

ESTECO
Achieving Perfection

mode **FRONTIER**

Multi-Objective Design Environment

General Information

Copyright © 1999 - 2002 ES.TEC.O. s.r.l. All rights reserved

General Information

Contents

[Overview](#)[Why It's the Best](#)[Who should use it](#)[How does it work](#)[Technical Features](#)[FAQs](#)

Overview

Describing **modeFRONTIER** is very simple: It is the best solution for design optimization. Using its intuitive GUI (Graphical User Interface) you can easily describe your problem, define a set of constraints on the input and output variables, and express your goals. Then, just sit back and relax, while **modeFRONTIER** finds the set of input parameters that meet, as closely as possible, your design goals. And **modeFRONTIER** does this by taking advantage, and optimizing the usage, of all of your existing resources, whether they sit on your desk, in your company's computer room, or at an ASP (Application Service Provider) across the globe. All you need to take advantage of **modeFRONTIER** capabilities is to have a product or process which:

- is complete enough to allow the significant measures of its business effectiveness to be evaluated (preferably objectively and consistently via a clearly specified evaluation process)
- is one candidate from a well defined class of designs, allowing cause and effect to be established between a design change and the design's business effectiveness

Why It's the Best

Our claim that **modeFRONTIER** is the best solution for design optimization might sound as the classic marketing pitch. However we have plenty of evidence to substantiate this claim. For example:

1. **modeFRONTIER** is the only *truly multi-objective* commercial product on the market. Our competitors claim to have multi-objective products too, but in reality what they optimize is at best the weighted sum of a set of objectives. So, if for example you are really interested in maximizing strength and minimizing weight, you'll be forced to maximize something like the weighted sum of strength and the inverse of weight. In other words, those product force you to make a trade-off decision beforehand, and to sum apples and oranges! With **modeFRONTIER** things are radically different, as it is able to manage many goals at the same time, and you'll pick the best design only later on, and from a family of designs that represent the best designs for

various trade-offs.

2. When it comes to picking the best design, **modeFRONTIER** further helps you with highly an effective *decision making tool*. With this tool it is child's play to turn your intuitions into decision rules.
3. **modeFRONTIER** is intuitive and *easy to use*. Only a few days of training are sufficient to make you fully proficient in using it.
4. The various exploration and optimization algorithms can be mixed at will, so that powerful *hybrid optimization techniques* can be easily implemented.
5. Instead of using a legion of public-domain algorithms, **modeFRONTIER** does optimization by using a small number of *highly efficient and proprietary algorithms* developed by our team of researchers in multi-objective optimization.
6. **modeFRONTIER** is not only multi-platform (as it is written in Java), but it has also been designed so that its *projects are multi-platform* (i.e., projects developed under one platform can be used under any other platform without modifications).
7. **modeFRONTIER** embeds *non-linear response surfaces*, which can be used to speed up the optimization of complex problems. In contrast, other products have only polynomial response surfaces, which are usable only in the simplest cases.
8. With **modeFRONTIER** you can change goals and constraints during the optimization phase. This gives you the ability to *steer the optimizer* towards regions of the design space that are more appealing to you, but that you could not predict before starting the optimization.
9. Thanks to its *Plug-In Interface*, **modeFRONTIER** allows you to easily integrate and use third-party optimization algorithms.

Who should use modeFRONTIER?

modeFRONTIER can provide invaluable help to anybody who has goals, and needs to take decisions aimed at achieving those goals. The only requirement is that there must be a way to evaluate the effects of such decisions. This definition includes:

- Engineers who use parametric tools to design products, and evaluate them either through simulations or laboratory testing;
- Engineers involved in the set-up of critical manufacturing processes;
- Managers who design or review business process;
- Decision makers who need to make inferences from complex models;
- Researchers looking for innovative and unorthodox designs.

Each of these users has different needs and different constraints, dramatically different design spaces and design evaluation processes. Yet, **modeFRONTIER** can enable all of them to reach their goals better, faster, at lower cost, and with higher confidence. **modeFRONTIER** does this by squashing the need for guesswork, and by automatically identifying the shortest path to the best designs. The user only needs to use his/her problem-specific knowledge to tell **modeFRONTIER** what to look for; his/her energy can then be channeled towards more creative and important tasks.

How Does it Work?

Using **modeFRONTIER**, the optimization of a product (or process) is accomplished in four simple steps:

1. [Describing the Problem](#): the user defines the inputs (i.e., parameters that affect the product or process), the outputs (i.e., measures of (some of) the characteristics of the product or process), and the resources to be used to estimate the outputs given a set of the inputs.
2. [Setting the Goal\(s\)](#): the user expresses one or more measures of the business effectiveness of the product (or process) as a function of outputs (and possibly inputs).
3. [Pursuing Perfection](#): **modeFRONTIER** uses a wide set of DOE (Design of Experiments) algorithms and optimization techniques to explore the design space, looking, in the most efficient manner, for the Pareto Frontier (i.e., the set of designs for which it is not possible to get closer to one goal without moving away from the others).
4. [Achieving Perfection](#): the user inspects the designs found by **modeFRONTIER**, and selects, with the aid of **modeFRONTIER**'s Decision Making tool, that (or those) representing the best trade-off between the indicated goals.

Technical Features

System Features

- True multi-objective optimization aimed at finding designs on the Pareto Frontier
- Seamless handling of both continuous and discrete variables
- Complete platform independence
- Integration with any hardware platform, regardless of location (local/intranet/extranet/internet)
- Integration with any software simulation code executable in batch mode
- Distributed architecture for collaborative design
- Support for hybrid exploration and optimization techniques
- Intuitive graphical user interface for logic specification
- Carefully tuned proprietary optimization algorithms for maximum robustness and efficiency
- Linear and non-linear response surfaces for data modeling and approximation
- Availability of decision support and visualization tools
- Integration of third-party optimization algorithms through a Plug-In Interface
- Remote monitoring of optimization progress

Underlying Technologies

Implementation language	JAVA
Distributed object model	RMI
Inter-platform communication protocol	CORBA
User interface model	HTML Browser
System configuration, logic specification	XML

and inter-process
communications

Preliminary Exploration Methods

- Random
- Full factorial
- Reduced factorial
- Cubic face centered
- Sobol
- Taguchi
- Box Behnken
- Latin Square
- Montecarlo
- User defined
- Several proprietary algorithms

Multi-Objective Optimization Methods

- Genetic algorithms (Proprietary)
- De-randomized evolution strategies
- Multi-membered evolution strategies

Local Refinement Methods

- Nelder & Mead simplex
- Conjugate gradient
- Quasi-Newton method
- Sequential quadratic programming

Response Surfaces Computation Methods

- Polynomial interpolation
- Gaussian processes
- Neural networks
- Geographical interpolation
- Other proprietary algorithms

Decision Support Tools

- Goal programming
- Weighted sum of objectives
- Non linear utility functions

Post-processing

- Scatter plots
- Frequency plots
- Population plots
- Sensitivity plots
- 3D plots

FAQs

Design optimization hasn't yet become a mainstream discipline in the industry, and it is taught only in few graduate schools. Accordingly, people that comes in contact with **modeFRONTIER** often doesn't have a strong background in optimization, and would like to know more about it. Here we attempt to clear out some of the most common doubts and misconceptions.

1. [Why should I get involved with design optimization?](#)
2. [How does it work?](#)
3. [What is the difference between single and multi-objective optimization?](#)
4. [What is the role of parallel computing in design optimization?](#)
5. [How does the number of design parameters affect the optimization process?](#)
6. [Often optimization techniques make use of response surfaces. What are they?](#)
7. [Isn't the need to clearly define objectives a burden for designers?](#)
8. [Is there a class of problems that are inherently unsuited for design optimization?](#)
9. [What type of design projects can modeFRONTIER handle?](#)
10. [Are there any limits on the software packages I can interface with modeFRONTIER?](#)
11. [Will modeFRONTIER force me to change my work habits?](#)
12. [How will modeFRONTIER impact my productivity?](#)
13. [What is the maximum number of design evaluations modeFRONTIER can support?](#)
14. [How many concurrent processes can modeFRONTIER control?](#)
15. [Is there a limit on the number of input and output variables a modeFRONTIER project supports?](#)
16. [And what about constraints and goals?](#)
17. [Does modeFRONTIER impose any format constraint on my applications' input/output files?](#)
18. [Can I integrate modeFRONTIER with an application running across the Internet?](#)
19. [Can I monitor the progress of the optimization remotely?](#)
20. [Can I integrate custom optimization algorithms in modeFRONTIER?](#)
21. [Does modeFRONTIER allow to import/export design data?](#)
22. [Is modeFRONTIER fail-safe?](#)
23. [Is a demo version available?](#)
24. [What is the licensing policy?](#)

[Previous](#)

[Top](#)

[Next](#)

FAQs

1. *Why should I get involved with design optimization?*

Optimization is about selecting the best option from a range of possible choices. Because a job worth doing is worth doing well, it is natural to consider this when designing a new product. Why not produce a design which is the best in its class, if you have the means to do so? It may not cost any more in time and money to design and produce; and it might cost less. The customers for your product will be more pleased with it. Everyone will be happier, except your competitors.

2. *How does it work?*

The cornerstone of design optimization is design search, a process of intelligent exploration of the objective and constraint spaces. To put it simply, the results of those designs that have already been evaluated are used to forecast where, in the design space, better designs are likely to be found.

Search methods have been developed following two widely different approaches. Firstly, many deterministic methods have been proposed, usually based on assuming that the objective function being investigated is continuous and reasonably smooth. These are usually called *local* methods.

More recently, new algorithms have been developed to create entire populations of designs by breeding a set of initial designs. The "parents" of each new element are selected balancing the need to retain good designs against the desire to explore radically new designs. These methods, dubbed *global*, they tend to explore the entire design space, and enlist the so-called Genetic Algorithms.

3. *What is the difference between single and multi-objective optimization?*

In assessing a product, there is usually no single aspect by which it is judged. Cost, performance, operational suitability, durability, are usually all important factors. However, it is hard to add them up into a single figure of merit. What happens in real life is that the designer balances these factors off against each other to arrive at what he thinks is the best combination of properties in the final design.

This balancing act is complex and crucial at the same time. And, of course, it can be very subjective.

Certainly, each factor is relevant, in the sense that a very poor value for any one factor may make the design unacceptable. However, it is usually not straightforward to understand how far one factor can be improved at the expense of others. For example, one factor may be judged sufficiently good; so there is no merit in sacrificing other things to improve it further. Conversely, there may be little value in a small improvement; but reaching a threshold value may suddenly give large benefits, such as the entry into a new market. So the problem of selecting the right metrics is complex and nonlinear.

Accordingly, for most problems it is inadequate to fix on a single metric for design optimization, or to combine metrics in a premature way (i.e., single-objective design optimization). Means are needed to address problems with multiple objectives (i.e., multi-objective design optimization).

4. *What is the role of parallel computing in design optimization?*

There are two ways in which parallel computing can play a major role in optimization. One is the traditional

way, in which a parallel version of an analysis algorithm is used to provide a solution in a faster clock time, at the expense of more overall processor power. The other way uses the natural parallelism inherent in some optimization algorithms. When a number of designs are being considered simultaneously, they can be evaluated in parallel. This is particularly appropriate, for example, global search algorithms.

5. *How does the number of design parameters affect the optimization process?*

Adopting a model with a large number of parameters may appear to give more freedom for the choice of the final design, resulting in a better design. However, the more dimensions the parameter space has, the larger the design space to be searched for optimum designs. In reality the space dimension, and therefore the computational cost, snowballs as the number of parameters increases. Since the computation is quite likely to have to be guillotined for cost reasons, the choice of a model with a large number of parameters may actually be detrimental. In fact, a superficial search of a large design space, often leads to a less effective result than a more thorough search based on a carefully-chosen model.

6. *Often optimization techniques make use of response surfaces. What are they?*

For many design problems, the computational cost of one design evaluation is so large that the number of evaluations that can be afforded is very limited. Search algorithms do the best they can to minimize the number of designs evaluated, but for expensive to compute objective (or constraint) functions, it is often necessary reduce these evaluations further. One way of doing so is by building from the existing evaluations a reduced model of the relationship between design parameters and objective functions. It becomes then possible to use this reduced model to locate design candidates for evaluation. This model is known as a *response surface*. All the traditional forms of mathematical interpolant can be used. More recently nonlinear forms (such as neural networks and Gaussian processes) have also been used.

7. *Isn't the need to clearly define objectives a burden for designers?*

Absolutely not. On the contrary, the clarification of a design's measures of success is essential to ensuring that the design team is addressing the right set of problems for the business. The use of optimization ensures that this issue is brought clearly into focus, and that sensitive areas of the problem receive their rightful attention.

8. *Is there a class of problems that are inherently unsuited for design optimization?*

Yes. Unfortunately design optimization cannot be applied to problems in which for one or more of the design parameters it is not possible to define a meaningful coordinate system. In these cases, the impossibility of using optimization algorithm stems from the inability to order the possible values for the parameter, and thus to define the concepts of far and near. For example, a design parameter could be the material used to build something. Knowing that steel performs better than copper provides no information to the search algorithm to guess whether iron or lead should be tried next. Thus it becomes impossible to define concepts of good design regions and good search directions. A similar case could be made for a color parameter, as there is no ordering for colors.

This might seem to exclude a large set of problems from the realm of optimization. However, it is often possible to turn these problems into manageable ones by carefully examining what aspects of the parameter affect the quality of the design. For example, if what matters is the electrical conductivity of the material, we could think of basing the optimization on this value. Once the optimal design has been found, the material whose electrical conductivity is the closest to the optimal value can be evaluated. Similarly, we could find out that only the hue or the saturation of the color are important, and we could use those parameters to carry out the optimization.

9. *What type of design projects can modeFRONTIER handle?*

modeFRONTIER is the result of the active collaboration between several important players in the aerospace and automotive industry, and world-leading optimization researchers. As such, both the user friendliness required by the industry and the scientific quality sought by academic researchers, have been part of **modeFRONTIER** since its inception. Thanks to this approach, **modeFRONTIER** is today able to handle virtually any project.

modeFRONTIER's graphical process-flow editor enables the user to describe even the most complex project in a very short time, interfacing multiple applications easily and smoothly. Its scheduler engine can run multiple processes in parallel, explore efficiently huge design spaces, and find optimal solutions quicker than many thought possible. And its state-of-the-art decision making tool allows designers and managers to pick the design that better meets their current trade-off needs.

10. *Are there any limits on the software packages I can interface with modeFRONTIER?*

modeFRONTIER is a multi-disciplinary tool, so it can be interfaced with software packages from any field. However, there is a limited set of tasks that these applications must be able to carry out. They are:

- ◆ read inputs and processing instructions from either the command line or an external file;
- ◆ process the inputs in batch mode (i.e., without human intervention);
- ◆ write the results to an external file.

In reality, the vast majority of today's software packages meets these requirements, and can thus be used with **modeFRONTIER**. This includes anything from engineering simulation packages such as Nastran, CFX and Ansys, to less specific tools such as Matlab and MathCAD, to office automation tools like Microsoft Excel.

11. *Will modeFRONTIER force me to change my work habits?*

Very little. We have striven to impose on the user as few restrictions as possible. For example, constraints and functions of variables can be easily defined within **modeFRONTIER**, without the need to compute them separately. Also, multiple conflicting goals are handled automatically, without the need to take trade-offs decision that the user was not used to make before.

All in all **modeFRONTIER** seamlessly integrates into the way designers work today, requiring only minor flexibility on their part.

12. *How will modeFRONTIER impact my productivity?*

It will most definitely improve it, as **modeFRONTIER** dramatically reduces the time required to evaluate, and improve, a design. More importantly, with **modeFRONTIER** the designer is spared from all those repetitive, boring, and time consuming activities that characterize design improvement efforts. Thus, his/her energies and problem-specific know-how can be focused on more intellectually stimulating, and fulfilling, tasks.

13. *What is the maximum number of design evaluations modeFRONTIER can support?*

Of course, it depends on the amount of available memory. It is also a function of other factors, such as the number of inputs, outputs, constraints, and goals. In general, on an average PC (128Mb RAM) ten thousand designs for a complex project can be easily handled, and thus this parameter is rarely a limiting factor.

14. *How many concurrent processes can modeFRONTIER control?*

It all depends on the memory availability and CPU speed, as multiple process require multiple threads, which are notoriously resource intensive. Fortunately, running concurrent processes usually makes sense only on

multi-processor machines, which tend to be resource-rich. For example, we have successfully used **modeFRONTIER** to control 64 concurrent processes on a 256 nodes super-computer. Doing this on a single processor PC is certainly not recommended.

15. *Is there a limit on the number of input and output variables a modeFRONTIER project supports?*

There is no hard limit on either, but nobody can expect an optimization algorithm to do a good job with more than 30/40 input variables. The design space would be simply too large, and, unless there is a very simple relationship between inputs and outputs it would be impossible to achieve meaningful results. Similarly, when too many output variables exists, it becomes very complex to evaluate them, and to figure out how each input variable affects each output variable. Under those circumstances doing any sort of significant statistical analysis is then a pure chimera.

While these might look to some as undesirable limits, the matter of the fact is that those problems fall outside of the design optimization domain, and are usually addressed, necessarily in a less effective way, by operational research.

16. *And what about constraints and goals?*

Again, there are no hard limits, but having more than 10 goals or constraints makes the problem almost impossible to analyze. For example, for a problem with 11 goals the Pareto frontier would be a 10-dimensional surface. Making any kind of trade-off analysis on such a domain its virtually impossible.

17. *Does modeFRONTIER impose any format constraint on my applications' input/output files?*

No. Practically any ASCII format is supported for the input and output files, and the user can use existing files to instruct **modeFRONTIER** on how to input or extract data from these files.

18. *Can I integrate modeFRONTIER with an application running across the Internet?*

All the web services developed by ESTECO on the SP4web platform can be accessed from **modeFRONTIER** using the HTTPS protocol (secure and capable of tunneling through a company firewall). In contrast, arbitrary applications can currently be integrated only if the machine **modeFRONTIER** is running on has RPC access to the remote machine, or if the application has been retrofitted to handle **modeFRONTIER**'s HTTPS requests. We are already in the advanced development stage of a new application which will remove this obstacle, letting **modeFRONTIER** seamlessly integrate with any application, running anywhere.

19. *Can I monitor the progress of the optimization remotely?*

Yes. Today this is possible only if there is a separate web server that allows access to your project's data directory, but a web server will be integrated in the upcoming version of **modeFRONTIER**. Monitoring its progress will then become even easier than it is today.

20. *Can I integrate custom optimization algorithms in modeFRONTIER?*

Most definitely. This feature was part of the initial design, and a Plug-In Interface has been developed to allow external algorithms to be easily integrated within **modeFRONTIER**. The operation of this interface is transparent to the user, so that he/she cannot even tell whether an algorithm is native or plugged in.

Using this technology, several state-of-the-art optimization algorithms have already been successfully integrated in **modeFRONTIER** (e.g., evolution strategies).

21. *Does modeFRONTIER allow to import/export design data?*

Yes, it does. **modeFRONTIER** can read all kinds of ASCII tabular data, and it exports data either as space-separated ASCII tables or in a binary format that can be used to later import the data back into **modeFRONTIER**. Furthermore, you can copy-and-paste directly from and into most spreadsheet tools (e.g., Excel, Quattro Pro, Staroffice, etc.).

22. *Is modeFRONTIER fail-safe?*

As much as possible. As soon as a design has been fully evaluated, its properties are immediately saved on disk, and they are automatically recovered in the event of a computer crash. However, any design running during a crash is lost, and will need to be re-evaluated.

23. *Is a demo version available?*

Yes. You can obtain a fully-featured time-limited demo version by contacting any of our distributors.

24. *What is the licensing policy?*

modeFRONTIER licenses are managed through FLEXlm. It is possible to obtain licenses that are either time-limited or permanent, node locked or floating, and to select only the desired modules. Another important parameter is the number of concurrent processes that one wants to be able to control. Finally, there are licenses for the whole product, and licenses for a batch-only version (i.e., without the graphical interface). Pricing information can be obtained directly from our distributors.

[Previous](#)

[Top](#)

[Next](#)

Describing the Problem

Clearly, the first step in optimizing a design is to describe the problem, and the set of input/output parameters that define the design. In **modeFRONTIER** this process is carried out within an intuitive graphical environment (GUI). Here the user first describes the inputs or parameters that univocally define a design, setting their legal boundaries.

The next step is to describe the process(es) that computes the outputs from these inputs. It is at this stage that the user tells **modeFRONTIER** which resources (hardware and software) should be used to simulate the behavior (and thus the quality) of any given design. As far as software is concerned, **modeFRONTIER** can take advantage of virtually any package, regardless of its discipline, or input/output interchange formats. The only constraint is that the package must be able to compute the outputs from the inputs in batch mode (i.e., without requiring human intervention) This is the case for the vast majority of engineering tools, and **modeFRONTIER** has been coupled successfully to several, including: Catia, Nastran, Marc, Abaqus, CFX, Tascflow, Star-CD, Fidap, Fluent, Adams, Magma, Lagrange, HissD, Straus, Matlab, Femlab, and CAD-Mould.

A similar case can be made for the hardware on which these software packages run: **modeFRONTIER** is able to run software installed on:

- the same machine it is running on [local];
- machines within the same Local Area Network (e.g., computers in the enterprise computer center) [intranet];
- machines located at other sites within the enterprise connected via dedicated WAN links or VPNs [extranet];
- servers maintained by Application Service Providers anywhere in the world [internet].

Also, **modeFRONTIER** can take advantage of any parallel architecture, from the smallest multi-processor PC boards and Linux clusters, to the largest massively-parallel computers.

The final step to provide **modeFRONTIER** with a full problem description, is to indicate the outputs of the process that are necessary to evaluate the business effectiveness of the design.

Setting the Goals

Once the problem has been defined, the next step is to decide which measure(s) of the product (or process) we want to improve. Unfortunately, when we are looking for a best solution to a given problem in reality we are almost invariably looking for the best compromise between different, and often conflicting, goals. To complicate matters further, the perception of which goal is more important will often be subjective, and it will possibly change over time.

Historically, optimizers have solved this problem by forcing the user to decide from the start which goals are more important, and by how much. Then the optimization algorithms look for designs that support this particular trade-off. To better understand what this is like, suppose that you want to design an engine with a large torque, but with good fuel economy too. And assume also that you have a software tool that, for any given design, will tell you the expected torque (expressed in Nm) and the expected fuel economy (expressed in mpg or l/100Km). Your task is then to write a mathematical expression for a single measure to be maximized (or minimized). If the fuel economy is for you as important as the maximum torque, how would you express it? And if one were twice as important as the other?

As you can see, this approach is clearly flawed, as it is virtually impossible to assign numbers to our intuitions of what is a better design (i.e., to convert a trade-off into a mathematical expression). On the other hand, when we are faced with two design alternatives, it is far easier to pick what we consider the best.

Accordingly, **modeFRONTIER**'s multi-objective search algorithms aim to identify the set of designs that are on, or near, the **Pareto Frontier**. This mathematical construct identifies the set of designs (called the **Pareto Set**) that cannot be improved with regard to one goal without moving farther away from the others. The final selection will then be based on these designs.

With **modeFRONTIER** defining the goals is then very easy, as the user only needs to list the measures describing the business effectiveness of a design. The problem of selecting the appropriate trade-off between these goals is pushed to a later time, when it is much more manageable.

Pursuing Perfection

Once the problem has been described, and the goals have been set, it is time for **modeFRONTIER** to do its magic and to find designs that are near the Pareto Frontier. **modeFRONTIER** does this by using a combination of optimization technologies and strategies, which can be classified as follows:

- **Preliminary Exploration Methods**: used to provide an initial sampling of the design space, include algorithms derived from Design Of Experiments (DOE) theory.
- **Multi-Objective Optimization Methods**: used to find designs that lie anywhere in the design space but near the Pareto Frontier, include genetic algorithms and evolution strategies.
- **Local Refinement Methods**: used to investigate the space around a given design, these single objective methods look for a local maxima or minima employing so-called Hill Climbing algorithms.
- **Special-Purpose Plug-ins**: third-party optimization algorithms that communicate with **modeFRONTIER** through its proprietary Plug-In Interface and provide solutions to specific problems.

The approach used by **modeFRONTIER** to find optimal designs can be controlled by the user and can vary from the use of a single technique, to an hybridization of several techniques (e.g., one or more DOE methods can be used to generate an initial population of designs, which is then evolved using a Genetic Algorithm, and whose best candidates are then refined through an Hill Climbing algorithm).

It is important to note that, as the numerical models used to simulate a design are often computationally heavy, it is often unfeasible to carry out a large number of design evaluations. To overcome this problem it is possible to complement these expensive computations with an approximation of the design objectives computed using the available data sets. The values obtained using the approximation surface (called a **Response Surface**) are then used to suggest new candidate designs, which are computed and whose values are used to refine the response surface.

The combination of this tool with the others provided by **modeFRONTIER** leads to the most effective and robust optimization technique available.

Achieving Perfection

Once the set of optimal alternatives (the Pareto set) has been found, the final step is to select the best solution (or at most a very limited set of ideal candidates). The difficulties associated with this choice are obvious, as they are often subjective and cannot be quantified easily.

Classical methods to assist the user in this choice, such as goal programming and using the weighted combination of the objectives, are also of little help. In **modeFRONTIER** this problem is solved thanks to our Decision Making tool, which converts the user's intuitions (such as, design A is better than design B, and design C is better than design D) into an algorithm that extracts the best solution.

Verifying the sensitivity of this solution to parameters changes is then just a few key strokes and mouse clicks away. Finally, once the robustness of the solution has been so verified, perfection has been achieved.
