



Parameter Sweep Wizard & Monitor

User Manual

Prepared by:

Rajmund Bocsi

Gabor Szemes

Laszlo Gulyas

Marton Ivanyi



December 2007, Budapest

Contents

Introduction	3
Agent-Based Modeling	3
MASS 3	
Parameter Sweep Wizard & Monitor	3
1 Parameter Sweep Wizard	5
1.1 Preferences	5
1.1.1 General Settings	6
1.1.2 Network Settings	6
1.2 Model selection Page	8
1.2.1 Extendable Class Path	8
1.2.2 Resources	9
1.2.3 Load Wizard Settings	10
1.3 Parameters Page	11
1.3.1 The Parameter Tree	12
1.3.2 Modify Parameter Settings	12
1.3.3 New Parameters	13
1.4 Data Collection Page	14
1.4.1 Stop condition	14
1.4.2 Recorders	15
1.5 Scripting	16
1.5.1 Statistics	16
1.5.2 Advanced scripting	17
1.6 Built-in Monitor	19
2 The Monitor Application	21
2.1 Configuration	21
2.2 Current Simulation Tab	22
2.3 Finished Simulations Tab	22
3 Frequently Asked Questions (FAQ)	24
4 Troubleshooting	25
5 Summary	26
6 References	27

Introduction

Agent-Based Modeling

Agent-based modeling is a branch of computer simulation. It models the individual, together with its imperfections (e.g., limited cognitive or computational abilities), its idiosyncrasies and personal interactions. The approach builds the model from 'the bottom-up', focusing mostly on micro rules and seeking to understand the emergence of macro behavior. Participatory simulation - a branch of agent-based simulation - is a methodology building on the synergy of human actors and artificial agents, excelling in the training and decision-making support areas. In participatory simulations some agents are controlled by users, while others are software governed.

MASS

The Multi-Agent Simulation Suite (MASS) is a software package intended to enable modelers to utilize the tools of agent-based simulation in various fields, without having to develop heavy programming skills.

MAS consists of four applications built around a simulation core. The simulation suite has its own core called the Multi-Agent Core (MAC), but it is also able to run on the popular Repast core. Being multi-core enables modelers to verify that results are core-independent, thus we plan to further develop this option. The Functional Agent-Based Language for Simulation (FABLES) is a programming language and its integrated programming environment specially designed for creating agent-based simulations. The Model Exploration Module (MEME) is a tool that enables orchestrating experiments, managing results and has support for their analysis. The PET (Participatory Extension) is an optional web-based environment for multi-agent and participatory simulations. The fourth element of MASS, the Visualization Package does not translate into a standalone application. It consists of the various implementations of charts and visualizations used in all the other software.

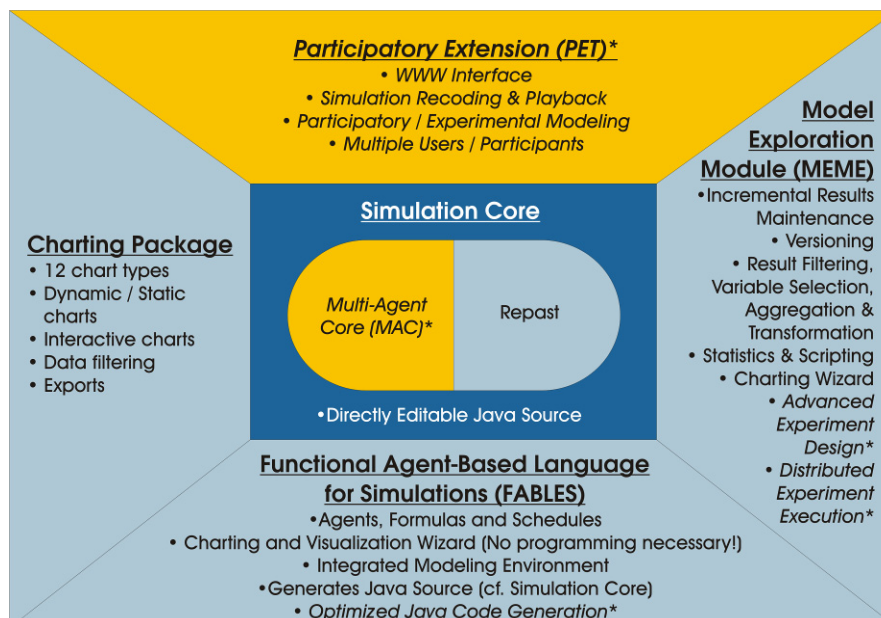


Figure 1 - The Multi-Agent Simulation Suite

Parameter Sweep Wizard & Monitor

The MASS/MEME Parameter Sweep Wizard supports Repast users to perform distributed mass parameter sweep experiments on a cluster or grid of computers using a point-and-

click graphical user interface. Users can create batch versions of their Repast models with the appropriate recorders and settings and run them through this application. It is also possible to run distributed parameter sweep experiments without having to write model versions explicitly for this purpose.

The MASS/MEME Monitor application enables users to follow the progress of current simulations running on the given cluster or grid of computers and download the output of finished simulations.

The current version of MASS/MEME Parameter Sweep Wizard & Monitor offers limited functionality that is going to be expanded in the near future. The following chapters explain the use of these functions.

1 Parameter Sweep Wizard

The MASS/MEME Parameter Sweep Wizard enables users to create batch versions of their Repast models in three steps and run them. The following sections describe these steps in detail. The figure below displays the opening screen (*Model selection* page).

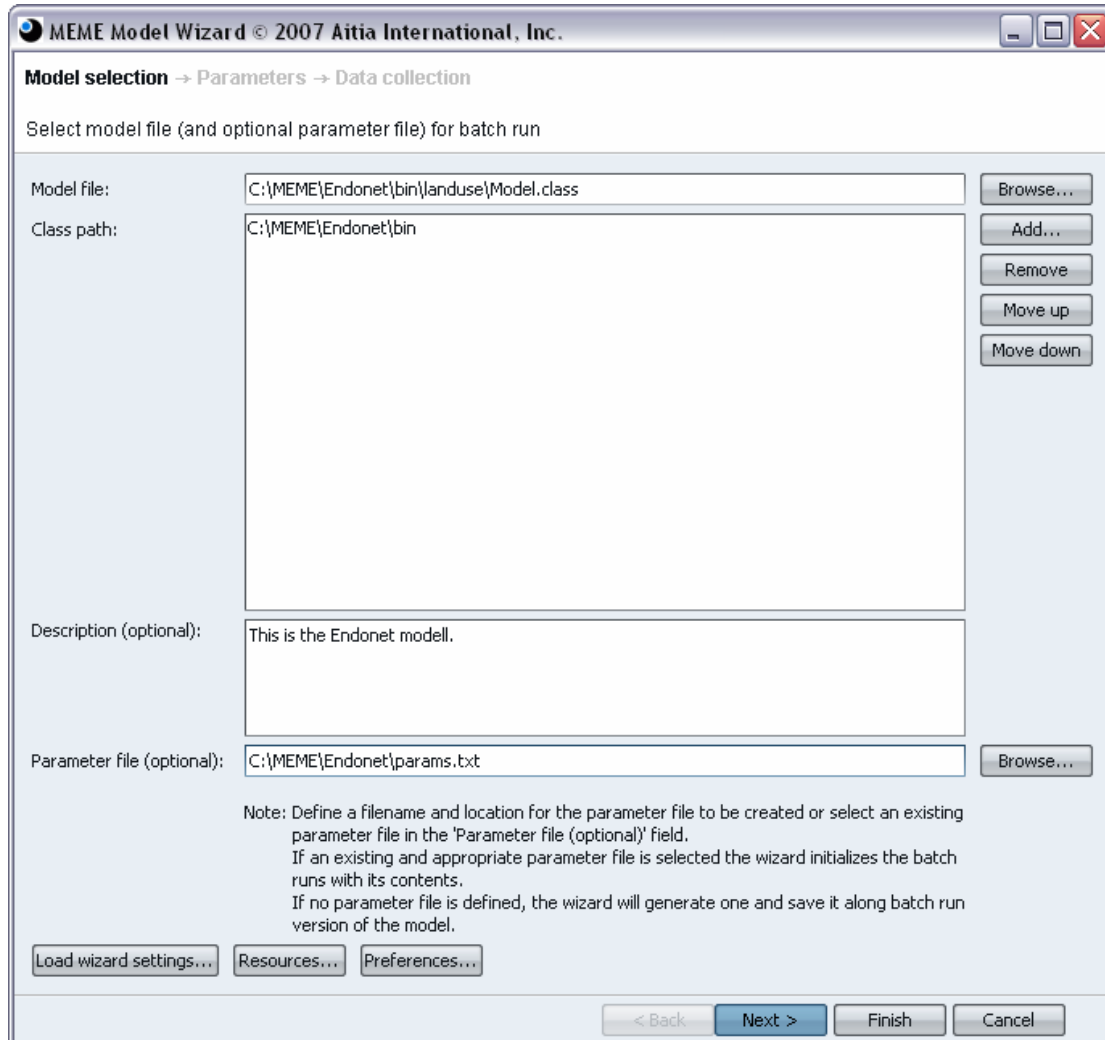


Figure 2 - Model selection page

The wizard runs the generated models on the local computer by default. For distributed runs the default settings have to be changed in the *Preferences* dialogue. Other options and settings can also be specified here. To open the *Preferences* dialogue press the *Preferences...* button on the *Model selection* page.

1.1 Preferences

The *Preferences* dialogue has two pages; the *Network* page dealing with distributed run settings and the *General* page for all other settings. To switch between the pages select their names in the tree on the left side of the dialogue.

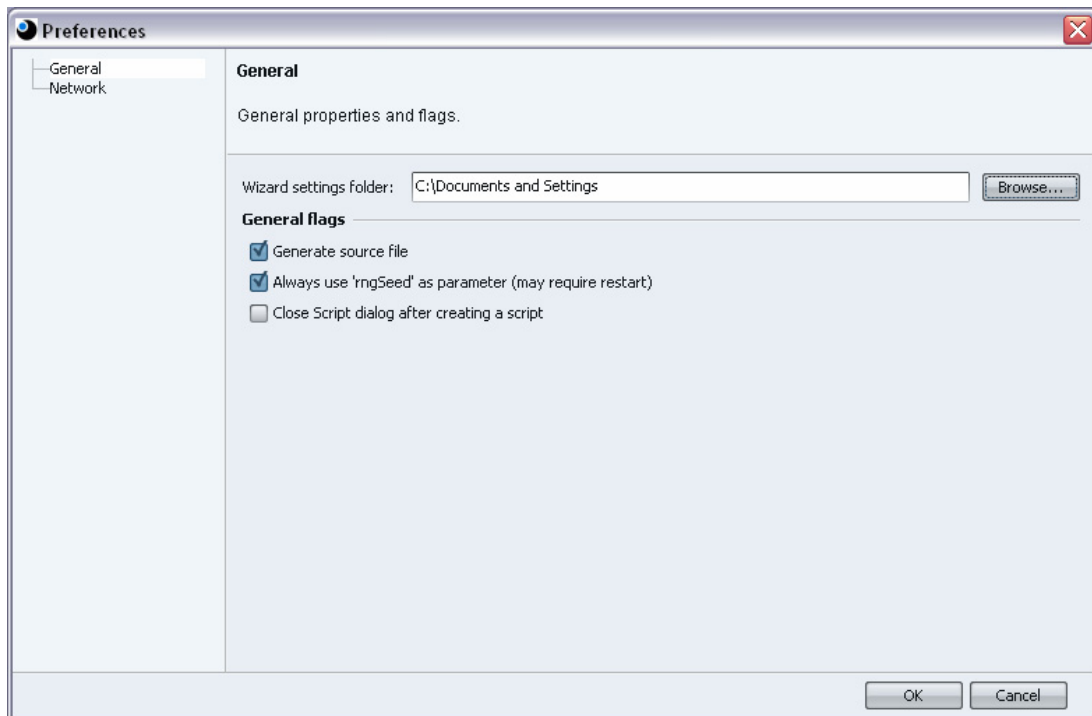


Figure 3 - Preferences dialogue, General settings

Pressing *OK* button applies the changes. In case of an error in the settings a warning message is displayed in the information panel below the title of the page.

1.1.1 General Settings

The following settings can be made on the *General* page:

Wizard settings folder: When a model is ran the wizard automatically saves all model settings (i.e. parameter file, recorders, scripts, etc.) which allows you to use these settings later (see details in section 1.2.3 Load Wizard Settings). In this field you can define the location of these descriptor files. By default this the user folder (in Windows).

Generate source file: If checked the wizard generates the source code of the model whenever a model is generated. This source file is placed in the directory where the original model file (the *.class* file) is.

Always use 'rngSeed' as a parameter: If checked *rngSeed* (initializing value of the random number generator) becomes a parameter of the model whether is contained in it originally or not. (See details about the parameters in section 1.3 Parameters Page and about the *rngSeed* and the *getInitParam()* in the Repast documentation [1].)

Note: This can only be changed before model selection.

Close script dialog after creating a script: Determines whether the *Script dialog* is closed when a single script is created or not. In the latter case multiple scripts can be created and the dialogue must be closed manually by pressing the *Close* button on it (see in section 1.5 Scripting)

1.1.2 Network Settings

The Network settings page determines whether the runs are executed locally or remotely, on one or on multiple machines. (See Figure 4 - Preferences dialogue, Network settings)

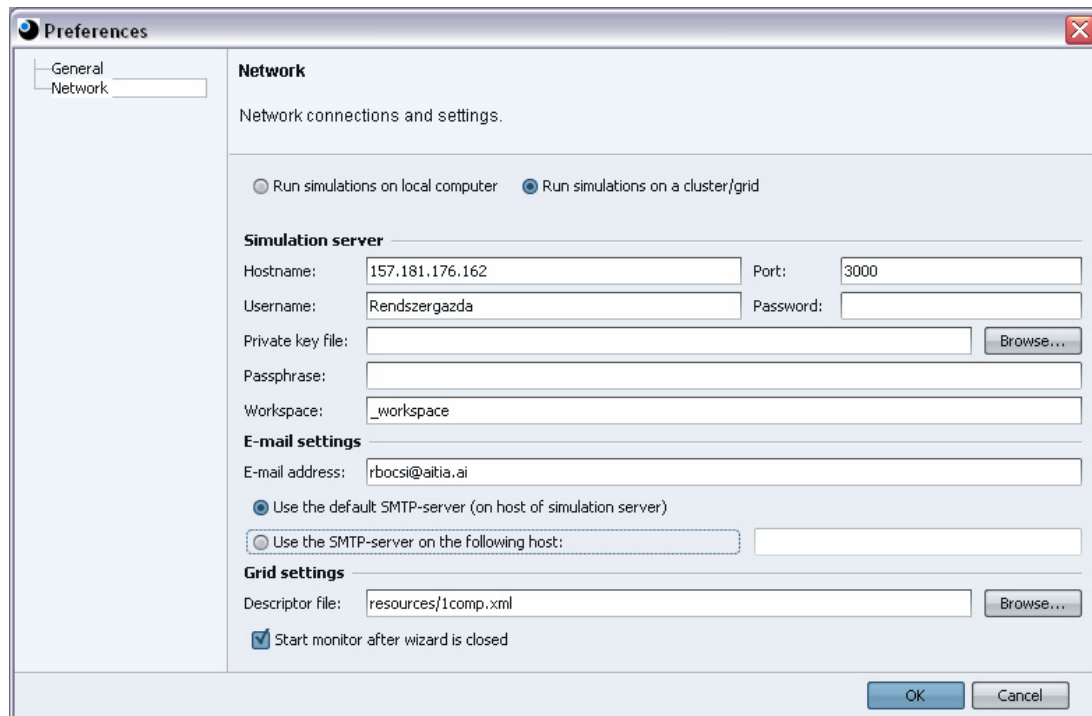


Figure 4 - Preferences dialogue, Network settings

As mentioned before the *Run simulations on local computer* option is the default setting. To switch to distributed mode select *Run simulations on a cluster/grid*.

Simulation server: In distributed mode the wizard runs simulations by connecting to a server application (called the *simulation server*) on a remote host. The details of the server-connection are described in this section.

- *Hostname:* The hostname (or IP-address) of the computer where the simulation server-application is located;
- *Port:* The simulation server port (default: 3000);
- *Username/Password:* The application transfers the files associated with the model through SFTP protocol. Usually this protocol requires a username and a password.
- *Private key:* Some SFTP-servers do not support password identification, they use private key files instead.
- *Passphrase:* Some SFTP-servers require a passphrase in addition to the above mentioned private key.
- *Workspace:* The working directory on the remote host.

Warning: The workspace must be defined relative to the default user directory of the SFTP-server.

E-mail settings: In distributed mode e-mail notifications can be requested about simulation events (completion, error, pause, etc.). The server uses the settings in the *E-mail settings* section (e-mail address, available SMTP-server). These settings are optional.

Grid settings: The simulation server application uses the ProActive grid solution to distribute the runs (see ProActive documentation [2]), where an XML file that describes the ProActive cluster has to be given. This file is specific to the computer infrastructure used in the cluster.

In the *<application directory>/resources* directory a file called *1comp.xml* can be found. This XML file is valid for all ProActive clusters, but it uses only a single computer (a cluster made up of one computer). It can be useful for running simulations on a remote

host instead of the local computer and is an example for creating cluster description files.

Note: These settings (including the appropriate grid descriptor file) should be provided by the system administrator of the given cluster or grid.

Start monitor after wizard is closed: If checked option the Monitor application is automatically started when running in distributed mode. This allows for the observation of distributed runs. See details about the monitor application in [2 The Monitor Application](#).

1.2 Model selection Page

The model can be selected (the class file of the model) on the opening page of the Parameter Sweep wizard (see [Figure 2](#)). This can be a general model description file (without GUI or batch specific code) or a batch model file. If the selected model is not a batch model the wizard will help in creating its batch version. In both cases, the wizard helps generating the parameter file to be explored and/or running the model on a cluster or grid of computers (or on the local computer).

In order for the wizard to load the complete model, the paths to all related classes must be specified. The directory where the model class can be found (in regard the package structure) is automatically added to the *Class path*. You can extend this list manually by pressing the *Add...* button and selecting one or more directories or JAR files in the file dialogue. To delete elements from the *Class path*, select them and press the *Remove* button. In the case of certain models the order of the *Class path* can be of importance, use the move up/down buttons to achieve the right order. (See further details in section [Extendable Class Path!](#))

The *Description* and the *Parameter file* fields are optional.

If an existing and appropriate parameter file is selected the wizard initializes the experiment with its content (but it is still modifyable). If no parameter file is defined, the wizard will generate one.

1.2.1 Extendable Class Path

Upon pressing the *Next* button the wizard tries to load the model. If the *Class path* is defined properly the wizard will move on to the next phase. Otherwise, if the wizard is unable to find a class it will ask for its location in the file dialogue shown in [Figure 5](#).

The name of the missing class is displayed in bold with yellow background on the top of the dialogue window. Select the class file of the class in the file system or the JAR file that contains this class and add it to the class path.

Note that the class path information can be provided in an incremental way.

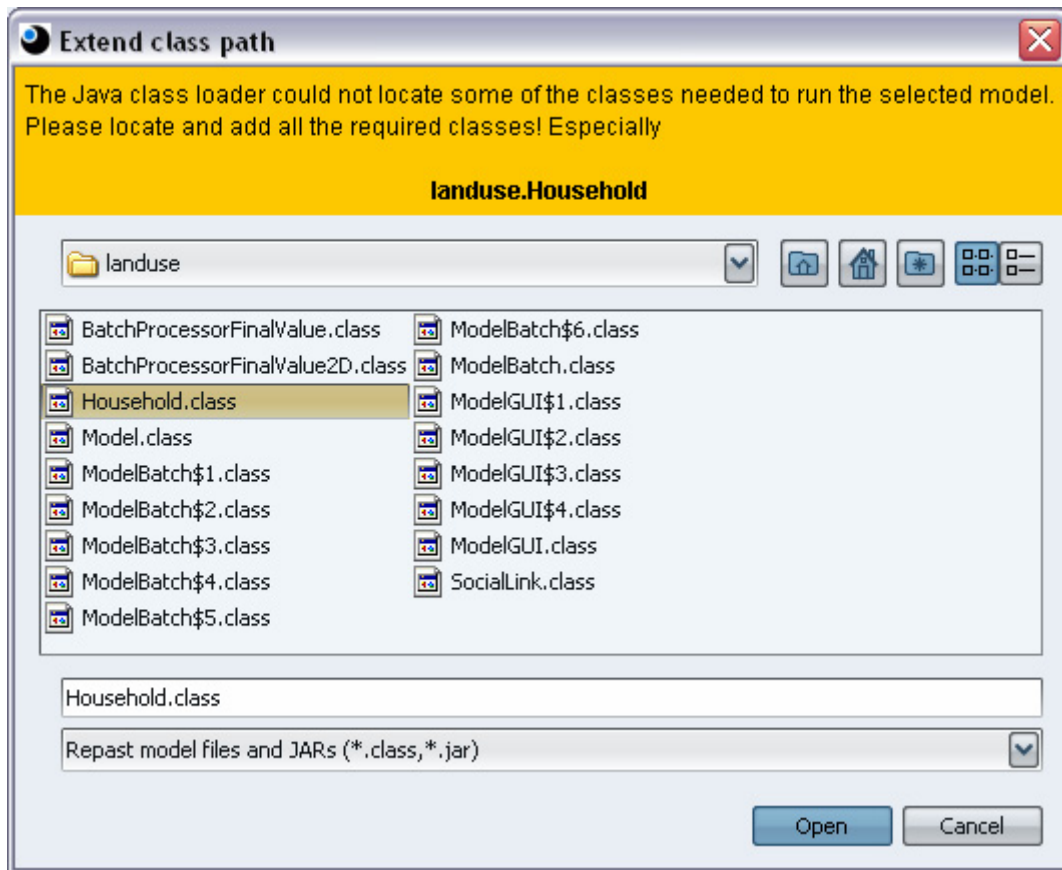


Figure 5

1.2.2 Resources

Certain models use files (e.g. text files, XML files, database files etc.) called resources. Running these models on a remote grid or cluster of computers requires the wizard to know the name and location of the resource files. Resources can be defined in the *Resources* dialogue (see Figure 6) which is appears when you press the *Resources...* button on the opening page of the wizard.

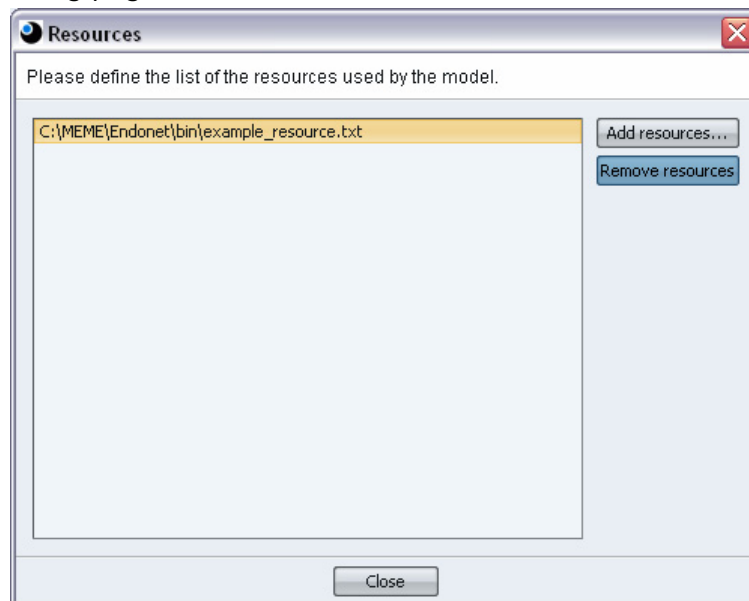


Figure 6

Extend the list of resources by pressing the *Add resources...* button and selecting one or more files in the file dialogue. The *Remove resources* button allows for deleting selected files from the list.

Due to technical reasons the selected resources must be in the model directory or in its subdirectory. (The model directory is the starting point of the package structure of the selected model.)

If you choose a file from an other location, a warning message is displayed (see Figure 7).

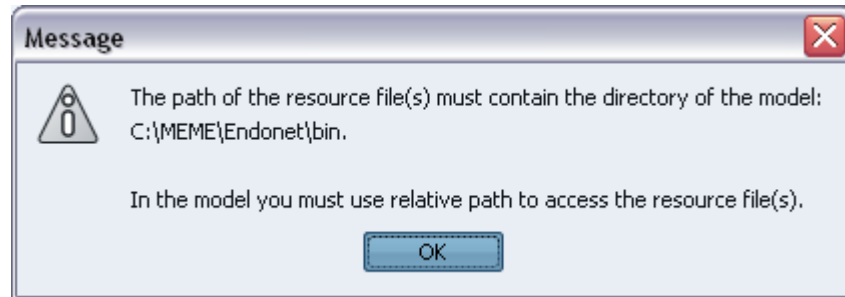


Figure 7

Note: As the message says above, only those models can be run in distributed mode that access their resources in relative way. If a model uses a resource defined by its absolute path, it won't find the file when running on a cluster or grid of computers.

When models are run locally, there is no need for defining resources, hence in local mode the *Resources...* button remains disabled.

1.2.3 Load Wizard Settings

When running a model, the wizard automatically saves all model settings (e.g. parameter file, recorders, scripts, etc.). These settings can be loaded by pressing the *Load wizard settings...* button on the *Model selection* page and selecting a wizard settings descriptor file in the file dialog shown in Figure 8.

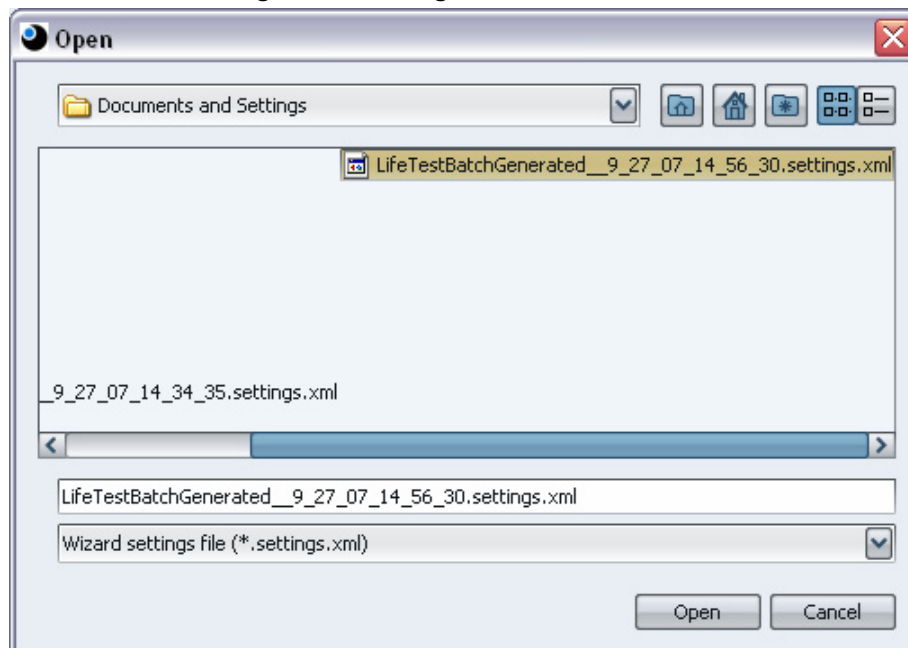


Figure 8

This file dialogue shows contents of the *Wizard settings folder* specified in the *Preferences* dialogue. A wizard settings descriptor file name always ends with

.settings.xml. These files have long names, but they hold a lot of information that helps to identify certain wizard settings.

For example, the filename *LifeTestBatchGenerated__9_27_07_14_56_30* indicates that:

- The original model name is *LifeTest*.
- The model is generated by the wizard (it contains the word „BatchGenerated“).
- The date of the generation is 27 September, 2007.
- The time of the generation is 14:56:30.

After pressing the *Open* button the wizard initializes with the selected settings. These settings can be later modified or the model can be run with the exact same settings.

Note: When you press the *Finish* button to run a model, the wizard automatically saves the settings even if you didn't do any modifications on previous settings.

1.3 Parameters Page

The parameter space the model is going to be run on can be defined on the second page of the wizard (see Figure 9). The parameter combinations that are going to be explored are defined, in other words the parameter file is created, or an existing parameter file is modified here.

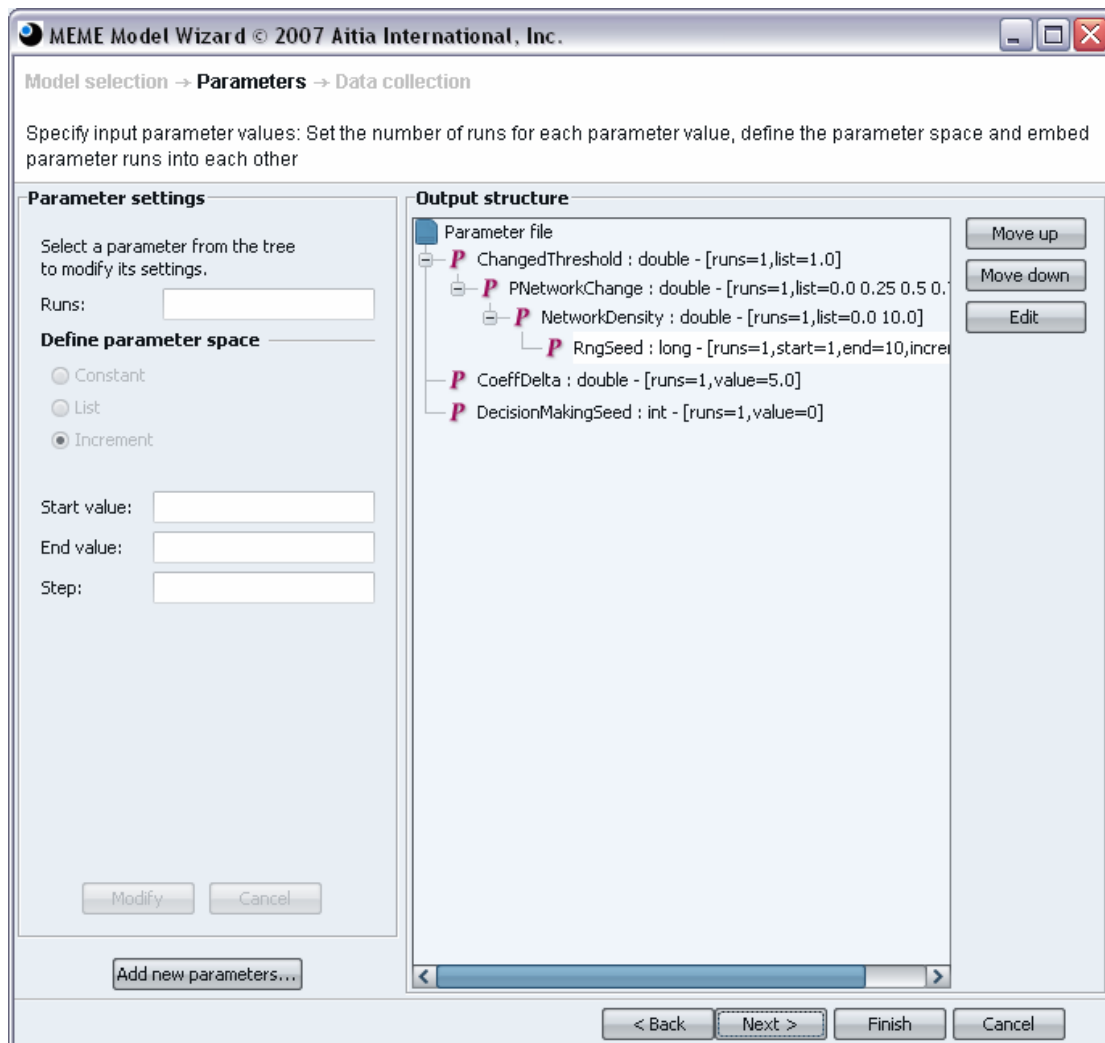


Figure 9

1.3.1 The Parameter Tree

The parameter combinations are represented by a tree. If a parameter is “under” another, it means that its values will be set of *all* values taken by the “parent” parameter. For example, the *NetworkDensity* and *RngSeed* parameters on the figure above will generate the following lines in the parameter file:

```
runs: 1
NetworkDensity {
set_list: 0.0 10.0
    {
        runs: 1
        RngSeed {
            start: 1
            end: 10
            incr: 1
        }
    }
}
```

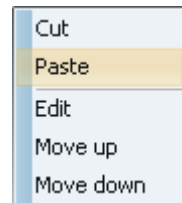
According to the parameter file syntax (see details in the Repast documentation [1]), this means that the *NetworkDensity* parameter will take two values, 0.0 and 10.0, and for both of these values the *RngSeed* parameter will iterate over the values of 1, 2, ... up to 10.

Parameters can be moved “under” one another in the tree by dragging and dropping them. To change the order of the parameters, select one and use the move buttons.

For editing the settings of a parameter, select it in the tree and press the *Edit* button or double click on it in the tree.

All these functions are available in the context menu (right click) of the parameter tree too.

This menu also contains *Cut* and *Paste* commands as an alternative of the drag and drop technique in parameter embedding.



Note: The Repast parameter file syntax doesn't allow parameters to be “under” a constant parameter. The wizard checks this requirement after pressing the *Finish* button.

1.3.2 Modify Parameter Settings

After pressing the *Edit* button (or double click on the parameter in the tree) the settings of the selected parameter can be defined in the left panel. The values can be specified in three ways (according to general conventions):

- by specifying a constant value (see below),
- by specifying a list of values (see below), or
- by specifying an iteration by providing a start value, an end value and a specific increment (see below).

Runs settings describe how many times a parameter is going to takes a value before it goes on to the next value.

Pressing the *Modify* button commits the changes.

The image shows three panels of the Parameter Sweep Wizard, each for a different parameter:

- Panel 1: CoeffDelta: double**
 - Runs: 1
 - Define parameter space:
 - Constant
 - List
 - Increment
 - Constant value: 5
- Panel 2: PNetworkChange: double**
 - Runs: 1
 - Define parameter space:
 - Constant
 - List
 - Increment
 - Value list: 0 0.25 0.5 0.75 1
 - (Separate values with spaces!)
- Panel 3: RngSeed: long**
 - Runs: 1
 - Define parameter space:
 - Constant
 - List
 - Increment
 - Start value: 1
 - End value: 10
 - Step: 1

1.3.3 New Parameters

It is also possible to extend the group of available parameters (that is normally specified in the *getInitParam()* method of the Repast model) by declaring a non-parameter variable to a parameter. For this, use the *Add new parameters...* button at the bottom of the left panel (see Figure 10).

The dialog box titled "Add new parameters" contains the following table:

Selected	Name	Type	Initial value
<input type="checkbox"/>	isChangeStats	boolean	false
<input checked="" type="checkbox"/>	coeffDelta	double	0.0
<input type="checkbox"/>	numOfAgents	int	0
<input type="checkbox"/>	numOfChars	int	0
<input type="checkbox"/>	probMakingAChoice	double	0.0
<input checked="" type="checkbox"/>	networkDensity	double	0.0
<input checked="" type="checkbox"/>	pNetworkChange	double	0.0
<input checked="" type="checkbox"/>	changedThreshold	double	0.0
<input type="checkbox"/>	staticThreshold	double	0.0
<input type="checkbox"/>	changedValue	int	0
<input type="checkbox"/>	staticValue	int	0
<input type="checkbox"/>	numOfMovers	int	0
<input type="checkbox"/>	initialChoiceSeed	int	0
<input checked="" type="checkbox"/>	decisionMakingSeed	int	0
<input type="checkbox"/>	hasMoved	int	0
<input type="checkbox"/>	name	String	null
<input type="checkbox"/>	autoStep	boolean	false
<input type="checkbox"/>	shuffle	boolean	false
<input type="checkbox"/>	seed	long	0
<input type="checkbox"/>	isGui	boolean	false
<input type="checkbox"/>	startAt	long	0

Figure 10

Select the variables by checking the box before their name in the table. To define initial values, simply double click in the *Initial value* field. Press the *OK* button to create parameters from the selected variables.

Note: Use this option before any other operation related to the parameters or the parameter tree because all previous user settings will be lost.

The wizard will modify the Repast model upon adding new parameters, creating the appropriate getter/setter methods for the new parameter variables and adding them to the parameter list.

1.4 Data Collection Page

The last page of the wizard is shown on Figure 11. This page helps specifying what kind of information is to be collected during the experiment and the stop condition for each simulation can be defined here.

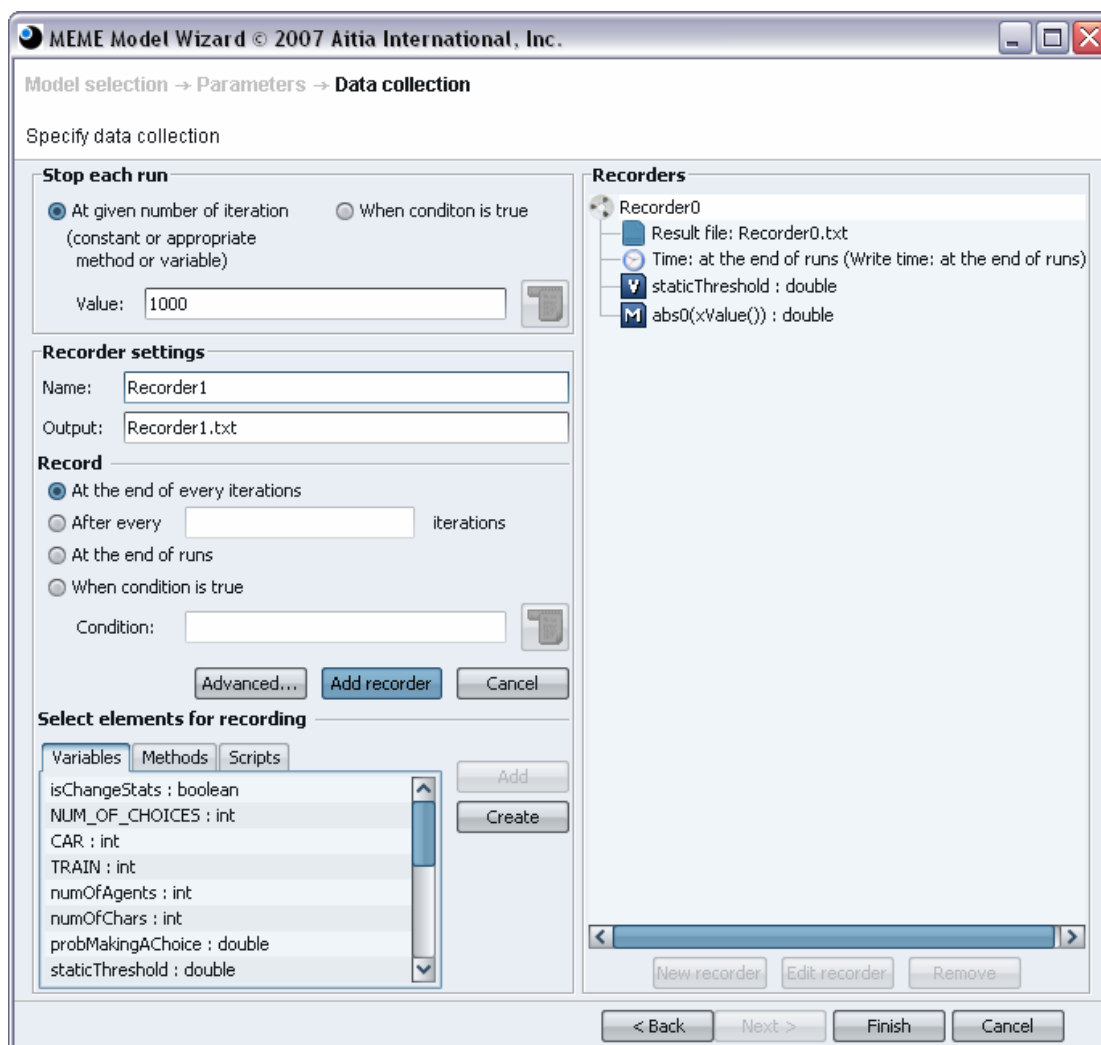


Figure 11

Pressing *Finish* button will generate the model, parameter file etc. corresponding with the settings just provided and will start running the model.

1.4.1 Stop condition

The stop condition can be a number (i.e. the specific time step at which runs will be stopped), it can be a variable with appropriate type (i.e. integer type), a method with the appropriate return type, or a logical expression built from the model's variables and methods (i.e. in the time step when this condition becomes true the simulation will be stopped).

Note: The wizard checks the stop condition when you press the *Finish* button, but only if there is a new model generated (you defined any recorders and/or new parameters).

1.4.2 Recorders

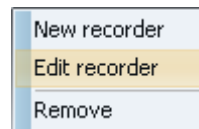
The tree on the right panel (see Figure 11) lists the specified recorders. It shows all relevant information concerning the recorders:

- name,
- name of the output file (one per recorder),
- time/frequency of recording (and time of writing data to file in brackets, see details below),
- what to record (names of variables and/or methods returning numeric, textual or logical values), etc.

The recorder collects the values of the specified variables and methods during the simulation. The collected values are then eventually saved into the specified output file. In order to speed up simulations, recorders store collected information in memory first and save them to files only at a given frequency.

In the current version of the MASS/MEME Parameter Sweep Wizard, beside variables or methods already existing in the Repast model, derivative values can also be specified as the source of information to be collected (see section 1.5 for details!).

A new data recorder can be added to the tree using the *New recorder* button. An existing recorder can be modified by selecting any element (e.g. recorder's name, a recordable element, etc.) of the recorder in the tree and pressing the *Edit recorder* button or double clicking on it. To remove a recordable element from a recorder simply select it and then press the *Remove* button. This button deletes the whole recorder if the selected element of the recorder is other than a recordable element, such as its name or the name of the output file, etc. You can also access these functions through the context menu (right click) of the tree.



The settings of the new recorder can be defined in the *Recorder settings* section on the left panel. The wizard offers a default name and a default output name that can be changed. There are four time/frequency options for recording:

- Record at the end of each iteration (i.e. every time step), or
- record after every x iterations (e.g. every 10th time step), or
- record at the end of each run (i.e. one recording for every defined parameter combinations), or
- record, when a condition is true.

The abovementioned condition must be a logical expression built from the model's variables and methods. The wizard checks it when it tries to build/modify the recorder.



You can specify the time of actual writing of the data in to a file on the *Advanced recorder settings* dialogue. This dialogue appears if you press the *Advanced...* button. There are three options here:

- Write to file after every recording, or
- write to file at the end of the runs (this is the default option), or
- write to file after x iterations (e.g. every 20th time step).

Note: Some of these options may be disabled corresponding with the time/frequency of the recording. For example, the *After x iterations* option is disabled if *At the end of the runs* option is enabled.

Note: The *Write to file* option may be changed if you change the recording time/frequency option *LATER*. The general rule is the following: There is no writing before recording.

Recorder settings can be added to the recorder only after pressing *Add recorder* button, which results in the data recorder appearing in the tree on the right. In case of editing recorder settings you must press the *Modify recorder* button to commit the changes.

Adding recordable elements to a recorder is done by selecting any elements of the recorder in the tree and selecting the elements from the tabs at the bottom of the left panel. Selected values can be added by pressing the *Add* button.

1.5 Scripting

As it is mentioned above the wizard allows the user to calculate derivative values (e.g. statistics) from "raw" information (e.g. already existing variables and methods) using a simple point-and-click interface and/or simple scripting.

To create a script, press the *Create* button on the *Data Collection* page (below the *Add* button). The *Create script* dialogue appears (see Figure 12).

There are two kinds of scripts: the statistics and advanced scripting (i.e. where you can actually write code).

1.5.1 Statistics

The *Statistics* tab of the *Create script* dialogue is shown on Figure 12.

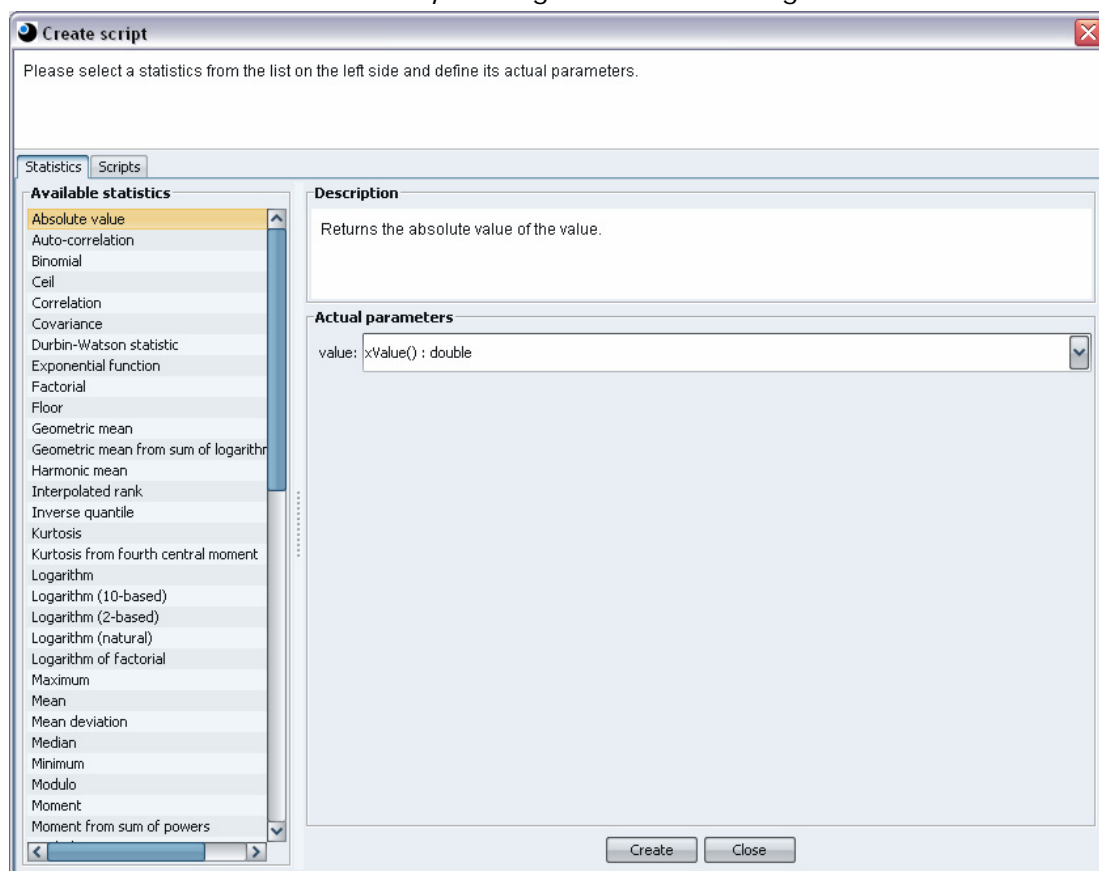


Figure 12

On the left side of the dialogue there is a list of available statistics (in alphabetic order). In the current version there are more than 60 of these.

When you select one of them a short description appears on the right panel. The statistics are based on the Colt library, for further details see the Colt documentation ([3]).

Below the description field the parameters of the selected statistic can be given. Generally, there are two kinds of parameters:

- Where a single number is specified (e.g. in the case of the Absolute value statistic on the figure above),
- Where a number collection is specified (e.g. in the case of the Maximum statistic on the Figure 13.)

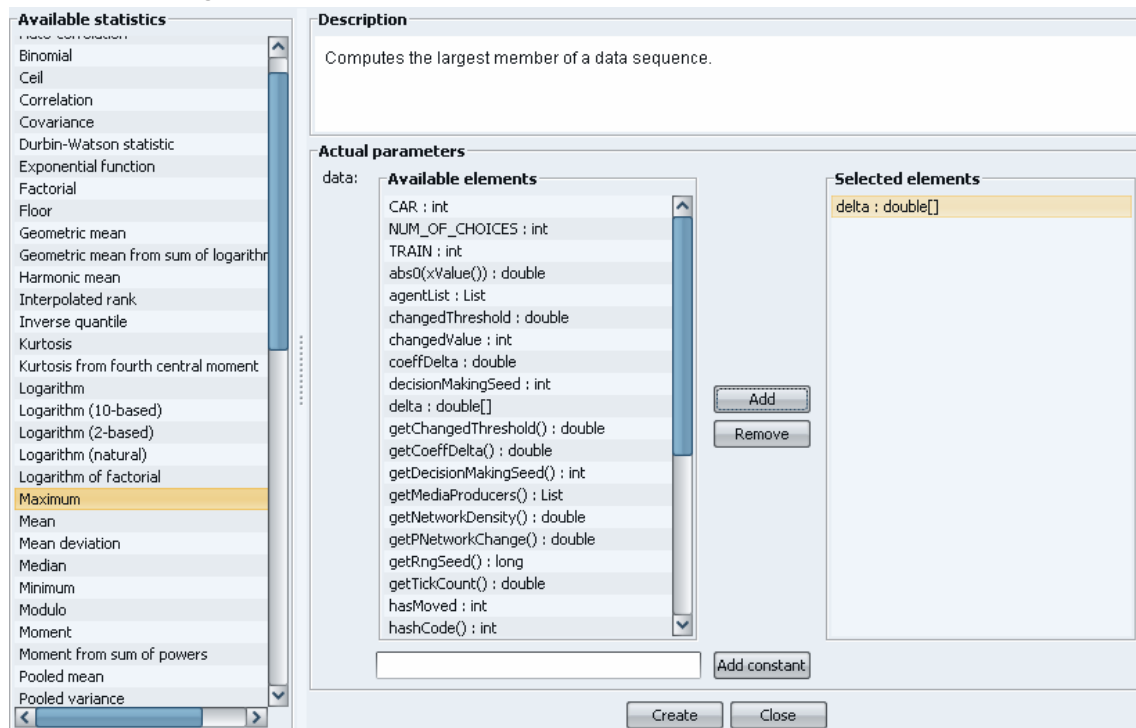


Figure 13

In the first case select the actual parameter from an editable drop down list. This list contains all numeric variables and appropriate methods (whose return value is a number) from the model (existed and previously generated too). Constant parameters can also be defined by editing the drop down list.

In the latter case a number collection has to be specified in the list on the right (*Selected elements*). In order to do so, select elements from the left list (*Available elements*) then press *Add* button. Remove elements from the right list by selecting them and then pressing the *Remove* button. The left list contains all number and number collection (e.g. `int[]`, `double[]`, etc.) variables and appropriate methods (whose return value is a number or number collection) from the model (existed and previously generated too). Extend this list with constant values by editing the field below the list and then pressing the *Add constant* button.

To create a statistic instance press the *Create* button. Error message (for example if the *Selected elements* list is empty) are shown on the information panel at the top of the dialogue.

1.5.2 Advanced scripting

Note: Advanced scripting, as its name shows, is for advanced users. It requires some programming skills in Java.

The *Scripts* tab of the *Create script* dialogue is shown on Figure 1.

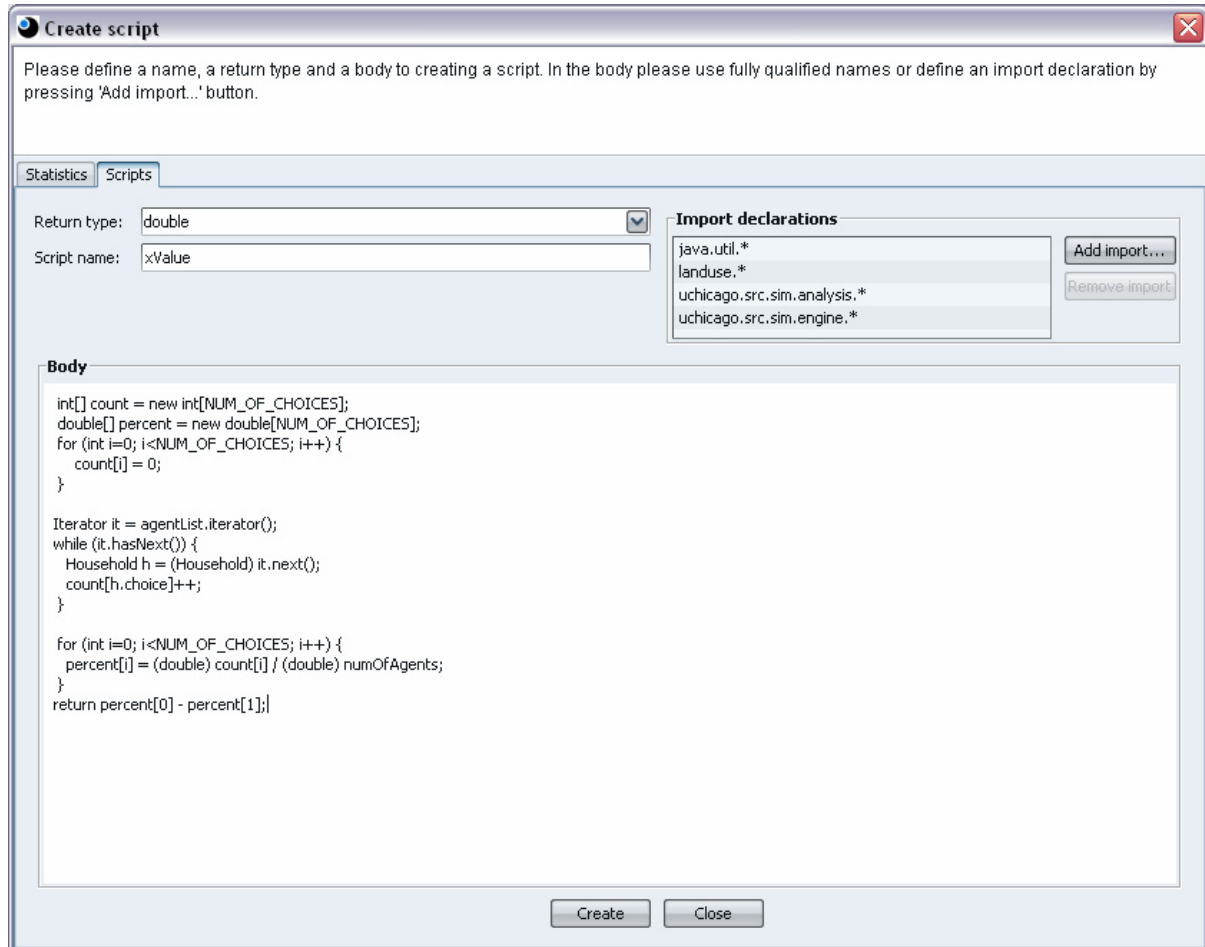


Figure 1

To create a script you define the following:

- *Script name*: Must be a unique name (not used in the model yet). The wizard offers a default name, but it can be changed. If the specified name is already taken the wizard returns an error when it checks the script.
- *Return type*: The return type of the script, which can be selected from the drop down list or type a new one.

Note: If the return type is not a numeric type, logical type or string, the created script can't be used as a recordable element, only in other scripts.

- *Body*: the source code of the script. See the *Limitations* section below.

In the body (and in the return type) the fully qualified names of the Java classes have to be used, unless they are in one the following packages:

- java.lang
- java.util
- uchicago.src.sim.analysis
- uchicago.src.sim.engine

Instead of using fully qualified names, import declarations can be defined by pressing the *Add import...* button and specifying an import declaration (such as in Java, except the usage of keyword *import*) on the *Import declaration* dialogue (see Figure 14). To remove an import declaration from the list select it, then press the *Remove import* button. Note that the predefined import declarations can't be removed.

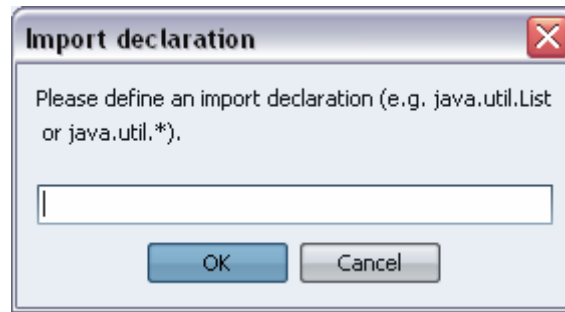


Figure 14

In the body of a script you can't use all features of the Java language. There are some limitations.

Limitations:

- The new syntax introduced by J2SE 5.0 (including enums and generics) is not supported.
- Array initialisers, comma-separated lists of expressions enclosed by braces { and }, are not available unless the array dimension is one.
- Inner classes or anonymous classes are not supported.
- Labeled `continue` and `break` statements are not supported.

To create the defined script press the *Create* button. It is then checked and error messages (if any) are shown on the information panel at the top of the dialogue.

Note: In the near future the advanced script support will be more sophisticated. The *Create script* dialogue will provide a more user friendly graphical user interface.

1.6 Built-in Monitor

The MASS/MEME Parameter Sweep Wizard contains a built-in monitor shown on Figure 15.

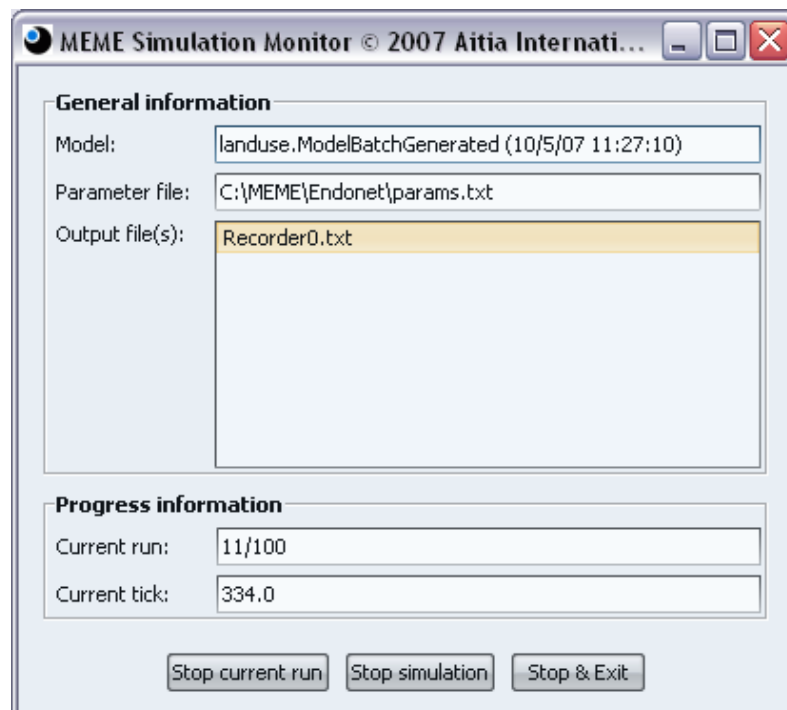


Figure 15

The built-in monitor is automatically launched whenever the wizard runs a model on local computer. The monitor displays the following information about the running model:

- Full name of the model (with the date and time of the generation/start),
- Name of parameter file,
- Name of the output file(s),
- Number of the current and the total runs, and
- Number of the current time step in the current run (usually called *tick*).

The *Stop current run* button stops the current run and starts the next one (if any). The *Stop simulation* button stops the simulation, but doesn't close the application while the *Stop&Exit* button stops the simulation and then closes the application.

2 The Monitor Application

The MASS/MEME Monitor application enables users to follow the progress of simulations running on clusters or grids of computers and downloading the output of finished simulations.

2.1 Configuration

Currently the Parameter Sweep Monitoring tool is a separate application from the Parameter Sweep Wizard. If the monitor application is started automatically by the wizard application (see section 1.1.2), the monitor inherits the network settings from the wizard so there is no need for manual configuration. If started separately though, the details of the network connection have to be specified in the *Configuration* dialogue (see Figure 16).

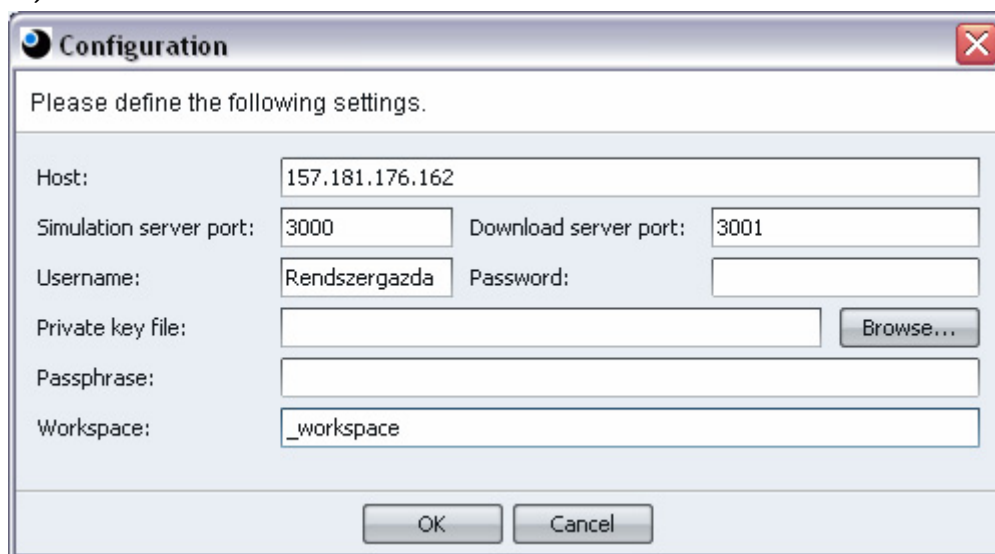


Figure 16

When running the application for the first time, this dialogue appears automatically. Later you can change the settings by pressing the *Configure...* button on the main window of the monitor. As you can see, the settings on this dialogue are very similar to settings on the *Preferences* dialogue of the Parameter Sweep Wizard.

- *Hostname*: Hostname (or IP-address) of the simulation server and the download server (this is an other server application that is used to download the output files of the simulations);
- *Simulation server port*: Port of the simulation server (default: 3000);
- *Download server port*: Port of the download server (default: 3001);
- *Workspace*: workspace on the servers (where they are on the remote host).

A username/password pair also needs to be given, because the monitor uses the SFTP protocol to download the output of simulations. Some SFTP-servers doesn't support this kind of authentication, they use private keys instead (with or without a password called passphrase). The *Configuration* dialogue allows to specify both types of authentication methods. Optionally you can select a file which holds the private key.

Warning: You must define the workspace relative to the default user directory on the SFTP-server.

If the monitor application is started automatically by the wizard the only setting that might need to be provided is the *Download server port* because the wizard doesn't use the download server. By default the wizard gives the *Simulation server port + 1* value as the port of the download server.

2.2 Current Simulation Tab

The monitor collects and displays information about the model currently run by the simulation server (see Figure 17).

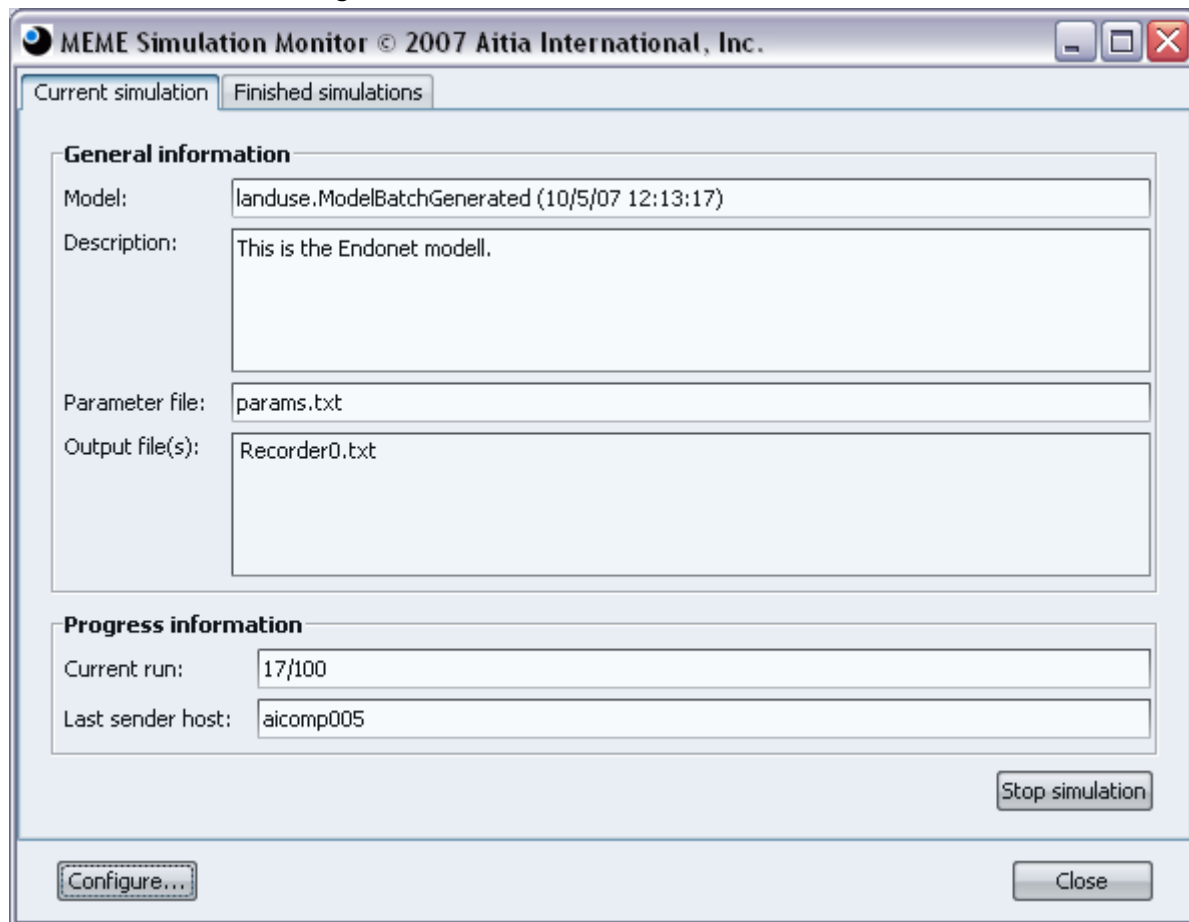


Figure 17

The full name and description of the models is shown, together with the time of uploading, the name of the parameter file and output files. Naturally, the progress of the given model is also displayed. The number of the current run and the number of the maximal run are shown at the bottom section. The *Last sender host* field contains the name of the computer that performs the current run.

Note: Before the start of the first run the simulation server executes some initializations which may take some time. Hence first run seems to be slower than the other runs.

By pressing the *Stop simulation* button you can order the simulation server to finish the model that is currently run and start the next one. Note that the simulation server finishes the current runs before it stops the running of a model.

2.3 Finished Simulations Tab

The second tab (Figure 18) of the monitor application displays information about and gives access to *Finished simulations*. Results of completed experiments can be downloaded from here.

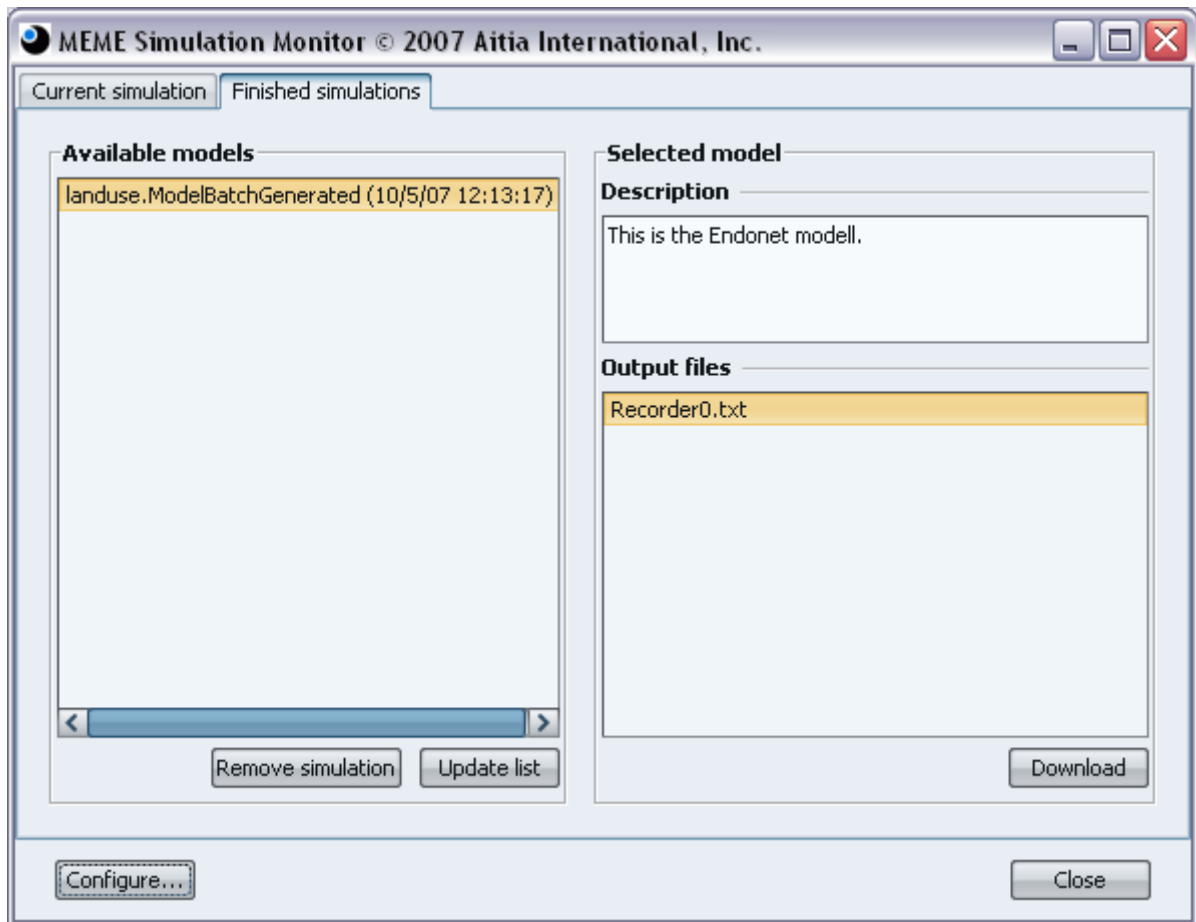


Figure 18

The list on the left displays finished simulations, identified by the full name of the model and the time of upload. This list is updated automatically whenever a simulation experiment is completed on the server. (The list can also be updated manually by pressing the *Update list* button.)

Upon selecting a finished simulation from the list, the description of the selected simulation and the list of its output files appears on the right. By pressing the *Download* button the *selected* output files are transferred to the local computer running the monitor.

You can remove the selected simulation from the server by pressing the *Remove simulation* button. All output files are deleted and this cannot be undone, so use this button carefully.

3 Frequently Asked Questions (FAQ)

Question: On the page *Data collection* of the wizard I define a stop condition, but I don't add any recorders to my model. Will the wizard generate a new model?

Answer: No. Model generation happens only if you add recorders and/or new parameters to the model.

Question: I select a batch model (with a hand-coded stop condition) then I define recorders and an other stop condition in the wizard. When will the generated model be stopped?

Answer: In this case, both stopping condition are active. Any of them becomes true (the simulation reaches the time step specified), the simulation will be stopped.

Question: I started two long simulations with the wizard. Will the simulations run in parallel? How does the monitor display their progress?

Answer: When you start a simulation with the wizard, the wizard sends a message to the server application about the simulation request after it transfers all relevant files. The server application stores the simulation requests in a queue and runs the simulations one after the other. The earliest request will be fulfilled the first. The monitor always shows information about the simulation running at the time.

Question: I'd like to run my model on the local host and follow the its progress, but the *Start monitor after wizard is closed* option on the *Network* page of the *Preferences* dialogue is checkable only in distributed mode. Why?

Answer: If you run models on the local host, the wizard shows progress automatically with the built-in monitor. The *Start monitor after wizard is closed* option relates to the stand-alone monitor application which is used only in distributed mode.

Question: I used the wizard application to run my model and I have a large result file. How can I use the Model Exploration Module (MEME) application to organize and visualize the data contained by the result file?

Answer: Currently, the MASS/MEME Parameter Sweep Wizard, the MASS/MEME Monitor and the MEME are separate applications. You must start the MEME manually and import results into MEME's database by using *File / Import / Repast result file* menu item. After that, you can use all available functions of MEME to organize and visualize the data. (See details in the MEME User Manual [4]!). In the near future we will integrate the wizard and the monitor with the MEME; after that the MEME will collect the results automatically, which will further simplify the modeling work.

4 Troubleshooting

The MASS/MEME Parameter Sweep Wizard and the MASS/MEME Monitor are under rapid development, therefore any feedback about errors and requests is appreciated, and, on the other hand, new updated versions are made available regularly. If you have experienced a software error or have suggestions about features, please send an e-mail to the support address of MEME (mass@aitia.ai).

If you send an error report, please include the *PSW.log* file from the installation directory of the applications, and also describe the steps that you have taken in the program before the error occurred. If you notice an error when you run a simulation in distributed mode, please also include the *outLog.txt* and *errLog.txt* files received from the simulation server.

5 Summary

The MASS/MEME Parameter Sweep Wizard supports modelers to perform distributed mass parameter sweep experiments on a cluster or grid of computers using a point-and-click graphical user interface.

The MASS/MEME Monitor application enables users to follow the progress of current simulations running on a cluster or grid of computers and download the output of finished simulations.

In the near future we will improve these applications. The planned improvements among others are the following:

- Integration of the MASS/MEME Parameter Sweep Wizard & Monitor with Model Exploration Module (MEME) application.
- Integration of the MASS/MEME Parameter Sweep Wizard & Monitor with FABLES IME application.
- Enhanced support for statistics.
- Enhanced support for advanced scripting – use of external libraries are also used; the possibility of defining and initializing own members.
- Support for intelligent parameter space exploration – currently the wizard supports only the simplest strategy of parameter space exploration known as Italian cube where all parameter combination are independent. There are other – more effective strategies, where the next parameter combination can be dependent on the previous results.

6 References

- [1] Repast - Recursive Porus Agent Simulation Toolkit
<http://repast.sourceforge.net/>
- [2] ProActive
<http://proactive.inria.fr/>
- [3] Colt
<http://dsd.lbl.gov/~hoschek/colt/>
- [4] The Model Exploration Module – User Manual