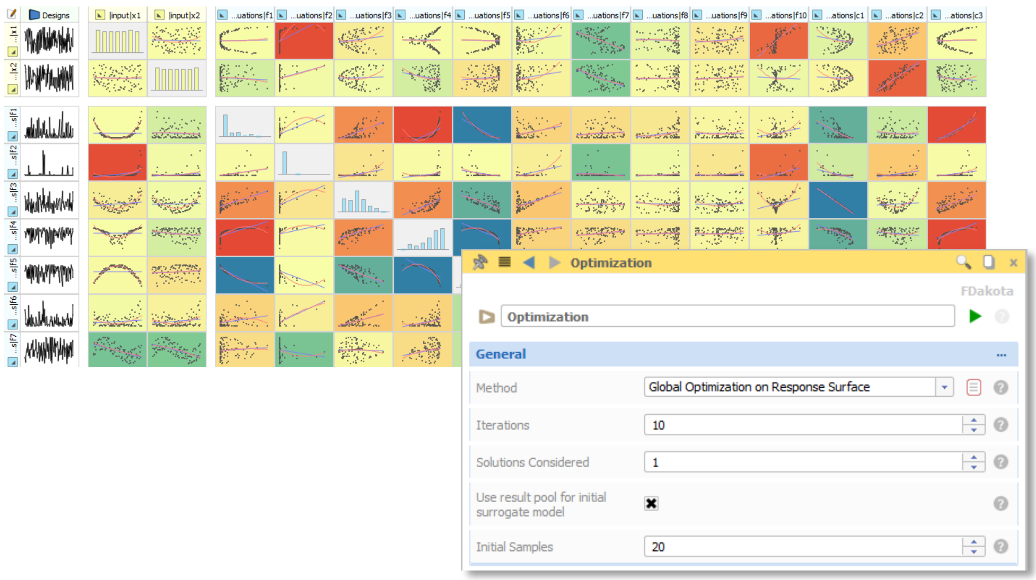
Dakota:
Using Advanced Optimization Strategies in CAESES®

Besides the standard sampling and optimization methods of CAESES®, you can also run additional advanced optimization strategies such as response-surface based methods. *Response surfaces* are surrogate models of a real engineering problem, and by carefully generating and using them you can save tremendous computational time. The mathematical methods behind response surfaces can be quite complex, while using them in CAESES® is easy. There are also further strategies available such as multi-start methods which run local optimizations from different starting points simultaneously – again very easy to use with a single click.

CAESES® plugs in these new methods and advanced strategies by wrapping [Dakota](), the optimization toolkit from *Sandia National Laboratories*. Users of the *AdvancedOpt* add-on of CAESES® simply get a new design engine called "Dakota" in the *optimization* menu where you can choose between preconfigured strategies. This add-on also enables users to do post-processing with handy and beautiful 2D charts.

Experienced Dakota users can also define and use their own Dakota input files in CAESES®, and fully customize the graphical user interface (i.e. customizing the visible design engine input attributes for each method). With this customization possibility, a huge variety of state-of-the-art numerical methods can be immediately used for comprehensive investigation of almost any engineering application.

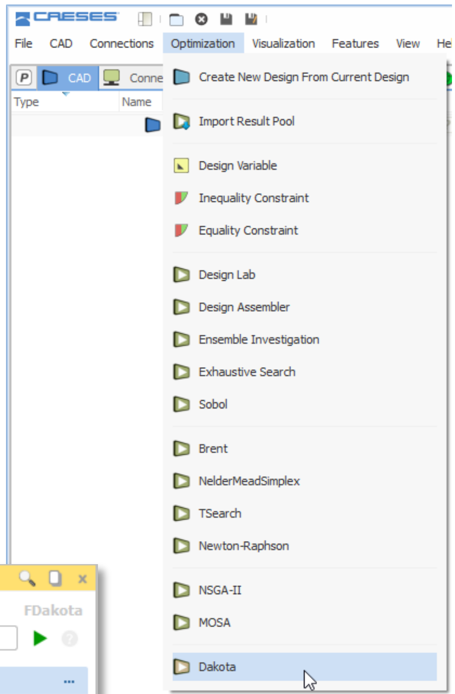
This tutorial gives a brief overview about how to run these additional optimization strategies in CAESES®.

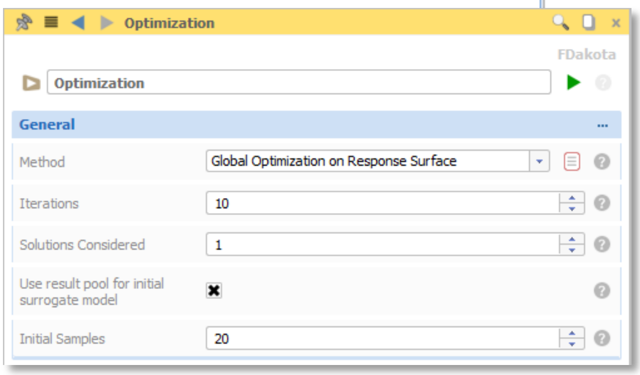


1 Getting Started

The *AdvancedOpt* add-on of CAESES® comes with a set of preconfigured algorithms that can be readily used. The following steps outline how you can run these methods from within your CAESES® project:

- Choose *optimization > dakota*.
- Select one of the preconfigured methods.
  These methods are explained in more detail on the next pages.
- Set a good name for your design engine, e.g. "Optimization", "DoE", "Study", ...
- Configure your design variables, evaluations and constraints as usual.
- Run the design engine.





2 Preconfigured Methods

The preconfigured methods complement the existing optimization strategies of CAESES®. Here is a brief overview of them:

- **Sensitivity Analysis:** This method is a good starting point for many tasks and uses a *Latin Hypercube Sampling*. Suitable for simple parameter studies and for finding correlations and trends. These randomly-generated designs can be recycled for building up a response surface, see "Global Optimization on Response Surface" below.

- **Local Optimization:** Gradient-free local optimization strategy. Recommended method to further optimize an existing good design.

- **Local Optimization Multi-Start:** This strategy allows you to run several local optimizations from different starting points at the same time. The starting points are randomly generated. For each local search, a gradient-free strategy is applied.

- **Global Optimization:** Single-/Multi-Objective Genetic Algorithm (MOGA). It can be used for global optimization tasks. Note that genetic algorithms need many evaluations. This makes this method suitable only for problems where the evaluation (analysis) is not too expensive. The initial population size as well as the number of generations can be set by the user.

- **Global Optimization on Response Surface:** In this method, a MOGA is conducted on a response surface that is iteratively built-up. For the initial response surface, data from a previous run (e.g. a sensitivity analysis) can be considered as well. The best designs of each MOGA run get evaluated and are added to the response surface. This updates and improves the response surface model in each iteration. With this approach, the method tries to reduce the number of expensive evaluations such as CFD runs. Suitable for single- and multi-objective problems.

3 Best Practices

There is no optimization method that can be recommended as the solution for each engineering problem. However, here are some best practices that can guide you in selecting the right algorithm:

- **Step 1: Sensitivity Analysis**

  As a starting point, run a sensitivity analysis to get a better understanding of your engineering problem. Check the 2D charts for correlations (colored by red and blue, plus regression curves), and find out which variables you can omit in the next step. The lower the number of variables, the less computational time you will have during optimizations.

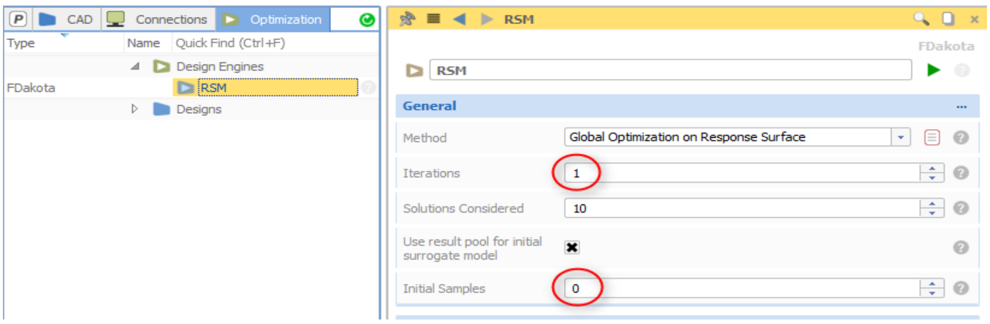
- **Step 2: Check the Response-Surface Based Optimization**

  You can try out whether a response-surface based optimization is suitable for your problem. Use a low number of iterations in the first stage by setting the attribute “Iterations” to 1, 2 or 3.

  The attribute “Solutions Considered” defines how many designs from the Pareto Front are picked and evaluated (e.g. ‘10’ corresponds to 10 CFD evaluations *in each iteration*).

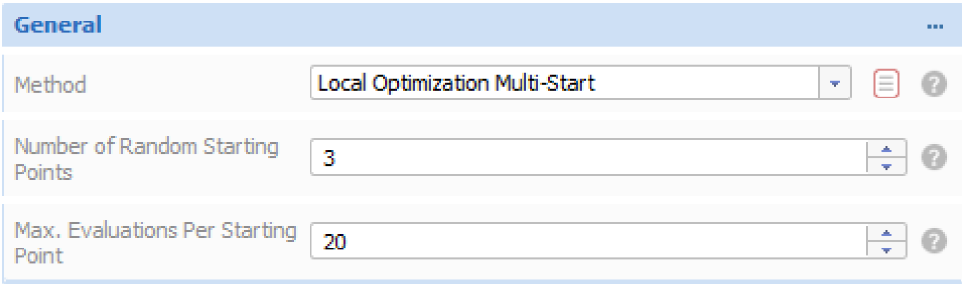
  Important: Check the generated designs in the design table i.e. whether they improve and head into the right direction. If it looks promising, continue with a higher number of iterations (e.g. 10). You can also recycle the previous design results by selecting them as result pool input in the next run.

  If you set “Initial Samples” to ‘0’, then no sampling takes place for building the initial response surface. Instead, you can choose the designs from the previous sensitivity analysis (or from any other run). If you set a number that is non-zero, a Latin Hypercube Sampling is triggered internally to create samples that are used for the response surface. Note: Usually, the number of initial samples needs to be ~5 times the number of design variables.



▶ **Alternative: Multi-Start Local Optimization**

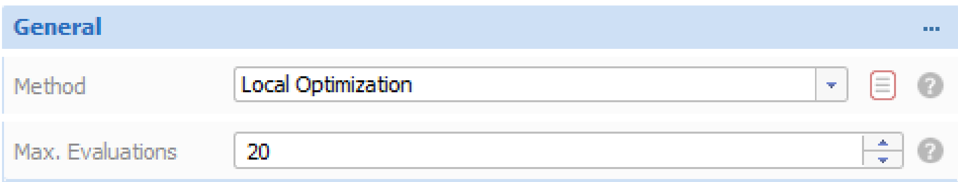
If the response surface method seems to fail for your task, then you can try to run gradient-free local searches in your domain. The multi-start strategy allows you to run these local optimizations from several different starting points. You can start with a lower number of maximum evaluations to observe trends first.



▶ **Improving Good Designs**

Suitable design candidates e.g. from a sensitivity analysis can also be further fine-tuned by selecting a single local optimization for this specific design. Here is a short guide how to do this:

- Create a manual design of the promising candidate: Activate the design by double-clicking on the design in the design tree. Then, choose *menu > optimization > create new design from current design*.
- Select the existing Dakota design engine and create another one. This automatically takes over the design variable settings.
- Choose "Local Optimization" and run the engine with the current design variable settings (i.e. the ones from the promising candidate).

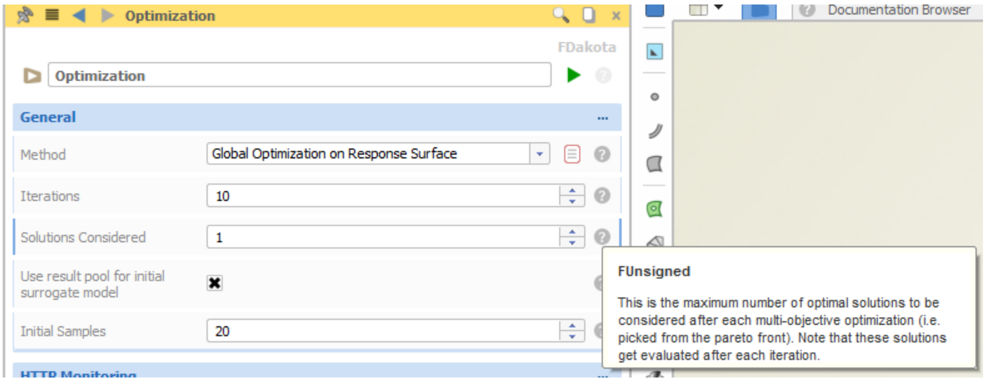


4

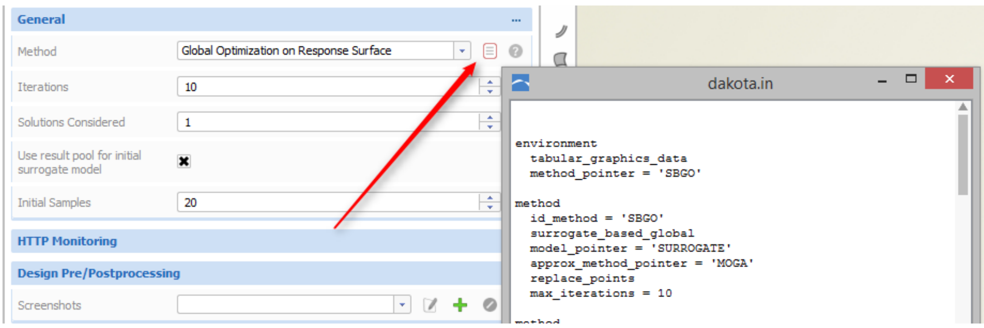
More Information

The brief summary from the previous pages does not cover all the methods' details. How can you find out more? There you go:

- Click on the help icon ("?") for more information about the single attribute.



- Click on the template icon in order to take a look at the input file that is used for Dakota:



- Check out the Dakota user's manual that is also shipped with CAESES®. See the installation folder *FRIENDSHIP SYSTEMS > CAESES > etc > dakota* where you'll find the PDF document.

- And finally, we recommend to check out the Dakota website if you would like to modify the existing preconfigured methods or if you want to set up your own methods – see also the last page of this tutorial:

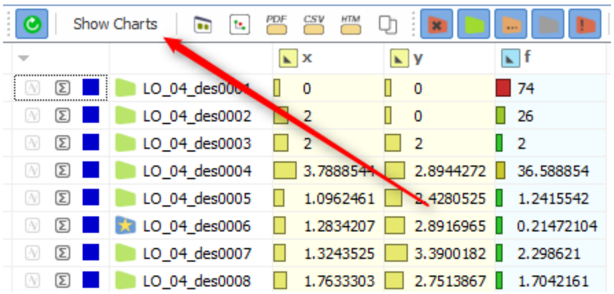
  https://dakota.sandia.gov/

Charts

5

All design engines – including the Dakota engine – generate interactive 2D charts if the *AdvancedOpt* add-on is licensed. The charts help you understand your results and supports you e.g. in finding the most relevant variables as well as picking the best designs e.g. from a Pareto Front.

- You can access them at the top of the result table.



	x	y	f
LO_04_des0001	0	0	74
LO_04_des0002	2	0	26
LO_04_des0003	2	2	2
LO_04_des0004	3.7888544	2.8944272	36.588854
LO_04_des0005	1.0962461	2.4280525	1.2415542
LO_04_des0006	1.2834207	2.8916965	0.21472104
LO_04_des0007	1.3243525	3.3900182	2.298621
LO_04_des0008	1.7633303	2.7513867	1.7042161

- **Correlations:**

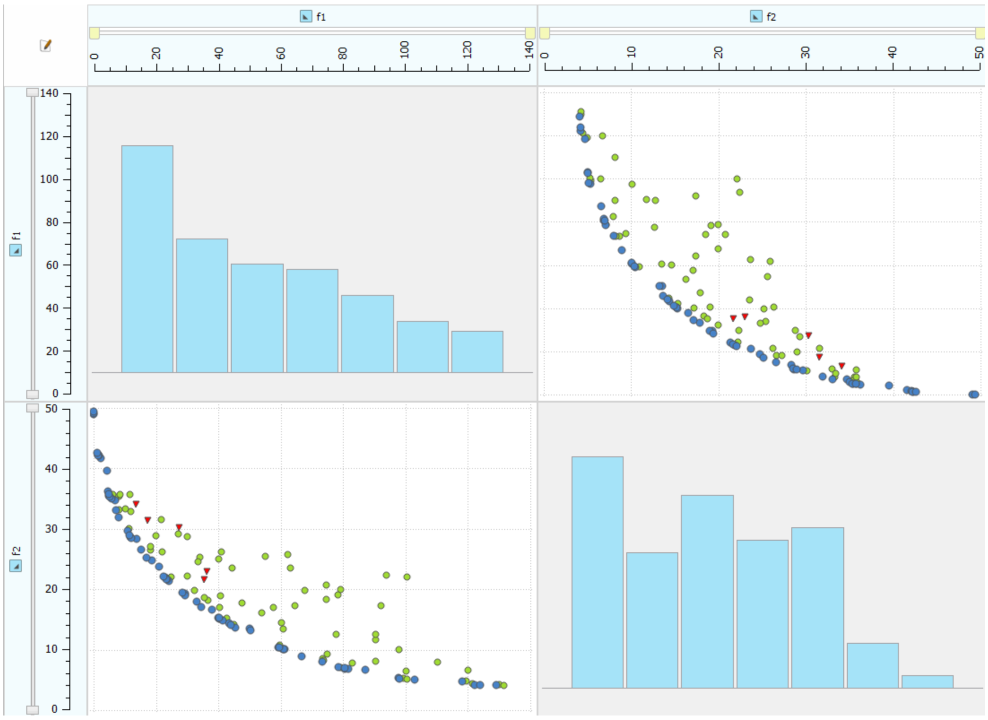
  For a sensitivity analysis, the red colored fields indicate a positive correlation. If the variable value increases, the objective will increase as well. Blue-colored fields indicate a negative correlation.

  You can configure the charts by clicking on the upper left edit icon.

► **Pareto Front (Multi-Objective Optimization)**

In CAESES®, the best designs receive a special blue-colored icon (with a star on it). In addition, the best designs are colored blue in the 2D charts. This makes it easy to pick them e.g. from a Pareto Front where you have several competing objectives. Violated designs receive a red triangle icon in the charts.

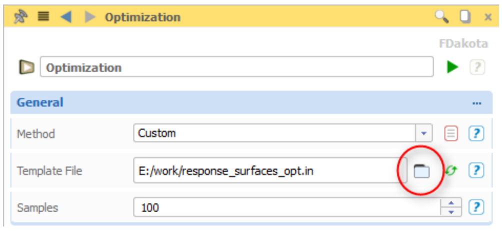
As an option, you can use the controls on the left and on the top to set the boundaries (limits) of what is shown.



5 Custom Templates

You can also set up your own Dakota input files, to make use of the entire Dakota method set, and hand it over to the Dakota design engine.

- As a start, check out the preconfigured input files that are presented in this tutorial. See the installation folder *FRIENDSHIP SYSTEMS > CAESES > etc > dakota*. There you will find these files (*.in) for Dakota that are used by CAESES®. Make a copy of one of them, store them somewhere else and start modifing it.
- For the *method* attribute, select "Custom" and choose your input file.



CAESES® Keywords

- In the *etc > dakota* folder, you will also find a file called "template_definitions.txt". This gives you an overview of the CAESES® keywords that you need to involve for setting up your own template. For instance, CAESES® automatically writes the number of objectives into the Dakota input file, for which a CAESES® keyword `eval_count_objective` needs to be used.

Customizing the CAESES® GUI

- When you write your own method setup for Dakota, you can also configure the user interface. This means that you can control what is transferred to the interface and what's hidden (e.g. when the default value should not be changed). Finally, you can add documentation to a Dakota attribute. Here is a snippet where the Dakota keyword "samples" has been wrapped by the CAESES® user interface:

```
method
  sampling
  sample_type lhs
  <samples, unsigned, 100, Samples, number of samples>
  seed = 12345
```