

6. UM EXPERIMENTS MODULE	6-2
6.1. Introduction	6-2
6.2. Scanning	6-4
6.2.1. Overview	6-4
6.2.2. Describing a new scanning project	6-6
6.2.3. Alternatives	6-6
6.2.3.1. Hierarchy of parameters	6-7
6.2.3.2. Tree of alternatives	6-7
6.2.3.3. Identifiers	6-8
6.2.3.4. Initial condition	6-8
6.2.3.5. Finish conditions	6-8
6.2.3.6. Variables	6-9
6.2.3.7. Integrator	6-9
6.2.3.8. Remark	6-9
6.2.4. Running	6-10
6.2.5. Results	6-11
6.2.5.1. Wizard of graphs	6-12
6.2.5.2. Wizard of surfaces	6-13
6.2.6. Methodical remarks	6-15
6.3. Optimization	6-16
6.3.1. Description of an optimization project	6-17
6.3.1.1. Description of design parameters	6-17
6.3.1.2. Identifiers	6-17
6.3.1.3. Initial conditions	6-17
6.3.1.4. Stop conditions	6-17
6.3.1.5. Solver parameters	6-17
6.3.2. Description of objective function	6-18
6.3.2.1. Intermediate criteria	6-19
6.3.2.2. Terminal criteria	6-20

6. UM Experiments module

6.1. Introduction

UM Experiments module is developed to help the researcher to analyze dynamical behavior of models and fulfill their parametrical optimization. Manual optimization according to Fig. 6.1 is a quite time-consuming and laborious process where researcher has to control many different parameters of a model and settings that often leads to mistakes.

UM Experiments module is built in the **UM Simulation** program and includes the following components:

- tool for describing and executing scanning projects (*scanning*);
- tool that is based on classical optimization methods such as Hook-Jeevse's, Nealder-ead's, Pawell's etc (*optimization*);
- tool that is based on quadratic approximation of a response surface;
- additional tool for describing the objective function based on the analytic hierarchy process by T. Saaty;
- service of distributed calculations.

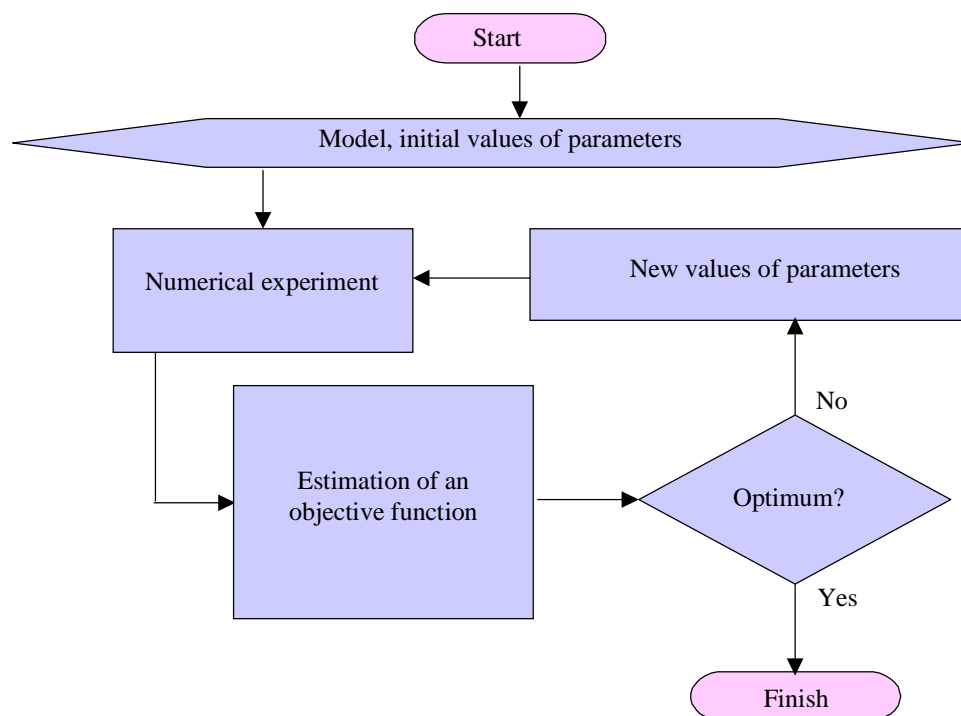


Figure 6.1. Optimization process

It is often required in engineering practice to carry out series of numerical experiments, for example to analyze dynamical behavior and sensitivity of mechanical system or to find out optimal parameters of the system. **UM Experiments** includes a set of tools (*scanning, optimization, approximation*) for advanced analysis of dynamics of mechanical systems.

All tools automate fulfillment of series of numerical experiments and save results of experiments on a hard disk for posterior analysis. Thus, the designer is released from monotonous execution of series of numerical experiments "manually" what saves working hours and removes errors, which people unfortunately usually do. In other words, the researcher defines the design of experiments for scanning and approximation or parameters, its limits, precision and goal function for optimization. Then the project is started and executed automatically. Current process statistics is available during the execution: number of executed experiments, time left. Series of

numerical experiments are resistant to shut-down of power supply. In that case all results are saved on a hard disk and only results of the latter experiments are lost.

There is a possibility to plot an oscillogram of any saved performance of dynamical behavior of a mechanism. Moreover, the designer can plot so-called summary graphs and surfaces. All implemented tools have no limitation in number of parameters. In other words, they all are multi-parametrical. Please, note that the demo version allows describing the projects of scanning (optimization, approximation) with no more than one parameter. Dimensions of projects are set by the designer so as to solve the specified problem. But on the other hand the designer has to take into account calculating efforts that are necessary for the project and find a compromise between them.

Every tool has its own merits and demerits. However they all give the designer possibilities to solve quite many problems devoted to optimization of mechanical systems.

After the execution of series of numerical experiments for scanning the problem of choosing the optimal parameters arises. How can the designer find out the best solution from lots of alternatives? It becomes much more difficult in the case of conflicting criteria. In order to help the designer to formalize his/her opinion the special tool, which helps the designer to sort alternatives according to his/her opinion about optimality, was developed. This tool is based on the analytic hierarchy process by Tomas L. Saaty. It is a multi-criteria method, which supports weighting of criteria, conflicting criteria and verification of correctness of expert's opinion.

There is a special extension of the module - service of distributed calculations. It allows using all computational powerful of a network for execution of series of numerical experiments that decreases time efforts correspondingly. This possibility is very easy and effective to use in computer centers and laboratories. Server of distributed calculations is based on using TCP/IP that allows employing for the needs of your project any computer not only in local network, but also in Intranet and Internet.

6.2. Scanning

6.2.1. Overview

Scanning of the design space is the most simple and reliable method to find the optimal solution or analyze dynamical behavior of the system. Scanning gives the researcher clear idea about response surface and global optimum. However practical application of scanning is limited up to 3-4 parameters due to dramatically increasing the number of numerical experiments:

$$N = m^k, \text{ where}$$

k is the dimension of the problem,

m is the number of levels for each parameter.

Let us consider basic features of the scanning how it realized in UM. The researcher can unite several models within on scanning project, see Fig. 6.2. Set of numerical experiments is generated automatically. For each model the researcher describes default values of parameters, initial conditions, problem-specific parameters (road vehicles, railway vehicles, and flexible bodies), finish conditions, parameters of numerical method and list of variables that will be saved for each numerical experiment.

List of calculated variables can include all kinds of dynamical performances: coordinates, velocities, accelerations, components and modules of applied forces and moments etc.¹ Read <Sect. 4.3.3. List of variables> for details.

Finish conditions are formulated as follows:

<Variable1><logical operation1><Value1> OR

<Variable2><logical operation2><Value2> ..., where

<Variable i > is time or any UM variable.

After fulfillment of all numerical experiments their results are available for analyzing, see Fig. 6.3. It is possible to get an oscillogram of any variable from the list of variables as well as summary graphs, surfaces and tables.

Let us consider usual work flow.

- Researcher adds all necessary models to the project and set all parameters for them.
- A list of numerical experiments is generated automatically.
- Scanning project starts and fulfills all numerical experiments, possibly with the help of Service of distributed calculations.

¹ Practical experience shows that the list of variables can contain up to several decades and even hundreds different variables.

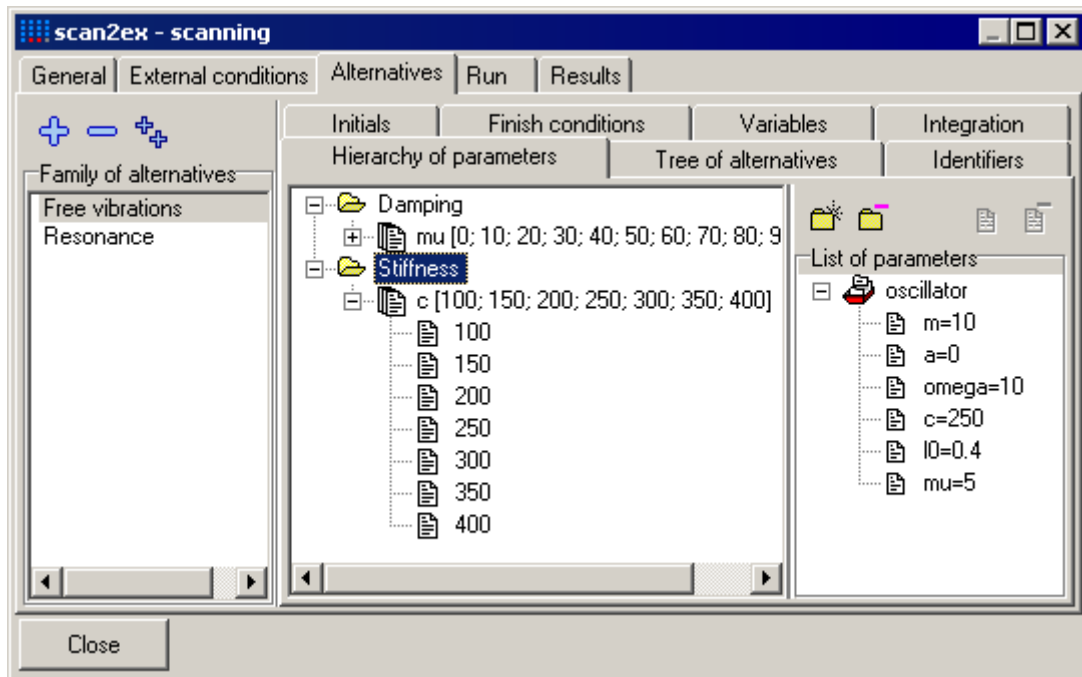


Figure 6.2. Scanning: hierarchy of parameters

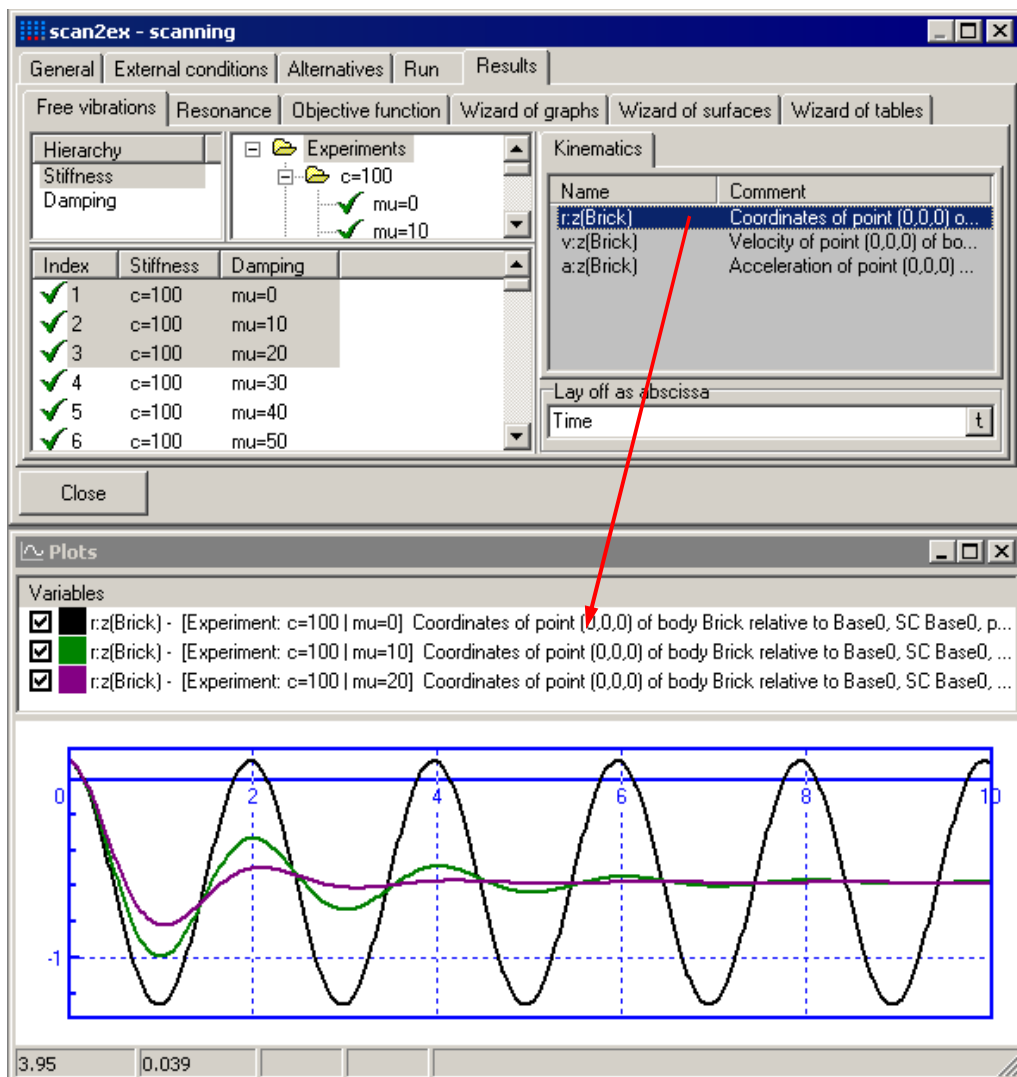


Figure 6.3. Scanning: processing results

6.2.2. Describing a new scanning project

Use the menu command **Advanced analysis / Scanning: new project** to create a new scanning project. Every project is situated in a separate directory. Therefore it is necessary to point out the directory when you create new scanning project, see Fig. 6.4.

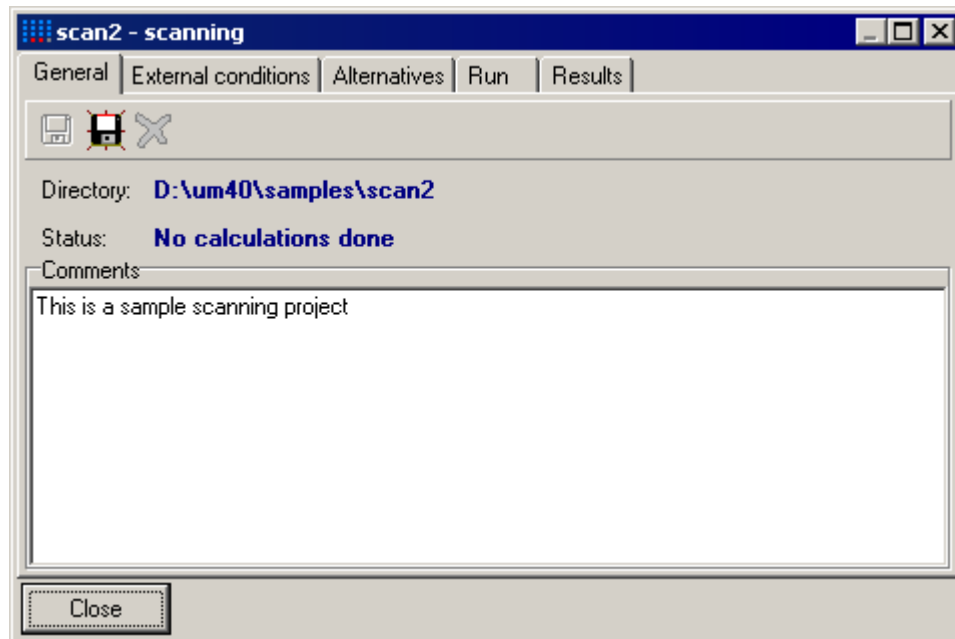


Figure 6.4. New scanning projects

6.2.3. Alternatives

A tab for description design parameters is shown in Fig. 6.6. **Family of alternatives** is a group of alternatives generated by one model. Use **+** and **-** buttons or context menu (see Fig. 6.5, 6.6) to add/remove families. It is necessary to describe hierarchy of parameters for each family. Hierarchy of parameters can be compare to nested loops. It means that for each value of upper level all values of lower level will be tabulated. Within a level, the parameters are changed jointly; number of iterations is determined by number of values of the first parameter at the level. If the number of iterations in the first parameter is less than ones of other parameters then final extra values are ignored, else the values of other parameters are taken over again.

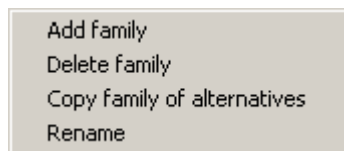


Figure 6.5. Family of alternatives: context menu

Possibly it might be more suitable not to add new *families* and set all their parameters but just copy families (the **+** button) and alter them.

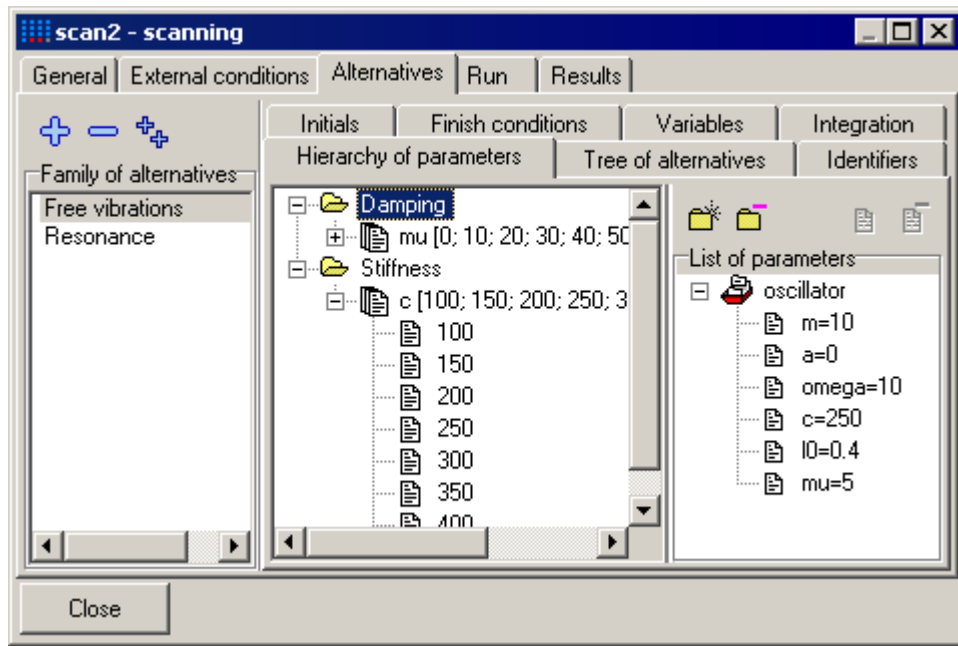


Figure 6.6. Hierarchy of parameters

6.2.3.1. Hierarchy of parameters

Use and buttons to add/remove levels of hierarchy, and and buttons to add/remove parameters to/from current level. Double click onto any parameter from the list adds that parameter to current level of hierarchy. For example, the hierarchy in Fig. 6.7 consists of two levels: **Damping** and **Stiffness**. **Stiffness** level consists of one parameter **c** having values [100, 200, 300, 400]. **Damping** level consists of one parameter **mu** with values [0, 10, 20, 30..100].

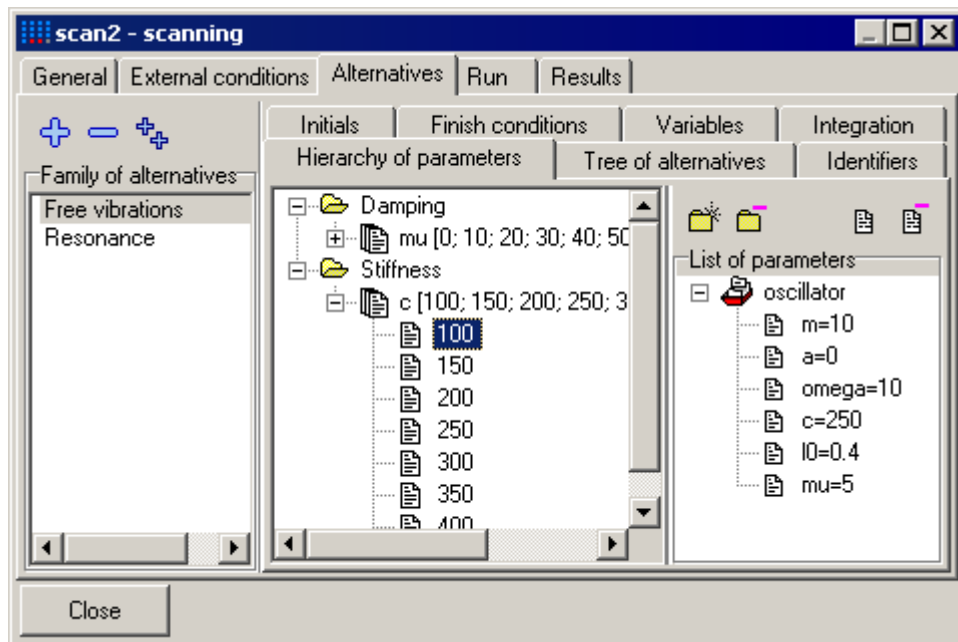


Figure 6.7. Design parameters

6.2.3.2. Tree of alternatives

After description of hierarchy of parameters the full list (tree) of alternatives is generated automatically (see Fig. 6.8). Tree of alternatives is on the top and list of alternatives is on the bottom.

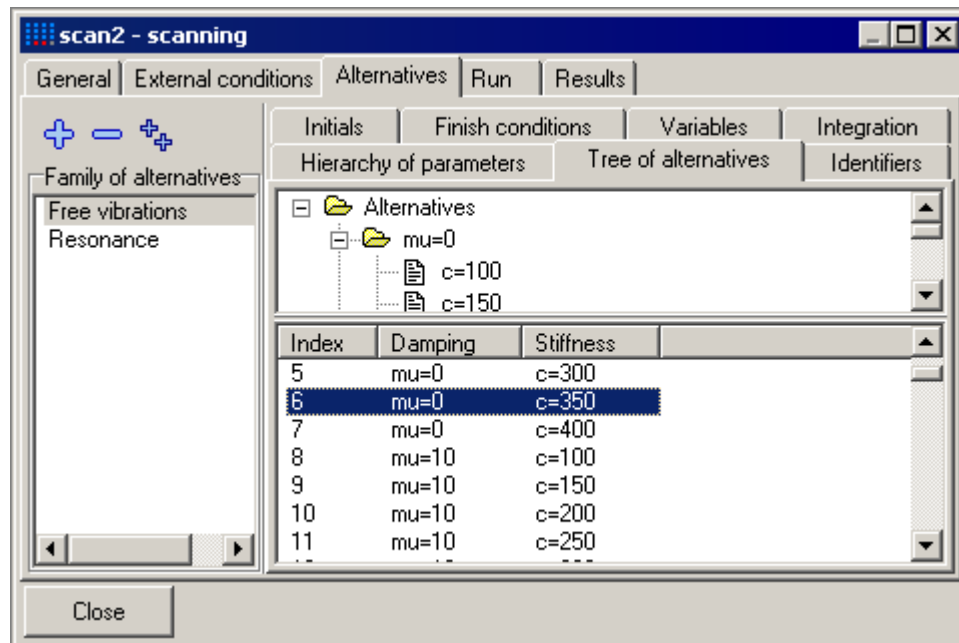


Figure 6.8. Tree of alternatives

6.2.3.3. Identifiers

Here you can find all of the system identifiers (see Sect. 4.4.2.4). These values of identifiers will be used as default values for numerical experiments. The values are loaded before each numerical experiment. Of course, if an identifier changes as a design parameter then such an identifier will be redefined with the correspondent value.


6.2.3.4. Initial condition

Here you can describe initial conditions each numerical experiment starts with (see Sect. 4.4.2.5).

6.2.3.5. Finish conditions

Here you can describe finish conditions for each numerical experiment in the current family, see Fig. 6.9. Finish conditions are formulated in the following way: “Interrupt a numerical experiment if at least one of the conditions is satisfied”.

For example, stop condition in Fig. 6.9 is: “Interrupt a numerical experiment if simulation time is more than 25 seconds or the position of the center of mass of the body ‘Body’ is more than one meter”.

You can use time and any other variables as criteria for stop conditions. Declare the necessary variable in the **Wizard of variables** (see Sect. 4.3.2. Wizard of variables) and drag it to the correspondent field. Use  button to set *time* as a criterion.

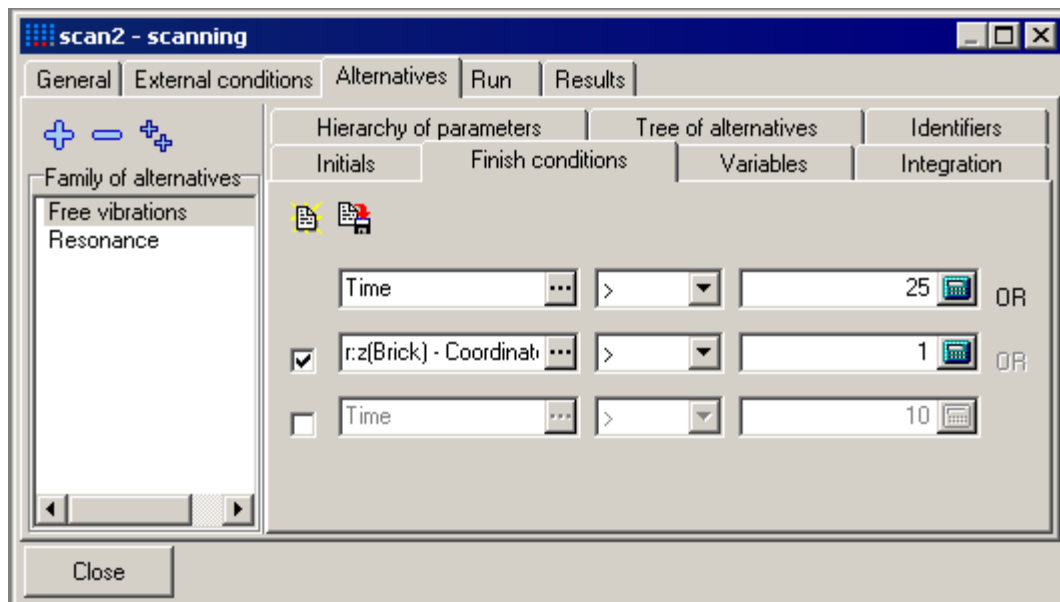


Figure 6.9. Finish conditions

6.2.3.6. Variables

Here you should form a *list of variables* (see Sect. 4.3.3), which will be stored for every numerical experiment. Variables from this list will be available as results of *scanning*.

6.2.3.7. Integrator

Here you should choose integration method and set its parameters. This setting will be used for numerical simulation (see Sect. 4.4.2.1).

6.2.3.8. Remark

Setting considered in 6.2.3.1-6.2.3.7 are valid for active (current) family. You should go through these steps for every family that you introduced in the project.

6.2.4. Running

When you activate the **Run** tab (see Fig. 6.10), checking the description of scanning project is performed. You can see its results in the **Report** box. If some errors are found, you should fix them and activate **Run** tab again.

When scanning project starts, the following information for each numerical experiment is available: time information, values of identifiers of hierarchy of parameters, summary information. A progress bar shows the ratio between number of done experiments and total number of experiments. Use the **Stop** button to interrupt execution. If any software or hardware breakdown appears, the results of the scanning project are not lost and execution will go on from the last unfulfilled experiment.

Let us consider an example, see Fig. 6.10. This project includes 93 numerical experiments and 21 of them are already done. It is expected that completing the project will take us 20 seconds more.

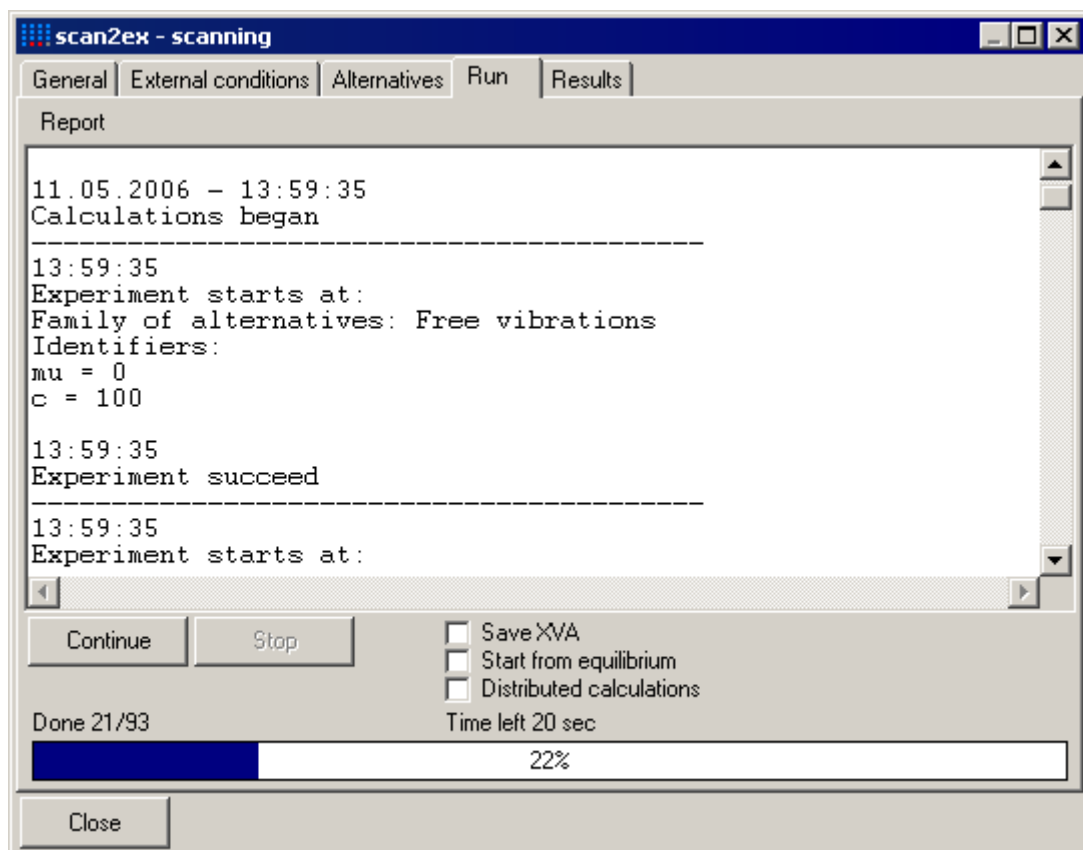


Figure 6.10. Running the project

6.2.5. Results

If at least one numerical experiment is done then its results are available in the **Results** tab (see Fig. 6.11). For each family of alternatives, there is the tab of the same name. There is a list of calculated variables on the right. On the left there is the list of numerical experiments. Fulfilled experiments are marked with the ✓ sign, and unfulfilled ones are marked with the ✗ sign.

Select necessary experiments (use the **Shift** and the **Ctrl** keys in a standard way) from the list of experiments, then select necessary variables from the list and perform drag them to a graphical window. Variables in the graphical window have a correspondent comment, see Fig. 6.11.

The tree of experiments corresponds to the list of experiments. Selection of a branch in the tree leads to selection the corresponding items in the list. Use the context menu of the hierarchy to change position of its levels and, as a result, fast selection of necessary experiments.

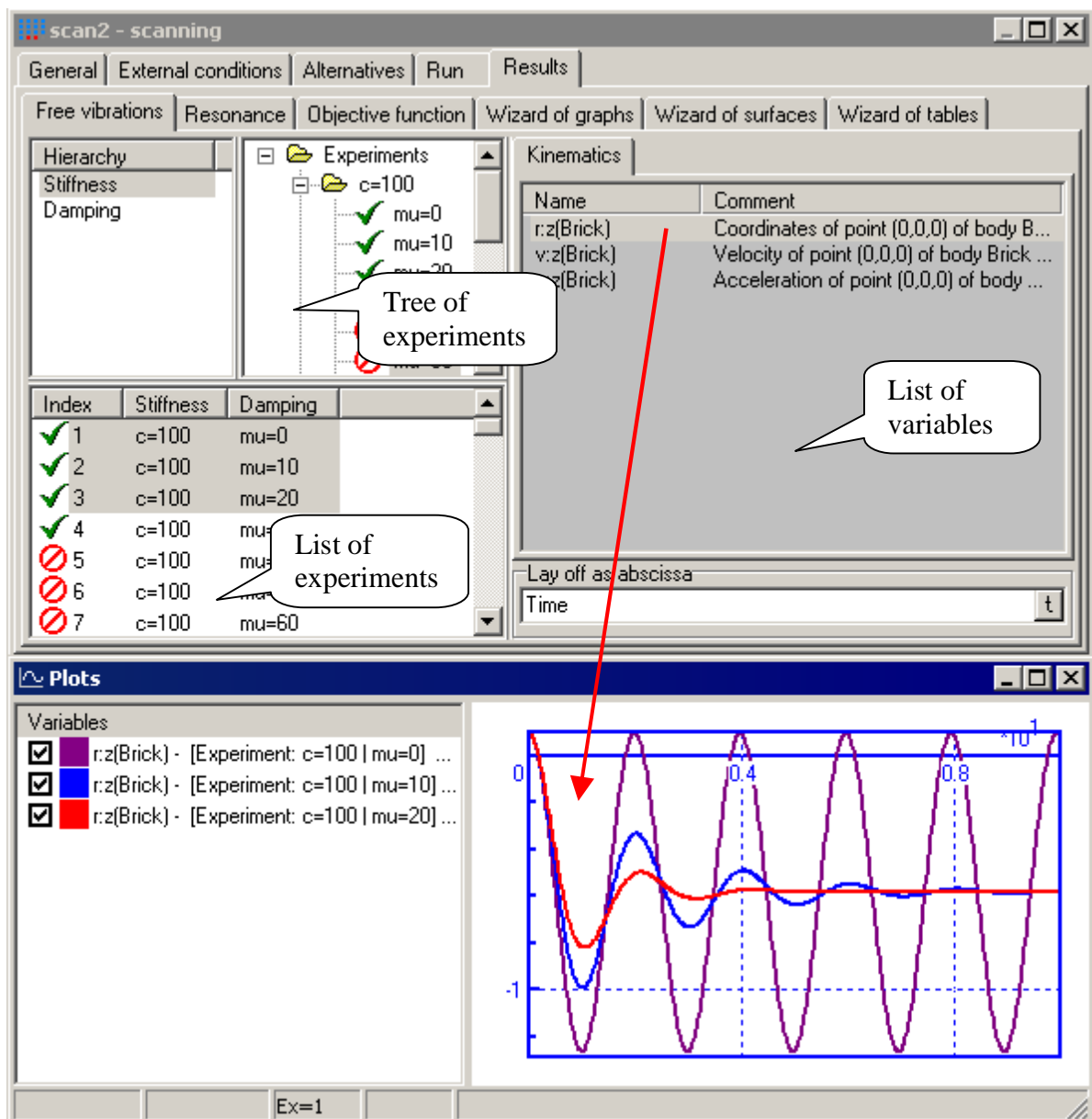



Figure 6.11. Processing results

6.2.5.1. Wizard of graphs

The *wizard of graphs* (see Fig. 6.12) is intended for forming dependence between a function of any calculated variable on any parameter, when other parameters are fixed. Comments in the right bottom corner of the window will help you.

To add a summary graph you should go through the following steps.

- Choose a family, a variable to be analyzed, functional and parameter for computing a dependence, see Fig. 6.12.
- Optionally you can use limit for abscissa that can be used for excluding transient processes, as well as scale factors/units and interval for abscissa.
- Click the  button to plot the summary graph.
- To add new plot change the **Other parameters** if necessary and double click on the same or new variable in the **Criteria and variables** tree.

Note **Other parameters** list is enabled if only scanning project has more than one level.

Let us consider an example, see Fig. 6.12. Here we can see the root mean square of the vertical position of the 1 d.o.f. oscillation system in dependence on *omega* parameter. In this case of forced oscillation *omega* means the frequency of the driving force. The resonance case is clearly shown.

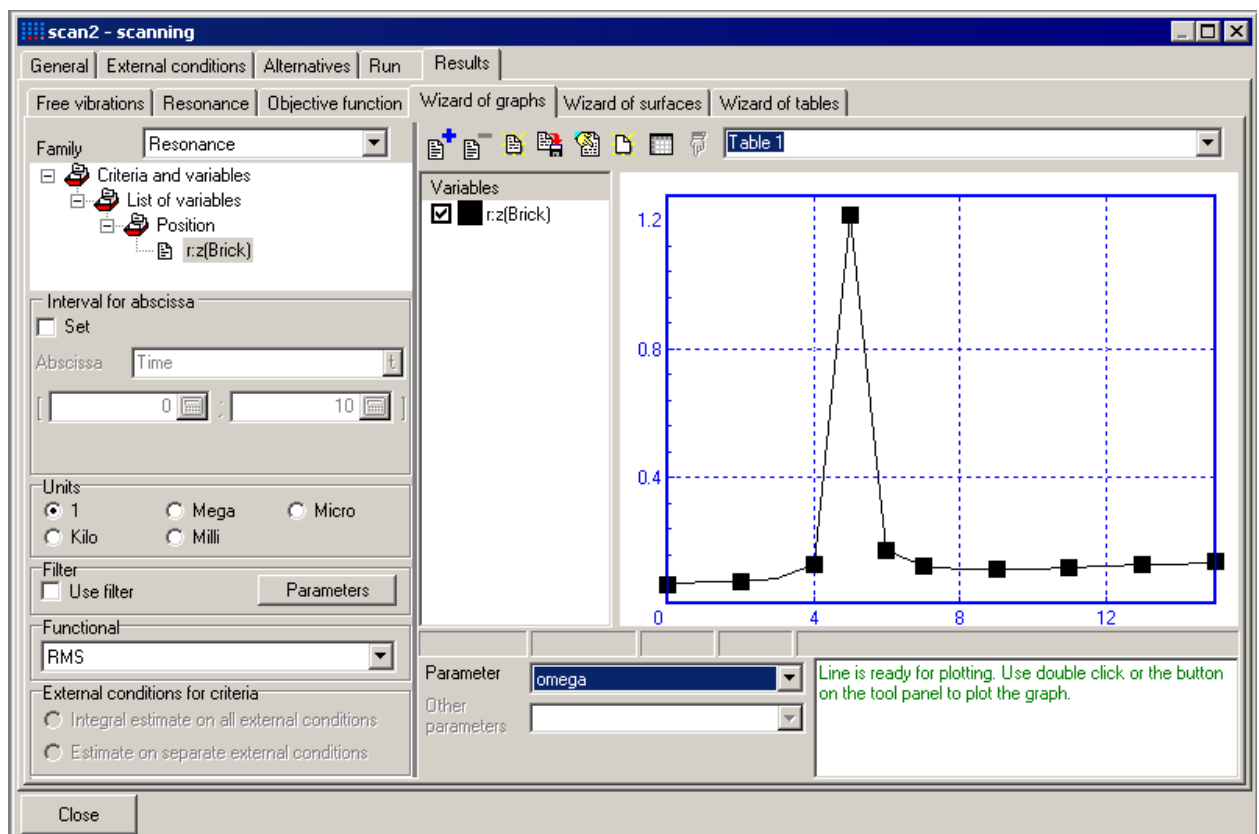









Figure 6.12. Wizard of graphs

You can switch between graphical and table mode with the help of  /  button. In the table mode you can drag any column of the table to the graphical window. Drag the first column to plot all graphs.

Note

- Choosing a new family of alternatives or a new parameter clears the table/graphs.
- Changing variables in the **Criteria and variables** tree keeps the table contents and can be used to obtain a group of dependences.

You can determine settings for **Abscissa limit**, **Units** and **Filter** during describing the scanning project in the **Variables** tab. To show these processing options use the  button. In the presence of such processing options they will be used automatically.

It is usually more suitable to use different tables for summary graphs of variables with different dimensions. Use ,  buttons to add/delete tables. To save/read template (configuration) of the wizard of graphs use  and  buttons.

6.2.5.2. Wizard of surfaces

Wizard of surfaces is intended for creating the response surface as functional of any calculated variable in dependence on two any design parameters keeping other parameters constant. Working with **Wizard of surfaces** make sense if scanning has at least two design parameters. **Wizard of surfaces** is quite similar to **Wizard of graphs**. An example of using the **Wizard of surfaces** is given in Fig. 6.13. Corresponding surface in *Microsoft Excel* is given in Fig. 6.14.

Working with the **Wizard of surfaces** use the following work flow.

- Select the family, calculated variable or criterion and necessary functional, see Fig. 6.13.
- Then choose two parameters for the surface and fix values of **other parameters** if any.
- Set **Abscissa limit**, **Units** and **Filter** parameters if necessary.
- You can export prepared surface to text file or clipboard as text table and to the *Microsoft Excel* as a diagram.

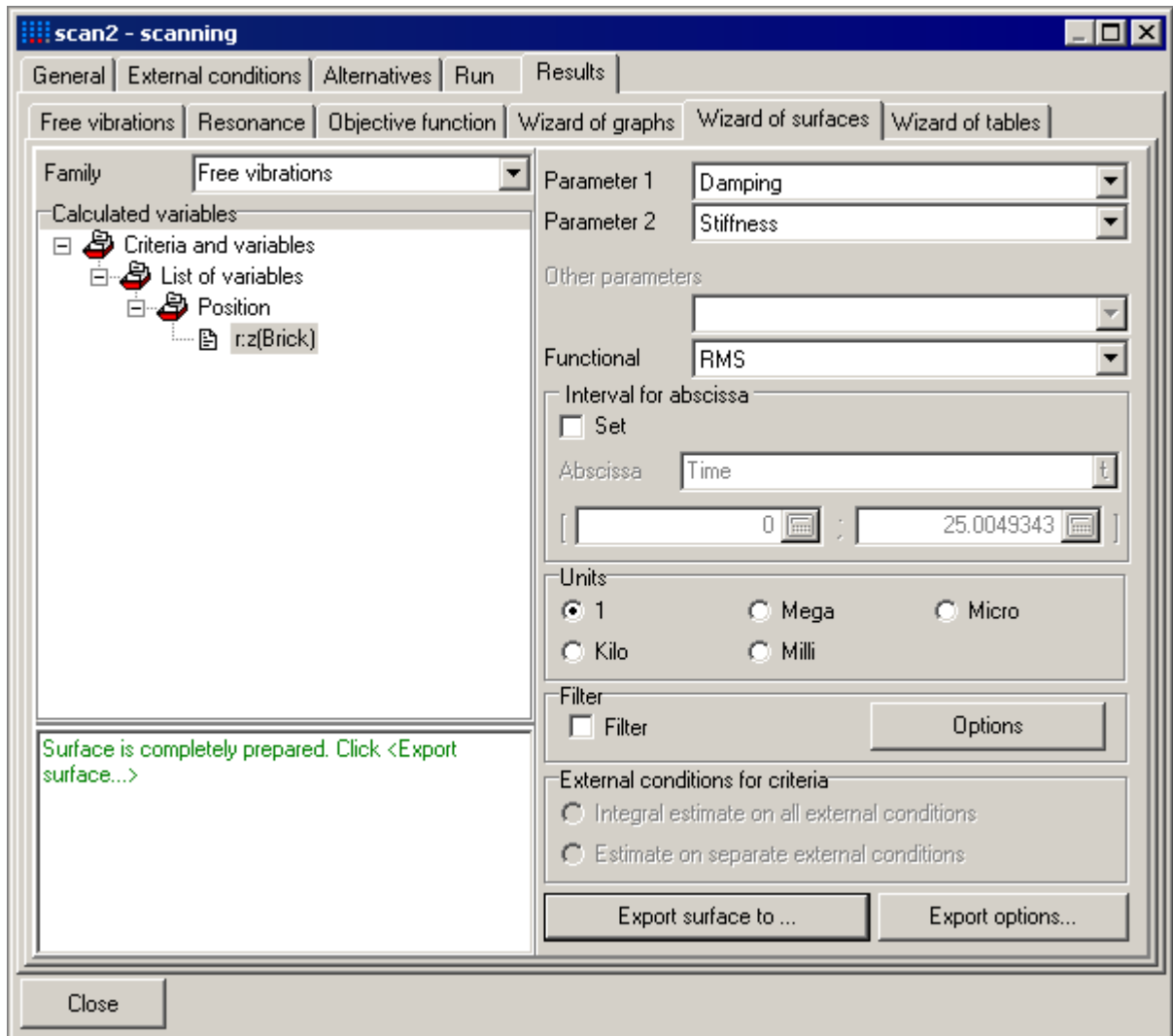


Figure 6.13. Wizard of surfaces

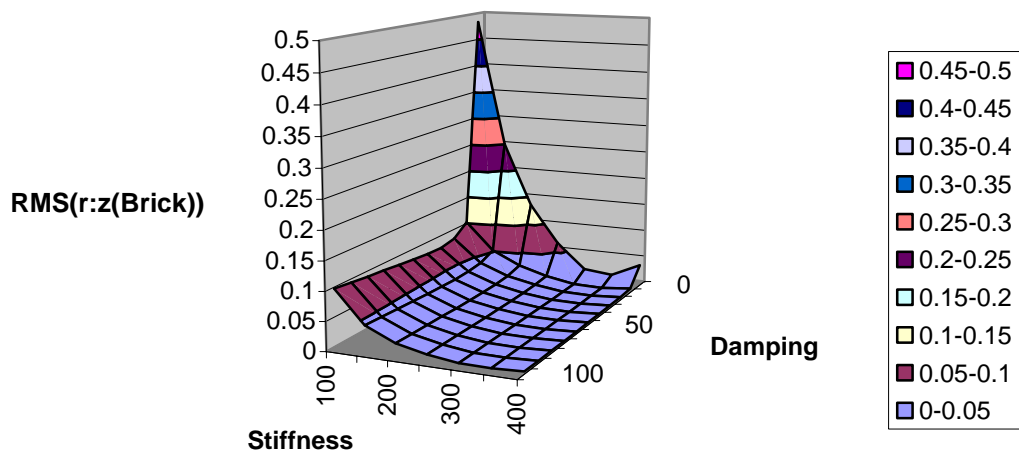


Figure 6.14. Exported surface

6.2.6. Methodical remarks

Here some comments and methodical remarks are considered.


Project structure

The root directory of the scanning project includes subdirectories with models that are considered in this project. Once a model is added to the scanning project its source files are copied to the project directory. Then all numerical experiments are fulfilled for this copy. All changes in original model are ignored.

Portability

To copy/move the project it is necessary to copy/move its root directory. Such copying does not break the integrity of the project, you will be able to start/continue numerical experiments and analyze results.


Save as

Carrying out applied researches it is often necessary to fulfill the same scanning project with small differences. The simplest way to save the project to another directory ( button), modify the project and start its fulfillment.

Data files

After fulfillment every numerical experiment several data files are saved. Time history for every variable from the *list of variables* is saved, see Sect. 6.2.3.6. The more variables in the *list of variables* and the more simulation time the more size of data files. Practical experience shows that data files can reach *gigabytes*. So if you have not enough free disk space you have to find compromise between the number of variables in the *list of variables*, simulation time and size of data files of the scanning project.

Deleting data files

To delete all data files use  button on the **General** tab. You also can delete data files for some numerical experiments to recalculate them. Select these numerical experiments in the list of experiments (see Fig. 6.11) on the **Results** tab and select **Clear results** from a context menu.

Project modification

You can modify the project at any stage of its fulfillment. For example, after fulfillment the project you can add a new model to a project. In this case to finish the project it is necessary to fulfill numerical experiments for the new model only.

You can also add some new levels for parameters for any model (family of alternatives). All already calculated experiments will keep their results. It gives a possibility to add new points in design space and thus specify dynamical behavior of the models step by step if necessary.

Also please note that all numerical experiments for every family are fulfilled with the same initial conditions, identifiers, parameters of numerical method, railway, road and other parameters. If any changes in any family settings are found the program shows dialog and asks the user saves these changes with or without clearing data files. Normally you are recommended to clear data files and fulfill all numerical experiments again.

6.3. Optimization

Use the **Optimization | New project** menu command to create a new optimization project. Every optimization project is situated in a separate directory. Therefore it is necessary to point out a directory for the new optimization project. The main window of the optimization project is shown in **Figure 6. 1**.

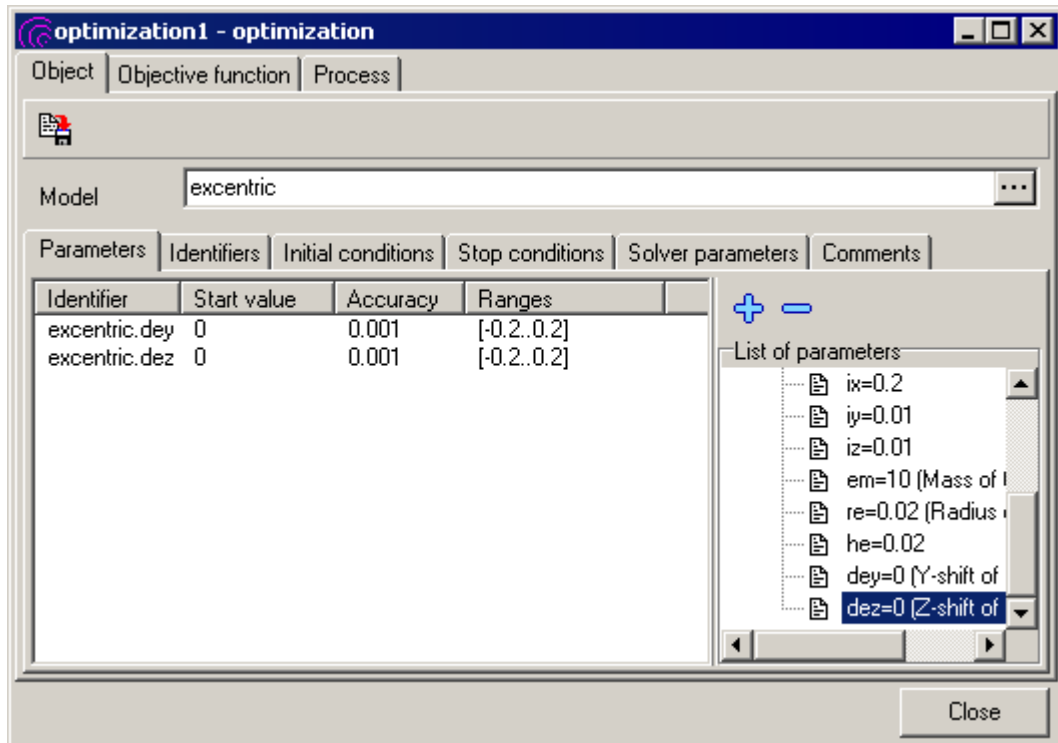


Figure 6. 1. Optimization project

Work flow

- Choose the model to optimize, describe its design parameters (initial values, precision, constraints), the **Object** tab, see Fig. 6.1.
- Describe the objective function, the **Objective function** tab.
- Choose an optimization method and start optimization project, the **Process** tab.

The project finishes when the optimal solution with required accuracy is achieved.

After starting optimization project, numerical experiments begin. A scalar objective function returns a result based in time histories of dynamical performances obtained from each numerical experiment. Optimization method varies the values of design parameters using the goal function values and internal search strategy. The general scheme of optimization is depicted below, see Fig. 6.1.

6.3.1. Description of an optimization project

6.3.1.1. Description of design parameters

Use the **Object | Parameters** tab to describe design parameters. Use the **+** **-** buttons and *Insert* and *Delete* keys to add/remove parameters. Click the necessary identifier in the list on the right to add this identifier to the list of design parameters. Choose initial value, accuracy and constraints for each identifier (**Fig. 6. 2**).

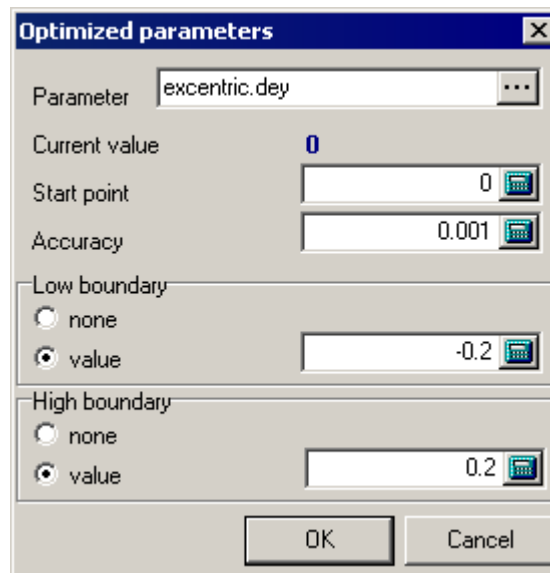


Fig. 6. 2. Description of parameter

6.3.1.2. Identifiers

Here you can find all of the system identifiers (see. Sect. 4.4.2.4). This setting will be used as default setting for each numerical experiment. These values are loaded before each numerical experiment. Of course, if an identifier is changed as a design parameter then such an identifier will be redefined with the correspondent value.

6.3.1.3. Initial conditions

Here you can describe initial conditions each numerical experiment starts with (see Sect. 4.4.2.5).

6.3.1.4. Stop conditions

Here you should describe stop conditions for each numerical experiment (6.2.3.5).

6.3.1.5. Solver parameters

Here you should choose integration method and set its parameters. This setting will be used for numerical integration (see Sect. 4.4.2.1).

Note Take care that the precision of numerical integration corresponds the precision of design parameters. Increasing the precision of design parameters do not forget to increase precision of numerical integration as well.

6.3.2. Description of objective function




The analytic hierarchy process was developed by T. Saaty¹. The method is based on principle of hierarchization, where the main, most common goal consists of several more detailed sub-goals, each sub-goal of the first level consists of the corresponding sub-goals of level two and so on. Every sub-goal has only one upper goal. Different sub-goals affect the upper goal with a different weight.

Further, the analytic hierarchy process involves the method to determine the weight with which the various elements in one level influence the elements on the next higher level, so that we may compute the relative weight of the impacts of the elements of the lowest level on the overall objectives. The method can be described as follows. Given one goal, e , and its sub-goals of the next level lower, compare the sub-goals pairwise in their weight of influence on e . Arrange the agreed upon numbers, reflecting the comparison, in a matrix and find the eigenvector with the largest eigenvalue. The eigenvector provides the priority ordering, and the eigenvalue is a measure of the consistency of the judgment.

To insert the agreed upon numbers the designer has to compare every pair of sub-goals and give an answer for the question “how stronger the influence of sub-goal B on the upper goal than the influence of sub-goal C on it”, this number will be included in the (B, C) matrix element. If B and C are equally important then the number is 1, if B is weakly more important than C then the number is 3 and so on up to number 9 when the B is absolutely more important than C.

An example of the description of the objective function for solving the rotor balancing problem is considered in [gs UM Experiments.pdf](#)²

Let us consider an example of describing the multi-criteria objective function. Assume that we need to find optimal parameters of the car suspension. We need to provide good enough road holding and comfort driving for driver and passengers. So, we have two criteria: road holding and comfort factor. It is well known that increasing spring stiffness improves road holding but makes worse comfort factor. So here we have conflicting criteria and have to find a compromise between them.

To describe our hierarchy (Fig. 6.15) we need to add one more level ( button) and two more criteria in this level ( button). Then we need to rename them according Fig. 6.15 via context menu and then interconnect them using the  button.

Note. This tool supports any number of levels and criteria at these levels. In this example we have a hierarchy of two levels and three criteria.

There are two kinds of criteria in the hierarchy: intermediate and terminal criteria. Intermediate criterion is the criterion that is expressed via other criteria. Terminal criteria are final criteria in the hierarchy; they do not expressed via other parameters and are estimated based on results of numerical experiments.

¹ Saaty, T. *The Analytic Hierarchy Process*, McGraw-Hill, 1980.

² http://www.umlab.ru/download/50/manual/eng/g_s_UM_Optimization.pdf

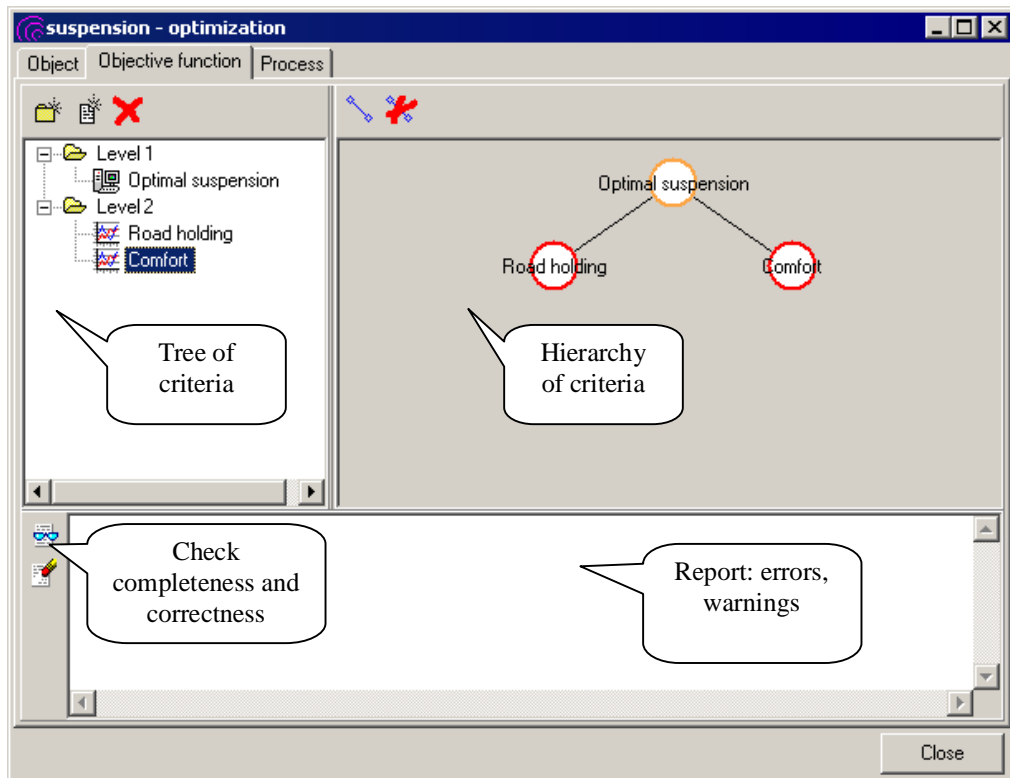


Figure 6.15. Multi-criteria objective function

6.3.2.1. Intermediate criteria

Estimation of an alternative relative to intermediate criteria is calculated as weighted sum of estimation of sub-goals. Weights of sub-goals reflect significance of goals relative to the upper goal.

To weight sub-goals you should double click the criteria in the *Tree of criteria*, see Fig. 6.15 or select the **Weight sub-goals** context menu item, see Fig. 6.16.

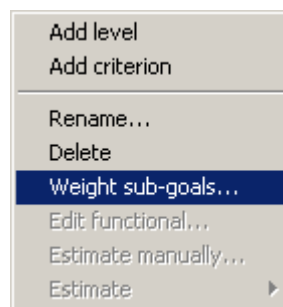


Figure 6.16. Context menu for intermediate nodes

In the **Weight criteria** dialog (see Fig. 6.17) it is possible to input weights of sub-goals directly or with the help of *Matrix of pairwise comparison* (recommended). Click **Compare on MPS** to start **Pairwise comparison wizard**. It will lead you through series of pairwise comparisons and then create so call *matrix of pairwise comparison* and finally calculate weights of sub-goals, see Fig. 6.18.

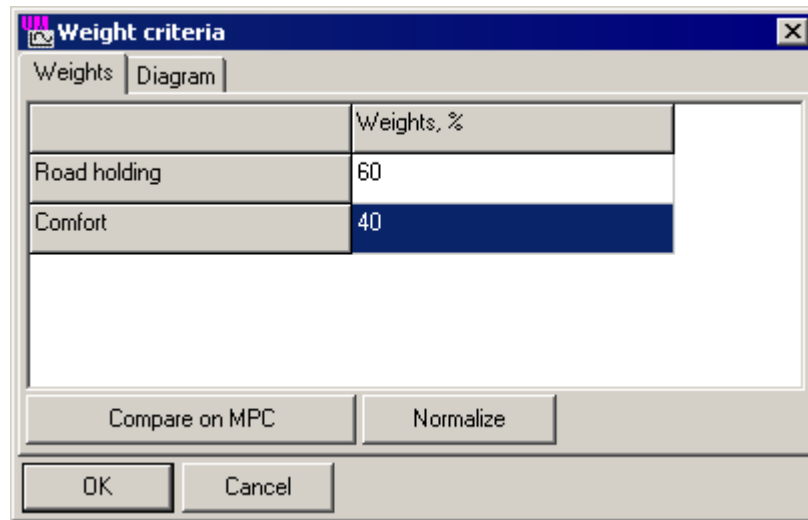


Figure 6.17. Weighting criteria

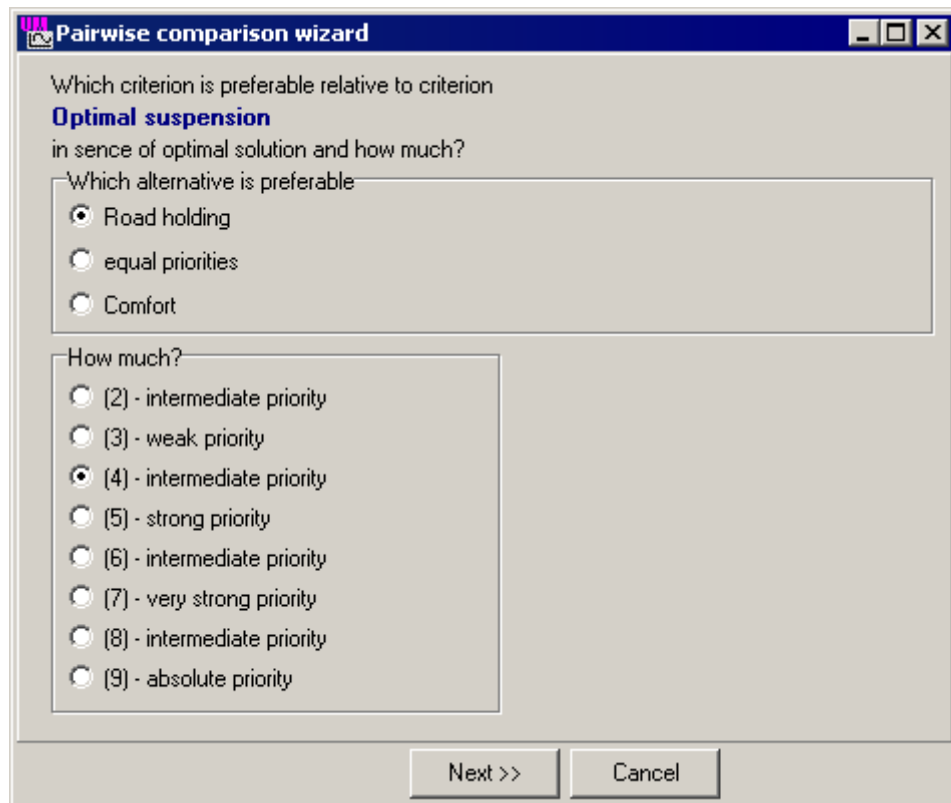


Figure 6.18. Pairwise comparison wizard

6.3.2.2. Terminal criteria

Estimation of an alternative relative to terminal criteria is fulfilled according to some user-defined rules. A designer should describe firstly what dynamical variable will be used for estimation and then the way how to analyze a time history of the variable to get estimation. The designer can use any variable that can be formed with the help of **Wizard of variables** (for more detailed information about **Wizard of variables** please turn to “Sect. 4.3.2 Wizard of variables” of UM User’s Manual). Besides wide selection of dynamical performances **Wizard of variables** supports some basic operations on variables that gives the designer a possibility to create practically any variable that would correspond to the sense of the optimization problem.

Variable

Create a variable with the help of **Wizard of variables** (**Tools/Wizard of variables** menu item) and drag it to the correspondent node of hierarchy of criteria. For example, the variable that correspond to *Road holding* is the normal force between road and a tyre; and vertical acceleration of the car body is the criterion for *riding comfort*.

Functional

After setting the variable it is necessary to describe rules for transformation its time history to a number – estimation of an alternative relative to corresponding criteria. Double click on a terminal criterion or select **Edit functional** context menu item, see Fig. 6.19. New dialog window (see Fig. 6.20) appears.

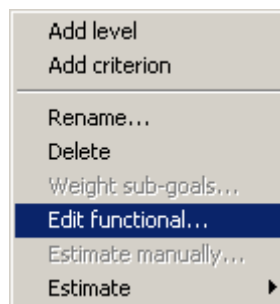


Figure 6.19. Context menu for terminal criteria

Let us come back to our example with car suspension. It is know the subjective riding comfort sense in the simplest might be expressed via Root-mean-square (RMS) of time history of acceleration. So, let us use the RMS functional to estimate riding comfort criterion, see Fig. 6.20.

The **Estimation is based on** group has two options: **value of functional** and **membership function**. Let us consider these options more detailed.

Value of functional

In some cases, when acceptable values of functional are limited in standards, the designer can limit acceptable values of functional turning on the **Use boundaries of values of functional** flag. In this case all alternatives those values of functional exceed the bounds will be ignored and excluded from the subsequent analysis.

By default analytic hierarchy process is intended for solving the maximization problems. To solve the minimization problem (when optimal solution correspond to minimal value of the functional) turn on the **Minimize functional** check box.

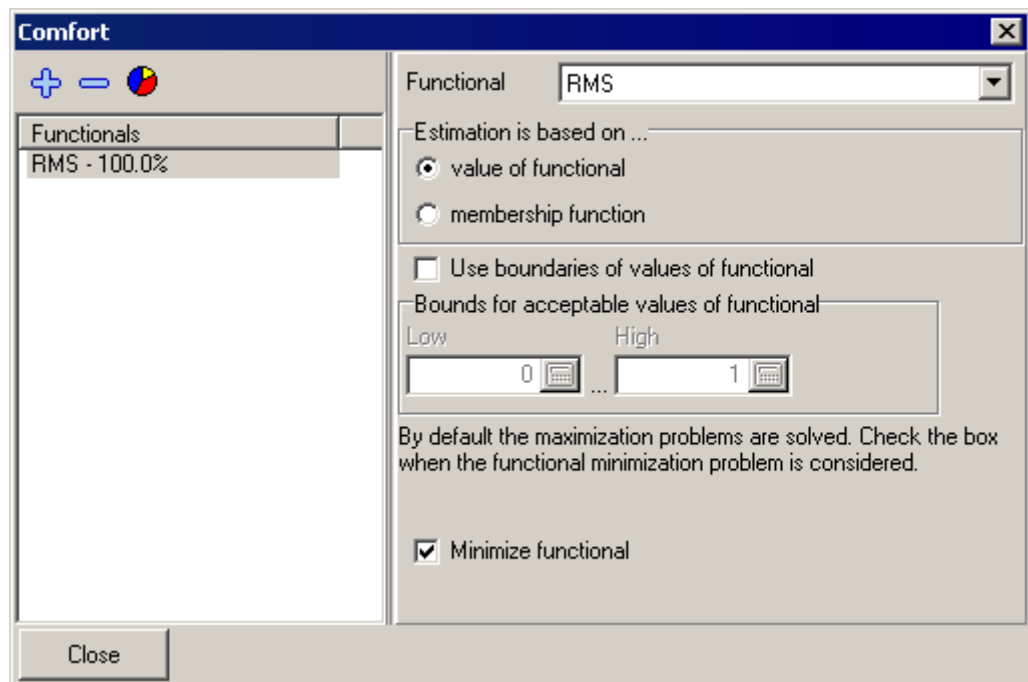


Figure 6.20. Describing functionals

Membership function

Concept of membership function is introduced in the fuzzy logic theory. Using the estimation that is based on membership function the designer should firstly input the bounds for acceptable values of functional and secondly choose the shape of membership function.

The membership function method uses the membership functions to map simulation results into dimensionless scale [0, 1]. Some frequently used membership functions are given in Fig. 6.21. Here the normalized estimation of a performance is laid off as abscissa (0 corresponds to low boundary, 1 – high boundary, see Fig. 6.23) and the weight of alternative relative to criterion is laid off as ordinate. Membership functions should be chosen so as the better value of a functional corresponds to greater value of membership function, see Fig. 6.23.



Figure 6.21. Frequently used membership functions

For example, the following comment may be given for the first membership function in Figure 10: “the less estimation of a performance the worse the weight of alternative, lower values are poorly acceptable” and for the fourth one: “middle estimations of a performance are most acceptable”.

Note. Membership function that is shown in Fig. 6.24 is used in the cases when it is not important what exactly value the functional has but important that functionals are in the preset interval. In other words all alternative those results show acceptable values of functional will be given the same estimation. All others will be eliminated.

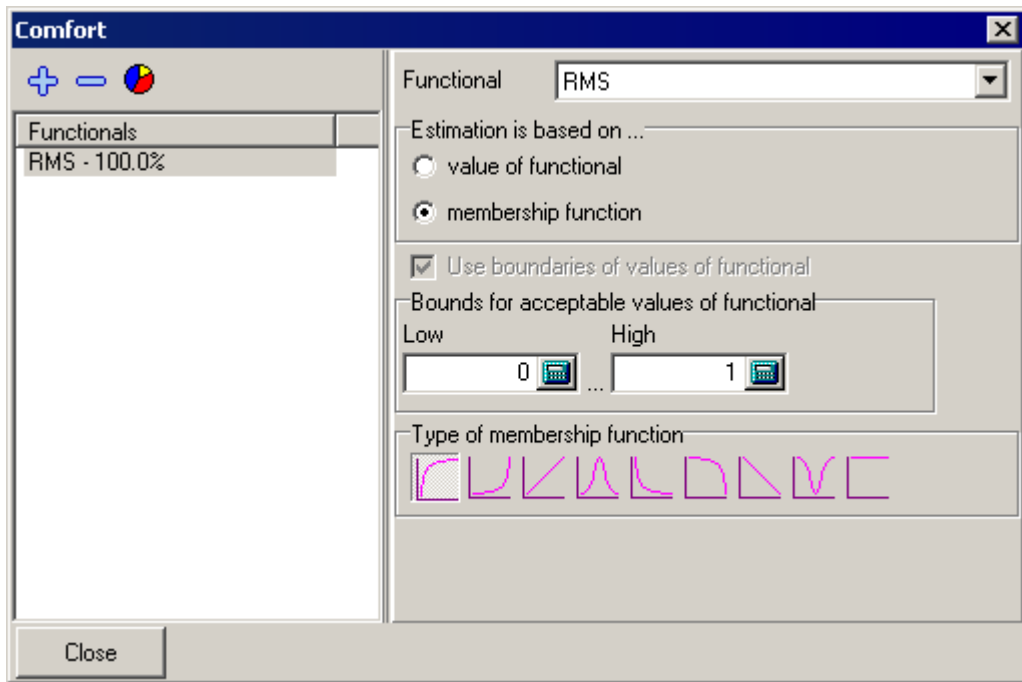


Figure 6.22. Estimation based on membership function

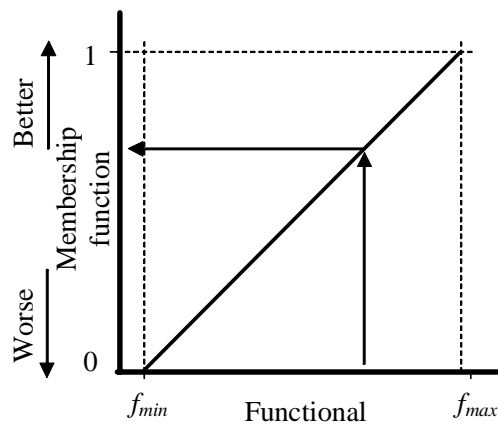


Figure. 6.23. Membership function

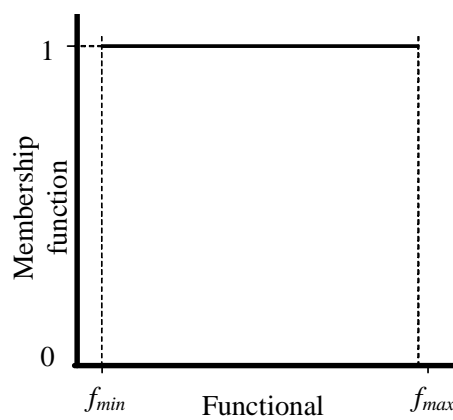


Figure 6.24.

Comparison of estimation methods

- 1) Estimation based on membership function gives a designer a possibility to select as well nonlinear membership function. In this sense using membership functions helps the user to form the objective function more flexible.
- 2) On the other hand using membership functions demands obligatory input of the bounds for acceptable interval that as a rule unknown at the first stage of the optimization and it needs to fulfill some additional research to obtain these boundaries.