows of a table
- Put the time dimension in the columns
- Use the pop-up menu in the table under “statistics” to show the statistics you are interested in the table

	1	2	3	4	5	6	7	8
d1	1887.22	1886.60	1885.96	1885.36	1884.80	1884.30	1883.85	1883.40
d2			1886.98	1887.24	1886.43	1885.43	1885.39	1885.35
d3			1887.70	1887.67	1887.72	1887.41	1886.75	1886.70
d4			1887.45	1887.52	1887.45	1886.91	1885.50	1885.45
d5			1889.03	1888.23	1888.23	1887.32	1886.39	1886.34
d6			1888.30	1888.54	1887.87	1886.53	1886.07	1886.02
d7			1885.80	1885.56	1885.88	1886.13	1886.68	1886.63
d8			1886.91	1886.93	1886.85	1886.93	1886.35	1886.30
d9			1886.92	1886.98	1886.58	1885.71	1884.75	1884.70
d10			1886.25	1885.01	1884.90	1885.45	1884.95	1884.90
d11			1888.40	1887.84	1888.18	1888.81	1888.33	1888.28
d12			1886.97	1886.29	1885.86	1886.50	1886.50	1886.45
d13			1887.26	1888.27	1887.85	1886.50	1885.70	1885.65
d14	1887.49	1886.48	1885.78	1885.77	1885.47	1886.44	1886.29	1886.24
d15	1887.49	1886.49	1886.63	1888.38	1888.21	1888.10	1886.73	1886.68
d16	1887.44	1886.70	1886.04	1886.06	1886.38	1886.28	1886.86	1886.81
d17	1887.66	1887.78	1886.97	1886.42	1887.85	1886.81	1887.87	1887.82
d18	1886.99	1886.80	1887.06	1887.40	1887.45	1886.71	1886.03	1885.98
d19	1888.64	1887.80	1886.66	1885.83	1885.22	1885.55	1884.59	1884.54

1. Add the statistic which you want to show:

Set statistics

Only show outliers

Set factor for mean/sigma

Set maximum percentage of outliers

Set outlier detection method: Standard deviation around mean

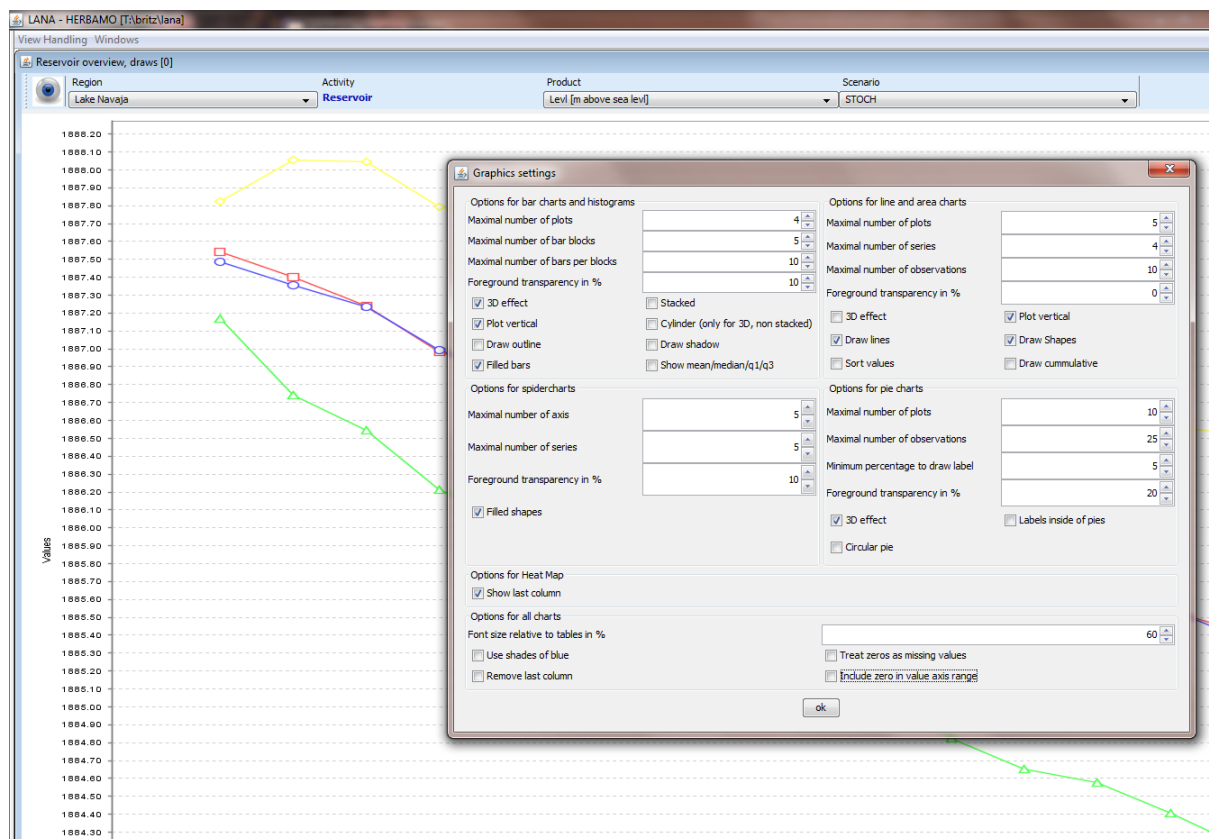
Select statistics

- Nobs
- Mean
- Median
- StdDev
- q1
- q3
- min
- max
- minOutlier
- maxOutlier

Buttons: ok, Cancel, Update

2. Switch to line chart views

Attention: the statistics do not work correctly if several dimensions are merged in the rows.



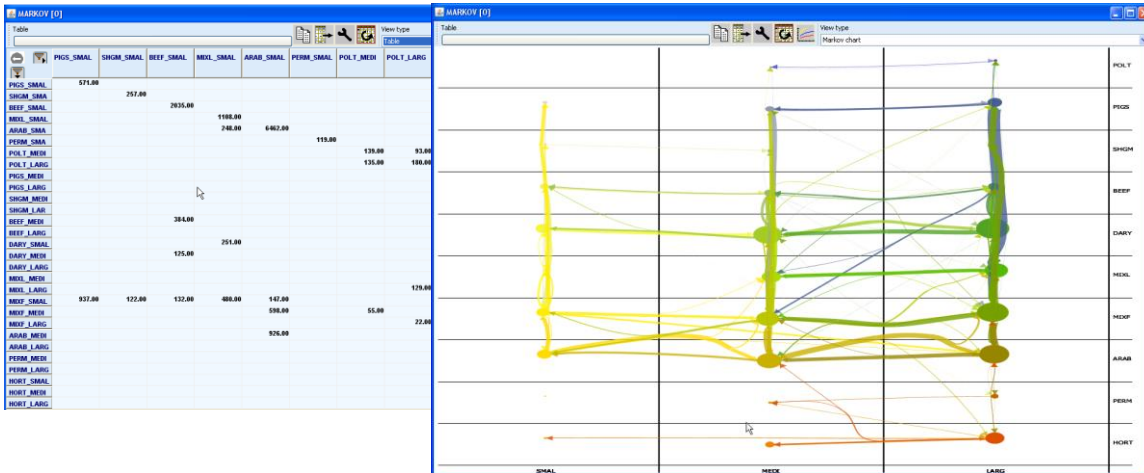
3. And use the graphic dialogue to select as many series as statistics selected (in the above example 4)

Markov charts

A still explorative type of graphics visualizes flows between entities which are placed in a two dimensional co-ordinate system. It is currently not yet used in CAPRI itself, but applied to show flows between farm groups classified by economic size and specialization. As with the flow maps below, the major code based for the graphics is based on work of Doantam Phan³.

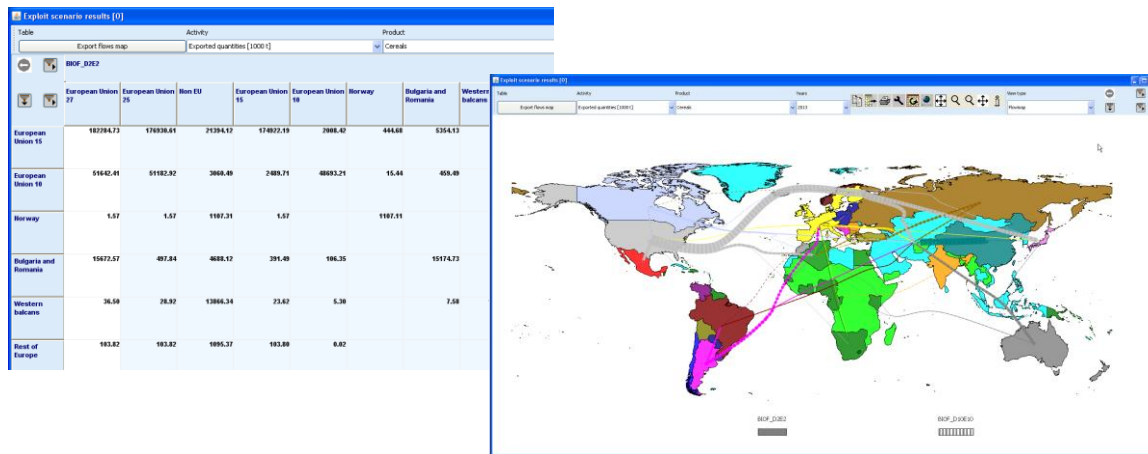
The positions on the x and y co-ordinate are deducted from the codes, taken from a specific section of the underlying XML-definitions which refers to a matching of sub-strings of the codes and x respectively y positions. The size of the dots is taken from the diagonal elements.

³ Flow Map Layout, Doantam Phan, Ling Xiao¹, Ron Yeh¹, Pat Hanrahan, and Terry Winograd, Stanford University, see http://graphics.stanford.edu/papers/flow_map_layout/flow_map_layout.pdf. I would like to thank Doantam Phan for letting the CAPRI team use and modify his source code.

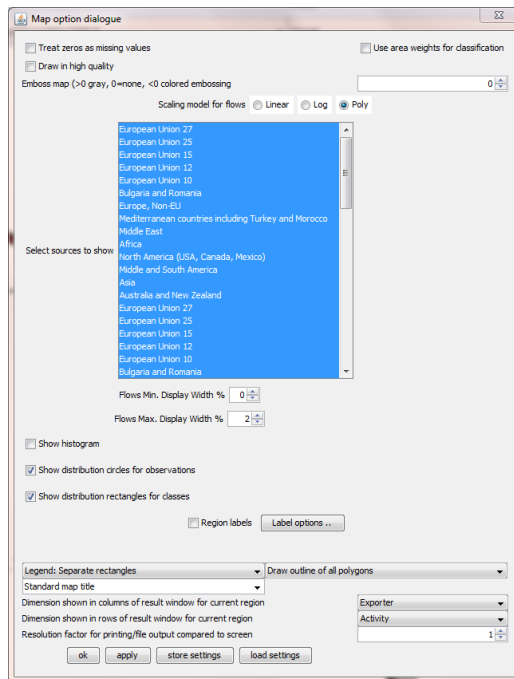


Flow maps

Flow maps visualize flows between regions - The maps are constructed by taken the elements in the rows as the origins of the flows, and the elements in the columns as the destinations. Flows from the same origin are drawn in the same color, the width of the flows relates to their size. Counterfactuals are taken from the column groups and receive a specific “dash”. The picture below shows a screen shot of a flow map for two scenarios.



When pressing the map option button , the following dialogue is opened:



The main options of interest for flow maps are the scaling model and the display width. The following *scaling models* are available:

- *Linear*: the width is determined by relating the flow quantity to the sum of all flows for the same scenario.
- *Log*: the width is determined by multiplying the log of the relation between the flow quantity and the minimal flow with the log of the relation of the maximal and minimal flows for the same scenario.
- *Polynomial*: the relation between the current flow and the maximal flow is raised to a power determined by taking the log of the relation between the maximal and minimal display width divided by the log of the regional between the maximal and minimal flow.

The user can prevent that small flows are drawn by setting a minimal width relative to the size of the window; equally, the maximal possible size of a flow relative to the size of the window can be determined.



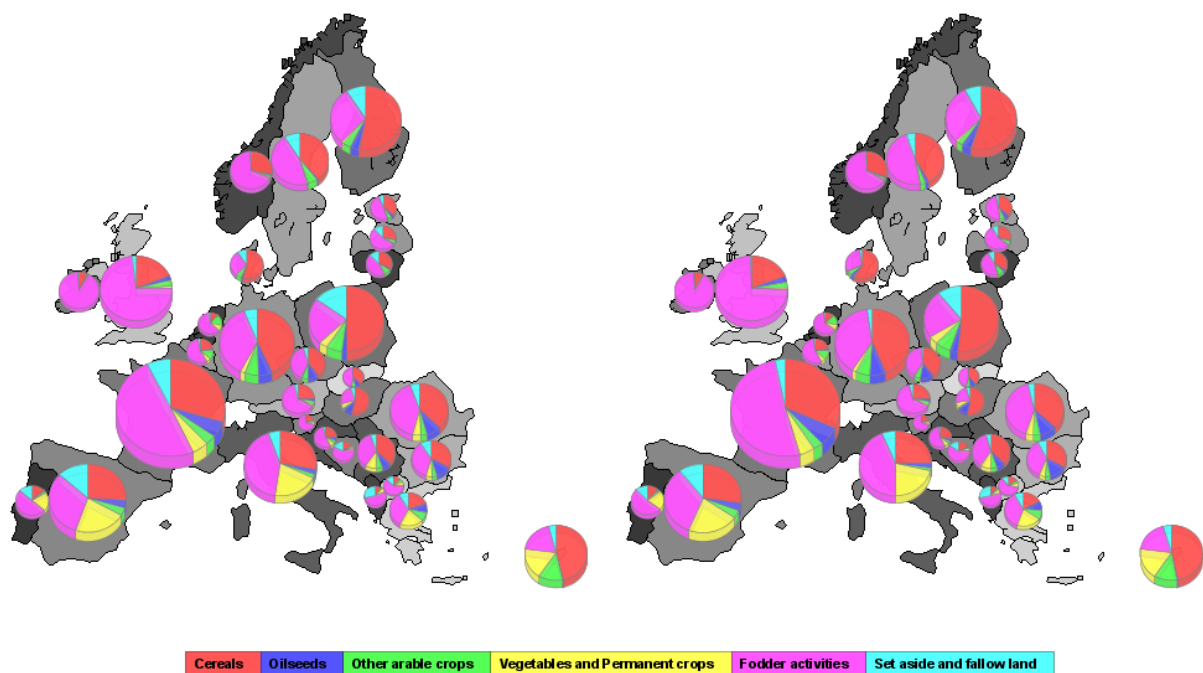
In order to show only a selection of the flows, the selection   buttons can be used.

The lower left one relates to the rows of the underlying tables, and thus allows excluding origins from the maps. The lower right one opens a dialogue to exclude destinations, whereas the upper right one allows exclusion of scenarios.

Most options described below for thematic maps such as zooming and dragging are also available for flow maps. However, classifications and color models cannot be supported.

Pie chart maps

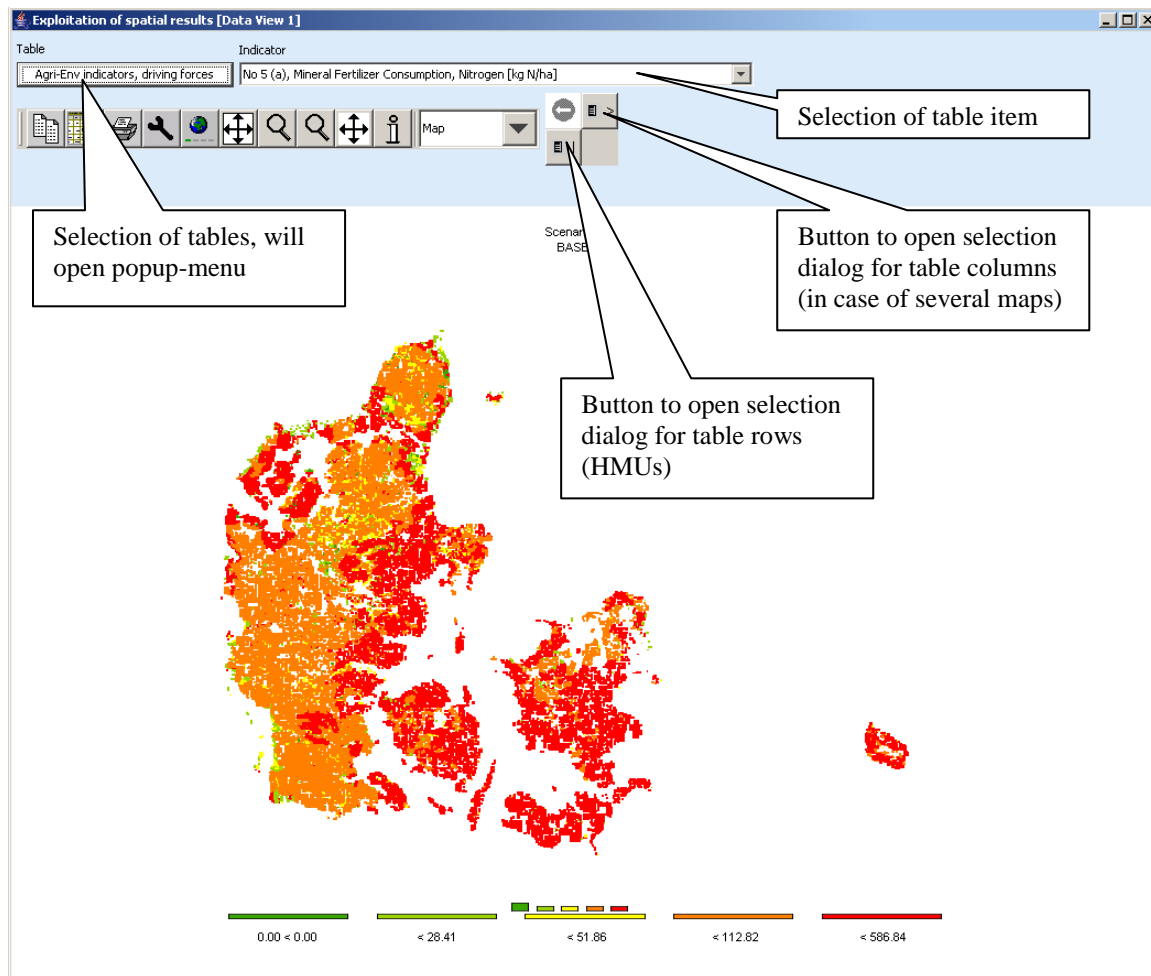
Another rarely used application of maps is the possibility to place pie charts above the geometry. The regions must – as always with maps – be placed in the rows of the underlying tables, and the cakes are calculated from the data in the columns. It is possible to produce maps for different scenarios when those are placed in the column groups as shown below. The size of the charts depends mainly on the bounds of the underlying polygon, so that smaller countries have smaller pies. The settings for pie chart diagrams (see Pie charts) can be applied to that view.



Colored thematic maps

For CAPRI, the GUI currently provides geometries for NUTS 2 regions, Member States, the regions with behavioral functions in the market model, trade blocks in the market model and finally, the Homogenous Soil Mapping Units (1x1 km resolution) underlying the spatial down-scaling component. The geometries are always linked to the rows of the underlying table.

The most obvious way to visualize results is the use of thematic maps. This holds true for NUTS2 results, but even more so for the results at the HSMU level. When starting the GUI, the mapping view uses some pre-sets which can be interactively changed as described below. The following screen-shot shows the result of first loading the base year results from the spatial dis-aggregation for Denmark and then switching from the tabular to the mapping view. As with other views, the content of the map can be changed by working with the drop-down boxes, or by (de)selecting columns and rows. There are specific possibilities to change class limits, colors and further features for maps which are discussed in the following.



Changing the classification and the legend

In order to change the layout of the map, click the mouse in the area of the legend or double-

click the map option button . The following dialogue will open.

Map option dialogue

Classification method: **Quantile** | Number of classes: **5**

Number of regions with small/large values excluded from classification: **0** | **0**

Treat zeros as missing values | Use area weights for classification

Draw in high quality

Emboss map (>0 gray, 0=none, <0 colored embossing): **0**

Color table: **Start color ..** **Mid color ..** **End color ..** **Green yellow**

Set value for middle color: **...**

#	label	class limit	% of obs	color	
1	-170.73 < 337.30		337,296	20,295	
2	< 422.00		421,998	19,926	
3	< 523.96		523,964	19,926	
4	< 654.04		654,04	19,926	
			1.733,805		

Show histogram | Show distribution circles for observations

Show distribution rectangles for classes | City labels: Min. city size (1000000.0)

Rivers: Min width **4** | Region labels: **Label options ..**

Legend: **Separate rectangles** | Draw outline in same color

Standard map title: **Standard map title**

Dimension shown in columns of result window for current region: **Scenarios**

Dimension shown in rows of result window for current region: **hide**

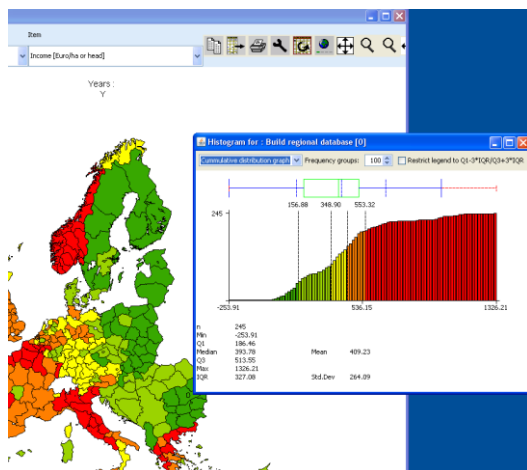
Resolution factor for printing/file output compared to screen: **1**

ok **apply** **store settings** **load settings**

It offers different options to change the way the map is drawn on screen and information supporting the classification.

Adding a histogram window to a map

In the map option dialogue, tick “Show histogram” and a separate window with a Histogram will be shown. It will use the current classification and color model to visualize the distribution of the values, reports some basic statistics and shows a box and whisker diagram.

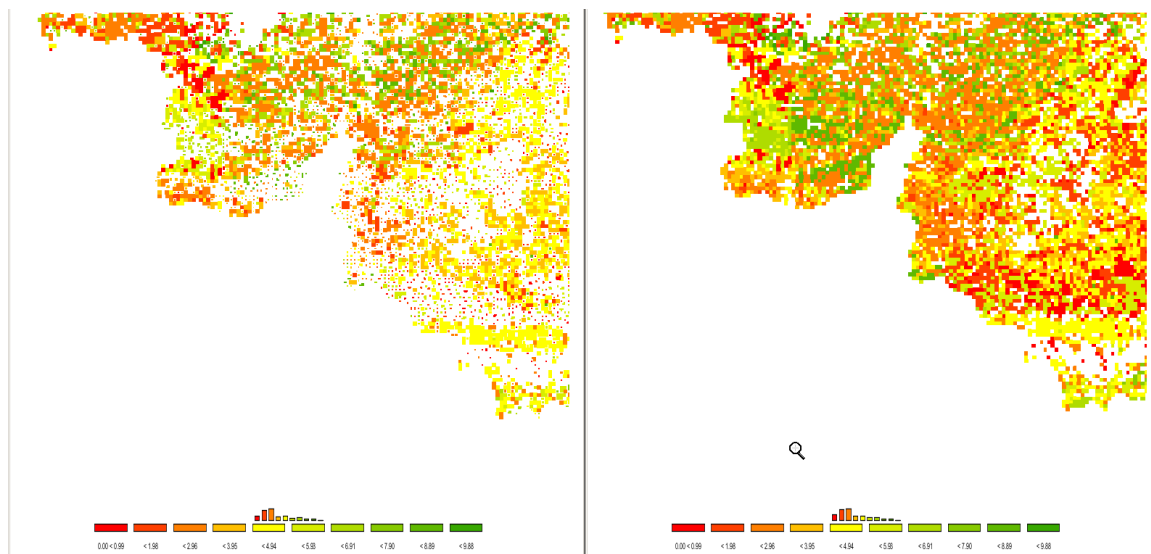


Shrinking polygons according to UAA share

The optical impression received of a map where colors are used to distinguish between values, depends to large extent on the area covered by a certain color. If the majority of the pixels is drawn in red, that will send a warning message to the user. In the case of the HSMUs and information relating to agriculture that message can be strongly biased as almost all HSMU comprise some other land cover than agriculture, and some of the HSMU comprise only very little agriculture, but e.g. forest, shrub lands, water bodies or artificial land cover. The HSMU geometry therefore comprises the information about the share of UAA assigned in the base year to each HSMU. That information can be used to shrink the area of the polygons when drawn on screen accordingly. That is done by drawing all points of the polygons towards the centroid of the polygon and then multiplying the distance between the point and the centroid with the square root of the share of the UAA. In the original HSMU geometry, such polygons had been broken down to simpler ones where the connection between a point and a centroid would cut through a segment of the polygon. In such cases, shrinking could let the new polygon hide other ones.

The graphs below show the very same map (same input data, classification and coloring) for the High Nature Value indicator for a part of Belgium. The right hand side map draws the HSMUs into their full size, the one on the left hand side one uses shrinking. The message perceived is probably very different. In the unshrunk right map, one may conclude that there is a lot of highly intensive agriculture (low HNV indicator drawn in red) in the lower diagonal triangle and some important areas of high nature farmland in the protruding area. This optical impression differs strongly from the polygons drawn with corrected shares for agricultural cover. It turns out that in the lower diagonal triangle, the density of agriculture is often low,

and especially low in the intensively managed HSMUs. Equally, it turns out, that the area covered by High Natural Farmland in the protruding part is relatively small.



Area weighted classification

The classification can be generally applied treating each “region” (a NUTS II or a HSMU) as an observation with equal weight or using the areas of the underlying polygons as weights. Those weights are multiplied with the share of UAA if shrinking is used as explained above.

Excluding zeros from classification and removing small and large values

In GAMS, zeros and missing values cannot be distinguished. For certain results, zero results are therefore coded as very small numbers to allow for that distinction. Zero observation can be excluded from classification and the polygons with zero observations will not be filled. Equally, a number of regions with small and large values can be excluded from classification.

Classification method

A first important feature is called “classification method” and defines how internally the class limits are set. For all types of automatic classification methods a clean-up procedure is used which removes classes with identical limits. It is generally recommended to use a number of classes which can be easily identified by the user, and to consult the frequency or cumulative distribution graphs present in the map option dialogue to check to what extent the class limits chosen represent the data well.

The following classification methods are currently supported:

Natural breaks

Natural breaks classification is a method to cluster the data into classes so that differences between the means of the classes become high while the standard deviation inside the classes becomes low (FISHER, W. D. (1958). "ON GROUPING FOR MAXIMAL HOMOGENEITY," JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION 53, 789-798. Code based on: HARTIGAN, J. A. (1975). CLUSTERING ALGORITHMS, JOHN WILEY & SONS, INC., NEW YORK. PAGES 130-142.). The algorithm does not only find the approximate best solution, but often gives rather appealing class limit definitions.

It works rather well if no extreme outliers are present in the distribution. In the latter case, classes solely comprising the outliers will be generated, and the vast majority of the values will be put in one or two classes.

The clustering algorithm is rather expensive to calculate, so that in cases in which the population exceeds 500 observations a somewhat simplified version is implemented in the CAPRI GUI. From the original observations, a “condensed” population is generated whose members represented means of consecutive observations of the original one. The members are set so that the number of observations from which the mean is calculated is not bigger than 1/500 of the original population size and that the spread of those observations is smaller than the minimum of 1/500 of the spread of the total population and 10% of the standard deviation. The actual calculations are then done taking the size of the resulting classes into account.

Quantile

The observations of the regions are split in a way so that approximately the same number of observations fall into each class. Quantiles are cheap to calculate and are therefore the default setting, and often appealing as colors occupy similar areas in the maps as long as the polygons have approximately the same size.

If unique values are found at the end of a quantile, the algorithm will either exclude all observation with that unique value from the class or include all of them. The decision will be based on the fact if with or without inclusion the size of the class comes closer to the desired size. If the user has e.g. chosen five classes, the desired class size should cover 20% of the observations or area weights.

Equal interval

The differences between the current minimum and maximum value is divided into classes of equal spread. This may lead to rather curious class limits when outliers are present. In those

cases, it may be appropriate to exclude some regions from the classification. See below for details how to exclude regions from the classification.

Mean standard dev

The class limits are defined according to the mean and the portions of the standard deviation of the data. It works best with normally distributed data, but may result in very small classes if the distribution is skewed, e.g. long tailed. The algorithm will always introduce at least four classes, then six, eight, ten and twelve. More than twelve classes are neglected.

The algorithm takes into account the spread of the data, and sets the class limits accordingly. If all observations fall into +/-25% of a standard deviation, class limits are introduced at 25% and 10% for four classes. If the number of classes is higher, new limits are introduced at 5%, 2.5%, 1% and 0.5%. In case of +/-50%, the smallest class is dropped and +/-50% added, and so forth up to +/- 3 standard deviations.






Nested mean

The nested mean classification will only work with 2, 4 or 8 classes. The classes will be defined such that one break is found at the mean of the sample. The resulting two halves of population are then again divided by their mean to get four classes, and the resulting quarters divided by their means to define eight classes. This works well with rather skewed distributions.

Manual classification

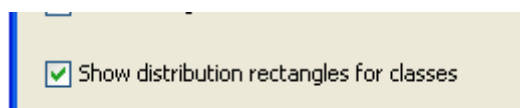
Finally, the user may set the class limits by hand. In order to do so, double click the mouse on the appropriate row in the table with the classification results in the column “class limit”. The value can now be changed with the keyboard. When this is done, click into another cell. The labels will be adjusted accordingly. Afterwards, when all class limits are defined, the user may also overwrite the label (e.g. using words as “low” or “high”).

Please keep in mind that currently the values will be lost if you load other data or change the classification, number of classes etc..

#	label	class limit	% of obs	color
1	0.00 < 0.00	0	30.488	
2	< 28.41	28.407	17.398	
3	< 51.86	51.81	17.398	
4	< 112.82	112.821	17.398	
5	< 586.84	586.835	17.317	

Integration distribution information in the map window

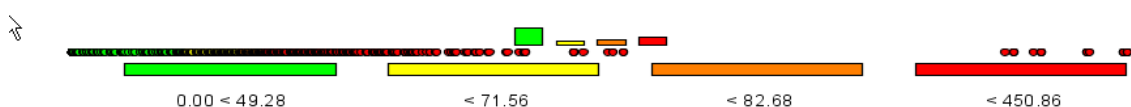
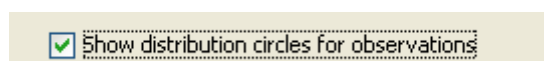
The GUI allows the user to enter distribution information in the map in different ways. The first possibility is to print a simple frequency diagram above the legend.



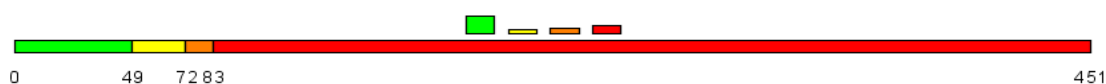
That gives a rather intuitive feel on how well the class limits represent the data distribution. In our example below, it is obvious, that the majority of the values lie in the first class.



Less suitable for final output, but useful while playing around with classification methods and class definition are the distribution dots which can be added. They carry additional information on the location of values in different classes.



Finally, switching to linear or logarithmic may be a way to help reading the map.



Color table

The color table defines the colors used for the classes. When choosing the color model, keep in mind that colors carry a meaning; red e.g. is generally interpreted as dangerous. Equally, it is important to think about the final medium with which the map will be published. Exporting colored maps to a black-white device will render it almost impossible to read the map. It is best to try different color tables and different classification methods on your data. The following color models are currently available, named according to the data order from minimal to maximal value:

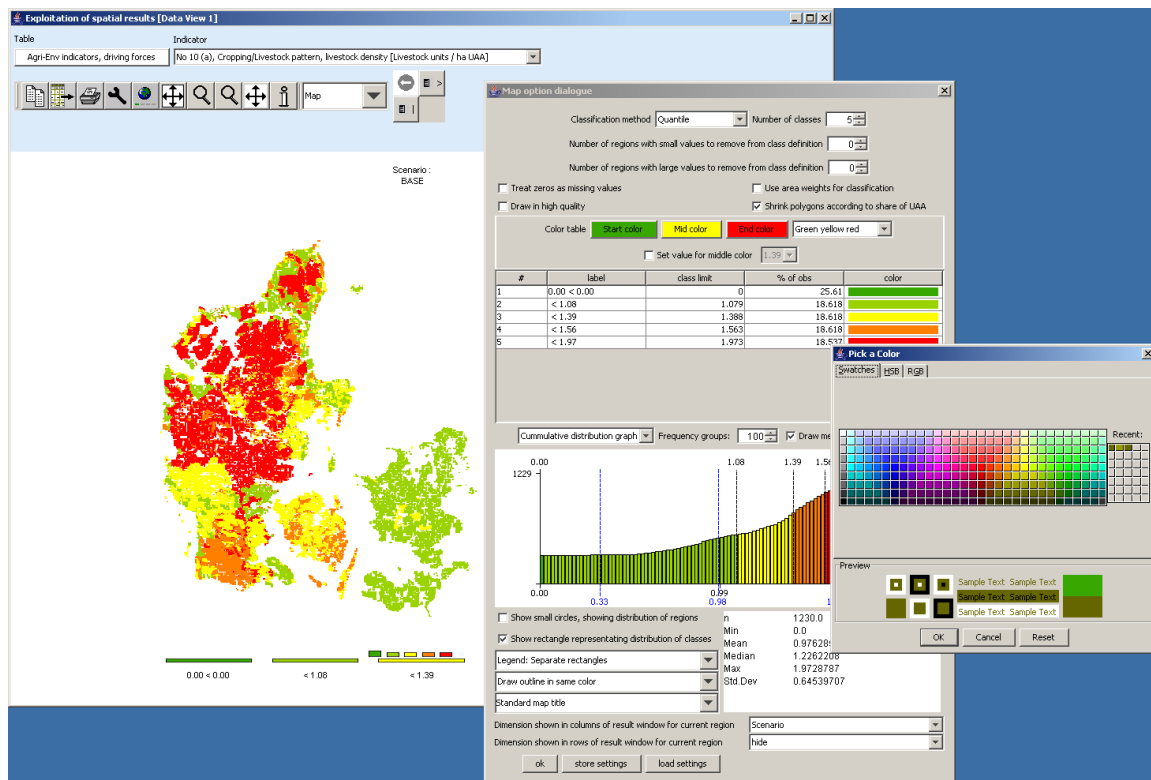
- *Green – Yellow – Red (standard)*: Normally, the middle class is drawn in yellow, smaller values in shades between yellow and green, and larger ones from green to red.

This should be applied e.g. to environmental indicators where the damage increases with the value of the indicator.

- *Red – Yellow – Green*: as above, only that high values are shown in green. Should be used e.g. for income indicators or environmental benefits.
- *Red – Gray - Green / Green – Gray – Red*: more available for historic reasons as they mimic the color tables of the original JAVA applet.
- *Blue – Gray - Green / Green – Gray – Blue*: introduced on demand of DG-AGRI. A good choice if the “good”/”bad” interpretation of the distribution is to be avoided.
- *Shades of grey*: sometimes needed for publications when color printing is not available in the final hardcopy. Beware to use a limited number of classes.
- *Shades of blue*: useful where the notion of “bad” or “good” inheritably comprised in greenish and reddish colors is to be avoided.

Defining an self-created color model

Once a color model is chosen, the user can re-define the start, middle and end color using the three buttons on the color table selection row, as shown below, given a lot of freedom to generate color ramps.



The screenshot shows the GGIG software interface. On the left, a map displays livestock density data for a region, with a legend below it showing three color-coded classes: 0.00 < 0.00 (dark green), < 1.08 (yellow-green), and < 1.39 (yellow). The 'Map option dialogue' window is open on the right, showing the following settings:

- Classification method: Quantile
- Number of classes: 5
- Number of regions with small values to remove from class definition: 0
- Number of regions with large values to remove from class definition: 0
- Treat zeros as missing values
- Use area weights for classification
- Draw in high quality
- Shrink polygons according to share of UAA
- Color table: Start color (dark green), Mid color (yellow), End color (red), Manual start/mid/end
- Set value for middle color: 1.39

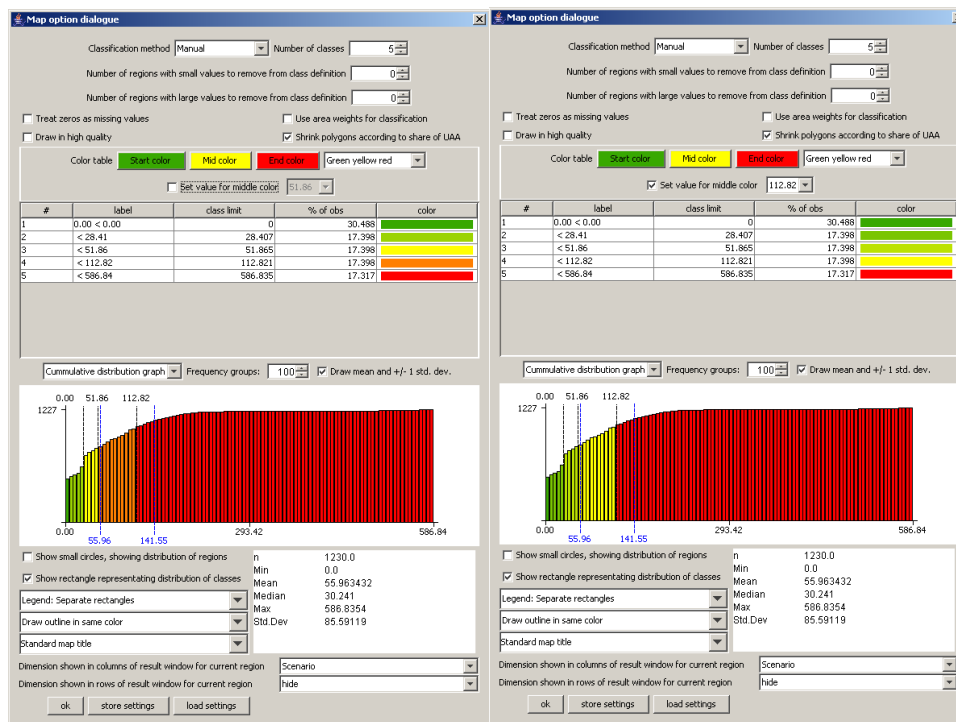
The window also displays a table of class limits and a histogram of the data distribution.

#	label	class limit	% of obs	color
1	0.00 < 0.00	0	25.61	Dark Green
2	< 1.08	1.079	18.618	Yellow-Green
3	< 1.39	1.388	18.618	Yellow
4	< 1.56	1.563	18.618	Orange
5	< 1.97	1.973	18.537	Red

The histogram shows the cumulative distribution of the data, with vertical lines indicating the class limits. The legend below the histogram shows the color coding for the classes: 0.00 < 0.00 (dark green), < 1.08 (yellow-green), and < 1.39 (yellow).

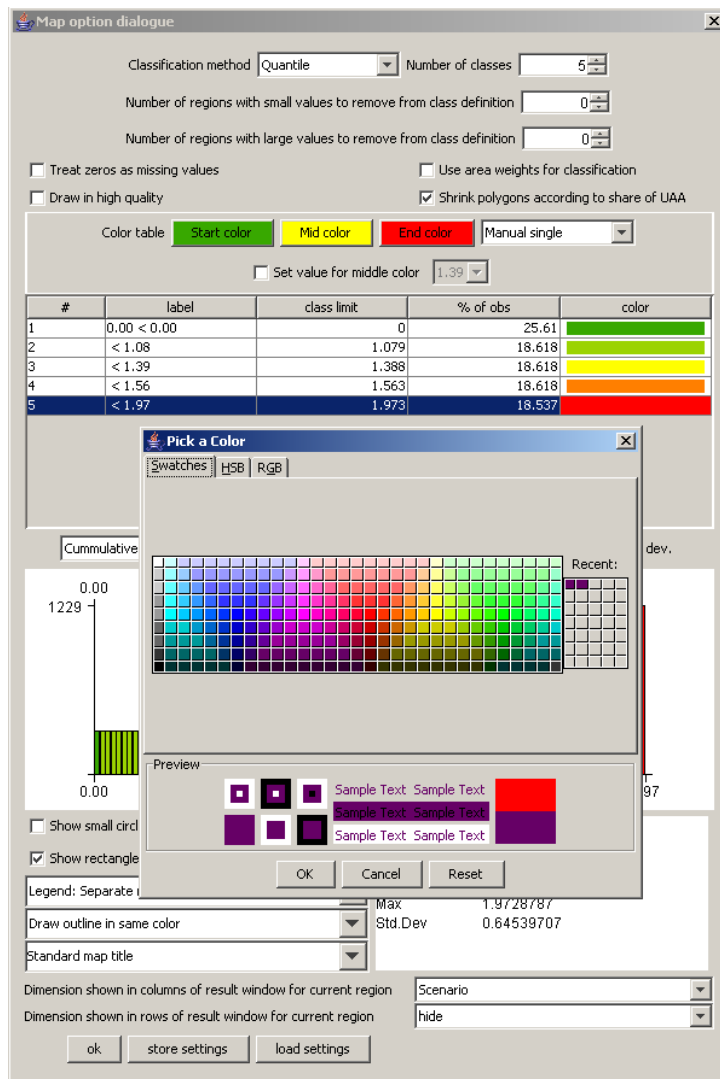
Changing the value for the medium color

Normally, the medium color (yellow or gray) is assigned to the middle class. Sometimes, the user may wish to change the class where the color switches. First, the “Set value for color change” must be ticked. Next, in the now enabled drop-down box, choose the class limit for which the middle color should be used. The effect is shown below. Before, values in the class below “392.70” – the middle class – were drawn in yellow. When the user now selects another class limit, the colors assigned to the classes change. Here one of the shades of green is dropped and shades of red are added.



Manual set colors

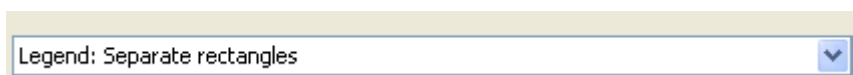
Finally, the user can choose its own colors by double clicking in a color field in the legend table. That should only be done after the final definition of the class limits is set as otherwise, the manually set color will be lost.



Changing the way the legend is drawn

The map viewer always puts the legend below the map. Currently it offers three options how legends are drawn:

1. **Separate, equally sized rectangles** which show the upper class limit with the exemption of the lowest class, which shows the lower limit.

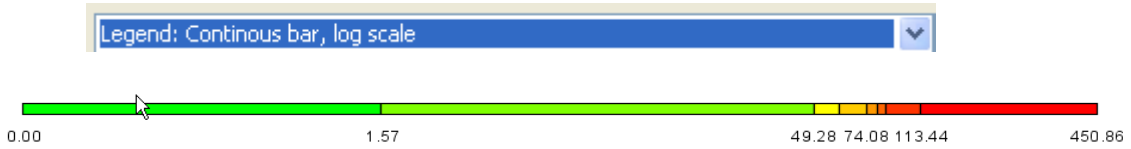


2. **A continuous linear scaling bar.** That gives an optical idea about the distribution of the class limits. Overlapping of the number is avoided by skipping class limits close to

each other



3. A continuous logarithmic scaling bar



In all the cases, the tool dialogue can be used to set number of digits shown, e.g. reducing the number of digits to zero leads to a linear bar as shown below:





The reader is reminded that the label can be changed manually as shown below.

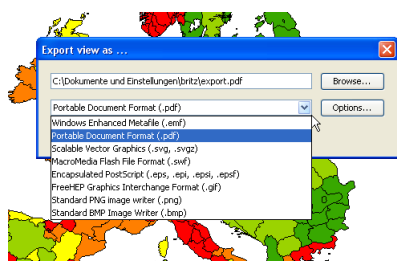
#	label	class limit	color
1	low	345.891	yellow
2	high	1,392.846	red

Statistics:

- n: 249.0
- Min: -418.85323
- Mean: 358.84164
- Median: 345.89148
- Max: 1392.846
- Std. Dev: 4468.262

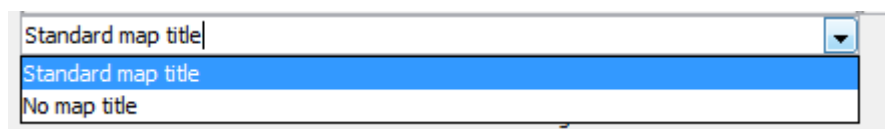
Copying the map to the clipboard or saving to disk

In order to export the map to other applications, the easiest way is to use the clipboard, in order to do so, press the “copy to clipboard”  button. Afterwards, the current map can be imported into other applications as e.g. MS Word. Another possibility is to store the current map in jpeg format on disk, to do so, use the “export”  button which will open a file dialog to choose the name of the file and select between different graphic formats. For MS Office users, the “Windows Enhanced Metafile (.emf)” format is especially interesting as it allows to change the graphic afterwards, e.g. by moving the legend or changing the text.




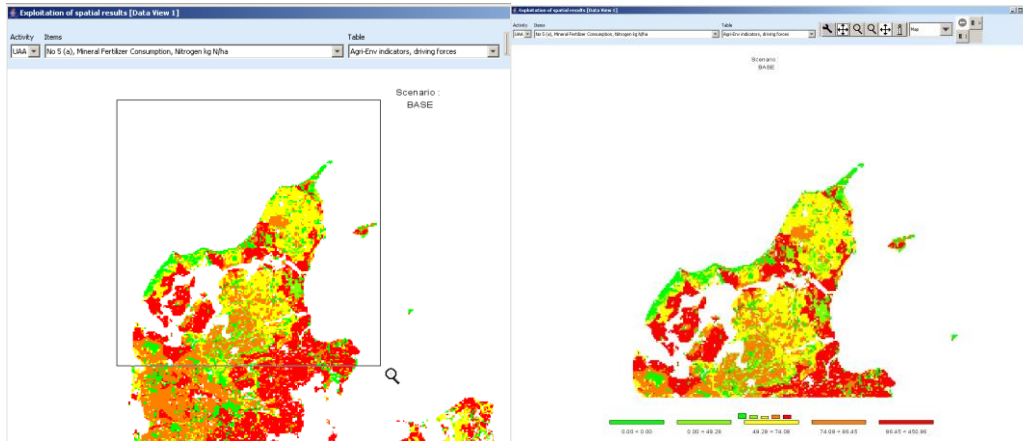
Changing the title of the map




When using output to clipboard or disk, the user may often prefer to choose his own title or no title at all on top of the map. This will be helpful when producing a caption for the map in another application. In order to refrain from drawing a title on top of the map, click into the legend part of the map, and in the dialog at the bottom, choose “none” in the row labeled “Title on top of map”. Alternatively, the user can simply write something in the box.




Zooming in and out and navigating in the map

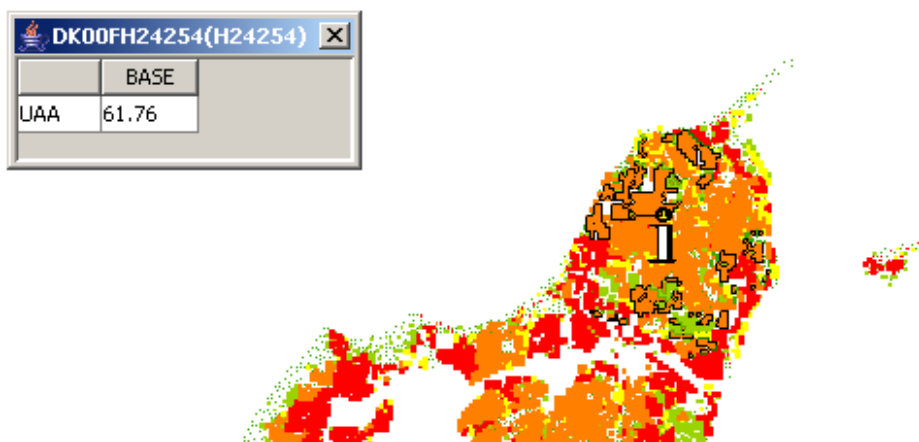
In order to zoom in part of the map, press the  button. The mouse pointer will change to a magnifying glass with a cross in it. You can then mark an area on the map by pressing the mouse button, dragging and then releasing the mouse. After the mouse is released, solely the selected zone of the map will be drawn, without changing the class limits or any other setting. Clicking with the mouse while being in zoom in mode will increase the map resolution step-wise by 25% and center the map at the current mouse position.




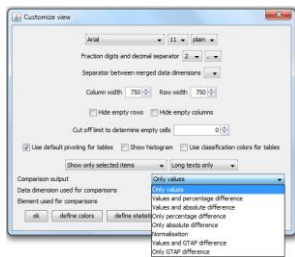
By clicking with the zoom out pointer  on a point of the map, the point becomes the new center point of the map and the map resolution is reduced stepwise by 25%. Equally, you may drag the map while keeping the current resolution by choosing the drag pointer . Finally, in order to return to the original full-sized map, use the “full extent” button . The reader should note that the “full extent” button shows a rectangle around the arrows.

Getting data for specific polygons

The info pointer  will open an additional window, as shown below, which displays information on the current polygon – the circle above the “i” being the focus point. The title bar of the new window shows the code and, if available, the long text of the polygon currently pointed to with the info pointer. The content of the info window is continuously updated when the mouse is moved over the map, and all polygons belonging to the same region as the one pointed on with the mouse is highlighted.

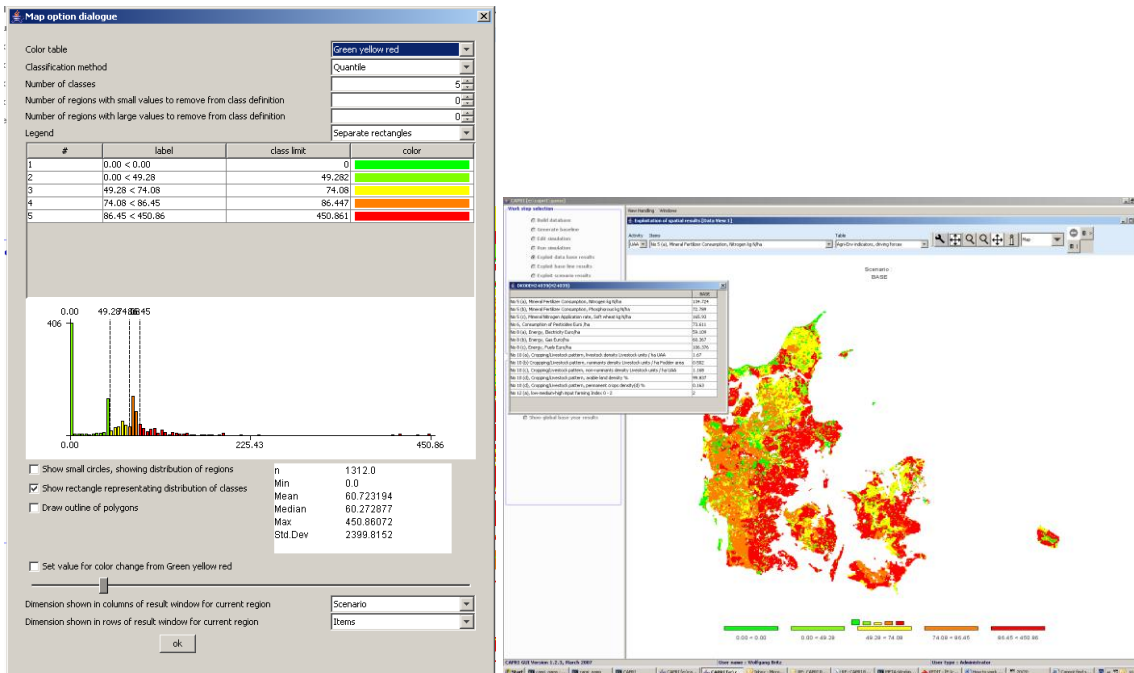


If the user opts to use one of the comparison options to be shown (percentages, differences, normalization) by clicking on the “customize”  button,




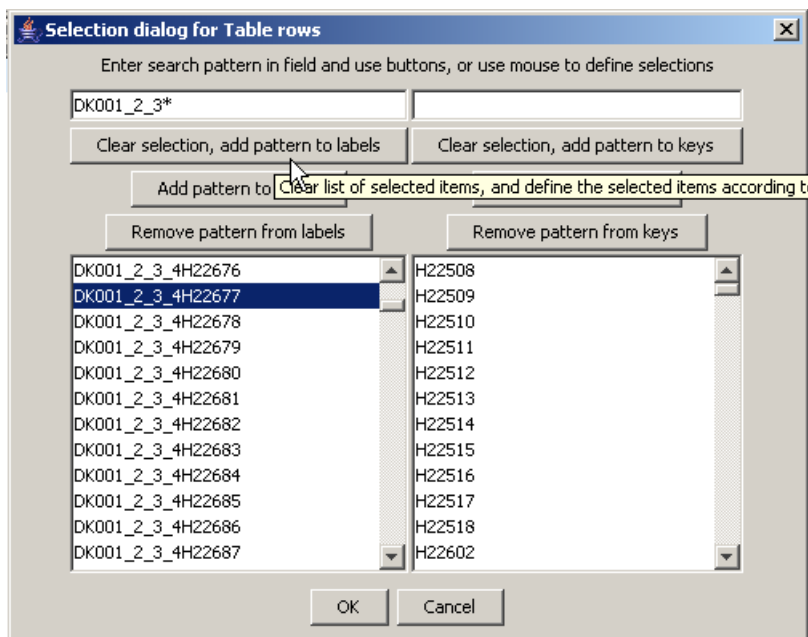
an additional column is automatically added to the info window showing the comparison value used. That is especially helpful when the map shows only differences.

The content shown in the info window is not fixed, rather, the user can decide which data dimensions to use for the columns and rows by using the “map option dialogue” by clicking on the legend of the map. If the user e.g. switches to “items” instead of “activity”, the “info” window will look like shown below. An alternative is to use a second tabular view in addition to the map.

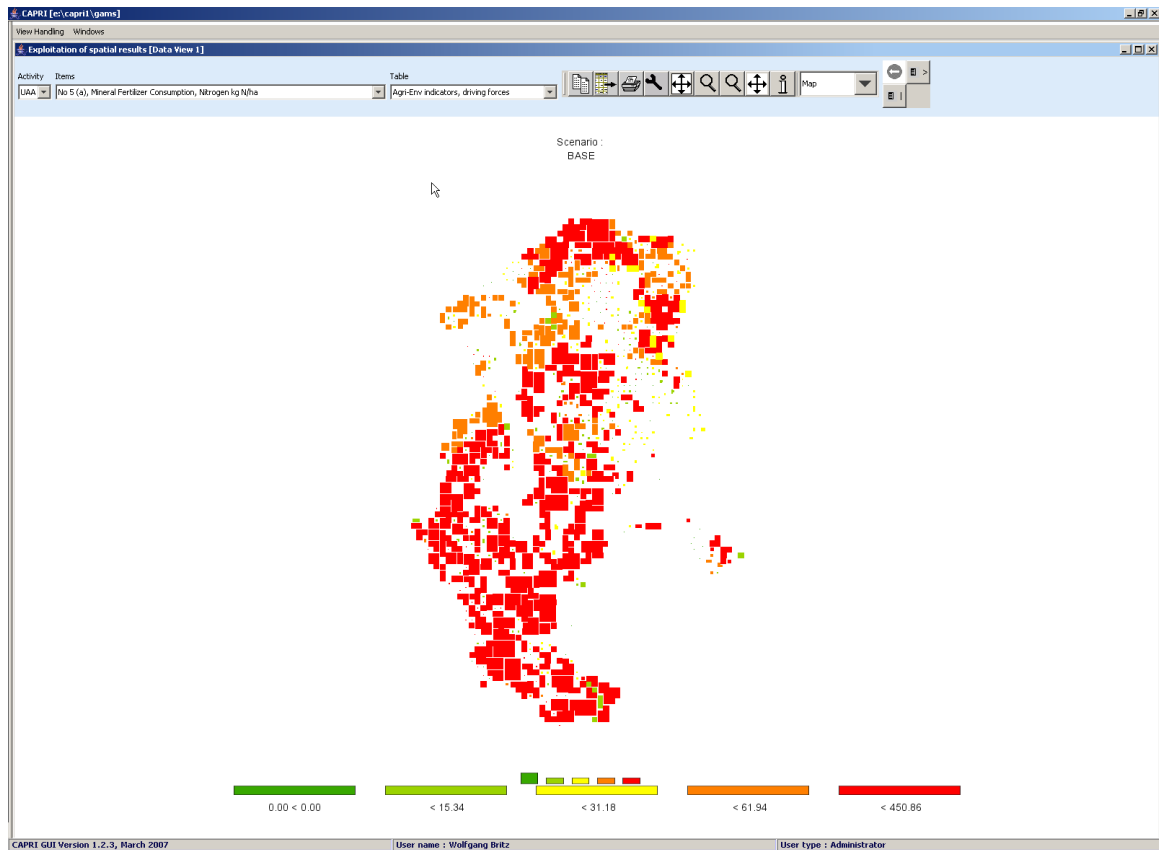


Highlighting specific regions in the map

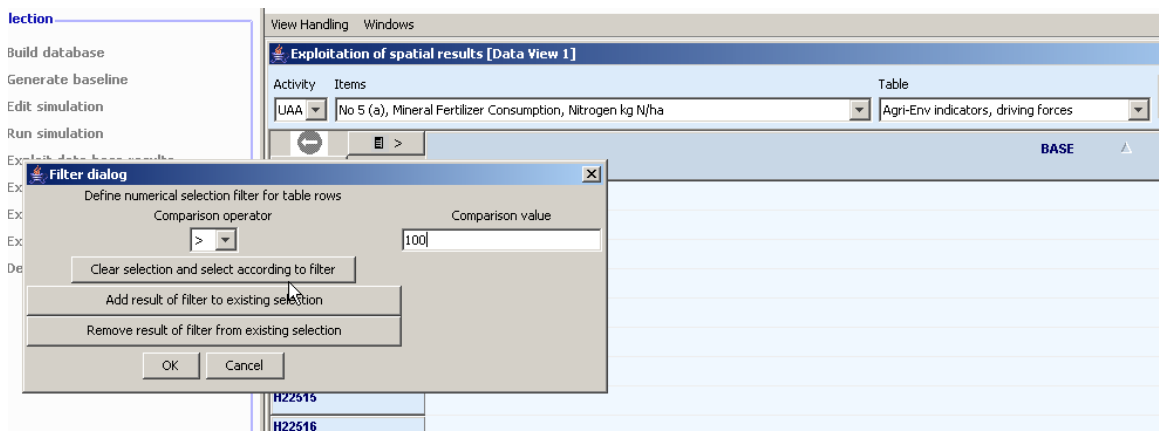
Sometimes it may be interesting to see the spatial distribution of specific data or data constellations. All views open the possibility to (de)select columns and rows, allowing e.g. to use the NUTS code in front of the numerical HSMU code to select only the HSMU belonging to specific administrative regions. That possibility is explained in short. First, double-click the row selection button  (“Open selection dialog for table rows”) which will open the following dialogue.



Now, we may e.g. select only the HSMU belonging to the FSS region DK000_1_2_3 by typing “DK001_2_3” in the left input box, and then choosing “Clear selection, add pattern to labels”. Afterwards, the map will look as shown below.

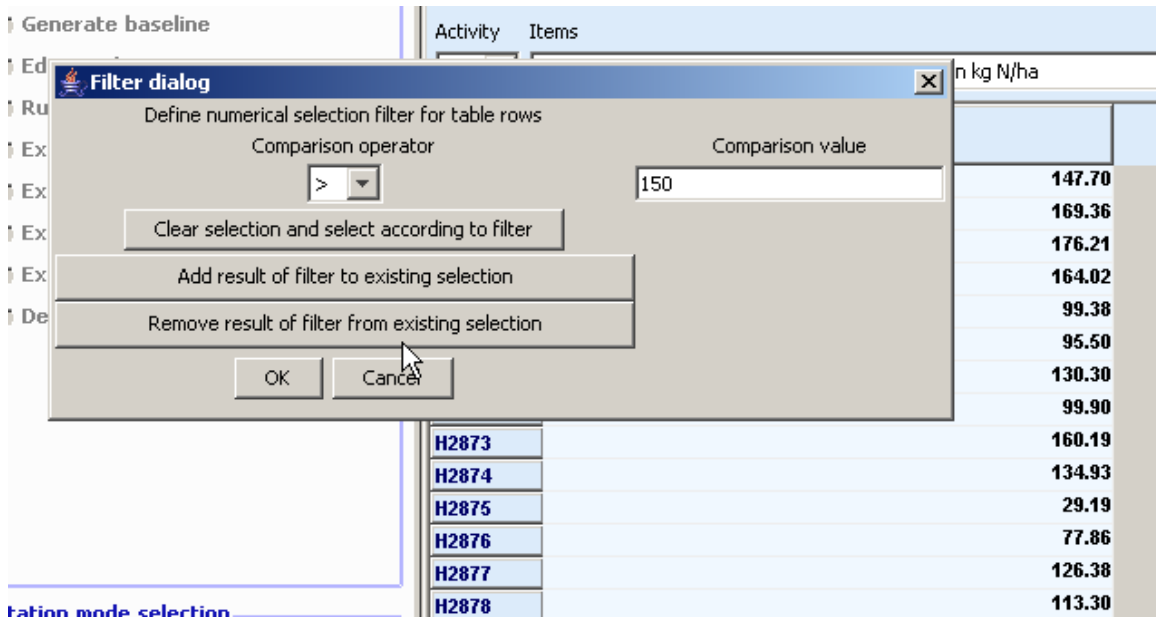


The tabular view opens up the possibility of using numeric filters, an option discussed in the following. Take for example the task to select all regions where the Nitrogen Fertilizer Consumption is between 100 and 150 kg/ha. First, switch from map to tabular view. In the table click with the right mouse button in the column header of that column holding the values to which the filter should be applied, as shown below. We will need to apply the filter step-wise, first e.g. selecting all values greater than 100 and then removing those which are above 150.

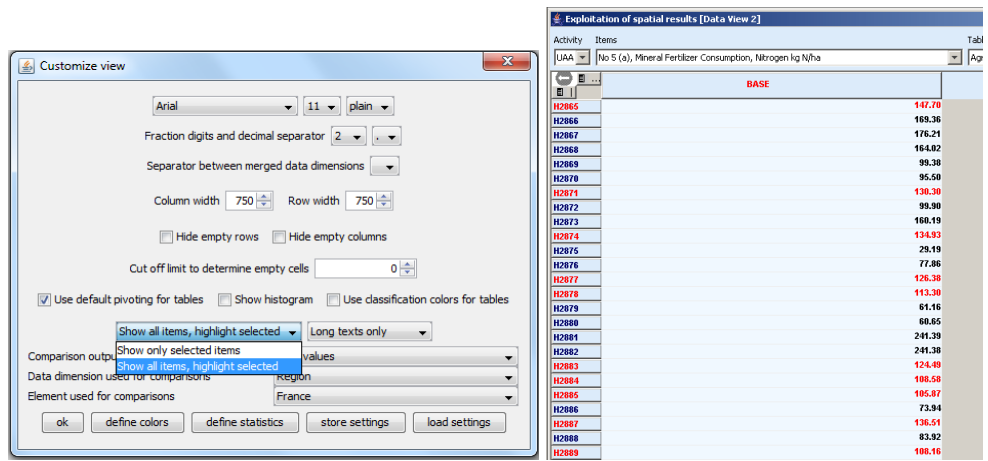


After clicking on “clear selection and select according to filter”, and then on “ok”, the table will only show such regions where the value in the column “BASE” is above 100, as shown

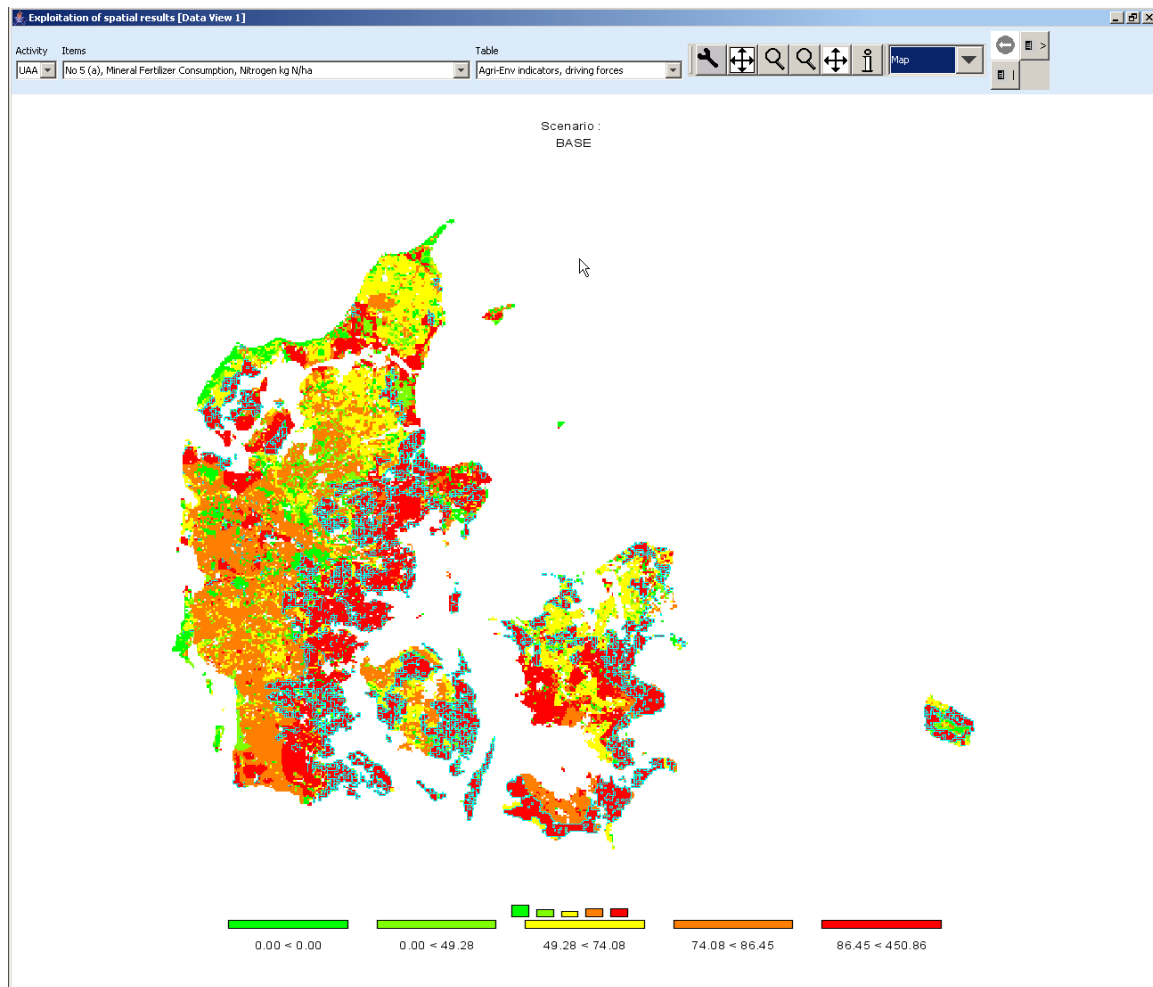
below. Next, we must exclude the regions above 150 kg/ha. To do so, set the filter to “>” “150”m and then press “remove result of filter from existing selection”.



Now, drawing a map with just those regions is not so interesting. However, with the tool dialogue, we can highlight the selected value instead of hiding all others. The selected rows are now shown in red in the tabular view.



When we now draw the outlines of the selected polygons only (see map option dialogue), the map will draw the outline of the selected regions in cyan and thus highlight them. The row selection will be maintained when the pivot or the table is changed, as long as one of the selected items can be found in the rows of the new table. The example map shown below is certainly not so interesting, as changed class limits could have done basically the same job. However, we could switch e.g. to grass land shares to see if fertilizer input is more often found on arable or on grass land.

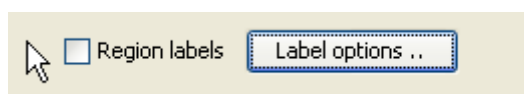


Updating the map

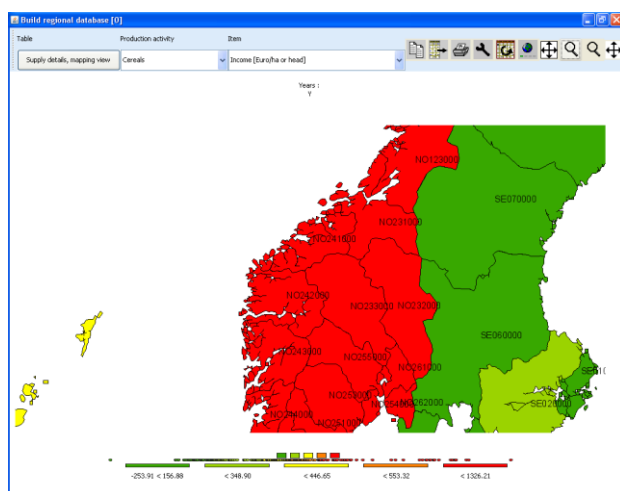
Generally, the map is updated automatically when the user changes an option with an impact on its layout, as long as the number of visible polygons is below 20.000. If that amount is exceeded, the classification dialogue is updated immediately, but not the underlying map. In order to apply the changes, the “apply” button must be clicked on. The user is informed that the “ok” button will also update the map, so that an “apply” immediately before an “ok” is not necessary.

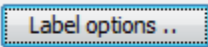
Adding region label to the map

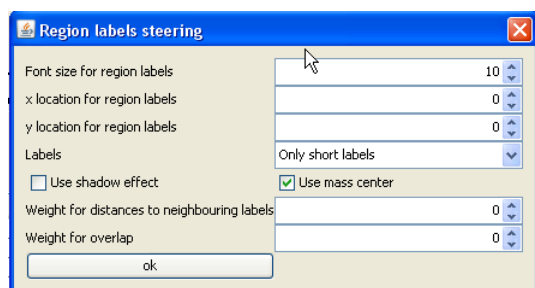
In the map option dialogue, tick the box “Show regions labels in map”



to add labels to the largest polygon for each region as shown below.

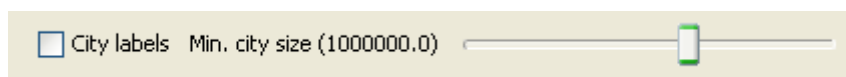
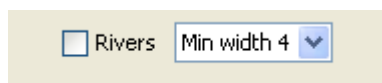


By clicking on the  button, the “Region label steering” dialogue box opens which allows changing some settings. For maps with just a few regions (or when zooming), it might be worthwhile trying to play around with the action to improve labeling.

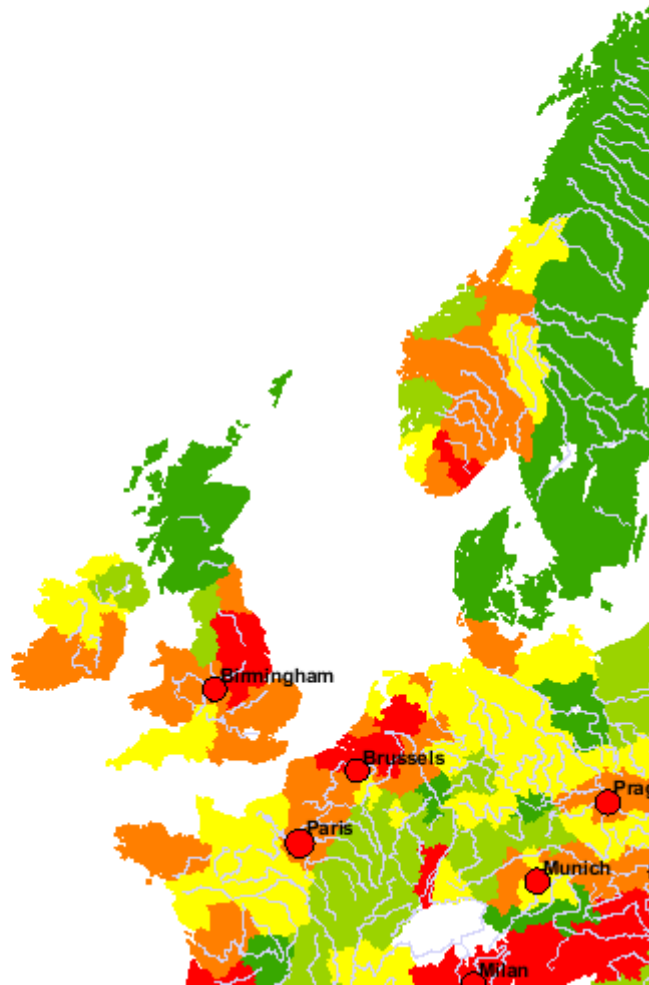


Showing river and cities


The NUTS2 map comprises geometry information about major rivers and cities above around 75.000 inhabitants, which can be added to the map:

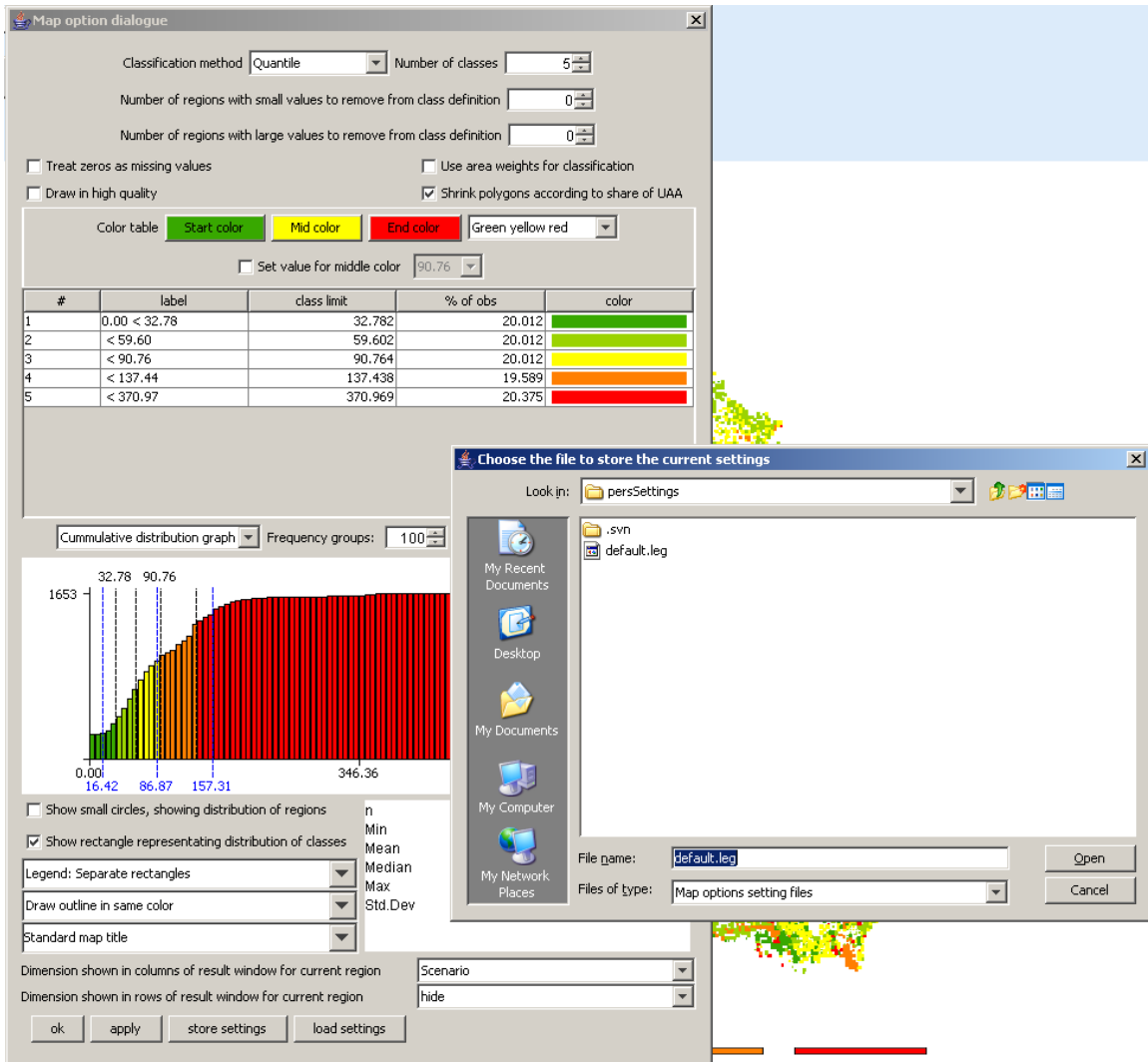


The label size for the rivers can be set as discussed above, however city labels are always shown in bold.




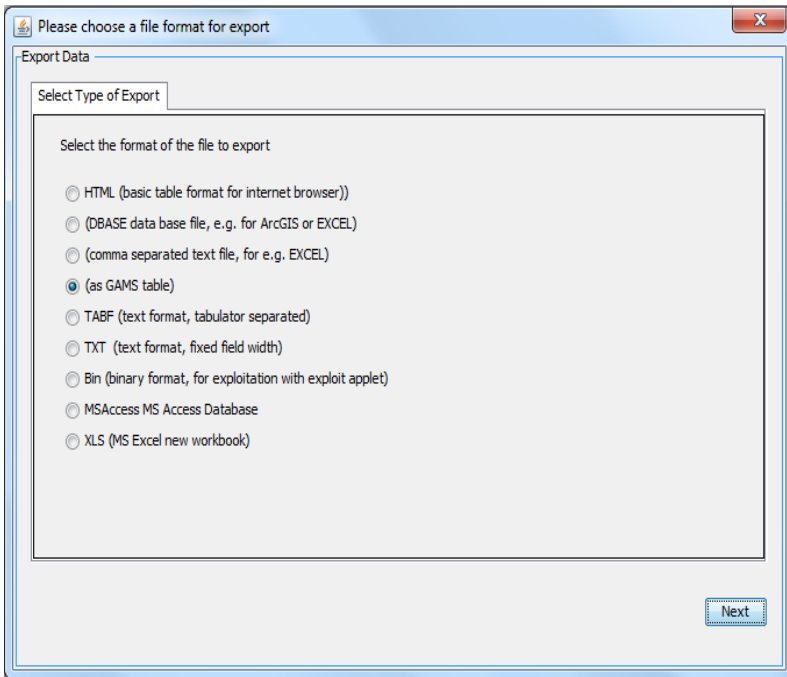
Storing and re-loading your settings

Open the map option dialogue by pressing the map option button “”. Change the settings according to your needs and then press the “store settings” button in the lower part of the dialogue. Choose a file name and a location. You may later use “load settings” to retrieve them again and apply them to another map.

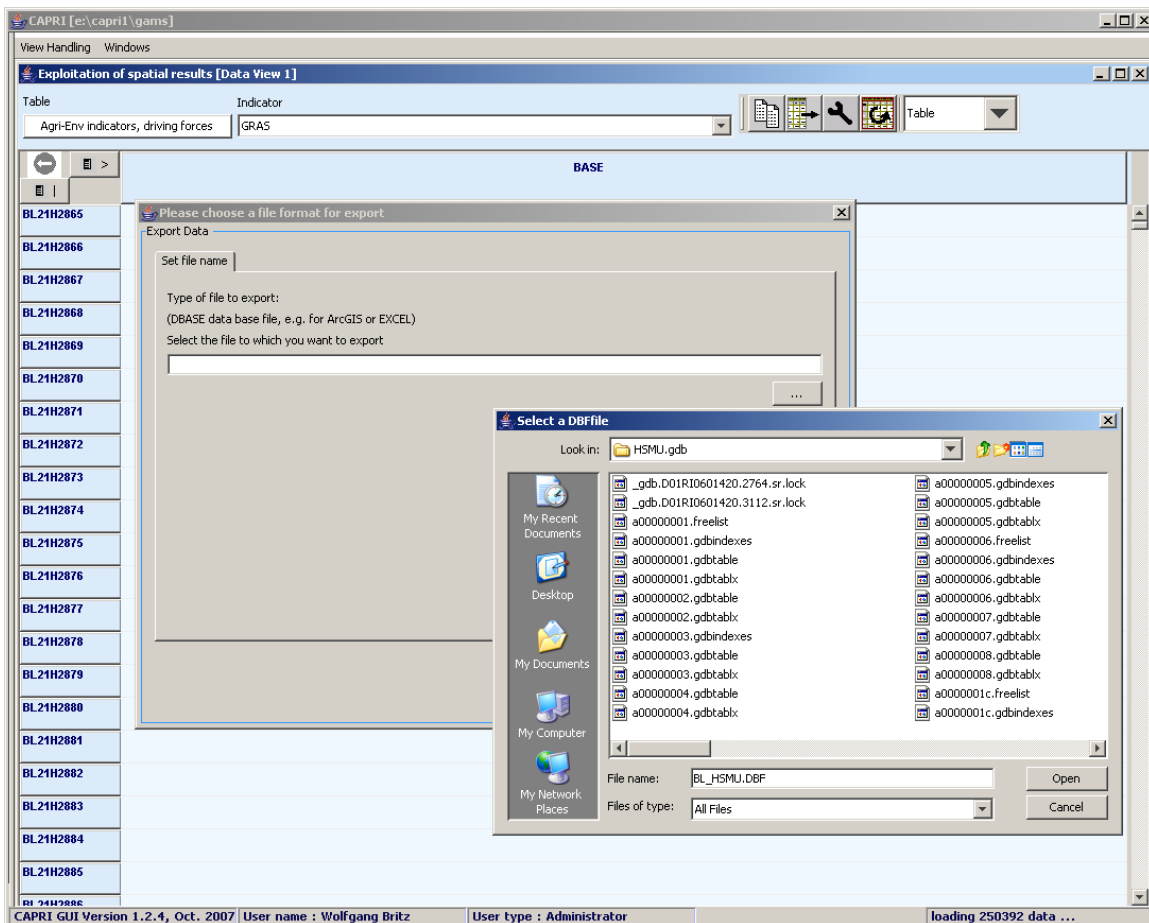


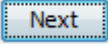
Exporting the data underlying the map

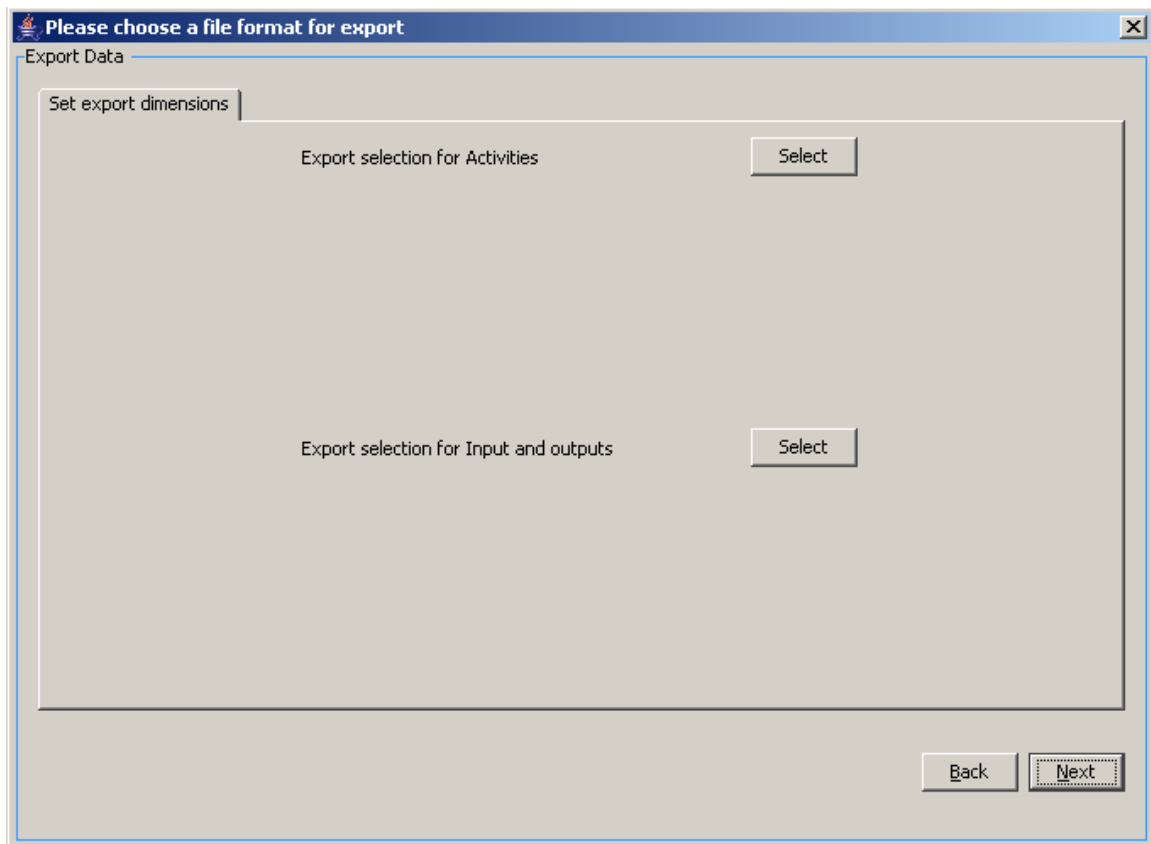
As mentioned above, the mapping viewer is part of the CAPRI exploitation tools which is in its core based on pivot tables. In order to export the data, e.g. to GIS system, the view must first be changed to tables. Afterwards, the  button will open a file dialog as shown below. For GIS-export, e.g. to ArcGIS, DBF is the recommended format.

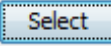


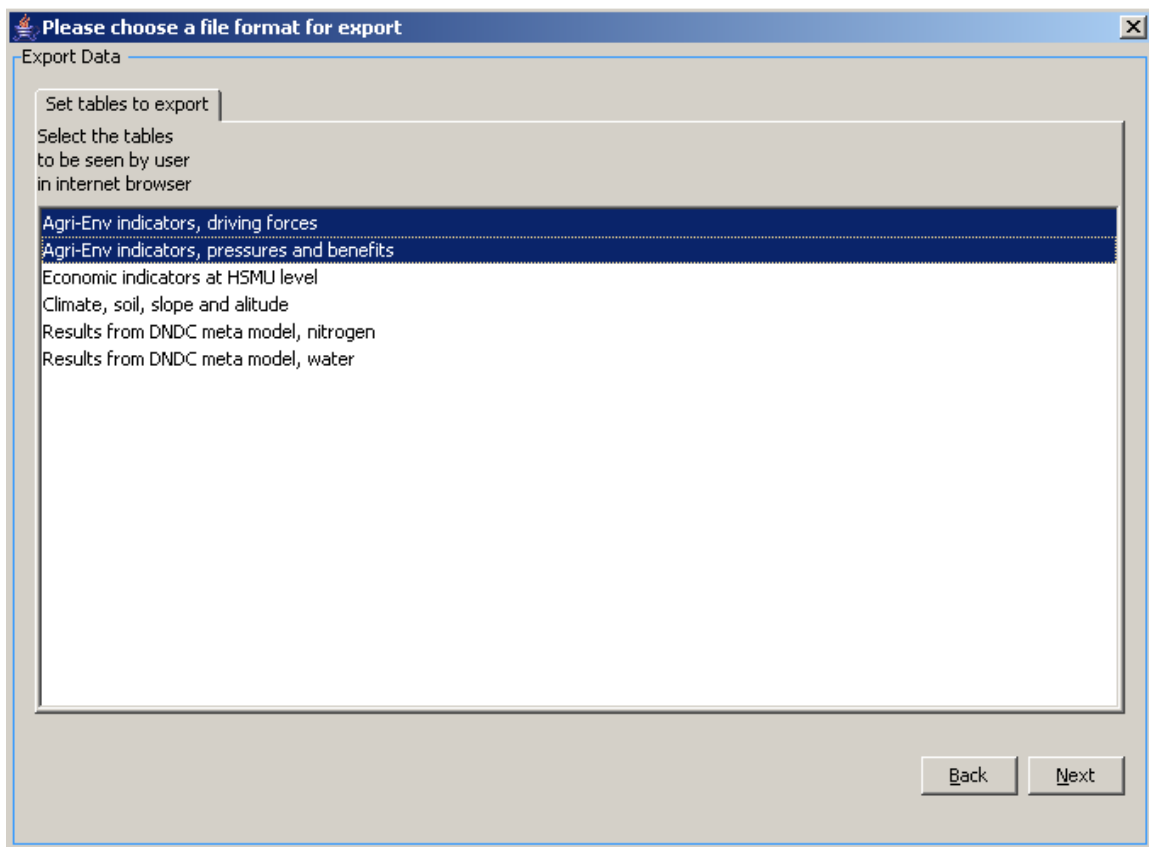
Once next is pressed, the next pane will open a file dialog to choose a file. In the case of export to a Microsoft Access Data Base, the file must exist.




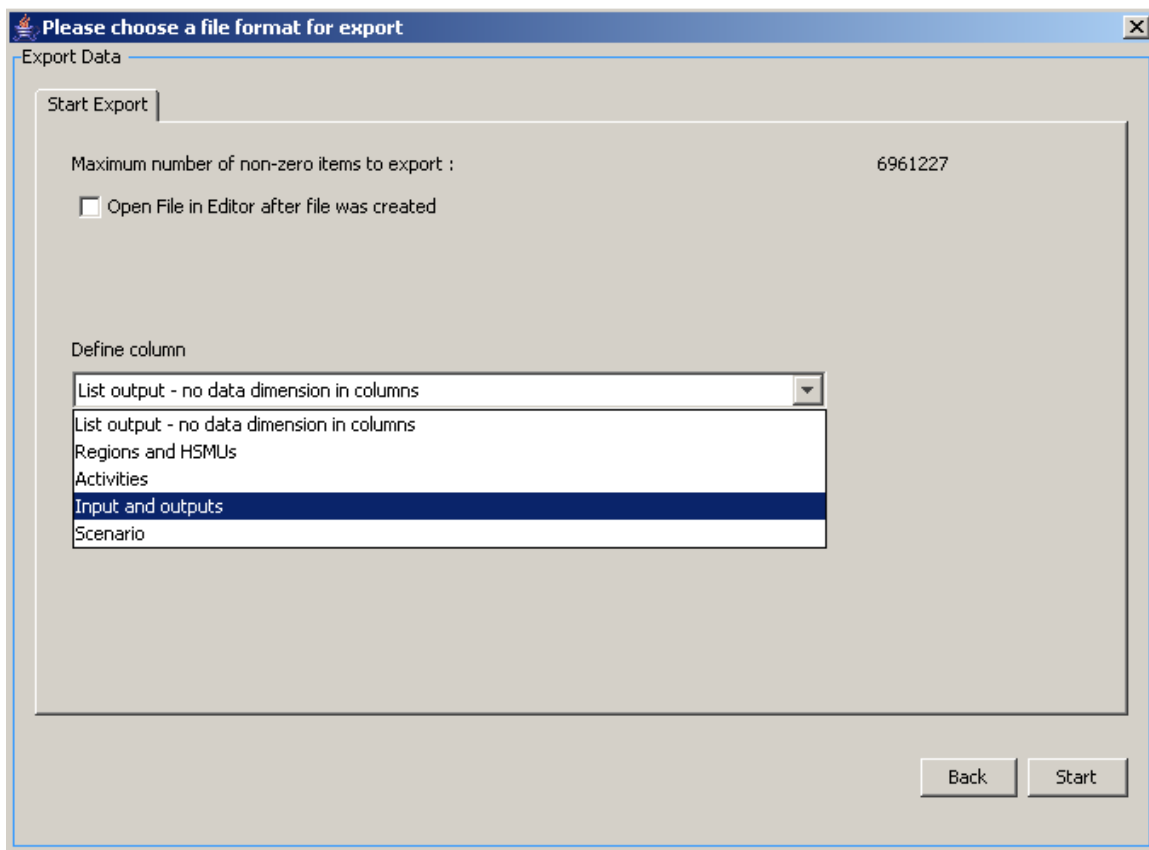
If desired, the  pane allows opening selection lists for the different data dimensions.



You can  next the tables for export.



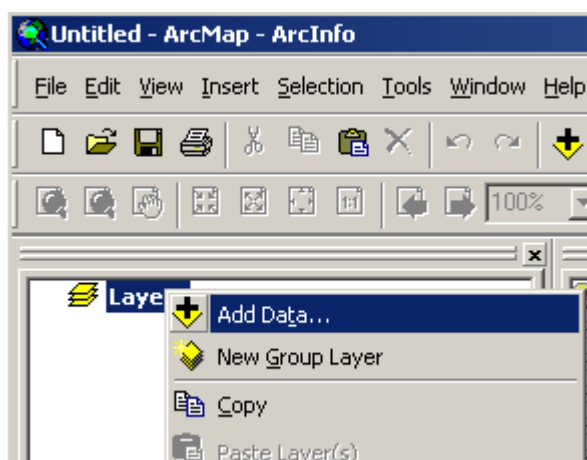
Beware: the pre-defined table structure will be lost, as will the long-texts and units attached to the tables. However, in the case of DBF-export, a second file with that information will be automatically created. If you solely want to export the table you have currently up front, use the “copy to clipboard”  button. The clipboard export will retain the pivoting and further information.



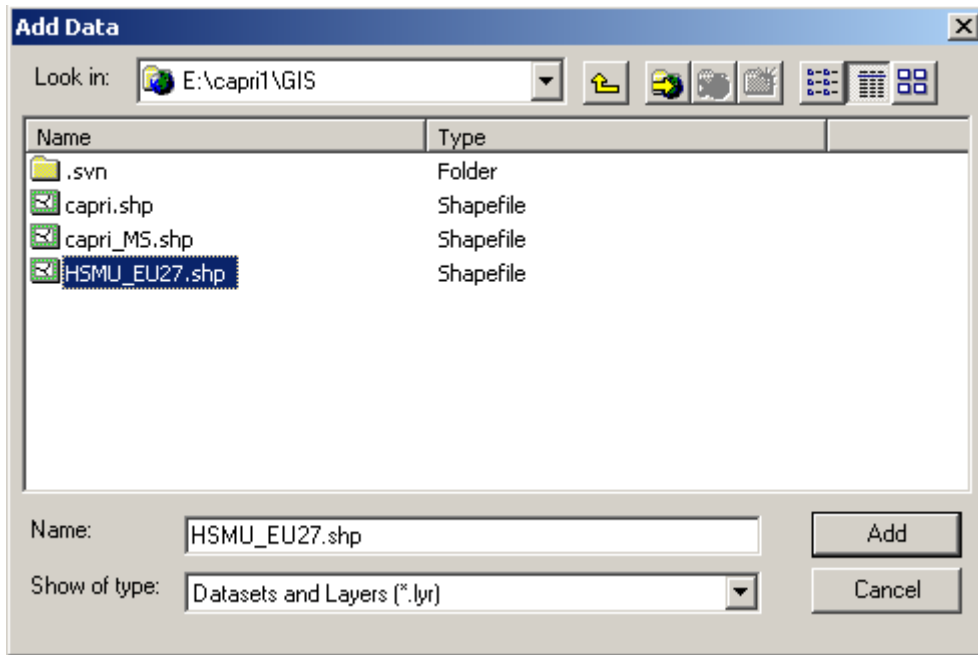
The last pane let you decide for DBF-export if you want a list, or if you want the data dimension spanned across the columns. For exporting the HSMU tables, it is recommended to put “Inputs and outputs” in the columns.

If everything has worked well, we should now find two files: one with the data, named as chosen in the file dialog, and a second one with “_meta” introduced before the file extension.

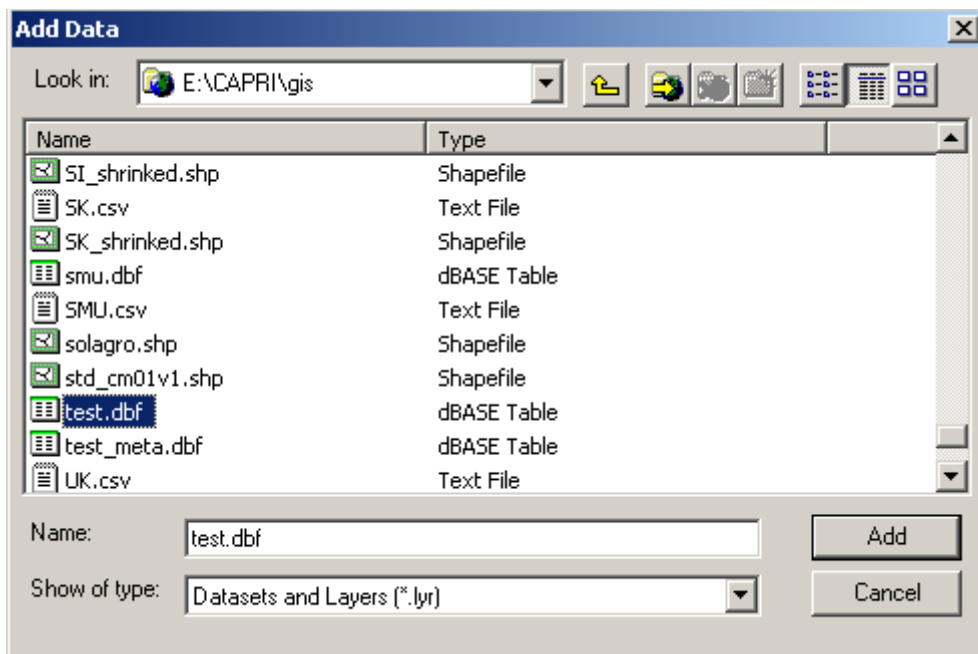
The following section will briefly explain how to work with the data in ArcGIS. Under Layers, choose add Data



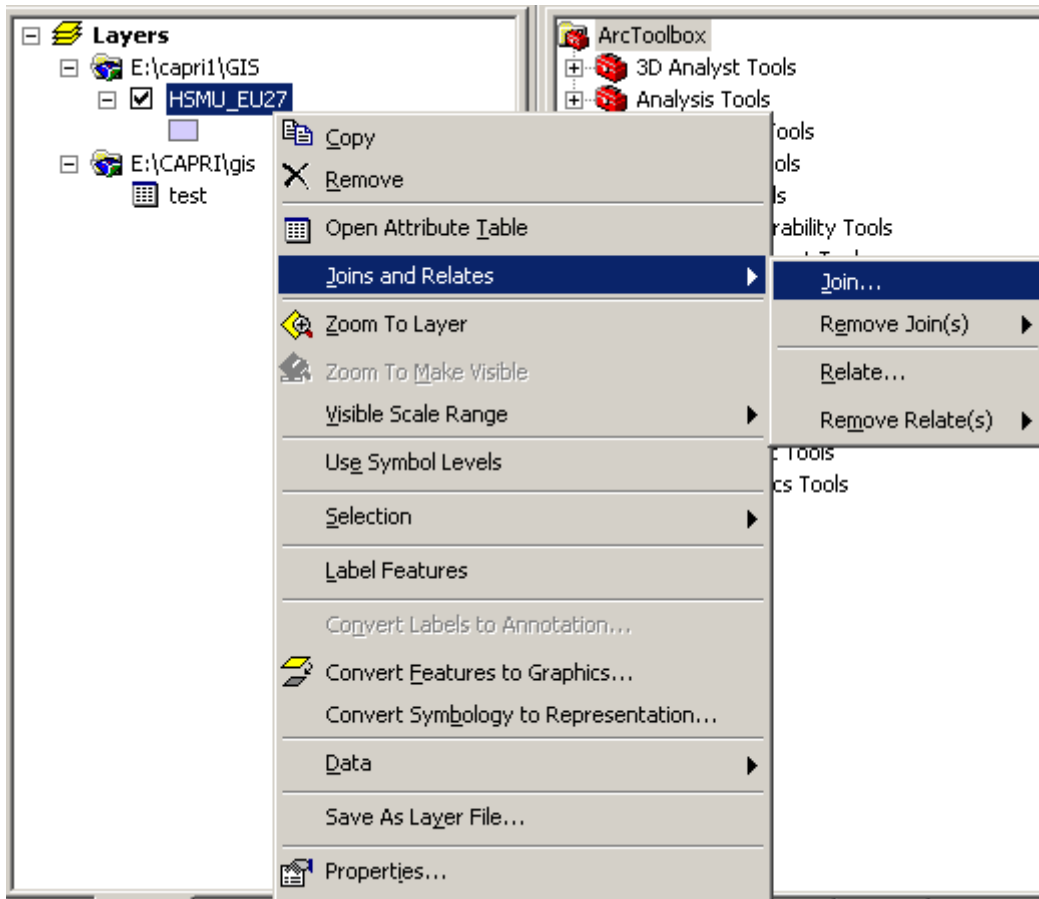
and in the case of the HSMUs, add the “HSMU_EU27.shp” shapefile.



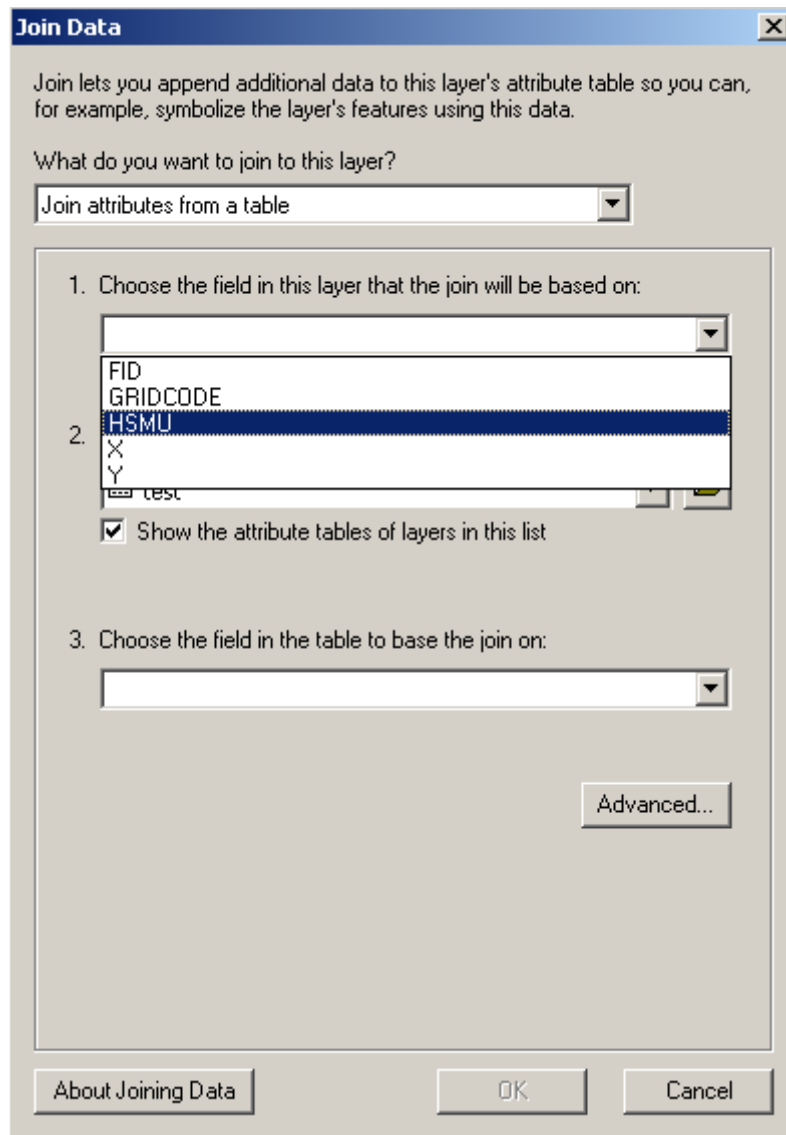
Then, choose add layers again, and add the dbf-file you have generated in the step explained above. You may also add the file with the meta data.



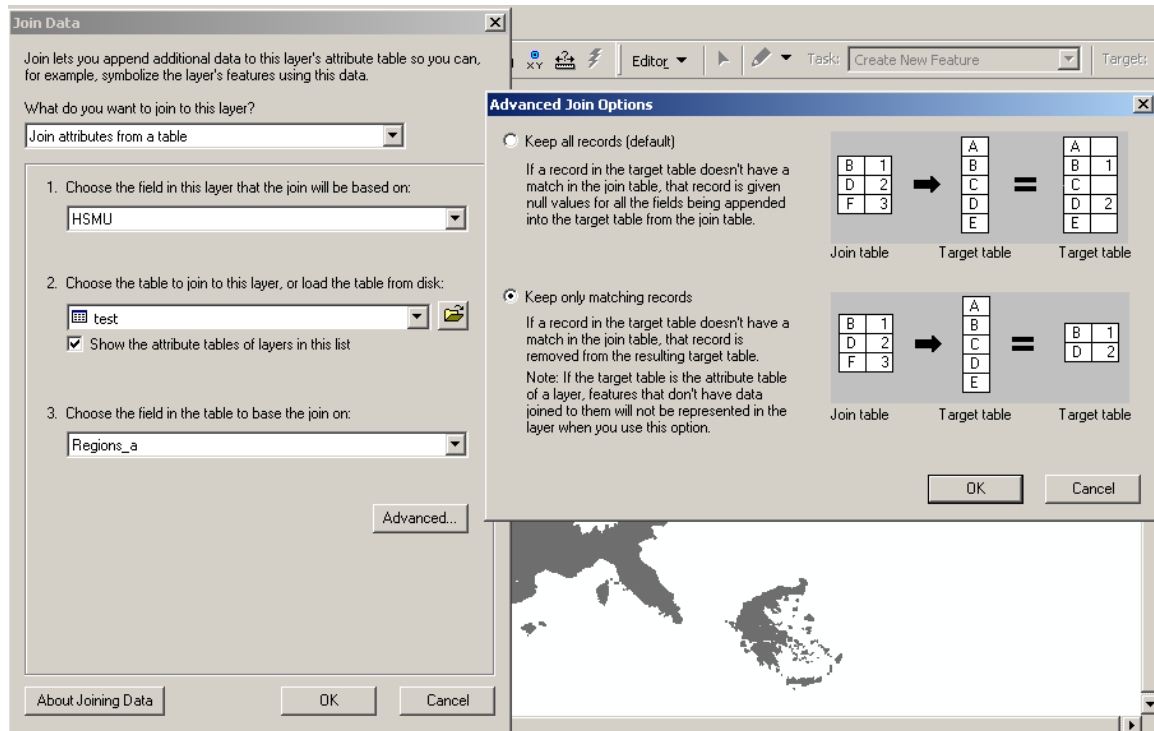
Next, we need to connect the HSMU geometry with the newly loaded data, a process called “joining” in ArcGis. In the context of “HSMU_EU27”, choose “Join and Relates”, then “Join ...”.



That will open the join dialogue as shown below.



Make sure that “Join attributes from a table” is set in the first drop down box, and under 1., select HSMU, i.e. the field in the HSMU_27 geometry where the codes for the HSMU polygons are stored. Use the name of the exported dbf-table under 2., and select the field “Regions_a” (the field name are restricted to 10 chars) under 3. Then press the button labeled “advanced”, and choose the radiobutton “keep only matching records”. If you are asked to build index, confirm.



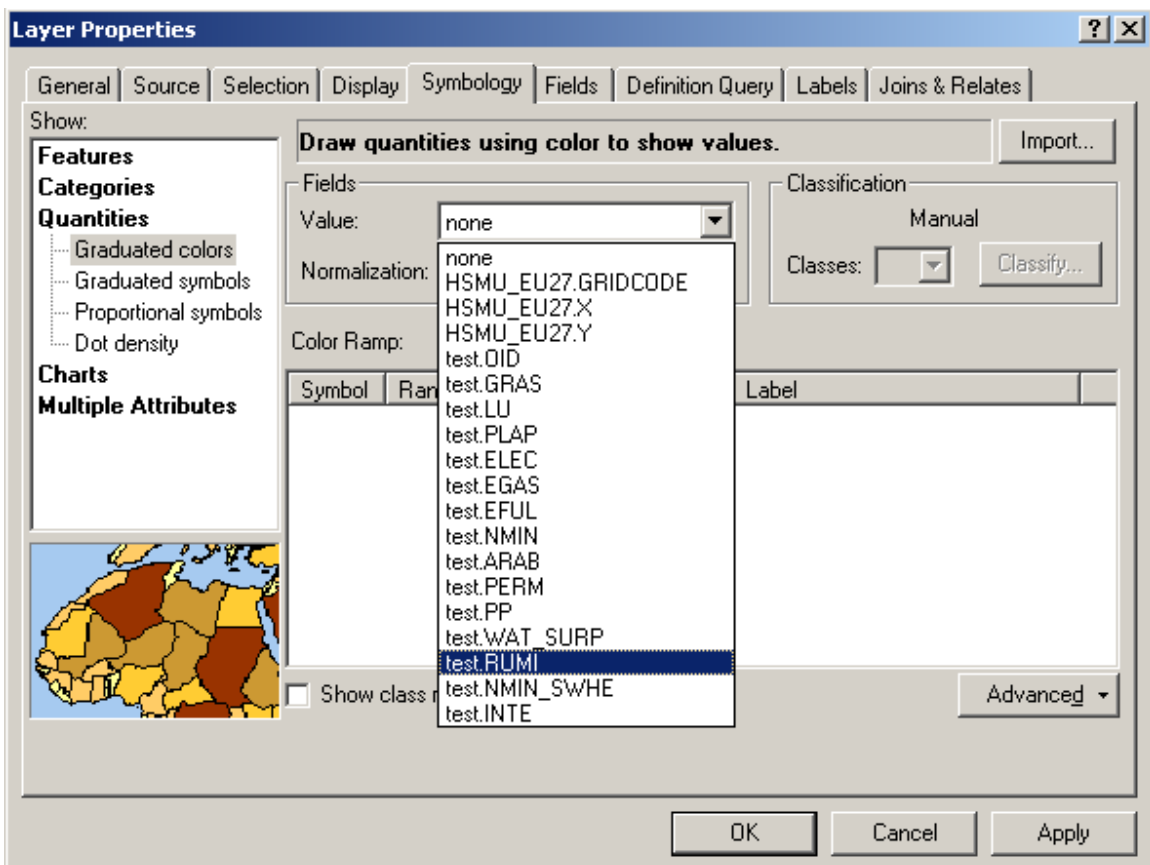
If anything has worked well, you should now see the country or countries you had in the original map.

There is a trap, though. If you export several tables, or results for several scenarios, your table will normally have several fields used as a row header (e.g. year, scenario, activity). If that is the case, the join will not work properly as several rows for the same regions will be joined to the very same polygon. Unfortunately, ArcGIS will not warn you about that. *First* you have to execute a definition query in the table, while selecting the rows which you are later going to draw a map from.

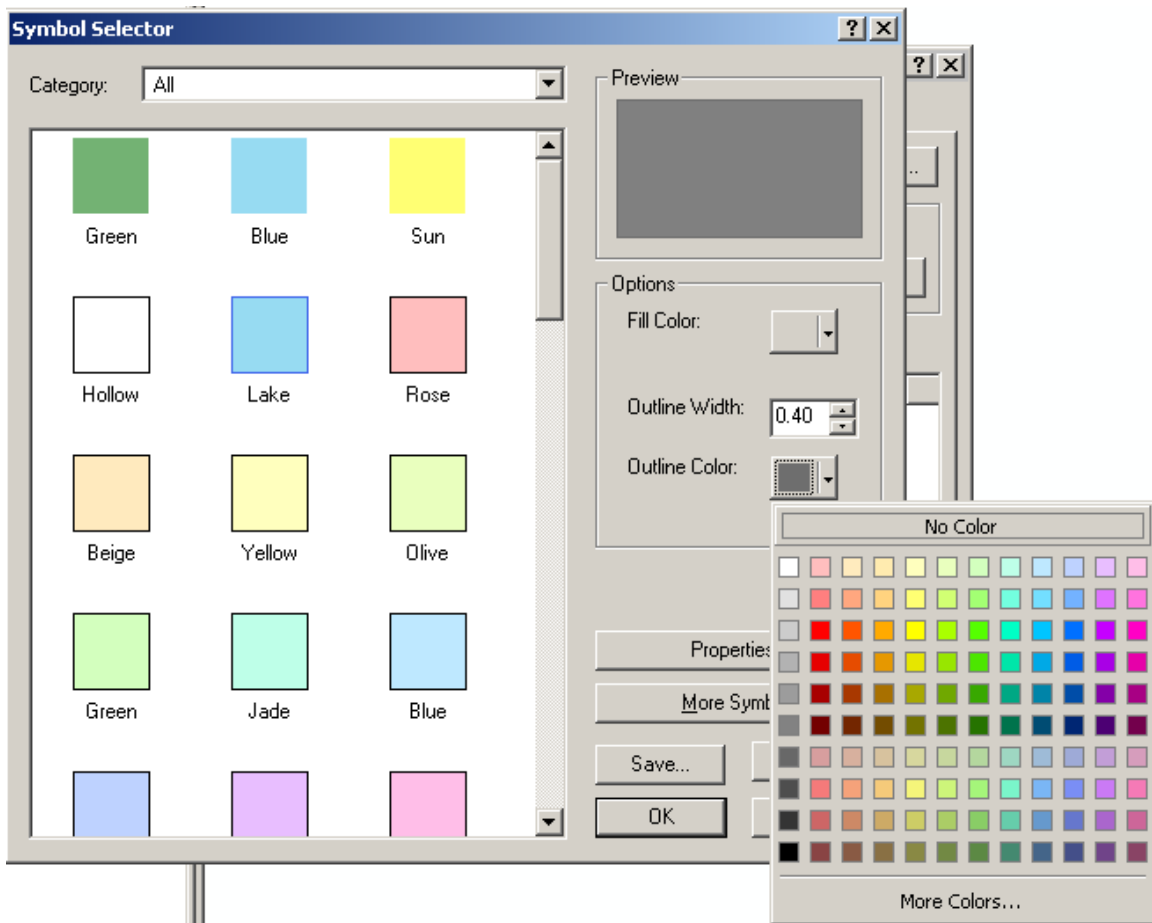
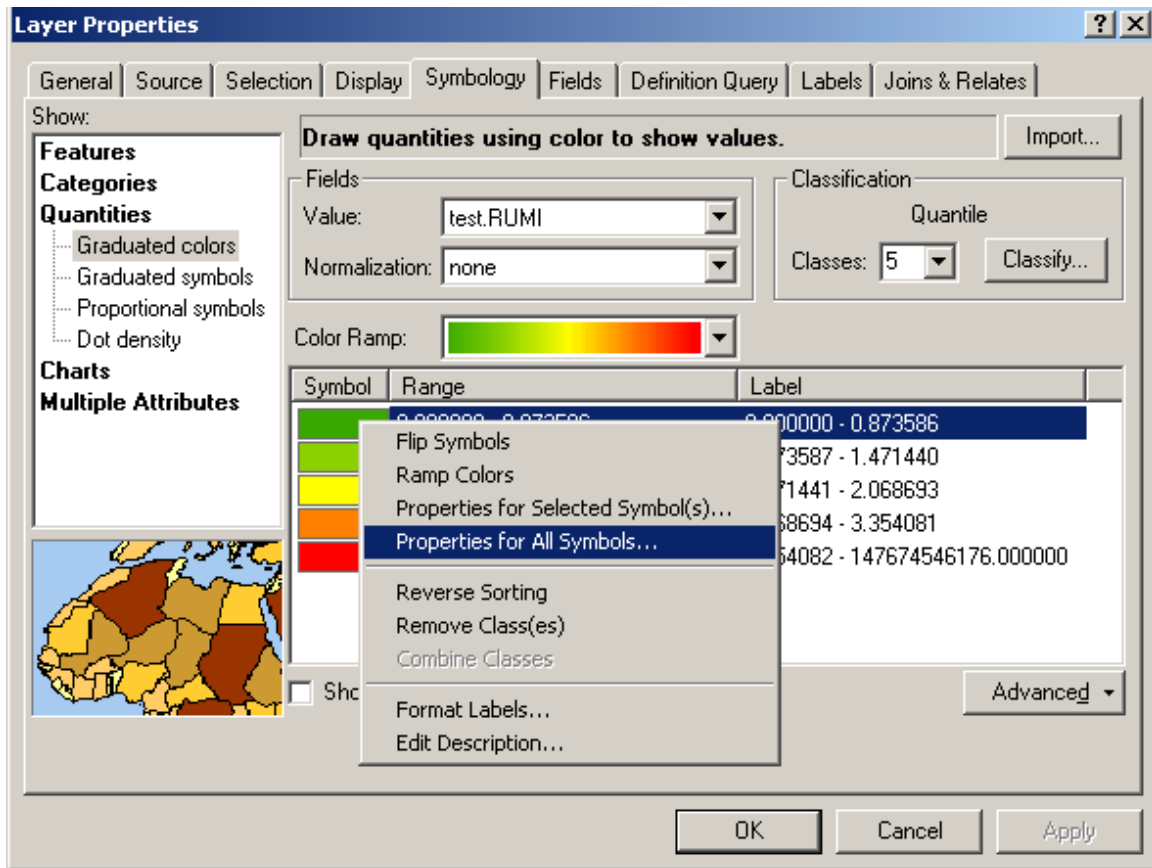
In order to draw a thematic map now, it may be helpful to add the file with the meta data to the map, and to open the meta-data table with the help of its context menu. It will give us the long description and units belonging to the data fields in the exported data table.

OID	Key	Name	Unit	LongText
0	NMIN	No 5 (a), Mineral Fertilizer Consumption, Nitrogen	kg N/ha	
1	PMIN	No 5 (b), Mineral Fertilizer Consumption, Phosphorous	kg N/ha	
2	NMIN_SWHE	No 5 (c), Mineral Nitrogen Application rate, Soft wheat	kg N/ha	
3	PLAP	No 6, Consumption of Pesticides	Euro/ha	
4	IRR	No 7(a), Irrigation, share	% irrigated	
5	WAT_SURP	No 7(b), Irrigation, abstraction	l/m2	
6	ELEC	No 8 (a), Energy, Electricity	Euro/ha	
7	EGAS	No 8 (b), Energy, Gas	Euro/ha	
8	EFUL	No 8 (c), Energy, Fuels	Euro/ha	
9	LU	No 10 (a), Cropping/Livestock pattern, livestock density	Livestock units / ha UAA	
10	RUMI	No 10 (b) Cropping/Livestock pattern, ruminants density	Livestock units / ha Fodder area	
11	PP	No 10 (c), Cropping/Livestock pattern, non-ruminants density	Livestock units / ha UAA	
12	ARAB	No 10 (d), Cropping/Livestock pattern, arable land density	%	
13	GRAS	No 10 (d), Cropping/Livestock pattern, grass land density	%	
14	PERM	No 10 (d), Cropping/Livestock pattern, permanent crops density(d)	%	
15	INTE	No 12 (a), low-medium-high input farming	Index 0 - 2	
16	H2865	BL21H2865		
17	H2866	BL21H2866		
18	H2867	BL21H2867		
19	H2868	BL21H2868		
20	H2869	BL21H2869		
21	H2870	BL21H2870		
22	H2871	BL21H2871		

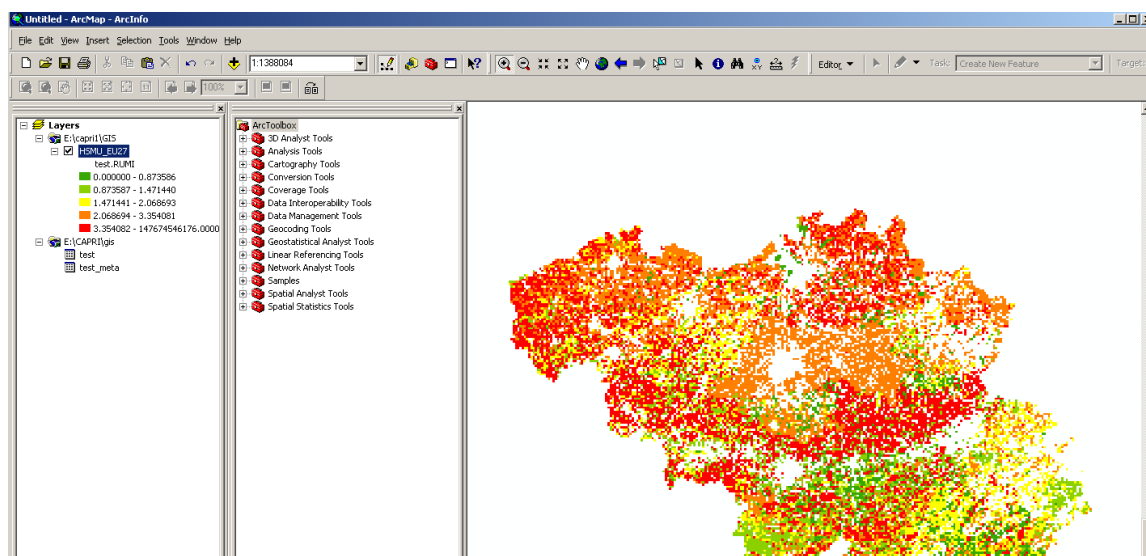
Assuming we want to draw a map now with the ruminant stocking density, we find it in row 10 under the key “RUMI”. In order to produce a map now, we have to open the context menu of “HSMU_EU27”, and choose properties, symbology, and choose “Quantities”. Under values, choose “RUMI”, the name before is the name of the DBF-file.



Afterwards, under classification, choose your preferred one. As there are many small polygons, the outline of the polygons should not be drawn. Therefore click on one of the colors, choose “Properties for all symbols ...” and under “Outline color” chose “No Color”.



Afterwards, if everything went well, you should see your map.



Machine learning

A serious challenge for large-scale economic models is the dimensionality of the results generated by model runs. These reflect the high level of dis-aggregation in different dimensions and the many aspects dealt with in these tools, such as relating to economic, social and environmental indicators. A single simulation run e.g. with CAPRI based on the farm type modules produces over 20 Mio non-zeros. Clearly, any of these numbers is generated by a computer based model and should hence be a non probabilistic outcome depending on the input and the code used. Specifically, the relation between the input and any single number outputted is determined by the model structure and parameterization, and pre and post-processing code. It must hence be possible to track any change quantitatively back to the shock analyzed.

But that rather theoretical point of view has very little to do with the task at hand when one has to distill from a set of model outcomes an analysis. The questions here are: what are the most important results, i.e. salient to the questions underlying the analysis and large enough to matter, and how can they be explained? For the client, the story behind the results is often at least equally important as the results themselves. If the story is well told, the “black box” character of the tool is removed and its usefulness in depicting major cause-effect relations becomes evident. Telling a good and right story requires however often quite some time in analyzing results in a systematic way.

The user will hence have to decide for which items of the huge data set a thorough analysis of underlying drivers is advisable. Limited time and human resources will set tight limits to the

extent of such systematic analysis. Typically, in any report, only a few dozen key results (perhaps complemented with a few maps showing several hundredths numbers) will be presented. But these key results, such as changes in aggregate welfare, farm income, GHG emissions or the nitrogen balance are calculated from thousands of simulated items. How can we discover “the story behind the results”, i.e. which regions, activities, price or policy changes etc. are most important for the aggregate changes communicated?

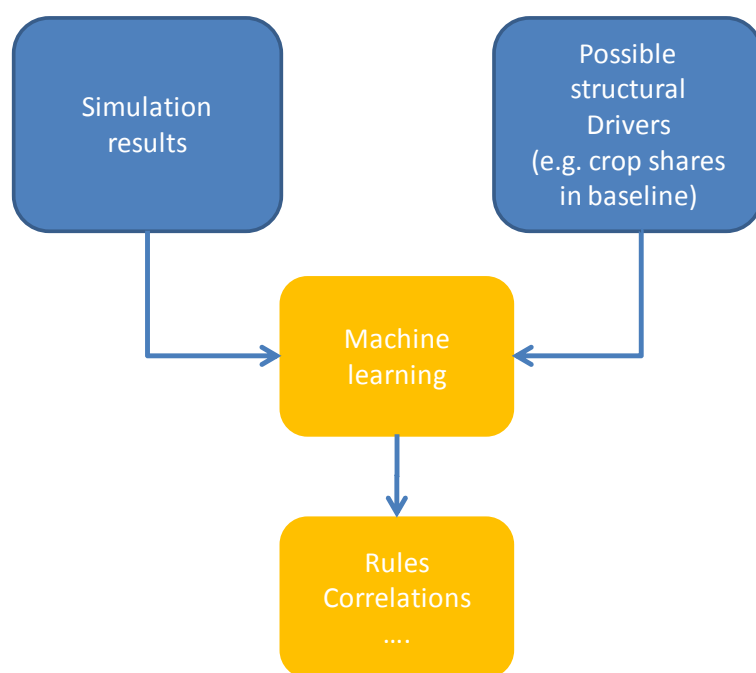
The exploitation tools of GGIG with a flexible on-the-fly approach to produce tables, graphs and maps had been an important step to improve the efficiency in exploiting and analyzing results. But in parallel, tools such as CAPRI has grown in scope and scale. It might be the time now to consider new approaches to analyze model outcomes.

Wikipedia gives the following definition: “**Machine learning**, a branch of [artificial intelligence](#), is a scientific discipline concerned with the design and development of [algorithms](#) that allow [computers](#) to evolve behaviors based on empirical [data](#), such as from [sensor](#) data or [databases](#). Machine Learning is concerned with the development of algorithms allowing the machine to learn via [inductive inference](#) based on observation data that represent incomplete information about statistical phenomenon. Classification which is also referred to as [pattern recognition](#), is a important task in Machine Learning, by which machines “learn” to automatically recognize complex pattern, to distinguish between exemplars based on their different patterns, and to make intelligent decisions.”

That is naturally a very general description. Machine learning has been widely in a wide range of application fields. A typical example is the analysis of which clients of a bank has been given credits. We have many observations with “credit granted” or “credit refused”, and probably a longer list of attributes of the clients (age, sex, income, amount of the credit asked for, time since being a customer with the bank, past bookings ...). Machine learning could be applied to define a set of rules which based on past decisions predict if a credit would be granted for a new application or not. Machine learning will in many cases also be able to tell something about the possible error range linked with the decision. That could e.g. allow the banks to make fast decisions in many cases, and spend more time on the tricky cases. The book by Witten et.al. 2011 gives many interesting examples. It might be interesting to note that one of the often cited applications of the WEKA package used in CAPRI refers to agriculture (Queen et.al. 2005), as the WEKA authors write: “New Zealand has several research centres dedicated to agriculture and horticulture, which provided the original impetus for our work, and many of our early applications.” The applications deal mostly with micro

level data. A recent application to price agricultural forecasting is found in Tielavilca et.al. 2010.

Now, we can e.g. see the income changes in each farm types in a simulation compared to the baseline as an outcome we want to predict, and their production program and changes in prices and premiums as the attributes used to explain that outcome. Some farm types might exhibit very large income changes, other little ones. What are common characteristics of the



one and the other group?

Machine learning might then come up with a “pattern” (e.g. based on a regression model) which determines the most important attributes impacting income changes in a given simulation. Machine learning has thus a lot of similarities with statistics – indeed many methods can also be found in statistical packages - but the focus to decide upon which

attributes and relations matters is shifted to a certain extent from the human being to the computer. And, the tool box used in machine learning differs to a certain degree from classical statistics. And, not of least, many of the algorithms had also been developed keeping computing time in mind.

Implementation in GGIG

The implementation in GGIG is based on the existing exploitations tool and the WEKA machine learning library (Witten et.al. 2011) which is also integrated into other well known packages such as RapidMiner. Thanks to the GNU license including full access to the underlying Java source code, it was possible to integrate the functionality of WEKA into the GGIG exploitation tools. Only a few code changes were necessary to pass data from the tables and maps shown in the GUI to the WEKA library (see below). That is done automatically in the background with the aim to reduce user input in the process.

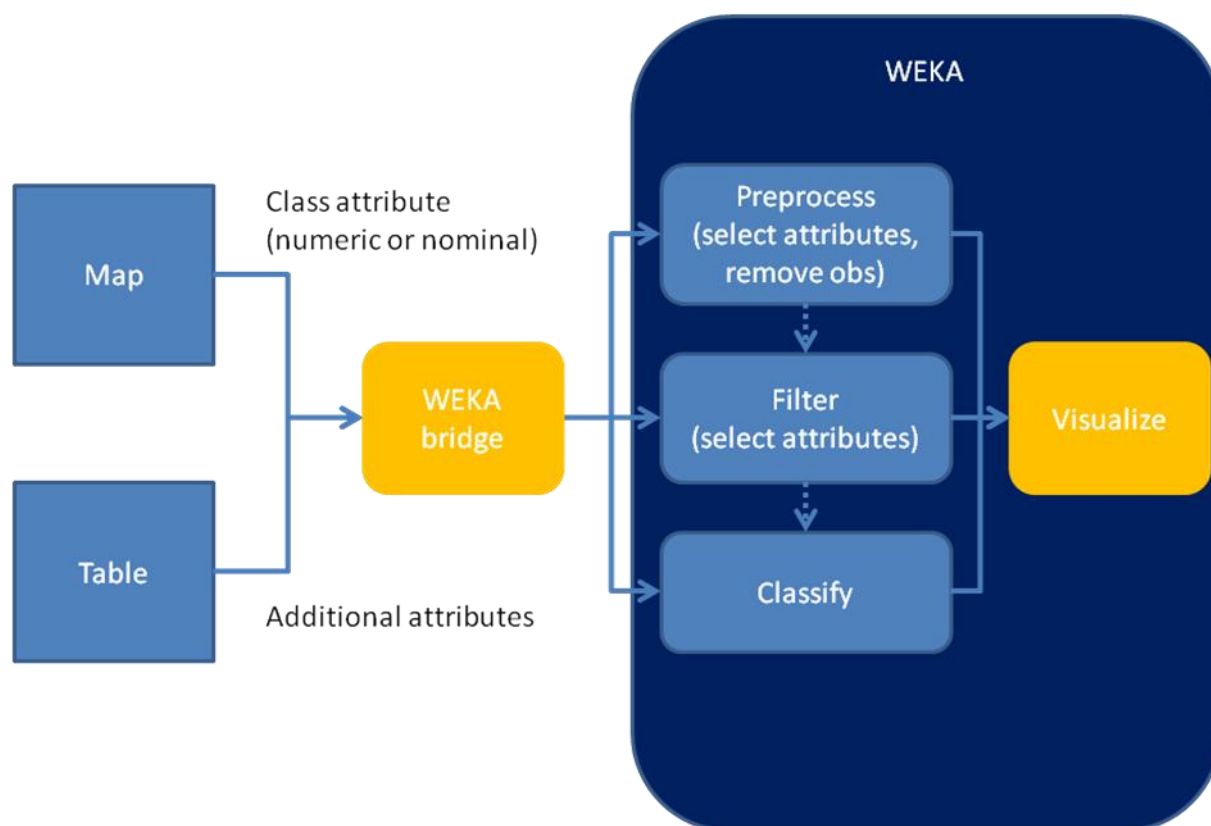
As a consequence, a very powerful set of filtering and classification as well as related visualization tools from machine learning can be applied to the result sets from CAPRI inside the existing exploitation tools.

The current implementation is based on the interaction of two views:

- A **map** or a table using classification colors – it defines the class attribute (=dependent variable) of the data to classify. For classification algorithms which require nominal values, the assigned class from the classification is used.
- A **table** with the “explanatory” attributes.

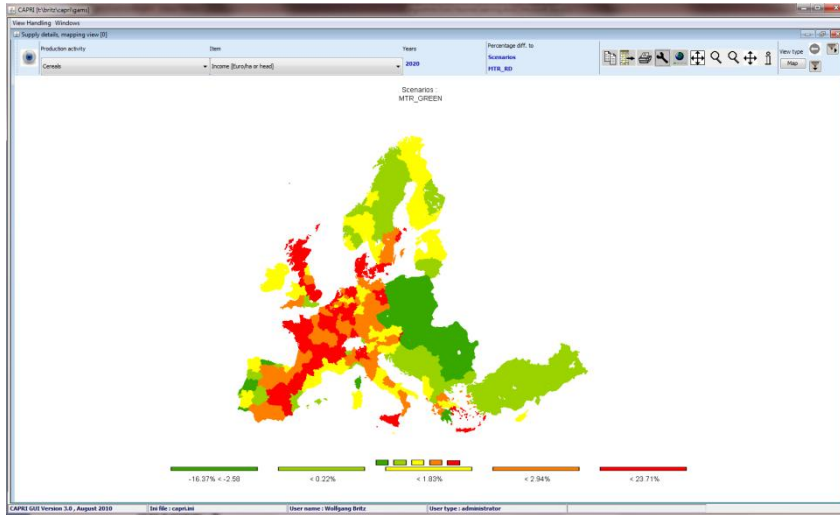
Both tables must be, as conventionally in the exploitation tools, the observations in the rows. For maps, each map carries the data for a region. But one might also work with two tables where the observations are not strictly geo-referenced entities such as farm types.

The GUI will automatically send new data to the WEKA GUI if either the map (or the table using classification colors) or the table is updated by a user action. The basic data flow is shown in the graphic below.



Interaction between the GGIG GUI and WEKA

Let's construct an example: we want to check if the income change in cereals in a simulation depends on the crop shares of cereals and the yields. In order to do so, we first render our map as usual (table “Farm details, mapping view”, use the option dialogue to show percentage changes against the baseline):



The regions shown are our instances and the value plotted for a region defines the class attribute we want to analyze. Any one instance consists of a vector of attributes of which one is the “class value”, i.e. the value to classify, which can be numeric or nominal. The other attributes are used for classification or clustering and stem from a second table (see below). Classification methods which use nominal values can also be used. In that case, the class chosen for the region, as seen from the color in which is drawn, defines the class attributes. In our example above, each region would fall into one of five classes.

Next, we open a second table with the data we want to use as explanatory attributes. The latest trunk comprises the table “Supply details, cluster view” which comprises promising attributes which are possible candidates to explain many changes in a simulation (for all activity aggregates: crop shares/stocking densities, revenues, income, yields).

	Cereals Revenues [Euro/ha or head]	Cereals Income [Euro/ha or head]	Cereals Yield [kg or 1/1000 heads/ha or head]	Cereals Crop share/Animal density [% or 0.01 animals/ha]	Oilseeds Revenues [Euro/ha or head]	Oilseeds Income [Euro/ha or head]	Oilseeds Yield [kg or 1/1000 heads/ha or head]	Oilseeds Crop share/Animal density [% or 0.01 animals/ha]	Other arable Revenues [Euro/ha or t
European Union 27	816.70	468.75	5524.26	38.49	893.92	517.03	2886.03	4.90	
European Union 25	838.28	482.28	5758.41	38.13	894.25	566.36	3195.66	4.28	
European Union 15	952.81	513.61	6318.16	25.80	1065.26	566.93	3388.98	3.49	
European Union 12	688.81	488.87	4278.35	42.65	722.84	475.08	2187.93	8.54	
European Union 10	578.69	419.93	4513.20	48.82	861.86	585.72	2563.58	7.56	
Belgium	1177.16	561.93	8640.45	24.77	1489.88	778.13	4336.85	3.78	
Denmark	1013.76	352.26	6855.91	53.84	1248.95	497.01	3778.61	3.26	
Germany	1059.39	441.45	7527.86	39.25	1314.72	638.88	4251.82	7.29	
Austria	889.55	497.57	6973.35	21.66	947.31	675.02	2404.66	3.17	
Netherlands	1266.88	711.29	8827.19	12.49	868.24	764.83	3895.68	0.52	
France	1085.12	438.28	7478.60	29.96	1014.91	456.67	3405.41	6.27	
Portugal	746.24	356.13	3898.35	6.25	191.71	144.97	497.99	0.60	
Spain	593.53	532.23	3488.86	28.42	433.16	473.81	1072.65	1.29	
Greece	794.69	891.87	4133.29	18.20	688.18	776.33	1655.66	0.04	
Italy	1053.24	786.85	5754.93	28.54	874.83	542.60	2871.15	1.68	

In order to start the clustering/classification, we click in the table to open its popup-men and then select “Classification”:

We clicking one of the option if we can then decide to:

- use *numerical classification* methods such as different regression methods. The observations in the map define the dependent variable.
- Use the class assigned by the maps input into *nominal classification*.
- To switch classification off.

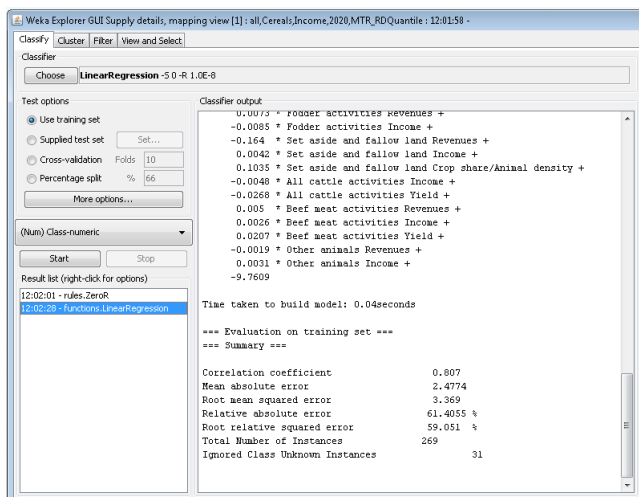
A new window will be opened which shows the WEKA GUI (see below).

The WEKA GUI

The classification is based on the complete functionality of the WEKA GUI regarding attribute selection/visualization, filtering and classification, see <http://www.cs.waikato.ac.nz/~ml/index.html>. There are very good manuals available from the site (the latest user manual is also available from <http://www.capri-model.org/docs/WekaManual-3-6-5.pdf>), so that only a few major tips are given below for fast start.

The tabs “Classify”, “Cluster”, “Filter” and “View and select” allow the user to access specific part of the WEKA functionality. The result set from the current classification run can be shown in the lower left panel (result list). For each result set, a popup menu opens options, e.g. to show a graph with the prediction errors.

Classification



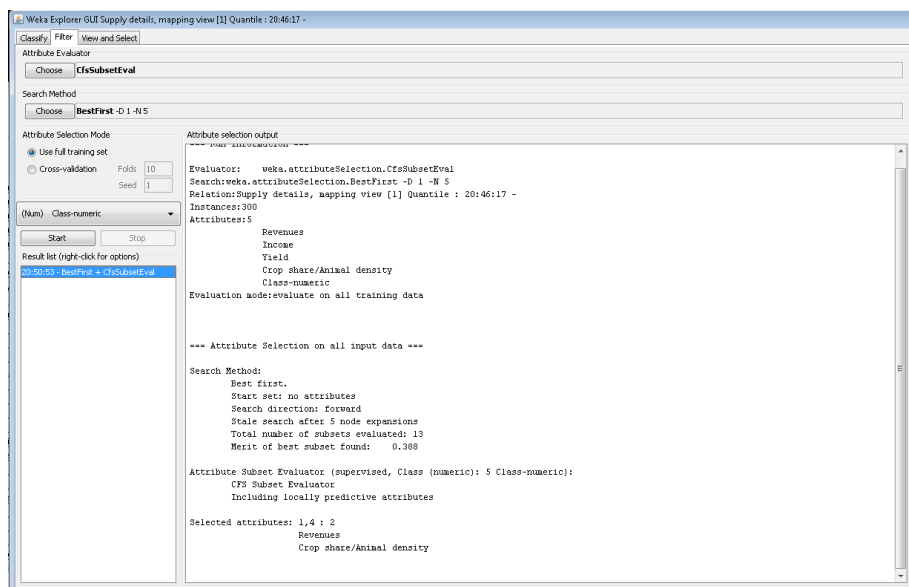
- The “**choose**” button will give access to a wide range of **different classifiers**, many of which have additionally options which can be edited by users. A multiple linear regression using the Akaike criterion for model selection is used as the default, assuming that most people will start with using numerical values as class attributes. Please not that switching between nominal and numerical class attributes might trigger error messages if the currently selected classifier cannot handle the newly selected class attribute type.
- It is recommended for our purposes to use under “Test options” “Use training set” (the default in our implementation) as we are typically not interested in an out-of-sample test of the prediction quality.

- The actual classification can be started with the “start” button. If the data in the background are updated, the actually chosen classifier with the chosen options will be started on the new data set automatically. In absence of errors the “Classifier output” on the RHS will hence typically show results based on the latest selected data.
- The results can be visualized by clicking with the mouse on an item in the result list, the last on in the list always being the newest. If one has tried several classifiers, the old results remain available. However, if the data in the background change, the old results are automatically removed.

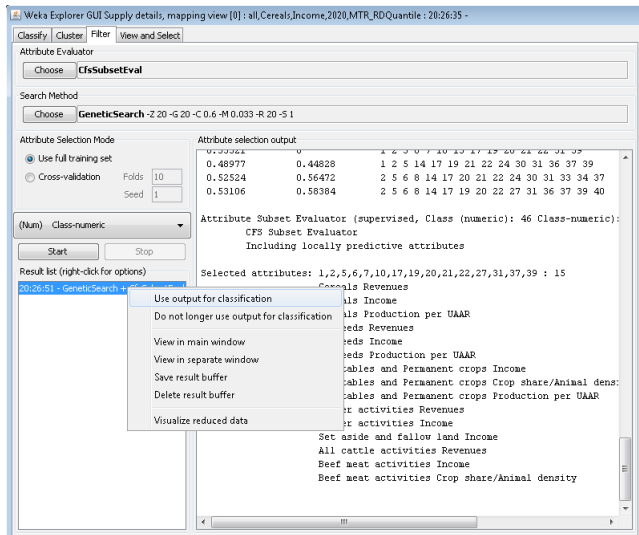
The reader should note that all the functionality described is from the standard WEKA GUI so that the user manual from WEKA can be used for further information.

PS: The cluster panel is not described, it works quite similar. Note however that filters are not applied to the cluster (see below).

Filtering



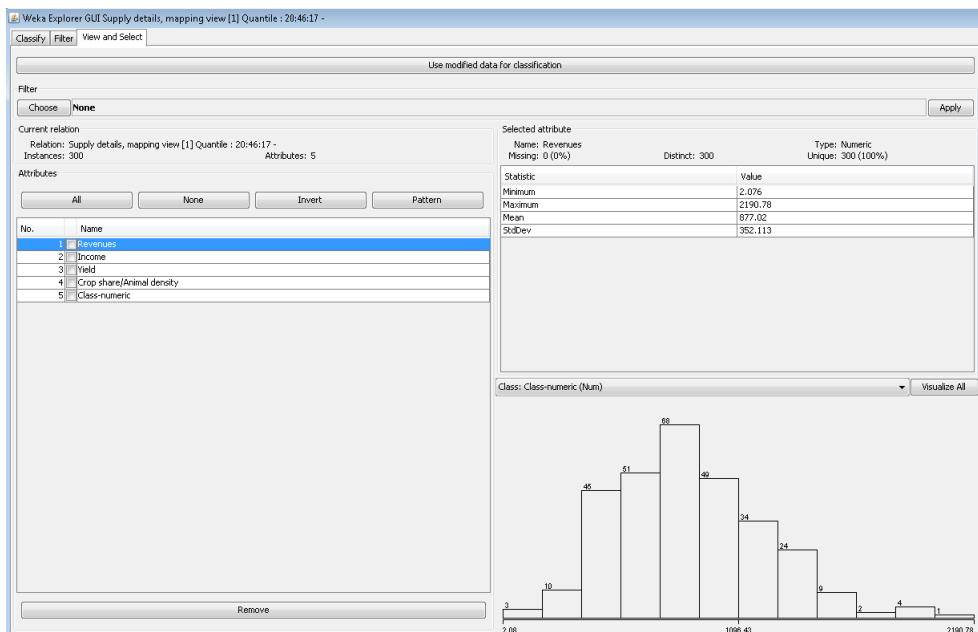
The filter panel allows running different types of filters which remove attributes, in many cases reflecting the correlation between attributes. In order to use the result from the filter run, click on the result set and chose “Use output for classification”:



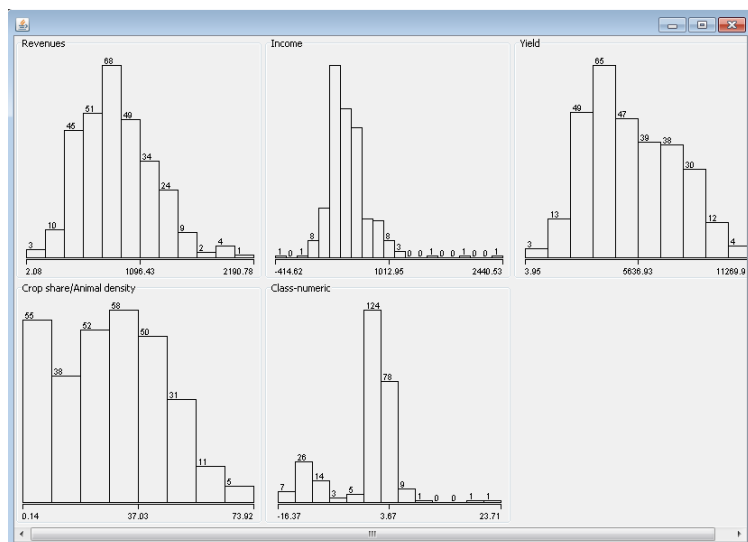
The last selected filter will be automatically restarted if a new data set is implicitly loaded (change of the map or of the data in the cluster table with the explanatory results). In order to switch off the use of the filter, select “Do not longer use output for classification”

Attribute viewing and selection

The last panel available is especially interesting to quickly analyze statistics of the underlying data:



The reader can manually remove attributes and the reduced set of attributes will then be passed to the filter and classifier. However, the attribute selection is not maintained when new data are loaded. The “Visualize All” button produces graphs of all current attributes:



Summary

The integration of algorithms from machine learning based on the WEKA library and GUI offers new possibilities to systematic analysis of result sets. Thanks to the open source policy of WEKA, it was possible to integrate these powerful tools transparently in the CAPRI GUI. Depending on the experiences made over the next months, further links might be included (e.g. rendering clusters in maps).

References

- Ian H. Witten, Eibe Frank, Mark A. Hall (2011). *Data Mining Practical Machine Learning Tools and Techniques*. Third edition. Elsevier, Amsterdam. 630 pages
- Remco R. Bouckaert, Eibe Frank, Mark Hall, Richard Kirkby, Peter Reutemann, Alex Seewald, David Scuse (2011). *WEKA Manual for Version 3-6-5*. June 28, 2011, University of Waikato, Hamilton, New Zealand.
- Robert J. McQueen, Stephen R. Garner, Craig G. Nevill-Manning, Ian H. Witten (1995). Applying machine learning to agricultural data, *Computers and Electronics in Agriculture*, Volume 12, Issue 4, June 1995, Pages 275-293, ISSN 0168-1699 (<http://www.sciencedirect.com/science/article/pii/0168169995986019>)
- Ticlavilca, A. M., Dillon M. Feuz and Mac McKee. 2010. "Forecasting Agricultural Commodity Prices Using Multivariate Bayesian Machine Learning Regression." *Proceedings of the NCCC-134 Conference on Applied Commodity Price Analysis, Forecasting, and Market Risk Management*. St. Louis, MO. [<http://www.farmdoc.illinois.edu/nccc134>].

Scenario editor

The scenario editor is an optional tool to be embedded in a GGIG user interface which supports the user in setting up run specific include files where the content is *not* stemming from GUI controls. That parallel way to define run specific input is typically necessary for more complex tools where e.g. policy scenarios are defined in GAMS code.

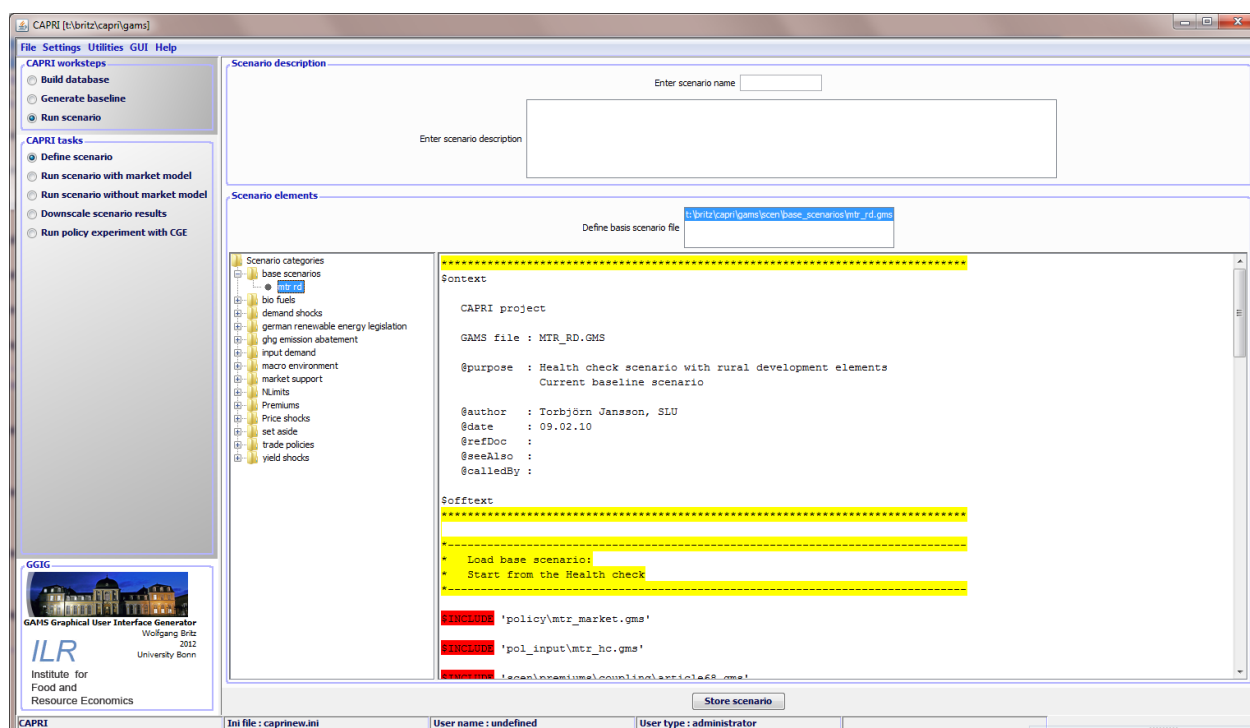
The scenario editor is a “predefined” task which must be named “Define scenario”, e.g.

```
<task>
  <name>Define scenario</name>
  <userLevels>runner,Administrator,developer,debugger</userLevels>
</task>
```

A related setting stores the directory where the input files are found:

```
<scenarioDir><attr>scen</attr></scenarioDir>
```

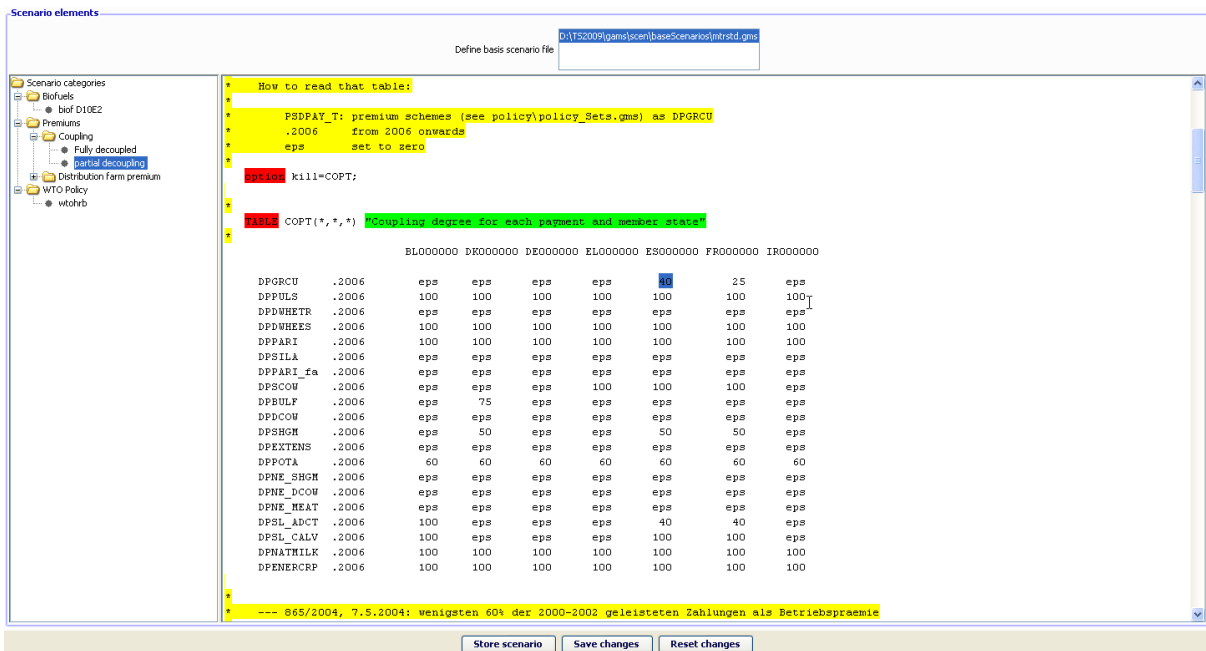
The screen shot below shows an example from CAPRI.



Choosing the scenario editor adds the panel with GUI elements shown above. The panel consist of two major panes:

1. A top pane where the user can enter a name for his new scenario, and a description text.

2. A bottom pane where the user can define the base scenario to start with (currently in the trunk “MTR_RD.gms”) and the snippet to add. The available snippets and their structure are shown on the left hand side in an expandable tree which shows the sub-directories found under “gams\scen”, with the exclusion of a sub-directory called “baseScenarios” and the “.svn” directories. Empty directories are not shown. The user may select any number of snippets, even several from the same sub-directory. Double-clicking on one of the snippets shows the content of the file on the right hand side, so that the user can inspect the code as seen below in more detail. GAMS keywords are shown in red, comments in yellow and strings in green. He can also edit the file – changes are shown in blue. Once changes had been saved, the tree shows a (user modified) behind the category. The user can also remove the changes from snippets.



Storing the scenario then generates a file as shown below, user name, the reference to CAPMOD.GMS and the date and time are automatically added by the GUI. The files will be added to the files stored in “gams\pol_input”.

Meta data handling

Why meta data?

Meta data are data about data. In many GAMS projects, it is impossible or cumbersome to tell exactly based on which shocks and settings results of a model run had been generated. That is due to the fact that run specific settings are not stored at all or not stored together stored with

the results of the run. Later on, result users are often left guessing what exactly the settings underlying the run might have been.

In order to overcome that problem, the GGIG, drawing on [CAPRI GUI concepts](#), passes all interface settings, plus the user name and the current time, forward to GAMS in one SET called META.

A correctly defined interface with GGIG should allow to steer *all* run specific settings. If that is the case, the meta data generated by GGIG will provide an exact and sufficient definition of all run specific inputs, ensuring that all relevant meta data about a run are stored along with quantitative results in the same GDX file. Accordingly, GDX files shipped to other desks or committed e.g. to a SVN server still carry all necessary information to identify exactly the run.

Technical concept

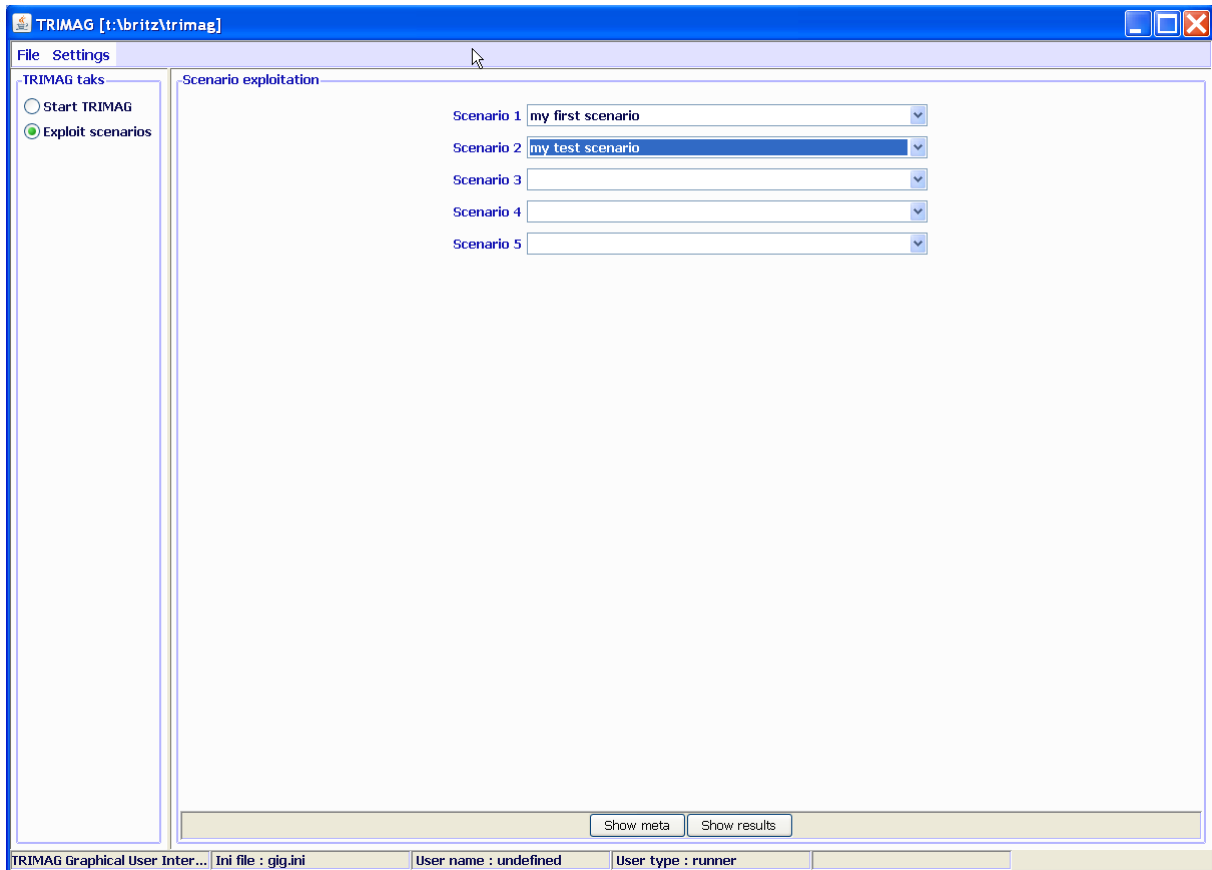
The meta handling is straight forward. The state of the different control is mapped into pairs of set elements and related long text descriptions as shown below from an example application:

```
SET META /  
'Scenario description' 'my test scenario'  
'Choose model type' 'CGE'  
'Relative weight flows' '30'  
'Use demand elasticities' 'true'  
'Set substitution elasticity' '6.0'  
'Countries' 'NL,'  
./;
```

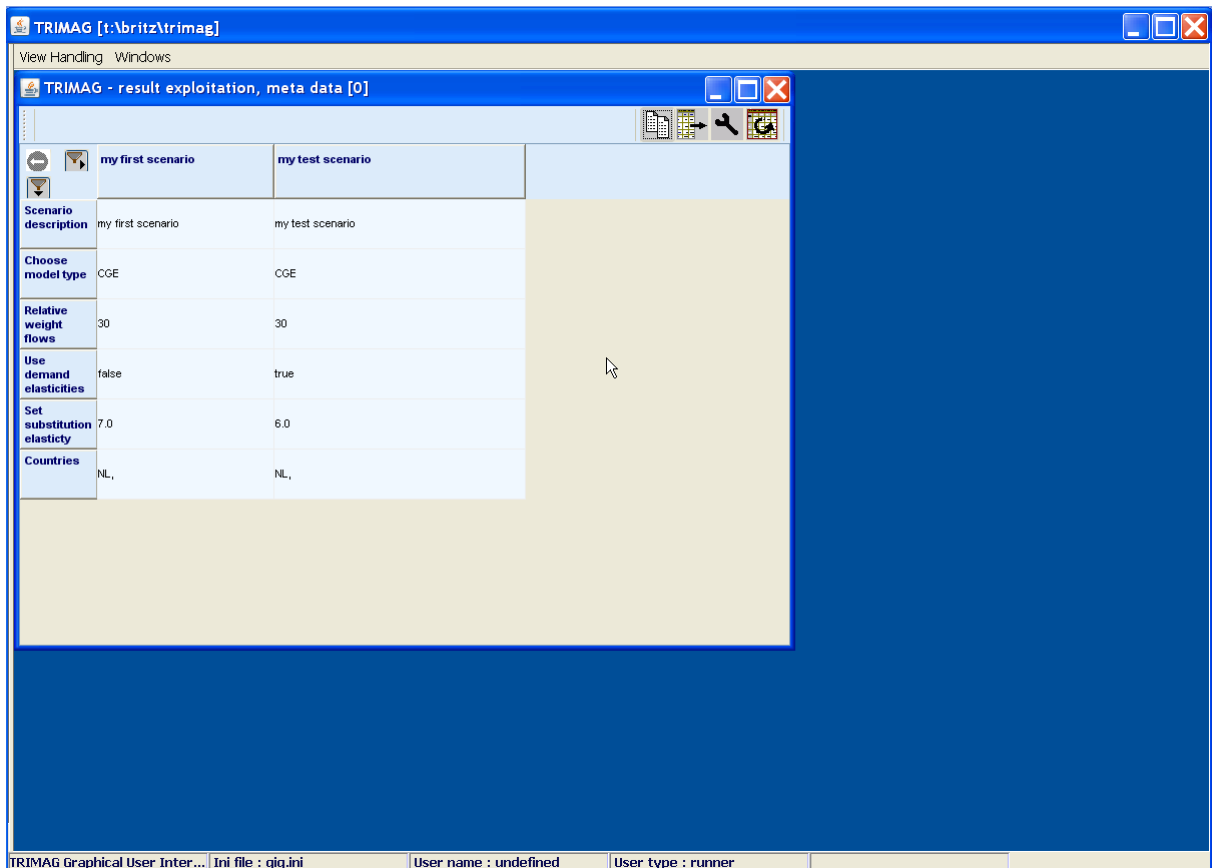
and, might with one GAMS statement as shown below, stored in the GDX files along with the results:

```
execute_unload "%scenario_description%.gdx" META,RESULT;
```

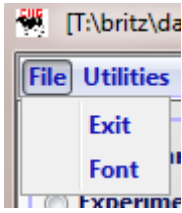
The user might then select some scenario:



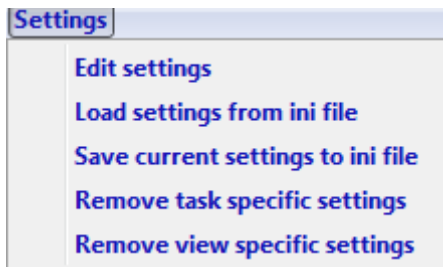
And then, by pressing “show meta”, view the settings used for these scenarios:



File menu



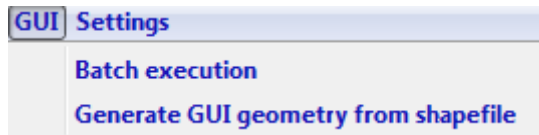
Settings menu



The settings dialogue was already discussed above.

Utilities and GUI menu

Most of the utilities are discussed below.



These utilities are discussed below.



Utilities: Batch execution

The batch execution facility is a tool which:

- Allows executing many different tasks after each other without requiring user input.
- Reports the settings used, any errors and GAMS result codes in a HTML page from which they may queried at a later time.
- Ensures that each new run generates its own listing file, which can be opened from the HTML page.

- Allows storing the output of the different runs in a separate directory, while reading input from unchanged result directories.

The purpose of the batch execution facility is therefore at least twofold. On the one hand, it allows setting up test suits for the GAMS code of a project such as checking for compilation without errors for all tasks and different settings such as with and without market parts etc. Secondly, production runs of e.g. different scenarios can be started automatically. Timer facilities allow starting the batch execution at a pre-scheduled time. Along with functionalities to compare in a more or less automated way differences in results between versions, the batch facility is one important step towards quality control.

The batch execution allows starting a file, defining settings and tasks from the different CAPRI work steps and executing them without user intervention. Once started, the batch processor may be stopped so that the currently running GAMS program ends on its own (“end batch execution after next finalised GAMS step”) or by sending a “CTRL-C” to the GAMS program. It will continue to run until the GAMS processor notices the CTRL-C – which may take a while – and then end with an error code. However, the GAMS processor will run some finalisation tasks as removing temporary files and directories.

Format of the batch execution steering file

- Generally, each line in the file comprises a keyword following by an equal sign and the related setting.
- Comment lines start with an asterisk. Blocks comment are between the keyword “ontext” and “offtext”, which thus allows easily excluding blocks of lines from execution.
- The keyword “exit” prevents further processing.

Header

A batch execution file starts with a header which defines settings otherwise entered by user under settings dialogue of the GUI, i.e. directories, the GAMS engine the use and some further settings:


```
gams engine=d:\GAMS23.9\gams.exe
user = wolfgangb

work dir = t:\britz\capri\gams
*
* --- where the HTML page and the listings
*       will be stored
*
output dir = d:\temp\batch

res dir = t:\britz\results
scr dir = d:\scrdir

gams options = scrdir=d:\scrdir --threads=on
*
* --- the following settings will write
*       the results into different directories.
*       The directory structure will be automatically
*       generated
*

restartOutDir = t:\britz\capri\restart
resultsOutDir = t:\britz\results

number of processors = 30
processor speed relative = 160
|
number of iterations = 99
```

Settings for tasks

Each include file generated by GGIG comprises a block, commented out from use by GAMS, which can be pasted into a batch execution file, see example below:

```
: * -----
: * Setting for executing the task in batch file mode
: * -----
: $ONTEXT
:
:   task =Run scenario with market model
:
:   Scenario description = caprese\subCaprese\caprese_1
:   Scenario description CGE = cge_rd_noChg
:   Generate GAMS child processes on different threads = ON
:   Use new global version = OFF
:   EU28 = OFF
:   Base year = 2004
:   Simulation years = 2020,
:   Countries = BL000000 "Belgium and Luxembourg",DK000000 "Denmark",DE000000 "Germany",EL000000 "Greece",ES000000 "Spain"
:   Regional breakdown = Countries
:   Global, spatial multi-commodity model = ON
:   Endogenous bio-fuel markets in global market model = ON
:   Policy blocks (additional geographical layer) = OFF
:   Endogenous margins between trade block and country prices = OFF
:   Endogenous young animal markets = ON
:   Regional CGEs = OFF
:   Number of iterations = 99.0
:   Use lower price iterations weights after iteration = 20.0
:   Alternative GAMS license file for GHG emission estimation = gamslice_cplex
:   Aggregates for activities and commodities = ON
:   Environmental Indicators = ON
:   Life-cycle assessment for energy = ON
:   Multi-functionality indicators = ON
:   Iteration tracking = ON
:   Sensitivity experiments with features in supply model = ON
:   Endogenous net migration = ON
:   Fixed budget for factor subsidies = ON
:   Capital stock = DPSU rule
:   Labor supply = Wage curve
:   Capital mobility = Sluggish
:   Labor mobility = Sluggish
:   Land mobility = Sluggish
:   Closure current account and trade balance = Exchange rate
:   Closure household account = Spending
:   Closure government account = Spending
:   Load meta information from older task = ON
:   Solution printing = Suppress
:   Determine point price elasticities = OFF
:   Print gams code to listing = offListing
:   Solprint = On
:   Limrow = 0.0
:   Limcol = 0.0
:   Maximum number of pre-steps market model = 15.0
:   Solution print at preparatory solve = OFF
:   Abort after preparatory solve = OFF
:   Solution print for pre-steps in 1st iteration with abort = OFF
:   Plus iterlim to zero for 1st pre-steps in 1st iteration = OFF
:   Number of presteps before abort = 1.0
:   Kill simini file = OFF
:   Additional result type identifier =
:
:   execute=gamsrun
:
: $OFFTEXT
:
: * end batch execution file
```

Settings which do not change between tasks need not to be repeated, executing e.g. different scenarios is then simply done by changing the scenario file, followed by the keyword “execute” as shown below:

```
scen name=MTR_DECPL
execute=gamsrun

scen name=MTR_GREENGRAS
execute=gamsrun

scen name=MTR_GREENCROPD
execute=gamsrun

scen name=MTR_GREENSET
execute=gamsrun

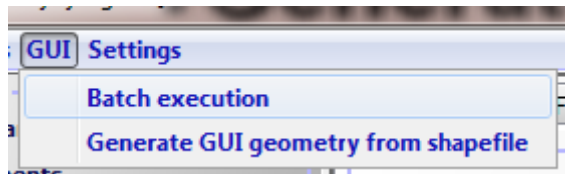
scen name=MTR_GREEN
execute=gamsrun

scen name=MTR_CONU
execute=gamsrun

scen name=MTR_GREEN_CONU
execute=gamsrun
```

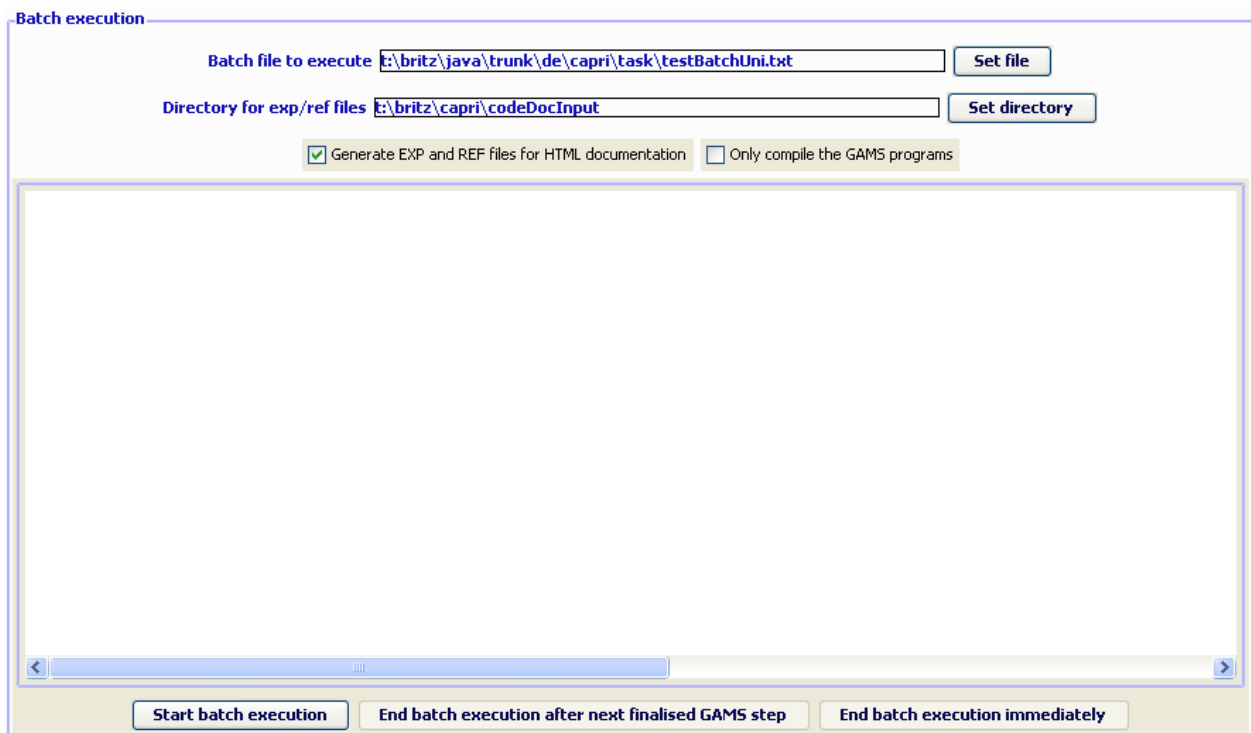
Using the batch execution facility

The batch execution utility can now be opened from the menu bar under “GUI”:



It will open a separate windows as shown below.

Graph: Batch execution panel



If the suite of tasks comprises “execute” statements, those can be downgraded to “compile” with “Only compile GAMS programs” check box.

The check box “Generate EXP/REF files for HTML documentation” adds settings to the GAMS calls which generate two specific reference files by the GAMS compiler which comprise information of files and symbols used by GAMS. For details on the code documentation facility see the technical document “Javadoc like technical documentation for CAPRI” to be found on the Capri web page under technical documents. The “directory for exp/ref files” defines where those files will be stored.

The batch language allows definition of a timer, i.e. to start the execution at a specified time.

The output from batch execution

As it is assumed that batch execution will not be monitored by the user during execution, a logging mechanism is established. Listing files and generated include files are stored in sub-directories of the “output dir” defined in the batch execution file:

```
*
* --- where the HTML page and the listings
*      will be stored
*
* output dir = d:\temp\batch
```

The sub-directory is named after the time point where the batch execution is started:

 24.07.2014_07.34.12	24.07.2014 08:57	Dateiordner
 24.07.2014_07.40.32	24.07.2014 08:57	Dateiordner

These sub-directories comprise the listing files and generated include files, labelled according their starting sequences, e.g.:

0_1.lst	24.07.2014 07:53	LST-Datei	158 317 KB
1_1.lst	24.07.2014 08:06	LST-Datei	158 403 KB
2_1.lst	24.07.2014 08:19	LST-Datei	172 658 KB
3_1.lst	24.07.2014 08:32	LST-Datei	172 673 KB
4_1.lst	24.07.2014 08:45	LST-Datei	156 906 KB
5_1.lst	24.07.2014 08:57	LST-Datei	955 KB
batch.html	24.07.2014 07:53	HTML-Dokument	15 KB
fortran.gms.0_1	24.07.2014 07:40	0_1-Datei	18 KB
fortran.gms.1_1	24.07.2014 07:53	1_1-Datei	18 KB
fortran.gms.2_1	24.07.2014 08:06	2_1-Datei	18 KB
fortran.gms.3_1	24.07.2014 08:19	3_1-Datei	18 KB
fortran.gms.4_1	24.07.2014 08:32	4_1-Datei	18 KB
fortran.gms.5_1	24.07.2014 08:45	5_1-Datei	18 KB

Additionally, a HTML page reports all tasks which have been started, the return code of the GAMS process and all major setting, as well as link to open the listing file with the editor. The following screen shot shows the first part of the HTML page resulting from executing a batch file. Tasks which did yield a non-zero GAMS return code and errors are shown in red.

CAPRI batch execution report

Runs are generated from		d:\temp\batch\24.07.2014_07.40.32\batch.html							
at		Do, 24 Jul 14 07:40:32							
Workstep	Task	User	Directories: work result dat restart	Settings	Gams options	Started/Ended/Time used	RC	Listing	Gdiff
Run scenario	Run scenario with market model	undefined	T:\britz\capri_liaise\gams T:\britz\results_liaise T:\britz\capri_liaise\dat null	'Scenario description' liaise\MTR_B1Tech_ad 'Scenario description CGE' cge_rd_noChg [default] 'Generate GAMS child processes on different threads' 'ON' 'Use new global version' 'ON' 'EU28' liaise [default] 'Base year' '2004 [default] 'Simulation years' '2050' 'Countries' 'BL000000 Belgium and Luxembourg,DK000000 Denmark,DE000000 Germany,EL000000 Greece,ES000000 Spain,FR000000 France,IR000000 Irland,IT000000 Italy,NL000000 The Netherlands,AT000000 Austria,PT000000 Portugal,SE000000 Sweden,FI000000 Finland' 'Regional breakdown' 'NUTS2' 'Global, spatial multi-commodity model' 'ON' 'Endogenous bio-fuel markets in global market model' 'ON' 'Policy blocks (additional geographical layers)' 'OFF' 'Endogenous margins between trade block and country prices' 'OFF' 'Endogenous young animal markets' 'ON' 'Regional CGEs' 'OFF' 'Number of iterations' '99 0' 'Use lower price iterations weights after iteration' '20 0' 'Alternative GAMS license file for GHG emission estimation' 'gamslice_cplex [default] 'Aggregates for activities and commodities' 'ON' 'Environmental indicators' 'ON' 'Life-cycle assessment for energy' 'ON' 'Multi-functionality indicators' 'ON' 'Iteration tracking' 'ON' 'Sensitivity experiments with features in supply model' 'ON' 'Endogenous net migration' 'ON' 'Fixed budget for factor subsidies' 'ON' 'Capital stock' 'DPSV rule [default] 'Labor supply' 'Wage curve [default] 'Sectoral capital stock' 'Sectoral capital stock [default]	run --scen=fortran --ggig=on	Do, 24 Jul 14 07:40:32 Do, 24 Jul 14 07:53:19 12 min and 48 secs	0	D:\temp\batch\24.07.2014_07.40.32 0_1.lst	

Utilities: Generate GAMS documentation in HTML pages

Graph: Panel to steer GAMS documentation generation



The GUI comprises a tool to generate for each GAMS file and each symbol used HTML pages which are interlinked. For details on the code documentation facility see the technical document “Javadoc like technical documentation for CAPRI” to be found on the Capri web page under technical documents.

The controls on top allow the user:

- To define in which directory the “EXP”, “REF” and “GDX” files are stored which serve as input into the documentation generator.
- To choose the directory where the HTML files will be generated.
- To select the tasks covered by the documentation generator.

Structure of the HTML pages

There are basically two types of HTML pages:

1. *Pages for individual objects* (parameters, sets, variables, equations, models, acronyms, functions, files and source files)
2. *Summary pages for classes of objects*, per project in alphabetical order. An additional page lists all set elements.

The *pages for the individual objects* carry the following information:

- Name of the object (e.g. DATA) and type (parameter, set, variable etc.)
- Long text description as given in GAMS declaration
- Domain information, as hyperlinks to the domain sets.
- In which files and for which projects (as capmod, capreg ...) the object is declared, defined, assigned and referenced.
- In the case of sets: derived subsets, and objects where the set is used as a domain. Elements of the sets and the subsets.
- In the case of source files: which symbols are declared, defined, assigned and references in the files. Information from SVN (version, local modification, s out-of-date with server). Included files, and files which include the file. For GDX files: where included and included by which file.
- “Tagged” in-line comments taken from the source code files, what is called “doclet” (see e.g. [Sun document about how to write Doc comment for JavaDoc](#)) in JAVADOC, see .e.g. [wikipedia article](#)

Tagged in-line comments

Similar to the element comments underlying JAVADOC (see e.g.), “tagged” in-line comments are proposed for the inline code of CAPRI (sometimes called “doclets”, e.g.). The following shows a possible implementation which is currently already operational:

```
...
* @start
* @author W.Britz
* @docRef perfect aggregation of production
* @seeAlso gams\capreg\cons_levels.gms
MODEL CONS_LEVELS / ...
..
```

In the example above, the REF file will comprise the information were the model CONS_LEVEL will be declared, and the JAVA application will search backwards for lines

with tags (@..). Those tags will be linked to the object, and integrated in the HTML pages. The @start tag must be used to declare the start of the documentation for the current symbol.

Refactoring Consequences for Gams Code

1. All files should carry a header which reports the purpose of the file, and if possible, an author (contact person). The file header should start with a line of stars and end with a line of stares. All lines in the file header should start with a “*”.
2. The use of *\$GDXIN* is discouraged as it may load in huge amounts of data at run-time. Equally, it will load element codes comprised in the data sets even if they are not referenced later in the code. The only exemption is when the symbol must be loaded at run time as in case of META data, instead, *execute_load* should be used.
3. An “*\$IF NOT EXIST myFile \$ ABORT myFile is missing*” statement should be in the line before “*execute_load myFile someSymbols*”.
4. An “*\$IF EXIST myFile \$ LOG myFile will be overwritten*” statement should be in the line before *execute_unload myFile someSymbols*”.
5. All symbols should be declared with a clear long text description, i.e. statement in the style “SET A;” are discouraged.
6. Code in lengthy files should be moved into new files which are included so that a more modular structure is evolving. The new file should have a clearly defined and encapsulated task which is described in the file header.
7. Symbol declaration should where necessary be preceded by a “doclet” of the form
 - * @start or, alternative, a blank line
 - * @DocRef reference to the methodological documentation (optional)
 - *@ seeAlso reference to other file or symbol (optional)
 - * Any commentsDeclaration (as SET A “The alternative technologies per production activty” / T1,T2 /;
8. Symbols, especially when they are not widely used across programs should carry meaningful names.

Other recommendations arising from analysing the files are:

1. Single lines in the code should not exceed the size of a normal screen width when using medium sized fonts.

2. Indentation should be used to render the program structure defined by loops, if statements and the like more visible.
3. Especially tricky statements which use complex \$ operators, several cross-sets and the like should be preceded by some explanatory comments.
4. Symbols which are only used locally in a file should be deleted from memory by “option kill= ...”⁴.
5. Before defining a new set one should check if not the very same collection of elements is not already defined.
6. Lengthy data tables should be moved into a.gdx file to reduce the number of code lines.
7. Data should be accompanied by meta data.

Clearly, the standards and recommendations require further discussion inside the network, and must become part of a programming guide.

⁴ A feature request was sent to GAMS to support local scope, so that a symbol can be declared local for a file and subdirectory, and the compiler will raise an error when it is used out of scope.

General overview

Project analyzed

Jump to list for specific project

Selection of symbols by type and project

Alphabetical list of symbols with domain information and description, links to symbol page

Example for a Symbol page

Name with Domains

Files where the symbol is declared

Opens declaration in Editor

Projects where the declaration is found

Variable AREQ	
Name	AREQ(*,*A,*)
Type	Variable
Description	Requirements per head and day
Number of dimensions	4
Used by:	capreg
Used by:	capmod

AREQ is declared in:

[GAMS\SUPPLY\SUPPLY_MODEL.GMS \(capreg;capmod;\) Edit](#)

AREQ is assigned in:

[GAMS\FEED\FEDTRM_CAL.GMS \(capreg;capmod;\)](#)
[GAMS\CAPREG.GMS \(capreg;\)](#)
[GAMS\BASELINE\FEDTRM_PREP.GMS \(capmod;\)](#)

AREQ is referenced in:

[GAMS\SUPPLY\SUPPLY_MODEL.GMS \(capreg;capmod;\)](#)
[GAMS\FEED\FEDTRM_MOD.GMS \(capreg;capmod;\)](#)
[GAMS\FEED\FEDTRM_CAL.GMS \(capreg;capmod;\)](#)
[GAMS\CAPREG.GMS \(capreg;\)](#)
[GAMS\BASELINE\FEDTRM_FIN.GMS \(capmod;\)](#)

Example for a GamsSourceFile page

CAPRI technical documentation

Automatically generated from :
t:\britz\capri\gams\captrd.ref

open all | close all

- Types
- Parameters
- Sets
 - captrd
 - capreg
 - capmod
 - at least in one project
- Files
- Equations
- Variables
- Elements
 - captrd
 - capreg
 - capmod
 - at least in one project
- Models
- Acronyms
- Functions
- SourceFiles
 - captrd
 - capreg
 - capmod
 - at least in one project

[Top](#) | [Definitions](#) | [Assignments](#) | [References](#) | [Elements](#)

SourceFile DAT\ARM\WORPRICES.GMS

Name	DAT\ARM\WORPRICES.GMS
Type	SourceFile
Used by:	capreg
Used by:	capmod
SVN version of working copy	1603
SVN last committed version	1110
SVN last author who committed	alexanderg
SVN last changed date on server	Fri Nov 09 18:09:55 CET 2007
Current status	Normal

[Edit](#)

Declarations found in DAT\ARM\WORPRICES.GMS :

Name	Type	Description
WorPrices (capreg, capmod)	Parameter	

Definitions found in DAT\ARM\WORPRICES.GMS :

Name	Type	Description
WorPrices (capreg, capmod)	Parameter	

SVN information

Opens editor

Symbol usage in the file

Example for a page for the a set

CAPRI technical documentation

Automatically generated from :
t:\britz\capri\gams\captrd.ref

open all | close all

- Types
- Parameters
- Sets
 - captrd
 - capreg
 - capmod
 - at least in one project
- Files
- Equations
- Variables
- Elements
 - captrd
 - capreg
 - capmod
 - at least in one project
- Models
- Acronyms
- Functions
- SourceFiles
 - captrd
 - capreg
 - capmod
 - at least in one project

[Top](#) | [Definitions](#) | [Assignments](#) | [References](#) | [Ele](#)

Set MAACT

Name	MAACT(MPACT)
Type	Set
Description	Animal production activities comprised in model
Number of dimensions	1
Used by:	captrd
Used by:	capreg
Used by:	capmod

Subsets based on set MAACT :

open all | close all

- subsets
 - elements
 - MAACT
 - elements
 - DCOL DCOH BULL BULH HEIL HEIH
 - CALF
 - CALR
 - MRUMI
 - MNRUMI
 - RUMILK
 - REQM_TO_MAACT
 - MAACT_TO_REQM
 - DCOW
 - HEIF
 - BULF

Superset

Elements of the current

Subset

File list

CAPRI technical documentation

Automatically generated from :

- t:\britz\capri\gams\captrd.ref
- t:\britz\capri\gams\capreg.ref
- t:\britz\capri\gams\capmod.ref

at 21-07-2008 09:30:24

Used in project : 'capreg'

open all | close all

open

D

- DAT\ARM\WORPRICES.GMS
- DAT\BIOFUEL\BIO_FUEL_PROD_DATA.GMS
- DAT\BIOFUEL\COEFF.GMS
- DAT\CAPREG\UKEXPERT.DAT
- DAT\FEED\FAT.DAT
- DAT\FEED\FEDCOF.DAT
- DAT\FEED\PORKREQ.DAT
- DAT\FEED\SHEEP_GOAT.DAT
- DAT\FERT\FAO_FERT.GMS
- DAT\FERT\IFA_DAT.GMS
- DAT\INPUTS\BAYER_INPUT.GMS
- DAT\INPUTS\EST_FROM_FADN
- DAT\INPUTS\ZSETTY.GMS

G

- GAMS\BIOFUEL\KIM_EXPOST.GMS
- GAMS\CAPREG.GMS Modified**
- GAMS\CAPREG\AGGREG_DATA.GMS
- GAMS\CAPREG\CAL_SERVER.GMS
- GAMS\CAPREG\CONS_INPUT.GMS
- GAMS\CAPREG\CONS_LEVLS.GMS Modified**
- GAMS\CAPREG\CONS_SETA.GMS Modified**
- GAMS\CAPREG\CONS_YIELDS.GMS
- GAMS\CAPREG\DEF_CRPR_COR.GMS
- GAMS\CAPREG\DEF_EAA.GMS
- GAMS\CAPREG\FSS_SETS.GMS
- GAMS\CAPREG\HP_FILTER.GMS
- GAMS\CAPREG\MAP_FROM_REGIO.GMS
- GAMS\CAPREG\MAP_POL.GMS
- GAMS\CAPREG\PRICE_YANI.GMS
- GAMS\CAPREG\REGIO_SETS.GMS

Files which are not in normal SVN state or where a newer version is available on the server are highlighted

HTML link to page for file

Set element list

CAPRI technical documentation

Automatically generated from :

- t:\britz\capri\gams\captrd.ref
- t:\britz\capri\gams\capreg.ref
- t:\britz\capri\gams\capmod.ref

at 21-07-2008 09:30:24

Used in project : 'capmod'

open all | close all

open

A

B

C

D

E

EFUL_heavy : ITEMS(*) SOIL_TYPES(*) SOIL_TYPES1(*)

EFUL_light : ITEMS(*) SOIL_TYPES(*) SOIL_TYPES1(*)

EFUL_medium : ITEMS(*) SOIL_TYPES(*) SOIL_TYPES1(*)

ENER_CONTENT : ENER_ITEM(*) ENER_CO2_CONTENT(*) ITEMS(*) COPY_SET(*) COPY_SET1(*) COPY_SET2(*)

ENER_LEVEL : ITEMS(*) ITEMS_1(*)

ENER_TOTAL : ENER_ITEM(*) COPY_SET(*) COPY_SET1(*) COPY_SET2(*)

ENER_kg : ITEMS(*) ITEMS_1(*)

ENER_sqm : ITEMS(*)

EN_SEED : PROC1(*) Ener_Type(*) Ener_Type_ANI(*) Ener_proc(*)

EXP : YEARS(*)

EXPORT : ENER_ITEM(*) COPY_SET(*) COPY_SET1(*) COPY_SET2(*) COPY_SET3(*)

Elec_cons : ITEMS(*) DRY_ENER(ITEMS)

Ener_ds : ITEMS(*)

Ener_fs : ITEMS(*)

Ener_weight : ITEMS(*)

Exp_lifetime : ITEMS(*)

F

G

H

I

K

L

M

N

O

P

Name of element

Sets comprising the elements with HTML link

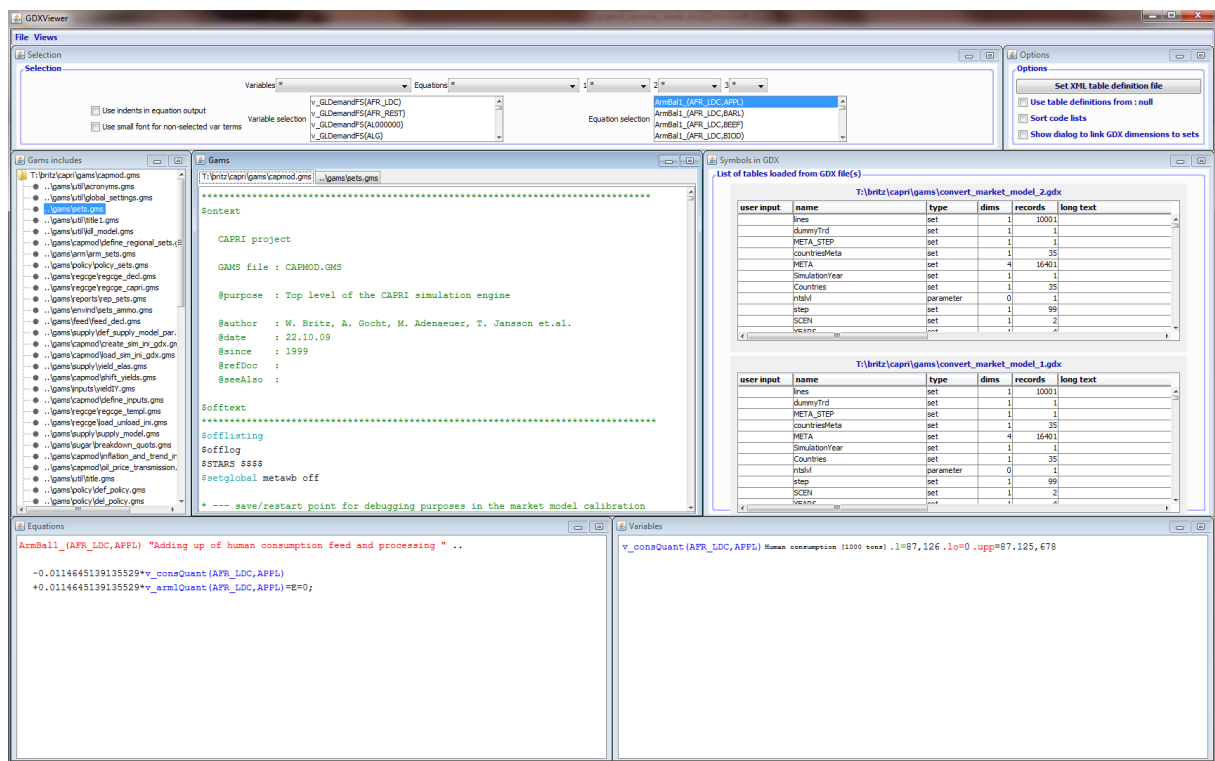
Utilities: Equation and variable viewer

Background and motivation

Complex GAMS code and related models such as the market model of CAPRI with its ~70.000 equations and variables are very hard to debug. The model listing produced by GAMS from such a model is quite long, and filtering out e.g. all lines belonging to a certain market not possible with all editors. Further on, linking the listing to the equation structure of the model is also far from easy. That paper describes a utility linked into the GGIG, the GUI generator used by CAPRI, which supports working with large model outputs (and more generally complex GAMS projects with many symbols).

The new tool also incorporates the functionality of the existing GDXViewer comprised in the GUI. The tool can be used to “track” changes to symbols in the GAMS code by producing a range of GDX files (currently up to 5) at different execution points.

An overview on the viewer



The viewer comprises a number of windows

- **Selection:** A window where variables and equations can be selected, and filters text for variables and equations defined
- **Options:** A window with options for the GDXViewer

- **Gams includes:** A tree view of the GAMS includes
- **GAMS:** A window showing the GAMS code
- **Symbol from GDY:** A view on a current selected GAMS symbol
- **Equations:** A view on the current selected equation(s), in linearized form
- **Variables:** A view on the currently selected variables

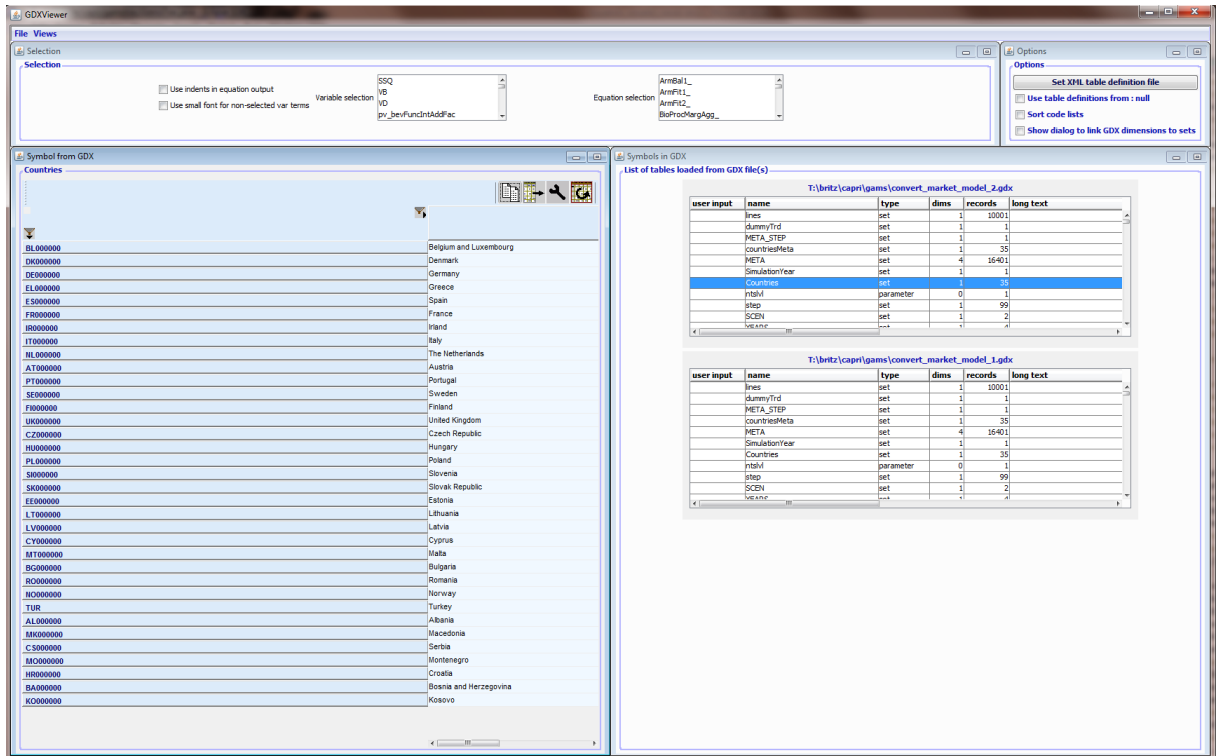
The windows can be dragged, resized and minimized.

Producing input for the view with GAMS

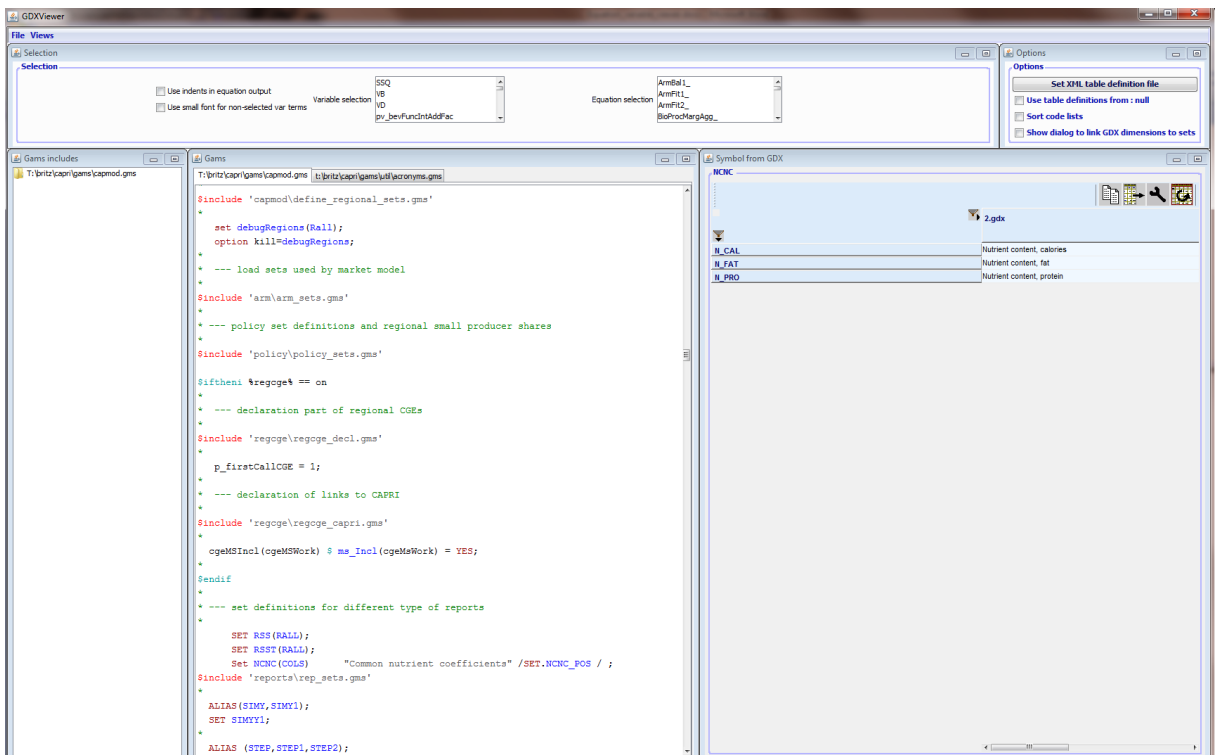
The viewer can be used in different configurations, which are available via the “File\run” menu:



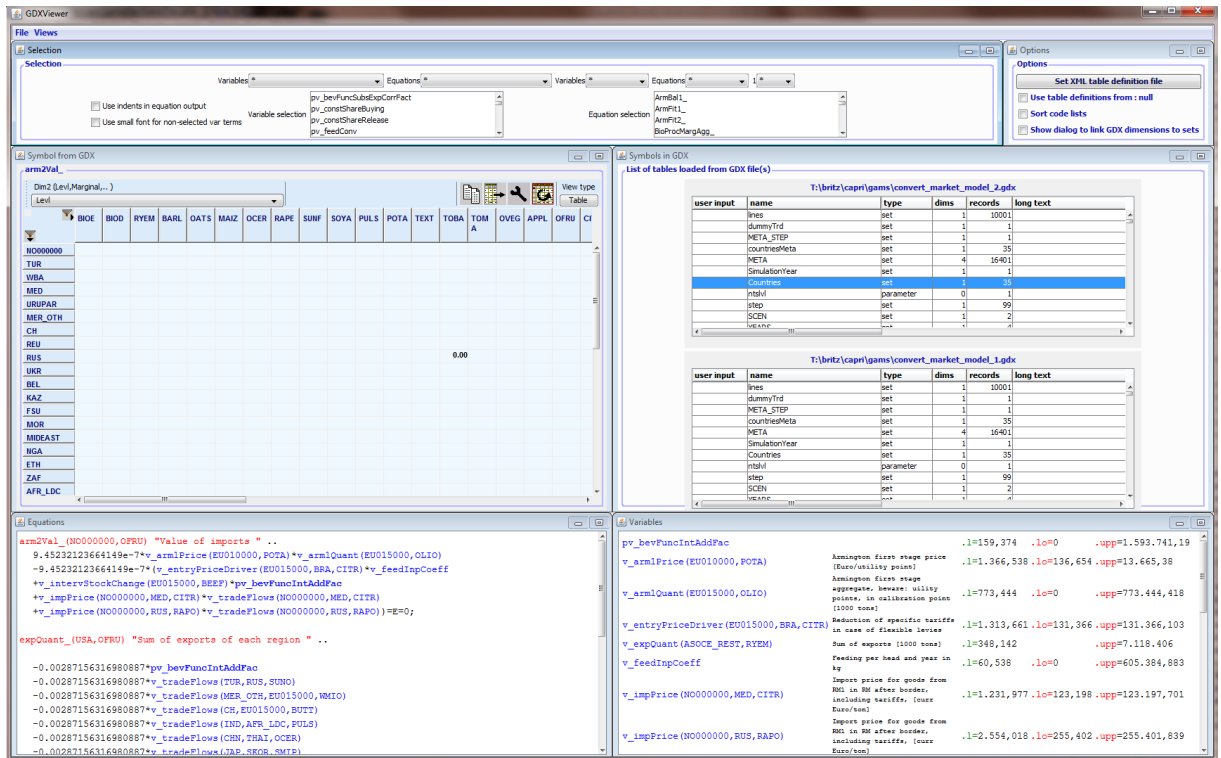
- “**Load only GDY files into viewer**” comprises basically the same functionality as the existing GDYViewer. However, the selection panel and the view on the symbol are both visible at the same time.



- **“Load GAMS files (and if existing GDX files) into viewer”**: With a GAMS file and one or several GDX file: browse the source code and click on highlighted symbols to load them from the GDX file(s). That is basically a GDX viewer linked to the GAMS code. The GDX files can be produced in GAMS with “execute_unload someName.gdx;” which will dump all GAMS symbols into a GDX file.



- “Load convert output into viewer”: with Convert output only. Convert is a “solver” shipped with GAMS which generates a linearized version of a model with obfuscated variable and equation names, e.g. to ship it to a solver developer for testing. Convert can produce a dictionary file which allows to link it to the obfuscated names.



In that case, it will replace working with the output of solprint=1 in combination with limrow/limcol. The necessary output can be produced as seen in the example below from “arm\prep_market.gms” in the CAPRI code:

```

$iftheni %abortAfterFirstMarketSolve% == on
*
* --- generate convert output to be accessed with interface
*
option DNLP=Convert;
execute "echo gams convert_market_model.gms > convert.opt";
execute "echo dict convert_market_model.txt >> convert.opt";

m_globMarket.solprint = 2;
m_globMarket.optfile = 1;
m_globMarket.limrow = 0;
m_globMarket.limcol = 0;
SOLVE m_globMarket USING DNLP Minimizing v_dummy;
execute_unload "convert_market_model.gdx";

abort "Program stopped after first test solve of market model, convert output generated",data;
$endif
    
```

Includes

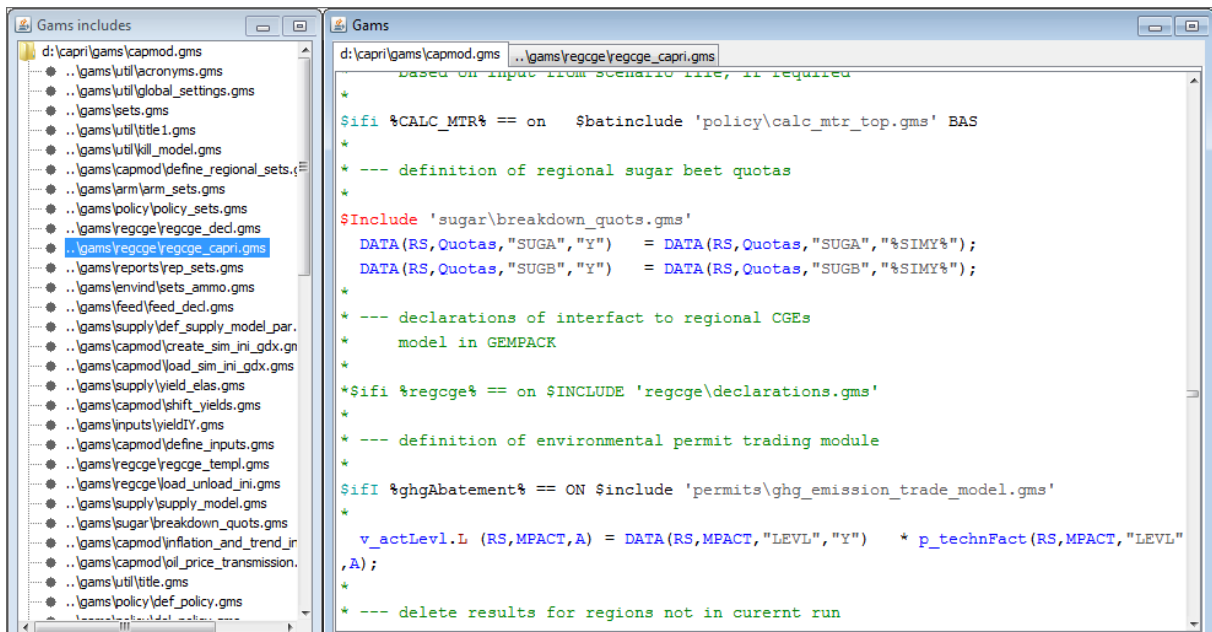
When the viewer is started, only the code for selected GAMS file is loaded in the window titled “GAMS”. The window titled “GAMS includes” will show all the (direct) includes used

by that GAMS file which could be successfully read (globals are not yet treated, it is planned to merge the tool with the HTML doc generation such that includes reflect the compilation stage of the GAMS code.).

In order to open included files:

- either double click on a node in the “GAMS includes” tree view
- or on a “\$(bat)include” in the GAMS code view marked in red

As seen below, the GAMS code in new file will be opened in an additional tab. In case the new file comprises includes, they will be added to its node.



Loading symbols

The GAMS code highlights in blue all symbols (sets, parameter, variables, and equations) found in the GDX file. These symbols can be opened in the GDX viewer by a mouse click in the GAMS file viewer. If several GDX files are provided, the symbol will be loaded from all the GDX files where there are non-empty records. That allows for a very rapid inspection of the data.

Alternatively, select the “GDX symbol table” from the “Views” menu :

Table 1: Symbols in GDX (convert_market_model_2.gdx)

user input	name	type	dims	records	long text
	lines	set	1	10001	
	dummyTrd	set	1	1	
	META_STEP	set	1	1	
	countriesMeta	set	1	35	
	META	set	4	16401	
	SimulationYear	set	1	1	
	Countries	set	1	35	
	ntslvl	parameter	0	1	
	step	set	1	99	
	SCEN	set	1	2	
	VEANE	set	1	1	

Table 2: Symbols in GDX (convert_market_model_1.gdx)

user input	name	type	dims	records	long text
	lines	set	1	10001	
	dummyTrd	set	1	1	
	META_STEP	set	1	1	
	countriesMeta	set	1	35	
	META	set	4	16401	
	SimulationYear	set	1	1	
	Countries	set	1	35	
	ntslvl	parameter	0	1	
	step	set	1	99	
	SCEN	set	1	2	
	VEANE	set	1	1	

In table above, symbols can be selected by the mouse. As in the GDXViewer, one might select several symbols with the same number of dimensions.

Working with the equation and variable viewer

Equations and variables can be loaded in the equation and variable viewers (the two windows in the lower part of the main window) by working with the selection boxes (see tab selection). These windows are thought as a replacement of inspecting the equation listings with an editor.

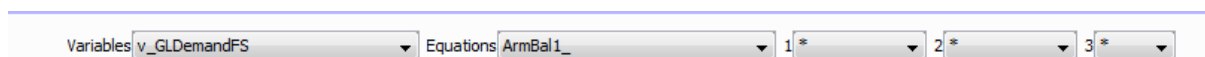
If an equation is selected with the **“equation selection” control**, the equation window will show it in “linearized” form, i.e. any non-linear functions and interactions terms with other variables are converted in a constant. The variable window will report the level and the lower and upper bound for all variables found in the equation. The user can add (or remove) equations with the control as well.

If a variable is selected with the **“variable selection” control**, the variable window will report the level and the lower and upper bound for the selected variable(s). The user can add (or remove) variables with the control as well.

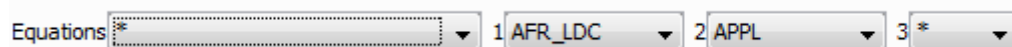
Double clicking on a variable in the equation window will show that variable in the variable window as well as load it in the GDX viewer.

Double clicking on a variable in the variable window will load all the equations comprising the variable, and also show in the variable window all the variables found in these equations.

The first line of selection boxes shown below first allows to select all instances of a variable or an equation:



The other boxes show all items found on the dimension of all symbols and allow to filter further. In order to see e.g. all equations which have on the first dimension “AFR_LDC” and on the second “APPL”, put the selection controls as seen below:



That produces an equation output as seen below:

```

ArmBall_(AFR_LDC,APPL) "Adding up of human consumption feed and processing " ..
-0.0114645139135529*v_consQuant(AFR_LDC,APPL)
+0.0114645139135529*v_armlQuant(AFR_LDC,APPL)=E=0;

CPri_(AFR_LDC,APPL) "Consumer price linked with fixed margin to first stage price " ..
-0.00187949947079955*v_armlPrice(AFR_LDC,APPL)
+0.00187949947079955*v_consPrice(AFR_LDC,APPL)=E=0.284525356896553;

GLDemandGiS_(AFR_LDC,APPL) "Part of GL demand function " ..
-0.521399433800188*(0.508934628613077
+1.14603816746872*sqrt(0.00188303864092867*v_consPrice(AFR_LDC,MAIZ))
+1.37181255843937*sqrt(0.00188303864092867*v_consPrice(AFR_LDC,OCER))

```

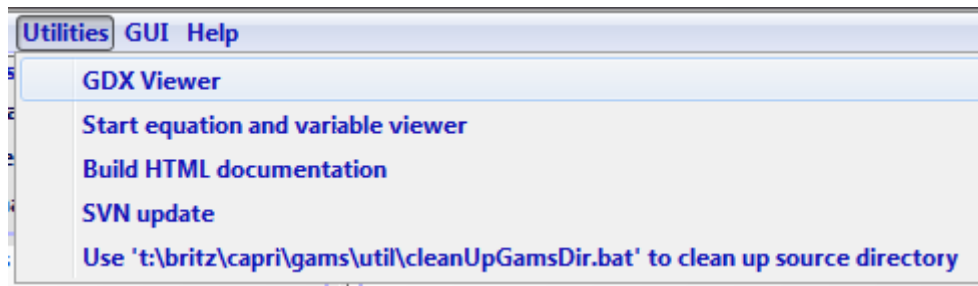
Note: currently, the viewer will not show more than 100 equations and 1000 variables simultaneously.

Utilities: Gdx-file(s) viewer

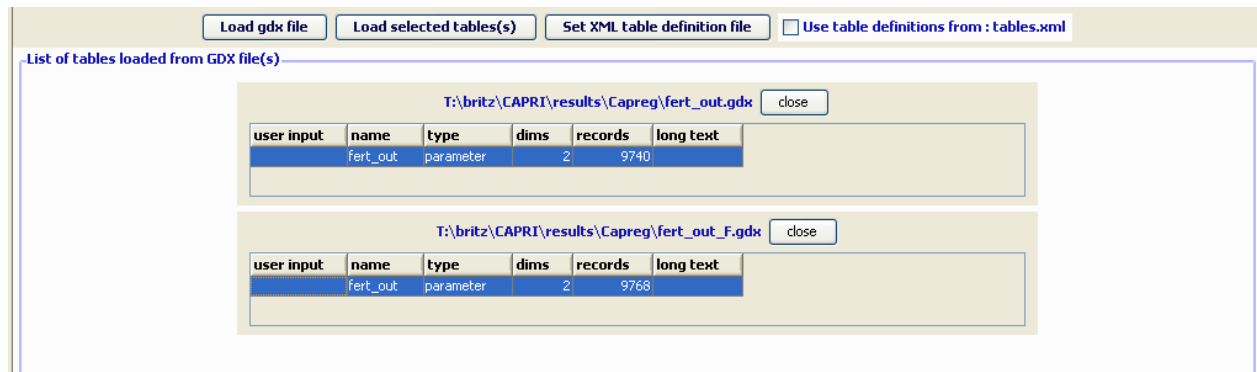
GDX-files are generated by GAMS and typically serve either an exchange format between different GAMS applications, or for exploitation purposes as the GAMS-IDE comprises a view for GDX-files. Further tools for GDX-files are available from GAMS company and are described in different documents. In opposite to listings generated by GAMS programs, the GDX files store the data in full numerical precision in an internal format.

The new CAPRI version passes information from one task to the next with the help of GDX files, so generates CoCo a gdx files with the time series at national level, which is read by CAPREG. And the regional time series generated by CAPREG are inputted by the trend projection tool CAPTRD. These gdx files are accessed when the different tasks of “Data base exploitation” are chosen. The user has on top the possibility to load one or several tables from one or several freely chosen gdx files.

The GDX exploitation utility can be reached via the menu bar:



Graph: Panel to GDV file exploitation



When the task “exploit gdx files” is selected by pressing the related button, four buttons are shown in the task panel. The first one, labelled “load gdx files” will open a file selection menu when pressed. When the ok button of the dialogue is operated, the content of the gdx file is partially loaded, and a table is added to the right upper window of the application showing the parameters and sets comprised in the gdx files, along with their number of dimensions and records. When the close button next to the table is pressed, the table is deleted. Pressing the “load gdx file” again will add more tables.

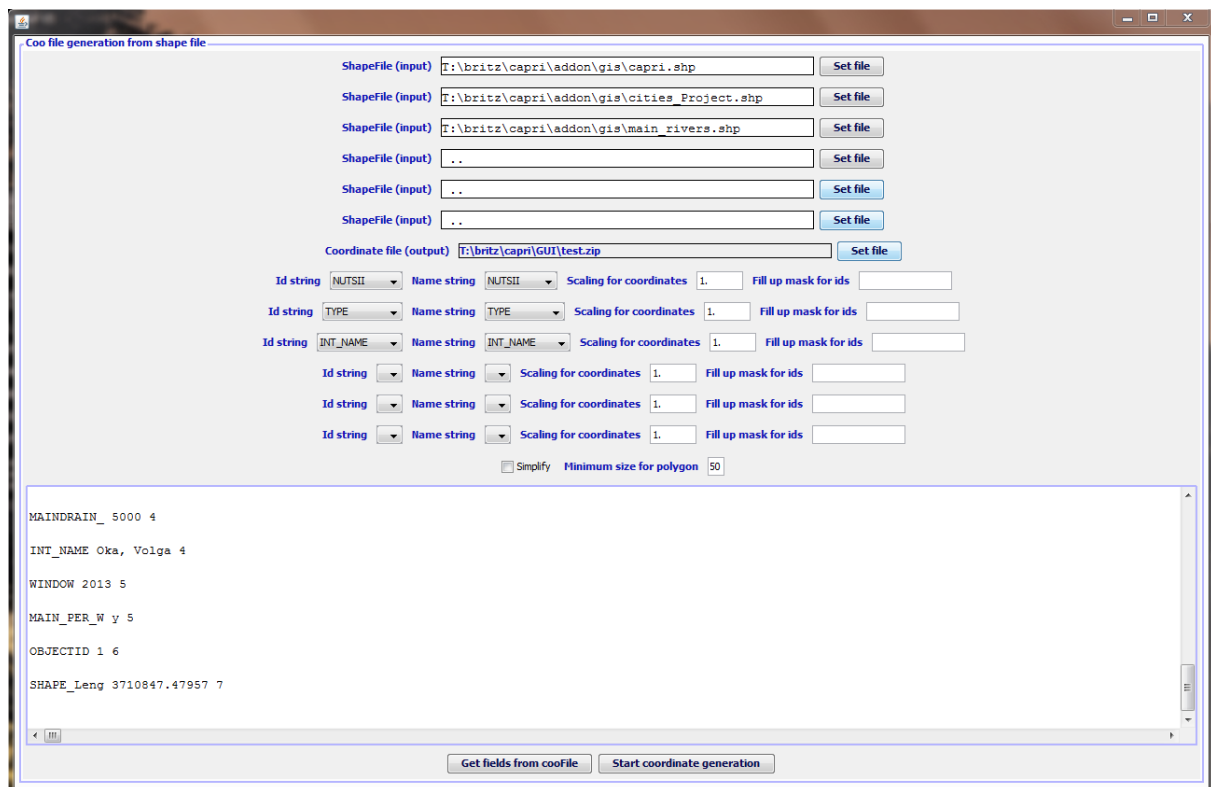
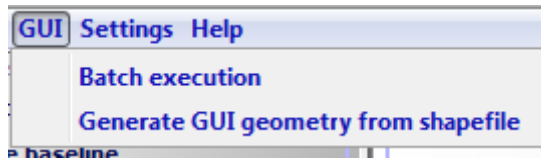
One parameter from each table may be selected (pressing the “ctrl” key when clicking with the mouse de-selects). If several parameters from one file need to be loaded, the user may open the same file several time.

The content of the different parameters is merged together, and the parameters themselves span an additional data dimension. If the user does not provide input in the first column of the tables labelled “user input”, the program will generate names automatically. The data loaded are shown in the table tool described above.

The user can use view definitions stored in a XML file to the tables by pressing the enabling the “Use table definitions from ...” tick box, and may use the “Set XML table definition file” button to change the file to use.

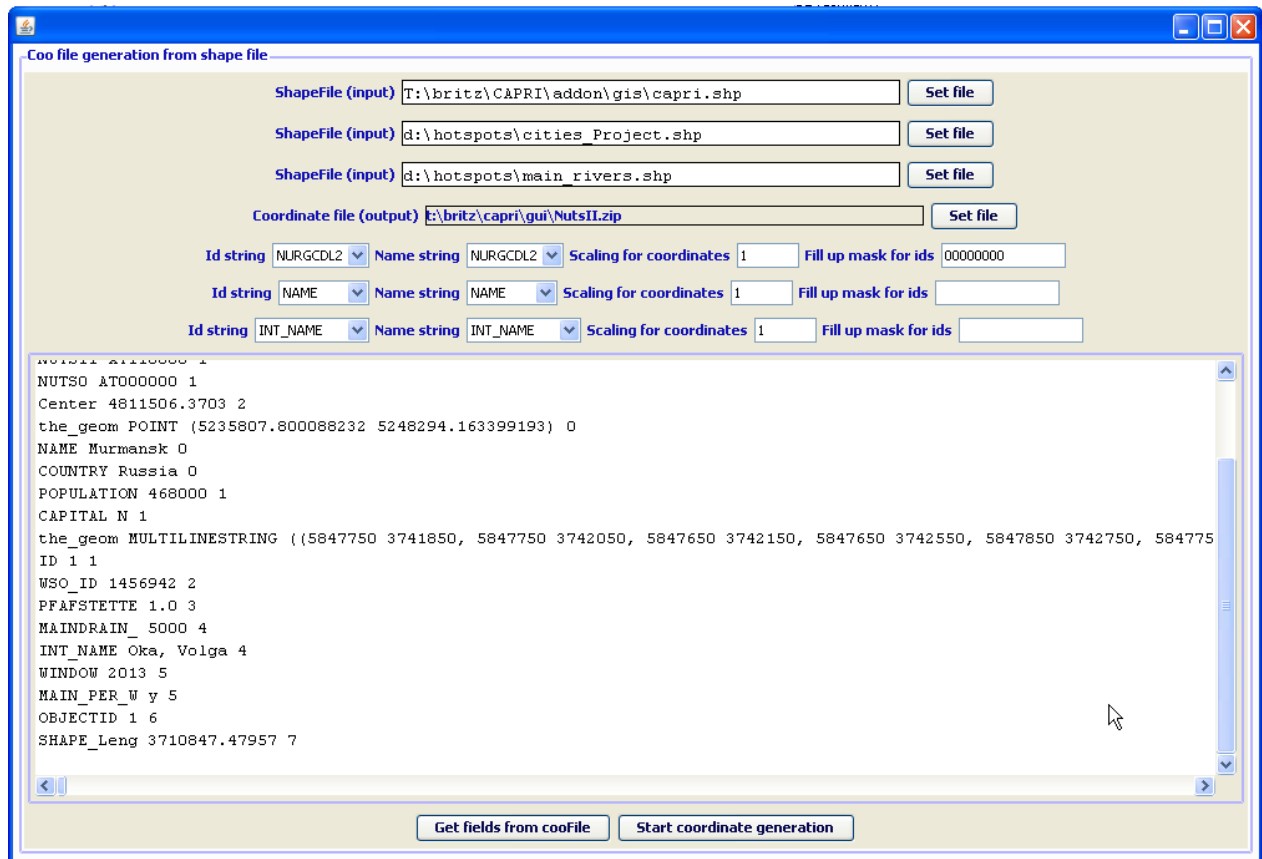
Utilities: Generating coordinate files for the exploitations tools from shapefiles

The exploitations tools use a proprietary format to store coordinate files. The utility allows to build from shape files a file in that proprietary format.



Note: Files and settings shown above are the ones used to generate the NUTS II map in CAPRI, the shape files can be found in the “addon\gis” folder

As a first step, the shapefiles must be analyzed by using the “Get field from coofile” button:



Once, that is done, the fields from the shapefiles used for keys and the long texts can be chosen, and some other settings. The interface will assume treat line strings as river, points as cities and polygons as regions.

The utility assumes that all shapefiles are in the same coordinate system and will simply store the coordinates one to one (applying where set a scaling factor) in the internal format used by GGIG.

Analysis differences in GAMS based data using GGIG

Background

In result analysis such as when comparing scenarios, but also when comparing different releases of data sets against each other, one frequently wants to see only those values with larger changes. When working with values generated by GAMS, one has different ways to proceed.

If the two data sets to compare are comprised in GDX files, one can use the GDXDIFF utility from GAMS. The disadvantage is that GDXDIFF does not have information about the

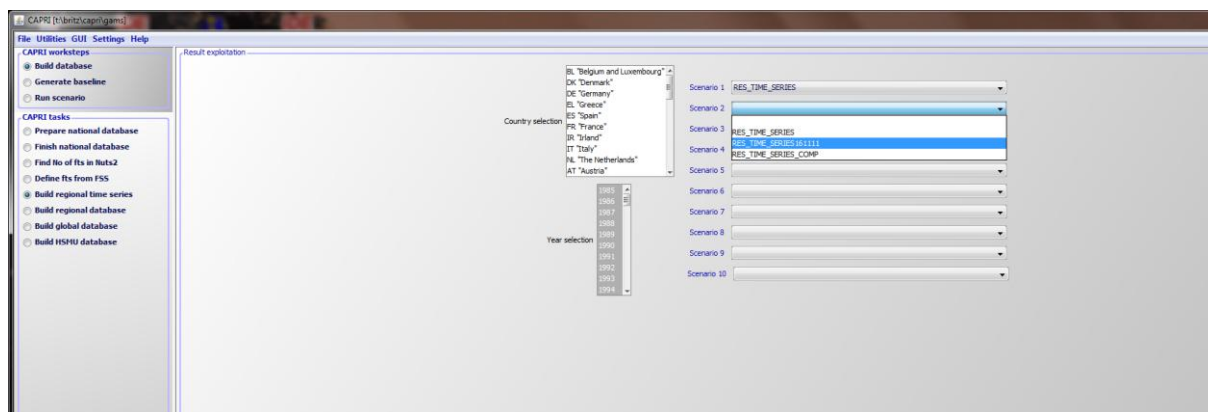
logical structure of the data or its content, such that it might be cumbersome to filter our large absolute or relative changes which matter.

The other extreme is to write a GAMS program which uses rules about the importance of data items to concentrate on changes which matter. In the CAPRI-RD project, to give an example, outlier statistics for many time series for EU Member States were calculated, leading to thousands of potentially “suspicious” values. It is clearly impossible to check manually each and every case, so that algorithms have to deal with the majority of the cases. The expensive manual checks have to concentrate on the items which are deemed important. Hence, by using e.g. national and EU crop shares, animal stocking densities and shares on sectoral revenues, a matrix of importance was constructed which assign a numeric indicator to each time series. A combination of that importance metric for a time series and its outlier statistics combined with a threshold delivers then the potential outliers to compare manually.

The following will describe a third way to proceed based on the in-built functionalities of the exploitation tools.

Comparing two data sets in GGIG, example from CAPRI

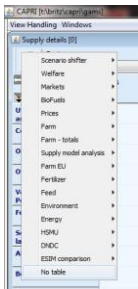
Generally, each task in CAPRI also allows viewing its results and selecting “Scenarios”. The example below show the task “Build regional time series”. On the disk, several versions where located, and these can be compared as if they were scenario. If one e.g. wants to compare the current version against the trunk or an earlier release, one can rename the current one .e.g. to “..._current.gdx” and then use “update” or “update to revision” to download the version to compare to from the server.



There are now different way how to proceed.

GGIG as GDXDIFF

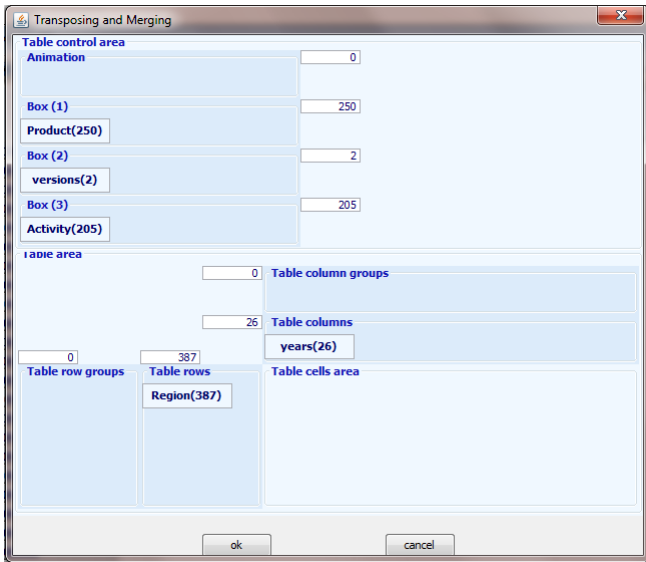
One can mimic the behavior of GDXDIFF by using as the view “No table”



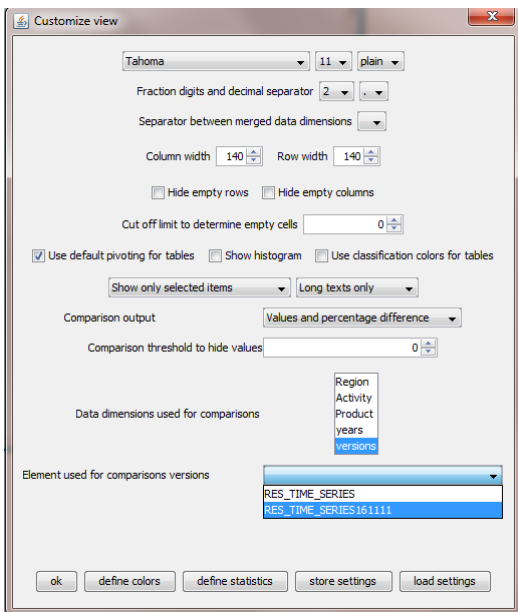
That allows scrolling through any cell, but requires familiarity with the codes and the structure of the data set one analyzes. One might end up with a view as below ... the first to do is to pivot.

Region	Year	BIOD	SWHE	DWHE	RYEH	BARL	OATS	MAIZ	OCER	RAPE	SURF
SWHE	1984										
DWHE	1984										
RYEH	1984										
BARL	1984										
OATS	1984										
MAIZ	1984										
OCER	1984										
RAPE	1984										
SURF	1984										
SOYA	1984										
OOIL	1984										
OIND	1984										
MBRS	1984										
FLOW	1984										
OCRO	1984										
NECR	1984										
HAIF	1984										
RODF	1984										
OFAR	1984										
GRAE	1984										
GRAB	1984										
PARI	1984										
OLIV	1984										
PULS	1984										
POTA	1984										
SUGB	1984										
TEXT	1984										
TOBA	1984										
TOHA	1984										
OVEG	1984										
APPL	1984										
OFRU	1984										
CTRR	1984										
TAGR	1984										
TABO	1984										
TWIN	1984										

Imagine you want to check acreages across the regions. A good way to proceed is to put the regions in the rows and the years in the columns as shown below.



Next, one uses the option dialogue to configure the view such that e.g. percentage differences against the old version are displayed:

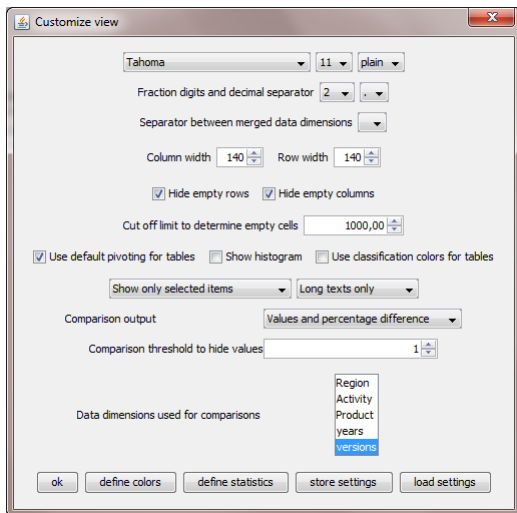


That produces a view as seen below ... but it is clearly not inviting to scroll now through about thirty years and almost 400 regions.

Product	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994
K0000000											
BL000000	191.82	193.53	193.94	198.26	217.70	219.46	218.99	213.93	214.54	211.95	
DK000000	318.33	326.64	341.43	381.43	296.88	424.96	516.55	594.20	559.60	601.09	
DE000000						2383.88	2383.88	2474.04	2621.66	2408.91	
EL000000	506.60	438.80	387.80	369.70	357.11	350.51	309.33	280.74	309.78	307.67	
ES000000	2038.83	1817.29	1898.56	2012.36	2163.35	2100.96	1736.56	1686.57	1526.07	1338.84	
FR000000	5925.58	4695.08	4600.89	4611.01	4500.03	4820.48	4868.61	4770.03	4769.83	4440.35	
IR000000	72.19	70.65	72.75	56.38	58.60	60.70	70.45	86.56	91.06	79.20	
IT000000	1375.60	1213.49	1190.63	1133.53	1050.04	1104.07	1009.22	953.70	963.35	872.49	

Here, three combined options in the dialogue can help:

- (1) Use hide empty and empty columns to throw out missing values or hidden cells.
- (2) Use a (approximate) cut-off for the value to show, e.g. start only with acreage > 1000 [1000 ha]
- (3) Hide cells where the difference is below a threshold, e.g. 1%



That delivers a much more usable view at important changes:

An interesting option here is to use the “GTAP” difference which is defined as

$$m = [\log(x) - \log(y)] \text{abs}(x - y)$$

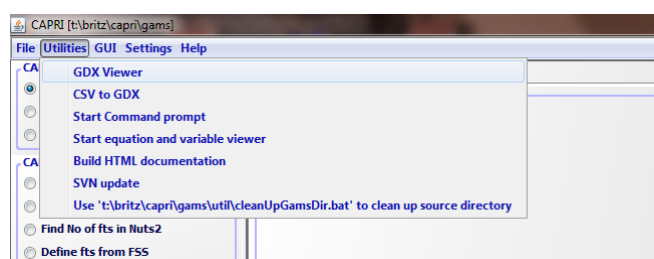
The first term looks at relative differences which are weighted with absolute ones.

Using the table definitions

For those not familiar with the codes, it might be easier to work with the pre-defined tables. As those tables are not always constructed e.g. to be used with time series, it might be required to pivot them as well.

Region	versions										
	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994
Utilized agricultural area											
Oleseeds									1073.71	1489.72	1752.25
Other arable crops	1094.70	1045.33	1036.32	1029.74	1062.57	912.60	927.55	918.26	838.66	825.81	725.81
Fodder activities											9275.60
Set aside and fallow land										3560.38	
All cattle activities	3524.90	3699.85	3676.01	3692.47	3725.92	3786.42	3808.13	3845.82	3886.38	3759.79	3759.79
Beef meat activities	607.87	752.53	818.96	928.83	1007.36	1049.77	1116.77	1210.29	1320.02	1455.83	1455.83
Other animals	4805.70	4598.54	4774.46	5476.63	6087.76	6257.88	6464.73	6691.29	6946.60	7028.16	7028.16
Utilized agricultural area											
Oleseeds									1073.71	1489.72	1752.25
Rape									11.96		16.81

Comparing two GDX files with GGIG



GGIG also comprises a GDX viewer:

Into which several GDX files can be loaded and compared as discussed above.

Index

- | | | | |
|--------------------------------|-----|------------------------------------|----|
| Batch execution | 103 | Clipboard export | 42 |
| Clipboard export | 32 | Cummulative distribution | 49 |
| Column and row selection | 24 | Deviation renderer | 50 |
| Drop-down boxes for selections | 24 | Export to file | 41 |
| Equation and variable viewer | 117 | Histograms | 48 |
| Export to file | 32 | Line and point charts | 43 |
| Flow maps | 54 | Markov charts | 53 |
| GDX viewer | 123 | Pie charts | 45 |
| Generate GAMS documentation | 110 | Spider plots | 46 |
| Generating coordinate files | 125 | Histogram | 30 |
| Graphics | 38 | legend | |
| Bar charts | 42 | continuous linear scaling | 67 |
| Box and whisker charts | 47 | continuous logarithmic scaling bar | 68 |

- Machine learning 89
- Maps
 - Classification
 - Area weighted classification 60
 - Classification method 60
 - Color table 63
 - Equal interval 61
 - Excluding zeros from classification 60
 - Manual classification 62
 - Mean standard dev 62
 - Natural breaks 61
 - Nested mean 62
 - Quantile 61
 - Clipboard export 69
 - Drag 70
 - File export 69
 - Frequency diagram in map 63
 - Full extent 70
 - Getting data for specific polygons 70
 - Highlighting specific regions in the map 72
 - Histogram window 58
 - Info pointer 70
 - Info pointer and window 70
 - Legend 67
 - Regional labels in map 75
 - Rivers and cities 76
 - Shrinking polygons according to UAA share 59
 - Store settings 77
 - Title 69
 - Updating the map 75
 - Zoom in 69
 - Zoom out 70
 - Zooming 69
- Menubar
 - File menu 103
 - Settings menu 103
 - Utilities and GUI menu 103
- Meta data 100
- Numeric filter 73
- Pie chart maps 56
- Pivoting 26
- Predefined selection groups 26
- Scenario editor 99
- Select scenarios 21
- Set up
 - Exploitation tools 17
 - GAMS.EXE 11
 - SVN 12
 - Work directory 11
- Set up 11
 - Result directory 11
- Starting GAMS 18
- Tables
 - Drill down 32
 - Filtering 33
 - Outlier detection 36

Sorting	33	Hiding empty columns or rows	29
Statistics	34	Histogram window	29
View options	28	Number formatting and rounding	28
Comparison output	29	Percentage differences	29
Cut off limit to determine empty cells	29	View Selection	23
Fonts	28	View type selection (tables, graphs, maps)	26