

CFD EXPERTS

Simulate the Future

WWW.CFDEXPERTS.NET



©2021 ANSYS, Inc.
All Rights Reserved.
Unauthorized use, distribution
or duplication is prohibited.

DesignXplorer User's Guide



ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
<http://www.ansys.com>
(T) 724-746-3304
(F) 724-514-9494

Release 2021 R2
July 2021

ANSYS, Inc. and
Ansys Europe,
Ltd. are UL
registered ISO
9001:2015
companies.

Copyright and Trademark Information

© 2021 ANSYS, Inc. Unauthorized use, distribution or duplication is prohibited.

Ansys, Ansys Workbench, AUTODYN, CFX, FLUENT and any and all ANSYS, Inc. brand, product, service and feature names, logos and slogans are registered trademarks or trademarks of ANSYS, Inc. or its subsidiaries located in the United States or other countries. ICEM CFD is a trademark used by ANSYS, Inc. under license. CFX is a trademark of Sony Corporation in Japan. All other brand, product, service and feature names or trademarks are the property of their respective owners. FLEXIm and FLEXnet are trademarks of Flexera Software LLC.

Disclaimer Notice

THIS ANSYS SOFTWARE PRODUCT AND PROGRAM DOCUMENTATION INCLUDE TRADE SECRETS AND ARE CONFIDENTIAL AND PROPRIETARY PRODUCTS OF ANSYS, INC., ITS SUBSIDIARIES, OR LICENSORS. The software products and documentation are furnished by ANSYS, Inc., its subsidiaries, or affiliates under a software license agreement that contains provisions concerning non-disclosure, copying, length and nature of use, compliance with exporting laws, warranties, disclaimers, limitations of liability, and remedies, and other provisions. The software products and documentation may be used, disclosed, transferred, or copied only in accordance with the terms and conditions of that software license agreement.

ANSYS, Inc. and Ansys Europe, Ltd. are UL registered ISO 9001: 2015 companies.

U.S. Government Rights

For U.S. Government users, except as specifically granted by the ANSYS, Inc. software license agreement, the use, duplication, or disclosure by the United States Government is subject to restrictions stated in the ANSYS, Inc. software license agreement and FAR 12.212 (for non-DOD licenses).

Third-Party Software

See the [legal information](#) in the product help files for the complete Legal Notice for Ansys proprietary software and third-party software. If you are unable to access the Legal Notice, contact ANSYS, Inc.

Published in the U.S.A.

Table of Contents

Ansys DesignXplorer Overview	13
The Ansys Product Improvement Program	13
Introduction to Ansys DesignXplorer	17
Available Tools	17
Design Exploration: What to Look for in a Typical Session	19
Initial Simulation: Analysis of the Product Under all Operating Conditions	19
Identify Design Candidates	19
Assess the Robustness of the Design Candidates	21
Determine the Number of Parameters	21
Increase the Response Surface Accuracy	21
Limitations	22
DesignXplorer Licensing Requirements	22
User Interface	23
Parameters	26
Design Points	28
Response Points	28
Workflow	29
Adding Design Exploration Systems to the Project Schematic	29
Duplicating Existing DesignXplorer Systems	30
Running Design Exploration Analyses	30
Monitoring Design Exploration Analyses	31
Reporting on the Design Exploration Analysis	32
Using DesignXplorer Charts	32
Exporting DesignXplorer Data	34
Design Exploration Options	34
Customizing DesignXplorer with Ansys ACT	40
DesignXplorer Systems and Components	41
What is Design Exploration?	41
DesignXplorer Systems	41
Parameters Correlation System	42
Response Surface System	42
Goal-Driven Optimization Systems	42
3D ROM System	43
Six Sigma Analysis System	43
DesignXplorer Components	43
Design of Experiments Component Reference	44
Parameters Parallel Chart	45
Design Points vs Parameter Chart	46
Parameters Correlation Component Reference	46
Correlation Scatter Chart	48
Correlation Matrix Chart	49
Determination Matrix Chart	49
Sensitivities Chart	49
Determination Histogram Chart	49
Response Surface Component Reference	50
Response Chart	52
Local Sensitivity Charts	53
Spider Chart	53
Optimization Component Reference	54

Convergence Criteria Chart	57
History Chart	57
Candidate Points Results	58
Tradeoff Chart	58
Samples Chart	59
Sensitivities Chart	59
3D ROM Component Reference	60
Design of Experiments (3D ROM)	60
ROM Builder	60
Six Sigma Analysis Component Reference	60
Design of Experiments (SSA)	61
Six Sigma Analysis	62
Sensitivities Chart (SSA)	63
Using Parameters Correlations	65
Running a Parameters Correlation	66
Setting Correlation Properties	66
Setting the Filtering Method for Correlation Parameters	67
Specifying the Filtering Output Parameters	67
Identifying Major and Minor Input Parameters	68
Sample Generation	69
Previewing, Monitoring, and Interrupting the Correlation	70
Correlation Convergence Process	70
Working with Correlation Results	71
Reviewing Filtered Correlation Data	71
Viewing the Quadratic Correlation Information	72
Viewing Significance and Correlation Values	72
Viewing Correlation Design Points	73
Editing Correlation Design Points	74
Working with Parameters Correlation Charts	75
Using the Correlation Matrix Chart	76
Using the Correlation Scatter Chart	77
Using the Determination Matrix Chart	79
Using the Determination Histogram Chart	80
Using the Sensitivities Chart	81
Using Design of Experiments	83
Setting Up the DOE	84
DOE Types	84
Central Composite Design (CCD)	84
Optimal Space-Filling Design (OSF)	85
Box-Behnken Design	87
Custom	87
Custom + Sampling	88
Sparse Grid Initialization	88
Latin Hypercube Sampling Design	88
External Design of Experiments	89
Number of Input Parameters for DOE Types	90
Comparison of LHS and OSF DOE Types	93
Using a Central Composite Design DOE	94
Upper and Lower Locations of DOE Points	97
DOE Matrix Generation	97
Exporting and Importing Design Points	98

Exporting Design Points to a CSV File	98
Importing Design Points from a CSV File	98
Copying Design Points	99
Using Response Surfaces	101
Response Surface Types	101
Genetic Aggregation	102
Genetic Aggregation Properties	103
Genetic Aggregation Tolerances Table	104
Tolerances Table Properties	105
Selecting and Removing Output Parameters for Auto-Refinement	106
Genetic Aggregation Convergence Curves Chart	107
Full 2nd-Order Polynomials	108
Kriging	111
Using Kriging Auto-Refinement	111
Kriging Properties	114
Kriging Auto-Refinement Output Parameter Properties	115
Kriging Convergence Curves Chart	115
Non-Parametric Regression	116
Neural Network	116
Sparse Grid	118
Using Sparse Grid Auto-Refinement	119
Sparse Grid Auto-Refinement Properties	121
Sparse Grid Convergence Curves Chart	121
Response Surface Refinement	122
Working with Response Surfaces	123
Changing the Response Surface	124
Refinement Points	126
Performing a Manual Refinement	126
Min-Max Search	127
Quality Metrics for Response Surfaces	130
Goodness of Fit for Output Parameters in a Response Surface	130
Goodness of Fit Criteria	131
Reviewing Goodness of Fit Information	134
Goodness of Fit Display Properties	134
Goodness of Fit Results	135
Goodness of Fit Table	135
Advanced Goodness of Fit Report	136
Goodness of Fit Calculations	139
Verification Points	140
Creating Verification Points	141
Verification Points Results	141
Predicted vs. Observed Chart	142
Response Surface Charts	143
Using the Response Chart	145
Understanding the Response Chart Display	145
Using the 2D Response Chart	148
Using the 3D Response Chart	149
Using the 2D Slices Response Chart	151
Response Chart: Example	154
Response Chart: Properties	156
Using the Spider Chart	157

Using Local Sensitivity Charts	157
Understanding the Local Sensitivities Display	158
Using the Local Sensitivity Chart	160
Local Sensitivity Chart: Example	161
Local Sensitivity Chart: Properties	163
Using the Local Sensitivity Curves Chart	164
Local Sensitivity Curves Chart: Example	166
Local Sensitivity Curves Chart: Properties	169
Exporting Response Surfaces	171
Exporting a Response Surface as a DX-ROM File	171
Tools for Importing a DX-ROM Response Surface into Workbench	172
Downloading the Response Surface Readers App	172
Using Goal-Driven Optimizations	175
Creating a Goal-Driven Optimization System	176
Transferring Design Point Data for Direct Optimization	179
Goal-Driven Optimization Methods	180
Performing a Screening Optimization	181
Performing a MOGA Optimization	183
Performing an NLPQL Optimization	187
Performing an MISQP Optimization	190
Performing an Adaptive Single-Objective Optimization	193
Performing an Adaptive Multiple-Objective Optimization	196
Performing an Optimization with an External Optimizer	200
Defining the Optimization Domain	201
Defining Input Parameters	201
Defining Parameter Relationships	202
Defining Optimization Objectives and Constraints	205
Working with Candidate Points	212
Viewing and Editing Candidate Points in the Table Pane	212
Retrieving Intermediate Candidate Points	214
Inserting Candidate Points as New Design, Response, Refinement, or Verification Points	214
Verifying Candidates by Design Point Update	215
Goal-Driven Optimization Charts and Results	216
Using the Convergence Criteria Chart	217
Using the Convergence Criteria Chart for Multiple-Objective Optimization	217
Using the Convergence Criteria Chart for Single-Objective Optimization	218
Using the History Chart	219
Working with the History Chart in the Chart Pane	220
Viewing History Chart Sparklines in the Outline Pane	224
Using the History Chart for an Objective or Constraint	225
Using the Input Parameter History Chart	226
Using the Parameter Relationship History Chart	227
Using Candidate Point Results	228
Understanding the Display of Candidate Point Results	228
Candidate Point Results: Properties	229
Using the Sensitivities Chart (GDO)	231
Using the Tradeoff Chart	231
Using the Samples Chart	233
Using ROMs	237
ROM Overview	237
ROM Workflow	238

ROM Production	238
ROM Consumption	240
ROM Production Example for Fluent	240
Heat Exchanger Model	240
Prerequisites	242
Producing the ROM	243
Setting Up ROM Building in Fluent	243
Creating the ROM in the ROM Builder	246
Exporting the ROM	251
Consuming an FMU 2.0 File in Twin Builder	251
Twin Builder Setup	252
Live Updating of the 3D Scene	254
Disabling and Enabling Automatic Updates	256
Replaying the Parametric Trajectory	257
Configuring ROM Results	258
Configuring Cutting Planes	258
Configuring Isosurfaces	259
Configuring Isovolumes	260
Configuring Trace Particles	261
Configuring Scalar Settings	262
Configuring the Minimum Display Time for a Set of Results	263
Consuming a ROMZ File in the ROM Viewer	263
Consuming a ROMZ File in Fluent	264
Analyzing and Troubleshooting ROM Production	264
ROM Snapshot Files	265
Viewing Statuses of ROM Snapshot Files	265
Displaying ROM Snapshot File Information	266
Setting Specific ROM Snapshot Files for Design Points	267
Manually Adding Design Points and Setting Snapshot Files	268
Exporting and Importing ROM Snapshot Archive Files	268
Importing Design Points and Snapshot File Settings from CSV Files	269
Clearing Unused ROM Snapshot Files	269
ROM Mesh Files	270
ROM Builder Log File	270
ROM Builder Log File Metrics	271
Quality Metrics for ROMs	276
Goodness of Fit for ROMs	277
ROM Goodness of Fit Criteria	279
Verification Points for ROMs	279
Refinement Points for ROMs	280
ROM Limitations and Known Issues	280
ROM General Limitations	280
ROM Viewer Limitations	281
Using Six Sigma Analysis	283
Performing a Six Sigma Analysis	283
Adding a Six Sigma Analysis System	283
Defining Uncertainty Variables	283
Distribution Histogram Example	284
Consistency Validations of the Distribution Definition	284
Solving the Six Sigma Analysis System	286
Using Statistical Postprocessing	286

Tables (SSA)	286
Using Parameter Charts (SSA)	287
Using the Sensitivities Chart (SSA)	287
Statistical Measures	287
Working with DesignXplorer	289
Working with Parameters	289
Input Parameters	292
Defining Discrete Input Parameters	292
Defining Continuous Input Parameters	294
Defining a Grid Interval	296
Defining Levels for Manufacturable Values	297
Changing Input Parameters	298
Design Variables and Uncertainty Variables	299
Output Parameters	299
Working with Design Points	300
Specifying Design Point Update Options	301
Preserving Generated Design Points to the Parameter Set	302
Retaining Data for Generated Design Points	302
Inserting Design Points	303
Viewing Design Point Images in Tables and Charts	304
Cache of Design Point Results	306
Raw Optimization Data	306
Design Point Log Files	307
Failed Design Points	308
Preventing Design Point Update Failures	309
Preserving Design Points and Retaining Data	311
Handling Failed Design Points	311
Working with Sensitivities	313
Working with Tables	314
Viewing Points in the Table Pane	314
Editable Output Parameter Values	314
Copying and Pasting Data	315
Exporting Table Data	316
Importing Data from a CSV File	316
Working with Remote Solve Manager and DesignXplorer	317
Working with Design Point Service and DesignXplorer	319
Sending an Optimization Study to DPS	319
Importing DPS Design Points into DesignXplorer	320
Working with DesignXplorer Extensions	320
Locating and Downloading Available Extensions	320
Installing a DesignXplorer Extension	321
Loading a DesignXplorer Extension	321
Selecting an External Optimizer or DOE	321
Working with Design Exploration Results in Workbench Project Reports	322
DesignXplorer Theory	325
Parameters Correlation Filtering Theory	325
Relevance of the Correlation Value	325
R ² Contribution	327
Maximum Number of Major Inputs	329
Response Surface Theory	329
Design of Experiments Types	330

Central Composite Design (CCD)	330
Box-Behnken Design	332
Response Surface Types	333
Genetic Aggregation	333
Standard Response Surface - Full 2nd-Order Polynomials	339
Kriging	340
Non-Parametric Regression	344
Sparse Grid	347
Goal-Driven Optimization Theory	350
GDO Principles	351
GDO Guidelines and Best Practices	353
Convergence Rate % and Initial Finite Difference Delta % in NLPQL and MISQP	353
Objectives vs. Constraint Satisfaction in Goal-Driven Optimization	355
Goal-Driven Optimization Theory	357
Sampling for Constrained Design Spaces	357
Shifted Hammersley Sampling (Screening)	360
Single-Objective Optimization Methods	361
Nonlinear Programming by Quadratic Lagrangian (NLPQL)	361
Mixed-Integer Sequential Quadratic Programming (MISQP)	367
Adaptive Single-Objective Optimization (ASO)	368
Multiple-Objective Optimization Methods	372
Pareto Dominance in Multi-Objective Optimization	372
Convergence Criteria in MOGA-Based Multi-Objective Optimization	373
Multi-Objective Genetic Algorithm (MOGA)	374
Adaptive Multiple-Objective Optimization	379
Decision Support Process	381
Six Sigma Analysis (SSA) Theory	385
SSA Principles	386
Guidelines for Selecting SSA Variables	388
Uncertainty Variables for Response Surface Analyses	388
Choosing a Distribution for a Random Variable	388
Measured Data	389
Mean Values, Standard Deviation, Exceedance Values	389
No Data	390
Distribution Functions	391
Sample Generation	395
Weighted Latin Hypercube Sampling (WLHS)	395
Postprocessing SSA Results	395
Histogram	396
Cumulative Distribution Function	396
Probability Table	397
Statistical Sensitivities in a SSA	398
SSA Theory	400
Theory References	405
Genetic Aggregation Response Surface References	405
MISQP Optimization Algorithm References	406
Reference Books	406
Reference Articles	407
Troubleshooting	417



List of Tables

1. Different Types of Kurtosis 401

Ansys DesignXplorer Overview

The following links provide quick access to information about Ansys DesignXplorer and its use:

- [Limitations \(p. 22\)](#)
- [What is Design Exploration? \(p. 41\)](#)
- [User Interface \(p. 23\)](#)
- [Parameters \(p. 26\)](#)
- [Design Points \(p. 28\)](#)
- [Response Points \(p. 28\)](#)
- [Workflow \(p. 29\)](#)
- [Design Exploration Options \(p. 34\)](#)
- [DesignXplorer Systems and Components \(p. 41\)](#)

The Ansys Product Improvement Program

This product is covered by the Ansys Product Improvement Program, which enables Ansys, Inc., to collect and analyze *anonymous* usage data reported by our software without affecting your work or product performance. Analyzing product usage data helps us to understand customer usage trends and patterns, interests, and quality or performance issues. The data enable us to develop or enhance product features that better address your needs.

How to Participate

The program is voluntary. To participate, select **Yes** when the Product Improvement Program dialog appears. Only then will collection of data for this product begin.

How the Program Works

After you agree to participate, the product collects anonymous usage data during each session. When you end the session, the collected data is sent to a secure server accessible only to authorized Ansys employees. After Ansys receives the data, various statistical measures such as distributions, counts, means, medians, modes, etc., are used to understand and analyze the data.

Data We Collect

The data we collect under the Ansys Product Improvement Program are limited. The types and amounts of collected data vary from product to product. Typically, the data fall into the categories listed here:

Hardware: Information about the hardware on which the product is running, such as the:

- brand and type of CPU
- number of processors available
- amount of memory available
- brand and type of graphics card

System: Configuration information about the system the product is running on, such as the:

- operating system and version
- country code
- time zone
- language used
- values of environment variables used by the product

Session: Characteristics of the session, such as the:

- interactive or batch setting
- time duration
- total CPU time used
- product license and license settings being used
- product version and build identifiers
- command line options used
- number of processors used
- amount of memory used
- errors and warnings issued

Session Actions: Counts of certain user actions during a session, such as the number of:

- project saves
- restarts
- meshing, solving, postprocessing, etc., actions

- times the Help system is used
- times wizards are used
- toolbar selections

Model: Statistics of the model used in the simulation, such as the:

- number and types of entities used, such as nodes, elements, cells, surfaces, primitives, etc.
- number of material types, loading types, boundary conditions, species, etc.
- number and types of coordinate systems used
- system of units used
- dimensionality (1-D, 2-D, 3-D)

Analysis: Characteristics of the analysis, such as the:

- physics types used
- linear and nonlinear behaviors
- time and frequency domains (static, steady-state, transient, modal, harmonic, etc.)
- analysis options used

Solution: Characteristics of the solution performed, including:

- the choice of solvers and solver options
- the solution controls used, such as convergence criteria, precision settings, and tuning options
- solver statistics such as the number of equations, number of load steps, number of design points, etc.

Specialty: Special options or features used, such as:

- user-provided plug-ins and routines
- coupling of analyses with other Ansys products

Data We Do Not Collect

The Product Improvement Program does *not* collect any information that can identify you personally, your company, or your intellectual property. This includes, but is not limited to:

- names, addresses, or usernames
- file names, part names, or other user-supplied labels
- geometry- or design-specific inputs, such as coordinate values or locations, thicknesses, or other dimensional values

- actual values of material properties, loadings, or any other real-valued user-supplied data

In addition to collecting only anonymous data, we make no record of where we collect data from. We therefore cannot associate collected data with any specific customer, company, or location.

Opting Out of the Program

You may *stop* your participation in the program any time you wish. To do so, select **Ansys Product Improvement Program** from the Help menu. A dialog appears and asks if you want to continue participating in the program. Select **No** and then click **OK**. Data will no longer be collected or sent.

The Ansys, Inc., Privacy Policy

All Ansys products are covered by the Ansys, Inc., [Privacy Policy](#).

Frequently Asked Questions

1. *Am I required to participate in this program?*

No, your participation is voluntary. We encourage you to participate, however, as it helps us create products that will better meet your future needs.

2. *Am I automatically enrolled in this program?*

No. You are not enrolled unless you explicitly agree to participate.

3. *Does participating in this program put my intellectual property at risk of being collected or discovered by Ansys?*

No. We do not collect any project-specific, company-specific, or model-specific information.

4. *Can I stop participating even after I agree to participate?*

Yes, you can stop participating at any time. To do so, select **Ansys Product Improvement Program** from the Help menu. A dialog appears and asks if you want to continue participating in the program. Select **No** and then click **OK**. Data will no longer be collected or sent.

5. *Will participation in the program slow the performance of the product?*

No, the data collection does not affect the product performance in any significant way. The amount of data collected is very small.

6. *How frequently is data collected and sent to Ansys servers?*

The data is collected during each use session of the product. The collected data is sent to a secure server once per session, when you exit the product.

7. *Is this program available in all Ansys products?*

Not at this time, although we are adding it to more of our products at each release. The program is available in a product only if this *Ansys Product Improvement Program* description appears in the product documentation, as it does here for this product.

8. *If I enroll in the program for this product, am I automatically enrolled in the program for the other Ansys products I use on the same machine?*

Yes. Your enrollment choice applies to all Ansys products you use on the same machine. Similarly, if you end your enrollment in the program for one product, you end your enrollment for all Ansys products on that machine.

9. *How is enrollment in the Product Improvement Program determined if I use Ansys products in a cluster?*

In a cluster configuration, the Product Improvement Program enrollment is determined by the host machine setting.

10. *Can I easily opt out of the Product Improvement Program for all clients in my network installation?*

Yes. Perform the following steps on the file server:

- a. Navigate to the installation directory: [Drive:]\v212\commonfiles\globalsettings
- b. Open the file **ANSYSProductImprovementProgram.txt**.
- c. Change the value from "on" to "off" and save the file.

Introduction to Ansys DesignXplorer

A good design point is often the result of a trade-off between various objectives. Consequently, during design exploration, you do not want to use optimization algorithms that lead to a single design point. You want to gather enough information about the current design to be able to answer "What-if" questions that quantify the influence of design variables on product performance. By doing so, you can make the right decisions based on accurate information, even in the event of an unexpected change in the design constraints.

Design exploration describes the relationship between the design variables and the performance of the product using Design of Experiments (DOEs) and response surfaces. DOEs and response surfaces provide all of the information required to achieve simulation-driven product development. Once the variation of product performance with respect to design variables is known, it becomes easy to understand and identify the changes required to meet requirements for the product. After response surfaces are created, you can analyze and share results using curves, surfaces, and sensitivities that are easily understood. You can use these results at any time during the development of the product without requiring additional simulations to test a new configuration.

Available Tools

DesignXplorer offers a powerful suite of DOE types:

- [Central Composite Design \(CCD\) \(p. 84\)](#)
- [Optimal Space-Filling Design \(OSF\) \(p. 85\)](#)
- [Box-Behnken Design \(p. 87\)](#)
- [Custom \(p. 87\)](#)

- [Custom + Sampling \(p. 88\)](#)
- [Sparse Grid Initialization \(p. 88\)](#)
- [Latin Hypercube Sampling Design \(p. 88\)](#)

Central Composite Design (CCD) provides a traditional DOE sampling set, while the objective of Optimal Space-Filling (OSF) is to gain the maximum insight with the fewest number of points. OSF is very useful when you have limited computation time.

After sampling, design exploration provides several different response surface types to represent the simulation's responses:

- [Genetic Aggregation \(p. 102\)](#)
- [Full 2nd-Order Polynomials \(p. 108\)](#)
- [Kriging \(p. 111\)](#)
- [Non-Parametric Regression \(p. 116\)](#)
- [Neural Network \(p. 116\)](#)
- [Sparse Grid \(p. 118\)](#)

These response surface types can accurately represent highly nonlinear responses, such as those found in high frequency electromagnetics.

Once the simulation's responses are characterized, DesignXplorer supplies the following optimization algorithms:

- [Shifted Hammersley Sampling \(Screening\) \(p. 360\)](#)
- [Multi-Objective Genetic Algorithm \(MOGA\) \(p. 374\)](#)
- [Nonlinear Programming by Quadratic Lagrangian \(NLPQL\) \(p. 361\)](#)
- [Mixed-Integer Sequential Quadratic Programming \(MISQP\) \(p. 367\)](#)
- [Adaptive Single-Objective Optimization \(ASO\) \(p. 368\)](#)
- [Adaptive Multiple-Objective Optimization \(p. 379\)](#)

You can also use extensions to integrate external optimizers into the DesignXplorer workflow. For more information, see [Performing an Optimization with an External Optimizer \(p. 200\)](#).

DesignXplorer provides several graphical tools for investigating a design. These tools include sensitivity plots, correlation matrices, curves, surfaces, trade-off plots and parallel charts with Pareto front display, and spider charts.

DesignXplorer also provides correlation matrix techniques to help you identify the key parameters of a design before you create response surfaces.

Additionally, from a series of 2D or 3D simulations, you can create a ROM (reduced order model). ROMs are stand-alone digital objects that offer a mathematical representation for computationally inexpensive, near real-time analysis. For more information, see [Using ROMs \(p. 237\)](#).

Design Exploration: What to Look for in a Typical Session

The main purpose of design exploration is to identify the relationship between the performance of the product and the design variables.

- Product performance includes maximum stress, mass, fluid flow, and velocities.
- Design variables include dimensions, loads, and material properties.

Based on exploration results, you can identify the key parameters of the design and how they affect product performance. You can then use this knowledge to influence the design so that it meets the product's requirements.

DesignXplorer provide tools to analyze a parametric design with a reasonable number of parameters. Supported response surface methods are suitable for problems using 10 to 15 input parameters.

Initial Simulation: Analysis of the Product Under all Operating Conditions

The first step of any design simulation is to create the simulation model. The simulation model can use a single physics or use multiple physics with many complex conditions and physics coupling.

In addition to performing the standard simulation, you must define the parameters to investigate. The input parameters, also called design variables, can include CAD parameters, loading conditions, material properties, and more.

You choose the output parameters, also called performance indicators, from the simulation results. Output parameters can include maximum stresses, fluid pressure, velocities, temperatures, masses, and custom-defined results. For example, product cost could be a custom-defined result based on masses and manufacturing constraints that you use as an output parameter.

CAD parameters can be defined in a CAD package or in Ansys DesignModeler. In Workbench, material properties are defined in the **Engineering Data** cell of an analysis system that you've inserted in the **Project Schematic**. The origin of other parameters are in the simulation model itself. Output parameters are defined in the various simulation environments (Mechanical, CFD, and so on). Custom parameters are defined in the **Parameter Set** bar.

Identify Design Candidates

Once you create the initial model and define parameters, your next step is to create a response surface. After you insert a **Response Surface** system in the **Project Schematic**, you define the design space by specifying the minimum and maximum values to consider for each input variable. Based on the information that you enter in the **Design of Experiments** (DOE) cell, the **Response Surface** system creates the design space sampling. The sampling depends on the selected DOE type and that the default CCD design type provides good accuracy for the final approximation, which is typically the case. Next, the DOE must be computed.

When you update the DOE, DesignXplorer creates a response surface for each output parameter. A response surface is an approximation of the response of the system. Its accuracy depends on several

factors, including complexity of the variations of the output parameters, number of points in the original DOE, and choice of the response surface type.

DesignXplorer provides a variety of response surface types. The default type, Genetic Aggregation, automates the process of selecting, configuring, and generating the response surface best suited to each output parameter in your problem. Several other response surface types are available for selection. For instance, Standard Response Surface - Full 2nd Order Polynomials, which is based on a modified quadratic formulation, provides satisfying results when the variations of the output parameters are slight. However, Kriging is more effective for problems with a broad range of variation.

After response surfaces are created, you can thoroughly investigate the design using a variety of graphical and numerical tools. Additionally, you can use optimization techniques to identify valid design points.

Usually, the investigation starts with viewing sensitivity graphs because they graphically display the relative influence of the input parameters. These bar or pie charts indicate how much output parameters are locally influenced by the input parameters around a given response point. Varying the location of the response point can provide a totally different graph. Explanations of these graphs typically use a hill and valley analogy. If the point is at the top of a steep hill, the influence of the parameters is large. If the response point is in a flat valley, the influence of the input parameters is small.

The response surfaces provide curves or surfaces that show the variation of one output parameter with respect to one or two input parameters at a time. These curves or surfaces also are dependent on the response point.

Both sensitivity charts and response surfaces are key tools for answering such what-if questions as "What parameter should we change if we want to reduce cost?"

DesignXplorer provides additional tools for identifying design candidates. In addition to thoroughly investigating response surface curves to determine design candidates, you can use optimization techniques to find design candidates from a **Response Surface** cell or any cell containing design points. DesignXplorer provides two types of goal-driven optimization (GDO) systems: **Response Surface Optimization** and **Direct Optimization**.

You can drag a GDO system from the **Toolbox** and drop it in the **Project Schematic**. If you drop a **Response Surface Optimization** system on an existing **Response Surface** system, these two systems share this portion of the data. For a **Direct Optimization** system, you can create data transfer links between the **Optimization** cell and any other cell containing design points. You can insert several GDO systems in the project, which is useful if you want to analyze several hypotheses.

Once a GDO system is inserted, you must define the optimization study. This includes choosing the optimization method, setting the objectives and constraints, and specifying the domain. You then solve the optimization problem. In many cases, there is not a unique solution, and several design candidates are identified.

The results of the optimization are also very likely to provide design candidates that cannot be manufactured. For example, a radius of 3.14523 mm is likely difficult to achieve. However, because all information about the variability of the output parameters is provided by the source of design point data, whether a **Response Surface** cell or another DesignXplorer cell, you can easily find an acceptable design candidate close to the one indicated by the optimization.

You should check the accuracy of the response surface for the design candidates. To verify a candidate, you update the design point, which checks the validity of the output parameters.

Assess the Robustness of the Design Candidates

Once design point candidates are identified, probabilistic analysis can be used to quantify the reliability or quality of the product statistically. Probabilistic analysis typically involves four areas of statistical variability: geometric shape, material properties, loading, and boundary conditions. For example, the statistical variability of the geometry of a product tries to capture product-to-product differences due to manufacturing imperfections quantified by manufacturing tolerances.

Probabilistic characterization provides a probability of success or failure rather than a simple yes or no evaluation. For instance, a probabilistic analysis could determine that one part in 1 million is likely to fail. Probabilistic analysis can also predict the probability of a product surviving its expected useful life.

To perform a robustness analysis, you insert a **Six Sigma Analysis** system in the **Project Schematic**. The process here is very similar to the process for response surfaces. The main difference is in the definition of the parameters. Instead of specifying a minimum and maximum value to define the range for each input parameter, you describe an input parameter in terms of a statistical curve and the curve's associated parameters. Once parameters are defined, you compute a new DOE and corresponding response surfaces and sensitivity charts. The results for the robustness analysis are in the form of probabilities and statistical distributions of the output parameters.

Determine the Number of Parameters

To obtain accurate response surfaces within a reasonable amount of time, you should limit the number of input parameters to 10 to 15. If you need to investigate more parameters, you can use a correlation matrix to identify key parameters before creating the response surface. By inserting a **Parameters Correlation** system before a **Response Surface** system, you can perform simulations based on a random sampling of the design space to identify the correlation between all parameters.

The number of simulations depend on the number of parameters as well as the convergence criteria for the means and standard deviations of the parameters. While you can provide a hard limit for the number of points to compute, the accuracy of the correlation matrix can be affected if not enough points are computed. Based on results, you reduce the number of parameters to the 10 to 15 with the highest correlations.

Increase the Response Surface Accuracy

A richer set of points for the DOE usually provides a more accurate response surface. However, when you select a response surface type other than Genetic Aggregation, which is the default, you generally need to specify how many additional points are required, which can be difficult without actually generating the response surface.

The first step to solving this problem is to set **Response Surface Type** to **Kriging**. This response surface type determines the accuracy of the response surface as well as the number of points that are required to increase the accuracy. With this method, you can set a manual or automatic refinement type. Manual refinement allows you to control the number of points to compute.

Sparse Grid is yet another of the response surface types available. This adaptive response surface is driven by the accuracy that you request. It automatically refines the matrix of design points where the gradient of the output parameters is higher to increase the accuracy of the response surface. To use this feature, you set **Response Surface Type** to **Sparse Grid** and **Design of Experiments Type** to **Sparse Grid Initialization**.

While no manual refinement is available when **Sparse Grid** is selected, manual refinement is available for other response surface types. With manual refinement, you can enter specific points into the set of existing design points used to calculate the response surface.

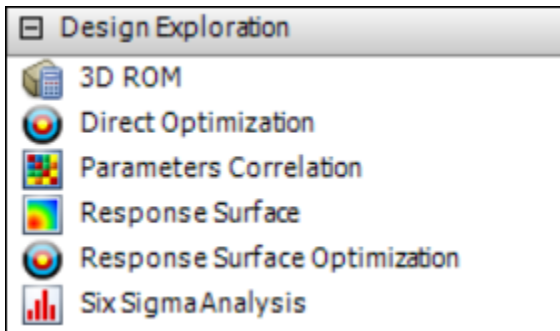
For more information, see [Response Surface Types \(p. 101\)](#).

Limitations

- Material parameters, which are input parameters linked to geometrical bodies, are not supported by DesignXplorer. A material parameter has a finite list of material names.
- Suppressed properties are not available for a **Design of Experiments** system.
- When you use an external optimizer for a goal-driven optimization system, unsupported properties and functionality are not displayed in the DesignXplorer interface.

DesignXplorer Licensing Requirements

The Workbench **Toolbox** contains the following **Design Exploration** systems:



To insert a design exploration system, you must have a DesignXplorer license. This license must be available when you preview or update a DesignXplorer system or cell and also when results need to be generated from a response surface.

You must also have licenses available for the systems that are to solve the design points that DesignXplorer generates. The licenses for the solvers must be available when you update a system that generate design points.

The DesignXplorer license is released when all DesignXplorer tabs are closed.

Note:

- If you do not have a DesignXplorer license, you can successfully resume an existing design exploration project and review the already generated results, with some interaction possible on charts. However, to update a DesignXplorer system or evaluate a response surface, a license is required. If you do not have a license, an error displays.
 - To update design points simultaneously, you must have one solver license for each simultaneous solve. Be aware that the number of design points that can be solved simultaneously is limited by hardware, your RSM configuration, and available solver licenses.
 - You do not need to reserve licenses for DesignXplorer components because DesignXplorer does not check licenses out of the reserve pool. However, if some licenses are reserved for design point updates, any update of a DesignXplorer component will require an extra license. This license can come from a bundle.
 - Inserting a **3D ROM** system for producing a [ROM \(p. 237\)](#) requires a **ROM Builder** license. A **ROM Builder** license also enables existing DesignXplorer capabilities.
-

User Interface

The Workbench user interface allows you to easily build your project in a workspace called the **Project Schematic**.

Project Schematic

From the **Project** tab, you add design exploration systems to the **Project Schematic** to perform different types of parametric analyses. From the **Toolbox**, you drag a system from under **Design Exploration** and drop it under the **Parameter Set** bar. Optionally, you can double-click the system in the **Toolbox**.

Each system added to the **Project Schematic** has a blue system header and one or more cells for analysis components. You generally interact with a system at the cell level. Right-clicking a system header or cell displays context menu options. Double-clicking a cell performs the default option, which appears in bold in the context menu. Because the default option is typically **Edit**, double-clicking a cell generally opens its component tab. Component tabs are described in more detail later.

Parameter Set Bar

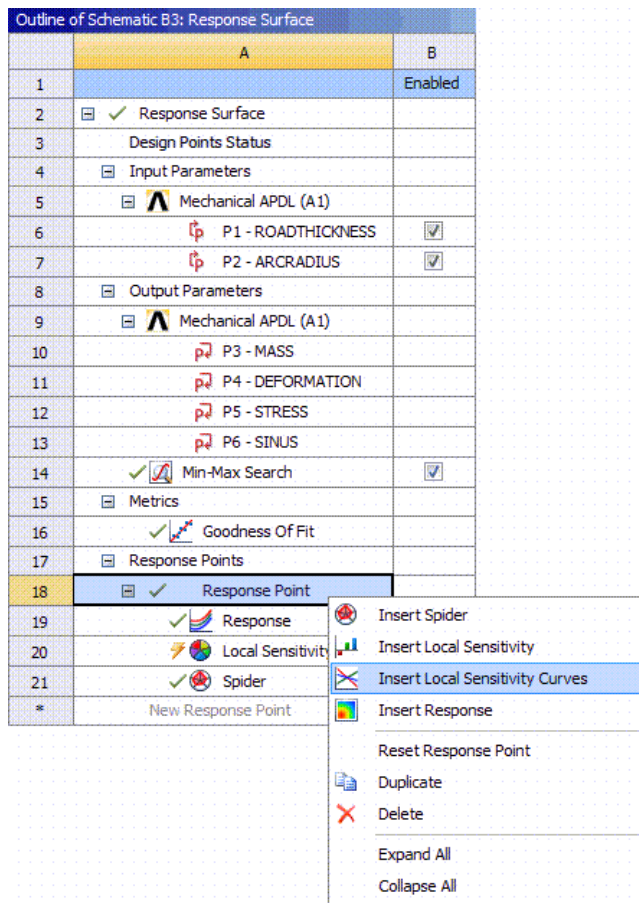
Once your systems and models are created, double-clicking the **Parameter Set** bar allows you to choose which parameters to expose to DesignXplorer. After all systems and parameters are defined, you add the design exploration systems that you need for your parametric analyses.

Toolbox

When you are viewing the **Project Schematic**, the **Toolbox** displays a **Design Exploration** category with DesignXplorer systems. To perform a particular type of design exploration, you drag the system from the **Toolbox** and drop it in the **Project Schematic** below the **Parameter Set** bar.

When you are viewing the component tab for a cell, the **Toolbox** displays the charts that can be added to the object currently selected in the **Outline** pane. For example, assume that you double-clicked a **Response Surface** cell to open its component tab. If you select a response point in the **Outline** pane, the **Toolbox** displays the charts that can be inserted for the response point.

Double-clicking a chart in the **Toolbox** adds it to the object selected in the **Outline** pane. You can also add a chart by dragging it from the **Toolbox** and dropping it on an object in the **Outline** pane. Additionally, you can right-click a node in the **Outline** pane and select the chart to add from the context menu.



Component Tabs

Once a DesignXplorer system is added to the **Project Schematic**, double-clicking a cell typically opens its component tab. For a cell where **Edit** is not the default menu option, you can right-click the cell and select **Edit**. A component tab displays a window configuration with multiple panes in which to set analysis options, run the analysis, and view results. For example, double-clicking a **Design of Experiments** cell displays the component tab for the DOE.


The screenshot displays the ANSYS Workbench interface for a Design of Experiments (DOE) setup. It is divided into four main panes:


- Outline:** A hierarchical tree view on the left showing the project structure. The root node is 'Design of Experiments', which includes 'Input Parameters' (P1-x_1 to P5-x_5), 'Output Parameters' (P6-f, P7-g), and 'Charts'. The 'Design Points vs Parameter' node is selected.
- Table:** A data table titled 'Table of Schematic B2: Design of Experiments (Central Composite Design : Auto Defined)'. It lists 27 design points (DP 0 to DP 20) with their corresponding parameter values for P1-x_1 through P7-g.
- Properties:** A pane titled 'Properties of Outline : Design Points vs Parameter' showing settings for the selected chart, such as 'Display Parameter Full Name' and 'Axes' (X and Y axes).
- Design Points vs Parameter:** A line chart showing the relationship between design points (X-axis) and a specific parameter (Y-axis, P6-f). The chart shows a sharp peak at design point 6 and a sharp dip at design point 8.

A component tab has four panes: **Outline**, **Table**, **Properties**, and either **Chart** or **Results**. In general, you select an object in the **Outline** pane and either set up its properties or view the table or chart associated with it.

- **Outline:** Provides a hierarchy of the main objects that make up the cell that you are editing.

The state icon on the root node tells you if the data for the cell is up-to-date or if it must be updated. It also helps you to figure out the effect of your changes on the cell and its parameter properties.

Quick help is associated with various states. If you see the information icon  to the right of the root node, click it to see what immediate actions must be taken. If links appear in the quick help, they take you to more information in the Ansys product help.

On nodes for result objects (such as response points, charts, and Min-Max search), state icons indicate if the objects are up-to-date or if they need to be updated. State icons help you to quickly assess the current status of your result objects. For example, if you change a DOE setting, the state icon of the corresponding chart is updated, given the pending changes. When the state icon indicates that the update on a result object has failed () , you can try to update the object by right-clicking it and selecting the appropriate menu option.

- **Table:** A tabular view of the data associated with the object selected in the **Outline** pane. The title bar contains a description of the table. You can right-click in the table to export the data to a CSV (Comma-Separated Values) file. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) in the *Workbench User's Guide*.
- **Properties:** Lists the properties that can be set for the object selected in the **Outline** pane. For example, when a parameter for a DOE is selected in the **Outline** pane, you set bounds for this parameter in the **Properties** pane. When the root node for a **Response Surface** cell is selected, you set the re-

sponse surface type and other options in the **Properties** pane. When a chart is selected, you set plotting and display options in the **Properties** pane.

- **Chart** or **Results**: Displays various charts or results for the object selected in the **Outline** pane. In the **Charts** pane, you can right-click a chart to export the data to a CSV (Comma-Separated Values) file. The **Results** pane is shown only on the component tab for a goal-driven optimization system.

You can insert and duplicate charts (or a response point with charts for a **Response Surface** cell) even if the system is out-of-date. When the system is out-of-date, the charts in the **Chart** pane are updated when the system is updated. For any DesignXplorer cell where a chart is inserted before the system is updated, all types of charts supported by the cell are inserted by default at the end of the update. If a cell already contains a chart, no new chart is inserted by default. For a **Response Surface** cell, if there is no response point, a response point is inserted by default with all charts. For more information, see [Using DesignXplorer Charts \(p. 32\)](#).

Note:

- In the **Table** and **Properties** panes, input parameter values and output parameter values obtained from a simulation are displayed in black text. Output parameters based on a response surface are displayed in the color specified on the **Response Surface** tab in the **Options** dialog box. For more information, see [Response Surface Options \(p. 38\)](#).
- In the **Properties** pane, the color convention for output parameter values is not applied to the **Calculated Minimum** and **Calculated Maximum** values. These values always display in black text.

Context Menu

Right-clicking a DesignXplorer system header or cell in the **Project Schematic** displays a context menu. Likewise, right-clicking a node in the **Outline** pane of a component tab displays a context menu. The options available depend on the state of the system or cell. The options typically available are **Update**, **Preview**, **Clear Generated Data**, and **Refresh**. The option selected is performed only on the selected system or cell.

Parameters

DesignXplorer makes use of two types of parameters:

- Input parameters
- Output parameters

For information about performing what-if studies to investigate design alternatives, see [Working with Parameters and Design Points](#) in the *Workbench User's Guide*. For information about grouping parameters by application, see [Tree Usage in Parameters Tab and Parameter Set Tabs \(p. 27\)](#).

Input Parameters

Input parameters define the values to analyze for the model under investigation. Input parameters include CAD parameters, analysis parameters, DesignModeler parameters, and mesh parameters.

- Examples of CAD and DesignModeler input parameters are length and radius.
- Examples of analysis input parameters are pressure, material properties, materials, and sheet thickness.
- Examples of mesh parameters are relevance, number of prism layers, and mesh size on an entity.

Input parameters can be discrete or continuous. Each of these parameter types has a specific form.

Input Parameter Type	Description	Examples
Discrete	Input parameter values are specified integers	Number of holes, number of weld points, and number of prism layers
Continuous	Input parameter values fall within a non-interrupted range of values	Thickness, force, and temperature

Discrete parameters physically represent different configurations or states of the model. An example is the number of holes in a geometry. Discrete parameters allow you to analyze different design variations in the same parametric study without having to create multiple models for parallel parametric analysis. For more information, see [Defining Discrete Input Parameters \(p. 292\)](#).

Continuous parameters physically vary in a continuous manner between some lower bound and upper bound. Examples are a CAD dimension and load magnitude. Continuous parameters allow you to analyze a continuous value within a defined range, with each parameter representing a direction in the design and treated as a continuous function in **Design of Experiments** and **Response Surface** systems. For a continuous parameter, you can impose additional limitations on the values within this range. For more information, see [Defining Continuous Input Parameters \(p. 294\)](#).

If you disable an input parameter, its initial value, which becomes editable, is used for the design exploration study. If you change the initial value of a disabled input parameter during the study, all depending results are invalidated. A disabled input parameter can have a different initial value in each DesignXplorer system. For more information, see [Changing Input Parameters \(p. 298\)](#).

Output Parameters

Output parameters either result from the geometry or are response outputs from the analysis. Examples include volume, mass, frequency, stress, velocity, pressure, force, heat flux, and so on. Output parameters can include *derived parameters*, which are calculated from output and input parameters using equations that you provide. Derived parameters are created in the system and then passed into DesignXplorer as output parameters.

Tree Usage in Parameters Tab and Parameter Set Tabs

A Workbench project can contain parameters defined in different systems and parameters defined in the **Parameters** tab and **Parameter Set** tab. You can view and edit the parameters of a given system

from the **Parameters** cell at the bottom of the system or from the **Parameter Set** bar. To view the corresponding component tab, double-click it.

When editing the **Parameter Set** tab, all parameters for the project are listed in the **Outline** pane, under **Input Parameters** and **Output Parameters**, depending on their nature.

The parameters are also grouped by system name to reflect the origin of the parameters and the structure of the **Project Schematic** when working in parametric environments. Because parameters can be manipulated from the component tabs for a **Parameters** cell and the **Parameter Set** bar, and also in DesignXplorer tabs, the same tree structure is always used.

Tip:

You can edit the system name by right-clicking the system name in the **Project Schematic**.

Design Points

A design point is defined by a snapshot of parameter values where output parameter values are calculated directly by a project update. Design points are created by design exploration. For instance, they are created when processing a DOE or correlation matrix or when refining a response surface.

It is also possible to insert a design point at the project level from an optimization candidate design to perform a validation update. Output parameter values are not copied to the created design point because they were calculated by design exploration and are, by definition, approximated. Actual output parameters are calculated from the design point input parameters when a project is updated.

You can also edit process settings for design point updates, including the order in which points are updated and the location where the update occurs. When submitting design points for update, you can specify whether the update is to run locally on your machine or sent via RSM for remote processing.

For more information, see [Working with Design Points \(p. 300\)](#).

Response Points

A response point is defined by a snapshot of parameter values where output parameter values are calculated in DesignXplorer from a response surface. As such, the parameter values are approximate and calculated from response surfaces. You should verify the most promising designs by a solve in the system using the same parameter values.

When editing a **Response Surface** cell, you can create new response points from either the **Outline** or **Table** pane. You can also insert response points and design points from the **Table** pane or **Chart** pane by right-clicking a table row or point on the chart and selecting an appropriate option from the context menu. For instance, you can right-click a point in a Response chart and select the option for inserting a new response point in this location.

You can duplicate a response point by right-clicking it in the **Outline** pane and selecting **Duplicate**. You can also duplicate a response point using the drag-and-drop operation. An update of the response

point is attempted so that the duplication of an existing up-to-date response point results in a new up-to-date response point.

Note:

- Duplicating a response point also duplicates all charts attached to it.
 - When you use the context menu to duplicate a chart that is a child of a response point, a new chart is inserted under the same response point. However, when you use the drag-and-drop operation, the duplicate is inserted under the other response point.
-

18	[-] Response Points	
19	[-] ✓ Response Point	
20	[-] ✓ Spider	
21	[-] ✓ Local Sensitivity	
22	[-] ✓ Response	
23	[-] ✓ Response Point 1	
24	[-] ✓ Local Sensitivity 1	
25	[-] ✓ Response 1	
*	New Response Point	

For more information, see [Working with Response Surfaces \(p. 123\)](#).

Workflow

To run a parametric analysis in DesignXplorer, you must:

- Create your model in DesignModeler or a [supported CAD system](#).
- Load the model into one of the systems available in Workbench.
- Select the parameters that you want to use.
- Add the DesignXplorer systems that you want to use.
- Set analysis options and parameter limits.
- Update the analysis.
- View the results of the analysis.
- Generate a Workbench project report with analysis results.

Adding Design Exploration Systems to the Project Schematic

You can add as many DesignXplorer systems as you like and rerun them with different parameters and limits as many times as you need to refine your design. To add a DesignXplorer system to the **Project Schematic**:

1. Create parts and assemblies in either DesignModeler or a [supported CAD system](#).

Features in the geometry that are important to the analysis should be exposed as parameters. These parameters can then be passed to DesignXplorer.

2. Drag a system analysis from the **Toolbox** and drop it in the **Project Schematic**, connecting it to the DesignModeler or CAD file.
3. Double-click the **Parameter Set** bar and do the following for each input parameter:
 - a. In the **Outline** pane, select the input parameter.
 - b. In the **Properties** pane, set the limits for the input parameter.
4. In the **Project Schematic**, drag the DesignXplorer system that you want to insert and drop it below the **Parameter Set** bar.

Duplicating Existing DesignXplorer Systems

You can duplicate any DesignXplorer system in the **Project Schematic**. The manner in which you duplicate a system dictates whether data is shared between the duplicated systems.

- For any DesignXplorer system, right-click the system header and select **Duplicate**. A new system of this type is added to the **Project Schematic** under the **Parameter Set** bar. No data is shared with the original system.
- For a DesignXplorer system with a **Design of Experiments** cell, click this cell and select **Duplicate**. A new system of this type is added to the **Project Schematic** under the **Parameter Set** bar. No data is shared with the original system.
- For a DesignXplorer system with a **Response Surface** cell, click this cell and select **Duplicate**. A new system of this type is added to the **Project Schematic** under the **Parameter Set** bar. The DOE data is shared with the original system.
- For a **Direct Optimization, Response Surface Optimization, ROM Builder, or Six Sigma Analysis** system, click the system header and select **Duplicate**. A new system of this type is added to the **Project Schematic** under the **Parameter Set** bar. The DOE and response surface data is shared with the original system.

Any cell in the duplicated DesignXplorer system that contains data that is not shared with the original system is marked as **Update Required**.

When you duplicate a DesignXplorer system, definitions of your data (such as charts, responses points, and metric objects) are also duplicated. An update is required to calculate the results for the duplicated data.

Running Design Exploration Analyses

After you've added the desired design exploration systems to your **Project Schematic**, you must set up the individual DesignXplorer systems and then run the analysis for each one.

1. Specify design point update options in the **Properties** pane of the **Parameter Set** tab. These options can vary from the global settings specified on the **Solution Process** tab in the **Options** dialog box.
2. For each cell in the design exploration system, double-click it to open the component tab and set up any analysis options that are needed. Options can include parameter limits, optimization objectives or constraints, optimization type, parameter distributions for Six Sigma Analysis, and more.
3. Run the analysis using one of the following methods:
 - From the component tab for the cell, right-click the root node in the **Outline** pane and select **Update**.
 - From the **Project Schematic**, right-click a cell and select **Update**.
4. Make sure that you set up and solve each cell in a DesignXplorer system to complete the analysis for this system.

Tip:

To update all systems in the entire project, either click **Update Project** on the toolbar or right-click in any empty area of the **Project Schematic** and select **Update Project**.

5. View the results for each DesignXplorer system from the component tabs for its cells. Results are in the form of tables, statistics, charts, and so on.



In the **Project Schematic**, cells display icons to indicate their states. If a cell is out-of-date and must be updated, right-click the cell and select **Update**.

Monitoring Design Exploration Analyses

After starting an update of a design exploration analysis, two different panes are available for monitoring your analysis.

Progress Pane

You can open the **Progress** pane from the **View** menu or click **Show Progress** in the lower right corner of the Workbench window. During execution of an update, the **Status** cell displays the component currently being updated. The **Details** cell displays additional information about updating this component. The **Progress** cell displays a progress bar.

Progress			
	A	B	C
1	Status	Details	Progress
2	Updating the Design of Experiments component in Response Surface	Updating the Geometry component in Static Structural for Design Point 18	 

Throughout the execution of the update, this pane continuously reports the progress. To stop the update, you would clicking the red stop button to the right of the progress bar. To restart a stopped update at a later time, you use any of the methods for starting a new update.

Messages Pane

If solution errors exist and the **Messages** pane is not open, in the lower right of the window, the button for showing messages flashes orange. Clicking this button, which indicates the number of messages generated, opens the **Messages** pane so that you can see the solution errors. You can also open the **Messages** pane from the **View** menu.

Reporting on the Design Exploration Analysis

You can generate a Workbench project report to summarize and display the results of your design exploration analysis. This report provides a visual snapshot of the project at a given point in time. The contents and organization reflect the layout of the **Project Schematic**. For more information, see [Working with Design Exploration Results in Workbench Project Reports](#) (p. 322).

Using DesignXplorer Charts

DesignXplorer charts are visual tools that help you to better understand your design exploration analysis. The variety of charts available for a DesignXplorer cell can be viewed in the **Charts** pane of the component tab.

All of the charts available for a cell are listed in the **Toolbox** for the component tab. When you update a cell for the first time, one of each chart available for the cell is automatically inserted in the **Outline** pane. For a **Response Surface** cell, a new response point is also automatically inserted. If a cell already contains charts, the charts are replaced with the next update.

Note:

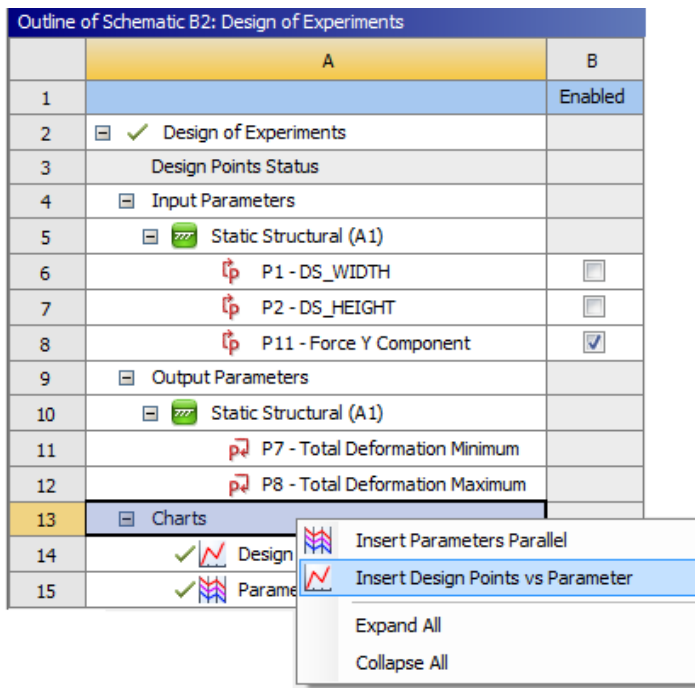
Charts are available for selection in the **Outline** pane. Most charts are created under **Charts**. However, charts for a **Response Surface** cell are an exception. The Predicated vs. Observed chart is inserted under **Quality** → **Goodness of Fit**. Other charts are inserted under respective response points under **Response Point**.

Adding or Duplicating Charts

You can insert and duplicate charts even if the system is out-of-date. If the system is out-of-date, the charts are displayed and updated in the **Chart** pane when the system is updated.

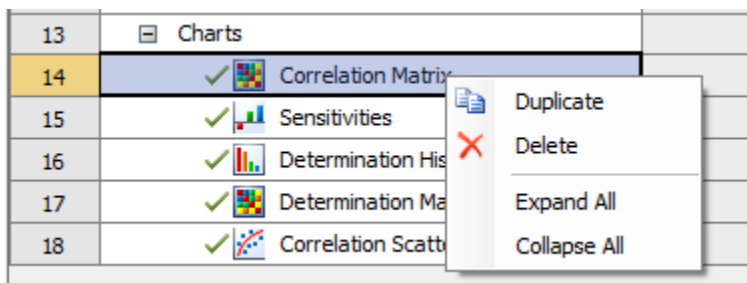
Insert a new chart by either of the following methods:

- Drag a chart from the **Toolbox** and drop it on the **Outline** pane.
- In the **Outline** pane, right-click the cell under which to add the chart and select the option for inserting the desired chart type.



Duplicate a chart using either of the following methods:

- To create a second instance of a chart with default settings or to create a new Response Surface chart under a different response point, drag the desired chart from the **Toolbox** and drop it on the parent cell.
- To create an exact copy of an existing chart, right-click the chart and select **Duplicate**.



For Response Surface charts, the **Duplicate** option on the context menu creates an exact copy of the existing chart under the same response point. To create a fresh instance of a chart type under a different response point, drag the existing chart and drop it on the new response point.

Chart duplication triggers a chart update. If the update succeeds, both the original chart and the duplicate are up-to-date.

Working with Charts

When you select a chart in the **Outline** pane, its properties display in the **Properties** pane, and the chart itself displays it in the **Chart** pane.

In the **Chart** pane, you can drag the mouse over various chart elements to view coordinates and other element details.

You can change chart properties in the **Properties** pane. You can also right-click the directory on the chart to use context-menu options for performing various chart-related operations. The options available depend on the chart type and state of the chart.

- To edit general chart properties, right-click a chart or chart element and select **Edit Properties**. For more information, see [Setting Chart Properties](#) in the *Workbench User's Guide*.
- To add new points to your design, right-click a chart point. Depending on the chart type, you can select from the following context menu options: **Explore Response Surface at Point**, **Insert as Design Point**, **Insert as Refinement Point**, **Insert as Verification Point**, and **Insert as Custom Candidate Point**.
- On charts that allow you to enable or disable parameters, you can:
 - Right-click a chart parameter and, depending on the parameter, select **Disable** <parameterID>, **Disable all Input Parameters but** <parameterID>, or **Disable all Output Parameters but** <parameterID>.
 - If at least one parameter is already disabled, you can right-click anywhere on the chart and select **Reverse Enable/Disable** to enable all disabled parameters or vice versa.

For general information about working with charts, see [Working with the Chart Pane](#) in the *Workbench User's Guide*.

Exporting DesignXplorer Data

DesignXplorer allows you to export design point values, response surfaces, and ROMs.

Exporting Design Point Values

You can export the design point values of a selected DesignXplorer table or chart to an ASCII file, which then can be used by other programs for further processing. For more information, see [Exporting and Importing Design Points \(p. 98\)](#).

Exporting Response Surfaces

Once a response surface is solved, you can export it as an independent reduced-order model (DX-ROM) for reuse in other environments. For more information, see [Exporting Response Surfaces \(p. 171\)](#).

Exporting ROMs

A **3D ROM** system is based on a Design of Experiments (DOE) and its design points, which automate the production of solution snapshots and the ROM itself. Once the ROM is produced, you can export it to a ROMZ file, which can be consumed by anyone who has access to the **ROM Viewer** or Workbench. You can also export the ROM to an FMU file, which can be consumed by anyone who has access to Ansys Twin Builder. For more information, see [ROM Consumption \(p. 240\)](#).

Design Exploration Options

You can set default behavior for design exploration features in the **Options** dialog box. These options specify global default values for the DesignXplorer analyses that you've added to the **Project Schematic**. In the **Properties** pane for a component tab, you can change the values for a specific analysis. Changing

values in the **Properties** pane does not affect the global default values set here in the **Options** dialog box.

To access options for DesignXplorer:

1. Select **Tools** → **Options**. The **Options** dialog box opens.
2. In the tree, expand **Design Exploration** to see its three child tabs:
 - **Design of Experiments**
 - **Response Surface**
 - **Sampling and Optimization**

As you click the four DesignXplorer tabs, you might need to scroll to see all options. A grayed-out section becomes available only if a previous option is changed to some specific value that enables the section.

For descriptions of options on DesignXplorer tabs, see:

- [Design Exploration Options \(p. 35\)](#)
- [Design of Experiments Options \(p. 36\)](#)
- [Response Surface Options \(p. 38\)](#)
- [Sampling and Optimization Options \(p. 39\)](#)

For descriptions of options on other tabs, see [Workbench User Preferences](#) in the *Workbench User's Guide*.

3. To change the default value for an option, click directly in the field for the option. Some fields require you to directly enter text while others require you to make selections from dropdown menus or select or clear check boxes.
4. When you finish changing options, click **OK**.

Design Exploration Options

Show Advanced Options: If selected, advanced properties display in *italics* type in various DesignXplorer panes.

Under **Design Points**:

- **Preserve Design Points After DX Run:** If selected, design points created for a design exploration cell are saved to the project's design points table once the solution completes. When this check box is selected, the **Retain Data for Each Preserved Design Point** option is enabled. If this check box is selected, data is retained for each design point saved to the project's design points table. For more information, see [Retaining Data for Generated Design Points \(p. 302\)](#) in the *Workbench User's Guide*.
- **Retry All Failed Design Points:** If selected, additional attempts are made to solve design points that failed to update during the first run. When this check box is selected, the following options are enabled:

- **Number of Retries:** Number of times to try to update failed design points. The default is **5**.
- **Retry Delay (seconds):** Number of seconds to elapse between tries. The default is **120**.

Under **Graph**, the **Chart Resolution** option specifies the number of points for a continuous input parameter axis in a 2D or 3D Response Surface chart. The range is from 2 to 100. The default is **25**. Increasing the number of points enhances the chart resolution.

Under **Sensitivity**:

- **Significance Level:** Relative importance or significance to assume for input variables. The allowable range is from 0.0 to 1.0, where 0 means that all input variables are assumed to be insignificant and 1.0 means that all input variables are assumed to be significant. The default is 0.025.
- **Correlation Coefficient Calculation Type:** Calculation method for determining sensitivity correlation coefficients. Choices are:
 - **Rank Order (Spearman):** Evaluates correlation coefficients based on the rank of samples (default).
 - **Linear (Pearson):** Evaluates correlation coefficients based on the sample values.

Under **Parameter Options**:

- **Display Parameter Full Name:** If selected, full parameter names display rather than short parameter names.
- **Parameter Naming Convention:** Naming style for input parameters within design exploration. Choices are:
 - **Taguchi Style:** Names for parameters are continuous variables and noise variables.
 - **Uncertainty Management Style:** Names for parameters are design variables and uncertainty variables (default).
 - **Reliability Based Optimization Style:** Names for parameters are design variables and random variables.

Under **Messages**, the **Confirm if Min-Max Search can take a long time** option specifies whether to display an alert before performing a Min-Max search operation when there are discrete input parameters. You might want to display such alerts because Min-Max searches can be time-consuming:

Design of Experiments Options

Design of Experiments Type: Algorithm for determining the location of sampling points. Choices are:

- **Central Composite Design** (default)
- **Optimal Space-Filling Design**
- **Box-Behnken Design**
- **Latin Hypercube Sampling Design**

For more information, see [DOE Types \(p. 84\)](#) and [Working with Design Points \(p. 300\)](#).

When **Central Composite Design** is selected, options under **Central Composite Design Options** are enabled:

- **Design Type:** Method for improving the response surface fit for the DOE. Choices are:
 - **Face-Centered**
 - **Rotatable**
 - **VIF-Optimality**
 - **G-Optimality**
 - **Auto-Defined**

For more information, see [Central Composite Design \(CCD\) \(p. 84\)](#) and [Using a Central Composite Design DOE \(p. 94\)](#).

Note:

If you change the setting for **Design Type** here in the **Options** dialog box, new design points are generated for a DOE that has not yet been solved.

- **Enhanced Template:** Specifies whether to use the enhanced template. This check box is enabled only for **Rotatable** and **Face-Centered** design types.

When either **Optimal Space-Filling Design** or **Latin Hypercube Sampling Design** is the algorithm selected, options under **Latin Hypercube Sampling or Optimal Space-Filling** are enabled:

- **Design Type:** Method for improving the response surface fit for the DOE. Choices are:
 - **Max-Min Distance** (default)
 - **Centered L2**
 - **Maximum Entropy**
- **Max Number of Cycles:** Maximum number of iterations that the base DOE is to undergo for the final sample locations to conform to the chosen DOE type.
- **Sample Type:** Method for determining the number of samples. Choices are:
 - **CCD Samples:** Number of samples is the same as that of a corresponding CCD design (default).
 - **Linear Model Samples:** Number of samples is the same as that of a design of linear resolution.
 - **Pure Quadratic Model Samples:** Number of samples is the same as that of a design of pure quadratic resolution, which uses constant and quadratic terms.
 - **Full Quadratic Model Samples:** Number of samples is the same as that of a design of full quadratic resolution, which uses all constant, quadratic and linear terms.

- **User Defined Samples:** Number of DOE samples that you want to have generated.
- **Number of Samples:** Default number of samples. The default is **10**.

For more information, see [Optimal Space-Filling Design \(OSF\) \(p. 85\)](#) and [Latin Hypercube Sampling Design \(p. 88\)](#).

Response Surface Options

- **Type:** Algorithm for generating the response surface. Choices are:
 - **Standard Response Surface - Full 2nd Order Polynomials**
 - **Kriging**
 - **Non-Parametric Regression**
 - **Neural Network**
 - **Genetic Aggregation** (default)

For more information, see [Response Surface Types \(p. 101\)](#) and [Using Response Surfaces \(p. 101\)](#). The algorithm that you select determines if subsequent categories are enabled.

- **Color for Response Surface Based Output Values:** Color in which to display output values that are calculated from a response surface. While simulation output values that are calculated from a design point update always display in black text, DesignXplorer applies the color that is selected here to response surface-based output values in the **Properties** and **Table** panes for all cells and in the **Results** pane for the **Optimization** component (specifically for the Candidate Points chart and Samples chart).
 - Design points, derived parameters with no output parameter dependency, verified candidate points, and all output parameters calculated in a **Direct Optimization** system are simulation-based. Consequently, these output values display in black text.
 - Response points, Min-Max search results, and candidate points in a **Response Surface Optimization** system are based on response surfaces. Consequently, these output values display in the color specified by this option.

In a **Direct Optimization** system, derived and direct output parameters are all calculated from a simulation and so display in black text. In a **Response Surface Optimization** system, the color used for derived values depends on the definition (expression) of the derived parameter. If the expression of the parameter depends on at least one output parameter, either directly or indirectly, the derived values are considered to be based on a response surface and so display in the color specified by this option.

Under **Kriging Options**, the **Kernel Variation Type** option specifies the mode for correlation parameter selection. This option is available only when **Kriging** is the algorithm selected. Choices are:

- **Variable Kernel Variation:** Radial basis function mode that uses one correlation parameter for each design variable (default).

- **Constant Kernel Variation:** Pure Kriging mode that uses a single correlation parameter.

Under **Neural Network Options**, the **Number of Cells** option specifies the number of cells that the neural network uses to control the quality of the response surface. This option is available only when **Neural Network** is the algorithm selected. A higher value allows the neural network to better capture parameter interactions. The recommended range is from 1 to 10. The default is **3**.

Once a response surface is solved, it is possible to switch to another response surface type or change the options for the current response surface in the **Properties** pane for the **Response Surface** cell. Anytime that you change options in the **Properties** pane, you must update the response surface to obtain the new fitting.

Sampling and Optimization Options

Under **Random Number Generation, Repeatability** specifies whether the random number generator is seeded with the same value each time that you generate uncertainty analysis samples (default). If you clear this check box, the random number generator is seeded with the system time when you generate a sample. This option applies to all design exploration systems where random numbers are needed, such as **Direct Optimization, Response Surface Optimization, and Six Sigma Analysis** systems.

Under **Weighted Latin Hypercube, Sampling Magnification** specifies the number of times to reduce regular Latin Hypercube samples while achieving a certain probability of failure (Pf). For example, the lowest probability of failure for 1000 Latin Hypercube samples is approximately 1/1000. The default is 5. A magnification of 5 is meant to use 200 weighted/biased Latin Hypercube samples to approach the lowest probability of 1/1000. You should not use a magnification greater than 5 because a significant Pf error can occur due to highly biased samples.

Under **Optimization:**

- **Method Selection:** Specifies whether **Auto** or **Manual** is the default for optimization method selection in a newly inserted optimization system. To make performing optimizations easy for non-experts, **Auto** is the original setting. For more information, see [Using Goal-Driven Optimizations \(p. 175\)](#).
- **Constraint Handling:** A *constraint satisfaction* filter on samples generated from a Screening, NLPQL, MOGA, or Adaptive Single-Objective optimization that determines what candidates to display in the candidates table. This option can be used for any optimization application and is especially useful for Screening samples to detect the edges of solution feasibility for highly constrained nonlinear optimization problems. Choices are:
 - **Relaxed:** Treats the upper, lower, and equality constraints as objectives. A candidate points that violates an objective is still considered feasible and so is shown in the table.
 - **Strict** (default): Treats the upper, lower, and equality constraints as hard constraints. If any constraint is violated, the candidate is not shown in the table. Depending on the extent of constraint violations, it is possible that no candidate points are shown in the table.

For more information, see [Constraint Handling \(p. 211\)](#).

- **Tolerance Settings:** For **Direct Optimization** and **Response Surface Optimization** systems, indicates whether to display options in the **Optimization** cell for entering tolerance values for objectives and constraints. When this check box is selected (default) and the **Solution Process Update** property for the **Parameter Set** bar is set to **Submit to Design Point Service (DPS)**, options also display in the

Optimization cell for entering initial values for objectives, which are sent to DPS when design points are updated. For more information, see [Tolerance Settings \(p. 209\)](#).

Customizing DesignXplorer with Ansys ACT

With Ansys ACT, you create *extensions* to customize supported Ansys products, including DesignXplorer. ACT provides internal mechanisms that allow you to customize DesignXplorer and then manage the interfaces between DesignXplorer and your extensions.

ACT provides two levels of customization for DesignXplorer:

Direct, API-driven product customization

ACT provides direct, API-driven product customization via standard extensions. In DesignXplorer, this includes the integration of optimizers and sampling (DOE) methods, both custom and third-party, into the design exploration workflow.

For more information, see [DesignXplorer Feature Creation](#) and [DesignExplorer APIs](#) in the *ACT Customization Guide for DesignXplorer*.

Process compression and automation

ACT enables process compression and automation via *wizard* extensions, which enable you to leverage the scripting capabilities of the Workbench framework API. A *wizard* extension loaded in Workbench provides simulation guidance within the **Project Schematic** workflow, walking non-expert users step-by-step through a simulation. As a data-integrated application, you can automate DesignXplorer using a simulation wizard launched from the **Project** tab.

For more information, see [Simulation Wizards](#) in the *Ansys ACT Developer's Guide*.

For information on using extensions in DesignXplorer, see:

- [External Design of Experiments \(p. 89\)](#)
- [Exporting Response Surfaces \(p. 171\)](#)
- [Performing an Optimization with an External Optimizer \(p. 200\)](#)
- [Working with DesignXplorer Extensions \(p. 320\)](#)

DesignXplorer Systems and Components

The topics in this section provide an introduction to using specific DesignXplorer systems and their individual components for your design exploration projects. A component is generally referred to as a cell.

[What is Design Exploration?](#)

[DesignXplorer Systems](#)

[DesignXplorer Components](#)

What is Design Exploration?

Design exploration is a powerful approach used by DesignXplorer for designing and understanding the analysis response of parts and assemblies. It uses a deterministic method based on Design of Experiments (DOE) and various optimization methods, with [parameters \(p. 26\)](#) as its fundamental components. These parameters can come from any supported analysis system, DesignModeler, and various CAD systems. Responses can be studied, quantified, and graphed. Using a [goal-driven optimization \(p. 175\)](#) method, the deterministic method can obtain a multiplicity of [design points \(p. 28\)](#). You can also explore the calculated response surface and generate design points directly from the surface.

After setting up your analysis, you can pick one of the system under **Design Exploration** in the **Toolbox** and then do any of the following:

- Parametrize your solution and view an interpolated response surface for the parameter ranges
- View the parameters associated with the minimum and maximum values of your outputs
- Create a correlation matrix that shows you the sensitivity of outputs to changes in your input parameters
- Set output objectives and see what input parameters meet these objectives
- Produce and consume ROMs for computationally inexpensive, near real-time analysis
- Perform a Six Sigma Analysis on your model

DesignXplorer Systems

The following DesignXplorer systems are available if you have installed Ansys DesignXplorer and have an appropriate license:

[Parameters Correlation System](#)

[Response Surface System](#)

[Goal-Driven Optimization Systems](#)

[3D ROM System](#)

Six Sigma Analysis System

Parameters Correlation System

A **Parameters Correlation** system is used to identify significant input parameters. This is achieved by analyzing the correlation and relative weight of the input parameters for each output parameter.

When a project has many input parameters (more than 10), building an accurate response surface becomes an expensive process. By using a **Parameters Correlation** system, you can identify the most significant input parameters and then disable those that are less significant when building the response surface. With fewer input parameters, the response surface is more accurate and less expensive to build.

The **Parameters Correlation** system contains a single cell: [Parameters Correlation \(p. 46\)](#)

Response Surface System

A response surface exists for each output parameter. Output parameters are represented in terms of the input parameters, which are treated as independent variables.

For the deterministic method, response surfaces for all output parameters are generated in two steps:

- Solving the output parameters for all design points as defined by a DOE
- Fitting the output parameters as a function of the input parameters using regression analysis techniques

The **Response Surface** system contains two cells:

- [Design of Experiments \(p. 44\)](#)
- [Response Surface \(p. 50\)](#)

Goal-Driven Optimization Systems

A [goal-driven optimization \(p. 350\)](#) (GDO) uses a series of design goals to use to generate optimized designs. You can define both objectives and constraints to each output parameter and weight goals in respect to importance.

DesignXplorer offers two types of GDO systems: **Response Surface Optimization** and **Direct Optimization**.

The **Response Surface Optimization** system contains three cells:

- [Design of Experiments \(p. 44\)](#)
- [Response Surface \(p. 50\)](#)
- [Optimization \(p. 54\)](#)

The **Direct Optimization** system contains a single cell: [Optimization \(p. 54\)](#)

3D ROM System

A [3D ROM \(p. 60\)](#) system is used to produce a [ROM \(p. 237\)](#) from a series of simulations. As a stand-alone digital object, a ROM offers a mathematical representation for computationally inexpensive, near real-time analysis.

The **3D ROM** system contains two cells:

- [Design of Experiments \(3D ROM\) \(p. 60\)](#)
- [ROM Builder \(p. 60\)](#)

Six Sigma Analysis System

[Six Sigma Analysis \(p. 385\)](#) (SSA) is a technique for assessing the effect of uncertain input parameters and assumptions on your model.

Using a **Six Sigma Analysis** system, you can determine the extent to which uncertainties in the model affect the results of the analysis. An uncertainty (random quantity) is a parameter whose value is impossible to determine at a given point in time (if it is time-dependent) or at a given location (if it is location-dependent). An ambient temperature is an example. You cannot know precisely what the temperature will be one week from now in a given city.

The **Six Sigma Analysis** system contains three cells:

- [Design of Experiments \(SSA\) \(p. 61\)](#)
- [Response Surface \(SSA\) \(p. 50\)](#)
- [Six Sigma Analysis \(p. 62\)](#)

DesignXplorer Components

Design Exploration systems are made up of one or more components or cells. Double-clicking a cell in the **Project Schematic** opens its component tab. Virtually all component tabs contain these four panes: **Outline**, **Properties**, **Table**, and **Chart**.

The content in a pane depends on the object selected in the **Outline** pane. The following topics summarize the component tabs for the various cells in **Design Exploration** systems:

[Design of Experiments Component Reference](#)

[Parameters Correlation Component Reference](#)

[Response Surface Component Reference](#)

[Optimization Component Reference](#)

[3D ROM Component Reference](#)

[Six Sigma Analysis Component Reference](#)

Design of Experiments Component Reference

Design of Experiments (DOE) is a technique for determining the location of sampling points. A **Design of Experiments** cell is present in the following design exploration systems: **Direct Optimization**, **Response Surface**, **Response Surface Optimization**, **ROM Builder**, and **Six Sigma Analysis**.

Note:

The DOE for a **3D ROM** system or a **Six Sigma Analysis** system can share data only with the DOE for another DesignXplorer system of the same type.

The **Design of Experiments** tab allows you to preview or generate design points. The **Preview** operation generates design points but does not solve them. The **Update** operation both generates and solves design points. On the **Design of Experiments** tab, you can set input parameter limits and properties for the DOE and view the design points table and several parameter charts. For more information, see:

- [Using Design of Experiments \(p. 83\)](#)
- [Working with Design Points \(p. 300\)](#)
- [Design of Experiments Options \(p. 36\)](#)

The following panes in the **Design of Experiments** tab allow you to customize your DOE and view the updated results.

Outline:

- Select **Design of Experiments** and change DOE properties.
- Select input parameters and change their limits.
- Select output parameter and view their minimum and maximum values.
- Select charts to view available charts and change chart types and data properties. You can use the **Toolbox** or context menu to insert as many charts as you want.

Properties:

- **Preserve Design Points After DX Run:** Specifies whether to retain design points at the project level each time that the DOE is updated. If you select this check box, **Retain Data for Each Preserved Design Point** is shown. If you also select this check box, in addition to saving the design points to the project's design points table, data for each design point is saved. For more information, see [Retaining Data for Generated Design Points \(p. 302\)](#) in the *Workbench User's Guide*.
- **Number of Retries:** Specify the number of times DesignXplorer is to try to update failed design points. If the **Retry All Failed Design Points** check box is not selected on the **Design Exploration** tab in the **Options** dialog box, the default is **0**. However, you can specify the default number of retries for this specific project here. When the **Number of Retries** property is not set to **0**, **Retry Delay (seconds)** specifies how much time is to elapse between tries.
- **Design of Experiments Type:** Specifies the DOE type to use. Choices follow. For descriptions and specific properties, see [DOE Types \(p. 84\)](#).

- Central Composite Design
- Optimal Space-Filling Design
- Box-Behnken Design
- Sparse Grid Initialization
- Custom
- Custom + Sampling
- Latin Hypercube Sampling Design
- External sampling methods as defined by the DOE extensions loaded to the project.

Table:

Displays the design points and input parameter data when previewing. On updating, the table also displays output parameter data. You can add data points manually if **Design of Experiments Type** is set to **Custom**.

Chart:

Displays the available charts:

[Parameters Parallel Chart](#)

[Design Points vs Parameter Chart](#)

Parameters Parallel Chart

The Parameters Parallel chart generates a graphical display of the design points table using parallel Y axes to represent all inputs and outputs. In the **Outline** pane under **Charts**, select **Parameters Parallel** to display this chart in the **Chart** pane. Use the **Properties** pane as follows:

- **Display Parameter Full Name:** Indicate whether to show the full parameter name or the short parameter name.
- Use the **Enabled** check boxes for the input parameters to enable or disable the display of parameter axes on the chart.
- Click a line on the chart to display input and output values for this line in the **Input Parameters** and **Output Parameters** sections of the **Properties** pane.
- Change various generic [chart properties](#).

The chart displays only updated design points. If the DOE does not yet contain any updated design points, output parameters are automatically disabled from the chart. The axes that are visible correspond to the input parameters for the design points.

The Parameters Parallel chart supports interactive exploration of the DOE. When you place the mouse cursor on the graph, sliders appear at the upper and lower bounds of each axis. You can use the sliders to easily filter for each parameter. Design points that fall outside of the bounds defined by the sliders are dynamically hidden.

You can also look at the data in this chart as a Spider chart. Right-click in an empty area of the chart and select **Edit Properties**. Then, in the **Properties** pane for the chart, change **Chart Type** to **Spider**. The Spider chart shows all input and output parameters arranged in a set of radial axes spaced equally. Each design point is represented by a corresponding envelope defined in the radial axes.

Design Points vs Parameter Chart

The Design Points vs Parameter chart generates a graphical display for plotting design points versus any input or output parameter. In the **Outline** pane under **Charts**, select **Design Points vs Parameter** to display this chart in the **Chart** pane. Use the **Properties** pane as follows:

- **Display Parameter Full Name:** Indicate whether to show the full parameter name or the short parameter name.
- **X-Axis (Bottom) X-Axis (Top), Y-Axis (Left), Y-Axis (Right):** Design points can be plotted for either **X-Axis (Bottom)** or **X-Axis (Top)** against input and output parameters on any of the other axes.
- Change various generic [chart properties](#).

Parameters Correlation Component Reference

A linear association between parameters is evaluated using Spearman's or Pearson's product-moment coefficient. Correlation matrix and sensitivity charts are generated to demonstrate the correlation between input and input parameters and the sensitivity of output to input parameters.

A nonlinear (quadratic) association between parameters is evaluated using coefficient of determination of quadratic regression between parameters. A determination matrix is generated between the parameters to convey information of quadratic correlation if there is any and it isn't detectable with linear Spearman's or Pearson's correlation coefficient.

For more information, see:

- [Using Parameters Correlations \(p. 65\)](#)
- [Working with Design Points \(p. 300\)](#)
- [Correlation Coefficient Theory \(p. 403\)](#)

The following panes in the **Parameters Correlation** tab allow you to customize your search and view the results.

Outline:

- Select **Parameters Correlation** and change properties and view the number of samples generated for this correlation.
- Select input parameters and change their limits.
- Select output parameters and view their minimum and maximum values.
- Select charts to view available charts and change chart types and data properties.

Properties:

- **Preserve Design Points After DX Run:** Specifies whether to retain design points at the project level from this parameters correlation. If this check box is selected, **Retain Data for Each Preserved Design Point** is shown. If you also select this check box, in addition to saving the design points to the project's design points table, data for each design point is saved. For more information, see [Retaining Data for Generated Design Points \(p. 302\)](#) in the *Workbench User's Guide*.
- **Number of Retries:** Number of times DesignXplorer is to try to update failed design points. If the **Retry All Failed Design Points** check box is not selected on the **Design Exploration** tab in the **Options** dialog box, the default is **0**. However, for a correlation that is not linked to a response surface, you can specify the default number of retries for this specific project here. When **Number of Retries** is not set to **0**, **Retry Delay (seconds)** specifies how much time is to elapse between tries.
- **Reuse the samples already generated:** Specifies whether to reuse the samples generated in a previous correlation.
- **Correlation Type:** Algorithm to use for the parameter correlation. Choices are:
 - **Spearman**
 - **Pearson**
- **Number Of Samples:** Maximum number of samples to generate for this correlation. This value must be greater than the number of enabled input parameters.
- **Auto Stop Type:** Choices are **Execute All Simulations** and **Enable Auto Stop**. When **Enable Auto Stop** is selected, set the additional options that are shown:
 - **Mean Value Accuracy:** Desired accuracy for the mean value of the sample set.
 - **Standard Deviation Accuracy:** Desired accuracy for the standard deviation of the sample set.
 - **Convergence Check Frequency:** Number of simulations to execute before checking for convergence. This value must be greater than the number of enabled input parameters.
 - **Size of Generated Sample Set:** Read-only value indicating the number of samples generated for the correlation solution.
 - **Converged:** Read-only value indicating whether the process has converged.
- **Relevance Threshold:** Threshold to use to filter the major input parameters.
- **Correlation Filtering:** Specifies whether to filter major input parameters based on correlation values.
- **R2 Contribution Filtering:** Specifies whether to filter major input parameters based on R^2 contributions.
- **Maximum Number of Major Inputs:** Maximum number of input parameters selected as major input parameters. By default, this number is the minimum of the current number of input parameters and 20 (where 20 is the recommended maximum number of input parameters for a response surface).

Table:

Displays both a correlation matrix and a determination matrix for the input and output parameters.

Chart:

Displays the charts available:

Correlation Scatter Chart

Correlation Matrix Chart

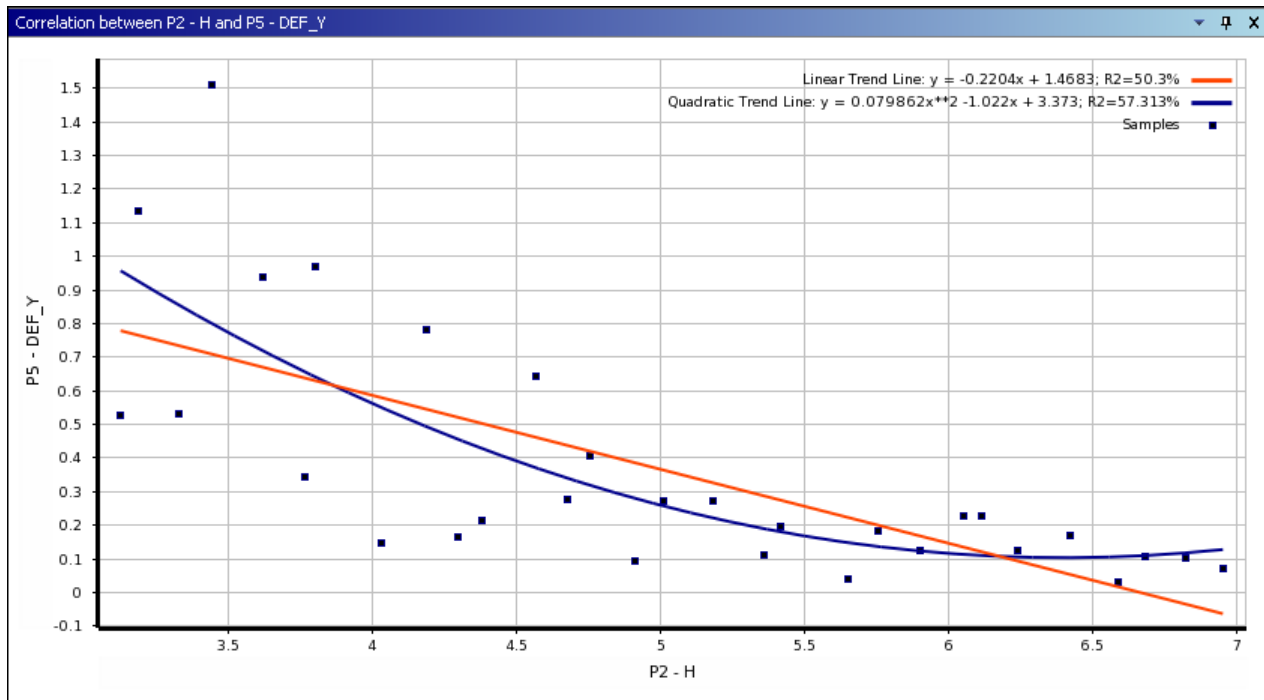
Determination Matrix Chart

Sensitivities Chart

Determination Histogram Chart

Correlation Scatter Chart

The Correlation Scatter chart shows a sample scatter plot of a parameter pair. Two trend lines, linear and quadratic (p. 72) curves, are added to the sample points of the parameter pair. The trend line equations are displayed in the chart legend. The chart conveys a graphical presentation of the degree of correlation between the parameter pair in linear and quadratic trends.



You can create a Correlation Scatter chart for a given parameter combination by right-clicking the associated cell in the Correlation Matrix chart and selecting **Insert <x-axis> vs <y-axis> Correlation Scatter**.

To view the Correlation Scatter chart, in the **Outline** pane under **Charts**, select **Correlation Scatter**. Use the **Properties** pane as follows:

- Choose the parameters to display on the X axis and Y axis.
- Enable or disable the display of the quadratic and linear trend lines.

- View the coefficient of determination and the equations for the quadratic and linear trend lines.
- Change various generic [chart properties](#).

Correlation Matrix Chart

The Correlation Matrix chart provides information about the linear correlation between a parameter pair, if any. The degree of correlation is indicated by color in the matrix. Placing your cursor over a particular square shows you the correlation value for the two parameters associated with that square.

To view the Correlation Matrix chart, in the **Outline** pane under **Charts**, select **Correlation Matrix**. Use the **Properties** pane as follows:

- Enable or disable the parameters that are displayed on the chart.
- Change various generic [chart properties](#).

Determination Matrix Chart

The Determination Matrix Chart provides information about the nonlinear ([quadratic \(p. 72\)](#)) correlation between a parameter pair, if any. The degree of correlation is indicated by color in the matrix. Placing your cursor over a particular square shows you the correlation value for the two parameters associated with that square.

To view the Determination Matrix chart, in the **Outline** pane under **Charts**, select **Determination Matrix**. Use the **Properties** pane as follows:

- Enable or disable the parameters that are displayed on the chart.
- Change various generic [chart properties](#).

Sensitivities Chart

The Sensitivities chart allows you to graphically view the global sensitivities of each output parameter with respect to the input parameters.

To view the Sensitivities chart, in the **Outline** pane under **Charts**, select **Sensitivities**. Use the **Properties** pane as follows:

- **Chart Mode:** Set to **Bar** or **Pie**.
- Enable or disable the parameters that are displayed on the chart.
- Change various generic [chart properties](#).

Determination Histogram Chart

The Determination Histogram chart allows you to see the effect of the input parameters for a given output parameter.

The full model R^2 represents the variability of the output parameter that can be explained by a linear (or quadratic) correlation between the input parameters and the output parameter.

The value of the bars corresponds to the linear (or quadratic) determination coefficient of each input associated to the selected output.

To view the Determination Histogram chart, in the **Outline** pane under **Charts**, select **Determination Histogram**. Use the **Properties** pane as follows:

- **Determination Type:** Set to **Linear** or **Quadratic**.
- **Threshold R^2 :** Enables you to filter input parameters by hiding those with a determination coefficient lower than the given threshold.
- Choose the output parameter
- Change various generic [chart properties](#).

Response Surface Component Reference

A response surface is built from the input and output values of design points for the **Design of Experiments** cell based on a chosen response surface type. In the **Response Surface** tab, you can view the input parameter limits and initial values, set the properties of the response surface algorithm, view tables containing different types of response surface data, and view several types of response charts.

For more information, see:

- [Using Response Surfaces \(p. 101\)](#)
- [Response Surface Types \(p. 101\)](#)
- [Response Surface Refinement \(p. 122\)](#)
- [Min-Max Search \(p. 127\)](#)
- [Quality Metrics for Response Surfaces \(p. 130\)](#)
- [Response Surface Charts \(p. 143\)](#)
- [Response Surface Theory \(p. 329\)](#)

The following panes in the **Response Surface** tab allow you to customize your response surface and view the results:

Outline:

- Select **Response Surface** to specify the **Response Surface Type**, change response surface properties, and view the tolerances table (Genetic Aggregation only) or the response points table (all other response surface types).
- Perform a Min-Max search and view results.

- Under **Refinement**, view the tolerances table (Genetic Aggregation only), Convergence Curves chart (Genetic Aggregation only), and refinement points table.
- Under **Quality**, view the goodness of fit results, the verification points table, and the Predicted vs Observed chart.
- Under **Response Points**, view the response points table and view the properties and available charts for individual response points.

Properties:

- **Preserve Design Points After DX Run:** Select this check box if you want to retain at the project level design points that are created when refinements are run for this response surface. If this property is set, **Retain Data for Each Preserved Design Point** is available. If this check box is selected, in addition to saving the design points to the project's design points table, data for each design point is saved. For more information, see [Retaining Data for Generated Design Points \(p. 302\)](#).

Note:

Selecting this check box does not preserve any design points unless you run either a manual refinement or one of the Kriging refinements because the response surface uses the design points generated by the DOE. If the DOE of the response surface does not preserve design points, when you perform a refinement, only the refinement points are preserved at the project level. If the DOE is set to preserve design points and the response surface is also set to preserve design points, when you perform a refinement, the project contains the DOE design points and the refinement points.

- **Number of Retries:** Number of times DesignXplorer is to try to update the failed design points. If the **Retry All Failed Design Points** option is not selected on the **Design Exploration** tab in the **Options** dialog box, the default is **0**. However, you can specify the default number of retries for this specific project here. When **Number of Retries** is not set to **0**, **Retry Delay (seconds)** specifies how much time is to elapse between tries.
- **Response Surface Type:** The type of response surface. Choices follow. For descriptions, see [Response Surface Types \(p. 101\)](#).
 - Genetic Aggregation
 - Standard Response Surface – Full 2nd-Order Polynomial
 - Kriging
 - Non-Parametric Regression
 - Neural Network
 - Sparse Grid
- **Refinement Type:** Where applicable, select **Manual** to enter refinement points manually or **Auto-Refinement** to automate the refinement process.

- **Generate Verification Points:** Specify the number of verification points to be generated. The default value is **1**. The results are included in the verification points table and the Goodness of Fit chart.
- Settings for selected response surface type, as applicable. For more information, see [Response Surface Types \(p. 101\)](#).

Table: Depending on your selection in the **Outline** pane, displays one of the following tables:

- Tolerances (Genetic Aggregation only)
- Min-Max Search
- Refinement Points
- Goodness of Fit
- Verification Points
- Response Points

Chart:

Displays the available charts for the response point selected in the **Outline** pane:

[Response Chart](#)

[Local Sensitivity Charts](#)

[Spider Chart](#)

Response Chart

The Response chart allows you to graphically view the effect that changing each input parameter has on the displayed output parameter.

You can add response points to the response points table by right-clicking the Response chart and selecting **Explore Response Surface at Point**, **Insert as Design Point**, **Insert as Refinement Point** or **Insert as Verification Point**.

Use the **Properties** pane as follows:

- **Display Parameter Full Name:** Specifies whether to display the full parameter name or the parameter ID on the chart.
- **Mode:** Set to **2D**, **3D**, or **2D Slices**.
- **Chart Resolution Along X:** Sets the number of points to display on the X axis response curve. The default is **25**.
- **Chart Resolution Along Y:** Sets the number of points to display on the Y axis response curve when **Mode** is set to **3D**. The default is **25**.
- **Number of Slices:** Sets the number of slices when **Mode** is set to **2D Slices** and there are continuous input parameters.
- **Show Design Points:** Specifies whether to display design points on the chart.

- Choose the input parameters to display in either the first axis option or the first and second axis options, depending on the chart mode.
- Choose the output parameter to display in the last axis option.
- Use the sliders or drop-down menus to change the values of the input parameters that are not displayed to see how they affect the values of displayed parameters. You can enter specific values in the boxes above the sliders.
- View the interpolated output parameter values for the selected set of input parameter values.
- Change various generic [chart properties](#).

For more information, see [Response Surface Charts \(p. 143\)](#).

Local Sensitivity Charts

The Local Sensitivity chart and Local Sensitivity Curves charts allow you to graphically view the effect that changing each input parameter has on the output parameters. Use the **Properties** pane as follows:

- **Display Parameter Full Name:** Specifies whether to display the full parameter name or the parameter ID on the chart.
Chart Mode: Set to **Bar** or **Pie**. This option is available only for the Local Sensitivity chart.
- **Axes Range:** Set to **Use Min Max of the Output Parameter** or **Use Chart Data**.
- **Chart Resolution:** Set the number of points per curve. The default is **25**.
- Use the sliders to change the values of the input parameters to see how the sensitivity changes for each output.
- View the interpolated output parameter values for the selected set of input parameter values.
- Change various generic [chart properties](#).

For more information, see [Response Surface Charts \(p. 143\)](#).

Spider Chart

The Spider chart allows you to visualize the effect that changing the input parameters has on all of the output parameters simultaneously. Use the **Properties** pane as follows:

- **Display Parameter Full Name:** Specifies whether to display the full parameter name or the parameter ID on the chart.
- Use the sliders to change the values of the input parameters to see how they affect the output parameters.
- View the interpolated output parameter values for the selected set of input parameter values.
- Change various generic [chart properties](#).

For more information, see [Response Surface Charts \(p. 143\)](#).

Optimization Component Reference

Goal-driven optimization (GDO) is a constrained, multi-objective optimization technique in which the best possible designs are obtained from a sample set given the objectives or constraints that you set for parameters. Both type of GDO systems (**Response Surface Optimization** and **Direct Optimization**) contain an **Optimization** cell. You use the **Design of Experiments** and **Response Surface cells** in the same way as described previously in this section. For a **Direct Optimization** system, the creation of data transfer links is also possible.

For more information, see:

- [Using Goal-Driven Optimizations \(p. 175\)](#)
- [Goal-Driven Optimization Theory \(p. 350\)](#)

The following panes in the **Optimization** tab allow you to customize your GDO and view the results:

Outline: Allows you to select the following nodes and perform related actions in the tab:

- **Optimization:**
 - Change optimization properties and view the size of the generated sample set.
 - View an optimization summary with details on the study, method, and returned candidate points.
 - View the Convergence Criteria chart for the optimization. For more information, see [Using the Convergence Criteria Chart \(p. 217\)](#).
- **Objectives and Constraints:**
 - View the objectives and constraints defined in the project.
 - Enable or disable objectives and constraints.
 - Select an objective or constraint and view its properties, the calculated minimum and maximum values of each of the outputs, and History chart. For more information, see [History Chart \(p. 57\)](#).
- **Domain:**
 - View the available input parameters for the optimization.
 - Enable or disable input parameters or parameter relationships.
 - Select an input parameter or parameter relationship to view and edit its properties or to see its History chart. For more information, see [History Chart \(p. 57\)](#).

- **Raw Optimization Data:** For **Direct Optimization** systems, when an optimization update is finished, DesignXplorer saves the design point data calculated during the optimization. You can access this data by selecting **Raw Optimization Data** in the **Outline** pane.

Note:

The design point data is displayed without analysis or optimization results. The data does not show feasibility, ratings, Pareto fronts, and so on.

- **Convergence Criteria:** View the Convergence Criteria chart and specify the criteria to display. For more information, see [Using the Convergence Criteria Chart \(p. 217\)](#).

- **Results:**

Select one of the result types available to view results in the **Charts** pane and, in some cases, in the **Table** pane. When a result is selected, you can change the data properties of its related chart (X axis and Y axis parameters, parameters to display on the bar chart, and so on), and edit its table data.

Properties: When **Optimization** is selected in the **Outline** pane, the **Properties** pane allows you to specify:

- **Method Name:** Choices for methods follow. If optimization extensions are loaded to the project, you can also choose an external optimizer.
 - MOGA
 - NLPQL
 - MISQP
 - Screening
 - Adaptive Single-Objective
 - Adaptive Multiple-Objective
- Relevant settings for the selected **Method Name**. Depending on the method of optimization, these can include specifications for samples, sample sets, number of iterations, and allowable convergence or Pareto percentages.

For more information, see [Goal-Driven Optimization Methods \(p. 180\)](#).

Table: Before the update, specify input parameter domain settings and objective and constraint settings:

- **Optimization Domain**
 - Set the **Upper Bound** and **Lower Bound** for each input parameter. For NLPQL and MISQP optimizations, also set the **Starting Value**.
 - Set **Left Expression**, **Right Expression**, and **Operator** for each parameter relationship.

For more information, see [Defining the Optimization Domain \(p. 201\)](#).

- **Optimization Objectives and Constraints**

- For each parameter, you can define an objective, constraint, or both. Options vary according to parameter type.
- For a parameter with **Objective Type** set to **Seek Target**, you specify a target.
- For a parameter with **Constraint Type** set to **Lower Bound <= Values <= Upper Bound**, you use **Lower Bound** and **Upper Bound** to specify the target range.
- For a parameter with an **Objective** or **Constraint** defined, you specify the relative **Objective Importance** or **Constraint Importance** of that parameter in regard to the other objectives.
- For a parameter with a **Constraint** defined (such as output parameters, discrete parameters, or continuous parameters with manufacturable values), you specify the **Constraint Handling** for that parameter.

For more information, see [Defining Optimization Objectives and Constraints \(p. 205\)](#).

During an update of a **Direct Optimization** system, if you select an objective, constraint, or input parameter in the **Outline** pane, the **Table** pane shows all of the design points being calculated by the optimization. For iterative optimization methods, the display is refreshed dynamically after each iteration, allowing you to track the progress of the optimization by simultaneously viewing design points in the **Table** pane, History charts in the **Charts** pane, and History chart sparklines in the **Outline** pane. For the Screening optimization method, these objects are updated only after the optimization has completed.

After the update, when you select **Candidate Points** under **Results** in the **Outline** pane, the **Table** pane displays up to the maximum number of requested candidates generated by the optimization. The number of gold stars or red crosses displayed next to each objective-driven parameter indicate how well the parameter meets the stated objective, from three red crosses for the worst to three gold stars for the best. The **Table** pane also allows you to add and edit your own candidate points, view values of candidate point expressions, and calculates the percentage of variation for each parameter for which an objective has been defined. For more information, see [Working with Candidate Points \(p. 212\)](#).

Note:

Goal-driven parameter values with inequality constraints receive either three stars to indicate that the constraint is met or three red crosses to indicate that the constraint is not met.

You can verify predicted output values for each candidate. For more information, see [Verifying Candidates by Design Point Update \(p. 215\)](#).

Results:

The following result types are available:

- [Convergence Criteria Chart](#)
- [History Chart](#)

[Candidate Points Results](#)

[Tradeoff Chart](#)

[Samples Chart](#)

[Sensitivities Chart](#)

Convergence Criteria Chart

The Convergence Criteria chart allows you to view the evolution of the convergence criteria for each of the iterative optimization methods. The chart, updated after each iteration, is not available for the Screening optimization method.

The Convergence Criteria chart is the default optimization chart, so it displays in the **Chart** pane unless you select another chart type. When **Convergence Criteria** is selected in the **Outline** pane, the **Properties** pane displays the convergence criteria relevant to the selected optimization method in read-only mode. Various generic [chart properties](#) can be changed for the Convergence Criteria chart.

The chart remains available when the optimization update is complete. The legend shows the color-coding for the convergence criteria.

For more information, see:

- [Using the Convergence Criteria Chart \(p. 217\)](#)
- [Using the Convergence Criteria Chart for Multiple-Objective Optimization \(p. 217\)](#)
- [Using the Convergence Criteria Chart for Single-Objective Optimization \(p. 218\)](#)

History Chart

The History chart allows you to view the history of a single enabled objective, constraint, input parameter, or parameter relationship during the update process. For iterative optimization methods, the History chart is updated after each iteration. For the Screening optimization method, it is updated only when the optimization is complete.

In the **Outline** pane, select an object under **Objectives and Constraints** or an input parameter or parameter relationship under **Domain**. The **Properties** pane displays various properties for the selected object. Various generic [chart properties](#) can be changed for both types of History chart.

In the **Chart** pane, the color-coded legend allows you to interpret the chart. In the **Outline** pane, a sparkline graphic of the History chart is displayed next to each objective, constraint, and input parameter object.

For more information, see:

- [Using the History Chart \(p. 219\)](#)
- [Working with the History Chart in the Chart Pane \(p. 220\)](#)
- [Viewing History Chart Sparklines in the Outline Pane \(p. 224\)](#)
- [Using the History Chart for an Objective or Constraint \(p. 225\)](#)

- [Using the Input Parameter History Chart \(p. 226\)](#)

Candidate Points Results

The **Candidate Points Results** object in the **Outline** pane has both **Chart** and **Table** panes in which you can see candidate points and the data for one or more selected parameters. The **Chart** pane displays a color-coded legend that enables you to interpret the samples, candidate points identified by the optimization, candidates inserted manually, and candidates for which output values have been verified by a design point update. The **Table** pane displays output parameter values calculated from a simulation in black text. Output parameter values calculated from a response surface are displayed in the custom color specified on the **Response Surface** tab in the **Options** dialog box. For more information, see [Response Surface Options \(p. 38\)](#)

In the **Outline** pane under **Results**, select **Candidate Points**.

Properties: The following options are applied to results in both the **Table** and **Chart** panes.

- **Display Parameter Relationships:** Select to display parameter relationships in the candidate points table.
- **Display Parameter Full Name:** Specifies whether to display the full parameter name or short parameter name.
- **Show Candidates:** Select to show candidates in the results.
- **Coloring Method:** Specifies whether to color the results according to candidate type or source type.
- **Show Samples:** Select to show samples in the results.
- **Show Starting Point:** Select to show the starting point on the chart (NLPQL and MISQP only).
- **Show Verified Candidates:** Select to show verified candidates in the results (**Response Surface Optimization** system only).
- Enable or disable the display of input and output parameters.
- Change various generic [chart properties](#) for the results in the **Chart** pane.

For more information, see:

- [Using Candidate Point Results \(p. 228\)](#)
- [Understanding the Display of Candidate Point Results \(p. 228\)](#)
- [Candidate Point Results: Properties \(p. 229\)](#)

Tradeoff Chart

The Tradeoff chart allows you to view the Pareto fronts created from the samples generated in the goal-driven optimization. In the **Outline** pane under **Charts**, select **Tradeoff** to display this chart in the **Chart** pane. Use the **Properties** pane as follows:

- **Chart Mode:** Set to **2D** or **3D**.

- **Number of Pareto Fronts to Show:** Set the number of Pareto fronts that are displayed on the chart.
- **Show infeasible points:** Enable or disable the display of infeasible points. This option is available when constraints are defined.
- Click a point on the chart to display a **Parameters** section that shows the values of the input and output parameters for this point.
- Set the parameters to display on the X axis and Y axis.
- Change various generic [chart properties](#) for the results in the **Chart** pane.

For more information, see:

- [Using the Tradeoff Chart \(p. 231\)](#)
- [Using Tradeoff Studies \(p. 231\)](#)

Samples Chart

The Samples chart allows you to visually explore a sample set given defined objectives. In the **Outline** pane under **Charts**, select **Samples** to display this chart in the **Chart** pane. Use the **Properties** pane as follows:

- **Chart Mode:** Set to **Candidates** or **Pareto Fronts**. When **Pareto fronts** is selected, the following options can be set:
 - **Number of Pareto Fronts to Show:** Either enter the value or use the slider to select the number of Pareto fronts to display.
 - **Coloring Method:** Set to **per Samples** or **per Pareto Front**.
- **Show infeasible points:** Enable or disable the display of infeasible points. This option is available when constraints are defined.
- Click a line on the chart to display the values of the input and output parameters for this line in the **Parameters** section. Use the **Enabled** check box to enable or disable the display of parameter axes on the chart.
- Change various generic [chart properties](#) for the results in the **Chart** pane.

Sensitivities Chart

The Sensitivities chart allows you to graphically view the global sensitivities of each output parameter with respect to the input parameters. In the **Outline** pane under **Charts**, select **Sensitivities** to display this chart in the **Chart** pane. Use the **Properties** pane as follows:

- **Chart Mode:** Set to **Bar** or **Pie**.
- Enable or disable the parameters that are displayed on the chart.
- Change various generic [chart properties](#) for the results in the **Chart** pane.

For more information, see:

- [Working with Sensitivities \(p. 313\)](#)
- [Using the Sensitivities Chart \(GDO\) \(p. 231\)](#)

3D ROM Component Reference

The **3D ROM** system consists of a **Design of Experiments (3D ROM)** cell and a **ROM Builder** cell. This type of system is used to produce a **ROM (p. 240)** (reduced order model), which is a stand-alone digital object that offers a mathematical representation for computationally inexpensive, near real-time analysis.

Design of Experiments (3D ROM)

The **Design of Experiments (3D ROM)** cell is basically the same as the [Design of Experiments \(p. 44\)](#) cell in other **Design Exploration** systems. It allows you to set up the input parameters and generate the samples for the analysis. However, the DOE for a **3D ROM** system can share data only with the DOE for another **3D ROM** system.

In the **Outline** pane for the **Design of Experiments (3D ROM)** cell, all input parameters are selected for use by default. In the **Properties** pane for each input variable, you set lower and upper bounds. Accepting the default values for all other properties is generally recommended.

When you perform an update, the design points for building the ROM are inserted in the **Table** pane and their results are calculated. As each design point is updated, its results are saved to a ROM snapshot file (ROMSNP).

ROM Builder

The **ROM Builder** cell provides for setting up the solver system for ROM production. While ROM setup is specific to the ANSYS product, the workflow for producing a ROM is generic. When you perform an update, the ROM is built. Once the update finishes, you can open the ROM in the **ROM Viewer** and export the ROM.

Note:

Currently, you can set up and build a ROM only for a Fluent system. ROM production [examples \(p. 240\)](#) are provided.

Six Sigma Analysis Component Reference

The **Six Sigma Analysis** system consists of a **Design of Experiments** cell, a **Response Surface** cell, and a **Six Sigma Analysis** cell. The **Design of Experiments** cell allows you to set up the input parameters, which Six Sigma Analysis refers to as *uncertainty parameters*, and generate the samples for the analysis. The **Response Surface** cell is the same as the **Response Surface** cell in a **Response Surface** system. The **Six Sigma Analysis** cell allows you to set up the Six Sigma Analysis and view the results.

Design of Experiments (SSA)

The **Design of Experiments** cell in a **Six Sigma Analysis** system is basically the same as the **Design of Experiments** cell in other **Design Exploration** systems. In the **Outline** pane, all input parameters are selected for use as uncertainty parameters by default. If you want any input parameters to be treated as deterministic parameters, clear the check boxes next to these parameters.

Before solving the DOE, you must set up input parameter options. The following panes for the **Design of Experiments** cell contain objects that are unique to Six Sigma Analysis:

Outline:

- Select the input parameters and change their distribution properties.
- View the **Skewness** and **Kurtosis** properties for each input parameter distribution.
- View the calculated mean and standard deviation for all distribution types except Normal and Truncated Normal where you can set those values.
- View the lower bound and upper bound for each parameter, which are used to generate the DOE.

Properties: When an input parameter is selected in the **Outline** pane, the **Properties** pane allows you to set its properties:

- **Distribution Type:** Type of distribution associated with the input parameter. Choices follow. For more information, see [Distribution Functions \(p. 391\)](#).
 - Uniform
 - Triangular
 - Normal
 - Truncated Normal
 - Lognormal
 - Exponential
 - Beta
 - Weibull
- **Maximum Likely Value** (Triangular Only)
- **Log Mean** (Lognormal only)
- **Log Standard Deviation** (Lognormal only)
- **Exponential Decay Parameter** (Exponential only)
- **Beta Shape R** (Beta only)
- **Beta Shape T** (Beta only)

- **Weibull Exponent** (Weibull only)
- **Weibull Characteristic Value** (Weibull only)
- **Non-Truncated Normal Mean** (Truncated Normal only)
- **Non-Truncated Normal Standard Deviation** (Truncated Normal only)
- **Mean** (Normal only)
- **Standard Deviation** (Normal only)
- **Distribution Lower Bound** (All but Normal and Lognormal)
- **Distribution Upper Bound** (Uniform, Triangular, Truncated Normal, and Beta only)

Table: When an output parameter or chart is selected in the **Outline** pane, the **Table** pane displays the design points table, which populates automatically during the solving of the points. When an input parameter is selected in the **Outline** pane, the **Table** pane displays data for each of the samples in the set:

- **Quantile:** Input parameter value point for the given PDF and CDF values.
- **PDF:** Probability Density Function of the input parameter along the X axis.
- **CDF:** Cumulative Distribution Function is the integration of PDF along the X axis.

Chart : When an input parameter is selected in the **Outline** pane, the **Chart** pane displays the Probability Density Function and Cumulative Distribution Function for the distribution type chosen for the input parameter. The Parameters Parallel chart and Design Points vs Parameter chart are also available, just as in a [standard DOE \(p. 44\)](#).

Six Sigma Analysis

Once you have assigned distribution functions to the input parameters in the DOE, you can update the project to perform the Six Sigma Analysis. Once the analysis is finished, you can see results in the **Six Sigma Analysis** cell. For more information, see [Using Six Sigma Analysis \(p. 283\)](#) and [Statistical Measures \(p. 287\)](#).

The following panes for the **Six Sigma Analysis** cell allow you to customize your analysis and view the results:

Outline:

- Select each input parameter and view its distribution properties, statistics, upper and lower bounds, initial value, and distribution chart.
- Select each output parameter and view its calculated maximum and minimum values, statistics and distribution chart.
- Set the table display format for each parameter to **Quantile-Percentile** or **Percentile-Quantile**.

Properties:

For Six Sigma Analysis, set the following properties:

- **Sampling Type:** Type of sampling for the Six Sigma Analysis. For more information, see [Sample Generation](#) (p. 395).
 - **LHS:** When selected, the Latin Hypercube Sampling technique is used.
 - **WLHS:** When selected, the Weighted Latin Hypercube Sampling technique is used.
- **Number of Samples:** Number of samples to generate. The default value is **10000**.

For parameters, **Probability Table** specifies how to display analysis information in the **Table** pane:

- **Quantile-Percentile**
- **Percentile-Quantile**

Table: Displays data for each of the samples in the set:

- **<Parameter Name>:** Valid value in the range for the parameter.
- **Probability:** Probability that the parameter is less than or equal to the specified value.
- **Sigma Level:** Approximate number of standard deviations away from the sample mean for the given sample value.

If **Probability Table** is set to **Quantile-Percentile** in the **Properties** pane, you can edit the parameter value and see the corresponding **Probability** and **Sigma Level** values. If **Probability Table** is set to **Percentile-Quantile**, the columns are reversed. You can then enter a **Probability** or **Sigma Level** value and see the corresponding changes in the other columns.

Chart: When a parameter is selected in the **Outline** pane, the **Chart** pane displays the Probability Density Function and Cumulative Distribution Function. A global Sensitivities chart is available in the **Outline** pane.

Sensitivities Chart (SSA)

The Sensitivities chart allows you to graphically view the global sensitivities of each output parameter with respect to the input parameters in the Six Sigma Analysis. In the **Outline** pane under **Charts**, select **Sensitivities** to display the chart in the **Chart** pane. Use the **Properties** pane as follows:

- **Chart Mode:** Set to **Bar** or **Pie**.
- Enable or disable the parameters that are displayed on the chart.
- Change various generic [chart properties](#) for the results in the **Chart** pane.

For more information, see [Working with Sensitivities](#) (p. 313) and [Statistical Sensitivities in a SSA](#) (p. 398).

Using Parameters Correlations

The application of goal-driven optimization and Six Sigma Analysis (SSA) in a finite element-based framework is always a challenge in terms of solving time, especially when the finite element model is large. For example, hundreds or thousands of finite element simulation runs in SSA is not uncommon. If one simulation run takes hours to complete, it is almost impractical to perform SSA at all with thousands or even hundreds of simulations.

To perform goal-drive optimizations or SSA in a finite element-based framework, it is always recommended to perform a DOE. From the DOE, a response surface is built within the design space of interest. After a response surface is created, all simulation runs of goal-drive optimization or SSA become function evaluations.

In a DOE, sampling points increase dramatically as the number of input parameters increases. For example, a total of 149 sampling points (finite element evaluations) are needed for 10 input variables using Central Composite Design with fractional factorial design. As the number of input variables increases, the analysis becomes more and more intractable. In this case, one would like to exclude unimportant input parameters from the DOE sampling to reduce unnecessary sampling points. A correlation matrix is a tool to help identify which input parameters are unimportant and therefore treated as deterministic parameters in SSA.

When to Use a Parameters Correlation

As you add more input parameters to your DOE, the increase in the number of design points can decrease the efficiency of the analysis process. In this case, you can focus on the most important inputs, while excluding inputs with a lesser effect on your intended design. Removing these less important parameters from the DOE reduces the generation of unnecessary sampling points.

Benefits of a Parameters Correlation

You use a **Parameters Correlation** system to:

- Determine which input parameters have the most (and the least) effect on your design.
- Identify the degree to which the relationship is linear or quadratic.

A **Parameters Correlation** system also provides a variety of charts to assist in your assessment of parametric effects. For more information, see [Working with Parameters Correlation Charts \(p. 75\)](#).

This section covers the following topics:

[Running a Parameters Correlation](#)

[Working with Correlation Results](#)

[Working with Parameters Correlation Charts](#)

Running a Parameters Correlation

To run a parameters correlation:

1. In the **Project Schematic**, drag a **Parameters Correlation** system from under **Design Exploration** in the **Toolbox** and drop it under the **Parameter Set** bar.
2. Double-click the **Parameters Correlation** cell to open the component tab.
3. Set correlation options.
4. If you want to review the samples to be calculated before generating them, in the **Outline** pane, right-click **Parameters Correlation** and select **Preview**.
5. To generate the samples, in the **Outline** pane, right-click **Parameters Correlation** and select **Update**.
6. When the update finishes, use the filtered results in the **Table** pane to find the most relevant inputs for a selected output. For more information, see [Reviewing Filtered Correlation Data \(p. 71\)](#).
7. Use the various charts in the **Outline** pane to examine the results. For more information, see [Using DesignXplorer Charts \(p. 32\)](#).

Setting Correlation Properties

The following correlation properties are available in the **Properties** pane:

Reuse the Samples Already Generated

Select this check box if you want to reuse the samples generated in a previous correlation.

Correlation Type

Select the **Spearman** or **Pearson** correlation type. The default value is determined by the **Correlation Coefficient Calculation Type** option in **Tools** → **Options**.

- **Spearman**: Select this option to correlate monotonic relationships.
- **Pearson**: Select this option to correlate linear relationships. This correlation method uses actual data to evaluate the correlation and bases correlation coefficients on sample values.

For more information on these correlation types, see [Sample Generation \(p. 69\)](#).

Number Of Samples

Specify the maximum number of samples to be generated in the correlation sample set. The default is **100**. This value must be greater than the number of enabled input parameters.

Auto Stop Type

If the **Auto Stop Type** property is set to:

- **Execute All Simulations**: DesignXplorer updates the number of design points specified by the **Number of Samples** property.

- **Enable Auto Stop:** The number of samples required to calculate the correlation is determined according to the convergence of the mean and standard deviation of the output parameters. At each iteration, the mean and standard deviation convergences are checked against the level of accuracy specified by the **Mean Value Accuracy** and **Standard Deviation Accuracy** properties. For more information, see [Correlation Convergence Process](#) (p. 70).

DesignXplorer attempts to minimize the number of design points to be updated by monitoring the evolution of the output parameter values and stops calculating design points as soon as the level of accuracy is met. This occurs when the mean and standard deviation are stable for all output parameters. If the process has converged, DesignXplorer stops, even if the number of calculated points specified by the **Number of Samples** property has not been reached.

Size of Generated Sample Set

Read-only. The number of samples to be generated in the correlation sample set.

Setting the Filtering Method for Correlation Parameters

The purpose of a correlation study is to identify the input parameters with the most effect on the output parameters. Not all output parameters have the same relevance in a design, however, so DesignXplorer provides filtering functionality for the correlation parameters. This functionality allows you to first specify the output parameters most relevant to your design and then filter the input parameters with regards to their effect on those outputs.

By default, all output parameters are taken into account for the filtering process. You can change these filtering options before updating the correlation study. You can also change the filtering options after the update has been completed. When you change options, a new design point update is not necessary. An **Update** operation updates only the display to show the results sorted according to your selected criteria.

Specifying the Filtering Output Parameters

By default, all output parameters are considered to be relevant and are included in the filtering process. You can specify the ones that have the most influence on your design as the output parameters that are included in the filtering process

To indicate that an output parameter is not to be considered in the filtering process, you can use either of these methods to select the **Ignore for Filtering** check box:

- In the **Outline** pane, select the output. Then, in the **Property** pane, select **Ignore for Filtering**.
- In the **Outline** pane, select one or more outputs, right-click, and select **Ignore for Filtering**.

To indicate that an output parameter is to be considered in the filtering process, in the **Outline** pane, select one or more outputs, right-click, and select **Use for Filtering**.

In the **Outline** pane, your filtering outputs are indicated by a funnel icon. When **Parameters Correlation** is selected in the **Outline** pane, your filtering parameters are also listed in the summary report in the **Table** pane.

Identifying Major and Minor Input Parameters

Once you have identified your filtering outputs, you can filter and sort your inputs with regards to their influence on the filtering outputs. Input parameters are categorized either as *major* inputs (with a greater influence on the outputs) or *minor* inputs (with a smaller influence on the outputs).

To specify your filtering criteria, select **Parameters Correlation** in the **Outline** pane. Then, in the **Properties** pane under **Filtering Method**, set properties:

- **Relevance Threshold:** Determines how strictly inputs are filtered for inclusion in the major input category. To estimate the relevance of parameter relationships, a metric is computed for any input-output pair. If one of the metrics for a given input parameter exceeds the value set for **Relevance Threshold**, that parameter is categorized as a major input. Otherwise, it is categorized as a minor input.

The default value is **0.5**, with possible values ranging from **0** to **1**. A value of **1** applies the strictest filter, and a value of **0** the most relaxed. For example, if there are 10 inputs and **Relevance Threshold** is set to **0**, all 10 of the outputs are categorized as major outputs. If **Relevance Threshold** is changed to **1**, a number of the outputs are filtered out and categorized instead as minor inputs.

For more information on how **Relevance Threshold** filters inputs, see [Parameters Correlation Filtering Theory \(p. 325\)](#).

- **Correlation Filtering:** This is based on Pearson's, Spearman's, and quadratic correlation. For each input-output pair, both correlation complexity and the number of samples are used to determine relevance. The greater the number of samples, the greater the probability that the correlation detected is valid. If the relevance is greater than the **Relevance Threshold** value, the input is categorized as a major input. This property is enabled by default.
- **R² Contribution Filtering:** Computes a reduced model and evaluates the R² contribution of each input parameter on the filtering output. The R² contribution is then used to calculate relevance. If the relevance is greater than the **Relevance Threshold** value, the input is categorized as a major input. This property is enabled by default.
- **Maximum Number of Major Inputs:** Maximum number of input parameters that can be categorized as major inputs.

The maximum value is equal to the number of enabled input parameters. The minimum value is **1**. The default is **20** if the number of input parameters is greater than or equal to 20. This is the recommended maximum number of inputs for a response surface.

Once you have set your filtering criteria, update the **Parameters Correlation** cell to filter the results. If design points have already been updated, they are not updated again. It only updates the results

according to the new filter. This allows you to change your filter settings and review multiple scenarios without having to run a new design point update each time.

Note:

When you select only one filtering method (**Correlation Filtering** or **R2 Contribution Filtering**), the relevance threshold can change the list of major input parameters but not their best relevance values.

When you select both filtering methods, **Correlation Filtering** is applied first, which retains a list of major input parameters based on the **Relevance Threshold** value. The **R2 Contribution Filtering** method is then applied, based on the list of major input parameters retained by the **Correlation Filtering** method. This can cause the best relevance values displayed in the tables of major and minor input parameters to change slightly when you change the **Relevance Threshold** value with both filtering methods enabled.

Sample Generation

Two methods of generating samples are available: **Pearson's** and **Spearman's**:

Pearson's Linear Correlation

- Uses actual data for correlation evaluation.
- Correlation coefficients are based on the sample values.

height (H)	Stress (S)
8.0357473	42.049729
7.9530498	41.707272
8.5795715	36.684773
8.2932448	37.968797
7.6854265	40.610932

- Used to correlate linear relationships.

Spearman's Rank Correlation

- Uses ranks of data.
- Bases correlation coefficients on the rank of samples.

height rank	Stress Rank
3	1
4	2
1	5
2	4
5	3

- Recognizes *monotonic relationships*, which are less restrictive than linear ones. In a monotonic relationship, one of the following two things happens:
 - As the value of one variable increases, the value of the other variable also increases.
 - As the value of one variable increases, the value of the other variable decreases.

The **Spearman's** method is considered to be more accurate.

Previewing, Monitoring, and Interrupting the Correlation

The **Preview** operation is available for a **Parameters Correlation** cell. It allows you to review the samples to be calculated based on selected options. Previewing allows you to try various sampling options and see the samples to be generated without actually generating the samples using the **Update** operation.

During the update of a **Parameters Correlation** cell, you can monitor progress in the **Progress** pane. The table of samples refreshes automatically as results are returned to DesignXplorer.

By clicking the red stop button to the right of the progress bar, you can interrupt the update. If enough samples are calculated, partial correlation results are generated. You can see the results in the **Table** pane of the component tab by selecting a chart object in the **Outline** pane.

To restart an interrupted update, you either right-click **Parameters Correlation** in the **Outline** pane and select **Update** or click **Update** on the toolbar. The update restarts where it was interrupted.

Correlation Convergence Process

Convergence of the correlation is determined as follows:

1. The convergence status is checked each time the number of points specified for **Convergence Check Frequency** has been updated.
2. For each output parameter:
 - The mean and the standard deviation are calculated based on all up-to-date design points available at this step.
 - The mean is compared with the mean at the previous step. It is considered to be stable if the difference is smaller than 1% by default (**Mean Value Accuracy = 0.01**).
 - The standard deviation is compared with the standard deviation at the previous step. It is considered to be stable if the difference is smaller than 2% by default (**Standard Deviation Accuracy = 0.02**).
3. If the mean and standard deviation are stable for all output parameters, the correlation is converged.

The convergence status is indicated by the **Converged** property. When the process is converged, **Converged** is set to **Yes** and the possible remaining unsolved samples are automatically removed. If the process has stopped because the value for **Number of Samples** is reached before convergence, **Converged** is set to **No**.

Working with Correlation Results

After the correlation study is complete, you can review the results:

- [Reviewing Filtered Correlation Data](#)
- [Viewing the Quadratic Correlation Information](#)
- [Viewing Significance and Correlation Values](#)
- [Viewing Correlation Design Points](#)
- [Editing Correlation Design Points](#)

Reviewing Filtered Correlation Data

When **Parameters Correlation** is selected in the **Outline** pane, the **Table** pane displays filtered results.

Under **Filtering Method**, you see the relevance threshold, configuration, and filtering output parameters used to filter the correlation data.

Under **Major Input Parameters** and **Minor Input Parameters**, you see inputs sorted in descending order according to their relevance to the output indicated. This output, shown in the **Output Parameter** column, is the output for which the input has the most relevance.

- The **R² Contribution** and **Correlation Value** values reference the relationship for that input-output pair.
- The **R² Contribution** is the difference between two R² coefficients of quadratic regressions (with and without this input). For more information, see [R² Contribution \(p. 327\)](#).
- **Correlation Value** corresponds to the most relevant correlation among Pearson, Spearman, or Quadratic correlations. For Pearson or Spearman, the value can have a (+/-) sign. For Quadratic, the value is positive. For more information, see [Relevance of the Correlation Value \(p. 325\)](#).

Table of Schematic C2: Parameters Correlation					
	A	B	C	D	E
1	Filtering Method				
2	Relevance Threshold	0.6			
3	Configuration	Filtering on Correlation Value and R ² Contribution, with a maximum of 7 major input parameters			
4	Filtering Output Parameters	P3 - MASS, P4 - DEFORMATION, P5 - STRESS, P6 - SINUS, P12 - DX_V, P13 - DX_SIG, P14 - DX_DIS, P15 - DX_BUCK			
5	Major Input Parameters				
6	Input Parameter	Best Relationship With Filtering Output Parameter			
7		Relevance	Output Parameter	R ² Contribution	Correlation Value
8	P2 - WB_RADIUS	1	P3 - MASS	0.99956	0.99978
9	P8 - DX_D	1	P13 - DX_SIG	0.5483	-0.75082
10	P9 - DX_L	1	P12 - DX_V	0.34004	0.57489
11	P7 - DX_B	0.98754	P12 - DX_V	0.32749	0.56761
12	P1 - WB_THICKNESS	0.85534	P4 - DEFORMATION	0.39531	-0.62205
13	P10 - DX_P	0.63194	P13 - DX_SIG	0.13837	0.38818
14	Minor Input Parameters				
15	Input Parameter	Best Relationship With Filtering Output Parameter			
16		Relevance	Output Parameter	R ² Contribution	Correlation Value
17	P11 - DX_E	0.49046	P15 - DX_BUCK	0.067866	0.29904

Viewing the Quadratic Correlation Information

Quadratic correlation between a parameter pair is proposed by its coefficient of determination (R²) of quadratic regression.

Note:

R² is shown as **R2** in DesignXplorer and correlation charts.

The closer R² is to 1, the better the quadratic regression.

Unlike the Correlation Matrix chart, the Determination Matrix chart is not symmetric. The Determination Matrix chart displays the R² for each parameter pair. To view this chart, select it in the **Outline** pane. Quadratic determination data is shown in both the **Table** and **Chart** panes.

In addition, quadratic information is shown in the Correlation Scatter chart and general parameters correlation table. The Correlation Scatter chart displays both the quadratic trend line and the linear trend line equation for the selected parameter pair. The general parameters correlation table shows quadratic data, linear data, and correlation design points.

Viewing Significance and Correlation Values

Significance of an input parameter to an output parameter is determined using a statistical hypothesis test. In the hypothesis test, a NULL hypothesis of insignificance is assumed, and is tested to see

whether it is statistically true according to the significance level (or acceptable risk) set by the user. From the hypothesis test, a p-value (probability value) is calculated, and is compared with the significance level. If the p-value is greater than the significance level, it is concluded that the NULL hypothesis is true, and that the input parameter is insignificant to the output parameter, and vice versa. For more information, see [Six Sigma Analysis \(SSA\) Theory \(p. 385\)](#).

If you select **Sensitivities** in the **Outline** pane for the **Six Sigma Analysis** cell, you can review the sensitivities derived from the samples generated for the correlation. The correlation sensitivities are global sensitivities. In the **Properties** pane for the Sensitivities chart, you can choose the output parameters for which you want to review sensitivities and the input parameters that you would like to evaluate for the output parameters.

On the **Design Exploration** tab of the **Options** dialog box, the default setting for **Significance Level** is **0.025**. Parameters with a sensitivity value above this significance are shown with a flat line on the Sensitivities chart. The value displayed for these parameters when you place the mouse cursor over them on the chart is **0**.

To view the actual correlation value of the insignificant parameter pair, you select **Correlation Matrix** in the **Outline** pane and then place the mouse cursor over the square for that pair in the matrix. Or, you can set **Significance Level** to **1** in the **Options** dialog box, which bypasses the significance test and displays all input parameters on the Sensitivities chart with their actual correlation values.

In addition, linear information is shown in the Correlation Scatter chart and the general parameters correlation table. The Correlation Scatter chart displays both the quadratic trend line and the linear trend line equation for the selected parameter pair. The general parameters correlation table shows quadratic data, linear data, and correlation design points.

Viewing Correlation Design Points


To view design points generated during the correlation, in the **Outline** pane, select **Design Points**. The **Table** pane displays a list of all the design points generated during the correlation process.

If you select one or more design points and right-click, the context menu offers many of the same options available for the project's design points table for the **Parameter Set** bar.

	A	B	C	D	E	F	G	H
1	Name ▾	P1 - x_1 ▾	P2 - x_2 ▾	P3 - x_3 ▾	P4 - x_4 ▾	P5 - x_5 ▾	P6 - f ▾	P7 - g ▾
2	1	5.8946	0.0983			5.4479	99.54	53.769
3	2	8.8291	4.5931			1.4692	76.944	103.82
4	3	6.085	1.4211			7.9086	300.39	57.757
5	4	7.8296	3.6717			5.0962	160.44	85.487
6	5	0.039055	2.9579			2.2936	19.902	2.8085
7	6	1.273	7.4118			0.90337	172.65	6.8175
8	7	4.1698	5.5332			1.2878	346.98	30.018
9	8	9.0592	9.1976			3.1195	57.207	107.84
10	9	2.8781	6.6919			3.13	75.042	21.107
11	10	3.6705	8.2896			9.5248	422.49	28.974
12	11	5.5967	5.4678	7.3869	5.9646	9.4275	307.9	51.592




Editing Correlation Design Points

The design points generated during the correlation are read-only by default. If you want to edit design points or add new ones, do one of the following in the **Parameters Correlation** cell to unlock the design points table:

- In the **Outline** pane, click the locked icon  to the right of the **Design Points** node.
- In the **Properties** pane, set **Sampling Type** from **Auto** (default) to **Custom**.

Changes in the Outline Pane

When you unlock the design points table, DesignXplorer makes these changes in the **Outline** pane:

- Changes the icon to the right of the **Design Points** node from an locked icon to an unlocked icon (). If you place the mouse cursor over the unlocked icon, the tooltip indicates that the sample set is customized.
- Displays a warning icon () in the **Message** column for the top **Parameters Correlation** node and the **Design Points** node. If you click the icon, the message indicates that the sampling is custom and that results might be inaccurate because the distribution of the design points might not be optimal.
- Displays an information icon () to the left of the **Parameters Correlation** node if there are not enough samples to update the correlation. If you click the icon, the message indicates that you must add more design points in the table or set **Sampling Type** back to **Auto**.

Changes in the Properties Pane

When you unlock the design points table, DesignXplorer makes these changes in the **Properties** pane:

- Sets **Sampling Type** from **Auto** to **Custom**
- Sets **Auto Stop Type** to **Execute All Simulations** and makes this property read-only.
- Hides **Number of Samples** because the read-only property **Size of Generated Sample Set** already displays the size of the sample set.

Changes in the Table Pane

When you unlock the design points table, DesignXplorer makes these changes in the **Tables** pane:

- Displays a row for inserting a new design point.
- Makes input parameter values editable. As you enter or edit a value, DesignXplorer validates the value against the ranges for the parameter.
- Allows you to set all output parameter values as editable if the custom correlation is not linked to a **Response Surface** cell. For more information, see [Editable Output Parameter Values \(p. 314\)](#).

For a response surface-based correlation, values for the output parameters are calculated by evaluating the response surface.

- Allows you to import design points from a CSV file, just like you do for a custom DOE. For more information, see [Importing Design Points from a CSV File \(p. 98\)](#).
 - For a standalone **Parameters Correlation** system, DesignXplorer imports values for input parameters. During [parsing and validation \(p. 99\)](#) of the design point data, DesignXplorer might ask whether you want to adjust parameter ranges. If values for output parameters exist in the CSV file, DesignXplorer also imports them.
 - For a response surface-based **Parameters Correlation** system, DesignXplorer imports values for input parameters if the design point is within the parametric space. DesignXplorer never imports values for output parameters.
- Allows you to copy all or selected design points from the **Parameter Set** into the custom correlation, just like you do for a custom DOE. For more information, see [Copying Design Points \(p. 99\)](#)
 - For a standalone **Parameters Correlation** system, DesignXplorer copies input parameter values. During [parsing and validation \(p. 99\)](#) of the design point data, DesignXplorer might ask whether you want to adjust parameter ranges. If the design points are up-to-date, DesignXplorer also copies the values for output parameters.
 - For a response surface-based **Parameters Correlation** system, DesignXplorer copies input parameter values if the design point is within the parametric space. DesignXplorer never copies values for output parameters.

Note:

- If you preview design points before unlocking the design points table, DesignXplorer keeps the design points from the preview. You can edit or delete these design points and add new ones.
 - If you edit values for output parameters, you can clear the edited values. DesignXplorer then marks these design points as out-of-date.
-

Working with Parameters Correlation Charts

A **Parameters Correlation** system provides five different charts for assessing parametric effects in your project: Correlation Matrix, Determination Matrix, Correlation Scatter, Sensitivities, and Determination Histogram.

When a **Parameters Correlation** system is updated, one instance of each chart is added to the project. In the **Outline** pane for the **Parameters Correlation** cell, you can select each object under **Charts** to view that chart in the **Charts** pane.

To add a new instance of a chart, double-click it in the **Toolbox**. The chart is added as the last entry under **Charts**.

To add a Correlation Scatter chart for a particular parameter combination, right-click the associated cell in the Correlation Matrix chart and select **Insert <x-axis> vs <y-axis> Correlation Scatter**.

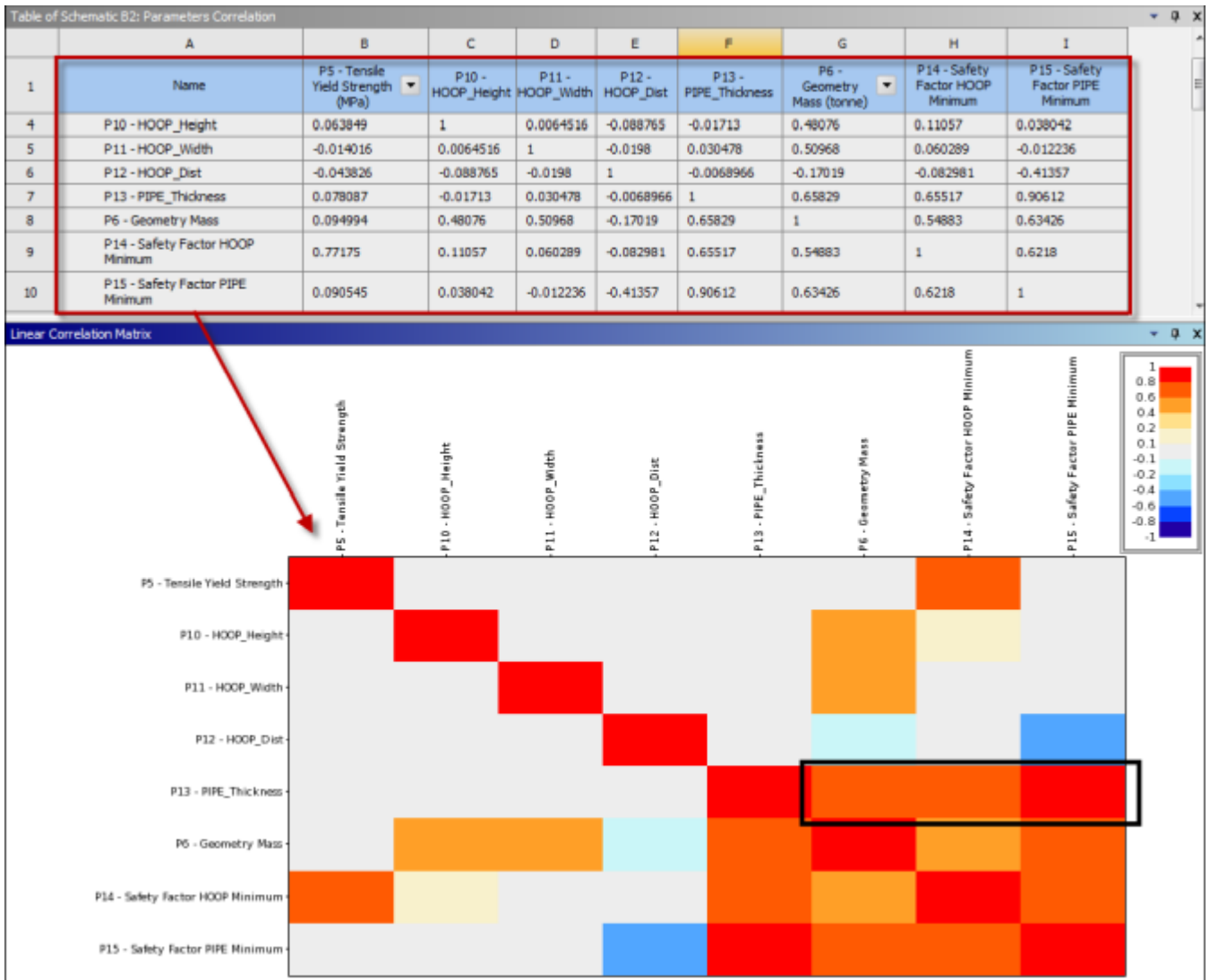
Using the Correlation Matrix Chart

The Correlation Matrix chart is a visual rendering of information in the parameters correlation table. The correlation coefficient indicates if there is a relationship between two variables and whether the relationship is a positive or negative number.

Color-coding of the cells indicates the strength of the correlation. The correlation value is displayed when you place the mouse cursor over a cell. The closer the absolute correlation value is to 1, the stronger the relationship. A value of 1 indicates a positive correlation, which means that when the first parameter increases, the second parameter increases as well. A value of -1 indicates a negative correlation, which means that when the first parameter increases, the second parameter decreases.

When you run a Pearson's correlation, the square of the correlation value corresponds to the R^2 of the linear fitting between the pair of parameters. When you run a Spearman's correlation, the correlation value corresponds to the R^2 of the linear fitting between rank values of the pair of parameters. For more information on these correlation types, see [Sample Generation \(p. 69\)](#).

In the following parameters correlation table, input parameter **P13-PIPE_Thickness** is a major input with a strong effect on the design.



On the other hand, input parameter **P12-Hoop Dist** is not important to the study because it has little effect on the outputs. In this case, you might want to disable **P12-Hoop Dist** by clearing the **Enabled** check box in the **Properties** pane. When the input is disabled, the chart changes accordingly.

To disable parameters, you can also right-click a cell corresponding to that parameter and select the desired option from the context menu. You can disable the selected input, disable the selected output, disable all other inputs, or disable all other outputs.

To generate a Correlation Scatter chart for a given parameter combination, right-click the corresponding cell in the correlation matrix chart and select **Insert <x-axis> vs <y-axis> Correlation Scatter**.

If desired, you can export the correlation matrix data to a CSV file by selecting the **Export Chart Data as CSV** context option. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) in the *Workbench User's Guide*.

Using the Correlation Scatter Chart

The Correlation Scatter chart enables you to plot linear and quadratic trend lines for the samples and extract the linear and quadratic coefficient of determination (R^2). This means that the chart conveys

the degree of quadratic correlation between two parameters pair via a graphical presentation of linear and quadratic trends.

Note:

R^2 is shown as **R2** in DesignXplorer and its correlation charts.

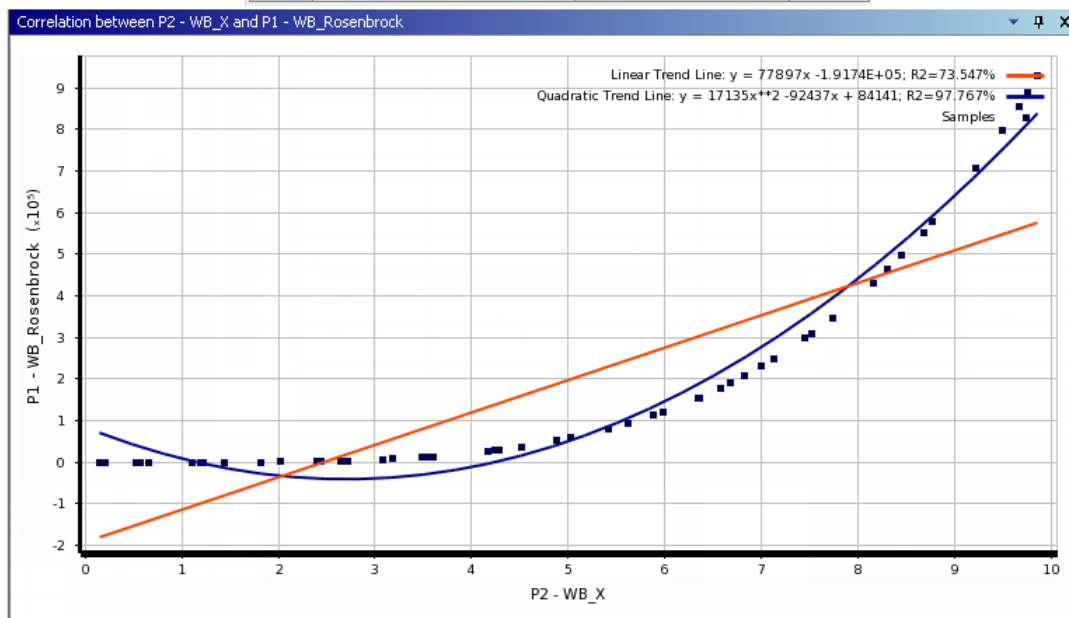
You can create a Correlation Scatter chart for a given parameter combination by right-clicking the corresponding cell in the correlation matrix chart and selecting **Insert <x-axis> vs <y-axis> Correlation Scatter**.

In this example, under **Trend Lines** in the **Properties** pane, **Linear** and **Quadratic** display equations for R^2 values.

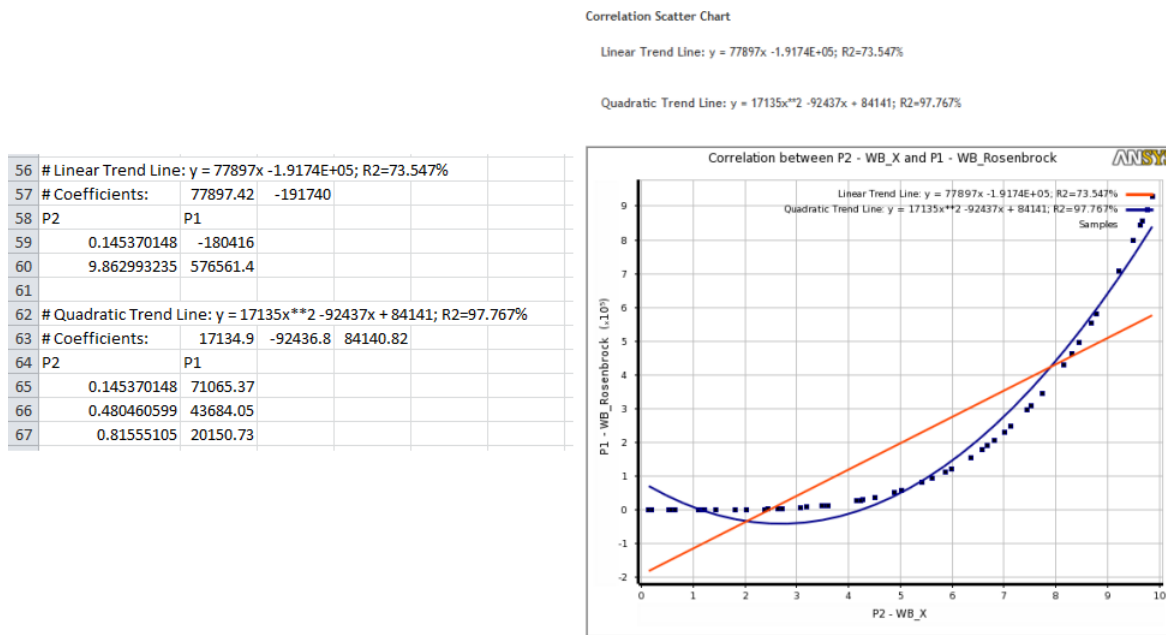
Because both the **Linear** and **Quadratic** properties are enabled in this example:

- The equations for the linear and quadratic trend lines are shown in the chart legend.
- The linear and quadratic trend lines are each represented by a separate line on the chart. The closer the samples lie to the curve, the closer the coefficient of determination is to the optimum value of **1**.

Properties of Outline A15: Correlation Scatter			
	A	B	C
1	Property	Value	Enabled
2	Chart		
3	Display Parameter Full Name	<input checked="" type="checkbox"/>	
4	Axes		
5	X Axis	P2 - WB_X	
6	Y Axis	P1 - WB_Rosenbrock	
7	Trend Lines		
8	Linear	$y = 77897x - 1.9174E+05$; R2=73.547%	<input checked="" type="checkbox"/>
9	Quadratic	$y = 17135x^{*2} - 92437x + 84141$; R2=97.767%	<input checked="" type="checkbox"/>



When you export the Correlation Scatter chart data to a CSV file or generate a report, the trend line equations are included in the export and are shown in the CSV file or Workbench project report. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) in the *Workbench User's Guide*.



Using the Determination Matrix Chart

The Determination Matrix chart is a visual rendering of the nonlinear (quadratic) information in the determination matrix. It shows the coefficient of determination (R^2) between all parameter pairs. Unlike the Correlation Matrix chart, the Determination Matrix chart is not symmetric.

Color-coding of the cells indicates the strength of the correlation (R^2). The R^2 value is displayed when you place the mouse cursor over a cell. The closer the R^2 value is to 1, the stronger the relationship.

In the following determination matrix, input parameter **P5–Tensile Yield Strength** is a major input because it drives all the outputs.

You can disable inputs that have little effect on the outputs. To disable a parameter in the chart:

- In the **Properties** pane, clear the **Enabled** check box for the parameter.
- Right-click a cell corresponding to the parameter and select an option from the context menu. You can disable the selected input, disable the selected output, disable all other inputs, or disable all other outputs.

You can also select the **Export Chart Data as CSV** context option to export the correlation matrix data to a CSV file. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) in the *Workbench User's Guide*.

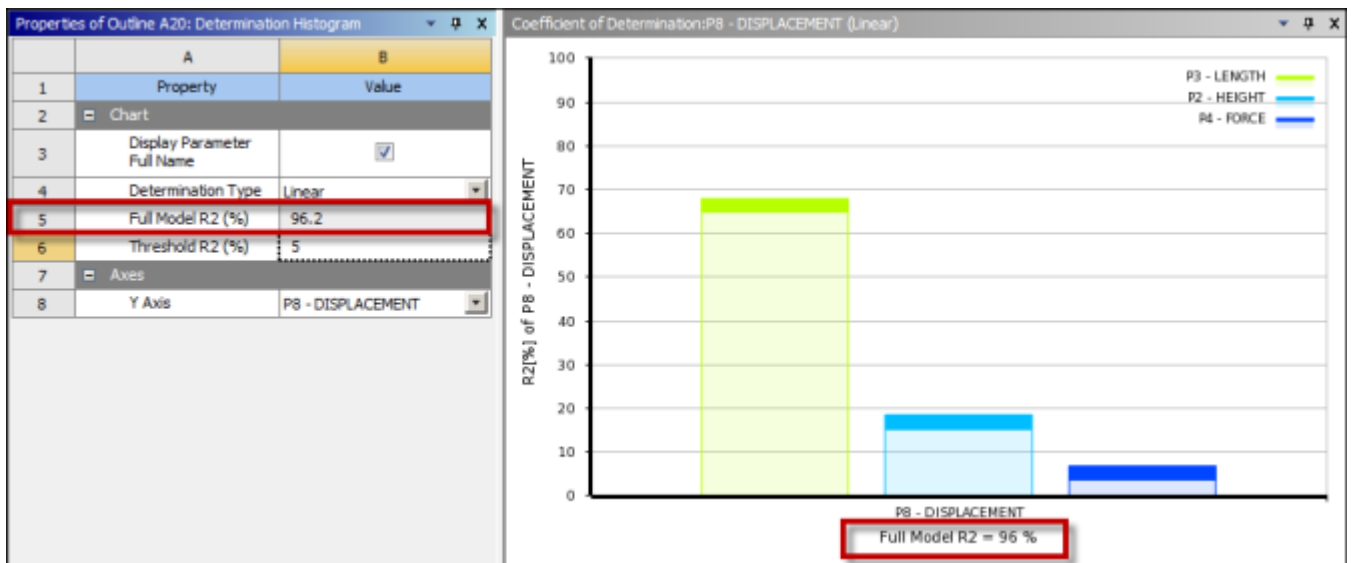


Using the Determination Histogram Chart

The Determination Histogram chart allows you to see what inputs drive a selected output parameter. You can set the **Determination Type** property to **Linear** or **Quadratic**. The **Threshold R2 (%)** property allows you to filter the input parameters by hiding the input parameters with a determination coefficient lower than the given threshold.

When you view a Determination Histogram chart, you should also check the **Full Model R2 (%)** value to see how well output variations are explained by input variations. This value represents the variability of the output parameter that can be explained by a linear or quadratic correlation between the input parameters and the output parameter. The closer this value is to **100%**, the more certain it is that output variations result from the inputs. The lower the value, the more likely it is that other factors such as noise, mesh error, or an insufficient number of points is causing the output variations.

In the following figure, you can see that input parameters **P3-LENGTH**, **P2-HEIGHT**, and **P4-FORCE** all affect output **P8-DISPLACEMENT**. You can also see that of the three inputs, **P3-LENGTH** has by far the greatest effect. The value for the linear determination is **96.2%**.

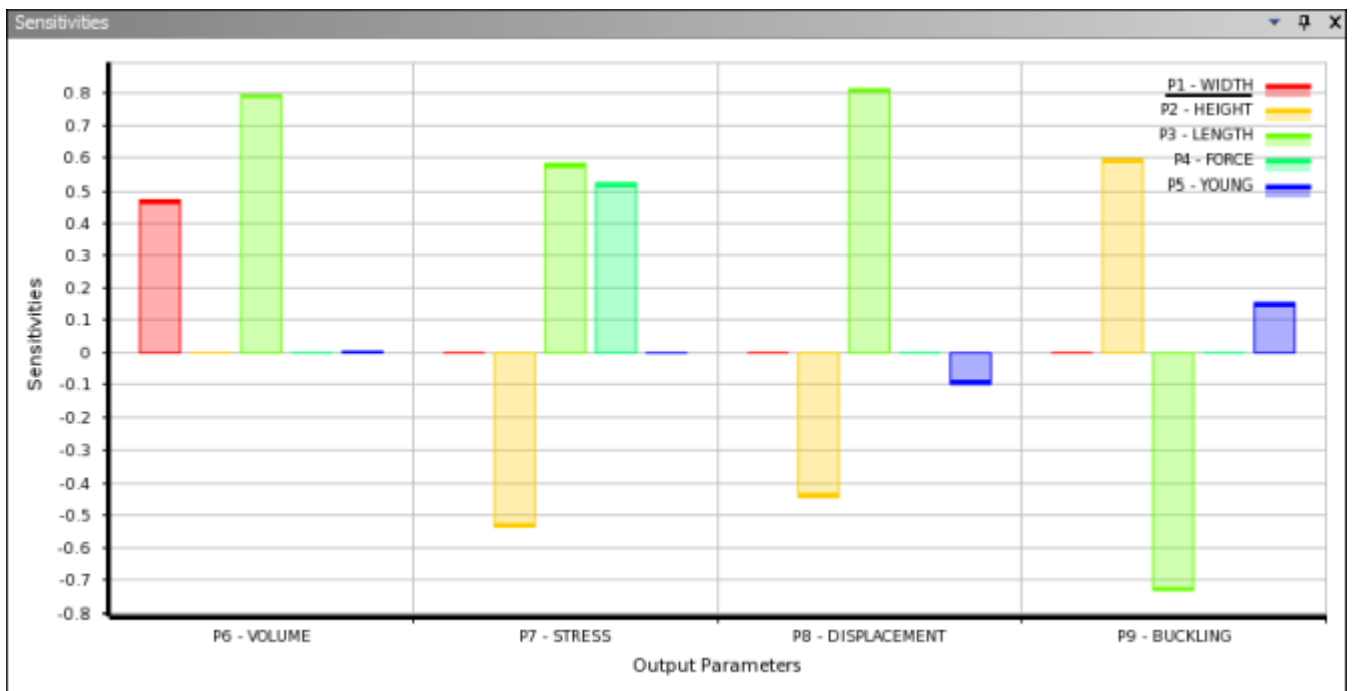


To view the chart for a quadratic determination, in the **Properties** pane, set **Determination Type** to **Quadratic**. With a quadratic determination type, input **P5-YOUNG** could also have a slight effect on **P8-DISPLACEMENT**. For example, **Full Model R2 (%)** could improved slightly, perhaps to **97.436%**. You can filter your inputs to keep only the most important parameters by selecting and clearing their check boxes in the **Outline** pane.

Using the Sensitivities Chart

The Sensitivities chart shows global sensitivities of the output parameters with respect to the input parameters. Positive sensitivity occurs when increasing the input increases the output. Negative sensitivity occurs when increasing the input decreases the output.

You can display the Sensitivities chart as a bar or pie chart.



Generally, the effect of an input parameter on an output parameter is driven by the following two things:

- The amount by which the output parameter varies across the variation range of an input parameter.
- The variation range of an input parameter. Typically, the wider the variation range, the larger the effect of the input parameter.

The statistical sensitivities are based on the Spearman rank order correlation coefficients, which simultaneously take both aspects into account.

Using Design of Experiments

Design of Experiments (DOE) is a technique used to scientifically determine the location of sampling points. It is included as part of response surface, goal-driven optimization, and other analysis systems.

There are a wide range of DOE algorithms or methods available in engineering literature. However, they all have common characteristics. They try to locate sampling points such that the space of random input parameters is explored in the most efficient way, or they try to obtain the required information with a minimum of sampling points.

Sample points in efficient locations not only reduce the required number of sampling points but also increase the accuracy of the response surface that is derived from the results of the sampling points. By default, the deterministic method uses Central Composite Design, which combines one center point, points along the axis of the input parameters, and the points determined by a fractional factorial design. For more information, see [DOE Types \(p. 84\)](#).

Once you set up your input parameters, you update the DOE, which submits the generated design points to the analysis system to determine a solution. Design points are solved simultaneously if the analysis system is set up to do so. Otherwise, design points are solved sequentially. After the solution is complete, you update the **Response Surface** cell, which generates response surfaces for each output parameter based on the data in the generated design points.

Note:

Requirements and recommendations regarding the number of input parameters depend on the DOE type. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).

If you change the DOE type after doing an initial analysis and preview the design points table, any design points generated for the new algorithm that are the same as design points solved for a previous algorithm appear as up-to-date. Only the design points that are different from any previously submitted design points need to be solved.

You should set properties for your DOE before generating your design points table. The following topics describe setting up and solving a DOE:

[Setting Up the DOE](#)

[DOE Types](#)

[Number of Input Parameters for DOE Types](#)

[Comparison of LHS and OSF DOE Types](#)

[Using a Central Composite Design DOE](#)

[Upper and Lower Locations of DOE Points](#)

[DOE Matrix Generation](#)

[Exporting and Importing Design Points](#)

Copying Design Points

Setting Up the DOE

To set up your DOE:

1. In the **Project Schematic**, double-click the **Design of Experiments** cell.
2. In the **Outline** pane for the DOE, select **Design of Experiments**.
3. In the **Properties** pane under **Design of Experiments**, make a selection for **Design of Experiments Type**. For descriptions, see [DOE Types \(p. 84\)](#).
4. Specify additional properties for the DOE. The selection for **Design of Experiments Type** determines what properties are available.
5. Update the DOE.

DOE Types

In the **Properties** pane for the **Design of Experiments** cell, **Design of Experiments Type** specifies the algorithm or method for locating sampling points. DesignXplorer supports several DOE types:

[Central Composite Design \(CCD\)](#)

[Optimal Space-Filling Design \(OSF\)](#)

[Box-Behnken Design](#)

[Custom](#)

[Custom + Sampling](#)

[Sparse Grid Initialization](#)

[Latin Hypercube Sampling Design](#)

[External Design of Experiments](#)

Central Composite Design (CCD)

When **Design of Experiments Type** is set to **Central Composite Design** (default), a screening set is used to determine the overall trends of the model to better guide the choice of options for **Optimal Space-Filling Design**. CCD supports a maximum of 20 input parameters. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).

The following properties are available for CCD.

- **Design Type:** Design type to use for CCD to improve the response surface fit for the DOE. For each design type, the alpha value is defined as the location of the sampling point that accounts for all quadratic main effects. The following CCD design types are available:
 - **Face-Centered:** A three-level design with no rotatability. The alpha value equals 1.0. A **Template Type** setting automatically appears.

- **Rotatable:** A five-level design that includes rotatability. The alpha value is calculated based on the number of input variables and a fraction of the factorial part. A design with rotatability has the same variance of the fitted value regardless of the direction from the center point. A **Template Type** setting automatically appears.
- **VIF-Optimality:** A five-level design in which the alpha value is calculated by minimizing a measure of non-orthogonality known as the Variance Inflation Factor (VIF). The more highly correlated the input variable with one or more terms in a regression model, the higher the VIF.
- **G-Optimality:** Minimizes a measure of the expected error in a prediction and minimizes the largest expected variance of prediction over the region of interest.
- **Auto-Defined:** Design exploration automatically selects the design type based on the number of input variables. Use of this option is recommended for most cases because it automatically switches between **G-Optimality** if the number of input variables is five or **VIF-Optimality** otherwise. However, you can select **Rotatable** as the design type if the default option does not provide good values for the goodness of fit from the response surface plots.

For more information, see [Using a Central Composite Design DOE \(p. 94\)](#)

- **Template Type:** Enabled when **Design Type** is set to either **Face-Centered** or **Rotatable**. Choices are **Standard** (default) and **Enhanced**. Choose **Enhanced** for a possible better fit for the response surfaces.

For more information, see [Using a Central Composite Design DOE \(p. 94\)](#).

Optimal Space-Filling Design (OSF)

When **Design of Experiments Type** is set to **Optimal Space-Filling Design**, an OSF DOE is created according to some specified criteria. Essentially, OSF is a Latin Hypercube Sampling (LHS) that is extended with post-processing. It is initialized as LHS and then optimized several times, remaining a valid LHS (without points sharing rows or columns) while achieving a more uniform space distribution of points (maximizing the distance between points). For more information, see [Comparison of LHS and OSF DOE Types \(p. 93\)](#).

To offset the noise associated with physical experimentation, classic DOE types such as CCD focus on parameter settings near the perimeter of the design region. Because computer simulation is not quite as subject to noise, OSF is able to distribute the design parameters equally throughout the design space with the objective of gaining the maximum insight into the design with the fewest number of points. This advantage makes it appropriate when a more complex modeling technique such as Kriging, non-parametric regression, or neural networks is used.

OSF has some of the same disadvantages as LHS, though to a lesser degree. Possible disadvantages follow.

- When **Samples Type** is set to **CCD Samples**, a maximum of 20 input parameters is supported. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).
- Extremes, such as the corners of the design space, are not necessarily covered.
- The selection of too few design points can result in a lower quality of response prediction.

The following properties are available for OSF.

- **Design Type:** Choices are:
 - **Max-Min Distance:** Maximizes the minimum distance between any two points (default). This strategy ensures that no two points are too close to each other. For a small size of sampling (N), the Max-Min Distance design generally lies on the exterior of the design space and fills in the interior as N becomes larger. Generally, this is the faster algorithm.
 - **Centered L2:** Minimizes the centered L2-discrepancy measure. The discrepancy measure corresponds to the difference between the empirical distribution of the sampling points and the uniform distribution. This means that the centered L2 yields a uniform sampling. This design type is computationally faster than the **Maximum Entropy** type.
 - **Maximum Entropy:** Maximizes the determinant of the covariance matrix of the sampling points to minimize uncertainty in unobserved locations. This option often provides better results for highly correlated design spaces. However, its cost increases non-linearly with the number of input parameters and the number of samples to be generated. Thus, it is recommended only for small parametric problems.
- **Maximum Number of Cycles:** Determines the number of optimization loops the algorithm needs, which in turns determines the discrepancy of the DOE. The optimization is essentially combinatorial, so a large number of cycles slows down the process. However, this makes the discrepancy of the DOE smaller. For practical purposes, 10 cycles is generally good for up to 20 variables. The value must be greater than 0. The default is **10**.
- **Samples Type:** Determines the number of DOE points the algorithm should generate. This option is suggested if you have some advanced knowledge about the nature of the model. Choices are:
 - **CCD Samples:** Supports a maximum of 20 inputs (default). Generates the same number of samples a CCD DOE would generate for the same number of inputs. You can use this to generate a space filling design that has the same cost as a corresponding CCD design.
 - **Linear Model Samples:** Generates the number of samples as needed for a linear model.
 - **Pure Quadratic Model Samples:** Generates the number of samples as needed for a pure quadratic model (no cross terms).
 - **Full Quadratic Samples:** Generates the number of samples needed to generate a full quadratic model.
 - **User-Defined Samples:** Specify the desired number of samples.
- **Random Generator Seed:** Enabled when LHS is used. Set the value used to initialize the random number generator invoked internally by LHS. Although the generation of a starting point is random, the seed value consistently results in a specific LHS. This property allows you to generate different samplings by changing the value or regenerate the same sampling by keeping the same value. The default is **0**.
- **Number of Samples:** Enabled when **Samples Type** is set to **User-Defined Samples**. Specifies the default number of samples. The default is **10**.

Box-Behnken Design

When **Design of Experiments Type** is set to **Box-Behnken Design**, a three-level quadratic design is generated. This design does not contain fractional factorial design. The sample combinations are treated in such a way that they are located at midpoints of edges formed by any two factors. The design is rotatable (or in cases, nearly rotatable).

One advantage of Box-Behnken Design is that it requires fewer design points than a full factorial CCD and generally requires fewer design points than a fractional factorial CCD. Additionally, Box-Behnken Design avoids extremes, allowing you to work around extreme factor combinations. Consider using Box-Behnken Design if your project has parametric extremes (for example, has extreme parameter values in corners that are difficult to build). Because a DOE based on Box-Behnken Design doesn't have corners and does not combine parametric extremes, it can reduce the risk of update failures. For more information, see the [Box-Behnken Design \(p. 332\)](#) theory section.

Possible disadvantages of Box-Behnken Design follow.

- Prediction at the corners of the design space is poor and that there are only three levels per parameter.
- A maximum of 12 input parameters is supported. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).

No additional properties are available for Box-Behnken Design.

Custom

When **Design of Experiments Type** is set to **Custom**, you can add points directly in the design points table by manually entering input parameters and optionally output parameter values. You can also [import and export \(p. 98\)](#) design points into the design points table of the custom DOE from the **Parameter Set** bar.

You can change the mode of the design points table so that output parameter values are editable. You can also copy and paste data and import data from a CSV file by right-clicking and selecting **Import Design Points**. For more information, see [Working with Tables \(p. 314\)](#).

Note:

- If you generate design points using a DOE type other than **Custom** and then later switch to **Custom**, all points existing in the design points table from the initial DOE type are retained.
- If you set the DOE type to **Custom** and add points directly to the design points table, these manually added design points are cleared if you later switch to another DOE type.
- The table can contain derived parameters. Derived parameters are always calculated by the system, even if the table mode is **All Output Values Editable**.
- Editing output values for a row changes the state of the **Design of Experiments** cell to **Update Required**. The DOE must be updated, even though no calculations are done.

- DOE charts do not reflect the design points added manually using the **Custom** DOE type until the DOE is updated.
- It is expected that the **Custom** DOE type is used to enter DOEs that were built externally. If you use this feature to manually enter all design points, you must make sure to enter enough points so that a good fitting can be created for the response surface. This is an advanced feature that should be used with caution. Always verify your results with a direct solve.

No additional properties are available for a custom DOE type.

Custom + Sampling

When **Design of Experiments Type** is set to **Custom + Sampling**, you have the same capabilities as when it is set to **Custom** (p. 87). You can complete the design points table automatically to fill the design space efficiently. For example, you can initialize the design points table with design points imported from a previous study. Or, your initial DOE (Central Composite Design, Optimal Space Filling Design or Custom type) can be completed with new points that you manually add. The generation of these new design points takes into account the coordinates of previous design points.

When **Custom + Sampling** is set, **Total Number of Samples** specifies the number of samples that you want, including the number of existing design points. You must enter a positive number. If the total number of samples is less than the number of existing points, no new points are added. If there are discrete input parameters, the total number of samples corresponds to the number of points that should be reached for each combination of discrete parameters.

Sparse Grid Initialization

When **Design of Experiments Type** is set to **Sparse Grid Initialization**, the adaptive model that runs is driven by the accuracy that you request. This DOE type increases the accuracy of the response surface by automatically refining the matrix of design points in locations where the relative error of the output parameter is higher. It generates the levels 0 and 1 of the Clenshaw-Curtis Grid. Because Sparse Grid Initialization is based on a hierarchy of grids, the DOE generated contains all of the design points for the smallest required grid: the level 0 (the point at the current values) plus the level 1 (two points per input parameters).

One advantage of Sparse Grid Initialization is that it refines only in the directions necessary, so that fewer design points are needed for the same quality response surface. Another is that it is effective at handling discontinuities. Although you must use this DOE type to build a Sparse Grid response surface, you can also use it for other types of response surfaces.

No additional properties are available for Sparse Grid Initialization.

Latin Hypercube Sampling Design

When **Design of Experiments Type** is set to **Latin Hypercube Sampling Design**, Latin Hypercube Sampling (LHS) generates the DOE. This advanced form of the Monte Carlo sampling method avoids clustering samples. In LHS, the points are randomly generated in a square grid across the design

space, but no two points share the same value. This means that no point shares a row or a column of the grid with any other point.

Possible disadvantages of LHS follow:

- When **Samples Type** is set to **CCD Samples**, a maximum of 20 input parameters is supported. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).
- Extremes, such as the corners of the design space, are not necessarily covered. Additionally, the selection of too few design points can result in a lower quality of response prediction.

Note:

The Optimal Space-Filling Design (OSF) DOE type is an LHS design that is extended with post-processing. For more information, see [Comparison of LHS and OSF DOE Types \(p. 93\)](#).

The following properties are available for LHS:

- **Samples Type:** Determines the number of DOE points the algorithm should generate. This option is suggested if you have some advanced knowledge about the nature of the model. The following choices are available:
 - **CCD Samples:** Supports a maximum of 20 inputs (default). Generates the same number of samples a CCD DOE would generate for the same number of inputs. You can use this to generate an LHS design that has the same cost as a corresponding CCD design.
 - **Linear Model Samples:** Generates the number of samples as needed for a linear model.
 - **Pure Quadratic Model Samples:** Generates the number of samples as needed for a pure quadratic model (no cross terms).
 - **Full Quadratic Samples:** Generates the number of samples needed to generate a full quadratic model.
 - **User-Defined Samples:** Specify the desired number of samples.
- **Random Generator Seed:** Enabled when LHS is used. Set the value used to initialize the random number generator invoked internally by LHS. Although the generation of a starting point is random, the seed value consistently results in a specific LHS. This property allows you to generate different samplings by changing the value or regenerate the same sampling by keeping the same value. The default is **0**.
- **Number of Samples:** Enabled when **Samples Type** is set to **User-Defined Samples**. Specifies the default number of samples. The default is **10**.

External Design of Experiments

In addition to the DOE types provided by DesignXplorer, you can use DOE extensions, integrating the features of an external sampling method into your design exploration workflow. A DOE extension specifies the properties for one or more DOEs.

Selecting an External Sampling Method

Once a DOE extension has been installed and loaded to your project as described in [Working with DesignXplorer Extensions](#) (p. 320), the external DOEs defined in it are available for selection in DesignXplorer. In the **Properties** pane for the **Design of Experiments** cell, external DOEs are included in the choices for **Design of Experiments Type**, along with the DOE types provided by DesignXplorer.

DesignXplorer filters the **Design of Experiments Type** list for applicability to the current project, displaying only those sampling methods that you can use to generate the DOE as it is currently defined. For example, assume that a given sampling allows a maximum of 10 input parameters. As soon as more than 10 inputs are defined, that sampling method is removed from the list.

Setting Up an External DOE

After you select the external DOE, you can edit its properties. DesignXplorer shows only the functionality that is specifically defined in the extension. Once you've defined the properties, the process of generating, previewing, and reviewing sampling results is the same as for the native DOE types.

Number of Input Parameters for DOE Types

The number of enabled input parameters has an effect on the generation of a DOE and response surface. The more inputs that are enabled, the longer it takes to generate the DOE and update corresponding design points. Additionally, having a large number of inputs makes it more difficult to generate an accurate response surface. The quality of a response surface is dependent on the relationships that exist between input and output parameters, so having fewer enabled inputs makes it easier to determine how they affect the outputs.

As such, the recommendation for most DOE types is to have as few enabled input as possible. Fewer than 20 inputs is ideal. Some DOE types have a limit on the number of inputs:

- CCD has a limit of 20 inputs.
- Box-Behnken Design has a limit of 12 inputs.
- LHS and OSF have a limit of 20 inputs when **Samples Type** is set to **CCD Samples**












The number of inputs should be taken into account when selecting a DOE type for your study—or when defining inputs if you know ahead of time which DOE type you intend to use.

If you are using a DOE that does not limit the number of inputs and more than 20 are enabled, in the **Outline** pane for the following cells, DesignXplorer shows an alert icon in the **Message** column for the root node:

- **Design of Experiments**
- **Response Surface**
- **Optimization** (for an Adaptive Single-Objective or Adaptive Multiple-Objective optimization)

The warning icon is displayed only when required component edits are completed. The number next to the icon indicates the number of active warnings. You can click the icon to review the warning messages. To remove a warning about having too many inputs defined, disable inputs by clearing check

boxes in the **Enabled** column until only the permitted number of inputs is selected. If you are unsure of which parameters to disable, you can use a **Parameters Correlation** system to determine the inputs that are least correlated with your results. For more information, see [Using Parameters Correlations \(p. 65\)](#).

Outline of Schematic B2: Design of Experiments			
	A	B	C
1		Enabled	Message
2	 Design of Experiments 		1 
3	 Input Parameters		
4	 Mechanical APDL (A1)		
5	 P1 - X1	<input checked="" type="checkbox"/>	
6	 P2 - X2	<input checked="" type="checkbox"/>	
7	 P3 - X3	<input checked="" type="checkbox"/>	
8	 P4 - X4	<input checked="" type="checkbox"/>	
9	 P5 - X5	<input checked="" type="checkbox"/>	
10	 P6 - X6	<input checked="" type="checkbox"/>	

Factors other than the number of enabled inputs affect response surface generation:

- Response surface types using automatic refinement add additional design points to improve the resolution of each output. The more outputs and the more complicated the relationship between the inputs and outputs, the more design points that are required.
- Increasing the number of output parameters increases the number of response surfaces that are required.
- Discrete input parameters can be expensive because a response surface is generated for each discrete combination, as well as for each output parameter.
- A non-linear or non-polynomial relationship between input and output parameters requires more design points to build an accurate response surface, even with a small number enabled inputs.

Such factors can offset the importance of using a small number of enabled input parameters. If you expect that a response surface can be generated with relative ease, it might be worthwhile to exceed the recommended number of inputs. For example, assume that the project has polynomial relationships between the inputs and outputs, only continuous inputs, and a small number of outputs. In this case, you might ignore the warning and proceed with the update.

If the DOE-response surface combination is not supported, quick help explains the underlying problem. The following example explains that the number of design points is less than the number of enabled input parameters. When you enable discrete input parameters, the number of required design points is affected. For each enabled discrete input parameter combination, the number of design points must be equal to or greater than the number of enabled continuous input parameters.

The Design of Experiments (DOE) type is not **CCD** and the response surface type is not **Standard Response Surface - Full 2nd Order Polynomials**, but there are fewer design points than enabled input parameters. The current DOE-response surface combination requires that the number of design points (per enabled discrete input parameter combination if at least one discrete parameter exists) is equal to or greater than the number of enabled continuous input parameters. You can change the DOE and response surface types, disable some input parameters, or add points to the response surface. The following bullets reference topics in the DesignXplorer User's Guide.

- For input parameter information for DOE types, see [Number of Input Parameters for DOE Types](#). This topic includes an explanation of when this particular quick help displays.
- To change the DOE type or disable input parameters, right-click the **DOE** cell and select **Edit**. See [Setting Up the DOE](#) or [Working with Parameters](#).
- To change the response surface type, right-click the **Response Surface** cell and select **Edit**. See [Response Surface Types](#).
- For information on how to add design points, see [Working with Design Points](#).

[ANSYS Workbench](#)

Two examples show how the minimum number for design points is determined when the number of enabled discrete parameters is first 0 and then 2.

Example 1

- Number of enabled continuous input parameters = 3 (x1, x2, and x3)
 - $1.1 < x1 < 1.9$
 - $2.1 < x2 < 2.9$
 - $3.1 < x3 < 3.9$
- Number of enabled discrete parameters = 0
- DOE must contain at least 3 points:
 - 1.1; 2.1; 3.1
 - 1.5; 2.3; 3.4
 - 1.6; 2.6; 3.2

Example 2

- Number of enabled continuous input parameters = 3 (x1, x2, and x3)
 - $1.1 < x1 < 1.9$

- 2.1 <x2 < 2.9
- 3.1 <x3 < 3.9
- Number of enabled discrete parameters = 2
 - Where the 1st parameter has 3 levels (for example, 10;15;20)
 - Where the 2nd discrete parameter has 2 levels (for example, 5;6)
 - Then the number of discrete combinations = $3*2 = 6$
 - {10; 5}, {15; 5}, {20; 5}, {10; 6}, {15; 6}, {20; 6}
- DOE must contain at least 3 points for each discrete combination:

1.1; 2.1; 3.1;	10; 5
1.5; 2.3; 3.4;	10; 5
1.6; 2.6; 3.2;	10; 5
1.2; 2.2; 3.2;	15; 5
1.4; 2.4; 3.5;	15; 5
1.7; 2.5; 3.8;	15; 5
1.3; 2.6; 3.3;	20; 5
1.5; 2.9; 3.9;	20; 5
1.2; 2.1; 3.2;	20; 5
1.1; 2.5; 3.7;	10; 6
1.4; 2.4; 3.4;	10; 6
1.8; 2.1; 3.1;	10; 6
1.6; 2.8; 3.8;	15; 6
1.3; 2.3; 3.2;	15; 6
1.2; 2.6; 3.5;	15; 6
1.9; 2.1; 3.1;	20; 6
1.2; 2.7; 3.6;	20; 6
1.7; 2.5; 3.9;	20; 6

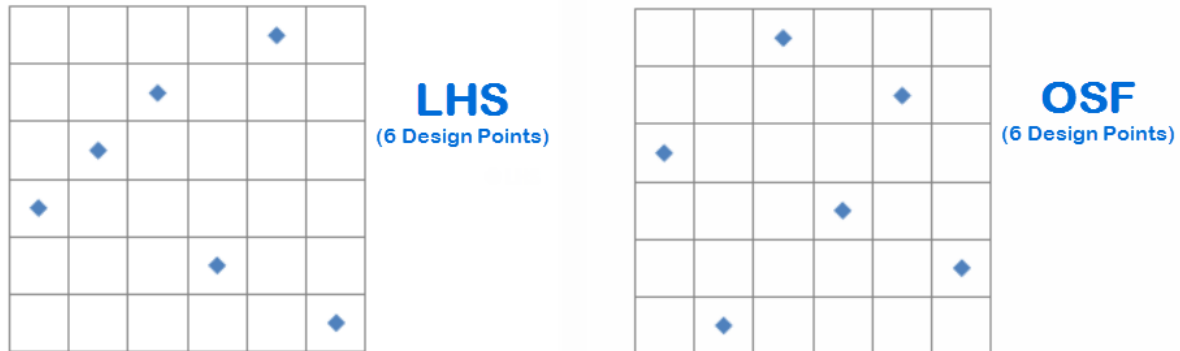
Comparison of LHS and OSF DOE Types

Latin Hypercube Sampling (LHS) and Optimal Space-Filling (OSF) designs both create DOEs according to specified criteria.

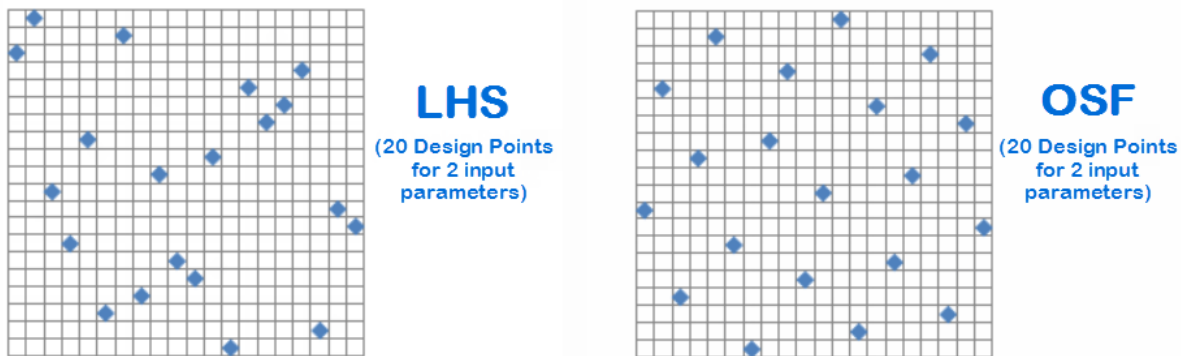
- LHS is an advanced form of the Monte Carlo sampling method. In LHS, no point shares a row or column of the design space with any other point.
- OSF is essentially an LHS design that is optimized through several iterations, maximizing the distance between points to achieve a more uniform distribution across the design space. Because it aims to gain the maximum insight into the design by using the fewest number of points, it is an effective DOE type for complex modeling techniques that use relatively large numbers of design points.

Because OSF incorporates LHS, both DOE types aim to conserve optimization resources by avoiding the creation of duplicate points. Given an adequate number of design points to work with, both methods result in a high quality of response prediction. OSF, however, offers the added benefit of fuller coverage of the design space.

For example, with a two-dimensional problem that has only two input parameters and uses only six design points, it can be difficult to build an adequate response surface. This is especially true in the case of LHS because of its nonuniform distribution of design points over the design space.



When the number of design points for the same scenario is increased to twenty, the quality of the resulting response surface is improved. LHS, however, can result in close, uneven groupings of design points and so can skip parts of the design space. OSF, with its maximization of the distance between points and more uniform distribution of points, addresses extremes more effectively and provides far better coverage of the design space. For this reason, OSF is the recommended method.



Using a Central Composite Design DOE

In Central Composite Design (CCD), you specify a design type. Choices are:

- **Face-Centered**
- **Rotatable**
- **VIF-Optimality**
- **Optimality**

- **Auto Defined**

A **Rotatable** (spherical) design is preferred because the prediction variance is the same for any two locations that are the same distance from the design center. However, there are other criteria to consider for an optimal design setup. The following two criteria are commonly considered in setting up an optimal design using the design matrix.

- The degree of non-orthogonality of regression terms can inflate the variance of model coefficients.
- The position of sample points in the design can be influential based on their position with respect to others of the input variables in a subset of the entire set of observations.

An optimal CCD design should minimize both the degree of non-orthogonality of term coefficients and the number of sample points having abnormal influence. In minimizing the degree of non-orthogonality, the *Variation Inflation Factor* (VIF) of regression terms is used. For a **VIF-Optimality** design, the maximum VIF of the regression terms is to be minimized, and the minimum value is 1.0. In minimizing the opportunity of influential sample points, the leverage value of each sample points is used. Leverages are the diagonal elements of the Hat matrix, which is a function of the design matrix. For a **G-Optimality** design, the maximum leverage value of sample points is to be minimized.

For a **VIF-Optimality** design, the alpha value or level is selected such that the maximum VIF is minimized. Likewise, for a **G-Optimality** design, the alpha value or level is selected such that the maximum leverage is minimized. The rotatable design is found to be a poor design in terms of VIF- and G-efficiencies.

For an optimal CCD, the alpha value or level is selected such that both the maximum VIF and the maximum leverage are the minimum possible. For an **Auto Defined** design, the alpha value is selected from either the VIF-Optimality or G-Optimality design that meets the criteria. Because it is a multi-objective optimization problem, in many cases, there is no unique alpha value such that both criteria reach their minimum. However, the alpha value is evaluated such that one criterion reaches minimum while the other approaches minimum.

For the current **Auto Defined** setup (except for a problem with five variables that uses a **G-Optimality** design) all other multi-variable problems use **VIF-Optimality**. In some cases, despite the fact that **Auto Defined** provides an optimal alpha meeting the criteria, this design might not give as good of a response surface as anticipated due to the nature of the physical data used for fitting in the regression process. In this case, you should try other design types that might give a better response surface approximation.

Note:

You can set any design type for CCD as the default in the **Options** dialog box. On the **Design of Experiments** tab, set **Design Type** to the type to want to use as the default. For more information, see [Design Exploration Options](#) (p. 34).

It is a good practice to always verify some selected points on the response surface with an actual simulation evaluation to determine its validity of use for further analyses. In some cases, a good response surface does not mean a good representation of an underlying physics problem. The response surface is generated according to the predetermined sampling points in the design space, which sometimes misses capturing an unexpected change in some regions of the design space. In this case, try using an enhanced design—a **Rotatable** or **Face-Centered** design type with **Template Type** set to **Enhanced**. For an enhanced DOE, a mini CCD is appended to a standard CCD design, where a second alpha value is added and set to half the alpha value of the standard CCD. The mini CCD is set up so that the rotat-

ability and symmetry of the CCD design are maintained. The purposes of the appended mini CCD are to capture any drastic changes in the design space and to provide a better response surface fit.

Note:

Alternatively, you can try to enrich the DOE by changing the selection for **Design of Experiments Type** from **Central Composite Design** to **Custom + Sampling** and then specifying a value for **Total Number of Samples**. For more information, see [Custom + Sampling \(p. 88\)](#).

The location of the generated design points for the deterministic method is based on a central composite design. If N is the number of input parameters, then a central composite design consists of:

- One center point.
- $2*N$ axis point located at the $-\alpha$ and $+\alpha$ position on each axis of the selected input parameters.
- $2^{(N-\mathcal{F})}$ factorial points located at the -1 and +1 positions along the diagonals of the input parameter space.

The fraction \mathcal{F} of the factorial design and the resulting number of design points are given in the following table:

Number of Generated Design Points as a Function of the Number of Input Parameters

Number of Input Parameters	Factorial Number \mathcal{F}	Number of Design Points
1	0	5
2	0	9
3	0	15
4	0	25
5	1	27
6	1	45
7	1	79
8	2	81
9	2	147
10	3	149
11	4	151
12	4	281
13	5	283
14	6	285
15	7	287
16	8	289
17	9	291
18	9	549
19	10	551
20	11	553

Upper and Lower Locations of DOE Points

The upper and lower levels of the DOE points depend on whether the input variable is a design variable (for optimization) or an uncertainty variable (for Six Sigma Analysis). Generally, a response surface is more accurate when closer to the DOE points. Therefore, the points should be close to the areas of the input space that are critical to the effects being examined.

For example, for goal-driven optimization, the DOE points should be located close to where the optimum design is determined to be. For a Six Sigma Analysis, the DOE points should be close to the area where failure is most likely to occur. In both cases, the location of the DOE points depends on the outcome of the analysis. Not having that knowledge at the start of the analysis, you can determine the location of the points as follows:

- For a design variable, the upper and lower levels of the DOE range coincide with the bounds specified for the input parameter. It often happens in optimization that the optimum point is at one end of the range specified for one or more input parameters.
- For an uncertainty variable, the upper and lower levels of the DOE range are the quantile values corresponding to a probability of 0.1% and 99.9%, respectively. This is the standard procedure, whether the input parameter follows a bounded distribution (such as uniform) or unbounded distribution (such as normal). This occurs because the probability that the input variable value exactly coincides with the upper or lower bound for a bounded distribution is exactly zero. Failure can never occur when the value of the input variable is equal to the upper or lower bound. Failure typically occurs in the tails of a distribution, so the DOE points should be located there, but not at the very end of the distribution.

DOE Matrix Generation

The DOE matrix is generated when you run the **Preview** or **Update** operation on your DOE. The matrix consists of generated design points that are submitted to the parent analysis systems for solution. Output parameter values are not available until after the update. When DOE matrix generation is taking a long time, you can monitor the progress by clicking **Show Progress** in the lower right corner of the window. Long generation times tend to occur for Optimal Space-Filling (OSF) and certain VIF (or G-optimal) designs.

Note:

The design points are solved simultaneously if the analysis system is configured to perform simultaneous solutions. Otherwise, they are solved sequentially.

To clear the design points generated for the DOE matrix, return to the **Project Schematic**, right-click the **Design of Experiments** cell, and select **Clear Generated Data**. You can clear data from any design

exploration cell in the **Project Schematic** in this way and regenerate your solution for the cell with changes to the parameters if desired.

Note:

The **Clear Generated Data** operation does not clear the design point cache. To clear this cache, right-click in an empty area of the **Project Schematic** and select **Clear Design Points Cache for All Design Exploration Systems**.

Exporting and Importing Design Points

DesignXplorer gives you the ability to export or import and copy design points:

[Exporting Design Points to a CSV File](#)

[Importing Design Points from a CSV File](#)

Exporting Design Points to a CSV File

You can export design points in a selected DesignXplorer table or chart to an ASCII file, which you can then use in other programs for further processing. This file is primarily formatted according to the "Extended CSV File Format" and has a CSV extension. However, it also supports some nonstandard formatting conventions. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) in the *Workbench User's Guide*.

To export design points to an ASCII file, you use the following options:

- Right-click a cell in the **Table** pane and select **Export Table Data as CSV**.
- Right-click a chart in the **Chart** pane and select **Export Chart Data as CSV**.

The parameter values for each design point in the table or chart are exported to a CSV file. The values are always exported in the units defined in Workbench. This means that they export as when **Display Values as Defined** is selected from the **Units** menu.

You can also import an external CSV file to create either design points in a custom **Design of Experiments** cell or refinement and verification points in a **Response Surface** cell. Right-click a cell in the **Table** pane for a **Design of Experiments** or **Response Surface** cell and select the appropriate import option. For more information, see [Working with Tables \(p. 314\)](#) and [Exporting Table Data \(p. 316\)](#).

Importing Design Points from a CSV File

You can import design points from an external CSV file into a DOE cell if the cell's **Design of Experiments Type** property is set to **Custom** or **Custom + Sampling**.

When a DOE cell is set to a custom type, **Import Design Points from CSV** is available on the context menu when you right-click any of the following:

- A **Design of Experiments** or **Design of Experiments (3D ROM)** cell in a system on the **Project Schematic**
- The **Design of Experiments** node in the **Outline** pane for a **Design of Experiments** cell

- The **Design of Experiments (3D ROM)** node in the **Outline** pane for a **Design of Experiments (3D ROM)** cell
- In the design points table for a **Design of Experiments** or **Design of Experiments (3D ROM)** cell

For more information, see [Importing Data from a CSV File \(p. 316\)](#).

Parsing and Validation of Design Point Data

Before either importing or copying design points into a DOE, DesignXplorer parses and validates the data. If the parameter values for the design points fall within the defined ranges of the parameters in the DOE, the design points are imported or copied into the DOE. If the parameter values for the design points are out of range or do not fill the range, a dialog box opens, giving you options for automatically expanding and shrinking the parameter ranges in the DOE to better fit the imported or copied values.

Note:

If the range of imported or copied values is more than 10 percent smaller than the DOE settings, the shrink option is available to reduce the DOE range to fit. If the values exceed the range defined in the DOE settings, the expand option is available to extend the range.

If you choose not to select either or both check boxes that are available for adjusting parameter ranges, when you click **Apply**, only the design points with all parameters falling within the predefined parameter ranges are imported or copied into the DOE. With the import or copy limited to the predefined parameter ranges, all out-of-bounds design points are ignored. If you click **Cancel**, the import or copy operation is terminated.

Copying Design Points

You can copy existing design points in the **Parameter Set** bar into a DOE cell if the **Design of Experiments Type** property for the DOE cell is set to **Custom** or **Custom + Sampling**. You can copy either all or selected design points. Any output parameter values for the design points copied to the DOE cell are read-only because they have been calculated by a solver.

Note:

Design point data is always parsed and validated before being either copied or imported into a DOE. The dialog boxes that you might see during this process are described in the previous topic.

Copying All Design Points from the Parameter Set Bar into a DOE Cell

When a DOE cell is set to a custom DOE type, you can copy all design points from the **Parameter Set** bar into the DOE cell:

1. In the **Project Schematic**, double-click the DOE cell to which you want to copy design points from the **Parameter Set** bar.

2. In the **Outline** pane, select the parent DOE node.
3. Either right-click the parent node in the **Outline** pane or right-click in the **Table** pane and select **Copy all Design Points from the Parameter Set**.

All design points from the **Parameter Set** bar are copied into the DOE cell.

Copying Selected Design Points from the Parameter Set Bar into a DOE Cell

When a DOE cell is set to a custom DOE type, you can copy selected design points from the **Parameter Set** bar into the DOE cell:

1. In the **Project Schematic**, double-click the **Parameter Set** bar.
2. In the **Table** pane, select one or more design points.
3. Right-click and select **Copy Design Points to** and then select a DOE cell on the submenu.

While the submenu lists all **Design of Experiments** and **Parameters Correlation** cells defined in the project, only custom DOEs and custom correlations are available for selection.

All selected design points in the **Parameter Set** bar are copied into the DOE cell.

Note:

If an unsolved design point was previously copied to a custom DOE, and subsequently this design point was solved in the **Parameter Set** bar, you can copy it to the custom DOE again to push the output values for this design point to the custom DOE.

Using Response Surfaces

Response surfaces are functions of varying natures in which the output parameters are described in terms of the input parameters. Built from the DOE, they quickly provide the approximated values of the output parameters throughout the design space without having to perform a complete solution.

The accuracy of a response surface depends on:

- Complexity of the variations of the solution
- Number of points in the original DOE
- Response surface type

DesignXplorer provides tools to estimate and improve the quality of a response surface.

Once a response surface is generated, you can create and manage response points and charts. These postprocessing tools help you to understand how each output parameter is driven by input parameters and how you can modify your design to improve its performance.

Once your response surface is solved, you can export it as an independent reduced-order model (DX-ROM) to be reused in other environments.

The following sections provide comprehensive information:

[Response Surface Types](#)

[Response Surface Refinement](#)

[Min-Max Search](#)

[Quality Metrics for Response Surfaces](#)

[Response Surface Charts](#)

[Exporting Response Surfaces](#)

Response Surface Types

When you open a **Response Surface** cell, you select **Response Surface** in the **Outline** pane to set properties for the response surface. For **Response Surface Type**, you set the algorithm to use to generate the response surface. The following choices are available:

[Genetic Aggregation](#)

[Full 2nd-Order Polynomials](#)

[Kriging](#)

[Non-Parametric Regression](#)

[Neural Network](#)

[Sparse Grid](#)

Genetic Aggregation

Genetic Aggregation is the default algorithm for generating response surfaces. It automates the process of selecting, configuring, and generating the type of response surface best suited to each output parameter in your problem. From the different types of response surface available (Full 2nd-Order Polynomials, Non-Parametric Regression, Kriging, and Moving Least Squares), Genetic Aggregation automatically builds the response surface type that is the most appropriate approach for each output.

Auto-refinement is available when you select at least one output parameter for refinement in the Tolerances table and specify a tolerance value for this parameter. Once begun, auto-refinement handles design point failures and continues until one of the stopping criteria is met.

Auto-refinement takes into account failed design points by avoiding the areas close to the failed points when generating the next refinement points. The **Crowding Distance Separation Percentage** property specifies the minimum allowable distance between new refinement points, providing a radius around failed design points that serves as a constraint for refinement points.

Genetic Aggregation takes more time than classical response surfaces such as Full 2nd order Polynomial, Non-Parametric Regression, or Kriging because of multiple solves of response surfaces and the cross-validation process. In general, Genetic Aggregation is more reliable than the classical response surface models.

How Genetic Aggregation Works

To select the best response surface, Genetic Aggregation uses a genetic algorithm that generates populations of different response surfaces solved in parallel. The fitness function of each response surface is used to determine which one yields the best approach. It takes into account both the accuracy of the response surface on the design points and the stability of the response surface (cross-validation).

The Genetic Aggregation response surface can be a single response surface or a combination of several different response surfaces (obtained by a crossover operation during the genetic algorithm). For more information, see [Genetic Aggregation \(p. 333\)](#) in the DesignXplorer theory section.

Using the Genetic Aggregation Response Surface

Before generating your response surface, you can set the advanced properties described in [Genetic Aggregation Properties \(p. 103\)](#).

Because a Genetic Aggregation response surface can take longer to generate than other response surfaces, you can monitor the generation process via the progress bar and messages. You also have the ability to stop the update. However, if an update is stopped, any data generated up to that point is discarded.

Once the response surface has been generated, a link to the log file is available in the **Properties** pane. The log file contains information that can help you to assess the quality of your response surface.

Genetic Aggregation Properties

For Genetic Aggregation, the following properties are available in the **Properties** pane for the response surface.

Note:

For advanced options to be visible, the **Show Advanced Options** check box must be selected on the **Design Exploration** tab in the **Options** window. For more information, see [Design Exploration Options \(p. 34\)](#).

Meta Model

- **Response Surface Type:** Determines the type of response surface. This section assumes **Genetic Aggregation** is selected and advanced options are shown.
- **Random Generator Seed:** Advanced option allowing you to specify the value used to initialize the random number generator. By changing this value, you start the Genetic Aggregation from a different population of response surfaces. The default is **0**.
- **Maximum Number of Generations:** Advanced option determining the maximum allowable number of iterations of the genetic algorithm. The default is **12**, with a minimum of **1** and a maximum of **20**.

Log File

- **Display Level:** Advanced option determining the content of the `ResponseSurface.log` file. You can select one of the following values:
 - **Off:** No log file is generated.
 - **Final:** Information on the best response surface generated by the last generation.
 - **Final With Details:** Information on the full population of response surfaces generated by the last generation.
 - **Iterative:** Information on the best response surface generated by each generation.
 - **Iterative With Details:** Information on the full population of response surfaces generated by each generation.

If you change the **Display Level** value after generating the response surface, you must update the response surface again to regenerate the log file with the new content.

- **Log:** Advanced option that displays after a response surface and its log file are generated. This property provides a link to the `ResponseSurface.log` file, which is stored in the project files in the subdirectory `dpall\global\DX`.

Refinement

- **Maximum Number of Refinement Points:** Determines the maximum number of refinement points that can be generated for use with Genetic Aggregation.

- **Number of Refinement Points:** Read-only property indicating the number of existing refinement points.
- **Output Variable Combinations:** Advanced option determining what combinations of output variables are considered.
 - **Maximum Output:** Only the output with the largest ratio between the maximum predicted error and the tolerance is considered. Only one refinement point is generated in each iteration.
 - **All Outputs:** All outputs are considered. Multiple refinement points can be generated. If two refinement points are too close (with a distance less than specified for **Crowding Distance Separation Percentage**), only one is inserted.
- **Crowding Distance Separation Percentage:** Advanced option determining the minimum allowable distance between new refinement points, implemented as a constraint in the search for refinement points.
- **Maximum Number of Refinement Points per Iteration:** Specifies the maximum number of refinement points that can be simultaneously updated at each iteration using your HPC resources. This property is available only when **Output Variable Combinations** is set to **Maximum Output**. The default for **Maximum Number of Refinement Points per Iteration** is **1**. However, to improve efficiency, you can increase this value. For simultaneous design point updates to occur, in the properties for the **Parameter Set** bar, you must set **Update Option** to **Submit to Remote Solve Manager**, specify an available **RSM Queue**, and set **Job Submission** to **One Job for Each Design Point**. For more information about using Remote Solve Manager, see [Working with Parameters and Design Points](#) in the *Workbench User's Guide*. For theoretical information about how design points are updated simultaneously, see [Genetic Aggregation with Multiple Refinement Points](#) (p. 337).
- **Convergence State:** Read-only value indicating the state of the convergence. Possible values are **Converged** and **Not Converged**. If the value is **Not Converged**, the reason for convergence failure is appended.

Verification Points

Generate Verification Points: Specifies whether [verification points](#) (p. 140) are to be generated. When this check box is selected, **Number of Verification Points** becomes visible. The default value is **1**. However, you can enter a different value if desired.

Design Point Report

Report Image: Specifies the image file to display for solved design points in DesignXplorer tables and charts. For more information, see [Viewing Design Point Images in Tables and Charts](#) (p. 304).

Genetic Aggregation Tolerances Table

The tolerances table is available only for the Genetic Aggregation response surface type. It shows output parameter data, enabling you to assess your outputs, select which ones to refine, and enter tolerance values for selected outputs before generating your response surface.

To use Genetic Aggregation auto-refinement, you must:

1. Disable any discrete parameters in your analysis.
2. Select at least one output parameter for refinement and specify its tolerance value. If no output parameters are selected for refinement, manual refinement is used.

To view the tolerances table, select an output parameter or one of the following in the **Outline** pane:

- **Response Surface**
- **Output Parameters**
- **Refinement**
- **Tolerances**

Tolerances Table Properties

For each output parameter defined, the tolerances table displays the following properties:

Calculated Minimum

Minimum value, which is calculated from design point values and Min-Max search results.

Calculated Maximum

Maximum value, which is calculated from design point values and Min-Max search results.

Maximum Predicted Error

Maximum predicted error, which is an estimation of the predicted error for the Genetic Aggregation response surface. For more information, see [Genetic Aggregation \(p. 333\)](#) in the theory section.

Refinement

Determines whether an output parameter and its tolerance are taken into account for the Genetic Aggregation refinement process. When this check box is selected for an output, a tolerance value must be specified.

Tolerance

Enabled and required when an output has been selected for refinement. The value must be greater than **0**.

Tolerance values have units and are refreshed according to whether **Display Values as Defined** or **Display Values in Project Units** is selected from the Workbench **Units** menu.

The tolerances table, the **Properties** pane for the output parameter, and the Convergence chart are fully synchronized. Consequently, changes in one are reflected in the others.

Note:

Discrete output parameters are excluded from the refinement. Property values are approximations based the resulting response surface.

Selecting and Removing Output Parameters for Auto-Refinement

For Genetic Aggregation auto-refinement, you select the output parameters for refinement and assign tolerance values.

Selecting Outputs for Auto-Refinement

To use Genetic Aggregation auto-refinement, you must select at least one output parameter for refinement and assign it a tolerance value.

1. Use one of the following methods to select at least one output parameter for refinement:
 - In the tolerances table, select the **Refinement** check box for the output.
 - In the **Outline** pane, right-click the output parameter and select **Use for Refinement**.
2. Enter a value greater than **0** in the **Tolerance** column.
3. Update the response surface.

All output parameters selected for refinement are included in the refinement process. Their tolerance values are taken into account when identifying new refinement points.

Removing an Output from Auto-Refinement

You can remove an output parameter from consideration for Genetic Aggregation auto-refinement.

1. Use one of the following methods to remove the output parameter:
 - In the tolerances table, clear the **Refinement** check box for the output.
 - In the **Outline** pane, right-click the output parameter and select **Ignore for Refinement**.
2. Update the response surface.

If at least one output parameter is still selected for refinement, the auto-refinement process is used. If the check boxes for all output parameters have been cleared, the refinement reverts to a manual process.

Cleared and derived output parameters are disabled. They are excluded from the refinement process and their tolerance values are ignored.

Example Output Auto-Refinement Settings

The following tolerances table shows output parameters with different auto-refinement settings. Because at least one output is selected for refinement, you know that auto-refinement is to be used.

- **P6**: Never selected for refinement (is not to be included in refinement).
- **P7**: Selected for refinement with tolerance value defined (is included in refinement).
- **P8**: Selected for refinement with tolerance value pending (prevents refinement until tolerance is defined).
- **P9**: Not selected for refinement with the previously defined tolerance value disabled (is not included in refinement).

	A	B	C	D	E	F
1	Name	Calculated Minimum	Calculated Maximum	Maximum Predicted Error	Refinement	Tolerance
2	P6 - WB_V (m ³)	8.3147E-07	1.1984E-06		<input type="checkbox"/>	
3	P7 - WB_SIG (Pa)	8.9135E+09	1.6096E+10		<input checked="" type="checkbox"/>	5900
4	P8 - WB_DIS (m)	0.049927	0.12959		<input checked="" type="checkbox"/>	
5	P9 - WB_BUCK (N)	672.4	1547.4		<input type="checkbox"/>	8650

Genetic Aggregation Convergence Curves Chart

The Genetic Aggregation Convergence Curves chart enables you to monitor auto-refinement of the Genetic Aggregation response surface for one or more selected output parameters. It is available only when the **Response Surface Type** is set to **Genetic Aggregation** and at least one output parameter is selected for auto-refinement.

The chart is automatically generated and dynamically updated as the Genetic Aggregation refinement runs. To view the chart, select one of the following in the **Outline** pane:

- **Refinement**
- **Tolerances**
- **Refinement Points**

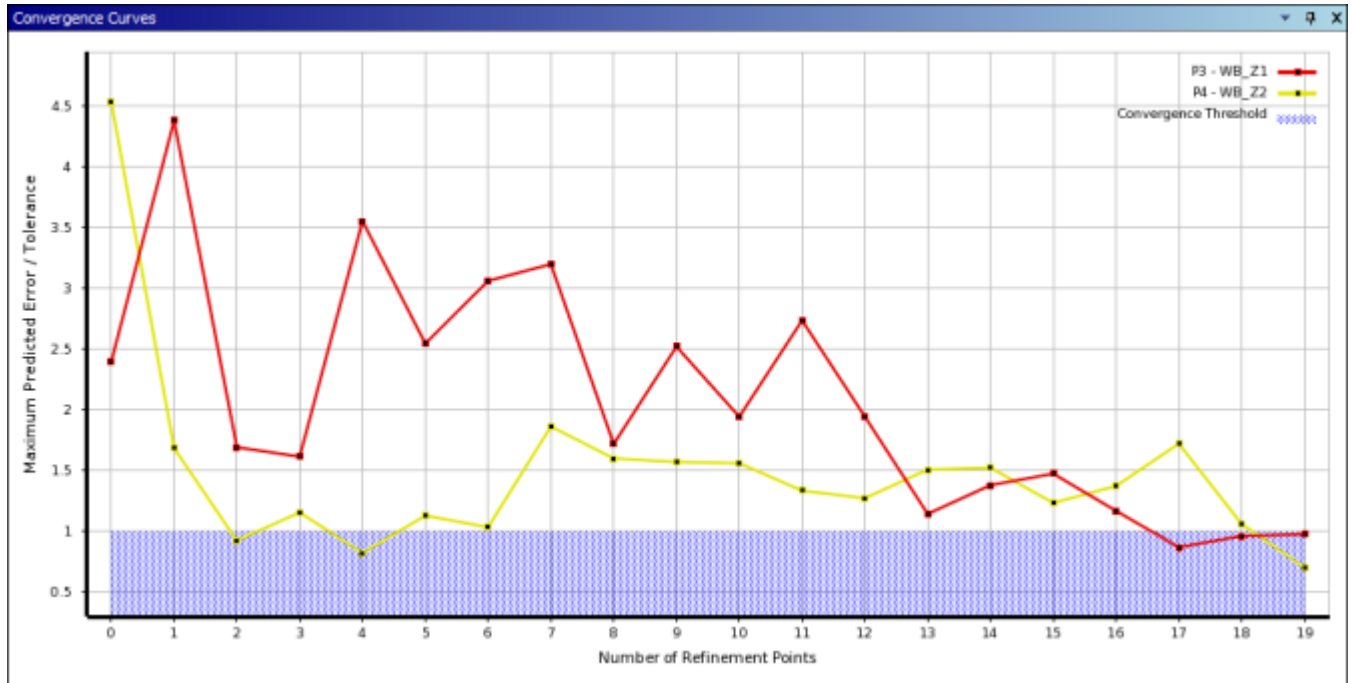
The X axis displays the number of refinement points (whether successfully updated or not) used to refine the response surface, while the Y axis displays the ratio between the maximum predicted error and the tolerance for each output parameter.

Each output parameter marked for auto-refinement is represented by a separate curve corresponding to this ratio, which is calculated for each output parameter to refine and at each iteration. Anything below the convergence threshold curve is in the convergence region, indicated by a shaded blue area on the chart.

Example

Before the first refinement point is run, if output **P3** has a tolerance of 0.2[g] and a maximum predicted error of 0.3[g], it has a ratio of $0.3/0.2=1.5$. If the ratio is equal to 1.5, you know that the maximum predicted error is 1.5 times larger than the tolerance.

The auto-refinement process can generate one point or several points per iteration. The objective is to reach a convergence threshold of less than or equal to 1 for all outputs used in the auto-refinement process, at which point all the convergence curves are in the area below the convergence threshold. The refinement process stops when either the maximum number of refinement points has been reached or the convergence threshold objective has been met.



You can enable or disable a convergence curve by selecting or clearing the check box for the output parameter in the **Properties** pane of the Convergence Curves chart.

Once the Convergence Curves chart has been generated, you can export it as a CSV file by right-clicking the chart and selecting **Export Chart Data as CSV**.

The chart is also included in the Workbench project report.

Full 2nd-Order Polynomials

Regression analysis is a statistical methodology that utilizes the relationship between two or more quantitative variables so that one dependent variable can be estimated from the others. A regression analysis assumes that there are a total of n sampling points and for each sampling point the corresponding values of the output parameters are known. The regression analysis then determines the relationship between the input parameters and the output parameter based on these sample points. This relationship also depends on the chosen regression model.

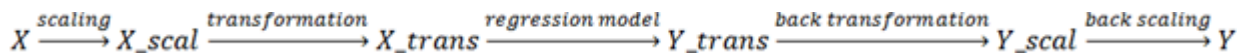
A second-order polynomial is typically preferred for the regression model. This model is generally an approximation of the true input-to-output relationship, and only in special cases does it yield a true

and exact relationship. Once this relationship is determined, the resulting approximation of the output parameter as a function of the input variables is called the response surface.

How Full 2nd-Order Polynomials Works

To select the best response surface, Full 2nd-Order Polynomials performs these steps:

- Scaling on input parameters.
- Transformation on input parameters.
- Calculation of polynomial coefficients based on these modified input values. Some polynomial terms can be filtered by using the **F-Test Filtering** and **Significance Level** properties.
- Back-transformation on the output parameter.
- Back-scaling on the output parameter.



Consequently, the response surface that is generated can fit more complex responses than simple parabolic curvatures.

Forward Stepwise Regression

In forward stepwise regression, the individual regression terms are iteratively added to the regression model if they are found to cause a significant improvement of the regression results. For the DOE, a partial F-test is used to determine the significance of the individual regression terms.

Transformation Functions for Improving the Regression Model

Only in special cases can output parameters of a finite element analysis, such as displacements or stresses, be exactly described by a second-order polynomial as a function of the input parameters. A second-order polynomial usually provides only an approximation. The quality of the approximation can then be significantly improved by applying a transformation function. By default, the Yeo-Johnson transformation is applied.

If the goodness of fit of the response surface is not as good as expected for an output parameter, you can select a different transformation type in its properties. The Yeo-Johnson transformation is more numerically stable in its back-transformation. While the Box-Cox transformation is more numerically unstable in its back-transformation, in some cases, it gives a better fit. If in the **Properties** pane for an output parameter, you set **Transformation Type** to **None**, the full 2nd-order polynomials response surface is computed without any transformation on the data for this output parameter.

Full 2nd-Order Polynomials Properties

When **Response Surface Type** is set to **Standard Response Surface - Full 2nd-Order Polynomials**, the **Properties** pane for the response surface displays the following properties under **Meta Model**:

Note:

For advanced options to be visible, the **Show Advanced Options** check box must be selected on the **Design Exploration** tab in the **Options** window. For more information, see [Design Exploration Options \(p. 34\)](#).

- **Response Surface Type:** Determines the type of response surface. This section assumes **Standard Response Surface - Full 2nd-Order Polynomials** is selected and advanced options are shown.
- **Inputs Transformation Type:** Advanced option determining the type of power transformation to apply to all continuous input parameters, with and without manufacturable values, before solving the response surface. Choices are **Yeo-Johnson** (default) and **None**. When **None** is selected, no transformations are applied to continuous input parameters.
- **Inputs Scaling:** Advanced option determining whether to scale the data for all continuous input parameters, with and without manufacturable values, before solving the response surface. This check box is selected by default. If you clear this check box, no scaling of input parameter data occurs.
- **Significance Level:** Advanced option indicating the threshold to use during model-building to filter significant terms of the polynomial regression. The range for possible values is from **0** to **1**. The default is **0.05**.

Output Parameter Properties

When an output parameter is selected in the **Outline** pane, the **Properties** pane displays the following properties under **Output Settings**:

Note:

As indicated earlier, for advanced options to be visible, the **Show Advanced Options** check box must be selected on the **Design Exploration** tab in the **Options** window.

- **Transformation Type:** Determines the type of power transformation to apply to the output parameter. Choices are **Yeo-Johnson** (default), **Box-Cox**, and **None**. When **None** is selected, no transformation is applied to the output parameter. Transformations are not applied to derived output parameters.
- **Scaling:** Advanced option determining whether to scale the data for the output parameter. This check box is selected by default. If you clear this check box, no scaling of the output parameter data occurs.

Kriging

Kriging is a meta-modeling algorithm that provides an improved response quality and fits higher order variations of the output parameter. It is an accurate multidimensional interpolation combining a polynomial model similar to the one of the standard response surface—which provides a "global" model of the design space—plus local deviations so that the Kriging model interpolates the DOE points. Kriging provides refinement capabilities for continuous input parameters, including those with manufacturable values. It does not support discrete parameters. The effectiveness of Kriging is based on the ability of its internal error estimator to improve response surface quality by generating refinement points and adding them to the areas of the response surface most in need of improvement.

In addition to manual refinement capabilities, Kriging offers an auto-refinement option that automatically and iteratively updates the refinement points during the update of the response surface. At each iteration of the refinement, Kriging evaluates a predicted relative error in the full parameter space. DesignXplorer uses the predicted relative error instead of the predicted error because this allows the same values to be used for all output parameters, even when the parameters have different ranges of variation.

At this step in the process, the predicted relative error for one output parameter is the predicted error of the output parameter normalized by the known maximum variation of the output parameter:

$$\text{PredictedRelativeError} = \frac{100 \times \text{PredictedRelativeError}}{(O_{max} - O_{min})}$$

Where O_{max} and O_{min} are the maximum and minimum known values (on design points) of the output parameter.

For guidelines on when to use Kriging, see [Changing the Response Surface \(p. 124\)](#).

How Kriging Auto-Refinement Works

Auto-refinement enables you to refine the Kriging response surface automatically by choosing certain refinement criteria and allowing an automated optimization-type procedure to add more points to the design domain where they are most needed.

The prediction of error is a continuous and differentiable function. To find the best candidate refinement point, the refinement process determines the maximum of the prediction function by running a gradient-based optimization procedure. If the prediction of the accuracy for the new candidate refinement point exceeds the required accuracy, the point is then promoted as a new refinement point.

The auto-refinement process continues iteratively, locating and adding new refinement points until either the refinement has converged or the maximum allowable number of refinement points has been generated. The refinement converges when the response surface is accurate enough for direct output parameters.

For more information, see [Kriging \(p. 340\)](#) in the theory section.

Using Kriging Auto-Refinement

When using the Kriging auto-refinement capabilities, there are four main steps to the process:

- Setting up the response surface

- Setting up the output parameters
- Updating the response surface
- Generating verification points

Setting Up the Response Surface

To set up a response surface for Kriging auto-refinement:

1. In the **Outline** pane for the response surface, select the **Response Surface** cell.
2. Under **Meta Model** in the **Properties** pane, set **Response Surface Type** to **Kriging**.
3. Under **Refinement** in the **Properties** pane:
 - Set **Refinement Type** to **Auto**.
 - For **Maximum Number of Refinement Points**, enter the maximum number of refinement points that can be generated.
 - For **Maximum Predicted Relative Error (%)**, enter the maximum predicted relative error that is acceptable for all parameters.

Note:

For advanced options to be visible, the **Show Advanced Options** check box must be selected on the **Design Exploration** tab in the **Options** window. For more information, see [Design Exploration Options \(p. 34\)](#).

4. If advanced options are shown, under **Refinement** in the **Properties** pane:
 - For **Output Variable Combinations**, select a value for determining how output variables are considered in terms of predicted relative error. This value controls the number of refinement points that are created per iteration.
 - For **Crowding Distance Separation Percentage**, enter a value for determining the minimum allowable distance between new refinement points.

During the refinement process, if one or more design points fail, auto-refinement uses the value specified for this property to avoid areas close to failed design points when generating the next refinement points. This property specifies the minimum allowable distance between new refinement points, providing a radius around failed design points that serves as a constraint for refinement points.

5. Under **Verification Points** in the **Properties** pane:
 - If you want to generate [verification points \(p. 140\)](#), select the **Generate Verification Points** check box. **Number of Verification Points** becomes visible. The default value is **1**.
 - If you want a different number of verification points to be generated, enter the desired value.

For more information, see [Kriging Auto-Refinement Properties \(p. 114\)](#).

Setting Up the Output Parameters

1. In the **Outline** pane for the response surface, select an output parameter.
2. Under **Refinement** in the **Properties** pane, select or clear the **Inherit From Model Settings** check box. This determines whether the maximum predicted relative error defined at the model level is applicable to the parameter.

If the **Inherit From Model Settings** check box is cleared, **Maximum Predicted Relative Error** becomes available so that you can enter the maximum predicted relative error that you find acceptable. This can be different than the maximum predicted relative error defined at the model level.

For more information, see [Kriging Auto-Refinement Properties \(p. 114\)](#).

Updating the Response Surface

Once you've defined the response surface and output parameter properties, complete the refinement by updating the response surface.

The generated points for the refinement appear in the refinement points table, which displays in the **Table** pane when either **Refinement** or a refinement point is selected in the **Outline** pane. As the refinement points are updated, the Convergence Curves chart updates dynamically, allowing you to monitor the progress of the Kriging auto-refinement. For more information, see [Kriging Convergence Curves Chart \(p. 115\)](#).

The auto-refinement process continues until either the maximum number of refinement points is reached or the response surface is accurate enough for direct output parameters. If all output parameters have a predicted relative error that is less than the **Maximum Predicted Relative Error** values defined for them, the refinement is converged.

Generating Verification Points

Once the auto-refinement has converged (and particularly at the first iteration without refinement points), you should generate verification points to validate the response surface accuracy.

As with Sparse Grid, Kriging is an interpolation. Goodness of fit is not a reliable measure for Kriging because the response surface passes through all of the design points, making the goodness of fit appear to be perfect. As such, the generation of verification points is essential for assessing the quality of the response surface and understanding the actual goodness of fit.

If the accuracy of the verification points is larger than the predicted relative error given by Kriging, you can insert the verification points as refinement points and then run a new auto-refinement so that the new points are included in the generation of the response surface. Verification points can only be inserted as refinement points in manual refinement mode. For more information, see [Verification Points \(p. 140\)](#).

Kriging Properties

For Kriging, the following properties are available in the **Properties** pane for the response surface.

Note:

For advanced options to be visible, the **Show Advanced Options** check box must be selected on the **Design Exploration** tab in the **Options** window. For more information, see [Design Exploration Options \(p. 34\)](#).

Meta Model

- **Response Surface Type:** Determines the type of response surface. This section assumes **Kriging** is selected and advanced options are shown.
- **Kernel Variation Type:** Determines the type of kernel variation. Choices are:
 - **Variable:** Sets the kernel variation to pure Kriging mode, which uses a correlation parameter for each design variable.
 - **Constant:** Sets the kernel variation to radial basis function mode, which uses a single correlation parameter for all design variables.

Refinement

- **Refinement Type:** Determines whether refinement criteria are to be specified manually (points of your choice in addition to the DOE points) or selected automatically by the response surface.
- **Maximum Number of Refinement Points:** Determines the maximum number of refinement points that can be generated for use with the Kriging algorithm.
- **Number of Refinement Points:** Read-only property indicating the number of existing refinement points.
- **Maximum Predicted Relative Error (%):** Determines the maximum predicted relative error that is acceptable for all parameters.
- **Output Variable Combinations:** Advanced option determining what combinations of output variables are considered.
 - **Maximum Output:** Only the output with the largest predicted relative error is considered. Only one refinement point is generated in each iteration.
 - **All Outputs:** All outputs are considered. Multiple refinement points are generated in each iteration.
 - **Sum of Outputs:** The combined predicted relative error of all outputs is considered. Only one refinement point is generated in each iteration.
- **Crowding Distance Separation Percentage:** Advanced option determining the minimum allowable distance between new refinement points, implemented as a constraint in the search for refinement points.

If two candidate refinement points are closer together than the defined minimum distance, only the first candidate is inserted as a new refinement point.

- **Predicted Relative Error (%)**: Read-only value indicating the predicted relative error for all parameters.
- **Converged**: Read-only value indicating the state of the convergence. Possible values are **Yes** and **No**.

Kriging Auto-Refinement Output Parameter Properties

If the **Refinement Type** for Kriging is set to **Auto**, auto-refinement properties are available in the **Properties** pane when an output parameter is selected in the **Outline** pane.

- **Inherit From Model Settings**: Indicates if the maximum predicted relative error that you've defined at the model level is applicable for this output parameter. This check box is selected by default.
- **Maximum Predicted Relative Error (%)**: Displays only when the **Inherit From Model Settings** check box is cleared. Determines the maximum predicted relative error that you accept for this output parameter. This value can be different than the maximum predicted relative error defined at the model level.
- **Predicted Relative Error**: Read-only value populated on update. Predicted relative error for this output parameter.

Kriging Convergence Curves Chart

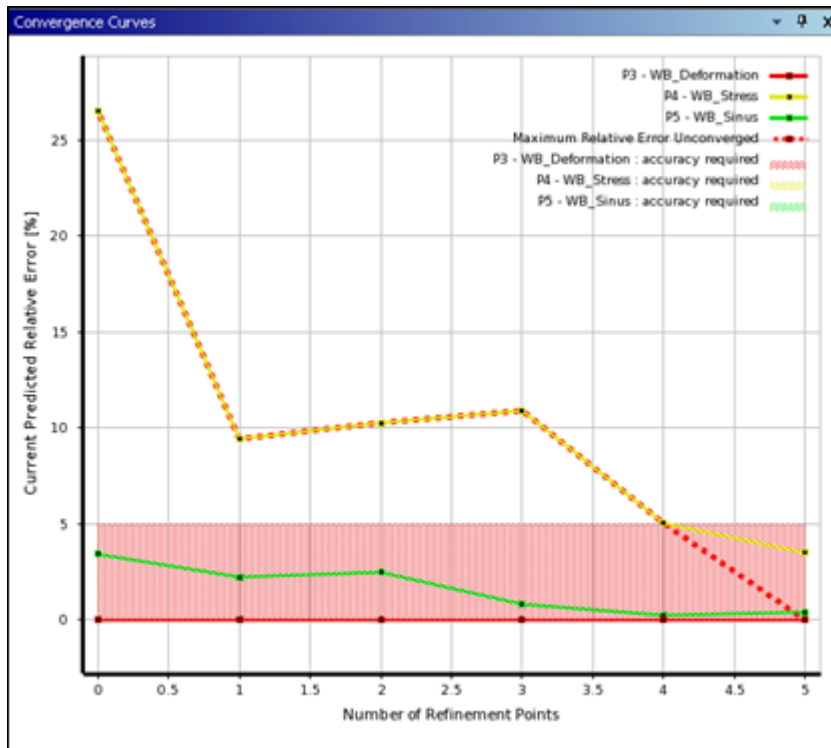
The Kriging Convergence Curves Chart allows you to monitor auto-refinement of the Kriging response surface for one or more selected output parameters. The X axis displays the number of refinement points used to refine the response surface. The Y axis displays the percentage of the current predicted relative error.

The chart is automatically generated and dynamically updated as the Kriging refinement runs. You can view the chart and its properties by selecting **Refinement** or a **Refinement Points** in the **Outline** pane.

There are two curves for each output parameter. One curve represents the percentage of the current predicted relative error. The other curve represents the maximum predicted relative error required that parameter.

Additionally, there is a single curve that represents the maximum of the predicted relative error for output parameters that are not converged.

During the run, you can click the red stop button in the **Progress** pane to interrupt the process so that you can adjust the requested maximum error or change chart properties before continuing.



Non-Parametric Regression

Non-parametric regression (NPR) provides improved response quality and is initialized with one of the available DOE types. NPR algorithm is implemented in DesignXplorer as a metamodeling technique prescribed for predictably high nonlinear behavior of the outputs with respect to the inputs.

NPR belongs to a general class of Support Vector Method (SVM) type techniques. These are data classification methods that use hyperplanes to separate data groups. The regression method works similarly. The main difference is that the hyperplane is used to categorize a subset of the input sample vectors that are deemed sufficient to represent the output in question. This subset is called the *support vector set*.

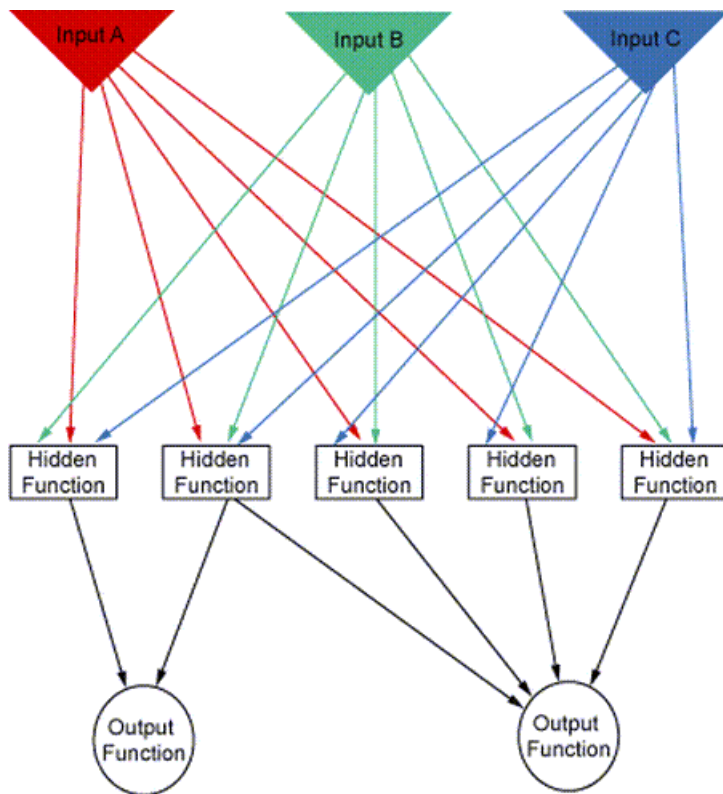
The internal parameters of the response surface are fixed to constant values and are not optimized. The values are determined from a series of benchmark tests and strike a compromise between the response surface accuracy and computational speed. For a large family of problems, the current settings provide good results. However, for some problem types (like ones dominated by flat surfaces or lower order polynomials), some oscillations might be noticed between the DOE points.

To circumvent this, you can use a larger number of DOE points or, depending on the fitness landscape of the problem, use one of the several optimal space-filling DOEs provided. In general, it is suggested that the problems first be fitted with a quadratic response surface and the NPR fitting adopted only when the goodness of fit from the quadratic response surface model is unsatisfactory. This ensures that the NPR is only used for problems where low order polynomials do not dominate.

Neural Network

A neural network is a mathematical technique based on the natural neural network in the human brain.

To interpolate a function, a network with three levels (input, hidden, and output) is built and the connections between them are weighted.



Each arrow is associated with a weight (w). Each ring is called a *cell* (like a neuron).

If the inputs are x_i , the hidden level contains function $g_j(x_i)$. The output solution is:

$$f_k(x_i) = K\left(\sum w_{jk} g_j(x_i)\right)$$

Where K is a predefined function, such as the hyperbolic tangent or an exponential based function, to obtain something similar to the binary behavior of the electrical brain signal (like a step function). The function is continuous and differentiable.

The weight functions (w_{jk}) are issued from an algorithm that minimizes (as the least squares method) the distance between the interpolation and the known values (design points). This is called *learning*. The error is checked at each iteration with the design points that are not used for learning. Learning design points need to be separated from error-checking design points.

The error decreases and then increases when the interpolation order is too high. The minimization algorithm is stopped when the error is the lowest.

This method uses a limited number of design points to build the approximation. It works better when the number of design points and the number of intermediate cells are high. It can give interesting results with several parameters.

Neural Network Properties

For Neural Network, the following properties are available in the **Properties** pane for the response surface.

Meta Model

- **Response Surface Type:** Determines the type of response surface. This section assumes **Neural Network** is selected.
- **Number of Cells:** Determines the number of neurons to use in the hidden layer of the neural network.

Sparse Grid

Sparse Grid provides refinement capabilities for continuous parameters, including those with manufacturable values. It does not support discrete parameters. Sparse Grid uses an adaptive response surface, which means that it refines itself automatically. A dimension-adaptive algorithm allows it to determine which dimensions are most important to the objectives functions, thereby reducing computational effort.

How Sparse Grid Works

Sparse Grid allows you to select certain refinement criteria. When you update the response surface, Sparse Grid uses an automated local refinement process to determine the areas of the response surface that are most in need of further refinement. It then concentrates refinement points in these areas, allowing the response surface to reach the specified level of accuracy more quickly and with fewer design points.

Sparse Grid is an adaptive algorithm based on a hierarchy of grids. The DOE type **Sparse Grid Initialization** generates a DOE matrix containing all the design points for the smallest required grid: the level **0** (the point at the current values) plus the level **1** (two points per input parameters). If the expected level of quality is not met, the algorithm further refines the grid by building a new level in the corresponding directions. This process is repeated until one of the following occurs:

- The response surface reaches the requested level of accuracy.
- The value specified for **Maximum Number of Refinement Points** is reached.
- The value specified for **Maximum Depth** is reached in one level. The maximum depth is the maximum number of levels that can be created in the hierarchy. Once the maximum depth for a direction is reached, there is no further refinement in that direction.

The relative error for an output parameter is the error between the predicted and the observed output values, normalized by the known maximum variation of the output parameter at this step of the process. Because there are multiple output parameters to process, DesignXplorer computes the worst relative error value for all of the output parameters and then compares this against the value for **Maximum Relative Error**. As long as at least one output parameter has a relative error greater than the expected error, the maximum relative error criterion is not validated.

Sparse Grid Requirements

Sparse Grid requires a specific Clenshaw-Curtis grid that is generated only by the DOE type **Sparse Grid Initialization**. Consequently, Sparse Grid can only be used with the DOE type **Sparse Grid Initialization**. If you've defined another DOE type, the Sparse Grid response surface cannot be updated. However, the DOE type **Sparse Grid Initialization** can be used by other types of response surfaces.

Because Sparse Grid uses an auto-refinement algorithm, it is not possible to add refinement points manually. As a result, for Sparse Grid, the following behaviors occur:

- **Sparse Grid Refinement Type** is set to **Auto** and cannot be changed.
- **Insert as Refinement Point operation** is not available on the right-click context menu. Also, if you use commands to attempt to insert a refinement point, an error occurs.

Using Sparse Grid Auto-Refinement

When using Sparse Grid auto-refinement, there are four main steps to the process:

- Setting up the DOE
- Setting up the response surface
- Updating the response surface
- Generating verification points

Setting Up the DOE

To set up the DOE for Sparse Grid auto-refinement:

1. In the **Outline** pane for the **Design of Experiments** cell, select **Design of Experiments**.
2. Under **Design of Experiments** in the **Properties** pane, set **Design of Experiments Type** to **Sparse Grid**.
3. Either preview or update the DOE to validate your selections. This causes the **Response Surface Type** property for **Response Surface** cells in downstream **Design Exploration** systems to default to **Sparse Grid**.

Note:

If the **Design of Experiments** cell is shared by multiple systems, the DOE definition applies to the **Response Surface** cells for each of those systems.

Setting Up the Response Surface

To set up refinement properties for a Sparse Grid auto-refinement:

1. In the **Outline** pane for the **Response Surface** cell, select **Response Surface**.
2. In the **Properties** pane under **Meta Model**, verify that **Response Surface Type** is set to **Sparse Grid**.

3. In the **Properties** pane under **Refinement**:

- For **Maximum Relative Error (%)**, enter the value to be allowed for all of the output parameters. The smaller the value, the more accurate the response surface.
- For **Maximum Depth**, enter the number of grid levels that can be created in a given direction. If needed, you can adjust this value later, according to your update results.
- For **Maximum Number of Refinement Points**, enter the maximum number of refinement points that can be generated as part of the refinement process.

For more information, see [Sparse Grid Auto-Refinement Properties \(p. 121\)](#).

Updating the Response Surface

Once the DOE and response surface are defined, update the response surface to adaptively generate the Sparse Grid response surface.

At any time, you can click the red stop button in the **Progress** pane to interrupt the Sparse Grid refinement so that you can change properties or see partial results. The refinement points that have already been calculated are visible, and the displayed charts are based on the response surface's current level of refinement. The refinement points that have not been calculated display the **Update Required** icon to indicate which output parameters must be updated.

If a design point fails during the refinement process, Sparse Grid refinement stops in the area where the failed design point is located, but refinement continues in the rest of the parameter space to the degree possible. Failed refinement points display the **Update Failed, Update Required** icon. You can attempt to pass the failed design points by updating the response surface once again.

The auto-refinement process continues until the **Maximum Relative Error (%)** objective is attained, the **Maximum Depth** limit is reached for all input parameters, the **Maximum Number of Refinement Points** is reached, or the response surface converges. If all output parameters have a **Maximum Relative Error (%)** that is higher than the **Current Relative Error** defined for them, the refinement is converged.

Note:

If Sparse Grid refinement does not appear to be converging, you can do the following to accept the current level of convergence:

1. Stop the process.
 2. Either set the value for **Maximum Relative Error (%)** slightly above the value for **Current Relative Error (%)** or set the value for **Maximum Number of Refinement Points** equal to the value for **Number of Refinement Points**.
 3. Update the response surface again.
-

Generating Verification Points

Like Kriging, Sparse Grid is an interpolation. To assess the quality of the response surface, verification points are needed. Once the Sparse Grid auto-refinement has converged, you should generate verification points to validate the response surface accuracy.

Note:

Because Sparse Grid uses an auto-refinement process, you cannot add refinement points manually as with other DOE types. To generate verification points automatically, under **Verification Points** in the **Properties** pane for the **Response Surface** cell, select the **Generate Verification Points** check box. Then, update the response surface.

For more information on assessing the quality of a response surface, see [Goodness of Fit for Output Parameters in a Response Surface](#) (p. 130) and [Verification Points](#) (p. 140).

Sparse Grid Auto-Refinement Properties

The **Refinement** properties in the **Properties** pane for a Sparse Grid response surface determine the number and the spread of the refinement points.

- **Refinement Type:** Type of refinement process. The default is **Auto** (read-only).
- **Maximum Relative Error (%):** Maximum relative error allowable for the response surface. This value is used to compare against the worst relative error obtained for all output parameters. So long as any output parameter has a relative error greater than the expected relative error, this criterion is not validated. This property is a percentage. The default is **5**.
- **Maximum Depth:** Maximum depth (number of hierarchy levels) that can be created as part of the Sparse Grid hierarchy. Once the number of levels defined in this property is reached for a direction, refinement does not continue in that direction. The default is **4**. A minimum value of **2** is required because the Sparse Grid Initialization DOE type is already generating levels **0** and **1**).
- **Maximum Number of Refinement Points:** Maximum number of refinement points that can be generated for use with the Sparse Grid algorithm. The default is **1000**.
- **Number of Refinement Points:** Number of existing refinement points.
- **Current Relative Error (%):** Current level of relative error.
- **Converged:** Indicates the state of the convergence. Possible values are **Yes** and **No**.

Sparse Grid Convergence Curves Chart

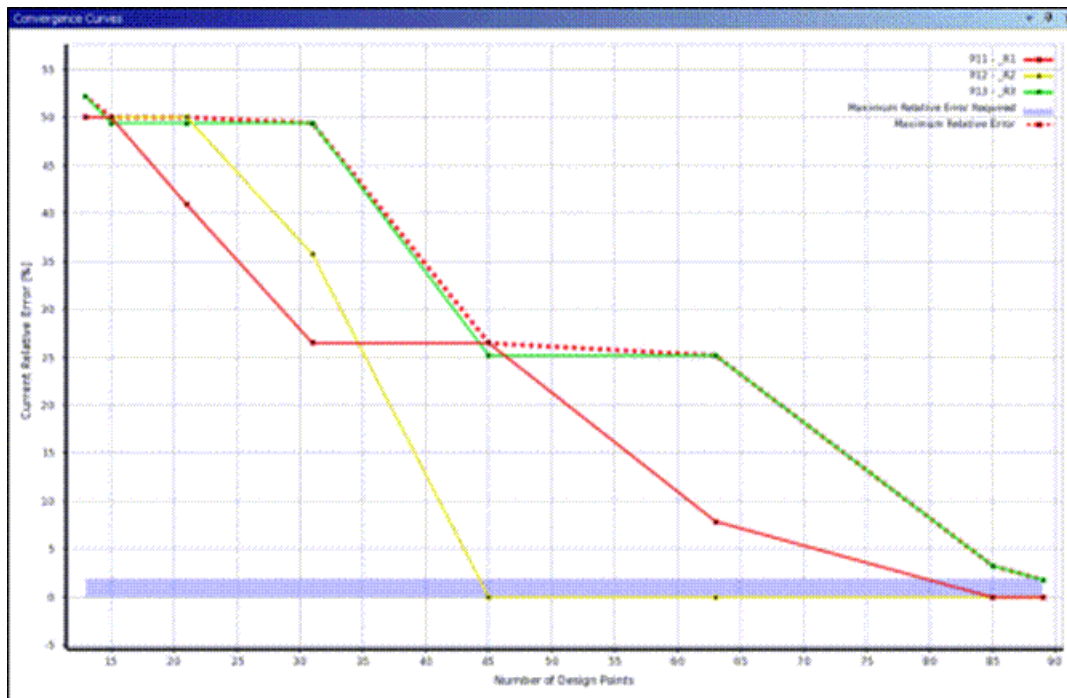
The Sparse Grid Convergence Curves chart allows you to monitor auto-refinement of the Sparse Grid response surface. The X axis indicates the number of design points used to build the response surface. The Y axis indicates the convergence criteria.

The chart is automatically generated and dynamically updated as the Sparse Grid refinement runs. You can view the chart and its properties by selecting **Refinement** or a refinement point in the **Outline** pane.

There is one curve per output parameter to represent the current relative error (in percentage), and one curve to represent the maximum relative error for all direct output parameters. Auto-refinement stops when the maximum relative error required (represented as a horizontal threshold line) has been met.

You can disable one or several output parameters curves and keep only the curve of the maximum relative error.

During the run, you can click the red stop button in the **Progress** pane to interrupt the process so that you can adjust the requested maximum error or change chart properties before continuing.



Response Surface Refinement

The quality of your response surface is affected by the type of meta-modeling algorithm used to generate it and is measured by the response surface's goodness of fit. For more information, see [Goodness of Fit for Output Parameters in a Response Surface](#) (p. 130).

The following sections provide recommendations on making your initial selection of a meta-modeling algorithm, evaluating the response surface performance in terms of goodness of fit, and changing the response surface as needed to improve the goodness of fit:

- [Working with Response Surfaces](#)
- [Changing the Response Surface](#)
- [Refinement Points](#)
- [Performing a Manual Refinement](#)

Working with Response Surfaces

To work with a response surface, you begin by preparing your model to enhance the goodness of fit. You then select the type of response surface to generate. After response surface generation, you assess its quality by reviewing the goodness of fit for each output parameter.

Prepare Your Model to Enhance Goodness of Fit

- Use a sufficient number of design points.

The number of design points should always exceed the number of inputs. Ideally, you should have at least twice as many design points as inputs. Most of the standard DOE types are designed to generate a sufficient number, but custom DOE types might not.

- Reduce the number of input parameters.

Only keep the input parameters that are playing a major role in your study. Disable any input parameters that are not relevant by clearing their check boxes in the DOE's **Outline** pane. You can determine which are relevant from a correlation or sensitivity analysis.

Requirements and recommendations regarding the number of input parameters vary according to the DOE type selected. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).

- As a general rule, always create verification points.

To specify that verification points are to be created when the response surface is generated, select the **Generate Verification Points** check box in the response surface's **Properties** pane.

Start with a Genetic Aggregation or Standard Response Surface

You can always begin your response surface study with the default **Genetic Aggregation** response surface type. The automated nature of this response surface helps to ensure that the most appropriate type of response surface is generated for each output parameter.

If you want to manually select a response surface type, however, it is usually a good practice to begin with the Standard Response Surface 2nd-Order Polynomials response surface. Once the response surface is built, you can assess its quality by reviewing the goodness of fit for each output parameter.

Review the Goodness of Fit

To assess the quality of the response surface, you should review the goodness of fit for each output parameter. To review the goodness of fit:

1. In the **Project Schematic**, right-click the **Response Surface** cell and select **Edit**.
2. In the **Outline** pane, under **Quality**, select the goodness of fit object.

The **Table** pane displays goodness-of-fit metrics for each output parameter. The **Chart** pane displays the Predicted vs. Observed chart.

3. Review the goodness of fit, paying particular attention to the **Coefficient of Determination** property. The closer this value is to **1**, the better the response surface. For more information, see [Goodness of Fit Criteria \(p. 131\)](#).
 - If the goodness of fit is acceptable, add verification points and then recheck the goodness of fit. If needed, you can further refine the response surface manually. For more information, see [Performing a Manual Refinement \(p. 126\)](#).
 - If the goodness of fit is poor, try changing your response surface. For more information, see [Changing the Response Surface \(p. 124\)](#).

Changing the Response Surface

Changing response surface types or changing the available options for the current response surface can improve the goodness of fit. To change your response surface type:

1. In the **Project Schematic**, right-click the **Response Surface** cell and select **Edit**.
2. In the **Outline** pane, select **Response Surface**.
3. In the **Properties** pane under **Meta Model**, select a different choice for **Response Surface Type**.
4. Update the response surface.
5. Review the goodness of fit for each output parameter to see if the new response surface provides a better fit.

Once the goodness of fit is acceptable, you should add verification points and then recheck the goodness of fit. If needed, you can further refine the response surface manually. For more information, see [Performing a Manual Refinement \(p. 126\)](#).

Guidelines for Changing the Response Surface Type

Standard Response Surface - Full 2nd-Order Polynomials

If the default **Genetic Aggregation** response surface type takes too long to generate response surfaces, you can switch to **Standard Response Surface - Full 2nd-Order Polynomials**. This response surface type is particularly effective when the variation of the output is smooth with regard to the input parameters.

Kriging

If **Standard Response Surface Full 2nd-Order Polynomials** does not produce a response surface with an acceptable goodness of fit, try **Kriging**. After updating the response surface with this method, recheck the goodness of fit. For Kriging, **Coefficient of Determination** must be set to **1**. If it is not, the model is over-constrained and not suitable for refinement via Kriging. Kriging fits the response surface through all the design points, which means that many of the other metrics are always perfect. Therefore, it is particularly important to run verification points with Kriging.

Non-Parametric Regression

If the model is over-constrained and not suitable for refinement via Kriging, try switching to **Non-Parametric Regression**.

If you decide to use one of the other response surface types available, consider your selection carefully to ensure that the response surface suits your specific purpose. Characteristics for response surface types follow.

Response Surface Characteristics

Each response surface type has its own set of characteristics, lending itself to different types of responses. The response surface type affects the quality and goodness of fit of the resulting response surface. Use the following characteristics to guide your selection of the response surface type most suited to your design scenario.

Genetic Aggregation

- Default response surface type
- Efficient and reliable
- Suited to all types of responses
- Can be slow to compute

Standard Response Surface - Full 2nd-Order Polynomials

- Creates a standard response surface
- Effective when the variation of the output is smooth with regard to input parameters

Kriging

- Efficient in a large number of cases
- Suited to highly nonlinear responses
- Don't use when results are noisy

Note:

Kriging is an interpolation that matches the points exactly. Always use verification points to check the goodness of fit.

Non-Parametric Regression

- Suited to nonlinear responses
- Use when results are noisy

Neural Network

- Suited to highly nonlinear responses
- Use when results are noisy
- Control over the algorithm is very limited

Sparse Grid

- Suited for studies containing discontinuities
- Use when the solve is fast

Refinement Points

Refinement points are points added to your model to enrich and improve your response surface. They can either be generated automatically with the response surface update or added manually as described in [Performing a Manual Refinement \(p. 126\)](#). As with design points, DesignXplorer must perform a design point update (a real solve) to obtain the output parameters for the refinement points.

On update, the refinement points are used to build the response surface and are taken into account for the generation of verification points. Along with DOE points, refinement points are used as *learning points* in calculations for the goodness of fit.

All refinement points are shown in the refinement points table, which you access by selecting **Refinement** → **Refinement Points** in the **Outline** pane.

Performing a Manual Refinement

Manual refinement is a way to force the response surface to take into account points of your choice, in addition to the points already in the DOE. You can insert a refinement point in the refinement points table. You do not need to do an initial solve of the response surface (without the refinement point) before updating your response surface with your manual refinement. Manual refinement is available for all response surface types except for Sparse Grid.

After a first optimization study, you can insert the best candidate design as a refinement point to improve the response surface quality in this area of the design space. To create a new refinement point, you can use one of the following methods:

Create a refinement point from existing results:

1. Right-click a response surface verification point, a response point, or an optimization candidate point.
2. Select **Insert at Refinement Point**.

The point is added to the refinement points table and is used in the next response surface generation.

Enter refinement point values manually into the table:

1. In the **Outline** pane, select **Refinement Points**.
2. In the **Table** pane, enter input parameter values into the bottom row.

Note:

By default, output values are not editable in the table. However, you can change the editing mode of the table. For more information, see [Editable Output Parameter Values \(p. 314\)](#).

Import values into the table:

1. In the **Table** pane, right-click in the table and select **Import Refinement Points**.
2. Browse to and select the CSV file containing the refinement points to import. For more information, see [Importing Data from a CSV File \(p. 316\)](#).

Note:

You can also copy information from a CSV file and paste them into the refinement points table.

To update the refinement points and then rebuild the response surface to take these points into account, click **Update** on the toolbar. Each out-of-date refinement point is updated, and the response surface is rebuilt from the DOE points and refinement points.

Min-Max Search

The Min-Max search examines the entire output parameter space from a response surface to approximate the minimum and maximum values of each output parameter. If the **Min-Max Search** check box is selected, a Min-Max search is performed each time the response surface is updated. Clearing the check box disables the Min-Max search. You can disable this feature in cases where the search could be very time-consuming, such as when there are a large number of input parameters or when there are discrete input parameters.

Outline of Schematic C3: Response Surface		Table of Outline A1: Min-Max Search									
	A	B	A	B	C	D	E	F	G	H	
1		Enabled	1	Name	P1 - WB_Thickness	P2 - WB_Radius	P3 - WB_Mass	P4 - WB_Deformation	P5 - WB_Stress	P6 - WB_Sinus	P7 - Output Parameter (n)
2	Response Surface		2	Output Parameter Minimum							
3	Input Parameters		3	P3 - WB_Mass	1	108	8.79E+06	1.976	3096.2	1.3972	4325.9
4	Microsoft Office Excel (A1)		4	P4 - WB_Deformation	2.5	108	8.94E+06	0.026	241.7	1.1542	278.96
5	P1 - WB_Thickness	<input checked="" type="checkbox"/>	5	P5 - WB_Stress	2.5	120	9.9E+06	0.074	64.101	1.3359	95.63
6	P2 - WB_Radius	<input checked="" type="checkbox"/>	6	P6 - WB_Sinus	3	108	8.99E+06	0.376	490.2	0.69682	334.61
7	Output Parameters		7	P7 - Output Parameter	2.5	120	9.9E+06	0.074	64.101	1.3359	85.63
8	Microsoft Office Excel (A1)		8	Output Parameter Maximum							
9	P3 - WB_Mass	<input checked="" type="checkbox"/>	9	P3 - WB_Mass	3	132	1.091E+07	0.76	892.2	1.0179	8746.3
10	P4 - WB_Deformation	<input checked="" type="checkbox"/>	10	P4 - WB_Deformation	1	132	1.07E+07	2.36	11208	1.7183	19259
11	P5 - WB_Stress	<input checked="" type="checkbox"/>	11	P5 - WB_Stress	1	132	1.07E+07	2.36	11208	1.7183	19259
12	P6 - WB_Sinus	<input checked="" type="checkbox"/>	12	P6 - WB_Sinus	1.5	132	1.079E+07	1.21	8761.7	1.8743	16422
13	P7 - Output Parameter	<input checked="" type="checkbox"/>	13	P7 - Output Parameter	1	132	1.072E+07	2.36	11208	1.7183	19259
14	Min-Max Search	<input checked="" type="checkbox"/>									
15	Refinement										
18	Quality										
22	Response Points										

Note:

If you have discrete input parameters, an alert displays before a Min-Max search is performed, reminding you that the search can be time-consuming. If you do not want this message to display, you can clear the **Confirm if Min-Max Search can take a long time** check box on the **Design Exploration** tab in the **Options** window. For more information, see [Design Exploration Options](#) (p. 34).

The algorithm used to search the output parameter space for the minimum and maximum values depends on the input parameters.

- If at least one input parameter is continuous with manufacturable values, MISQP is used.
- Otherwise, NLPQL is used.

Before updating your response surface, set the options for the Min-Max search. In the **Outline** pane, select **Min-Max Search**.

When there are only discrete parameters, in the **Properties** pane, **Number of Initial Samples** and **Number of Start Points** are not available because they are not applicable. DesignXplorer computes the number of parameter combinations and then sorts the sample points to get the minimum and maximum values. For example, if there are only two discrete input parameters (with four and three levels respectively), 12 sample points (4×3) are sorted to get the minimum and maximum values.

When there is at least one continuous parameter, in the **Properties** pane, **Number of Initial Samples** and **Number of Start Points** are available.

- For **Number of Initial Samples**, enter the size of the initial sample set to generate in the space of continuous input parameters.
- For **Number of Start Points**, enter the number of starting points that the Min-Max search algorithm is to use. Using more starting points lengthens the search times.

If all input parameters are continuous, one search is done for the minimum and one for the maximum. When the number of starting points is greater than one, two searches (minimum and maximum) are done for each starting point. To find the minimum or maximum of an output, the generated sample points are sorted and the n first points of the sort are used as the starting points. The search algorithm is then run twice for each starting point, once for the minimum and once for the maximum.

Example 1

Assume the following:

- There are only input parameters.
- All input parameters are continuous.
- **Number of Initial Samples** is set to **100**.
- **Number of Start Points** is set to **1**.

DesignXplorer generates 100 initial sample points and then sorts them to get the starting points. For each starting point, a local optimization is done.

If **Number of Start Points** is set to **3**, six local optimization are done (three for the minimum and three for the maximum).

When there are discrete input parameters, the number of searches increases by the number of parameter combinations. There are two searches per combination, one for the minimum and one for the maximum.

Example 2

Assume the following:

- There are two discrete input parameters (with three and four levels respectively).
- There are two continuous input parameters.
- **Number of Initial Samples** is set to **100**.
- **Number of Start Points** is set to **3**.

With multiple starting points, the number of searches is multiplied accordingly. For three starting points, six local optimization are done (three for the minimum and three for the maximum) for each combination. The total number of discrete combinations is equal to 12 (3×4). For each combination, DesignXplorer generates 100 initial sample points in the space of the continuous parameters and where discrete parameter values are fixed to the discrete combination values. Consequently, 12×100 initial sample points are generated and 12×6 local optimizations are done.

Search Results

Once your response surface is updated, selecting **Min-Max Search** in the **Outline** pane displays the sample points that contain the minimum and maximum values calculated for each output parameter in the response surface table. The minimum and maximum values in the output parameter **Properties** pane are also updated based on the results of the search. If the response surface is updated in any way, including changing the fitting for an output parameter or performing a refinement, a new Min-Max search is performed.

You can save the sample points shown in the response surface table by selecting **Insert as Design Points** from the context menu. You can also save sample points as response points (or as refinement points to improve the response surface) by selecting **Explore Response Surface at Point** from the context menu.

The sample points obtained from the Min-Max search are used by the Screening optimization. If you run a Screening optimization, the samples are automatically taken into account in the sample set used to run or initialize the optimization. For more information, see [Performing a Screening Optimization \(p. 181\)](#).

You can disable the Min-Max search by clearing the check box in the **Enabled** column in the **Outline** pane. If you disable the Min-Max search, no search is done when the response surface is updated. If you disable the search after performing an initial search, the results from the initial search remain in the output parameter properties and are shown in the **Table** pane for the response surface when you select **Min-Max Search** in the **Outline** pane.

Note:

- If no solutions have occurred yet, no minimum or maximum values are shown.
- If the **Design of Experiments** cell is solved but the **Response Surface** cell is not solved, the minimum and maximum values for each output parameter are extracted from the DOE solution's design points and displayed in the **Properties** pane for the output parameters.
- For discrete parameters and continuous parameters with manufacturable values, there is only one minimum and maximum value per output parameter, even if a discrete parameter has many levels. There is not one Min-Max value set per combination of parameter values.

Quality Metrics for Response Surfaces

Quality metrics for response surfaces include [goodness of fit \(p. 130\)](#) and [verification points \(p. 140\)](#). Used together, they enable you to assess and improve upon the quality of your response surface. You can view both metrics on the [Predicted vs. Observed chart \(p. 142\)](#).

Goodness of Fit for Output Parameters in a Response Surface

You can view the goodness of fit for the output parameters in a response surface. In the **Outline** pane for the response surface, select **Quality** → **Goodness Of Fit**.

A response surface is built from design points in the DOE and refinement points, which are collectively called *learning points*. Calculations for the goodness of fit compare the response surface outputs with the DOE results used to create them.

For response surface types that try to find the best fit of the response surface to DOE points (such as Standard Response Surface - Full 2nd-Order Polynomial), you can get an idea of how well the fit was accomplished. However, for interpolated response surface methods that force the response surface to pass through all of the DOE points (such as Kriging), the goodness of fit usually appears to be perfect. In this case, goodness of fit indicates that the response surface passed through the DOE points used to create it, but it does not indicate whether the response surface captures the parametric solution.

If any of the input parameters is discrete, a different response surface is built for each combination of the discrete levels and the quality of the response surface might be different from one configuration to another.

Goodness of fit is closely related to the response surface type used to generate the response surface. If the goodness of fit is not of the expected quality, you can try to improve it by changing the response surface. For more information, see [Changing the Response Surface \(p. 124\)](#).

To add a new goodness of fit object, right-click **Quality** and select **Insert Goodness of Fit**. Right-click the object to copy and paste, delete, or duplicate.

Goodness of Fit Criteria

The **Table** and **Chart** panes display goodness-of-fit metrics for each output parameter.

Note:

Criteria marked as advanced options are visible in the goodness of fit table only if you've selected the **Show Advanced Options** check box on the **Design Exploration** tab in the **Options** window. For more information, see [Design Exploration Options \(p. 34\)](#).

Several criteria are calculated for the points taken into account in the construction of the response surface. The mathematical representations in the descriptions for these criteria use the following notation:

y_i = value of the output parameter at the i -th sampling point

\hat{y}_i = value of the regression model at the i -th sampling point

\bar{y} = arithmetic mean of the values y_i

σ_y = standard deviation of the values y_i

N = number of sampling points

P = number of polynomial terms for a quadratic response surface (not counting the constant term)

Coefficient of Determination (R^2)

The coefficient of determination is the percent of the variation of the output parameter that can be explained by the response surface regression equation. It is the ratio of the explained variation to the total variation. The best value is 1.

The points used to create the response surface are likely to contain variation for each output parameter, unless all output values are the same, which results in a flat response surface. This variation is illustrated by the response surface that is generated. If the response surface were to pass directly through each point, which is the case for the Kriging response surface, the coefficient of determination would be 1, meaning that all variation is explained.

The coefficient of determination is mathematically represented as:

$$1 - \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2}$$

Adjusted Coefficient of Determination (Adjusted R²)

The adjusted coefficient of determination is an advanced option that is available only for standard response surfaces. It takes the sample size into consideration when computing the coefficient of determination. The best value is 1. When the number of samples is small (<30), the adjusted coefficient of determination is usually more reliable than the coefficient of determination.

The adjusted coefficient of determination is mathematically represented as:

$$1 - \frac{N-1}{N-P-1} \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2}$$

Maximum Relative Residual

The maximum relative residual is an advanced option. Relatively speaking, it is the maximum distance out of all generated points from the calculated response surface to each generated point. The best value is 0%. In general, the closer the value is to 0%, the better quality of the response surface.

However, in some situations, you can have a larger value and still have a good response surface. For example, this can be true when the mean of the output values is close to zero.

The maximum relative residual is mathematically represented as:

$$\text{Max}_{i=1:N} \left(\text{Abs} \left(\frac{y_i - \hat{y}_i}{y_i} \right) \right)$$

Root Mean Square Error

For regression methods, the root mean square error is the square root of the average square of the residuals at the DOE points. The best value is 0. In general, the closer the value is to 0, the better quality of the response surface.

The root mean square error is mathematically represented as:

$$\sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}$$

Relative Root Mean Square Error

The relative root mean square error is an advanced option. It is the square root of the average square of the residuals scaled by the actual output values at the points for regression methods. The best value is 0%. In general, the closer the value is to 0%, the better quality of the response surface.

However, in some situations, you can have a larger value and still have a good response surface. For example, this can be true when some of the output values are close to zero. You can obtain 100% of relative error if the observed value = 1e-10 and the predicted value = 1e-8. However, if the range of output values is 1, this error becomes negligible.

The relative root mean square error is mathematically represented as:

$$\sqrt{\frac{1}{N} \sum_{i=1}^N \left(\frac{y_i - \hat{y}_i}{y_i} \right)^2}$$

Relative Maximum Absolute Error

The relative maximum absolute error is the absolute maximum residual value relative to the standard deviation of the actual output data, modified by the number of samples. The best value is 0%. In general, the closer the value is to 0%, the better quality of the response surface.

The relative maximum absolute error and the relative average absolute error correspond to the maximum error and average absolute error scaled by the standard deviation. For example, the relative root mean square error becomes negligible if both of these values are small.

The relative maximum absolute error is mathematically represented as:

$$\frac{1}{\sigma_y} \text{Max}_{i=1:N} \left(\text{Abs}(y_i - \hat{y}_i) \right)$$

Relative Average Absolute Error

The relative average absolute error is the average of the residuals relative to the standard deviation of the actual outputs. This value is useful when the number of samples is low (<30). The best value is 0%. In general, the closer the value is to 0%, the better quality of the response surface.

The relative average absolute error and the relative maximum absolute error correspond to the maximum error and average absolute error scaled by the standard deviation. For example, the relative root mean square error becomes negligible if both of these values are small.

The relative average absolute error is mathematically represented as:

$$\frac{1}{\sigma_y} \frac{1}{N} \sum_{i=1}^N \text{Abs}(y_i - \hat{y}_i)$$

Reviewing Goodness of Fit Information

For all response surface types, goodness of fit is calculated for learning points and verification points. Learning points are the DOE points and refinement points used to generate the response surface. You can display or hide the verification points in the **Table** and **Chart** panes by changing the settings in the **Properties** pane.

If a response surface has discrete input parameters, you can view the response surface for each combination of discrete levels by selecting the discrete input values in the **Properties** pane.

For a Genetic Aggregation response surface, goodness of fit is also calculated for learning points by using the cross-validation technique. The advantages of cross-validation are that it assesses:

- Stability of a response surface without having to calculate additional verification points
- Richness of the DOE (instead of only measuring the quality of the fit)

When cross-validation produces bad metrics, it means that there are not enough points.

For more information on how goodness of fit is calculated for the different types of response surface points, see [Goodness of Fit Calculations \(p. 139\)](#).

Goodness of Fit Display Properties

When **Goodness of Fit** is selected in the **Outline** pane under **Quality**, the **Properties** pane displays properties for controlling the display in the goodness of fit table, the Predicted vs. Observed chart, and the Workbench project report.

Properties of Outline A18: Goodness Of Fit			
	A	B	C
1	Property	Value	<input type="checkbox"/> Enabled
2	[-] Chart		
3	Display Parameter Full Name	<input checked="" type="checkbox"/>	
4	Show Verification Points	<input checked="" type="checkbox"/>	
5	Show Learning Points	<input checked="" type="checkbox"/>	
6	[-] Output Parameters		
7	P4 - Equivalent Stress Maximum		<input type="checkbox"/>
8	P5 - Total Deformation Maximum		<input type="checkbox"/>
9	P6 - Safety Factor Minimum		<input checked="" type="checkbox"/>

Chart

This category provides display properties for charts.

- **Display Parameter Full Name:** Specifies whether to display the full parameter name.
- **Show Verification Points:** Specifies whether to display verification points.

- **Show Learning Points:** Specifies whether to display learning points.

Input Parameters

This category is shown only when there are discrete input parameters. You can select a discrete value to view the associated response surface.

General

This category is shown only when **Response Surface Type** is set to **Standard Response Surface - Full 2nd-Order Polynomial** and the **Show Advanced Options** check box is selected on the **Design Exploration** tab in the **Options** window. For more information, see [Design Exploration Options](#) (p. 34).

The single advanced property under this category, **Confidence Level**, is used post-modeling as an input for assessing the goodness of fit. The confidence level indicates how likely an output parameter being estimated falls within the confidence interval. The default setting is **0.95**, which means that the interval is calculated so that the true value will fall within the interval 95 out of 100 times. However, when this advanced option is shown, you can set any value between **0** and **1**.

Output Parameters

This category displays all output parameters so that you can indicate which to display.

Goodness of Fit Results

When **Goodness of Fit** is selected in the **Outline** pane under **Quality**, the [Predicted vs. Observed chart](#) (p. 142) and the [goodness of fit table](#) (p. 130) are shown.

For a Standard Response Surface - Full 2nd-Order Polynomials response surface, you can generate the [Advanced Goodness of Fit report](#) (p. 136) for the selected output parameter.

Goodness of Fit Table

In the goodness of fit table, each parameter is rated on how close it comes to the ideal value for each goodness of fit metric. The rating is indicated by the number of gold stars or red crosses next to the parameter. The worst rating is three red crosses. The best rating is three gold stars.

Table of Schematic C3: Response Surface				
	A	B	C	D
1		P6 - Geometry Mass	P14 - Safety Factor HOOP Minimum	P15 - Safety Factor PIPE Minimum
2	☐ Coefficient of Determination (Best Value = 1)			
3	Learning Points	★★★ 1	★★★ 0.99997	★★★ 0.99998
4	Cross-Validation on Learning Points	★★★ 1	★★★ 0.9994	★★★ 0.99949
5	☐ Root Mean Square Error (Best Value = 0)			
6	Learning Points	1.8876E-06	0.0044675	0.00051469
7	Verification Points	4.4587E-06	0.01796	0.00025673
8	Cross-Validation on Learning Points	5.0531E-06	0.020459	0.0023788
9	☐ Relative Maximum Absolute Error (Best Value = 0%)			
10	Learning Points	★★★ 0	★★ 1.3133	★★ 1.2211
11	Verification Points	★★★ 0	★ 2.1574	★★★ 0.24427
12	Cross-Validation on Learning Points	★★★ 0.011966	— 5.3961	— 6.4131
13	☐ Relative Average Absolute Error (Best Value = 0%)			
14	Learning Points	★★★ 0	★★★ 0.41866	★★★ 0.36291
15	Verification Points	★★★ 0	★ 2.1574	★★★ 0.24427
16	Cross-Validation on Learning Points	★★★ 0.0027904	★ 2.0526	★★ 1.7183

Note:

The root mean square error has no rating because it is not a bounded characteristic.

When calculated for different types of response surface points, goodness-of-fit metrics quantify different aspects of your response surface. You can interpret your goodness of fit table as follows:

Type of Point	Goodness of Fit Measure
Learning Point	Quality of the interpolation
Verification Point	Quality of the prediction
Cross-Validation Learning Point	Stability or reliability of the response surface

For the Genetic Aggregation response surface, goodness-of-fit metrics calculated for learning points via the cross-validation technique provide an especially effective way to assess the stability of your response surface without needing to add new verification points. If the metrics are good, you can be confident in the quality of your model. If the metrics are not good, you know that you need to enrich your model by adding new refinement points.

Advanced Goodness of Fit Report

The Advanced Goodness of Fit report is a text-based report that shows goodness-of-fit metrics for the response surface of a selected output parameter. This report can help you to assess whether your response surface is reliable and accurate enough for you to proceed with confidence. It is available for an output parameter only in a response surface generated when **Response Surface Type** is set to **Standard Response Surface - Full 2nd-Order Polynomials**.

Generating the Advanced Goodness of Fit Report

To generate the Advanced Goodness of Fit report for a given output parameter:

1. In the **Outline** pane for the response surface, select **Quality** → **Goodness of Fit**.
2. In the **Table** pane, right-click the column for the desired output parameter and select **Generate Advance Goodness of Fit Report**.

Reviewing the Advanced Goodness of Fit Report

Once generated, the Advanced Goodness of Fit report displays in a separate window. The following properties provide general information about analyzing the goodness of fit:

Regression Model

Type of polynomial regression. **Cross Quadratic Surface** means that the response surface uses constant terms, linear terms (X), pure quadratic terms ($X_i * X_i$) and cross-quadratic terms ($X_i * X_j$). You cannot change this property.

Input Transformation

Applied transformation on continuous input parameters before solving the response surface. To change the transformation type, [advanced properties \(p. 34\)](#) must be shown. In the **Outline** pane for the **Response Surface** cell, select **Response Surface**. Then, in the **Properties** pane under **Meta Model**, change the selection for **Inputs Transformation Type**. This property and the subsequent advanced property, **Inputs Scaling**, apply to all continuous input variables, with and without manufacturable values. If you clear the check box for **Inputs Scaling**, no scaling occurs of the data for input parameters.

Output Transformation

Applied transformation on the given output parameter before solving the response surface. To change the transformation type for the output parameter, in the **Outline** pane for the **Response Surface** cell, select it. Then, in the **Properties** pane under **Output Settings**, change the selection for **Transformation Type**. Transformations do not apply to derived output parameters. If [advanced properties \(p. 34\)](#) are shown, you can also change the **Scaling** property. When you clear this check box, no scaling occurs of the data for this output parameter.

Analysis Type

Algorithm used to select the relevant regression terms. For **Modified Linear Forward Stepwise Regression**, the individual regression terms are iteratively added to the regression model if they are found to cause a significant improvement of the regression results. A partial F-test is used to determine the significance of the individual regression terms. If a regression term is not significant, it is ignored.

Filtering Significance Level

Used to filter out insignificant regression terms. A regression term is ignored when its probability of being zero is greater than **Filtering Significance Level** and when its contribution to the regression sum of squares is insignificant. For example, if **Filtering Significance**

Level is 0.05, a regression term is ignored when its probability of being zero is greater than 0.05 and its probability (based on F-test) to contribute to the sum of squares is less than 0.95 (1 – 0.05). The contribution of a regression term to the regression model is evaluated by using the F-test and the **Filtering Significance Level**.

The following properties provide all the scaling models, transformation models, and regression term coefficients used to build the regression model:

Scaling Model for all Defined Input Variables

Scaling models used to build the regression model.

Transformation Model for all Defined Input Variables

Transformation models used to build the regression model.

Coefficient of Selected Regression Terms

Regression term coefficients used to build the regression model. These coefficients are displayed in a table with the following auxiliary values. The **Confidence Level** property for **Goodness of Fit** controls the probability value. For more information, see [Goodness of Fit Display Properties](#) (p. 134).

- **Std. Dev. of Coefficient:** Statistical estimation of the dispersion of the coefficient. A low standard deviation indicates stability of the coefficient. A high standard deviation indicates that the coefficient has a wide range of potential values.
- **Prob. Coeff. =0:** Corresponds to the probability that the coefficient is zero. If this probability is high, it suggests that the coefficient is insignificant. More significant coefficients have a lower probability of being zero.
- **Confidence Interval [Lower Bound; Upper Bound]:** Corresponds to the confidence interval of each coefficient. For example, if the confidence level is 0.95, there is a 95% probability that the coefficient is in this range. The smaller the confidence interval, the more likely it is that the coefficient is accurate.

The following properties provide statistical data that enable you to measure the stability of the regression model and to understand which input parameters are significant.

Variation Inflation Factor (VIF) of Selected Regression Terms

VIF measures the degree of multi-collinearity of the i -th independent variable with the other independent variables in the regression model. If the **Maximum VIF for Full Regression Model** value is very high (>10), there is a high interdependency among the terms and the response surface is not unique. If the response surface is not reliable in spite of a good R^2 value, you should add more points to enrich the response surface.

Correlation Coefficient Matrix of Selected Regression Terms

Pearson correlation of selected regression terms. A non-diagonal term measures the collinearity between two independent variables. If the absolute value of this term is close to 1, the response surface is not unique. As with a high VIF value, if the response surface is not

reliable in spite of the good R^2 value, you should add more points to enrich the response surface.

Comparison of Samples and Approximated Output Values

Each row of this table corresponds to a sample point used to build the response surface. The **Confidence Level** property for **Goodness of Fit** controls the probability value. For more information, see [Goodness of Fit Display Properties \(p. 134\)](#).

- **Residual Value:** Difference between the sample value and the approximated value.
- **Approx. Std Dev:** Standard deviation of the predicted value at this sample point. If the standard deviation is small, the predicted value is statistically stable. Consequently, you can have confidence in the predicted values in the area close to this point.
- **Lower Bound and Upper Bound:** Bounds of the confidence interval of the approximated value. For example, if the confidence level is 0.95, there is a 95% probability that the approximated value is in this range. The smaller the confidence interval, the more likely it is that the approximated value is accurate.

Sample-Based Goodness-of-Fit Measures

Each row of this table corresponds to a sample point used to build the response surface.

- **Hat Matrix Diagonal:** These elements are the leverages that describe the influence each observed value has on the fitted value for the same observation.
- **Student's Residual:** Corresponds to the residual divided by an estimate of its standard deviation.
- **Student's Deleted Residual:** Student's residual with the i -th observation removed.
- **Cook's Distance:** Measures the effect of the i -th observation on all fitted values. A point with a value greater than 1 can be considered influential.
- **P-value:** Probability of obtaining a test statistic at least as extreme as the one that was actually observed, assuming that the null hypothesis is true. This value represents the error percentage you can make if you reject the null hypothesis, where the null hypothesis is that the sample point has not influenced the observed result. Traditionally, following Fisher, one rejects the null hypothesis if the p-value is less than or equal to a specified significance level, often 0.05, or more stringent values, such as 0.02 or 0.01.

Scalar Goodness of Fit Measures

Summary of previous data and additional metrics commonly used to estimate the quality of the response surface. For more information, see [Goodness of Fit for Output Parameters in a Response Surface \(p. 130\)](#).

Goodness of Fit Calculations

Goodness-of-fit metrics are calculated for learning points, for verification points, and by cross-validating learning points. As indicated earlier, learning points are the DOE points and refinement points

used to generate the response surface. Verification points are not used in the generation of the response surface.

Learning Points

For learning points, goodness of fit measures the quality of the response surface interpolation. To calculate the goodness of fit, the error between the predicted and observed values is calculated for each learning point.

Verification Points

For verification points, goodness of fit measures the quality of the response surface prediction. To calculate the goodness of fit, the verification points are placed in locations that maximize the distance to the learning points. After the verification points are calculated with a design point update, the differences between verification point values and predicted values are calculated. For more information, see [Verification Points \(p. 140\)](#).

Learning Points with Cross-Validation

Goodness of fit obtained via cross-validation measures the stability and reliability of the response surface. To calculate goodness of fit for cross-validation learning points, the cross-validation computes the error between the observed value and the predicted value for each point. However, the predicted value is evaluated on a model built with all DOE points except the associated observed value. For example, the first response surface is built without the first design point, the second response surface is built without the second design point, and so on.

Mathematical notation follows:

- (X_i, Y_i) the i -th learning point where X and Y are respectively the observed input and output parameter values.
- S the response surface built with all the learning points.
- \hat{Y}_i the predicted value of the response surface S at X_i : $\hat{Y}_i = S(X_i)$.
- S_{-i} the response surface built with all the learning points except the i -th learning point.
- \hat{Y}_{-i} the predicted value of the sub-response surface S_{-i} at X_i : $\hat{Y}_{-i} = S_{-i}(X_i)$.

Standard goodness of fit on learning points compares Y_i to \hat{Y}_i . Goodness of fit based on cross-validation compares Y_i to \hat{Y}_{-i} .

For more information, see [Genetic Aggregation \(p. 333\)](#) in the theory section.

Verification Points

Verification points enable you to verify that the response surface accurately approximates the output parameter values. They compare the predicted and observed values of the output parameters. After the response surface is created, the verification points are placed in locations that maximize the distance from existing DOE points and refinement points (Optimal Space-Filling algorithm). Verification

points can also be added manually or imported from a CSV file. For more information, see [Creating Verification Points \(p. 141\)](#).

A design point update is a real solve that calculates each verification point. These verification point results are then compared with the response surface predictions and the difference is calculated.

Verification points are useful in validating any type of response surface. In particular, however, you should always use verification points to validate the accuracy of interpolated response surfaces, such as Kriging or Sparse Grid.

Creating Verification Points

You can create verification points using any of the following methods:

Generating Verification Points Automatically

You can specify that verification points are generated automatically by selecting the **Generate Verification Points** check box in the **Properties** pane for the response surface. When this check box is selected, **Number of Verification Points** can be set to the number of verification points to be generated.

Creating Verification Points (Input Values Only)

You can insert a new verification point directly in the verification points table:

1. In the **Outline** pane for the response surface, select **Quality** → **Verification Points**.
2. In the verification points table, enter the values of the input parameters into the **New Verification Point** row.
3. Update the verification points and goodness-of-fit metrics by right-clicking the row and selecting **Update**.

Creating Verification Points Manually (Input and Output Values)

By default, the output parameters of the verification point table are grayed out and filled in only by a real solve. However, to change the output values, you can change the editing mode of the verification points table as described in [Editable Output Parameter Values \(p. 314\)](#). This allows you to enter verification points manually, rather than by performing a real solve, and still compare them with the response surface.

Creating Verification Points via Import

You can import data from a CSV file by right-clicking and selecting **Import Verification Points**. This is a way to compare either experimental data or data run in another simulation with the simulation response surface. You can also copy information from a CSV input file and paste them into the verification points table.

Verification Points Results

When **Verification Points** is selected in the **Outline** pane, the verification points table and the Predicted vs Observed chart show the verification points generated by the response surface update.

- The verification points table shows the input and output values for each verification point generated.
- The Predicted vs Observed chart shows the values predicted from the response surface versus the values observed from the design points. For more information, see [Predicted vs. Observed Chart \(p. 142\)](#).

Verification points are not used to build the response surface until they are turned into refinement points and the response surface is recalculated. If a verification point reveals that the current response surface is of a poor quality, you can insert it as a refinement point so that it is taken into account to improve the accuracy of the response surface.

1. In the **Outline** pane for the response surface, select **Quality** → **Verification Points**.
2. In the verification points table, right-click the verification point and select **Insert as Refinement Point**.

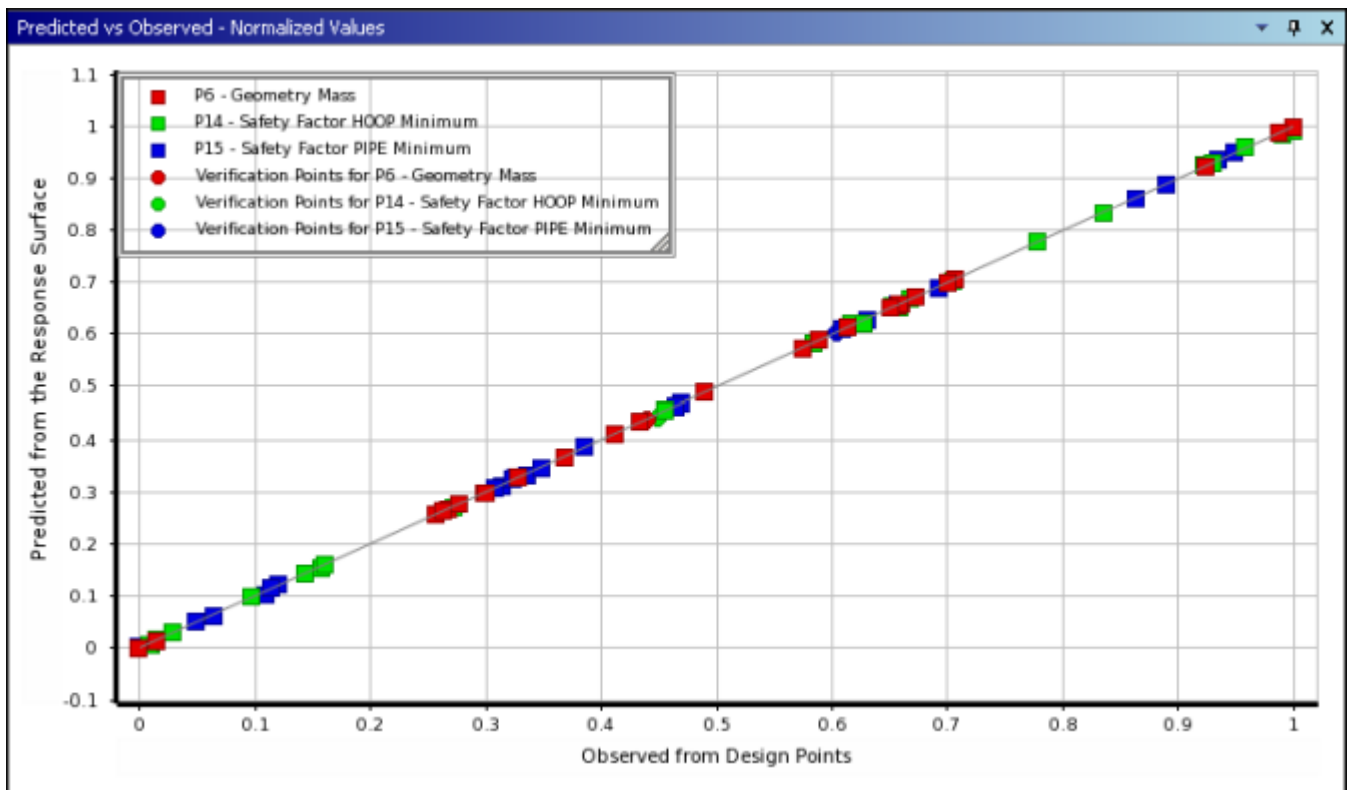
Note:

The **Insert as Refinement Point** option is not available for Kriging and Sparse Grid response surfaces.

Predicted vs. Observed Chart

The Predicted vs Observed chart shows the values predicted from the response surface versus the values observed from the design points. This scatter chart enables you to quickly determine if the response surface correctly fits the points of the design points table and refinement table. The closer the points are to the diagonal line, the better the response surface fits the points.

To view the Predicted vs. Observed chart, in the **Outline** pane for the response surface, under **Quality**, select either **Goodness Of Fit** or **Verification Points**. A chart is available for each goodness of fit object. The display is determined by your selections in the **Properties** pane.



By default, all output parameters are displayed on the chart, and the output values are normalized. However, if only one output parameter is plotted, the output values are not normalized.

Verification points are not used in the generation of the response surface. Consequently, if they appear close to the diagonal line, the response surface is correctly representing the parametric model. Otherwise, not enough data is provided for the response surface to detect the parametric behavior of the model. You must refine the response surface.

If you position the mouse cursor over a point of the chart, the corresponding parameter values appear in the **Properties** pane, including the predicted and observed values for the output parameters.

If you right-click a point on the chart, you can select from the context menu options available for this point. For example, you can insert the point as a refinement point or response point in the response surface table, which is a good way to improve the response surface around a verification point with an insufficient goodness of fit.

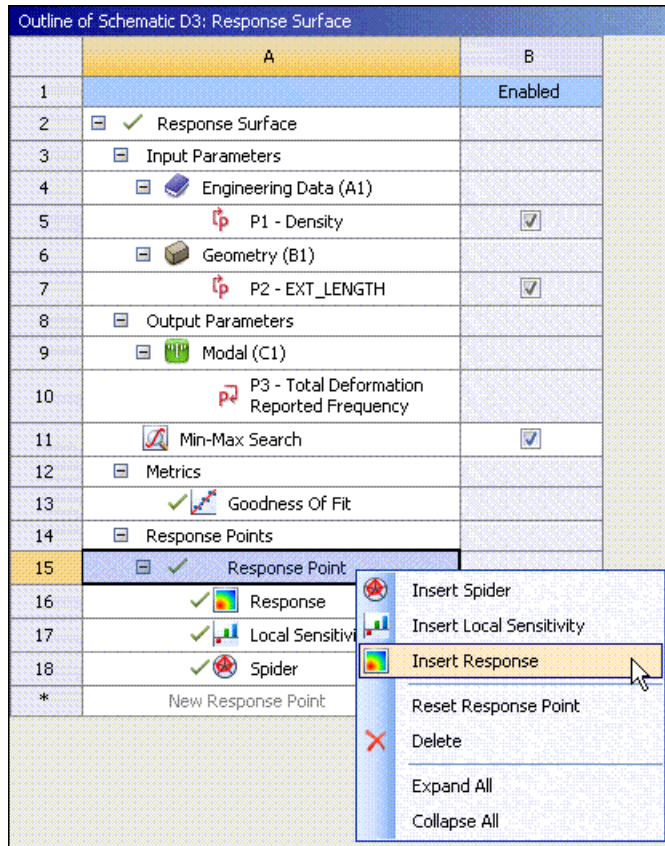
Response Surface Charts

Response surface charts facilitate the exploration of the design space by graphically depicting the effect that parameters have on one another. The following charts are available: Spider chart, Response chart, Local Sensitivity chart, and Local Sensitivity Curves chart.

When a response surface is updated, one response point and one of each of the chart types are created automatically. You can insert as many response points and charts as you want. To add a new chart:

1. Right-click a response point cell in the **Outline** pane.
2. Select the insert option for the type of chart that you want to add.

If an instance of the chart already exists for any response point in the **Outline** pane, additional instances of the chart have numbers appended to their names. For example, the second and third instances of a Response chart are named **Response 1** and **Response 2**.



To duplicate a chart that already exists in the **Outline** pane, right-click the chart and select **Duplicate**. This operation creates a duplicate chart under the same response point.

Note:

Chart duplication triggers a chart update. If the update succeeds, both the original chart and the duplicate are up-to-date.

Once you've created a chart, you can change the name of a chart cell by double-clicking it and entering the new name. This does not affect the title of the chart, which is set as part of the chart properties. You can also save a chart as a graphic by right-clicking it and selecting **Save Image As**. For more information, see [Saving a Chart](#) in the Ansys Workbench documentation.

Each chart provides the ability to visually explore the parameter space using options provided in the chart's **Properties** pane. For discrete parameters and continuous parameters with manufacturable values, you use drop-down menus. For continuous variables, you use sliders. The sliders allow you to modify an input parameter value and view its effect on the displayed output parameter.

All of the Response charts under a response point in the **Chart** pane use the same input parameter values because they are all based on the current parameter values for that response point. Thus, when you modify the input parameter values, the response point and all of its charts are refreshed to take the new values into account.

Using the Response Chart

The Response chart is a graphical representation that allows you to see how changes to each input parameter affect a selected output parameter. In this chart, you select an output parameter to be displayed on one axis, and (depending on the chart mode used) select one or more input parameters to display on the other axes.

Once your inputs and output are selected, you can use the sliders or enter values to change the values of the input parameters that are not selected to explore how these parameters affect the shape and position of the curve.

Additionally, you can opt to display all the design points currently in use (from the DOE and the response surface refinement) by selecting the **Show Design Points** check box in the **Properties** pane. With a small number of input parameters, this option can help you to evaluate how closely the response surface fits the design points in your project.

Response Chart Modes

The Response chart has three different viewing modes:

- **2D**: Displays a two-dimensional contour graph that allows you to view how changes to a single input affect a single output. For more information, see [Using the 2D Response Chart \(p. 148\)](#).
- **3D**: Displays a three-dimensional contour graph that allows you to view how changes to two inputs affect a single output. For more information, see [Using the 3D Response Chart \(p. 149\)](#).
- **2D Slices**: Displays a three-dimensional surface in an easy-to-read two-dimensional format by compressing data, thereby providing the benefits of both the 2D and 3D contour graphs. This graph displays an input on the X axis, an input on the slice axis, and a selected output on the Y axis. The input on the slice axis is calculated from the number of slices defined, the number of manufacturable values defined, or the number of discrete parameter levels defined. For more information, see [Using the 2D Slices Response Chart \(p. 151\)](#).

When you solve a response surface, a Response chart is automatically added for the default response point in the **Outline** pane for the response surface. You can add an additional response chart for either the default response point or a different response point. To add another Response chart, right-click the desired response point in the **Outline** pane and select **Insert Response**.

Understanding the Response Chart Display

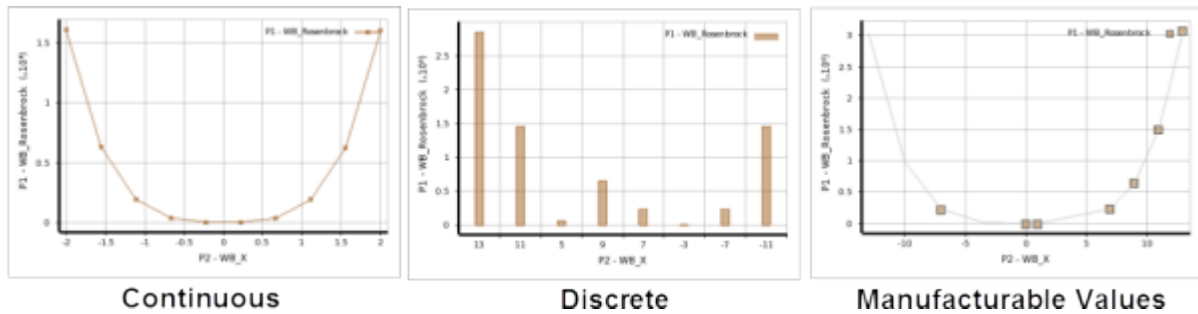
The graphical rendering of the Response chart varies according to the type of the parameters selected. The parameter type depends on the **Classification** property and the usage of manufacturable values. When two input parameters are selected for a Response chart, the display is further determined by the specific combination of parameter types. This section describes how parameters of different types and various combinations are rendered in the three Response chart modes.

Display by Parameter Type

Input parameters can be continuous or discrete. You can limit continuous input variables to only specified manufacturable values. The Response chart displayed for each these three input types is different. The following examples show how the graphical rendering reflects the nature of the parameter values.

- Continuous parameters are represented by colored curves that reflect the continuous nature of the parameter values.
- Discrete parameters are represented by bars that reflect the discrete nature of the parameter values. There is one bar for each discrete value.
- Continuous parameters with manufacturable values are represented by a combination of curves and markers. Transparent gray curves reflect the continuous values, and colored markers reflect the discrete nature of the manufacturable values. There is one marker for each manufacturable value.

The following examples show how each type of parameter is displayed on the **2D** Response chart.

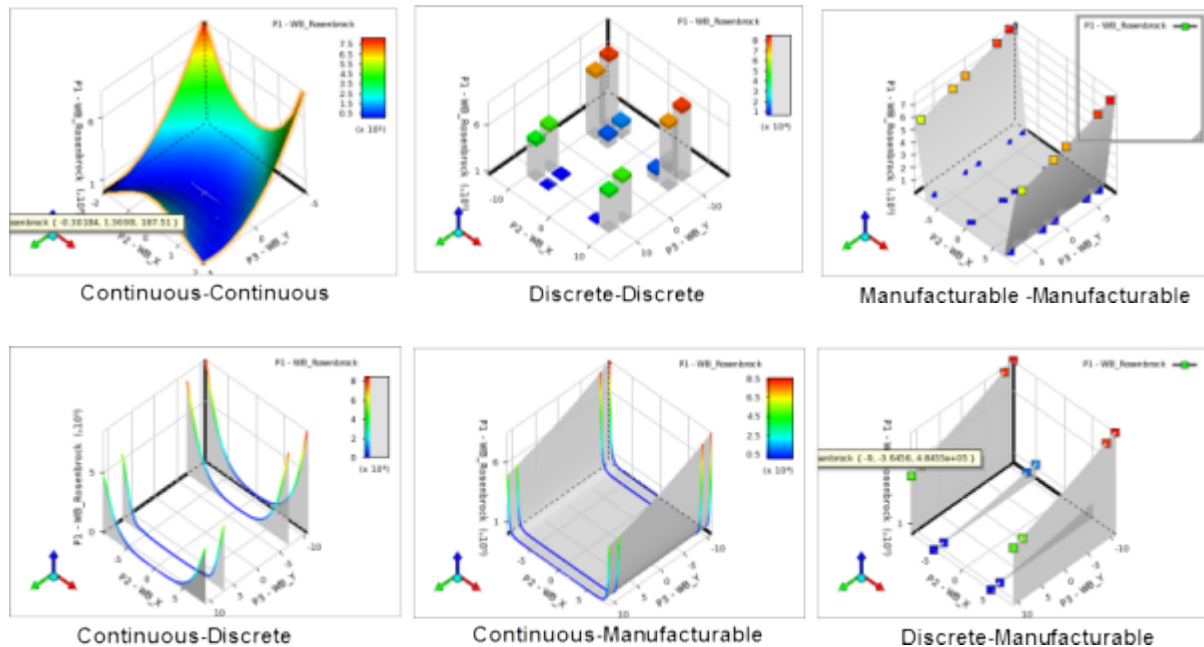


Display by Parameter Combination

When two input parameters are selected for the Response chart, different parameter type combinations are displayed differently, with a graphical rendering that reflects the nature of each parameter.

3D Response Chart Parameter Combinations

The 3D Response chart has two inputs and can have combinations of parameters with like types or unlike types. Response chart examples follow for possible combinations.

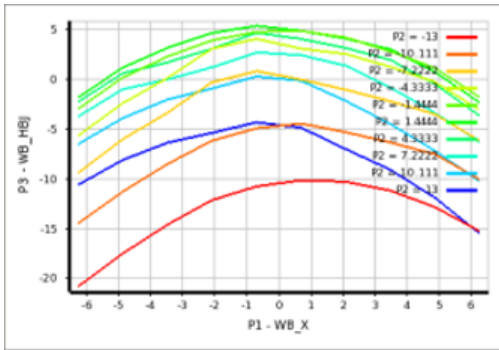


2D Slices Response Chart Parameter Combinations

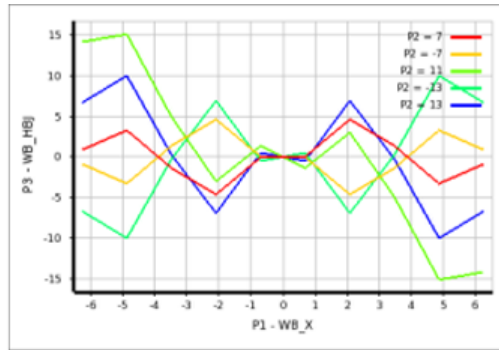
The 2D Slices Response chart has two inputs. Combinations of these inputs are categorized first by the parameter type of the select input (the **X Axis**) and then further distinguished by the parameter type of the calculated input (the **Slice Axis**). For each X-axis parameter type, there are two different renderings:

- The X axis in conjunction with continuous values (a continuous parameter). In this instance, you specify the number of curves or "slices".
- The X axis in conjunction with discrete values (either a discrete parameter or a continuous parameter with manufacturable values). In this instance, the number of slices is automatically set to the number of discrete levels or the number of manufacturable values.

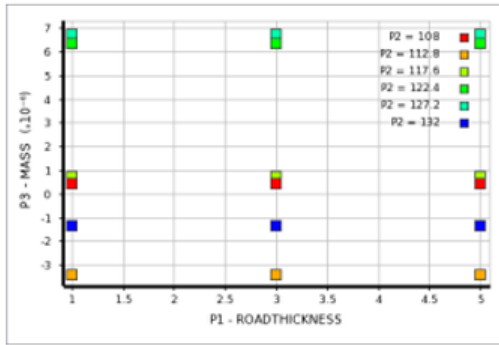
The following Response chart examples show possible combinations.



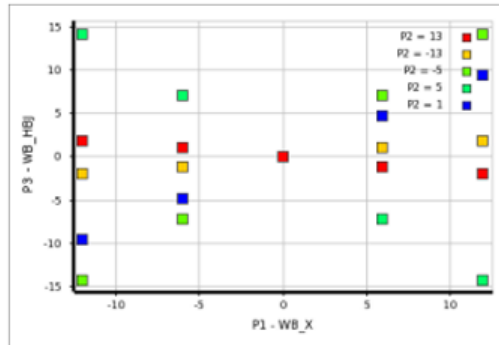
Continuous-Continuous



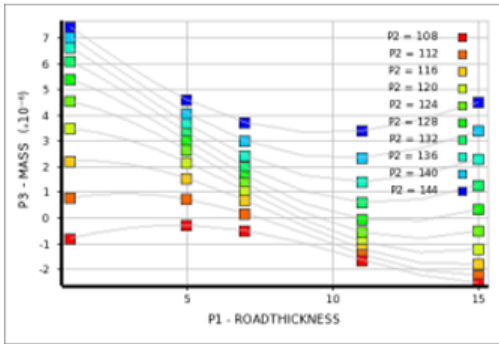
Continuous-Discrete OR
Continuous-Manufacturable



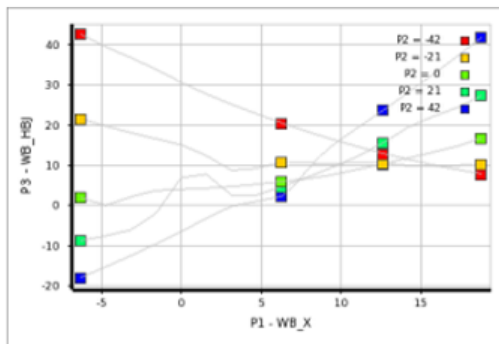
Discrete-Continuous



Discrete-Discrete OR
Discrete-Manufacturable



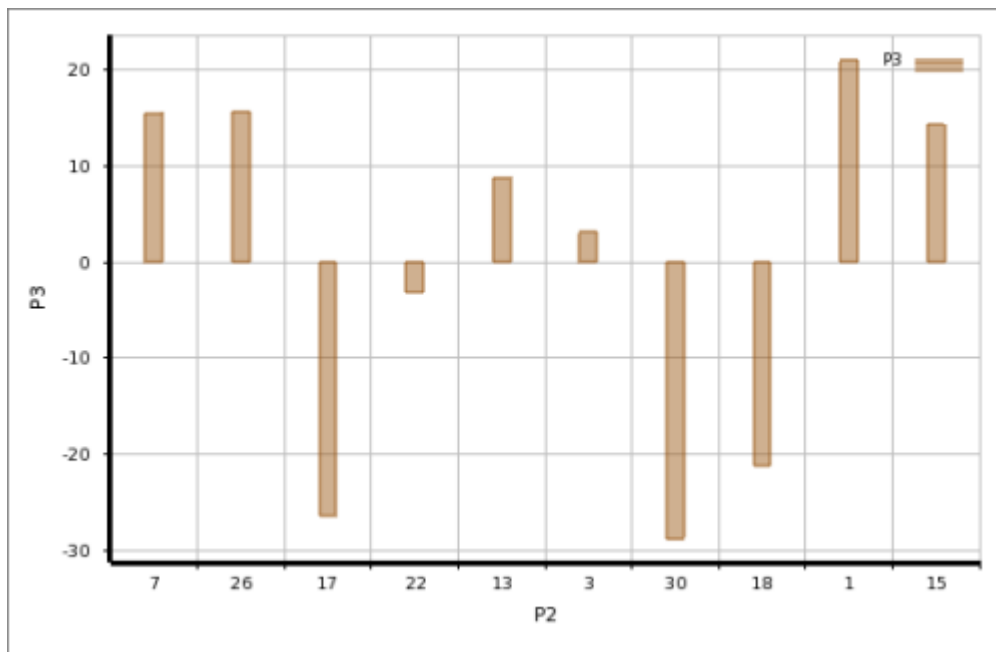
Manufacturable-Continuous



Manufacturable-Discrete OR
Manufacturable-Manufacturable

Using the 2D Response Chart

A 2D Response chart is a two-dimensional contour graph that allows you to view how changes to a single input affect a single output. Essentially, it is a flattened representation of a three-dimensional Response chart, displaying a selected input on the X axis and a selected output on the Y axis.



Viewing the 2-D Response Chart

To view a 2D Response chart:

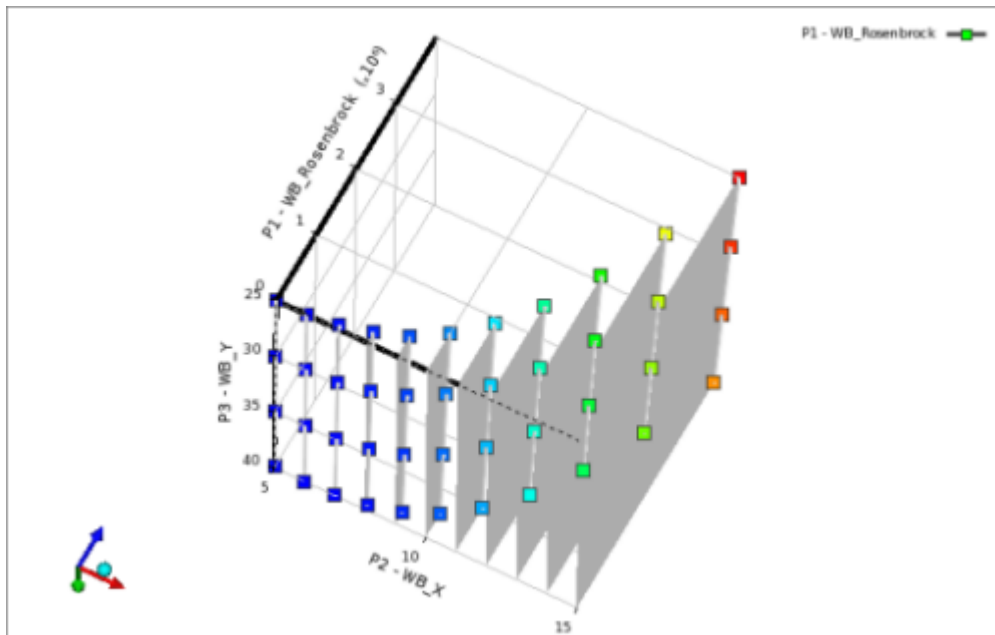
1. In the **Outline** pane for the response surface, select the Response chart object.
2. In the **Properties** pane, under **Chart**:
 - For **Mode**, select **2D**.
 - For **Chart Resolution Along X**, specify the resolution.
3. In the **Properties** pane, under **Axes**:
 - For **X Axis**, select an input parameter.
 - For **Y Axis**, select an output parameter.

The Response chart automatically updates according to your selections.

For more information on available properties, see [Response Chart: Properties \(p. 156\)](#).

Using the 3D Response Chart

The 3D Response chart is a three-dimensional contour graph that allows you to view how changes to two inputs affect a single output. It displays a selected input on the X axis, a selected input on the Y axis, and a selected output on the Z axis.



Viewing the 3-D Response Chart

1. In the **Outline** pane for the response surface, select the Response chart object.
2. In the **Properties** pane, under **Chart**:
 - For **Mode**, select **3D**.
 - For **Chart Resolution Along X** and **Chart Resolution Along Y**, specify the resolutions.
3. In the **Properties** pane, under **Axes**:
 - For **X Axis**, select an input parameter.
 - For **Y Axis**, select an input parameter.
 - For **Z Axis**, select an output parameter.

The Response chart automatically updates according to your selections. A smooth three-dimensional contour of Z versus X and Y displays.

For more information on available properties, see [Response Chart: Properties](#) (p. 156).

Manipulating the 3D Response Chart

In the **Chart** pane, the 3D Response chart can be rotated by clicking and dragging the mouse. Moving the cursor over the response surface shows the values of Z, X, and Y. The values of other input parameters can also be adjusted in the **Properties** pane using the sliders, showing different contours of Z. Additionally, the corresponding values of other output parameters can be seen in the **Properties** pane. Therefore, instantaneous design and analysis is possible, leading to the generation of additional design points.

The triad control at the bottom left of the **Chart** pane allows you to rotate the 3D response chart in freehand mode or quickly view the chart from a particular plane. To zoom in or out on any part

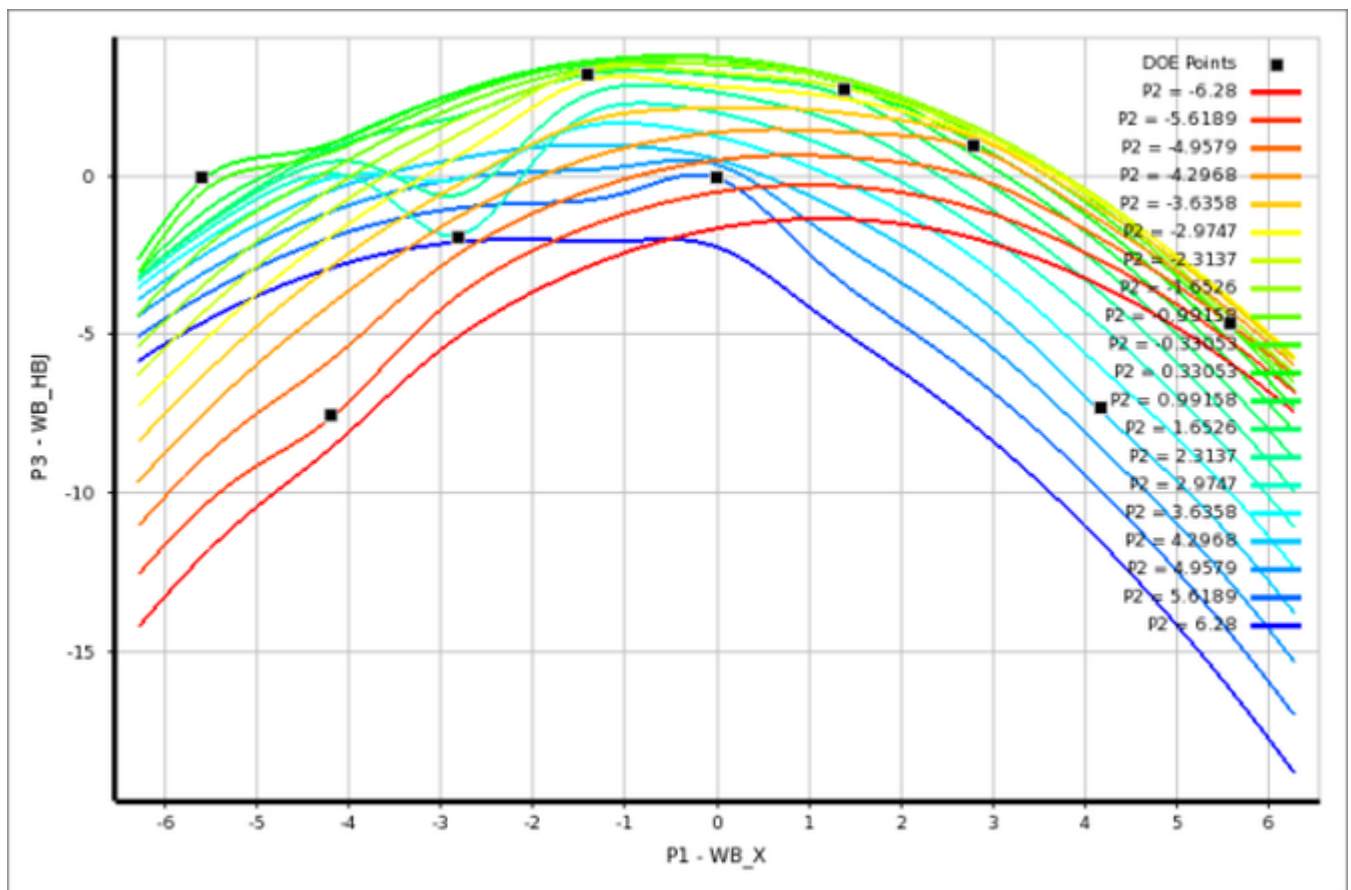
of the chart, use the shift-middle mouse button or scroll wheel. See [Setting Chart Properties](#) for details.

Using the 2D Slices Response Chart

The 2D Slices Response chart combines the benefits of the 2D and 3D Response charts by representing a three-dimensional surface in a two-dimensional format. It displays an input on the X axis, an input on the slice axis, and an output on the Y axis.

The value of the input on the slice axis is calculated from the number of curves defined for the X and Y axes.

- When one or both of the inputs are continuous parameters, you specify the number of slices to display.
- When one or both of the input parameters are either discrete or continuous with manufacturable values, the number of slices is determined by the number of levels defined for the input parameters.



Essentially, the first input on the X axis is varying continuously, while the number of curves, or slices, defined for the slice axis represents the second input. Both inputs are then displayed on the XY plane, with regard to the output parameter on the Y axis. You can think of the 2D Slices Response chart as a projection of the 3D response surface curves onto a flat surface.

Viewing the 2D Slices Response Chart

To view a 2D Slices Response chart:

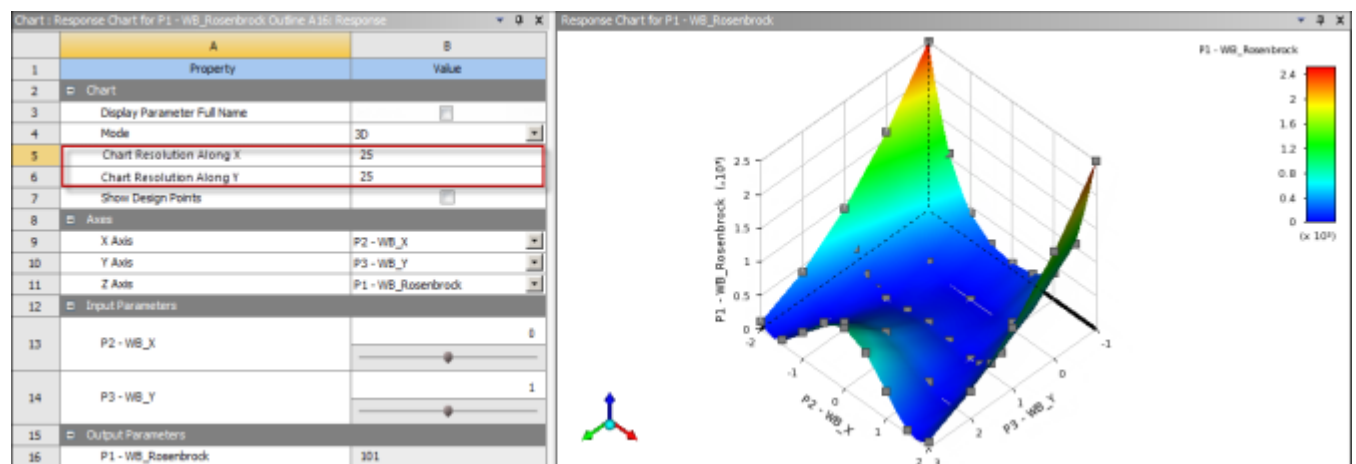
1. In the **Outline** pane for the response surface, select the Response Chart object.
2. In the **Properties** pane, under **Chart**:
 - For **Mode**, select **2D Slices**.
 - For **Chart Resolution Along X**, specify the resolution.
 - For **Number of Slices**, specify the number of slices to display.
3. In the **Properties** pane, under **Axes**:
 - For **X Axis**, select an input parameter.
 - For **Slice Axis**, select an input parameter.
 - For **Y Axis**, select an output parameter.

The Response chart automatically update according to your selections.

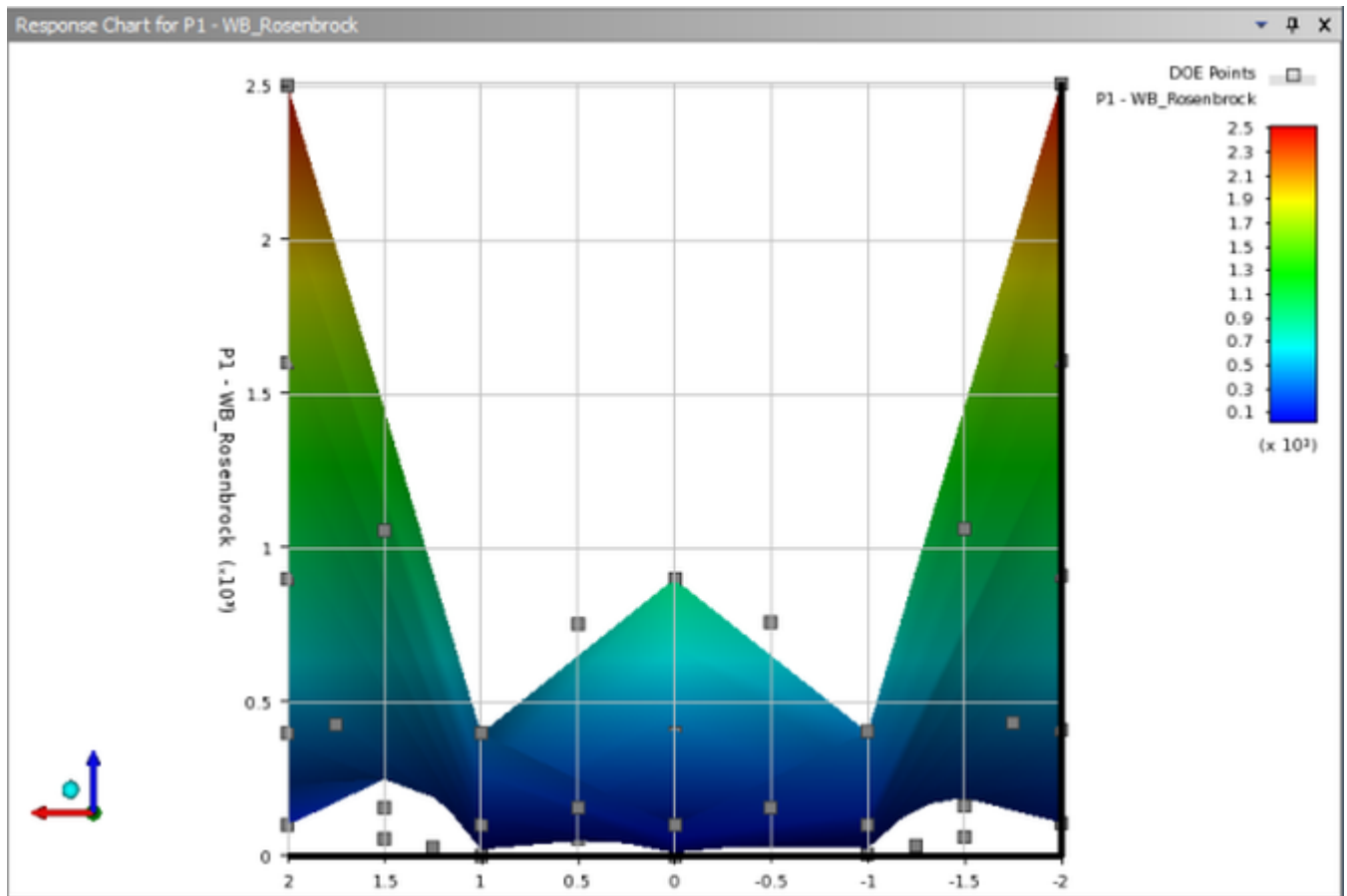
For more information on available properties, see [Response Chart: Properties \(p. 156\)](#).

2D Slices Response Chart Rendering: Example

To illustrate how the 2D Slices Response chart is rendered, this example starts with a 3D Response chart. Both inputs are continuous parameters. Chart resolution properties default to 25, which means there are 25 points on the X axis and 25 points on the Y axis.

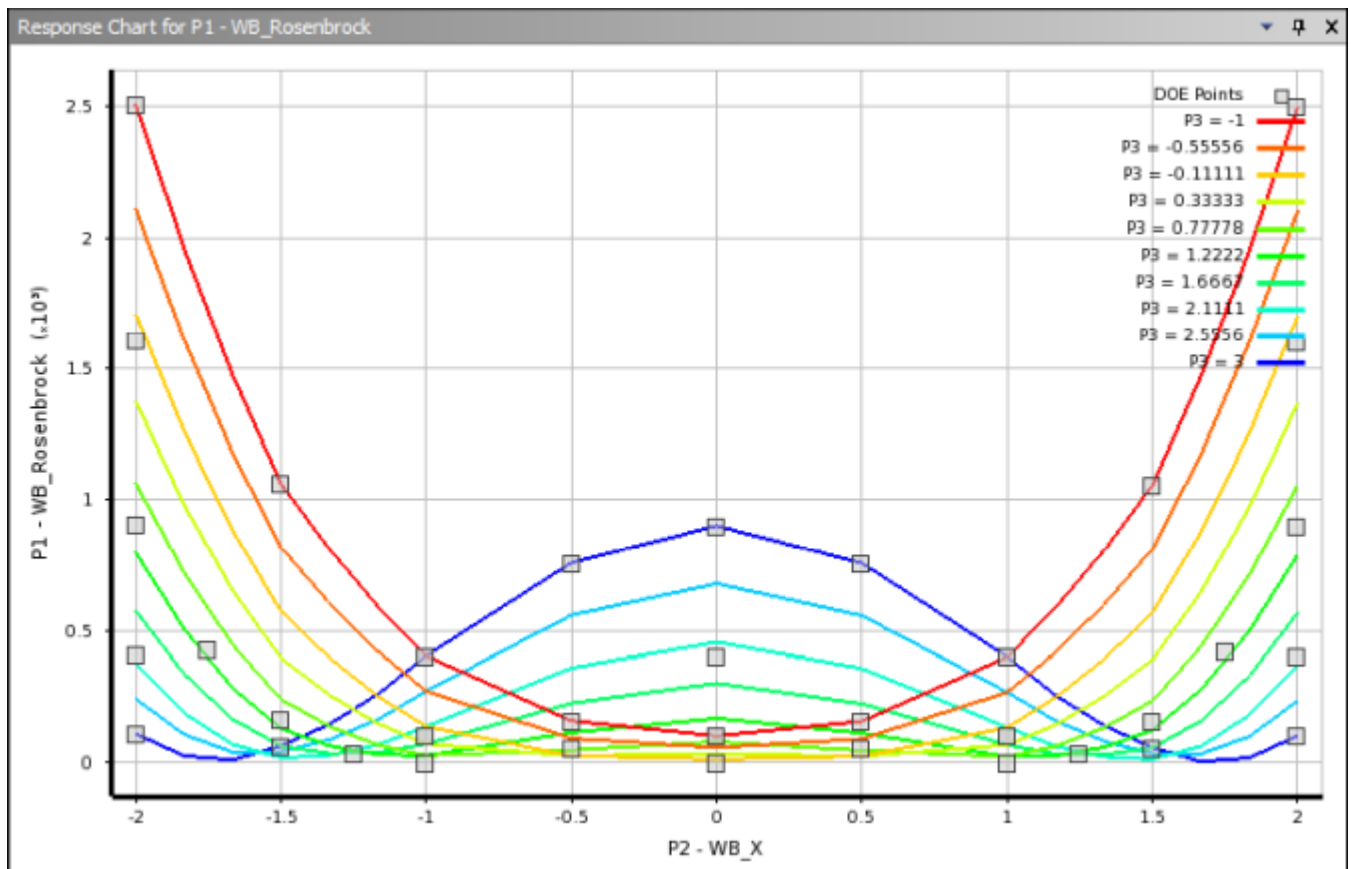


The 3D image is then rotated so that the X axis is along the bottom edge of the chart, mirroring the perspective of the 2D Slices Response chart for a better comparison.



Finally, **Mode** is switched to **2D Slices**. By comparing the following chart to the 3D version, you can see how the 2D Slices Response chart is actually a two-dimensional rendering of a three-dimensional image. From the following example, you can see observe the following:

- Along the Y axis, there are 10 slices, corresponding to the value for **Number of Slices**.
- Along the X axis, each slice intersects with 25 points, corresponding to the value for **Chart Resolution Along X**.



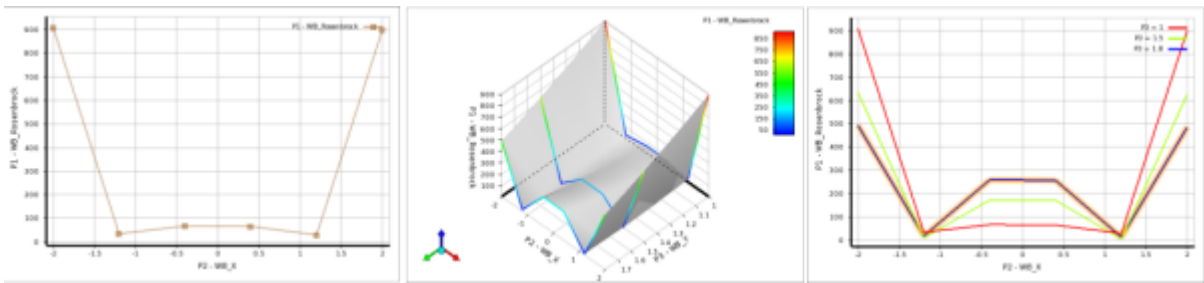
Response Chart: Example

This example uses three different modes of Response charts to demonstrate the effect of input parameters on an output parameter.

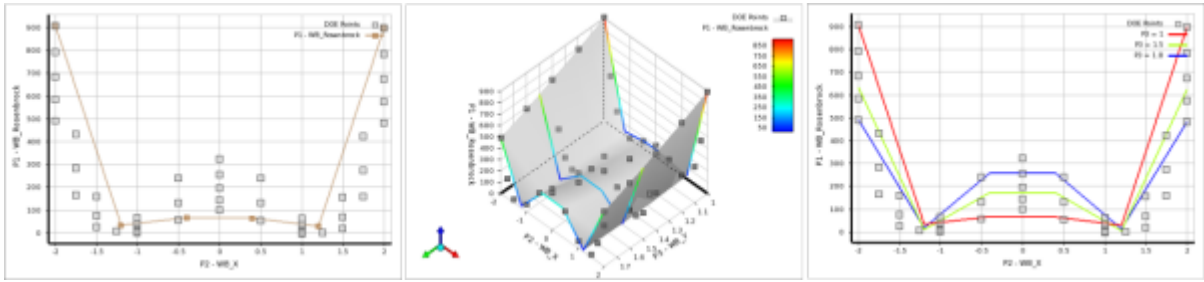
- DOE Type: Sparse Grid Initialization
- Input Parameter 1: WB_X (continuous, with range of -2; 2)
- Input Parameter 2: WB_Y (continuous with manufacturable values of 1, 1.5, and 1.8, with a range of 1; 1.8)
- Output Parameter: WB_Rosenbrock
- Design Points: Six design points on the X axis and six design points on the Y axis.

Because there is an input with manufacturable values, the number of slices is determined by the number of levels defined.

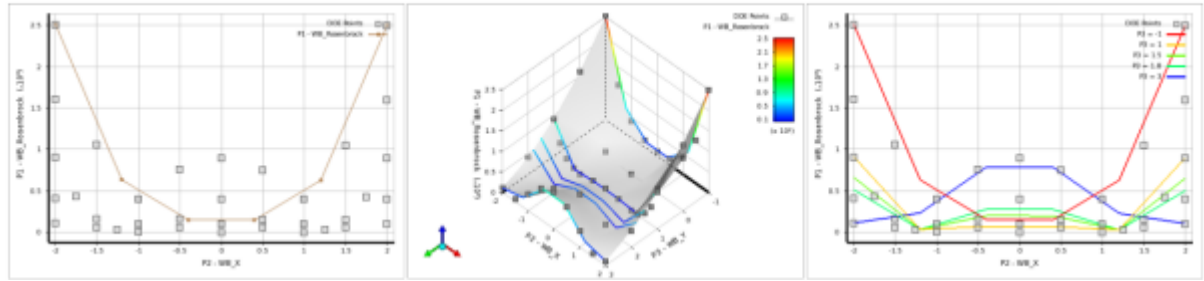
The following figures show the initial Response chart in **2D**, **3D**, and **2D Slices** modes.



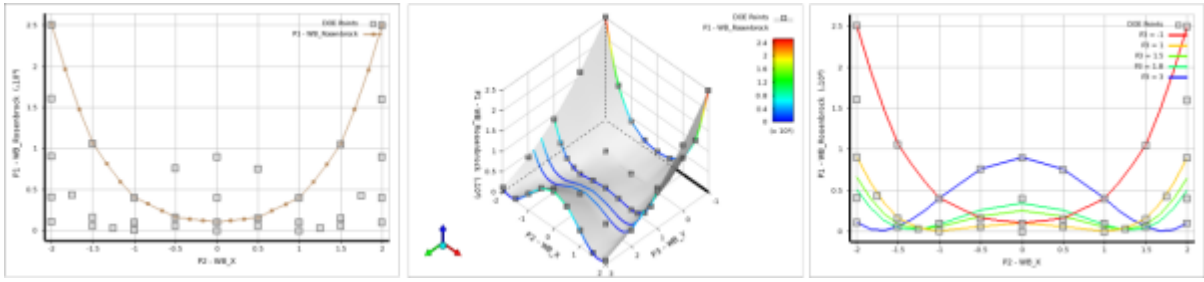
To display the design points from the DOE and the response surface refinement, select the **Show Design Points** check box in the **Properties** pane. The design points are superimposed on your chart.



When working with manufacturable values, you can improve chart quality by extending the parameter range. Here, the range is increased by adding manufacturable values of -1 and 3, which become the upper and lower bounds.



To improve the quality of your chart further, you can increase the number of points used in building it by entering values in the **Properties** pane. In the following figures, the number of points on the X and Y axes are increased from 6 to 25.



Response Chart: Properties

To specify the properties of a Response chart, select **Response** in the **Outline** pane for the response surface and then edit the properties in the **Properties** pane. The properties available depend on the chart mode and the type of parameters selected.

Chart Properties

Determines the properties of the chart.

- **Display Parameter Full Name:** Specifies whether to show the full parameter name or the short parameter name.
- **Mode:** Determines whether the chart is in **2D**, **3D**, or **2D Slices** mode.
- **Chart Resolution Along X:** Determines the number of points on the X axis. The number of points controls the amount of curvature that can be displayed. A minimum of 2 points is required and produces a straight line. A maximum of 100 points is allowed for maximum curvature. The default is 25.
- **Chart Resolution Along Y:** Determines the number of points on the Y axis (3D and 2D Slices modes only). The number of points controls the amount of curvature that can be displayed. A minimum of 2 points is required and produces a straight line. A maximum of 100 points is allowed for maximum curvature. The default is 25.
- **Number of Slices:** Determines the number of slices displayed in the 2D Slices chart.
- **Show Design Points:** Determines whether all of the design points currently in use, both in the Design of Experiments and from the response surface refinement, are used to build the response surface.

Axes Properties

Determines the data to display on each chart axis. For each axis, under **Value**, you can change what the chart displays on an axis by selecting an option from the drop-down list.

- For **X Axis**, available options are each of the input parameters enabled in the project.
- For **Y Axis**, the available options are:
 - For the **2D** mode, each of the output parameters in the project.
 - For the **3D** mode, each of the input parameters enabled in the project.
 - For the **2D Slices** mode, each of the output parameters in the project.
- For the **Z Axis (3D mode only)**, available options are each of the output parameters in the project.
- For **Slice Axis (2D Slices mode only)**, available options are each of the input parameters enabled in the project. This property is available only when both input parameters are continuous.

Input Parameters

For each input parameter, you can change the value.

- For continuous parameters, you move the slider. The number to the right of the slider represents the current value.
- For discrete parameters or continuous parameters with manufacturable values, you use the keyboard to enter a new value.

Output Parameters

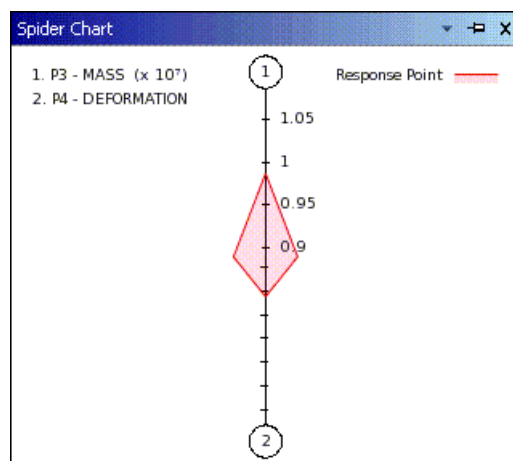
For each output parameter, you can view the interpolated value.

Generic Chart Properties

You can modify generic chart properties for this chart. For more information, see [Setting Chart Properties](#).

Using the Spider Chart

Spider charts allow you to visualize the effect that changing the input parameters has on all of the output parameters simultaneously. When you solve a response surface, a Spider chart appears in the **Outline** pane for the default response point.



You can use the slider bars in the **Properties** pane for the chart to adjust values for input parameters to visualize different designs. You can also enter specific values. In the top left of the **Chart** pane, the parameter legend box allows you to select the parameter that is in the primary (top) position. Only the axis of the primary parameter is labeled with values.

Using Local Sensitivity Charts

Local Sensitivity charts allow you to see the effect of continuous input parameters (including those with manufacturable values) on output parameters. At the response surface level, sensitivity charts are *single parameter sensitivities*. This means that design exploration calculates the change of the outputs based on the change of inputs independently, at the current value of each input parameter. The larger the change of the output parameters, the more significant is the role of the input parameters that were varied. As such, single parameter sensitivities are *local sensitivities*.

Types of Sensitivity Charts

DesignXplorer has two types of sensitivity charts: the standard Local Sensitivity chart and the Local Sensitivity Curves chart.

- The Local Sensitivity chart is a powerful project-level tool, allowing you see at a glance the effect of *all* the input parameters on output parameters. For more information, see [Using the Local Sensitivity Chart \(p. 160\)](#).
- The Local Sensitivity Curves chart helps you to further focus your analysis by allowing you to view independent parameter variations within the standard Local Sensitivity chart. It provides a means of viewing the effect of each input on specific outputs, given the current values of other parameters. For more information, see [Using the Local Sensitivity Curves Chart \(p. 164\)](#).

When you solve a response surface, a Local Sensitivity chart and a Local Sensitivity Curves chart are automatically added for the default response point in the **Outline** pane for the response surface. To add another chart (this can be either an additional chart for the default response point or a chart for a different response point), right-click the desired response point in the **Outline** pane and select either **Insert Local Sensitivity** or **Insert Local Sensitivity Curves**.

Manufacturable Values in Local Sensitivity Charts

Local sensitivity charts calculate sensitivities for continuous parameters, including those with manufacturable values. They require that you have at least one continuous parameter. If all of your input parameters are discrete, Local Sensitivity and Local Sensitivity Curves charts are not available.

For information on how continuous parameters with manufacturable values are represented on local sensitivity charts, see [Understanding the Local Sensitivities Display \(p. 158\)](#).

Discrete Parameters in Local Sensitivity Charts

Discrete parameters cannot be enabled as chart variables. However, their effect on the output can still be displayed on the chart. When there is a discrete parameter in the project, the chart calculates the sensitivities of continuous values (including those with manufacturable values) given their specific combination with the discrete parameter values.

For discrete parameters, the **Properties** pane includes a drop-down menu that is populated with the discrete values defined for the parameter. By selecting different discrete values for each parameter, you can explore the different sensitivities given different combinations of discrete values. The chart is updated according to the changed parameter values. You can check the sensitivities in a single chart, or you can create multiple charts to compare different designs.

Understanding the Local Sensitivities Display

Local sensitivity charts display continuous parameters, including those with manufacturable values. This section illustrates how local sensitivity charts render values for continuous parameters and continuous parameters with manufacturable values.

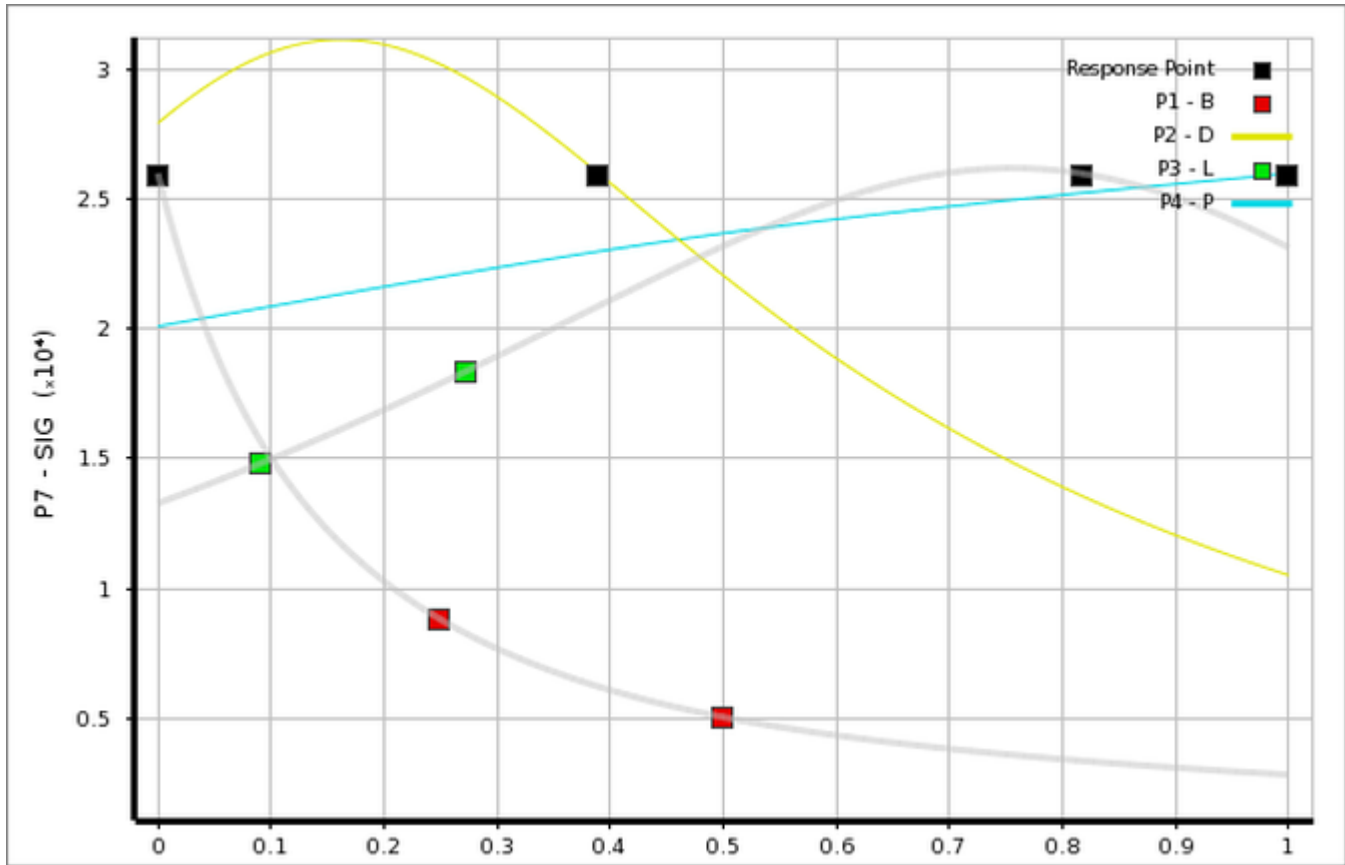
Local Sensitivity Curves Chart Display

On the Local Sensitivity Curves chart, continuous parameters are represented by colored curves, with a different color for each parameter.

For continuous parameters with manufacturable values, continuous values are represented by a transparent gray curve, while manufacturable values are represented by colored markers.

In the following figure:

- Input parameters **P2-D** and **P4-P** (the yellow and blue curves) are continuous parameters.
- Input parameters **P1-B** and **P3-L** (the gray curves) are continuous parameters with manufacturable values.
- The colored markers indicate the manufacturable values defined.
- The black markers indicate the location of the response point on each of the input curves.



Local Sensitivity Chart Display

On the Local Sensitivity chart, continuous parameters are represented by colored bars, with a different color for each parameter.

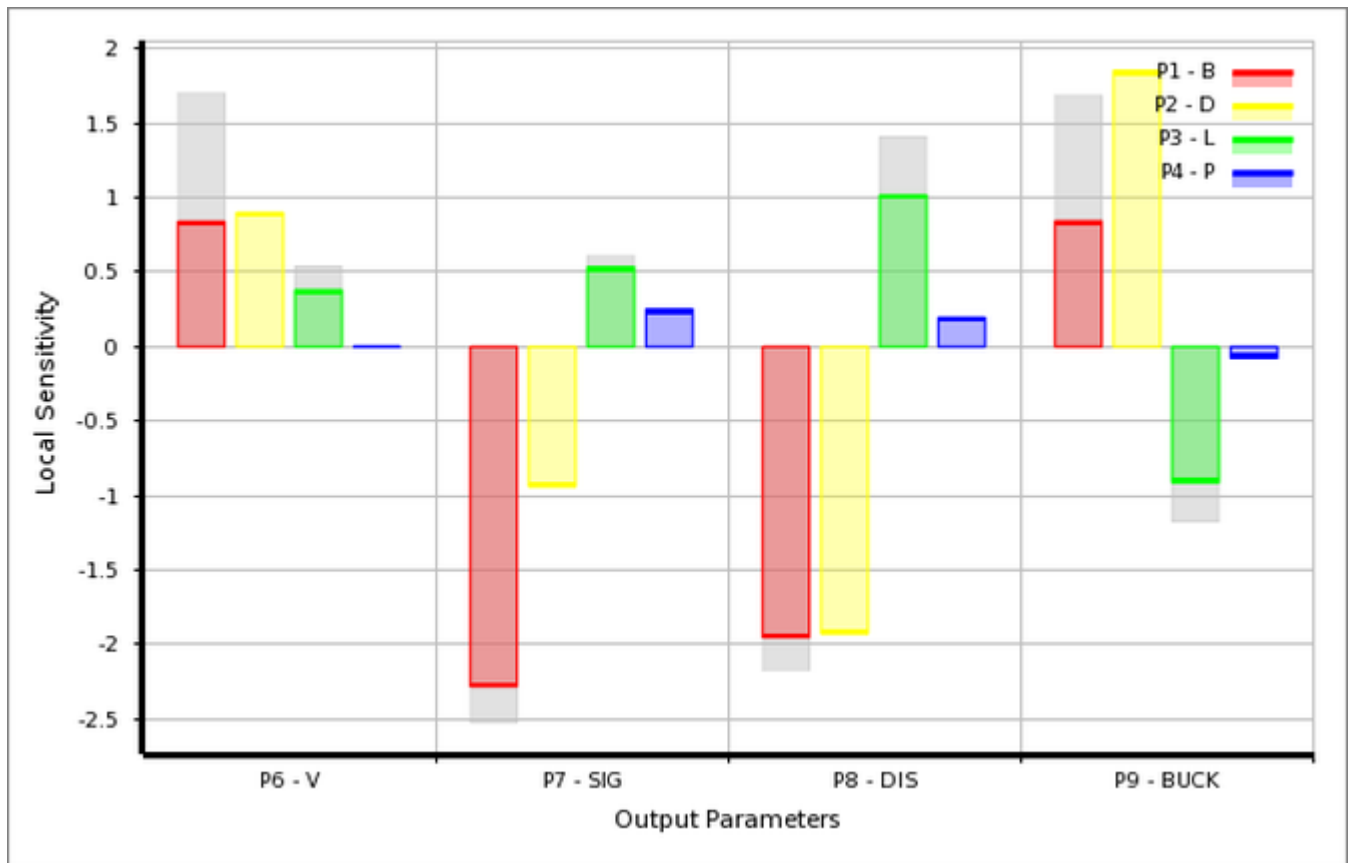
For continuous parameters with manufacturable values, continuous values are represented by a gray bar, while manufacturable values are represented by a colored bar in front of the gray bar. Each bar is defined with the minimum and maximum extracted from the manufacturable values

and the average calculated from the support curve. The minimum and maximum of the output can vary according to whether or not manufacturable values are used. In this case, both the colored bar and the gray bar for the input are visible on the chart.

Also, if the parameter range extends beyond the manufacturable values that are defined, the bar is topped with a gray line to indicate the sensitivity obtained while ignoring the manufacturable values.

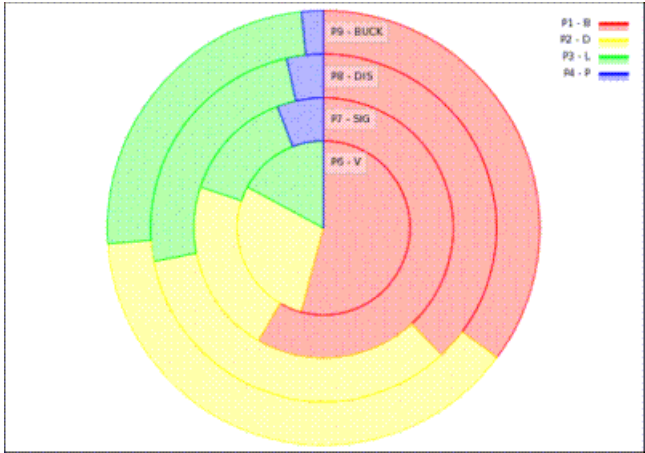
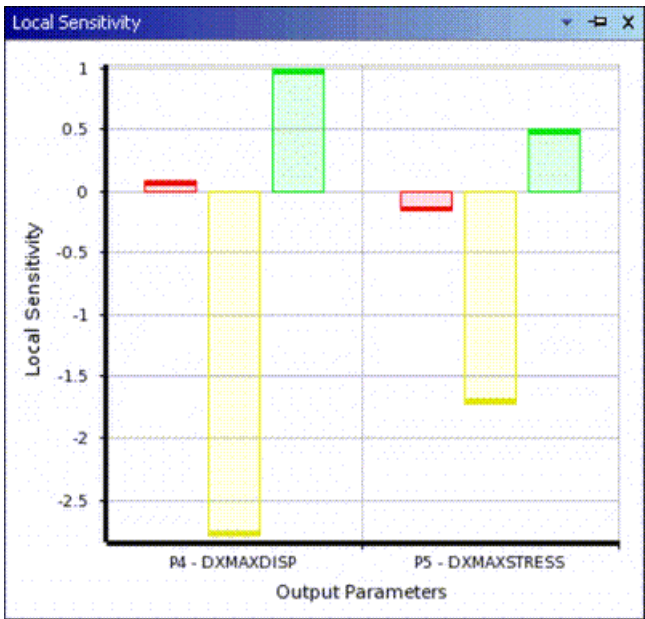
In the following figure:

- Input parameters **P2-D** and **P4-P** (the yellow and blue bars) are continuous parameters.
- Input parameters **P1-B** and **P3-L** (the red and green bars) are continuous with manufacturable values.
- The bars for inputs **P1-B** and **P3-L** show differences between the minimum and maximum of the output when manufacturable values are used (the colored bar in front) versus when they are not used (the gray bar in back, now visible).



Using the Local Sensitivity Chart

The Local Sensitivity chart can be a powerful exploration tool. For each output, it allows you to see the weight of the different input. This chart calculates the change of the output based on the change of each input independently, at the current value of each input parameter in the project. You can display the Local Sensitivity chart as a bar chart or pie chart.

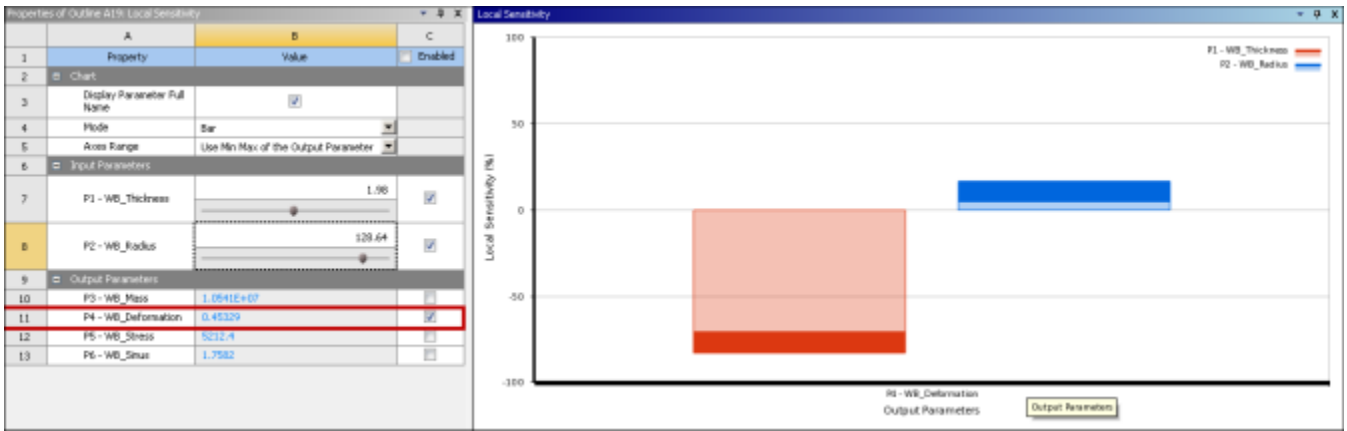


By default, the Local Sensitivity chart shows the effect of all input parameters on all output parameters, but it is also possible to specify the inputs and outputs to be considered. If you consider only one output, the resulting chart provides an independent sensitivity analysis for each single input parameter. To specify inputs and outputs, select or clear the **Enabled** check box.

For more information on available properties, see [Local Sensitivity Chart: Properties \(p. 163\)](#).

Local Sensitivity Chart: Example

An example follows of the Local Sensitivity chart displayed as a bar chart. It plots the effect of inputs **P1-WB_Thickness** and **P2-WB_Radius** on the output **P4-WB_Deformation**. An explanation of how the bar chart is built follows.

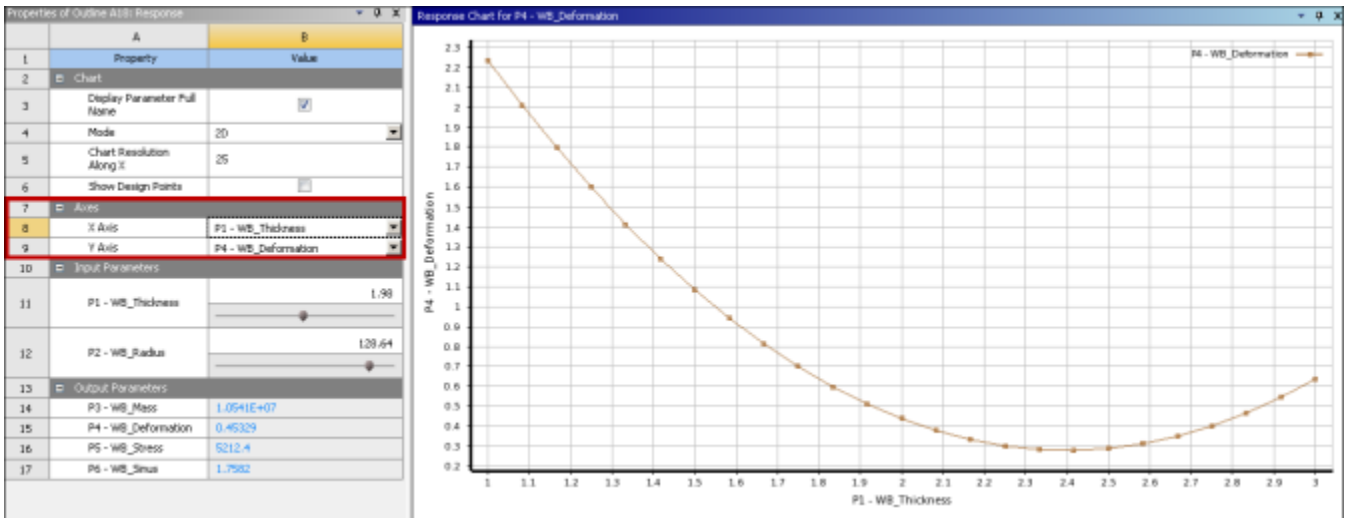


By default, **Axis Range** is set to **Use Min Max of the Output Parameter** and the maximum known variation of P4 is **2.36**. Using this chart, you want to determine what percentage of this range of variation is produced by varying only P1, or only P2.

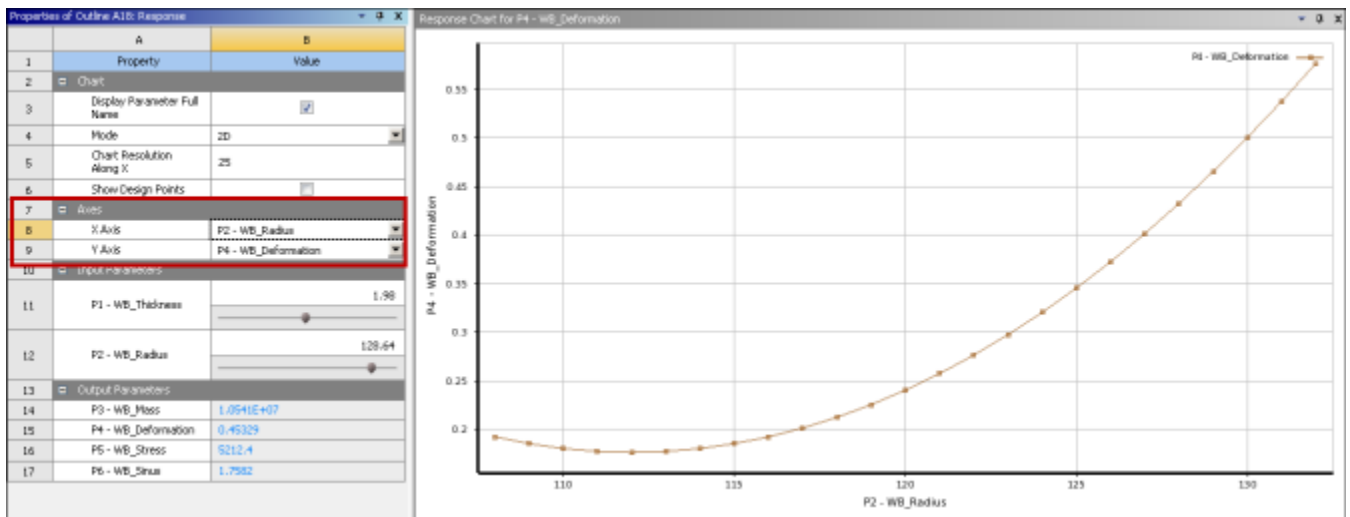
To determine the sensitivity of **P1-WB_Thickness** vs. **P4-WB_Deformation**:

1. Plot the response curve **P4 = f(P1)**.
2. Compute **Max(P4)-Min(P4) ≈ 1.96**. (This is about 83% of 2.36.)
3. If **P4** increases while **P1** increases, the sign is positive. Otherwise, the sign is negative.

For this example, the sensitivity is 83%. This corresponds to the red bar for **P1-WB_Thickness** in the Local Sensitivity chart.



Now plot **P4 = f(P2)**, **Max(P4)-Min(P4) ≈ 0.4**. This corresponds to 16.9% of 2.36, as represented by the blue bar for **P2-WB_Radius** in the Local Sensitivity chart.



On each of these curves, only one input is varying. The other input parameters are constant. However, you can change the value of any input and get an updated curve. This also applies to the standard Local Sensitivity chart. All the sensitivity values are recalculated when you change the value of an input parameter. If parameters are correlated, you'll see the sensitivity varying. The relative weights of inputs can vary, depending on the design.

If **Axes Range** is set to **Use Chart Data**, the Min-Max search results are ignored. The input parameter generating the largest variation of the output is taken as the reference to calculate the percentages for other input parameters.

In this example, P1 is the input generating the largest variation of P4 (1.96). The red bar for P1 is set to **100%**, and the blue bar for P2 is set to **20.4%**, meaning that P2 only generates 20.4% of the variation that P1 can generate.

Local Sensitivity Chart: Properties

To specify the properties of a local sensitivity chart, first select **Local** in the **Outline** pane for the response surface. Then, edit the properties in the **Properties** pane.

Chart Properties

- **Display Parameter Full Name:** Specify whether to show the full parameter name or the short parameter name.
- **Mode:** Determines whether the chart is in **Pie** or **Bar** format.
- **Axes Range:** Determines the lower and upper bounds of the Y axis. Available only for bar charts.
 - If **Use Min Max of the Output Parameter** is selected, values on the Y axis are scaled based on Min-Max search results. If the Min-Max search object was disabled, the output parameter bounds are determined from existing design points. The axis bounds are fixed to -100, 100, or 0.
 - If **Use Chart Data** is selected, values on the Y axis are scaled based on the input parameter that generates the largest variation of the output parameter. The axis bounds are adjusted to fit the displayed bars.

Input Parameters

Each of the input parameters is listed in this section. For each input parameter:

- Under **Value**, you can change the value by moving the slider or entering a new value with the keyboard. The number to the right of the slide represents the current value.
- Under **Enabled**, you can enable or disable the parameter by selecting or clearing the check box. Disabled parameters do not display on the chart.

Output Parameters

Each of the output parameters is listed in this section. For each output parameter:

- Under **Value**, you can view the interpolated value for the parameter.
- Under **Enabled**, you can enable or disable the parameter by selecting or clearing the check box. Disabled parameters do not display on the chart.

Generic Chart Properties

You can modify various generic chart properties for this chart. For more information, see [Setting Chart Properties](#).

Using the Local Sensitivity Curves Chart

The **Local Sensitivity Curves** chart helps you to focus your analysis by allowing you to view independent parameter variations within the standard Local Sensitivity chart. This multi-curve chart provides a means of viewing the effect of each input parameter on specific outputs, given the current values of the other parameters. The Local Sensitivities Curves chart shows *individual* local sensitivities, with a separate curve to represent the effect of *each* input on one or two outputs.

There are two types of Local Sensitivities Curves chart:

- Single output: Calculates the effect of each input parameter on a single output parameter of your choice.
- Dual output: Calculates the effect of each input parameter on two output parameters of your choice.

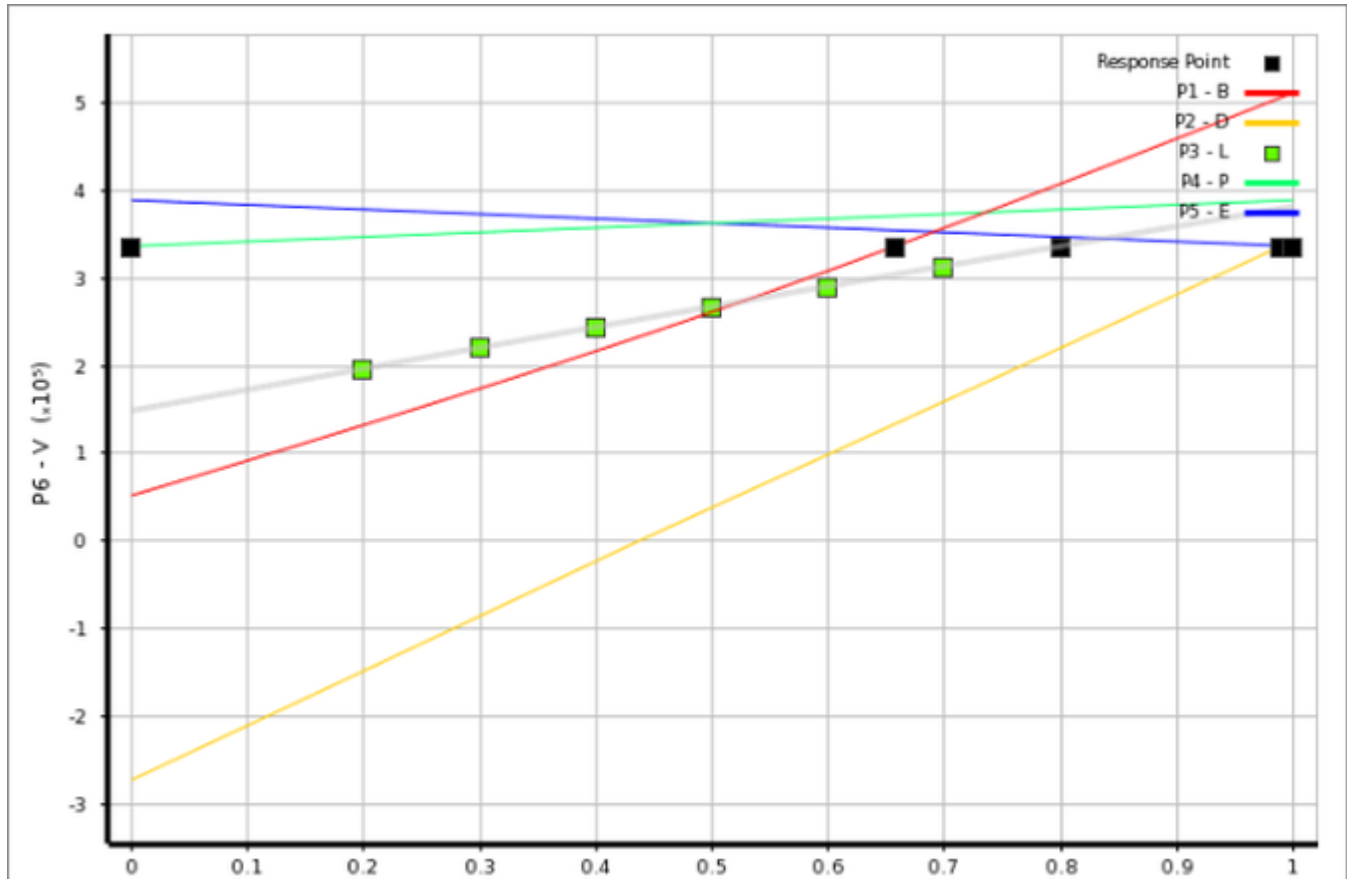
Local Sensitivity Curves Chart: Single Output

By default, the chart opens to the single output version. In this version:

- The X axis is all selected inputs (normalized).
- The Y axis is a single output parameter.
- Each curve represents the effect of an enabled input parameter on the selected output.
- For each curve, the current response point is indicated by a black point marker (all of the response points have equal Y axis values).

- Continuous parameters with manufacturable values are represented by gray curves. The colored markers are the manufacturable values that are defined.
- Where effects are the same for one or more inputs, the curves are superimposed. The curve of the input displayed first on the list hides the curves for the other inputs.

To change the output being considered, go to the **Properties** pane and select a different output parameter for the **Y axis** property.



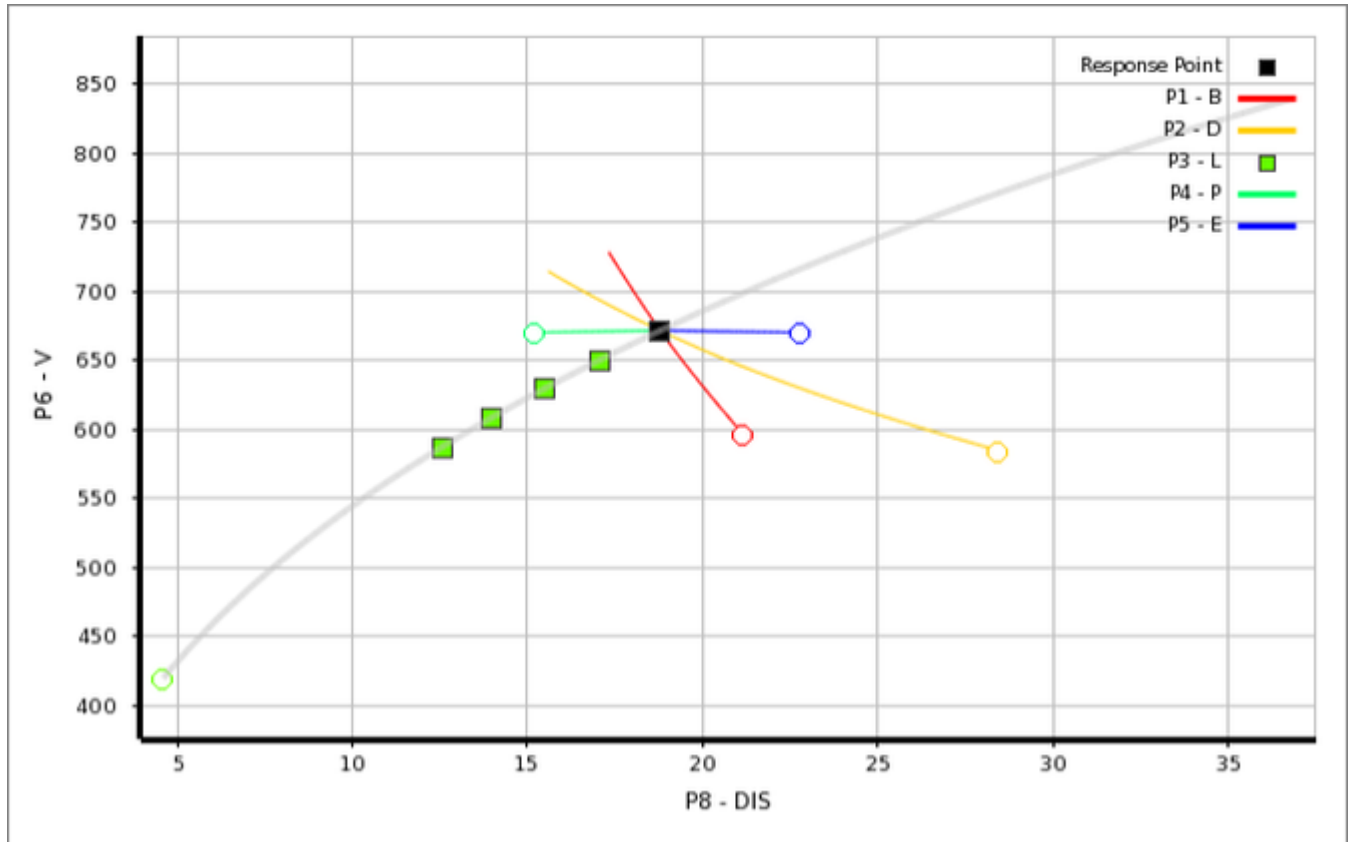
Local Sensitivity Curves Chart: Dual Output

To view the dual output version, go to the **Properties** pane and for both **X axis** and **Y axis**, select output parameters. In this version:

- The X axis is a selected output parameter.
- The Y axis displays a selected output parameter.
- Each curve represents the effect of an enabled input parameter on the two selected outputs.
- The circle at the end of a curve represents the beginning of the curve, which is the lower bound of the input parameter.
- For each curve, the current response point is indicated by a black point marker.
- Continuous parameters with manufacturable values are represented by gray curves. The colored markers are the manufacturable values that are defined.

- Where effects are the same for one or more inputs, the curves are superimposed. The curve of the input displayed first on the list hides the curves for the other inputs.

To change the outputs being considered, go to the **Properties** pane and select a different output parameter for one or both of the chart axes.



For more information, see [Local Sensitivity Curves Chart: Properties](#) (p. 169).

Local Sensitivity Curves Chart: Example

This section explains how to interpret the Local Sensitivity Curves chart and illustrates how it is related to the Local Sensitivity chart for the same response point.

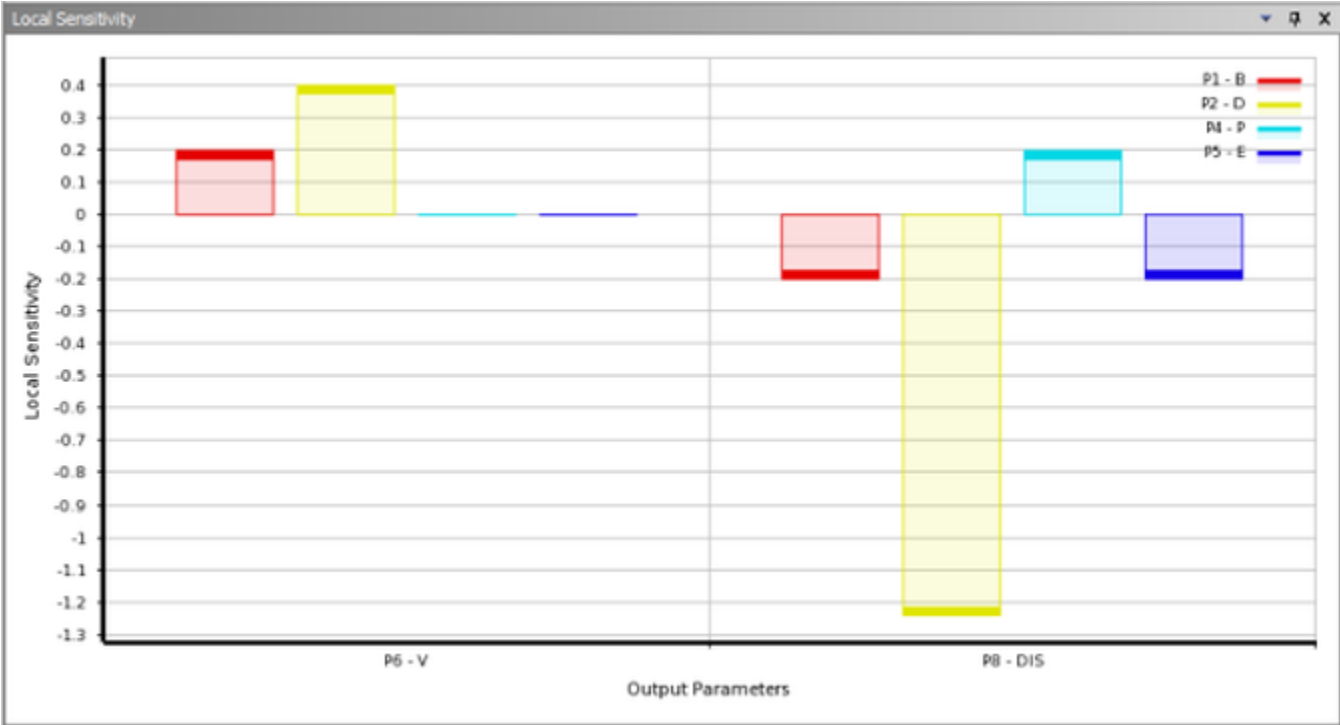
An example follows of a Local Sensitivity bar chart for a given response point. This chart shows the effect of four input parameters (**P1-B**, **P2-D**, **P4-P**, and **P5-E**) on two output parameters (**P6-V** and **P8-DIS**).

On the left side of the chart, you can see how input parameters affect the output parameter **P6-V**:

- **P2-D** (yellow bar) has the most effect and the effect is positive.
- **P1-B** (red bar) has a moderate effect, and effect is positive.
- Input parameters **P4-P** and **P5-E** (the teal and blue bars) have no effect at all.

On the right side of the chart, you can see the difference in how the same input parameters affect the output parameter **P8-DIS**:

- **P2-D** (yellow bar) has the most effect, but the effect is negative.
- **P1-B** (red bar) has a moderate effect, but the effect is negative.
- **P4-P** (teal bar) has now has a moderate effect, and the effect is positive.
- **P5-E** (blue bar) now has a moderate effect, and the effect is negative.

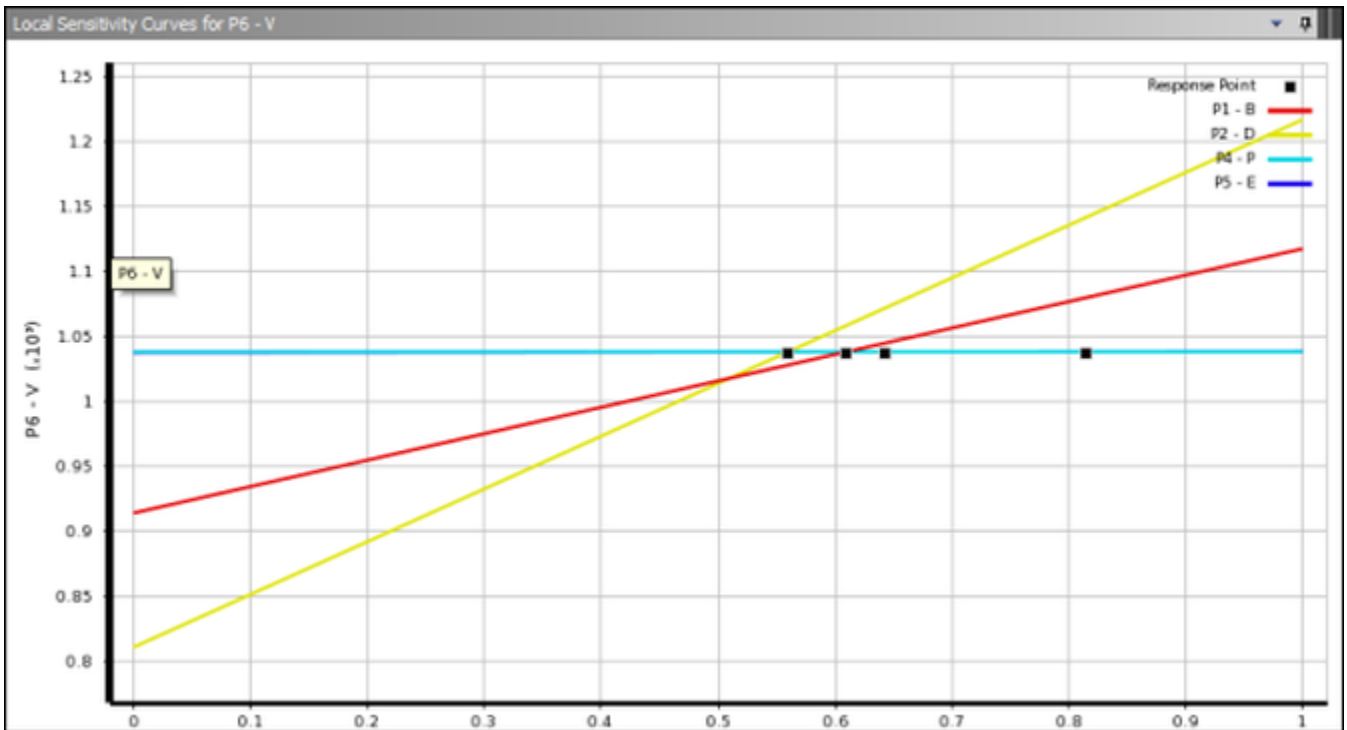


Single Output

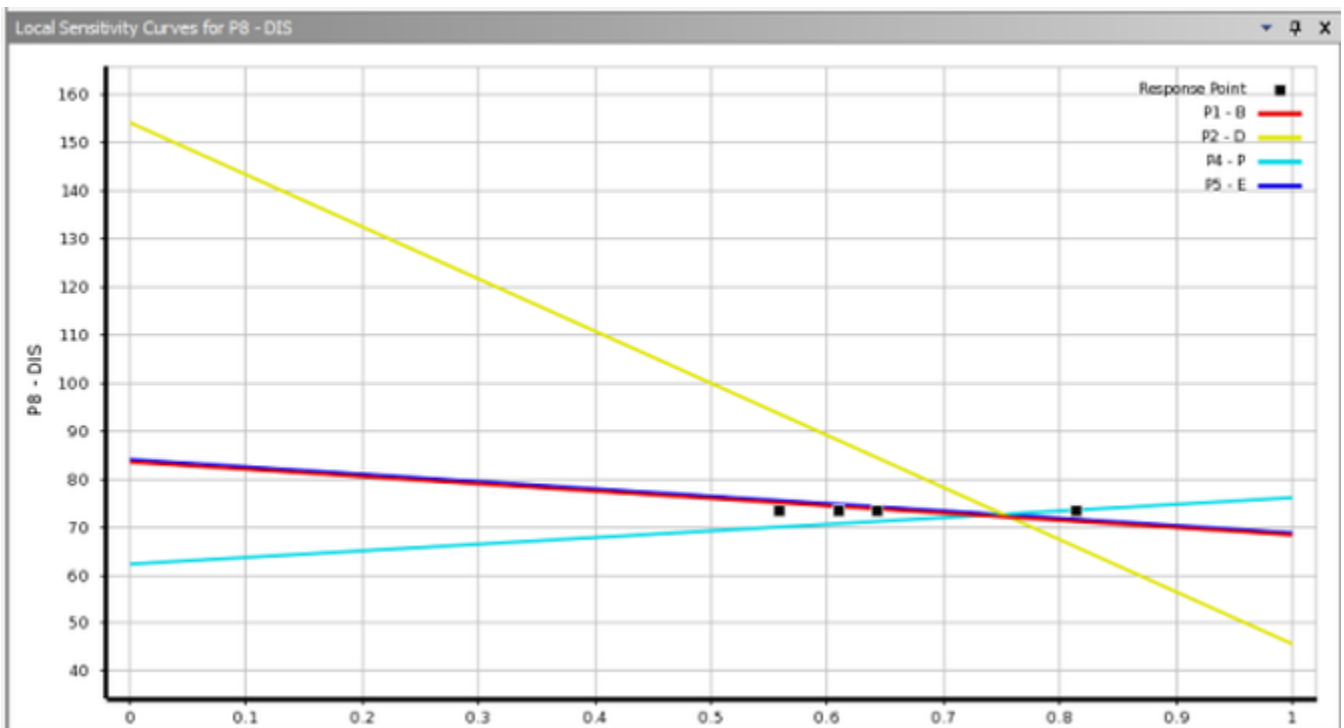
When you view the Local Sensitivity Curves chart for the same response point, the chart defaults to the Single Output version. This means that the chart shows the effect of all enabled inputs on a single selected output. The output parameter is on the Y axis, with effect measured horizontally.

In the two examples that follow, you can see that the Single Output curve charts for output parameters **P6-V** and **P8-DIS** show the same sensitivities as the Local Sensitivity bar chart.

For output **P6-V**, inputs **P4-P** and **P5-E** have the same level of effect. Consequently, the blue line is hidden behind the teal line.



For output **P8-DIS**, inputs **P1-B** and **P5-E** have the same level of effect. Consequently, the blue line is hidden behind the red line.



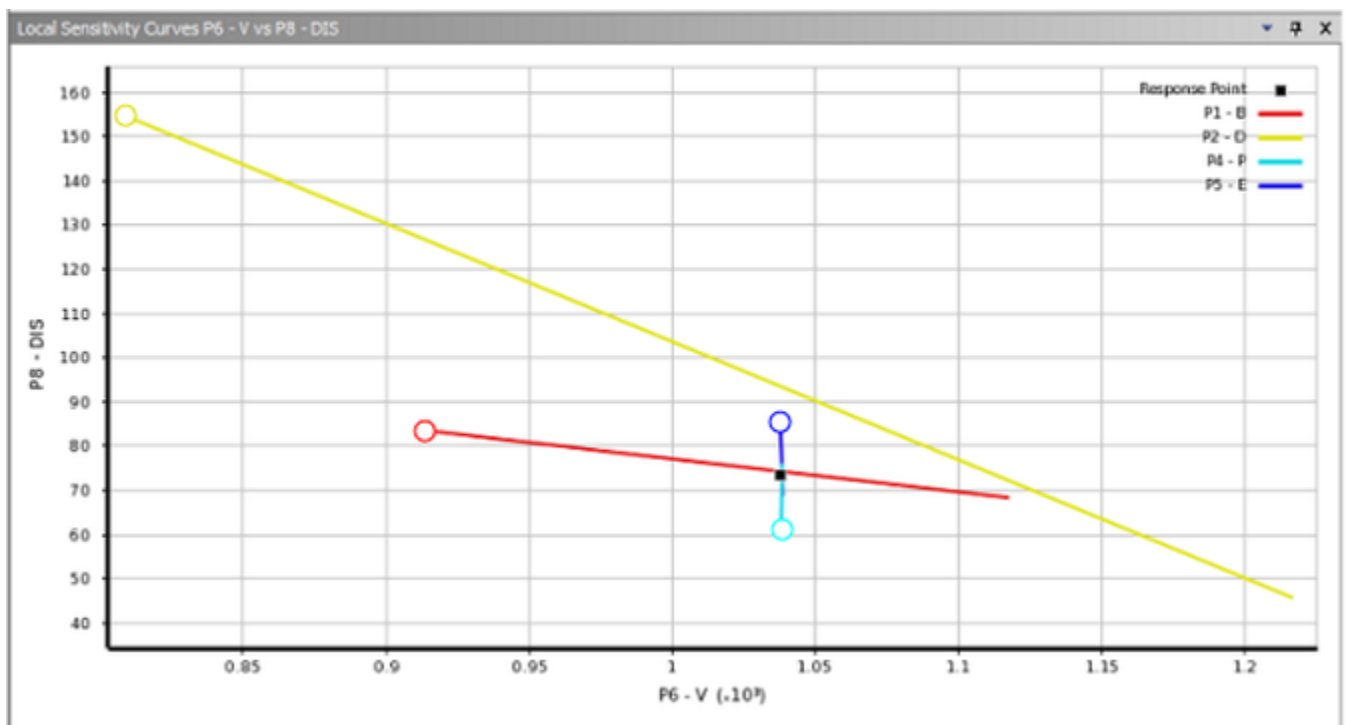
Dual Output

For more information, you can view the dual output version of the Local Sensitivity Curves chart. The dual output version shows the effect of all enabled inputs on two selected outputs. In this particular example, there are only two output parameters. If there were six outputs in the project,

however, you could narrow the focus of your analysis by selecting the two that are of most interest to you.

The following figure shows the effect of the same input parameters on both of the output parameters used previously. Output **P6-V** is on the X axis, with effect measured vertically. Output **P8-DIS** is on the Y axis, with effect measured horizontally. From this dual representation, you can see the following:

- **P2-D** has the most significant effect on both outputs. The effect is positive for output **P6-V** and is negative for output **P8-DIS**.
- **P1-B** has a moderate effect on both outputs. The effect is positive for output **P6-V** and is negative for output **P8-DIS**.
- **P4-P** has no effect on output **P6-V** and has a moderate positive effect on output **P8-DIS**.
- **P5-E** has no effect on output **P6-V** and a moderate negative effect on output **P8-DIS**. Due to duplicate effects, its curve is hidden for both outputs.



Local Sensitivity Curves Chart: Properties

To specify the properties of a local sensitivities chart, first select **Local Sensitivities Curves** in the **Outline** pane for the response surface. Then, edit the properties in the **Properties** pane.

Chart Properties

Determines the properties of the chart.

- **Display Parameter Full Name:** Specify whether to show the full parameter name or the short parameter name.

- **Axes Range:** Determines the lower and upper bounds of the output parameters axes.
 - If **Use Min Max of the Output Parameter** is selected, the values on the Y axis are scaled based on Min-Max search results. If the Min-Max search object was disabled, the output parameter bounds are determined from existing design points. The axis bounds are fixed to -100, 100, or 0.
 - If **Use Chart Data** is selected, the values on the Y axis are scaled based on the input parameter that generates the largest variation of the output parameter. The axis bounds are adjusted to fit the displayed bars.
- **Chart Resolution:** Determines the number of points per curve. The number of points controls the amount of curvature that can be displayed. A minimum of 2 points is required and produces a straight line. A maximum of 100 points is allowed for maximum curvature. The default is 25.

Axes Properties

Determines what data is displayed on each chart axis. For each axis, under **Value**, you can change what the chart displays on an axis by selecting an option from the drop-down.

- For **X-Axis**, available options are **Input Parameters** and each of the output parameters defined in the project. If **Input Parameters** is selected, you are viewing a single output chart. Otherwise, you are viewing a dual output chart.
- For **Y-Axis**, available options are each of the output parameters defined in the project.

Input Parameters

Each of the input parameters is listed in this section. For each input parameter:

- Under **Value**, you can change the value by moving the slider or entering a new value with the keyboard. The number to the right of the slider represents the current value.
- Under **Enabled**, you can enable or disable the parameter by selecting or clearing the check box. Disabled parameters do not display on the chart.

Note:

Discrete input parameters display with their levels values associated to a response point, but they cannot be enabled as chart variables.

Output Parameters

Each of the output parameters is listed in this section. For each output parameter, under **Value**, you can view the interpolated value.

Generic Chart Properties

You can modify various generic chart properties for this chart. For more information, see [Setting Chart Properties](#).

Exporting Response Surfaces

Once a response surface is solved, you can export it as a lightweight, independent reduced-order model (DX-ROM). You can reuse exported response surfaces in other Workbench projects (using the Ansys DX-ROM reader), the Excel DX-ROM add-in, or software that implements the Functional Mock-up Interface (FMI). For example, Ansys Twin Builder implements FMI and Mathworks MATLAB®. For more information, see the Twin Builder help and MathWorks [FMI Toolbox](#) documentation.

An update of a DX-ROM consumes fewer resources and is faster than updating the actual solvers. While a solver-based evaluation can consume gigabytes of memory and take hours (or even days), a DX-ROM evaluation consumes only kilobytes and is nearly instantaneous. To take advantage of this, you might want to import a response surface back into Workbench and use it as a lightweight replacement for one or more systems in a simulation project.

Response surface export is supported for all DesignXplorer response surface types except Neural Network and Sparse Grid. Because DesignXplorer does not build a response surface for derived parameters, derived parameters are not exported. All other parameter types are supported.

DesignXplorer can export a ROM response surface in the following file formats:

Functional Mock-up Unit (version 1.0): * .**fmu** and Functional Mock-up Unit (version 2.0): * .**fmu**

The **.fmu** format has the broadest applicability. It can be used by Twin Builder, MATLAB, or any software with functional mock-up interface (FMI) support. It is the recommended format for export to external software.

Note:

Using an FMU file exported from DesignXplorer implies the approval of the terms of use supplied in the **License.txt** file. To access **License.txt**, use a zip utility to manually extract all files from the **.fmu** package.

DesignXplorer Native Database: * .**dxrom**

The format can be imported into Workbench and DesignXplorer using custom DX-ROM tools. Ansys provides the **Response Surface Reader** app, which contains three custom tools for working with any response surface exported as a DesignXplorerNative Database file (DXROM): Response Surface Reader System, Excel DX-ROM Add-in, and DX-ROM Postprocessing Utility. For more information, see [Tools for Importing a DX-ROM Response Surface into Workbench](#) (p. 172).

Once exported, the DX-ROM response surface file is available to be imported into the software or reader of your choice. Disabled parameters and derived parameters are not included in the export.

Exporting a Response Surface as a DX-ROM File

To export your solved response surface as a DX-ROM file:

1. Access the **Export Response Surface** dialog box use one of the following methods:
 - In the **Project Schematic**, right-click the **Response Surface** cell and select **Export Response Surface**.

- Double-click the **Response Surface** cell to open it:
 - On the toolbar, click the **Export Response Surface** button.
 - In the **Outline** pane, right-click **Response Surface** and select **Export Response Surface**.
 - In the **Outline** pane, under **Response Surface** select the Response chart for the desired point. Then, in the **Chart** pane, right-click and select **Export Response Surface**.
- 2. In the **Export Response Surface** dialog box, specify a file type, location, and name. Then, click **Save**.

You can import your DX-ROM response surface file into the software or FMI-supported reader of your choice. For more information, see [Tools for Importing a DX-ROM Response Surface into Workbench](#) (p. 172).

Tools for Importing a DX-ROM Response Surface into Workbench

If you've exported your DX-ROM response surface as a DXROM file, you can use one of DesignXplorer's custom DX-ROM tools to incorporate it into your simulation.

Each of these tools is delivered in the **Response Surface Reader** app, which is available for download from the [Ansys Store](#).

Ansys provides the following custom DX-ROM tools for reading the DesignXplorer Native Database (DXROM) file format:

Response Surface Reader System

ACT extension that exposes a new system under **Design Exploration** in the Workbench **Toolbox**.

Excel DX-ROM Add-in

Exposed as an **DESIGNXPLORER** tab in the Microsoft Excel application.

DX-ROM Postprocessing Utility

Lightweight command line tool that can be used to evaluate your exported DX-ROM file, either in Workbench or in your own external simulation environment. You can use the utility independently (as delivered, with no need for additional development) or in the development of your own custom response surface tools.

Downloading the Response Surface Readers App

To download the **Response Surface Reader** app:

1. Go to the [Ansys Store](#).
2. For **Target Application**, select **DesignXplorer**.

While you can optionally select a product version, on the page for the app, you can select the version that you want to download. The default is the latest version.

3. Locate and click the **Response Surface Reader** app.
4. Download the desired version, which requires agreeing to the software license agreement.
5. Decompress the downloaded ZIP file for the app to a directory of your choice.

The DX-ROM tools included in the app are now available to install, load, and use. For more information, see the help document **DX_ResponseSurfaceReaders.pdf**, which is included in the app's **doc** directory.

Using Goal-Driven Optimizations

DesignXplorer offers two different types of goal-driven optimization systems: **Response Surface Optimization** and **Direct Optimization**.

- A **Response Surface Optimization** system draws its information from its own **Response Surface** cell and so is dependent on the quality of the response surface. The available optimization methods are Screening, MOGA, NLPQL, and MISQP, which all use response surface evaluations rather than real solves.
- A **Direct Optimization** system has only one cell, which utilizes real solves rather than response surface evaluations. The available optimization methods are Screening, NLPQL, MISQP, Adaptive Single-Objective, and Adaptive Multiple-Objective.

Tip:

On the DesignXplorer page In the Ansys Help, *DesignXplorer Optimization Tutorials* provides several optimization examples that you can step through to learn how to use DesignXplorer to analyze and optimize design spaces.

To make performing optimizations easy for non-experts, DesignXplorer automatically selects an optimization method by default. Based on the number and types of input parameters, the number of objectives and constraints, any defined parameter relationships, and the run time index (for a **Direct Optimization** system only), DesignXplorer selects the most appropriate method and sets its properties.

Any time that you make a change to any input used for automatic method selection, DesignXplorer once again assesses the scenario to select the most appropriate method. For method availabilities and capabilities, see [Goal-Driven Optimization Methods \(p. 180\)](#).

Note:

In **Tools** → **Options** → **Design Exploration** → **Sampling and Optimization**, you can change **Method Selection** from **Auto** to **Manual** if you want this to be the default for newly inserted optimization systems. As indicated in the next topic, you can always change the method selection for a particular optimization system in the properties of the **Optimization** cell.

Although a **Direct Optimization** system does not have a **Response Surface** cell, it can draw information from any other system or component that contains design point data. It is possible to reuse existing design point data, reducing the time needed for the optimization, without altering the source of the design points. For example:

- You can transfer design point data from an existing **Response Surface Optimization** optimization and improve upon it without actually changing the original response surface.

- You can use information from a **Response Surface** system that has been refined with the Kriging method and validated. You can transfer the design point data to a **Direct Optimization** system and then adjust the quality of the original response surface without affecting the attached direct optimization.
- You can transfer information from any DesignXplorer system or component containing design point data that has already been updated, saving time and resources by reusing existing, up-to-date data rather than reprocessing it.

Note:

The transfer of design point data between two **Direct Optimization** systems is not supported.

You can monitor optimization progress of a **Direct Optimization** system from the **Table** pane. During the direct optimization, the **Table** pane displays all the design points as they are calculated, allowing you to see how the optimization proceeds, how it converges, and so on. Once the optimization is complete, the raw design point data is stored for future reference. You can access the data by selecting **Raw Optimization Data** in the **Outline** pane. The **Table** pane displays the design points that were calculated during the optimization.

Note:

This list is compiled of raw data and does not show feasibility, ratings, and so on for the included design points.

The following sections provide comprehensive information on performing goal-driven optimizations:

- [Creating a Goal-Driven Optimization System](#)
- [Transferring Design Point Data for Direct Optimization](#)
- [Goal-Driven Optimization Methods](#)
- [Defining the Optimization Domain](#)
- [Defining Optimization Objectives and Constraints](#)
- [Working with Candidate Points](#)
- [Goal-Driven Optimization Charts and Results](#)

Creating a Goal-Driven Optimization System

To create a goal-driven optimization system in the **Project Schematic**:

1. From under **Design Exploration** in the Workbench **Toolbox**, drag either the **Response Surface Optimization** system or **Direct Optimization** system and drop it in the **Project Schematic**. You can drag it:
 - Directly under either the **Parameter Set** bar or an existing system under the **Parameter Set** bar, in which case it does not share any data with any other systems in the **Project Schematic**.
 - On the **Design of Experiments** cell of a system containing a response surface, in which case it does share all data generated by the **Design of Experiments** cell.

- On the **Response Surface** cell of a system containing a response surface, in which case it does share all data generated for the **Design of Experiments** and **Response Surface** cells.

For more information on data transfer, see [Transferring Design Point Data for Direct Optimization](#) (p. 179).

2. For a **Response Surface Optimization** system, if you are not sharing the **Design of Experiments** and **Response Surface** cells, edit the DOE, setting it up as described in [Design of Experiments Component Reference](#) (p. 44). Then, solve both the **Design of Experiments** and **Response Surface** cells.
3. For a **Direct Optimization** system, if you have not already shared data via the options in step 1, you can create data transfer links to provide the system with design point data.

Note:

If no design point data is shared, design point data is generated automatically by the **Update** operation.

4. On the **Project Schematic**, double-click the **Optimization** cell of the new system to open the component tab.
5. Indicate whether DesignXplorer is to automatically select the most appropriate optimization method or if you are to manually select the method.

Automatic Selection

To have DesignXplorer select the most appropriate method:

1. In the **Outline** pane, select **Optimization**.
2. In the **Properties** pane, for **Method Selection**, select **Auto**.
 - For a **Response Surface Optimization** system, **Maximum Number of Candidates** is the only other property available.
 - For a **Direct Optimization** system, **Maximum Number of Candidates** and **Run Time Index** are available. For **Run Time Index**, the default value is **5 – Medium**. However, you can change this value if you want. This value and the number and types of input parameters, number of objectives and constraints, and any defined parameter relationships determine the method selected and its properties.

Manual Selection

To manually select the method yourself:

1. In the **Outline** pane, select **Optimization**.
2. In the **Properties** pane, for **Method Selection**, select **Manual**. If after automatic method selection you switch to manual selection, the method and properties initially proposed by automatic selection are kept so that you can simply modify only the needed properties.

3. Select the method and specify properties. For more information, see [Goal-Driven Optimization Methods](#) (p. 180).

Estimated Number of Evaluations or Design Points

Regardless of what you select for **Method Selection** or **Method Name**, a read-only property is visible: **Estimated Number of Evaluations** (for a **Response Surface Optimization** system) or **Estimated Number of Design Points** (for a **Direct Optimization** system). For manual method selection, this read-only property appears below **Method Name** because the number depends on the method. For automatic method selection, this read-only property appears above **Method Name** because the number is more important to a non-expert user than the method. The value for **Estimated Number of Design Points** depends on the selection for **Run Time Index** firstly (for a **Direct Optimization** system) and **Method Name** secondly.

This read-only property displays an estimate of the number of either evaluations or design points that will be calculated during the optimization process. However, this process can be stopped before this estimate is reached. For a **Direct Optimization** system, you can change the selection for **Run Time Index** to increase or decrease the number shown in **Estimated Number of Design Points**.

Objectives and Constraints

After setting up the optimization method, you specify objectives and constraints and then start the update:

1. In the **Outline** pane, select either **Objectives and Constraints** or an object under it.
2. In the **Table** or **Properties** pane, define the optimization objectives and constraints. For more information, see [Defining Optimization Objectives and Constraints](#) (p. 205).
3. Specify the optimization domain:
 - In the **Outline** pane, select **Domain** or an input parameter or parameter relationship under it.
 - In the **Table** or **Properties** pane, define the selected domain object. For more information, see [Defining the Optimization Domain](#) (p. 201).
4. Click **Update** on the toolbar.

Automatic Selection Scenarios

When DesignXplorer is selecting the most method, three scenarios can occur:

- A stopping criterion, such as the maximum number of iterations or maximum number of points, is reached. To continue the convergence, you can set **Method Selection** to **Manual** and modify the stopping criterion. Or, you can adjust the problem, perhaps by changing the starting point or domain definition.
- The optimization converges before reaching the maximum number of points or iterations. In this case, in the **Property** pane under **Optimization Status**, DesignXplorer sets the read-only property **Converged** to **Yes**.

- The optimization is unable to converge before reaching the maximum number of points or iterations. In this case, in the **Property** pane under **Optimization Status**, DesignXplorer sets the read-only property **Converged** to **No**.

Transferring Design Point Data for Direct Optimization

In DesignXplorer, you can transfer design point data to a **Direct Optimization** system by creating one or more data transfer links from a component containing design point data to the system's **Optimization** cell.

Note:

The transfer of design point data between two **Direct Optimization** systems is not supported.

Data Transferred

The data transferred consists of all design points that have been obtained by a real solution. Design points obtained from the evaluation of a response surface are not transferred. Design point data is transferred according to the nature of its source component.

- **Design of Experiments** cell: All points, including those with custom output values.
- **Response Surface** cell: All refinement and verification points used to create or evaluate from the response surface, because they are obtained by a real solution.

Note:

The points from the DOE are not transferred. To transfer these points, you must create a data transfer link from the **Design of Experiments** cell.

- **Parameters Correlation** cell (stand-alone): All points.
- **Parameters Correlation** cell (linked to a **Response Surface** cell): No points because all of them are obtained from a response surface evaluation rather than from a real solve.

Data Usage

Once transferred, the design point data is stored in an initial sample set. In the **Properties** pane for the **Optimization** cell, **Method Name** is available under **Optimization**.

- If you set **Method Name** to **NLPQL** or **MISQP**, the samples are not used.
- If you set **Method Name** to **Adaptive Single-Optimization**, the initial sample set is filled with transferred points and additional points are generated to reach the requested number of samples and the workflow of LHS.
- If you set **Method Name** to **MOGA** or **Adaptive Multiple-Objective**, the initial sample set is filled with transferred points and additional points are generated to ready the requested number of samples.

The transferred points are added to the samples generated to initiate the optimization. For example, if you have requested 100 samples and 15 points are transferred, a total of 115 samples are available to initiate the optimization.

When there are duplicates between the transferred points and the initial sample set, the duplicates are removed. For example, if you have requested 100 samples, 15 points are transferred, and 6 duplicates are found, a total of 109 samples are available to initiate the optimization.

For more information on data transfer links, see [Links](#) in the *Workbench User's Guide*.

Goal-Driven Optimization Methods

DesignXplorer offers the following goal-driven optimization methods:

Method	Description	Response Surface Optimization	Direct Optimization
Screening	Shifted-Hammersley Sampling	X	X
NLPQL	Nonlinear Programming by Quadratic Lagrangian	X	X
MISQP	Mixed-Integer Sequential Quadratic Programming	X	X
MOGA	Multi-Objective Genetic Algorithm	X	X
Adaptive Single-Objective	Hybrid optimization method using Optimal Space-Filling Design, a Kriging response surface, MISQP, and domain reduction in a Direct Optimization system		X
Adaptive Multiple-Objective	Hybrid optimization method using a Kriging response surface and in a Direct Optimization system		X
External Optimizer	External optimizer as defined in a loaded optimization extension	Availability is determined by the optimizer, as defined in the optimization extension.	

The following table indicates the general capabilities of each goal-driven optimization method.

Method	Single Objective	Multiple Objectives	Local Search	Global Search	Discrete	Manufacturable Values	Parameter Relationships
Screening		X		X	X	X	X
NLPQL	X		X				X
MISQP	X		X		X	X	X
MOGA		X		X	X	X	X

Adaptive Single-Objective	X			X		X	
Adaptive Multiple-Objective		X		X		X	X
External Optimizer	Capabilities are determined by the optimizer, as defined in the optimization extension.						

Method Name, which appears in the **Properties** pane for the **Optimization** cell, specifies the optimization method for the design study. DesignXplorer filters **Method Name** choices for applicability to the current project, displaying only the methods that can be used to solve the optimization problem as it is currently defined. For example, if your project has multiple objectives defined and an external optimizer does not support multiple objectives, **External Optimizer** is excluded. When no objectives or constraints are defined for a project, all optimization methods are available for **Method Name**. If you already know that you want to use a particular external optimizer, you should select **External Optimizer** as the method before setting up the rest of the project. Otherwise, it could be inadvertently filtered from the list.

For instructions for setting the optimization properties for each algorithm, see:

- [Performing a Screening Optimization](#)
- [Performing a MOGA Optimization](#)
- [Performing an NLPQL Optimization](#)
- [Performing an MISQP Optimization](#)
- [Performing an Adaptive Single-Objective Optimization](#)
- [Performing an Adaptive Multiple-Objective Optimization](#)
- [Performing an Optimization with an External Optimizer](#)

Performing a Screening Optimization

The Screening method can be used for both **Response Surface Optimization** and **Direct Optimization** systems. This method allows you to generate a new sample set and sort the samples based on objectives and constraints. This non-iterative method is available for all types of input parameters. Screening is typically used to find a first set of candidate points for a preliminary design. Then, if you want to refine, these candidate points are used as starting points for gradient methods. For more information, see [GDO Principles \(p. 351\)](#).

To perform a Screening optimization:

1. In the **Outline** pane for the **Optimization** cell, select **Optimization**.
2. In the **Properties** pane, enter the following input properties:
 - **Method Selection:** Set to **Manual**.
 - **Method Name:** Set to **Screening**.
 - **Verify Candidate Points:** Select to verify candidate points automatically at end of an update for a **Response Surface Optimization** system. This property is not applicable to a **Direct Optimization** system.

- **Number of Samples:** Enter the number of samples to generate for the optimization. Samples are generated very rapidly from the response surface and do not require an actual solve of design points.

The number of samples must be greater than the number of enabled input parameters. The number of enabled input parameters is also the minimum number of samples required to generate the Sensitivities chart. You can enter a minimum of **2** and a maximum of **10,000**. The default is **1000** for a **Response Surface Optimization** system and **100** for a **Direct Optimization** system.

- **Maximum Number of Candidates:** Maximum number of candidates that the algorithm is to generate. The default is **3**. For more information, see [Viewing and Editing Candidate Points in the Table Pane](#) (p. 212).

3. Specify optimization objective and constraints:

- In the **Outline** pane, select **Objectives and Constraints** or an object under it.
- In the **Table** or **Properties** pane, define the optimization objectives and constraints. For more information, see [Defining Optimization Objectives and Constraints](#) (p. 205).

4. Specify the optimization domain:

- In the **Outline** pane, select **Domain** or an input parameter or parameter relationship under it.
- In the **Table** or **Properties** pane, define the selected domain object. For more information, see [Defining the Optimization Domain](#) (p. 201).

5. Update the **Optimization** cell.

The result is a group of points or sample set. The **Table** pane displays the points that are most in alignment with the objectives and constraints as the *candidate points* for the optimization. The **Properties** pane displays **Size of Generated Sample Set**, which is read-only.

When the Screening method is used for a **Response Surface Optimization** system, the sample points obtained from a response surface Min-Max search are automatically added to the sample set used to initialize or run the optimization.

For example, if the Min-Max search results in 4 points and you run the Screening optimization with **Number of Samples** set to **100**, the final optimization sample set contains up to 104 points.

If a point found by the Min-Max search is also contained in the initial screening sample set, it is only counted once in the final sample set.

Note:

When constraints exist, the Tradeoff chart indicates which samples are feasible (meet the constraints) or infeasible (do not meet the constraints). There is a display option for the infeasible points.

6. In the **Outline** pane, select **Optimization**. Then, in the **Properties** pane under **Optimization Status**, review the outcome:

- **Converged:** Indicates whether the optimization has converged.

Number of Evaluations: Number of design point evaluations to perform. This value takes all points used in the optimization, including design points pulled from the cache. It can be used to measure the efficiency of the optimization method to find the optimum design point.

- **Number of Failures:** Number of failed design points for the optimization. When design points fail, a Screening optimization does not attempt to solve additional design points in their place. The Samples chart for a **Direct Optimization** system does not include failed design points.
- **Size of Generated Sample Set:** Number of samples generated in the sample set. For a **Direct Optimization** system, this is the number of samples successfully updated. For a **Response Surface Optimization** system, this is the number of samples successfully updated plus the number of different (non-duplicate) samples generated by the Min-Max search if it is enabled.

Number of Candidates: Number of candidates obtained. This value is limited by the **Maximum Number of Candidates** input property.

7. In the **Outline** pane, select **Domain** or any object under it to view domain data in the **Properties** and **Table** panes.
8. For a **Direct Optimization** system, select **Raw Optimization Data** in the **Outline** pane. The **Table** pane displays the design points that were calculated during the optimization. If the raw optimization data point exists in the design points table for the **Parameter Set** bar, the corresponding design point name is indicated in parentheses in the **Name** column.

Note:

This list is compiled of raw data and does not show feasibility, ratings, and so on for the included design points.

Performing a MOGA Optimization

The MOGA (Multi-Objective Genetic Algorithm) method can be used for both **Response Surface Optimization** and **Direct Optimization** systems. It allows you to generate a new sample set or use an existing set for providing a more refined approach than the Screening method. MOGA is available for all types of input parameters and can handle multiple goals. For more information, see [Multi-Objective Genetic Algorithm \(MOGA\) \(p. 374\)](#).

Note:

MOGA requires advanced options. Ensure that the **Show Advanced Options** check box is selected on the **Design Exploration** tab of the **Options** window. Advanced options display in italic type.

To perform a MOGA optimization:

1. In the **Outline** pane for the **Optimization** cell, select **Optimization**.
2. In the **Properties** pane, enter the following input properties:

- **Method Selection:** Set to **Manual**.
- **Method Name:** Set to **MOGA**.
- **Verify Candidate Points:** Select to verify candidate points automatically at the end of the optimization update.
- **Type of Initial Sampling:** Advanced option for generating different kinds of sampling. If you do not have any parameter relationships defined, set to **Screening** (default) or **Optimal Space-Filling**. If you have parameter relationships defined, the initial sampling must be performed by the constrained sampling algorithms because parameter relationships constrain the sampling. For such cases, this property is automatically set to **Constrained Sampling**.
- **Random Generator Seed:** Advanced option that displays only when **Type of Initial Sampling** is set to **Optimal Space-Filling**. Value for initializing the random number generator invoked internally by the Optimal Space-Filling (OSF) algorithm. The value must be a positive integer. You can generate different samplings by changing the value or regenerate the same sampling by keeping the same value. The default is **0**.
- **Maximum Number of Cycles:** Advanced option that displays only when **Type of Initial Sampling** is set to **Optimal Space-Filling**. Number of optimization loops the algorithm needs, which in turns determines the discrepancy of the OSF. The optimization is essentially combinatorial, so a large number of cycles slow down the process. However, this makes the discrepancy of the OSF smaller. The value must be greater than **0**. For practical purposes, 10 cycles is usually good for up to 20 variables. The default is **10**.
- **Number of Initial Samples:** Initial number of samples to use. This number must be greater than the number of enabled input parameters. The minimum recommended number of initial samples is 10 times the number of enabled input parameters. The larger the initial sample set, the better your chances of finding the input parameter space that contains the best solutions.

The number of enabled input parameters is also the minimum number of samples required to generate the Sensitivities chart. You can enter a minimum of **2** and a maximum of **10000**. The default is **100**.

If you switch from the Screening method to the MOGA method, MOGA generates a new sample set. For the sake of consistency, enter the same number of initial samples as you used for the Screening method.

- **Number of Samples Per Iteration:** Number of samples to iterate and update with each iteration. This number must be greater than the number of enabled input parameters but less than or equal to the number of initial samples. The default is **100** for a **Response Surface Optimization** system and **50** for a **Direct Optimization** system.

You can enter a minimum of **2** and a maximum of **10000**.

- **Maximum Allowable Pareto Percentage:** Convergence criterion. Percentage value that represents the ratio of the number of desired Pareto points to the number of samples per iteration. When this percentage is reached, the optimization is converged. For example, a value of **70** with **Number of Samples Per Iteration** set to **200** would mean that the optimization should stop once the resulting front of the MOGA optimization contains at least 140 points. Of course, the optimization stops before that if the maximum number of iterations is reached.

If the **Maximum Allowable Pareto Percentage** is too low (below 30), the process can converge prematurely. If the value is too high (above 80), the process can converge slowly. The value of this property depends on the number of parameters and the nature of the design space itself. The default is **70**. Using a value between **55** and **75** works best for most problems. For more information, see [Convergence Criteria in MOGA-Based Multi-Objective Optimization \(p. 373\)](#).

- **Convergence Stability Percentage:** Convergence criterion. Percentage value that represents the stability of the population based on its mean and standard deviation. This criterion allows you to minimize the number of iterations performed while still reaching the desired level of stability. When the specified percentage is reached, the optimization is converged. The default percentage is **2**. To not take the convergence stability into account, set to **0**. For more information, see [Convergence Criteria in MOGA-Based Multi-Objective Optimization \(p. 373\)](#).
- **Maximum Number of Iterations:** Stop criterion. Maximum number of iterations that the algorithm is to execute. If this number is reached without the optimization having reached convergence, iterations stop. This also provides an idea of the maximum possible number of function evaluations that are needed for the full cycle, as well as the maximum possible time it can take to run the optimization. For example, the absolute maximum number of evaluations is given by:

$$\text{Number of Initial Samples} + \text{Number of Samples Per Iteration} * (\text{Maximum Number of Iterations} - 1)$$

- **Mutation Probability:** Advanced option for specifying the probability of applying a mutation on a design configuration. The value must be between **0** and **1**. A larger value indicates a more random algorithm. If the value is 1, the algorithm becomes a pure random search. A low probability of mutation (<0.2) is recommended. The default is **0.01**. For more information on mutation, see [MOGA Steps to Generate a New Population \(p. 376\)](#).
- **Crossover Probability:** Advanced option for specifying the probability with which parent solutions are recombined to generate offspring solutions. The value must be between **0** and **1**. A smaller value indicates a more stable population and a faster (but less accurate) solution. If the value is **0**, the parents are copied directly to the new population. A high probability of crossover (>0.9) is recommended. The default is **0.98**.
- **Maximum Number of Candidates:** Maximum number of candidates that the algorithm is to generate. The default is **3**. For more information, see [Viewing and Editing Candidate Points in the Table Pane \(p. 212\)](#).
- **Type of Discrete Crossover:** Advanced option for specifying the type of crossover for discrete parameters. This property is visible only if there is at least one discrete input variable or continuous input variable with manufacturable values. Three crossover types are available: **One Point**, **Two Points**, and **Uniform**. According to the type of crossover selected, the children are closer to or farther from their parents. The children are closer for **One Point** and farther for **Uniform**. The default is **One Point**. For more information on crossover, see [MOGA Steps to Generate a New Population \(p. 376\)](#).
- **Maximum Number of Permutations:** Read-only property showing the number of combinations possible for parameter values. This property is visible only if all input parameters are discrete or continuous with manufacturable values. The value corresponds to the product of the number of levels per input parameter. For example, if the optimization problem contains 2 discrete parameters with 4 and 5 levels respectively and 1 continuous parameter with 6 manufacturable values, **Maximum Number of Permutations** is equal to 120 (4*5*6).

3. Specify optimization objectives and constraints:

- In the **Outline** pane, select **Objectives and Constraints** or an object under it.
- In the **Table** or **Properties** pane, define the optimization objectives. For more information, see [Defining Optimization Objectives and Constraints \(p. 205\)](#).

Note:

For MOGA, at least one output parameter must have an objective defined. Multiple objectives are allowed.

4. Specify the optimization domain:

- In the **Outline** pane under **Domain**, enable the desired input parameters.
- In the **Table** or **Properties** pane, define the selected domain object. For more information, see [Defining the Optimization Domain \(p. 201\)](#).

5. Update the **Optimization** cell.

The result is a group of points or sample set. The **Table** pane displays the points that are most in alignment with the objectives as the *candidate points* for the optimization. The **Properties** pane displays **Size of Generated Sample Set**, which is read-only.

6. In the **Outline** pane, select **Optimization**. Then, in the **Properties** pane under **Optimization Status**, view the outcome:

- **Converged:** Indicates whether the optimization has converged.
- **Obtained Pareto Percentage:** Percentage representing the ratio of the number of Pareto points obtained by the optimization.
- **Number of Iterations:** Number of iterations executed. Each iteration corresponds to the generation of a population.
- **Number of Evaluations:** Number of design point evaluations performed. This value takes all points used in the optimization, including design points pulled from the cache. It can be used to measure the efficiency of the optimization method to find the optimum design point.
- **Number of Failures:** Number of failed design points for the optimization. When a design point fails, a MOGA optimization does not retain this point in the Pareto front and does not attempt to solve another design point in its place. Failed design points are also not included on the Direct Optimization Samples chart.
- **Size of Generated Sample Set:** Number of samples generated in the sample set. This is the number of samples successfully updated for the last population generated by the algorithm. It usually equals the **Number of Samples Per Iteration**.
- **Number of Candidates:** Number of candidates obtained. This value is limited by the **Maximum Number of Candidates** input property.

7. In the **Outline** pane, select **Domain** or any object under it to review domain data in the **Properties** and **Table** panes.
8. For a **Direct Optimization** system, select **Raw Optimization Data** in the **Outline** pane. The **Table** pane displays the design points that were calculated during the optimization. If the raw optimization data point exists in the design points table for the **Parameter Set** bar, the corresponding design point name is indicated in parentheses in the **Name** column.

Note:

This list is compiled of raw data and does not show feasibility, ratings, and so on for the included design points.

Performing an NLPQL Optimization

The NLPQL (Nonlinear Programming by Quadratic Lagrangian) method can be used for both **Response Surface Optimization** and **Direct Optimization** systems. It allows you to generate a new sample set to provide a more refined approach than the Screening method. Available for continuous input parameters only, NLPQL can handle only one output parameter goal. Other output parameters can be defined as constraints. For more information, see [Convergence Rate % and Initial Finite Difference Delta % in NLPQL and MISQP \(p. 353\)](#) and [Mixed-Integer Sequential Quadratic Programming \(MISQP\) \(p. 367\)](#) in the theory section.

To generate samples and perform an NLPQL optimization:

1. In the **Outline** pane for the **Optimization** cell, select **Optimization**.
2. In the **Properties** pane, enter the following input properties:
 - **Method Selection:** Set to **Manual**.
 - **Method Name:** Set to **NLPQL**.
 - **Verify Candidate Points:** Select to verify candidate points automatically at end of the optimization update.
 - **Finite Difference Approximation:** When analytical gradients are not available, NLPQL approximates them numerically. This property allows you to specify the method of approximating the gradient of the objective function. Choices are:
 - **Central:** Increases the accuracy of the gradient calculations by sampling from both sides of the sample point but increases the number of design point evaluations by 50%. This method makes use of the initial point, as well as the forward point and rear point. This is the default method for preexisting databases and new **Response Surface Optimization** systems.
 - **Forward:** Uses fewer design point evaluations but decreases the accuracy of the gradient calculations. This method makes use of only two design points, the initial point and forward point, to calculate the slope forward. This is the default method for new **Direct Optimization** systems.

- **Initial Finite Difference Delta (%)**: Advanced option for specifying the relative variation used to perturb the current point to compute gradients. Used in conjunction with **Allowable Convergence (%)** to ensure that the delta in NLPQL's calculation of finite differences is large enough to be seen above the noise in the simulation problem. This wider sampling produces results that are more clearly differentiated so that the difference is less affected by solution noise and the gradient direction is clearer. The value should be larger than both the value for **Initial Finite Difference Delta (%)** and the noise magnitude of the model. However, smaller values produce more accurate results, so set **Initial Finite Difference Delta (%)** only as high as necessary to be seen above simulation noise.

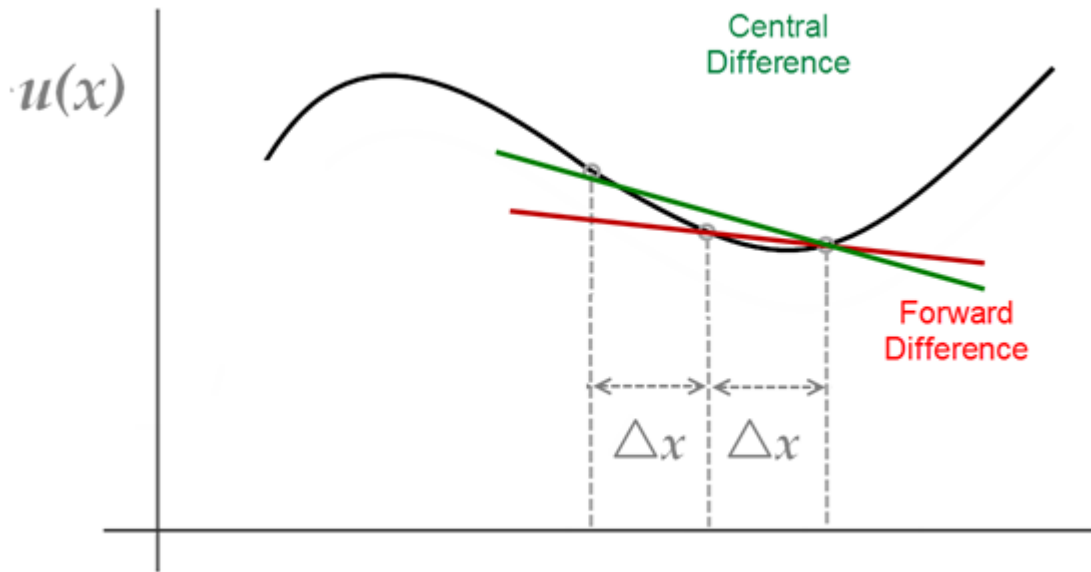
For a **Direct Optimization** system, the default percentage value is **1**. For a **Response Surface Optimization** system, the default percentage value is **0.001**. The minimum is **0.0001**, and the maximum is **10**.

For parameters with **Allowed Values** set to **Manufacturable Values** or **Snap to Grid**, the value for **Initial Finite Difference Delta (%)** is ignored. In such cases, the closest allowed value is used to determine the finite difference delta.

- **Allowable Convergence (%)**: Stop criterion. Tolerance to which the Karush-Kuhn-Tucker (KKT) optimality criterion is generated during the NLPQL process. A smaller value indicates more convergence iterations and a more accurate (but slower) solution. A larger value indicates fewer convergence iterations and a less accurate (but faster) solution.

For a **Direct Optimization** system, the default percentage value is **0.1**. For a **Response Surface Optimization** system, the default percentage value is **0.0001**. The maximum percentage value is **100**. These values are consistent across all problem types because the inputs, outputs, and gradients are scaled during the NLPQL solution.

- **Maximum Number of Iterations**: Stop criterion. Maximum number of iterations that the algorithm is to execute. If convergence happens before this number is reached, the iterations stop. This also provides an idea of the maximum possible number of function evaluations that are needed for the full cycle. For NLPQL, the number of evaluations can be approximated according to the Finite Difference Approximation gradient calculation method, as follows:
 - For **Central**: $\text{number of iterations} * (2 * \text{number of inputs} + 1)$
 - For **Forward**: $\text{number of iterations} * (\text{number of inputs} + 1)$
- **Maximum Number of Candidates**: Maximum number of candidates that the algorithm is to generate. The default is **3**. For more information, see [Viewing and Editing Candidate Points in the Table Pane](#) (p. 212).



3. Specify optimization objectives and constraints:

- In the **Outline** pane, select **Objectives and Constraints** or an object under it.
- In the **Table** or **Properties** pane, define the optimization objectives. For more information, see [Defining Optimization Objectives and Constraints \(p. 205\)](#).

Note:

For NLPQL, exactly one output parameter must have an objective defined. Multiple parameter constraints are permitted.

4. Specify the optimization domain:

- In the **Outline** pane, select **Domain** or an input parameter or parameter relationship under it.
- In the **Table** or **Properties** pane, define the selected domain object. For more information, see [Defining the Optimization Domain \(p. 201\)](#).

5. Update the **Optimization** cell.

The result is a group of points or sample set. The points that are most in alignment with the objective are displayed in the table as the candidate points for the optimization. In the **Properties** pane, the result **Size of Generated Sample Set** is read-only. This value is always equal to 1 for NLPQL.

6. In the **Outline** pane, select **Optimization**. Then, in the **Properties** pane under **Optimization Status**, view the outcome:

- **Converged:** Indicates whether the optimization has converged.

- **Number of Iterations:** Number of iterations executed. Each iteration corresponds to one formulation and solution of the quadratic programming subproblem, or alternatively, one evaluation of gradients.
 - **Number of Evaluations:** Number of design point evaluations performed. This value takes all points used in the optimization, including design points pulled from the cache. It can be used to measure the efficiency of the optimization method to find the optimum design point.
 - **Number of Failures:** Number of failed design points for the optimization. When a design point fails, NLPQL changes the direction of its search. It does not attempt to solve an additional design point in its place and does not include it on the Direct Optimization Samples chart.
 - **Size of Generated Sample Set:** Number of samples generated in the sample set. This is the number of iteration points obtained by the optimization and should be equal to the number of iterations.
 - **Number of Candidates:** Number of candidates obtained. This value is limited by the **Maximum Number of Candidates** input property.
7. In the **Outline** pane, select **Domain** or any node under it to review domain information in the **Properties** and **Table** panes.
 8. For a **Direct Optimization** system, in the **Outline** pane, select **Raw Optimization Data**. The **Table** pane displays the design points that were calculated during the optimization. If the raw optimization data point exists in the design points table for the **Parameter Set** bar, the corresponding design point name is indicated in parentheses in the **Name** column.

Note:

This list is compiled of raw data and does not show feasibility, ratings, and so on for the included design points.

Performing an MISQP Optimization

The MISQP (Mixed-Integer Sequential Quadratic Programming) method can be used for both **Response Surface Optimization** and **Direct Optimization** systems. It allows you to generate a new sample set to provide a more refined approach than the Screening method. MISQP is available for both continuous and discrete input parameters, which is why mixed is in its name. MISQP can handle only one output parameter goal. Other output parameters can be defined as constraints. For more information, see [Convergence Rate % and Initial Finite Difference Delta % in NLPQL and MISQP \(p. 353\)](#) and [Mixed-Integer Sequential Quadratic Programming \(MISQP\) \(p. 367\)](#) in the theory section.

To generate samples and perform a MISQP optimization:

1. In the **Outline** pane of the **Optimization** cell, select **Optimization**.
2. In the **Properties** pane, enter the following input properties:
 - **Method Selection:** Set to **Manual**.
 - **Method Name:** Set to **MISQP**.

- **Verify Candidate Points:** Select to verify candidate points automatically at end of the optimization update.
- **Finite Difference Approximation:** When analytical gradients are not available, MISQP approximates them numerically. This property allows you to specify the method of approximating the gradient of the objective function. Choices are:
 - **Central:** Increases the accuracy of the gradient calculations by sampling from both sides of the sample point but increases the number of design point evaluations by 50%. This method makes use of the initial point, as well as the forward point and rear point. This is the default method for preexisting databases and new **Response Surface Optimization** systems.
 - **Forward:** Uses fewer design point evaluations but decreases the accuracy of the gradient calculations. This method makes use of only two design points, the initial point and forward point, to calculate the slope forward. This is the default method for new **Direct Optimization** systems.
- **Initial Finite Difference Delta (%):** Advanced option for specifying the relative variation used to perturb the current point to compute gradients. Used in conjunction with **Allowable Convergence (%)** to ensure that the delta in MISQP's calculation of finite differences is large enough to be seen above the noise in the simulation problem. This wider sampling produces results that are more clearly differentiated so that the difference is less affected by solution noise and the gradient direction is clearer. The value should be larger than both the value for **Initial Finite Difference Delta (%)** and the noise magnitude of the model. Because smaller values produce more accurate results, set **Initial Finite Difference Delta (%)** only as high as necessary to be seen above simulation noise.

For a **Direct Optimization** system, the default percentage value is **1**. For a **Response Surface Optimization** system, the default percentage value is **0.001**. The minimum is **0.0001**, and the maximum is **10**.

For parameters with **Allowed Values** set to **Manufacturable Values** or **Snap to Grid**, the value for **Initial Finite Difference Delta (%)** is ignored. In such cases, the closest allowed value is used to determine the finite difference delta.

- **Allowable Convergence (%):** Stop criterion. Tolerance to which the Karush-Kuhn-Tucker (KKT) optimality criterion is generated during the MISQP process. A smaller value indicates more convergence iterations and a more accurate (but slower) solution. A larger value indicates fewer convergence iterations and a less accurate (but faster) solution.

For a **Direct Optimization** system, the default percentage value is **0.1**. For a **Response Surface Optimization** system, the default percentage value is **0.0001**. The maximum value is **100**. These values are consistent across all problem types because the inputs, outputs, and gradients are scaled during the MISQP solution.

- **Maximum Number of Iterations:** Stop criterion. Maximum number of iterations that the algorithm is to execute. If convergence happens before this number is reached, the iterations stop. This also provides an idea of the maximum possible number of function evaluations that are needed for the full cycle. For MISQP, the number of evaluations can be approximated according to the Finite Difference Approximation gradient calculation method, as follows:
 - For **Central**: $\text{number of iterations} * (2 * \text{number of inputs} + 1)$

– For **Forward**: `number of iterations * (number of inputs + 1)`

- **Maximum Number of Candidates**: Maximum number of candidates that the algorithm is to generate. The default is 3. For more information, see [Viewing and Editing Candidate Points in the Table Pane \(p. 212\)](#).

3. Specify optimization objectives and constraints:

- In the **Outline** pane, select **Objectives and Constraints** or an object under it.
- In the **Table** or **Properties** pane, define the optimization objectives. For more information, see [Defining Optimization Objectives and Constraints \(p. 205\)](#).

Note:

For MISQP, exactly one output parameter must have an objective defined, but multiple parameter constraints are permitted.

4. Specify the optimization domain:

- In the **Outline** pane, select **Domain** or an input parameter under it.
- In the **Table** or **Properties** pane, define the optimization domain. For more information, see [Defining the Optimization Domain \(p. 201\)](#).

5. Update the **Optimization** cell.

The result is a group of points or sample set. The points that are most in alignment with the objective are displayed in the table as the candidate points for the optimization. In the **Properties** pane, the **Size of Generated Sample Set** result is read-only. For MISQP, this value is always equal to 1.

6. In the **Outline** pane, select **Optimization**. Then, in the **Properties** pane under **Optimization Status**, view the outcome:

- **Converged**: Indicates whether the optimization has converged.
- **Number of Iterations**: Number of iterations executed. Each iteration corresponds to one formulation and solution of the quadratic programming subproblem, or alternatively, one evaluation of gradients.
- **Number of Evaluations**: Number of design point evaluations performed. This value takes all points used in the optimization, including design points pulled from the cache. It can be used to measure the efficiency of the optimization method to find the optimum design point.
- **Number of Failures**: Number of failed design points for the optimization. When a design point fails, MISQP changes the direction of its search. It does not attempt to solve an additional design point in its place and does not include it on the Direct Optimization Samples chart.
- **Size of Generated Sample Set**: Number of samples generated in the sample set. This is the number of design points updated in the last iteration.

- **Number of Candidates:** Number of candidates obtained. This value is limited by the **Maximum Number of Candidates** input property.
7. In the **Outline** pane, select **Domain** or any node under it to review domain information in the **Properties** and **Table** panes.
 8. For a **Direct Optimization** system, in the **Outline** pane, select **Raw Optimization Data**. The **Table** pane displays the design points that were calculated during the optimization. If the raw optimization data point exists in the design points table for the **Parameter Set** bar, the corresponding design point name is indicated in parentheses in the **Name** column.

Note:

This list is compiled of raw data and does not show feasibility, ratings, and so on for the included design points.

Performing an Adaptive Single-Objective Optimization

The Adaptive Single-Objective (OSF + Kriging + MISQP with domain reduction) method can be used only for **Direct Optimization** systems. This gradient-based method employs automatic intelligent refinement to provide the global optima. It requires a minimum number of design points to build the Kriging response surface, but, in general, this method reduces the number of design points necessary for the optimization. Failed design points are treated as inequality constraints, making it fault-tolerant.

The Adaptive Single-Objective method is available for input parameters that are continuous, including those with manufacturable values. It can handle only one output parameter goal, although other output parameters can be defined as constraints. It does not support the use of parameter relationships in the optimization domain. For more information, see [Adaptive Single-Objective Optimization \(ASO\)](#) (p. 368).

Note:

The Adaptive Single-Objective method requires advanced options. Ensure that the **Show Advanced Options** check box is selected on the **Design Exploration** tab of the **Options** window. Advanced options display in italic type.

To generate samples and perform an Adaptive Single-Objective optimization:

1. In the **Outline** pane of the **Optimization** cell, select **Optimization**.
2. In the **Properties** pane under **Optimization**, enter the following properties:
 - **Method Selection:** Set to **Manual**.
 - **Method Name:** Set to **Adaptive Single-Objective**.
 - **Number of Initial Samples:** Number of samples generated for the initial Kriging and after all domain reductions for the construction of the next Kriging.

You can enter a minimum of $(NbInp+1) * (NbInp+2) / 2$ (also the minimum number of OSF samples required for the Kriging construction) or a maximum of **10,000**. The default is $(NbInp+1) * (NbInp+2) / 2$ for a **Direct Optimization** system. There is no default for a **Response Surface Optimization** system.

Because of the Adaptive Single-Objective workflow (in which a new OSF sample set is generated after each domain reduction), increasing the number of OSF samples does not necessarily improve the quality of the results and significantly increases the number of evaluations.

- **Random Generator Seed:** Advanced option that displays only when **Type of Initial Sampling** is set to **Optimal Space-Filling**. The value for initializing the random number generator invoked internally by OSF. The value must be a positive integer. This property allows you to generate different samplings by changing the value or to regenerate the same sampling by keeping the same value. The default is **0**.
- **Maximum Number of Cycles:** Number of optimization loops that the algorithm needs, which in turns determines the discrepancy of the OSF. The optimization is essentially combinatorial, so a large number of cycles slows down the process. However, this makes the discrepancy of the OSF smaller. The value must be greater than **0**. For practical purposes, 10 cycles is usually good for up to 20 variables. The default is **10**.
- **Number of Screening Samples:** Number of samples for the screening generation on the current Kriging. This value is used to create the next Kriging (based on error prediction) and verified candidates.

You can enter a minimum of $(NbInp+1) * (NbInp+2) / 2$ (also the minimum number of OSF samples required for the Kriging construction) or a maximum of **10,000**. The default is $100 * NbInp$ for a **Direct Optimization** system. There is no default for a **Response Surface Optimization** system.

The larger the screening sample set, the better the chances of finding good verified points. However, too many points can result in a divergence of the Kriging.

- **Number of Starting Points:** Determines the number of local optima to explore. The larger the number of starting points, the more local optima explored. In the case of a linear surface, for example, it is not necessary to use many points. This value must be less than the value for **Number of Screening Samples** because these samples are selected in this sample. The default is the value for **Number of Initial Samples**.
- **Maximum Number of Evaluations:** Stop criterion. Maximum number of evaluations (design points) that the algorithm is to calculate. If convergence occurs before this number is reached, evaluations stop. This value also provides an idea of the maximum possible time it takes to run the optimization. The default is $20 * (NbInp + 1)$.
- **Maximum Number of Domain Reductions:** Stop criterion. Maximum number of domain reductions for input variation. (No information is known about the size of the reduction beforehand.) The default is **20**.
- **Percentage of Domain Reductions:** Stop criterion. Minimum size of the current domain according to the initial domain. For example, with one input ranging between 0 and 100, the domain size is equal to 100. The percentage of domain reduction is 1%, so the current working domain size cannot be less than 1 (such as an input ranging between 5 and 6). The default is **0.1**.

- **Convergence Tolerance:** Stop criterion. Minimum allowable gap between the values of two successive candidates. If the difference between two successive candidates is smaller than the value for **Convergence Tolerance** multiplied by the maximum variation of the parameter, the algorithm is stopped. A smaller value indicates more convergence iterations and a more accurate (but slower) solution. A larger value indicates fewer convergence iterations and a less accurate (but faster) solution. The default is **1E-06**.
 - **Retained Domain per Iteration (%):** Advanced option that allows you to specify the minimum percentage of the domain you want to keep after a domain reduction. The percentage value must be between **10** and **90**. A larger value indicates less domain reduction, which implies better exploration but a slower solution. A smaller value indicates a faster and more accurate solution, with the risk of it being a local one. The default percentage value is **40**.
 - **Maximum Number of Candidates:** Maximum number of candidates that the algorithm is to generate. The default is **3**. For more information, see [Viewing and Editing Candidate Points in the Table Pane \(p. 212\)](#).
3. Specify optimization objectives and constraints:
- In the **Outline** pane, select **Objectives and Constraints** or an object under it.
 - In the **Table** or **Properties** pane, define the optimization objectives. For more information, see [Defining Optimization Objectives and Constraints \(p. 205\)](#).

Note:

- For the Adaptive Single-Objective method, exactly one output parameter must have an objective defined.
 - After you have defined an objective, a warning icon displays in the **Message** column of the **Outline** pane if the recommended number of input parameters has been exceeded. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).
-

4. Specify the optimization domain:
- In the **Outline** pane, select **Domain** or an input parameter under it.
 - In the **Table** or **Properties** pane, define the optimization domain. For more information, see [Defining the Optimization Domain \(p. 201\)](#).

5. Update the **Optimization** cell.

The result is a group of points or sample set. The points that are most in alignment with the objectives are displayed in the table as the candidate points for the optimization. In the **Properties** pane, the **Size of Generated Sample Set** result is read-only.

6. In the **Outline** pane, select **Optimization**. Then, in the **Properties** pane under **Optimization Status**, view the outcome:
- **Converged:** Indicates whether the optimization has converged.

- **Number of Evaluations:** Number of design point evaluations performed. This value takes into account all points used in the optimization, including design points pulled from the cache. It corresponds to the total of LHS points and verification points.
 - **Number of Domain Reductions:** Number of times the domain is reduced.
 - **Number of Failures:** Number of failed design points for the optimization. When a design point fails, the Adaptive Single-Objective method changes the direction of its search. It does not attempt to solve an additional design point in its place and does not include it on the Direct Optimization Samples chart.
 - **Size of Generated Sample Set:** Number of samples generated in the sample set. This is the number of unique design points that have been successfully updated.
 - **Number of Candidates:** Number of candidates obtained. This value is limited by the **Maximum Number of Candidates** input property.
7. For a **Direct Optimization** system, in the **Outline** pane, select **Raw Optimization Data**. The **Table** displays the design points that were calculated during the optimization. If the raw optimization data point exists in the design points table for the **Parameter Set** bar, the corresponding design point name is indicated in parentheses in the **Name** column.

Note:

This list is compiled of raw data and does not show feasibility, ratings, and so on for the included design points.

Performing an Adaptive Multiple-Objective Optimization

The Adaptive Multiple-Objective (Kriging + MOGA) method can be used only for **Direct Optimization** systems. It is an iterative algorithm that allows you to either generate a new sample set or use an existing set, providing a more refined approach than the Screening method. It uses the same general approach as MOGA, but applies the Kriging error predictor to reduce the number of evaluations needed to find the global optimum.

The Adaptive Multiple-Objective method is available only for continuous input parameters, including those with manufacturable values. It can handle multiple objectives and multiple constraints. For more information, see [Adaptive Multiple-Objective Optimization \(p. 379\)](#).

Note:

The Adaptive Multiple-Objective method requires advanced options. Ensure that the **Show Advanced Options** check box is selected on the **Design Exploration** tab of the **Options** window. Advanced options display in italic type.

To generate samples and perform an Adaptive Multiple-Objective optimization:

1. In the **Outline** pane for the **Optimization** cell, select **Optimization**.
2. In the **Properties** pane under **Optimization**, enter the following properties:

- **Method Selection:** Set to **Manual**.
- **Method Name:** Set to **Adaptive Multiple-Objective**.
- **Type of Initial Sampling:** If you do not have any parameter relationships defined, set to **Screening** (default) or **Optimal Space-Filling**. If you do have parameter relationships defined, the initial sampling must be performed by the constrained sampling algorithms (because parameter relationships constrain the sampling). In such cases, this property is automatically set to **Constrained Sampling**.
- **Random Generator Seed:** Advanced option that displays only when **Type of Initial Sampling** is set to **Optimal Space-Filling**. Value for initializing the random number generator invoked internally by the Optimal Space-Filling (OSF) algorithm. The value must be a positive integer. This property allows you to generate different samplings by changing the value or to regenerate the same sampling by keeping the same value. The default is **0**.
- **Maximum Number of Cycles:** Determines the number of optimization loops the algorithm needs, which in turns determines the discrepancy of the OSF. The optimization is essentially combinatorial, so a large number of cycles slows down the process. However, this makes the discrepancy of the OSF smaller. The value must be greater than **0**. For practical purposes, 10 cycles is usually good for up to 20 variables. The default is **10**.
- **Number of Initial Samples:** Initial number of samples to use. This number must be greater than the number of enabled inputs. The minimum recommended number of initial samples is 10 times the number of enabled input parameters. The larger the initial sample set, the better your chances of finding the input parameter space that contains the best solutions.

The number of enabled input parameters is also the minimum number of samples required to generate the Sensitivities chart. You can enter a minimum of **2** and a maximum of **10000**. The default is **100**.

If you are switching the method from **Screening** to **MOGA**, MOGA generates a new sample set. For the sake of consistency, enter the same number of initial samples as used for the Screening method.

- **Maximum Allowable Pareto Percentage:** Convergence criterion. Percentage value representing the ratio of the number of desired Pareto points to the **Number of Samples Per Iteration**. When this percentage is reached, the optimization is converged. For example, a value of 70% with **Number of Samples Per Iteration** set at 200 samples would mean that the optimization should stop once the resulting front of the MOGA optimization contains at least 140 points. Of course, the optimization stops before that if the **Maximum Number of Iterations** is reached.

If the **Maximum Allowable Pareto Percentage** is too low (below 30), the process can converge prematurely, and if it is too high (above 80), it can converge slowly. The value of this property depends on the number of parameters and the nature of the design space itself. The default is **70**. Using a value between **55** and **75** works best for most problems. For more information, see [Convergence Criteria in MOGA-Based Multi-Objective Optimization](#) (p. 373).

- **Convergence Stability Percentage:** Convergence criterion. Percentage value representing the stability of the population, based on its mean and standard deviation. This property

allows you to minimize the number of iterations performed while still reaching the desired level of stability. When the specified percentage is reached, the optimization is converged. The default percentage value is **2**. To not take the convergence stability into account, set the percentage value to **0**. For more information, see [Convergence Criteria in MOGA-Based Multi-Objective Optimization](#) (p. 373).

- **Maximum Number of Iterations:** Stop criterion. Maximum number of iterations that the algorithm is to execute. If this number is reached without the optimization having reached convergence, iterations stop. This also provides an idea of the maximum possible number of function evaluations that are needed for the full cycle, as well as the maximum possible time it can take to run the optimization. For example, the absolute maximum number of evaluations is given by: **Number of Initial Samples + Number of Samples Per Iteration * (Maximum Number of Iterations - 1)**.
 - **Mutation Probability:** Advanced option for specifying the probability of applying a mutation on a design configuration. The value must be between **0** and **1**. A larger value indicates a more random algorithm. If the value is **1**, the algorithm becomes a pure random search. A low probability of mutation (<0.2) is recommended. The default is **0.01**. For more information on mutation, see [MOGA Steps to Generate a New Population](#) (p. 376).
 - **Crossover Probability:** Advanced option for specifying the probability with which parent solutions are recombined to generate offspring solutions. The value must be between **0** and **1**. A smaller value indicates a more stable population and a faster (but less accurate) solution. If the value is **0**, the parents are copied directly to the new population. A high probability of crossover (>0.9) is recommended. The default is **0.98**.
 - **Maximum Number of Candidates:** Maximum number of candidates for the algorithm to generate. The default is **3**. For more information, see [Viewing and Editing Candidate Points in the Table Pane](#) (p. 212).
 - **Type of Discrete Crossover:** Advanced option for specifying the kind of crossover for discrete parameters. This property is visible only if there is at least one discrete input variable or continuous input variable with manufacturable values. Three crossover types are available: **One Point**, **Two Points**, and **Uniform**. According to the type of crossover selected, the children are closer to or farther from their parents. Children are closer for **One Point** and farther for **Uniform**. The default is **One Point**. For more information on crossover, see [MOGA Steps to Generate a New Population](#) (p. 376).
 - **Maximum Number of Permutations:** Read-only property showing the number of combinations possible for parameter values. This property is visible only if all input parameters are discrete or continuous with manufacturable values. The value corresponds to the product of the number of levels per input parameter. For example, if the optimization problem contains 2 discrete parameters with 4 and 5 levels respectively and 1 continuous parameter with 6 manufacturable values, **Maximum Number of Permutations** is equal to 120 (4*5*6).
3. Specify optimization objectives and constraints:
- In the **Outline** pane, select **Objectives and Constraints** or an object under it.

- In the **Table** or **Properties** pane, define the optimization objectives. For more information, see [Defining Optimization Objectives and Constraints \(p. 205\)](#).

Note:

- For the Adaptive Multiple-Objective method, at least one output parameter must have an objective defined. Multiple objectives are allowed.
 - After you have defined an objective, a warning icon displays in the **Message** column of the **Outline** pane if the recommended number of input parameters is exceeded. For more information, see [Number of Input Parameters for DOE Types \(p. 90\)](#).
-

4. Specify the optimization domain:

- In the **Outline** pane, select **Domain** or an input parameter or parameter relationship under it.
- In the **Table** or **Properties** pane, define the selected domain object. For more information, see [Defining the Optimization Domain \(p. 201\)](#).

5. Update the **Optimization** cell.

The result is a group of points or sample set. The points that are most in alignment with the objectives are displayed in the table as the candidate points for the optimization. In the **Properties** pane, the result **Size of Generated Sample Set** is read-only.

6. In the **Outline** pane, select **Optimization**. Then, in the **Properties** pane under **Optimization Status**, view the outcome:

- **Converged:** Indicates whether the optimization has converged.
- **Number of Evaluations:** Number of design point evaluations performed. This value takes all points used in the optimization, including design points pulled from the cache. It can be used to measure the efficiency of the optimization method to find the optimum design point.
- **Number of Failures:** Number of failed design points for the optimization. When a design point fails, an Adaptive Multiple-Objective optimization does not retain this point in the Pareto front to generate the next population, attempt to solve an additional design point in its place, or include it on the Direct Optimization Samples chart.
- **Size of Generated Sample Set:** Number of samples generated in the sample set. This is the number of samples successfully updated for the last population generated by the algorithm. It usually equals the **Number of Samples Per Iteration**.
- **Number of Candidates:** Number of candidates obtained. This value is limited by the **Maximum Number of Candidates** input property.

7. In the **Outline** pane, select **Domain** or any object under it to view domain data in the **Properties** and **Table** panes.

- For a **Direct Optimization** system, select **Raw Optimization Data** in the **Outline** pane. The **Table** pane displays the design points that were calculated during the optimization. If the raw optimization data point exists in the design points table for the **Parameter Set** bar, the corresponding design point name is indicated in parentheses in the **Name** column.

Note:

This list is compiled of raw data and does not show feasibility, ratings, and so on for the included design points.

Performing an Optimization with an External Optimizer

In addition to the optimization algorithms provided by DesignXplorer, you can use external optimizers. When you install and load optimization extensions, the features of the external optimizers are integrated into your design exploration workflow. An optimization extension specifies the properties, objectives and constraints, and functionality for one or more optimizers.

Once an optimization extension is installed and loaded to the project as described in [Working with DesignXplorer Extensions](#) (p. 320), you are ready to start using its extended functionality.

Selecting an External Optimizer

When an optimization extension is loaded to your project, you can select and use this external optimizer. In the **Properties** pane for the **Optimization** cell, the **Method Name** list includes all optimization methods provided by DesignXplorer and all loaded external optimizers.

DesignXplorer filters the **Method Name** list for applicability to the current project, displaying only those optimization methods that you can use to solve the optimization problem as it is currently defined. When no objectives or constraints are defined for a project, all optimization methods are listed. If you already know that you want to use a particular external optimizer, you should select it as the method before setting up the rest of the project. Otherwise, the optimization method could be inadvertently filtered from the list.

Setting Up an External Optimizer Project

Once an external optimizer is selected, you can edit its optimization properties.

DesignXplorer shows only the optimization functionality that is specifically defined in the extension. Additionally, DesignXplorer filters objectives and constraints according to the optimization method selected, making only those objects supported by the selected optimization method available for selection. For example, if you have selected an optimizer that does not support the **Maximize** objective type, **Maximize** is not included in the **Objective Type** list.

If you already have a specific problem you want to solve, you should set up the project before selecting an optimization method for **Method Name**. Otherwise, the desired objectives and constraints could be filtered from the lists.

Performing the Optimization

Once the external optimizer is selected and the optimization study is defined, the process of running the optimization is the same as for any native optimization algorithm. DesignXplorer's intermediate and postprocessing functionalities are still available. Intermediate functionalities includes iterative updates of History charts and sparklines. Postprocessing functionalities includes candidate points and other optimization charts.

Defining the Optimization Domain

When you edit the **Optimization** cell of a goal-driven optimization system, you can define the parameter space for each input parameter in multiple ways. The optimization domain settings allow you to reduce the range of variation for the input by defining bounds. Samples are then generated only within this reduced space, ensuring that can be used in your optimization. You can do this for two types of domain objects: input parameters and parameter relationships. Both types are under **Domain** in the **Outline** pane.

When you select **Domain** or any object under it, the **Table** pane displays the input parameters and parameter relationships that are defined and enabled for the optimization. It does not display disabled domain objects.

Cell A3: Analysis Schematic B2: Optimization			
A	B	C	
1	Enabled	Monitoring	
2	<input checked="" type="checkbox"/>	Optimization	
3	<input checked="" type="checkbox"/>	Objectives and Constraints	
4		Minimize P3	
5	<input checked="" type="checkbox"/>	Domain	
6		Microsoft Office Excel (A1)	
7	<input checked="" type="checkbox"/>	P1 - x1_	
8	<input checked="" type="checkbox"/>	P2 - x2_	
9	<input checked="" type="checkbox"/>	Parameter Relationships	
10	<input checked="" type="checkbox"/>	P2 >= P1	
11	<input checked="" type="checkbox"/>	P2-P1 <= 3[mm]	

Table of Schematic B2: Optimization			
A	B	C	D
1	Input Parameters		
2	Name	Lower Bound	Upper Bound
3	P1 - x1_ (mm)	0	30
4	P2 - x2_ (mm)	0	30
5	Parameter Relationships		
6	Name	Left Expression	Operator Right Expression
7	P2 >= P1	P2	>= P1
8		0 [m]	0 [m]
9	P2-P1 <= 3[mm]	P2-P1	<= 3[mm]
10		0 [m]	0.003 [m]
*	New Parameter Relationship	New Expression	<= New Expression

Defining Input Parameters

Under **Domain** in the **Outline** pane, you can enable or disable the input parameters for the parameter space by selecting and clearing check boxes in the **Enabled** column. When an input parameter is enabled, you can edit the bounds and, where applicable, the starting value using either of these methods:

- Select **Domain** and then edit the input parameter domain in the **Table** pane.
- Select an input parameter under **Domain** and then edit the input parameter domain in either the **Properties** or **Table** pane.

For enabled input parameters in the **Properties** and **Table** panes, the following settings are available:

Lower Bound

Defines the lower bound for the optimization input parameter space. Increasing the lower bound confines the optimization to a subset of the DOE domain. By default, the lower bound corresponds to the following values defined in the DOE:

- **Lower Bound** for continuous parameters
- Lowest discrete **Level** for discrete parameters
- Lowest manufacturable **Level** for continuous parameters with manufacturable values

Upper Bound

Defines the upper bound for the input parameter space. By default, the upper bound corresponds to the following values defined in the DOE:

- **Upper Bound** for continuous parameters
- Highest discrete **Level** for discrete parameters
- Highest manufacturable **Level** for continuous parameters with manufacturable values

Starting Value

Available only for NLPQL and MISQP. Specifies where the optimization starts for each input parameter.

Because NLPQL and MISQP are gradient-based methods, the starting point in the parameter space determines the candidates found. With a poor starting point, NLPQL and MISQP might find a local optimum, which is not necessarily the same as the global optimum. This setting gives you more control over your optimization results by allowing you specify exactly where in the parameter space the optimization should begin.

- Must fall between the **Lower Bound** and **Upper Bound**.
- Must fall within the domain constrained by the enabled parameter relationships. For more information, see [Defining Parameter Relationships \(p. 202\)](#).

For each disabled input parameter, specify the desired value to use in the optimization. By default, the value is copied from the current design point when the optimization system was created.

Note:

When the optimization is refreshed, disabled input values persist. However, they are not updated to the current design point values.

Defining Parameter Relationships

Parameter relationships give you greater flexibility in defining optimization limits on input parameters than standard single-parameter constraints such as **Greater Than**, **Less Than**, and **Inside Bounds**. When you define parameter relationships, you can specify expression-based relationships between

multiple input parameters, with the values remaining physically bounded and reflecting the constraints on the optimization problem.

Note:

To specify parameter relationships for outputs, you can create derived parameters. You create derived parameters for an analysis system by providing expressions. The derived parameters are then passed to DesignXplorer as outputs. For more information, see [Creating or Editing Parameters](#) in the *Workbench User's Guide*.

A parameter relationship has one operator and two expressions. In the following example, you can see that two parameter relationships have been defined, each involving input parameters **P1** and **P2**.

Table of Schematic C4: Optimization			
	A	B	C
1	Input Parameters		
2	Name	Lower Bound	Upper Bound
3	P1 - WB_B (mm)	1.8	2.2
4	P2 - WB_D (mm)	4.5	5.5
5	P3 - WB_L (mm)	90	110
6	P4 - WB_P (N)	900	1100
7	P5 - WB_E (MPa)	1.8E+05	2.2E+05
8	Starting Value	2	5
9	100	1000	2E+05
Parameter Relationships			
	Name	Left Expression	Operator
10	2*P1 >= P2	2*P1	>=
11		4 [mm]	
12	P1+P2 <= 0.3[n]	P1+P2	<=
13		7 [mm]	
*	New Parameter Relationship	New Expression	<=

You can create, edit, enable and disable, and view parameter relationships.

Creating Parameter Relationships

You can create new domain parameter relationships using either of these methods:

- In the **Outline** pane, right-click **Parameter Relationships** and select **Insert Parameter Relationship**.
- In the **Table** pane under **Parameter Relationships**, enter parameter relationship data in the bottom row.

Editing Parameter Relationships

Once a parameter relationship has been created, you can edit it using either of these methods:

- In the **Outline** pane, select **Domain** or **Parameter Relationships** under it. Then, edit the parameter relationship in the **Table** pane.
- In the **Outline** pane under **Parameter Relationships**, select a parameter relationship. Then, edit the parameter relationship in the **Properties** or **Table** pane.

The following table indicates the editing tasks that the **Properties**, **Table**, and **Outline** panes allow you to perform.

Task	Properties	Table	Outline
Rename a relationship by double-clicking its name and typing a custom name	X	X	X
Edit expressions by typing in the expression cell	X	X	
Change operators by selecting from the drop-down menu	X	X	
Duplicate a relationship by right-clicking it and selecting Duplicate	X	X	X
Delete a relationship by right-clicking it and selecting Delete	X	X	X

Enabling and Disabling Parameter Relationships

You can enable or disable a parameter relationship by selecting or clearing the **Enabled** check box. Enabling a parameter relationship causes it to display on the corresponding History chart and sparkline. For more information, see [Using the History Chart \(p. 219\)](#).

Parameter Relationship Properties

For parameter relationships, the following properties are available:

Name

Editable in the **Properties** and **Table** panes. Each parameter relationship is given a default name such as **Parameter** or **Parameter 2**, based on the order in which the parameter relationship was created. When you define both the left and right expressions for the parameter relationship, the default name is replaced by the relationship. For example, v can become **P1<=P2**. The name is updated accordingly when either of the expressions is modified.

The **Name** property allows you to edit the name of the parameter relationship. Once you edit this property, the name persists. To resume the automated naming system, you must delete the custom name, leaving the property empty.

Left Expression and Right Expression

Editable in the **Properties** and **Table** panes. Allows you to define the parameter relationship expressions.

Left Expression Quantity Name and Right Expression Quantity Name

Viewable in the **Properties** pane. Shows the quantity type for the expressions in the parameter relationship.

Operator

Editable in the **Properties** and **Table** panes. Allows you to select the expression operator from a drop-down menu. Available values are **<=** and **>=**.

Viewing Current Expression Values

For optimization methods that have a **Starting Value** property (NLPQL and MISQP), each expression for a parameter relationship is evaluated based on the starting parameter values.

When the evaluation is complete, the value for each expression is displayed:

- Under **Domain** in the **Table** pane. To view expression values for a parameter relationship, click the plus icon next to the name. The values display below the corresponding expression.
- Under **Candidate Points** in the **Table** pane. In the **Properties** pane, select the **Show Parameter Relationships** check box. Parameter relationships that are defined and enabled, along with their expressions and current expression values for NLPQL and MISQP, are shown in the candidate points table in the **Table** pane. For more information, see [Viewing and Editing Candidate Points in the Table Pane \(p. 212\)](#).

If the evaluation fails, the **Outline** and **Properties** panes display an error message. Review the error to identify problems with the corresponding parameter relationship.

Note:

- The evaluation can fail because the selected optimization method does not support parameter relationships or because the optimization includes one or more invalid parameter relationships. Parameter relationships can be invalid if they contain quantities that are not comparable, parameters for which the values are unknown, or expressions that are incorrect.
- The gradient-based optimization methods, NLPQL and MISQP, require a feasible value for **Starting Value** that falls within the domain constrained by the enabled parameter relationships. If this value is infeasible, the optimization cannot be updated.

Defining Optimization Objectives and Constraints

In the **Optimization** cell of a goal-driven optimization system, you define design goals in the form of objectives and constraints to generate optimized designs. After selecting **Objectives and Constraints** in the **Outline** pane, you insert objectives and constraints in the **Table** pane:

	A	B	C	D	E	F	G	H	I	J
1	Name	Parameter	Objective				Constraint			
2			Type	Initial	Target	Tolerance	Type	Lower Bound	Upper Bound	Tolerance
3	Minimize P8; P8 <= 10 mm	P8 - Displacement	Minimize	4.8	3		Values <= Upper Bound		10	0.01
4	Seek P6 = 9E+05 mm^3	P6 - Volume	Seek Target	9.45E+05	9E+05	1000	No Constraint			
5	Minimize P7; P7 <= 110 MPa	P7 - Stress	Minimize	92	70		Values <= Upper Bound		110	1
*		Select a Parameter								

Additional optimization properties can be set in the **Properties** pane. The optimization approach used in the design exploration environment departs in many ways from traditional optimization techniques, giving you added flexibility in obtaining the desired design configuration.

Note:

If you are using an external optimizer, DesignXplorer filters objectives and constraints according to the optimization method selected, displaying only the types that the method supports. For example, if you have selected an optimizer that does not support the **Maximize** objective type, DesignXplorer does not display **Maximize** as a choice.

In both the **Table** and **Properties** pane, the following optimization options are available:

Name (Table pane) or Objective Name (Properties pane)

After you select a parameter in the empty row of the table, DesignXplorer automatically assigns the name of the objective or constraint based on the properties that you then define for it. If you later change these properties, DesignXplorer automatically changes the name.

For example, assume that you have selected parameter **P1** in the empty row and set the objective type to **Minimize**. DesignXplorer assigns **Minimize P1** as the name. If you change the objective type to **Maximize**, DesignXplorer changes the name to **Maximize P1**. If you then add a constraint type of **Values >= Bound** and set **Lower Bound** to **3**, DesignXplorer changes the name to **Maximize P1; P1 >= 3**.

In either the **Table** pane or **Outline** pane, you can manually change the name of an objective or constraint. Any custom name that you assign is persisted, which means that DesignXplorer no longer changes the name if you change the properties. To restore automated naming, you delete the custom name, leaving the option empty. When you click elsewhere, DesignXplorer will restore automated naming.

Available options depend on the type of parameter and whether it is an input or output.

Parameter Type	Supported Settings
Continuous input without manufacturable variables	Objective only
Continuous input with manufacturable variables	Constraint only
Discrete input	Constraint only
Output	Objective and Constraint

Parameter

In the last row in the **Table** pane, this option allows you to select the input or output parameter for which to add an objective or constraint. In the newly inserted row, you then define properties for this parameter. For existing rows, this option is display-only. However, you can delete rows.

Objective Type

Available for continuous input parameters without manufacturable values and output parameters. Allows you to define an objective by setting the objective type. See the following tables for available objective types.

Objective Initial

Visible only when tolerance settings are enabled and the **Solution Process Update** property for the **Parameter Set** bar is set to **Submit to Design Point Service (DPS)**. This property does not affect DesignXplorer but rather is included in the fitness terms that DesignXplorer sends when the update of an optimization study is sent to DPS. For more information, see [Initial Values for Objectives to Send to DPS](#) (p. 209).

Objective Target

For a parameter with an objective, allows you to set the best estimated goal value that the optimization method can achieve for the objective. This value is not a stopping criterion. If the optimization method can find a better value than the target value, it will do so. On the history chart and sparkline, the target value is shown by a dashed line.

Objective Tolerance

Visible when tolerance settings are enabled. For a parameter with a **Seek Target** objective type, allows you to set the level of accuracy for reaching the target value. The tolerance value is not a strict constraint to satisfy but rather a goal to reach. The tolerance value must be positive. For more information, see [Tolerance Settings](#) (p. 209).

Constraint Type

Available for continuous input parameters with manufacturable values, discrete input parameters, and output parameters. Allows you to define a constraint by setting the constraint type. See the following tables for available constraint types.

Constraint Lower Bound and Constraint Upper Bound

Depending on the constraint type, allows you to set one or more values for limiting the target value for the constraint.

- For a discrete input parameter or a continuous input parameter with manufacturable variables, set the lower or upper limit for the input value.
- For an output parameter, set the range for limiting the target value for the constraint.

Constraint Tolerance

Allows you to set a feasibility tolerance. The optimization method considers any point with a constraint violation less than the tolerance value as a feasible point. The tolerance value must be positive.

Objectives for Continuous Input Parameters

Objectives	Description	Mathematical Meaning
$(X = \text{input parameter}, X_{\text{Lower}} = \text{Lower Bound}, X_{\text{Upper}} = \text{Upper Bound}, X_{\text{Target}} = \text{Target})$		
No Objective	Keep the input parameter within its upper and lower bounds.	$X_{\text{Lower}} \leq X \leq X_{\text{Upper}}$
Minimize	Minimize the input parameter within its range.	$X \rightarrow X_{\text{Lower}}$

Objectives	Description	Mathematical Meaning
Maximize	Maximize the input parameter within its range.	$X \rightarrow X_{Upper}$
Seek Target	Achieve an input parameter value that is close to the objective target.	$X \rightarrow X_{Target}$

Objectives for Output Parameters

Objective	Description	Mathematical Meaning	GDO Treatment
(Y = output parameter, $Y_{Target} = Target$)			
No Objective	No objective is specified.	N/A	N/A
Minimize	Achieve the lowest possible value for the output parameter.	Minimize Y	Objective
Maximize	Achieve the highest possible value for the output parameter.	Maximize Y	Objective
Seek Target	Achieve an output parameter value that is close to the objective target.	$Y \rightarrow Y_{Target}$	Objective

Constraints for Discrete Input Parameters or Continuous Input Parameters with Manufacturable Values

Constraints	Description	Mathematical Meaning
(X = an input parameter, $X_{Lower} = Lower Bound$, $X_{Upper} = Upper Bound$, $X_{Target} = Target$)		
No Constraint	Uses the full discrete range allowed for the parameter.	$X_{Lower} \leq X \leq X_{Upper}$
Values = Bound	Set the constraint to in-range values close to the lower bound.	$X \rightarrow X_{LowerBound}$
Values \geq Lower Bound	Set the constraint to in-range values above the lower bound.	$X \geq X_{LowerBound}$
Values \leq Upper Bound	Set the constraint to in-range values below the lower bound.	$X \leq X_{UpperBound}$

Constraints for Output Parameters

Constraints	Description	Mathematical Meaning	GDO Treatment
(Y = output parameter, $Y_{Target} = Target$)			
No Constraint	No constraint is specified.	N/A	N/A
Values = Bound	If a lower bound is specified, then achieve an output parameter value that is close to that bound.	$Y \rightarrow Y_{LowerBound}$	Constraint
Values \geq Lower Bound	If a lower bound is specified, then achieve an output parameter value that is above the bound.	$Y \geq Y_{LowerBound}$	Inequality constraint

Constraints	Description	Mathematical Meaning	GDO Treatment
Values \leq Upper Bound	If an upper bound is specified, then achieve an output parameter value that is below that bound.	$Y \leq Y_{UpperBound}$	Inequality constraint
Lower Bound \leq Values \leq Upper Bound	If a lower bound and upper bound are specified, then achieve an output parameter that is within the defined range.	$Y_{LowerBound} \leq Y \leq Y_{UpperBound}$ given $Y_{LowerBound} < Y_{UpperBound}$	Constraint

Tolerance Settings

DesignXplorer optimization methods can use tolerance settings to improve convergence and the relevance of results. The Decision Support Process can also use tolerance settings to sort candidate points and define their ratings values.

	A	B	C	D	E	F	G	H	I	J
1	Name	Parameter	Objective			Constraint				
2			Type	Initial	Target	Tolerance	Type	Lower Bound	Upper Bound	Tolerance
3	Minimize P8; P8 \leq 10 mm	P8 - Displacement	Minimize	4.8	3		Values \leq Upper Bound		10	0.01
4	Seek P6 = 9E+05 mm ³	P6 - Volume	Seek Target	9.45E+05	9E+05	1000	No Constraint			
5	Minimize P7; P7 \leq 110 MPa	P7 - Stress	Minimize	92	70		Values \leq Upper Bound		110	1
*		Select a Parameter								

Note:

In the Workbench **Options** window under **Design Exploration** → **Sampling and Optimization**, the **Tolerance Settings** check box is selected by default. This preference value is used to initialize the **Tolerance Settings** check box in the properties for the **Optimization** cell when an optimization system is newly inserted in the **Project Schematic**. When this check box is selected, tolerance values can be entered for objectives of the **Seek Target** type and for constraints. Additionally, if the **Solution Process Update** property for the **Parameter Set** bar is set to **Submit to Design Point Service (DPS)**, the **Initial** option is shown for objectives. The next section describes how DesignXplorer initializes and sends these values to DPS.

Initial Values for Objectives to Send to DPS

The **Initial** option displays only if tolerance values are enabled and the **Solution Process Update** property for the **Parameter Set** bar is set to **Submit to Design Point Service (DPS)**. Initial values do not affect DesignXplorer but rather are included in the fitness terms that DesignXplorer sends when an update of an optimization study is sent to DPS. For more information, see [Sending an Optimization Study to DPS \(p. 319\)](#).

- For an input parameter, this property is initialized as follows:

- If the selected optimization method requires a starting value, the initial value is synchronized with the starting value.
- If the selected optimization method does not require a starting value, the initial value is set based on values for the project's current design point. The initial value must be consistent with the input parameter type (continuous or discrete). For continuous input parameters, the initial value must also be consistent with the setting for the **Allowed Values** property (**Any, Manufacturable Values, or Snap to Grid**), meaning it must be set to the closest discrete level, closest manufacturable value, or closest point on the grid.
- For an output parameter, if the design point is up-to-date or has been already updated formerly, this property is initialized from the values of the current design point of the project. Otherwise, if the design point has never been updated, this property is blank and is highlighted in yellow to indicate that a value is required.

To update an **Optimization** cell using DPS, the type, target, and initial values for all objectives must be consistent.

Note:

For Workbench projects created in 2019 R3 or earlier, no changes are required for an up-to-date **Optimization** cell in which tolerance settings are enabled. However, if you make any change to the optimization definition, initial values must be present before an update can be submitted to DPS successfully.

Tolerance Values for MOGA and AMO Methods

For the MOGA and AMO methods, tolerance values are used in the selection of the best parents to generate the next population of points. The objective and constraint definitions are formulated through a single objective (fitness function) to minimize. This fitness function is a weighted sum of fitness terms, where each fitness term represents the rating for an objective or a constraint. The weight of an objective (or a constraint) is defined by its importance value, which is a postprocessing property described in the next section.

For a given design point P , the fitness function can be written as:

$$F(P) = \sum_i w_i D_i(P)$$

Where $D_i(P)$ represents the rating for an objective or a constraint and w_i corresponds to its relative weight.

Note:

While tolerance settings are available for external optimizers in DesignXplorer, they are not available for external optimizers in the DesignXplorer API.

Postprocessing Properties

Under **Decision Support Process** in the **Properties** pane, the following postprocessing properties for objectives and constraints are available:

Decision Support Process	
Objective Importance	Default
Constraint Importance	Default
Constraint Handling	Strict

Objective Importance

For a parameter with an objective, allows you to select the relative importance of this parameter compared to other objectives. Choices are **Default**, **Higher**, and **Lower**.

Constraint Importance

For a parameter with a constraint defined, allows you to select the relative importance of this parameter compared to other constraints. Choices are **Default**, **Higher**, and **Lower**.

Constraint Handling

For a parameter with a constraint defined, allows you to specify the handling of the constraint for this parameter. This option can be used for any optimization application and is best thought of as a *constraint satisfaction filter* on samples generated from optimization runs. It is especially useful for screening samples to detect the edges of solution feasibility for highly constrained nonlinear optimization problems. Choices are:

- **Relaxed:** Samples are generated in the full parameter space, with the constraint only being used to identify the best candidates. When constraint handling is relaxed, the upper, lower, and equality constrained objectives of the candidate points are treated as objectives. Therefore, any violation of the objective is still considered feasible.
- **Strict:** Samples are generated in the reduced parameter space defined by the constraint. When constraint handling is strict (default), the upper, lower, and equality constraints are treated as hard constraints. If any of these constraints is violated, the candidate point is no longer shown.

Objectives, Constraints, and Optimization Methods

Objective and constraint requirements depend on the optimization method:

- Screening requires that an objective or a constraint is defined for at least one parameter. Multiple output objectives are allowed.
- MOGA and Adaptive Multiple-Objective require that an objective is defined for at least one parameter. Multiple output objectives are allowed.
- NLPQL, MISQP, and Adaptive Single-Objective require that an objective is defined for one parameter. Only a single output objective is allowed.

Working with Candidate Points

DesignXplorer provides multiple ways of viewing and working with candidate points, depending on the node selected in the **Outline** pane for the **Optimization** cell.

- When you select **Optimization**, the **Table** pane displays a summary of candidate data.
- When you select **Candidate Points**, the **Table** pane displays existing candidate points and allows you to add new custom candidate points. The **Chart** pane displays results graphically. For more information, see [Using Candidate Point Results \(p. 228\)](#).

Once candidate points are created, you can verify them and also have the option of inserting them into the response surface as other types of points:

[Viewing and Editing Candidate Points in the Table Pane](#)

[Retrieving Intermediate Candidate Points](#)

[Inserting Candidate Points as New Design, Response, Refinement, or Verification Points](#)

[Verifying Candidates by Design Point Update](#)

Viewing and Editing Candidate Points in the Table Pane

When **Candidate Points** is selected in the **Outline** pane, the **Table** pane displays data for the candidate points that the optimization has generated.

The maximum number of candidate points that can be generated is determined by **Maximum Number of Candidates**. The recommended maximum number of candidate points depends on the optimization method selected for use. For example, only one candidate is generally needed for gradient-based, single-objective methods (NLPQL and MISQP). For multiple-objective methods, you can request as many candidates as you want. For each Pareto front that is generated, there are several potential candidates.

Note:

Because the number of candidate points does not affect the optimization, you can experiment by changing the value for **Maximum Number of Candidates** and then updating the optimization. Providing that only this property changes, the update performs only postprocessing operations, which means candidates are rapidly generated.

The **Table** pane displays each candidate point, along with its input and output values. Output parameter values calculated from design point updates display in black text. Output parameter values calculated from a response surface are displayed in the custom color defined in **Tools** → **Options** → **Design Exploration** → **Response Surface**. For more information, see [Response Surface Options \(p. 38\)](#). The number of gold stars or red crosses displayed next to each goal-driven parameter indicates how well the parameter meets the stated goal. Parameters with three gold stars are the best, and parameters with three red crosses are the worst.

For each parameter with a goal defined, the optimization also calculates the percentage of variation for all parameters with regard to an initial reference point. By default, the initial reference point for NLPQL or MISQP is the **Starting Point** defined in the optimization properties. For Screening or MOGA, the initial reference point is the most viable candidate, **Candidate 1**. You can set any candidate point

as the initial reference point by selecting it in the **Reference** column. The **Parameter Value** column displays the parameter value and the stars or crosses indicating the quality of the candidate. In the **Variation from Reference** column, green text indicates variation in the expected direction. Red text indicates variation that is not in the expected direction. When there is no obvious direction (as for a constraint), the percentage value displays in black text.

The **Name** property for each candidate point indicates whether it corresponds to a design point in the **Table** pane for the **Parameter Set** bar. A candidate point corresponds to a design point when they both have the same input parameter values. If the design point is deleted from the **Parameter Set** bar or the definition of either point is changed, the link between the two points is broken, without invalidating your model or results. Additionally, the indicator is removed from the candidate point's name.

If parameter relationships are defined and enabled, you can opt to also view parameter relationships in the candidate points table. In the **Properties** pane for the **Optimization** cell, select the **Show Parameter Relationships** check box. In the **Table** pane, the candidate points table then displays parameter relationships and their expressions. For NLPQL and MISQP, the candidate points table also displays current values for each expression.

	A	B	C	D	E	F	G	H	I	H	I	J	K
1	Reference	Name	P1...	P2 - X_2	P3 - X_3	P4 - X_4	P5 - X_5	P2-P1 <= 3[mm]		P6 - f		P7 - g	
2								P2-P1	3[mm]	Parameter Value	Variation from Reference	Parameter Value	Variation from Reference
3		Starting Point						0 [mm]	3 [mm]	☆☆ -9043.7	9.97 %	☆☆ 166.32	3,235.91 %
4		Starting Point (verified) (DP 150)	11.51	77.866	3	-94	7	0 [mm]	3 [mm]	☆☆ -8237.3	18.00 %	☆☆ 166.32	3,235.91 %
5		Candidate Point 1						1.043 [mm]	3 [mm]	☆☆ -10045	0.00 %	☆☆ 6.0882	22.11 %
6		Candidate Point 1 (verified) (DP 151)	1.3168	94.64%	1	-100	-1	1.043 [mm]	3 [mm]	☆☆ -10094	-0.49 %	☆☆ 6.0882	22.11 %

Creating Custom Candidate Points

You can create custom candidate points in the candidate points table. Perhaps you want to add candidate points that represent the existing design of a product, the initial design of the parametric study, or other points of interest. You can add a custom candidate point using any of the following methods:

- In the **Outline** pane under **Results**, select **Candidate Points**. In the **Table** pane, enter data into the cells of the bottom table row. For a **Response Surface Optimization** system, you can also right-click a candidate point row in the **Table** pane and select **Insert as Custom Candidate Point**.
- In the **Outline** pane, select **Optimization**. For a **Response Surface Optimization** system, you can also right-click a candidate point in the **Table** pane and select **Insert as Custom Candidate Point**.
- In the **Outline** pane under **Results**, select any chart. Right-click a point in the chart and select **Insert as Custom Candidate Point**.

When a custom candidate point is created in a **Response Surface Optimization** system, the outputs of custom candidates are automatically evaluated from the response surface. When a custom candidate is created in a **Direct Optimization** system, the outputs of the custom candidates are not brought up-to-date until the next real solve.

Once a candidate point is created, it is automatically plotted in the results in the **Chart** pane and can be treated as any other candidate point. You have the ability to edit the name, edit input parameter values, and select options from the right-click context menu. In addition to the standard context menu options, an **Update Custom Candidate Point** option is available for out-of-date candidates in a **Direct Optimization** system. Additionally, a **Delete** option allows you to delete a custom candidate point.

Retrieving Intermediate Candidate Points

DesignXplorer allows you to monitor an optimization while it is still running by watching the progress from the History chart. If the History chart indicates that candidate points meeting your criteria have been found midway through the optimization, you can stop the optimization and retrieve these results without having to run the rest of the optimization.

To stop the optimization, click **Show Progress** in the lower right corner of the window to open the **Progress** pane. To the right of the progress bar, click the red stop button. In the dialog box that opens, select either **Interrupt** or **Abort**. Intermediate results are available in either case.

When the optimization is stopped, candidate points are generated from the data available at this time, such as solved samples, results of the current iteration, the current populations, and so on.

To assess the state of the optimization at the time it was stopped, under **Optimization** in the **Properties** pane, look at the optimization status and counts.

To view the intermediate candidate points, under **Results**, select **Candidate Points**.

Note:

DesignXplorer might not be able to return verified candidate points for optimizations that have been stopped.

When an optimization is stopped midway, the **Optimization** cell remains in an unsolved state. If you change any settings before updating the optimization again, the optimization process must start over. However, if you do not change any settings before the next update, DesignXplorer makes use of the design point cache to quickly return to the current iteration.

For more information, see [Using the History Chart \(p. 219\)](#).

Inserting Candidate Points as New Design, Response, Refinement, or Verification Points

Once objectives are stated and candidate points are generated, the **Table** pane for the **Optimization** cell displays candidate points. You can insert candidate points as new design, response, refinement, or verification points.

When either **Optimization** or **Candidate Points** is selected in the **Outline** pane, you can select one or more candidate points in the **Table** pane and then right-click one of them to select an option for inserting them as new points. You can also right-click a point in an optimization chart. The options available on the context menu depend on the type of optimization. Possible options are:

- **Explore Response Surface at Point.** Inserts new response points in the **Table** pane for the **Response Surface** cell by copying the input and output parameter values of the selected candidate points.
- **Insert as Design Point.** Inserts new design points in the **Table** pane for the **Parameter Set** bar by copying the input parameter values of the selected candidate points. The output parameter values are not copied because they are approximated values provided by the response surface.
- **Insert as Refinement Point.** Inserts new refinement points in the **Table** pane for the **Response Surface** cell by copying the input parameter values of the selected candidate points.
- **Insert as Verification Point.** Inserts new verification points in the **Table** pane for the **Response Surface** cell by copying the input and output parameter values of the selected candidate points.
- **Insert as Custom Candidate Point.** Inserts new custom candidate points in the candidate points table by copying the input parameter values of the selected candidate points.

For a **Response Surface Optimization** system, the same insertion operations are available for the raw optimization data table and most optimization charts, depending on the context. For instance, it is possible to right-click a point in a Tradeoff chart to insert the corresponding sample as a response point, refinement point, or design point. The same operations are also available from a Samples chart.

Note:

For a **Direct Optimization** system, only the **Insert as Design Point** and **Insert as Custom Candidate Point** options are available.

Verifying Candidates by Design Point Update

For a **Response Surface Optimization** system, candidate points are verified automatically at the end of the optimization update if the **Verify Candidate Points** check box is selected in the **Properties** pane for the **Optimization** cell. You can also verify candidate points after they are created.

With either **Optimization** or **Candidate Points** is selected in the **Outline** pane, select one or more candidate points in the **Table** pane and then right-click one of them and select **Verify by Design Points Update**. This context menu option is available for both optimization-generated candidate points and custom candidate points.

DesignXplorer verifies candidate points by creating and updating design points with a real solve, using the input parameter values of the candidate points. The output parameter values for each candidate point are displayed in a separate row. For a **Response Surface Optimization** system, verified candidates are placed next to the row containing the output values generated by the response surface. The sequence varies according to sort order. Output parameter values calculated from design point updates are displayed in black text. Output parameter values calculated from a response surface are displayed in the custom color defined in **Tools** → **Options** → **Design Exploration** → **Response Surface**. For more information, see [Response Surface Options \(p. 38\)](#).

Table of Schematic B2: Optimization , Candidate Points													
	A	B	C	D	E	F	G	H	I	H	I	J	K
1	Reference	Name	P1...	P2 - x_2	P3 - x_3	P4 - x_4	P5 - x_5	P2-P1 <= 3[mm]		P6 - f		P7 - g	
2								P2-P1	3[mm]	Parameter Value	Variation from Reference	Parameter Value	Variation from Reference
3		Starting Point	11.51	77.866	3	-94	7	0 [mm]	3 [mm]	☆☆☆ -9043.7	9.97 %	☆☆☆ 166.32	3,235.91 %
4		Starting Point (verified) (DP 150)						0 [mm]	3 [mm]	☆☆☆ -8237.3	18.00 %	☆☆☆ 166.32	3,235.91 %
5		Candidate Point 1	1.3168	94.64%	1	-100	-1	1.043 [mm]	3 [mm]	☆☆☆ -10045	0.00 %	☆☆☆ 6.0882	22.11 %
6		Candidate Point 1 (verified) (DP 151)						1.043 [mm]	3 [mm]	☆☆☆ -10094	-0.49 %	☆☆☆ 6.0882	22.11 %

In a **Response Surface Optimization** system, if a large difference exists between the results of the verified and unverified rows for a point, the response surface might not be accurate enough in that area. In such cases, perhaps refinement or other adjustments are necessary. If desired, you can insert the candidate point as a refinement point. You then recompute the optimization so that the refinement point and new response surface are taken into account.

Note:

- This option is convenient when updating a design point is quick.
- Often candidate points do not have practical input parameters. For example, ideal thickness could be 0.127 instead of the more practical 0.125. If desired, you can right-click the candidate and select **Insert as Design Point**, edit the parameters of the design point, and then run this design point instead of the candidate point.

To solve the verification points, DesignXplorer uses the same mechanism that is used to solve DOE points. The verification points are either deleted or persisted after the run as determined by the **Preserve Design Points after a DX Run** option for DesignXplorer. As usual, if the update of the verification point fails, it is preserved automatically in the project. You can explore it as a design point by editing the **Parameter Set** bar in the **Project Schematic**.

Goal-Driven Optimization Charts and Results

After an **Optimization** cell is solved, the following results or charts are created by default in the **Outline** pane under **Results**: Candidate Points, Sensitivities, Tradeoff, and Samples.

After the first solve of the **Optimization** cell:

- The Convergence Criteria chart displays by default in the **Chart** pane.
- The History chart is available for the following objects in the **Outline** pane:
 - Objectives and constraints
 - Enabled input parameters
 - Enabled parameter relationships

With the exception of the Candidate Points chart, the Convergence Criteria chart, and the History chart, it is possible to duplicate charts. Right-click the chart in the **Outline** pane and select **Duplicate**. Or, use

the drag-and-drop operation, which attempts an update of the chart so that the duplication of an up-to-date chart results in the creation of an up-to-date chart.

Using the Convergence Criteria Chart

Once the optimization has been updated for the first time, the Convergence Criteria chart allows you to view the evolution of the convergence criteria for the selected iterative optimization method.

Note:

- The Convergence Criteria chart is not available for the Screening method.
- When an external optimizer is used, the Convergence Criteria chart is generated if data is available.

When **Optimization** or **Convergence Criteria** is selected in the **Outline** pane, the **Chart** pane displays the Convergence Criteria chart. The rendering and logic of the chart varies according to whether you are using a multiple-objective or single-objective optimization method.

Because the chart is updated after each iteration, you can use it to monitor the progress of the optimization. When the convergence criteria have been met, the optimization stops and the chart remains available.

Using the Convergence Criteria Chart for Multiple-Objective Optimization

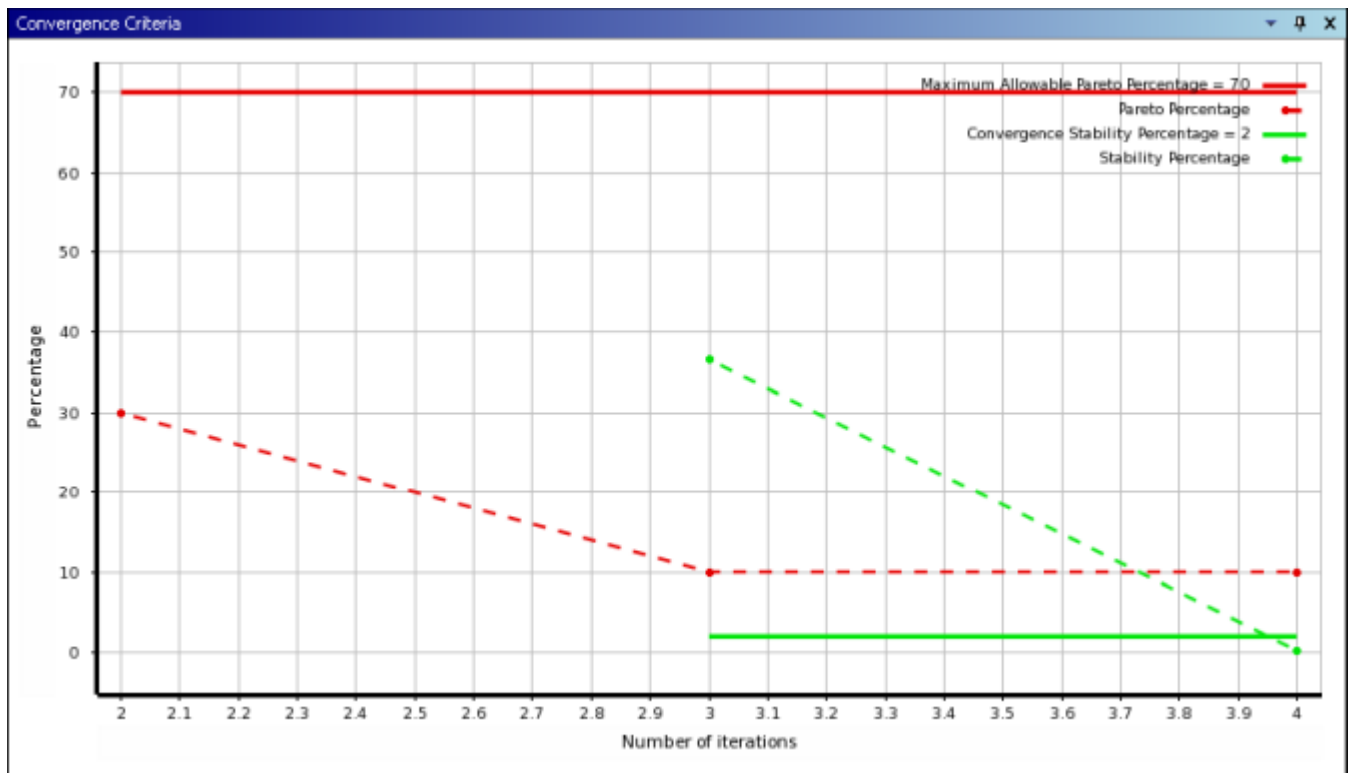
When a multiple-objective optimization method such as MOGA or Adaptive Multiple-Objective is being used, the Convergence Criteria chart plots the evolution of the **Maximum Allowable Pareto Percentage** and **Convergence Stability Percentage** convergence criteria. For more information, see [Convergence Criteria in MOGA-Based Multi-Objective Optimization \(p. 373\)](#).

Note:

If you are using an external optimizer that supports multiple objectives, the Convergence Criteria chart displays the data that is available.

Before running your optimization, you specify values for the convergence criteria. After selecting **Optimization** in the **Outline** pane, edit the values under **Optimization** in the **Properties** pane.

An example follows of a Convergence Criteria chart for a multiple-objective optimization.



In this Convergence Criteria chart, you see:

- Number of iterations along the X axis
- Convergence criteria percentage along the Y axis
- A solid red line representing the maximum allowable Pareto percentage
- A dashed red line representing the evolution of the Pareto percentage
- A solid green line presenting the convergence stability percentage
- A dashed green line representing the stability percentage
- For MOGA, a legend displaying convergence criteria values

You can enable or disable the convergence criteria that display on the Convergence Criteria chart by. Select **Convergence Criteria** in the **Outline** pane. In the **Properties** pane under **Criteria**, select or clear the **Enabled** check box for a criterion to enable or disable it.

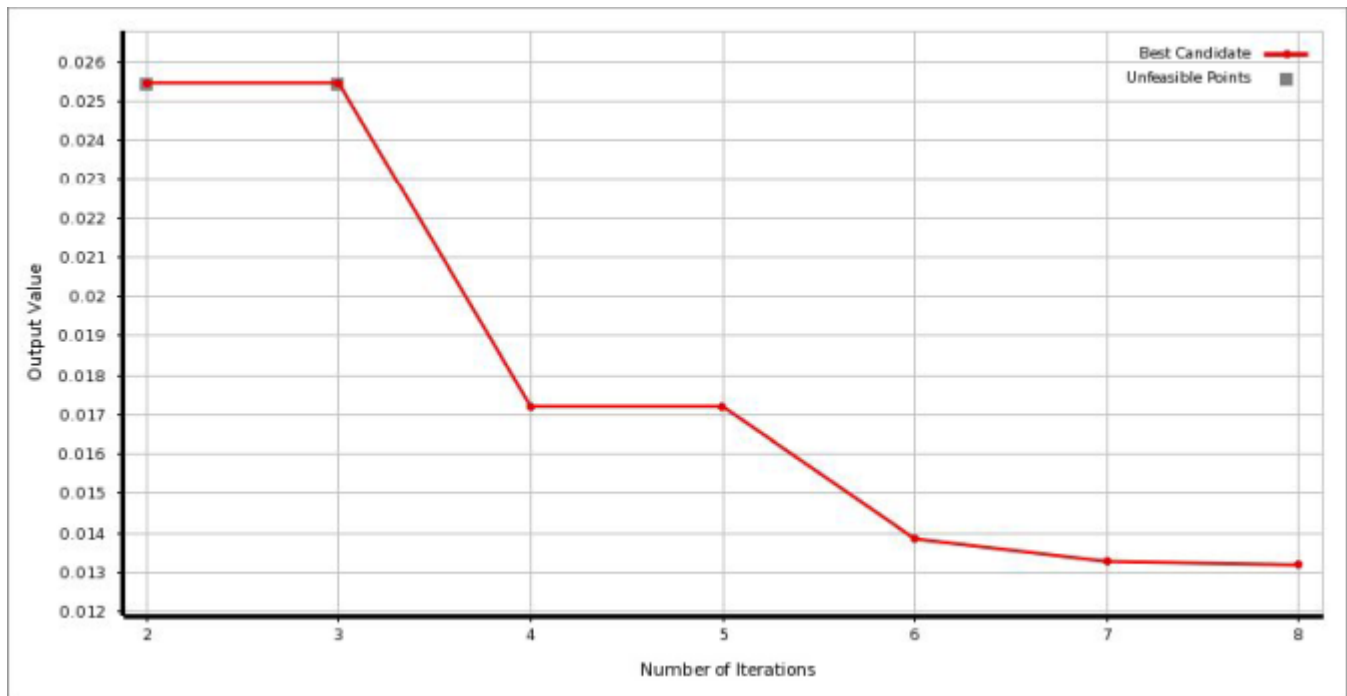
Using the Convergence Criteria Chart for Single-Objective Optimization

When a single-objective optimization method such as NLPQL, MISQP, or Adaptive Single-Objective is being used, the Convergence Criteria chart plots the evolution of the best candidate point for the optimization by identifying the best point for each iteration.

Before running your optimization, you specify values for the convergence criteria relevant to your selected optimization method. Although these criteria are not explicitly shown on the chart, they affect the optimization and the selection of the best candidate.

To specify convergence criteria values, select **Optimization** in the **Outline** pane. Then, edit the values under **Optimization** in the **Properties** pane.

An example follows of a Convergence Criteria chart for a single-objective optimization.



In this Convergence Criteria chart, you see:

- Number of iterations along the X axis
- Output values along the Y axis
- A solid red line representing the evolution of the best candidate
- Red points representing the best candidates that are feasible points
- Gray squares representing best candidates that are infeasible points

Using the History Chart

Once the optimization is updated for the first time, the History chart allows you to view the optimization history of an enabled objective, constraint, input parameter, or parameter relationship.

Additionally, it gives you the option of monitoring the progress of the selected object while the optimization is still in progress. If you select an object during an update, the chart refreshes automatically and shows the evolution of the objective, constraint, input parameter, or parameter relationship throughout the update.

For the iterative optimization methods, the chart is refreshed after each iteration. For the Screening method, it is updated only when the optimization update is complete. You can select a different object at any time during the update to plot and view a different chart.

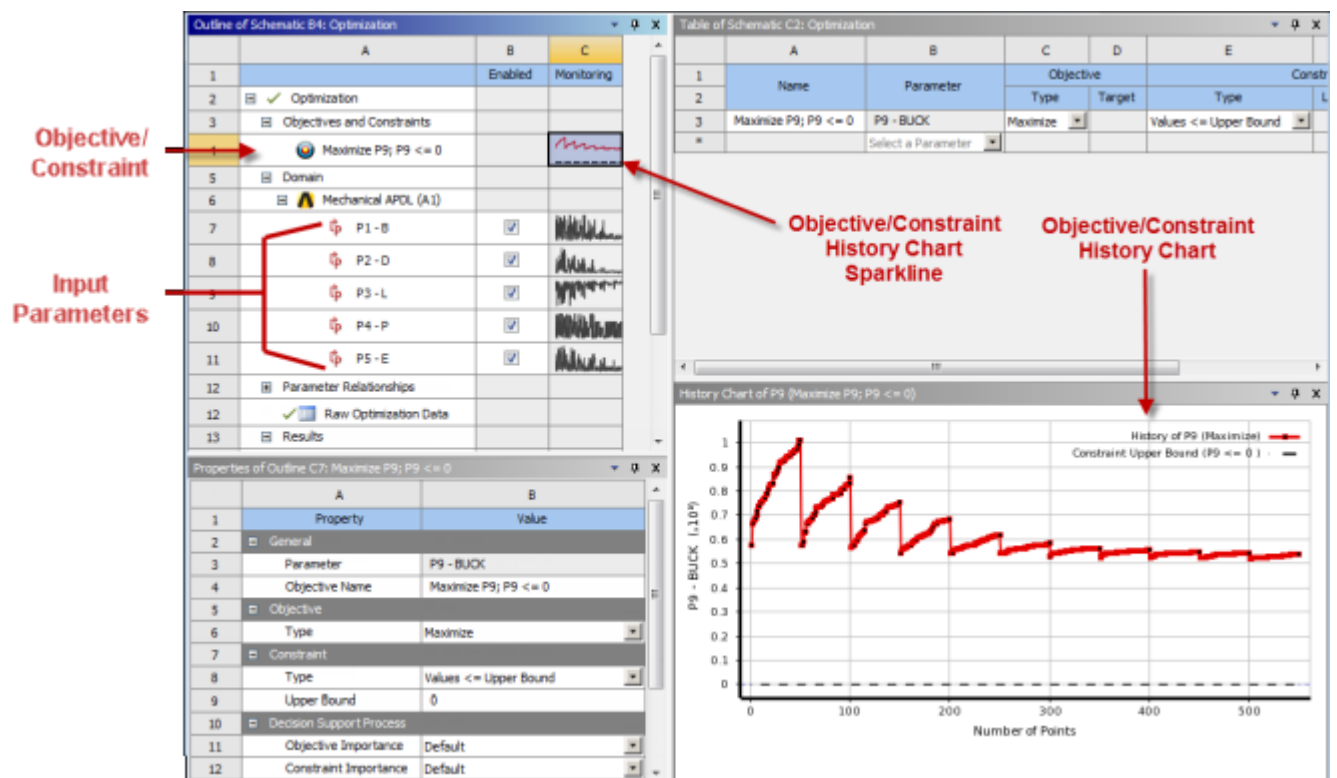
The History charts remain available when the update is completed. In the **Outline** pane, a sparkline version of the History chart is displayed for each objective, constraint, input parameter, or parameter relationship.

If the History chart indicates that the optimization has converged midway through the process, you can stop the optimization and retrieve the results without having to run the rest of the optimization. For more information, see [Retrieving Intermediate Candidate Points](#) (p. 214).

Note:

The History chart does not display failed design points.

You can access the History chart by selecting an objective or a constraint under **Objectives and Constraints** in the **Outline** pane. Or, you select an input parameter or parameter relationship under **Domain**.



Working with the History Chart in the Chart Pane

The rendering of the History chart varies, not only according to whether an objective, constraint, input parameter, or parameter relationship is being charted but also according to the optimization method being used. In the History chart, you can see:

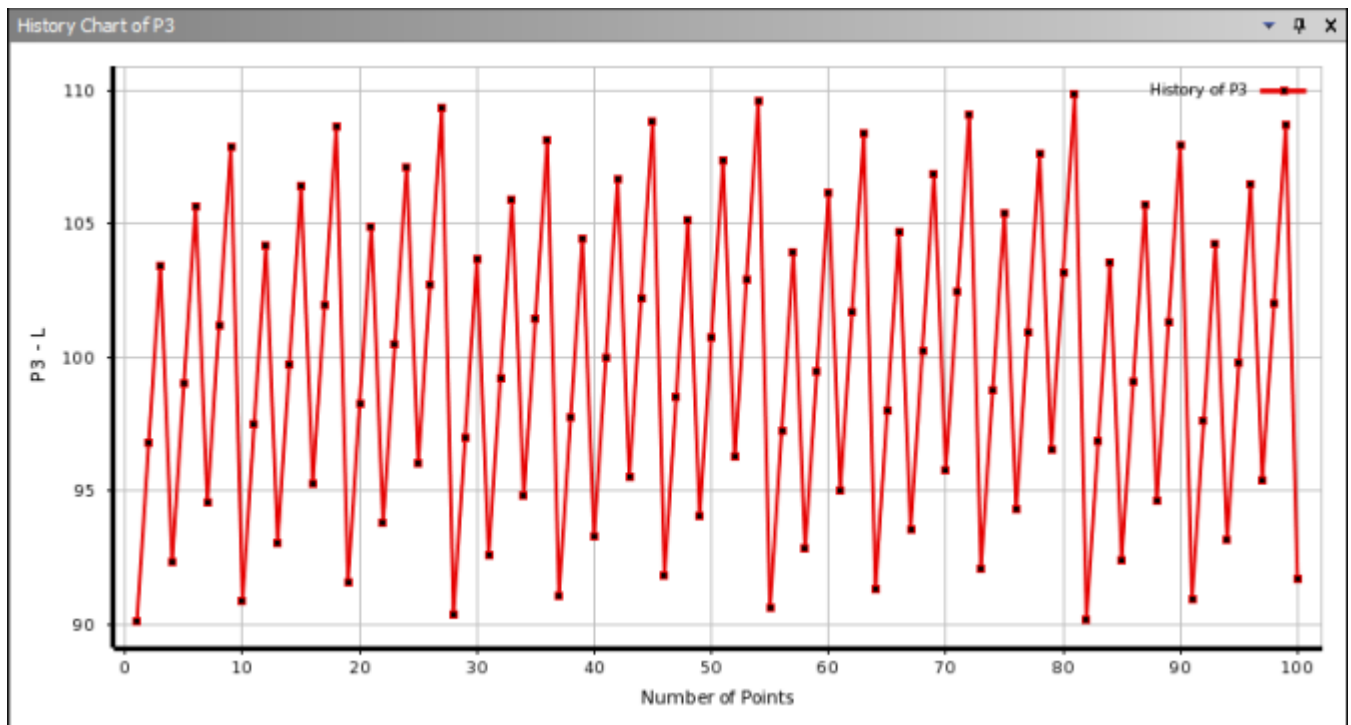
- Number of points in the sample set (as defined by the **Size of Generated Sample Set** property) along the X axis
- Objective values, which fall within the optimization domain defined for the associated parameter, along the Y axis

- A red line representing evolution of the object
- Gray dashed lines representing bounds for constraints
- Blue dashed lines presenting target values

You can place the mouse cursor over any data point in the chart to view the X and Y coordinates.

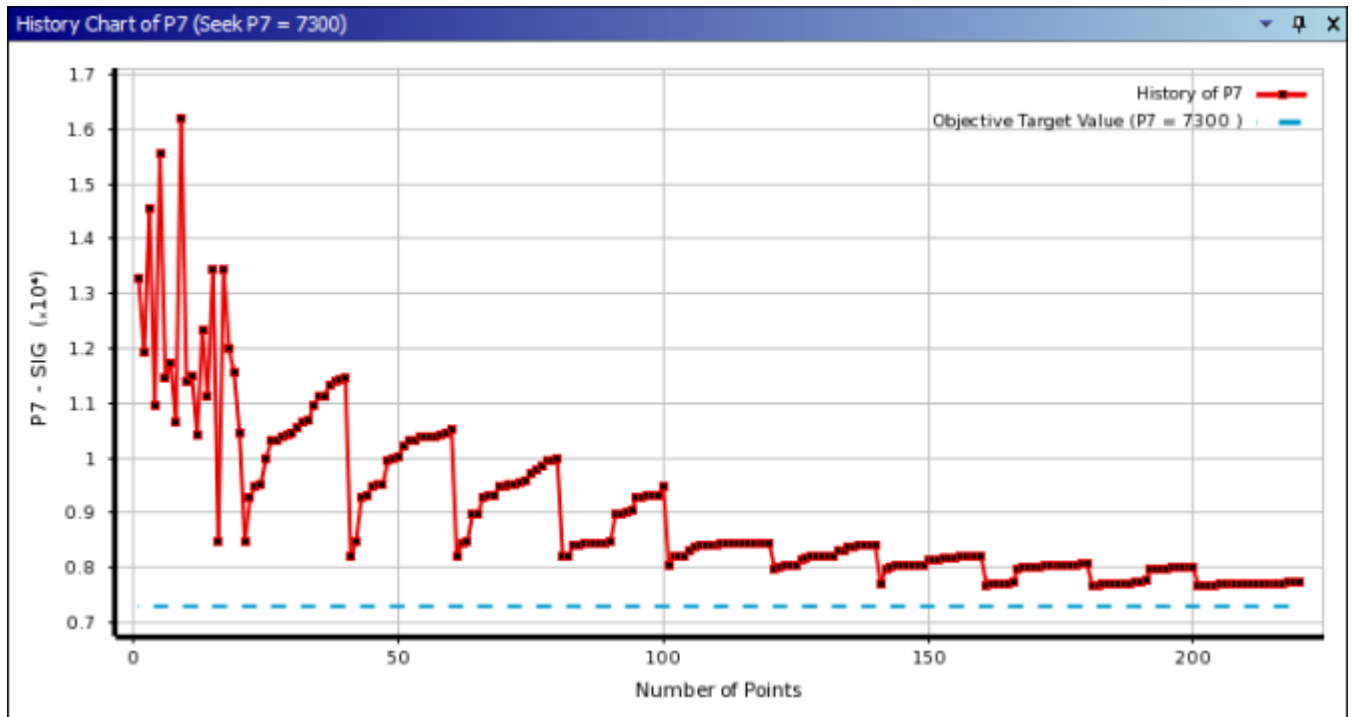
Screening

For a Screening optimization, which is non-iterative, the History chart displays all the points of the sample set. The chart is updated when all points have been evaluated. The plot reflects the non-iterative process, with each point visible on the chart.



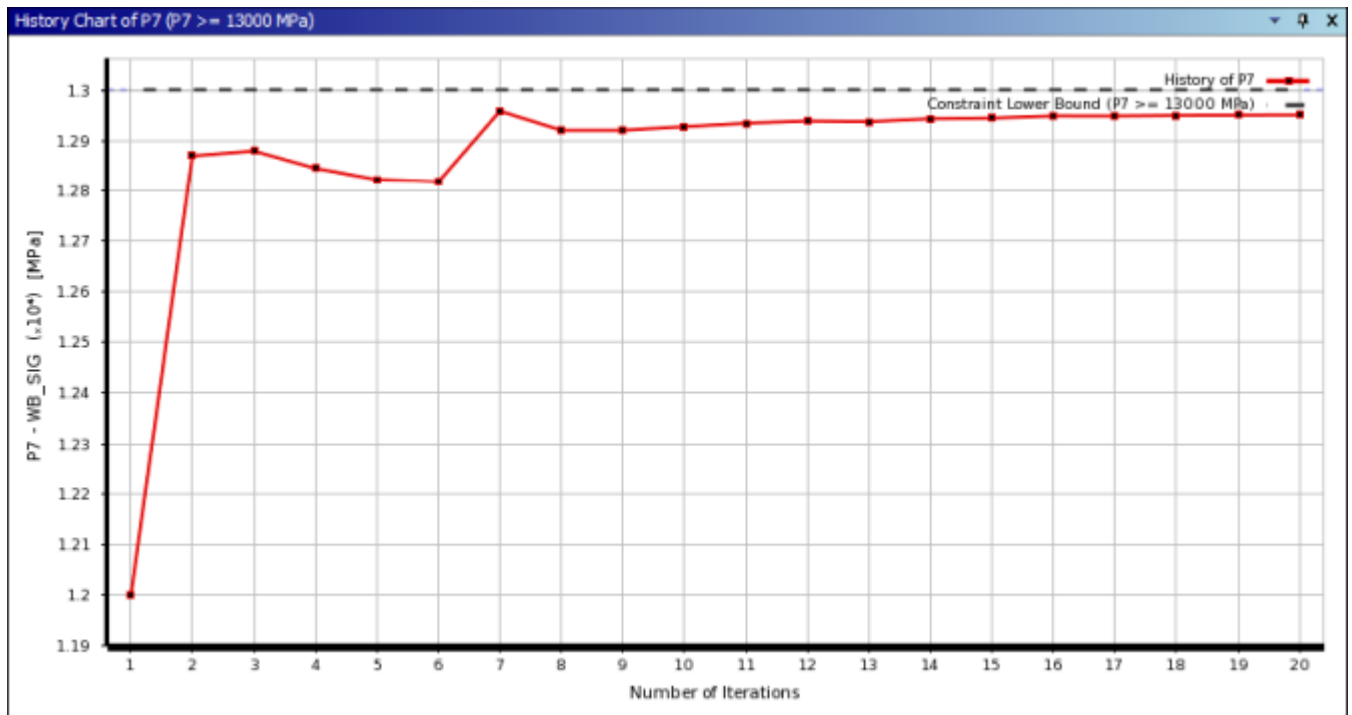
MOGA

For a MOGA optimization, the History chart displays the evolution of the population of points throughout the iterations in the optimization. The chart is updated at the end of each iteration with the most recent population (as defined by the **Number of Samples per Iteration** property).



NLPQL and MISQP

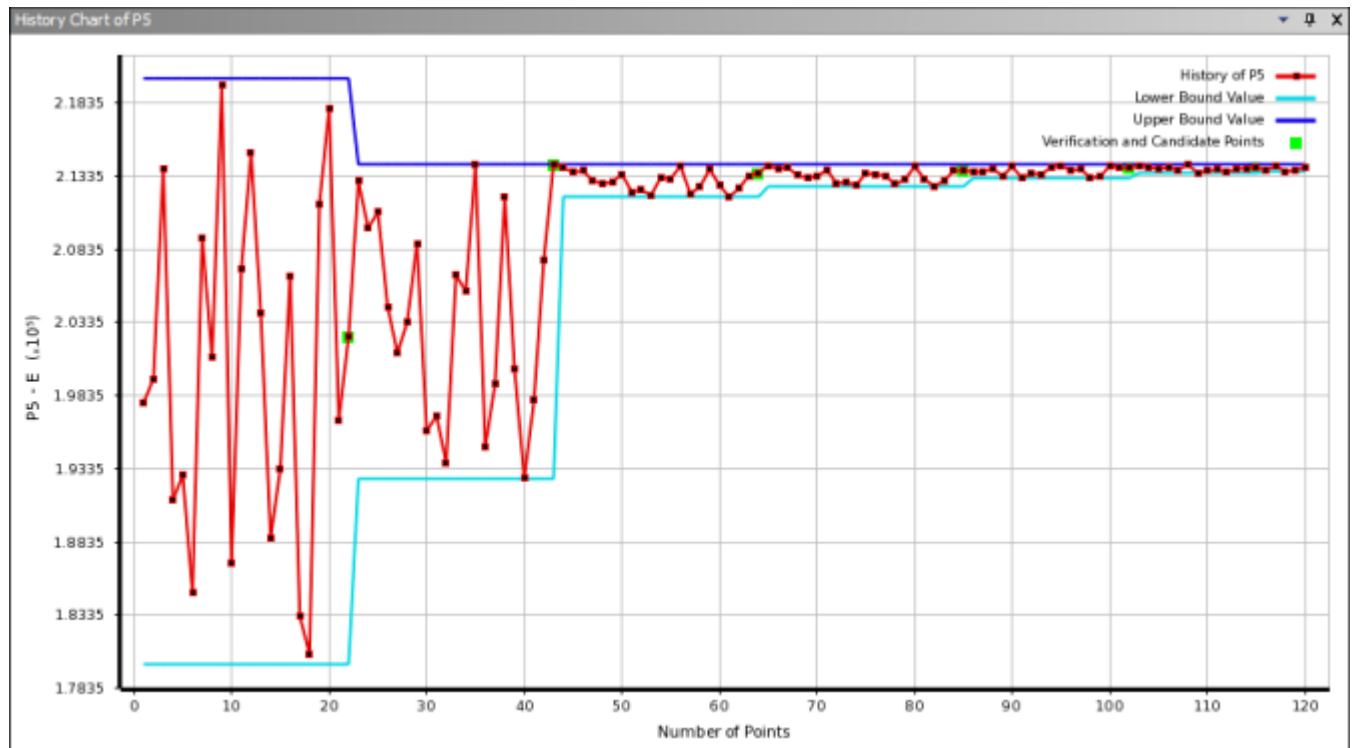
For an NLPQL or MISQP optimization, the History chart enables you to trace the progress of the optimization from a defined starting point. The chart displays the objective value associated with the point used for each iteration. The chart does not display the points used to evaluate the derivative values. It reflects the gradient optimization process, displaying a point for each iteration.



Adaptive Single-Objective

For an Adaptive Single-Objective optimization, the History chart enables you to trace the progress of the optimization through a specified maximum number of evaluations. On the Input Parameter History chart, the upper and lower bounds of the input parameter are represented by blue lines, allowing you to see the domain reductions narrowing toward convergence.

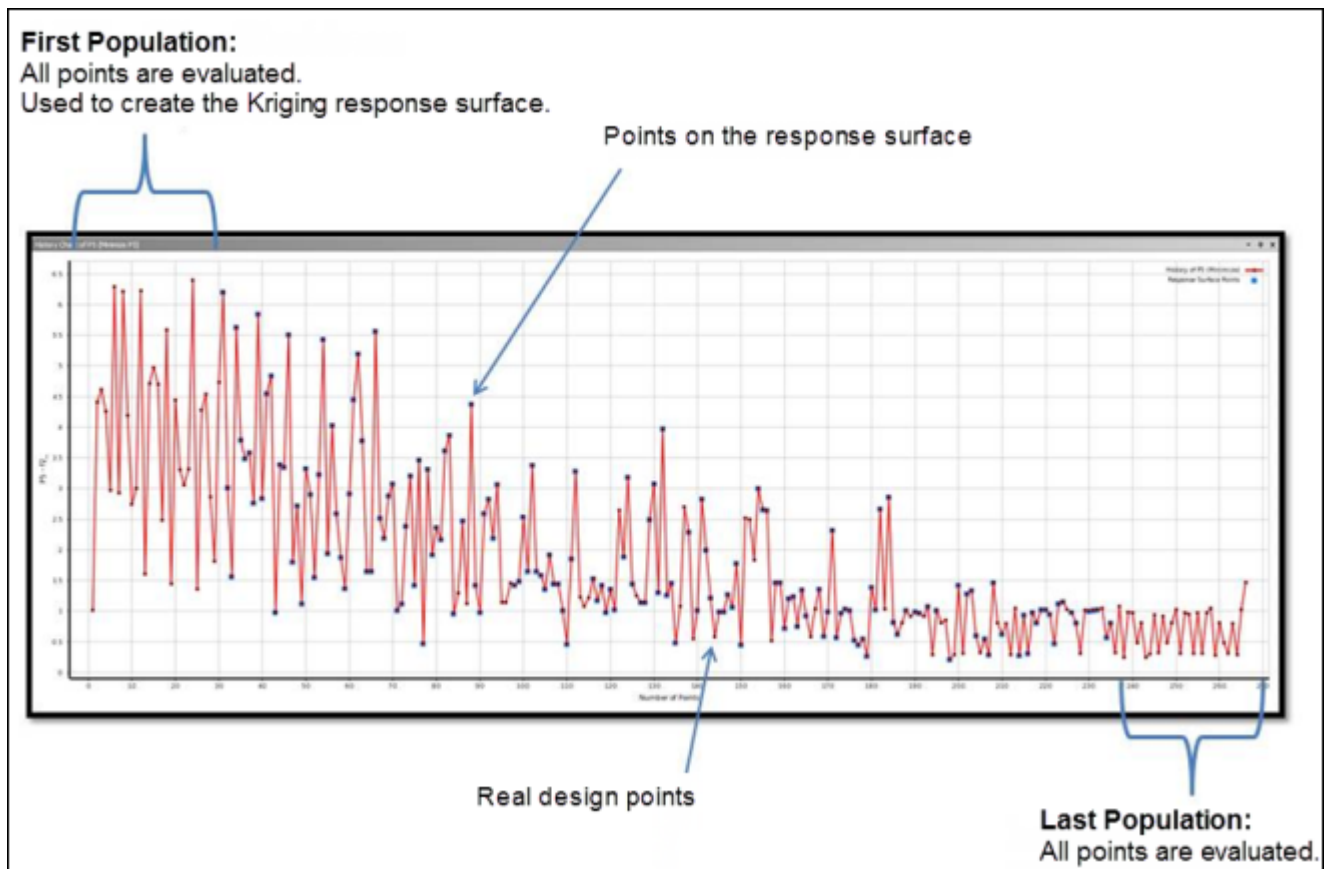
The chart displays the objective value corresponding to LHS or verification points, showing all evaluated points.



Adaptive Multiple-Objective

For an Adaptive Multiple-Objective optimization, the History chart displays the evolution of the population of points throughout the iterations in the optimization. Each set of points (the number of which is defined by the **Number of Samples Per Iteration** property) corresponds to the population used to generate the next population. Points corresponding to real solve are plotted as black points. Points from the response surface are plotted with a square colored as specified in **Tools** → **Options** → **Design Exploration** → **Response Surface**. For more information, see [Response Surface Options \(p. 38\)](#).

The plot reflects the iterative optimization process, with each iteration visible on the chart. All candidate points generated by the optimization are real design points.



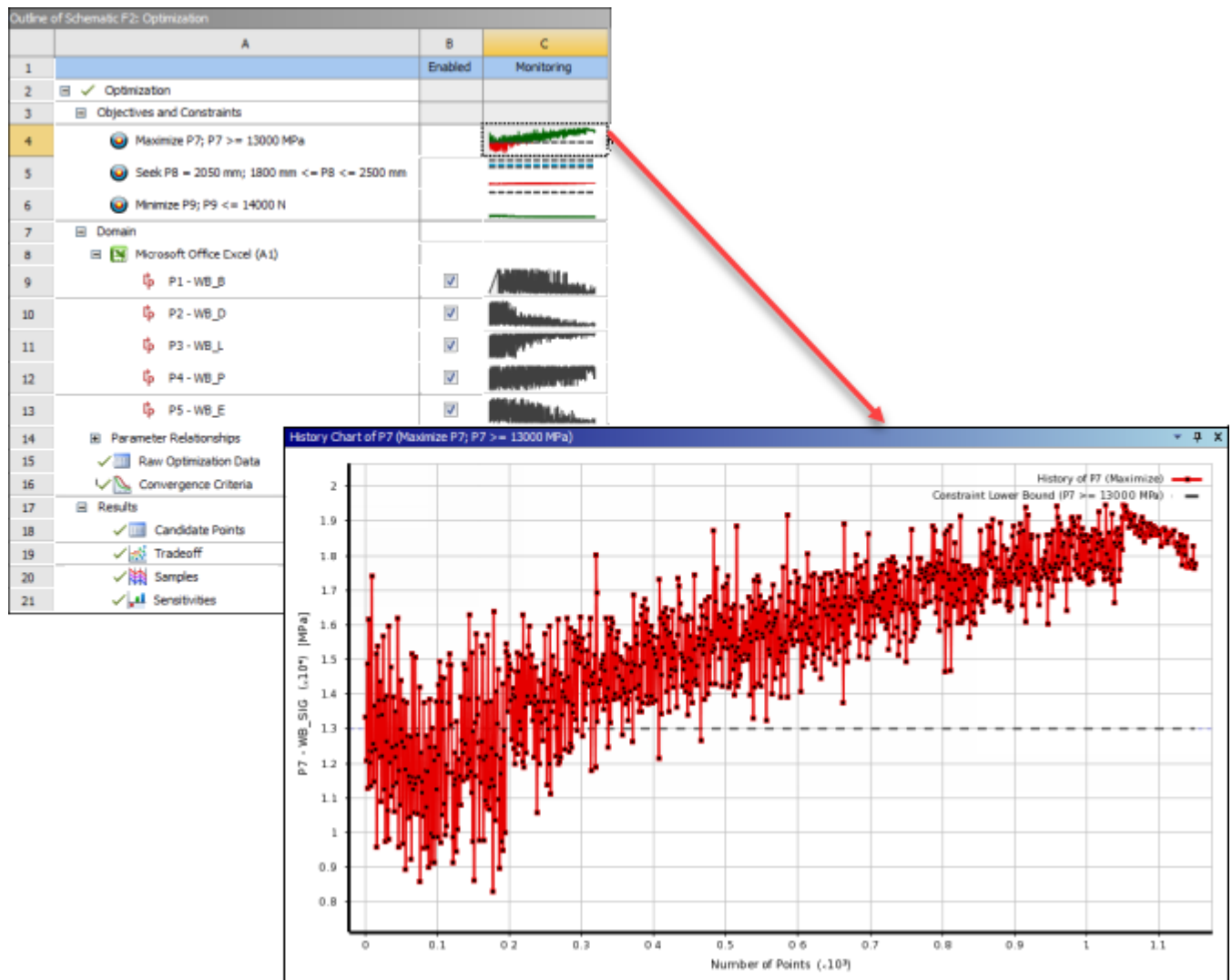
Viewing History Chart Sparklines in the Outline Pane

When a History chart for an objective, constraint, input parameter, or parameter relationship is generated, the same data is used to generate a sparkline image of the History chart. The sparklines for defined objects are displayed in the **Outline** pane and are refreshed dynamically during the update, allowing you to follow the update progress for all objects simultaneously.

The History chart sparkline is similar to the History chart in the **Chart** pane:

- Sparklines are gray if no constraints are present. However, if constraints are present:
 - Sparklines are green when the constraint or parameter relationship is met
 - Sparklines are red when the constraint or parameter relationship is not met. When parameter relationships are enabled and taken into account, the optimization should not pick unfeasible points.
- Blue dashed lines represent targets.
- Gray dashed lines represent bounds for constraints.

An example follows of a History chart sparkline.



In this History chart sparkline, you see:

- In the **Outline** pane, the sparkline for **Minimize P9; P9 <= 14000 N** is entirely green, indicating that the constraints are met throughout the optimization history.
- In the **Outline** pane, the sparkline for **Maximize P7; P7 >= 13000** is both red and green, indicating that the constraints are violated at some points and met at others.
- In the **Charts** pane, the History chart is shown for the constraint **Maximize P7; P7 >= 13000**. The points beneath the dotted gray line for the lower bound are infeasible points.

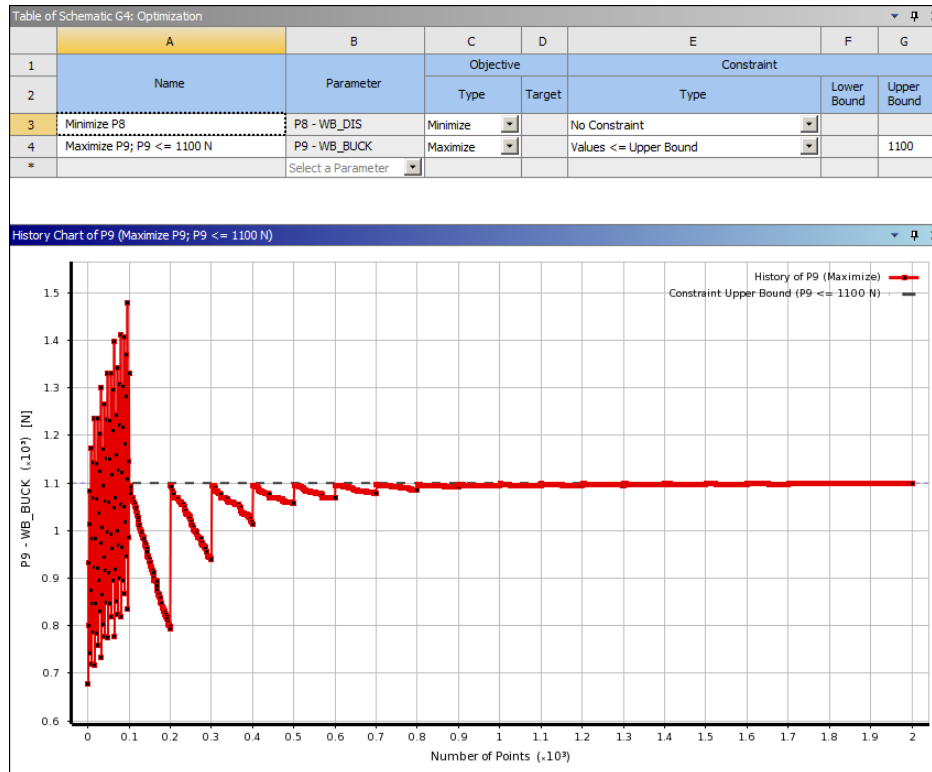
Using the History Chart for an Objective or Constraint

To generate the History chart for an enabled objective or constraint, update the **Optimization** cell. Then, in the **Outline** pane under **Objectives and Constraints**, select the objective or constraint. If you change your selection during the update, the chart refreshes automatically, allowing you to monitor the update process.

The History chart for an objective or constraint displays a red line to represent the evolution of the parameter for which an objective or constraint has been defined. Constraints are represented by gray dashed lines. The target value is represented by a blue dashed line.

In the following History chart for a MOGA optimization, the output parameter **P9 – WB_BUCK** is plotted. The parameter is constrained such that it must have a value lesser than or equal to **1100**. The dotted gray line represents the constraint.

Given that **Constraint Type** is set to **Maximize** and **Upper Bound** is set to **1100**, the area under the dotted gray line represents the infeasible domain.

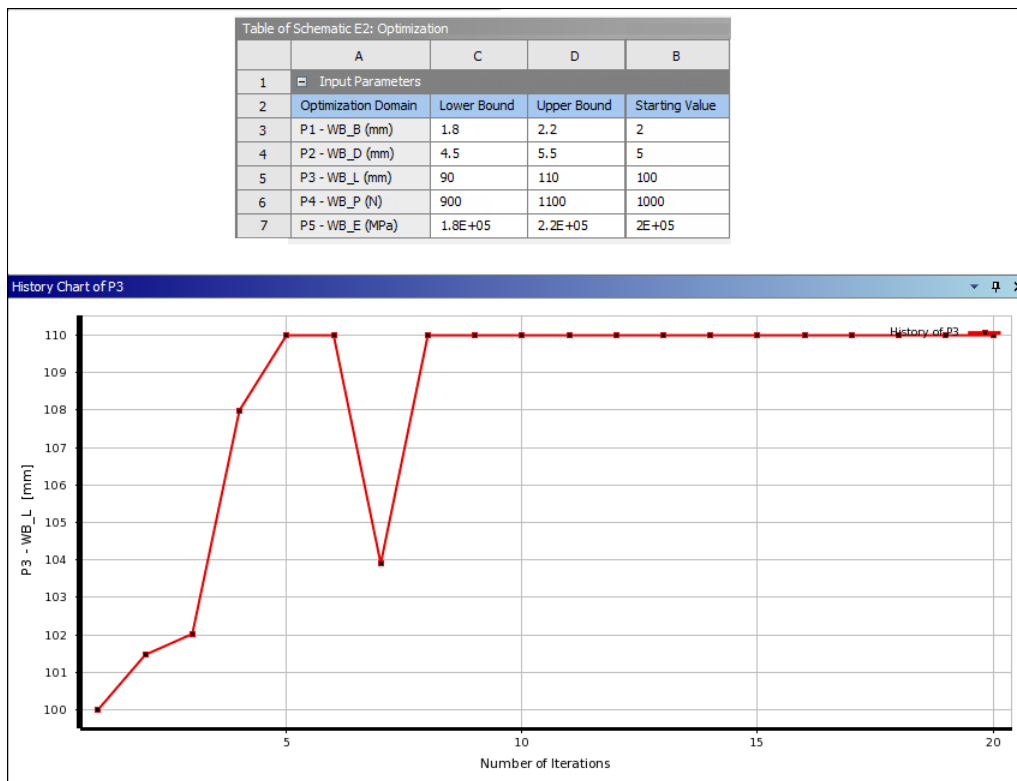


Using the Input Parameter History Chart

To generate the History chart for an enabled input parameter, update the **Optimization** cell. Then, in the **Outline** pane under **Domain**, select the input parameter. If you change your selection during the update, the chart refreshes automatically, allowing you to monitor the update process.

For an input parameter, the History chart displays a red line to represent the evolution of the parameter for which the objective has been defined. If an objective or constraint is defined for the parameter, the same chart displays when the objective, constraint, or input parameter is selected.

In the following History chart for an NLPQL optimization, the input parameter **P3 – WB_L** is plotted. For **P3 – WB_L**, **Starting Value** is set to **100**, **Lower Bound** is set to **90**, and **Upper Bound** is set to **110**. The optimization converged upward to the upper bound for the parameter.

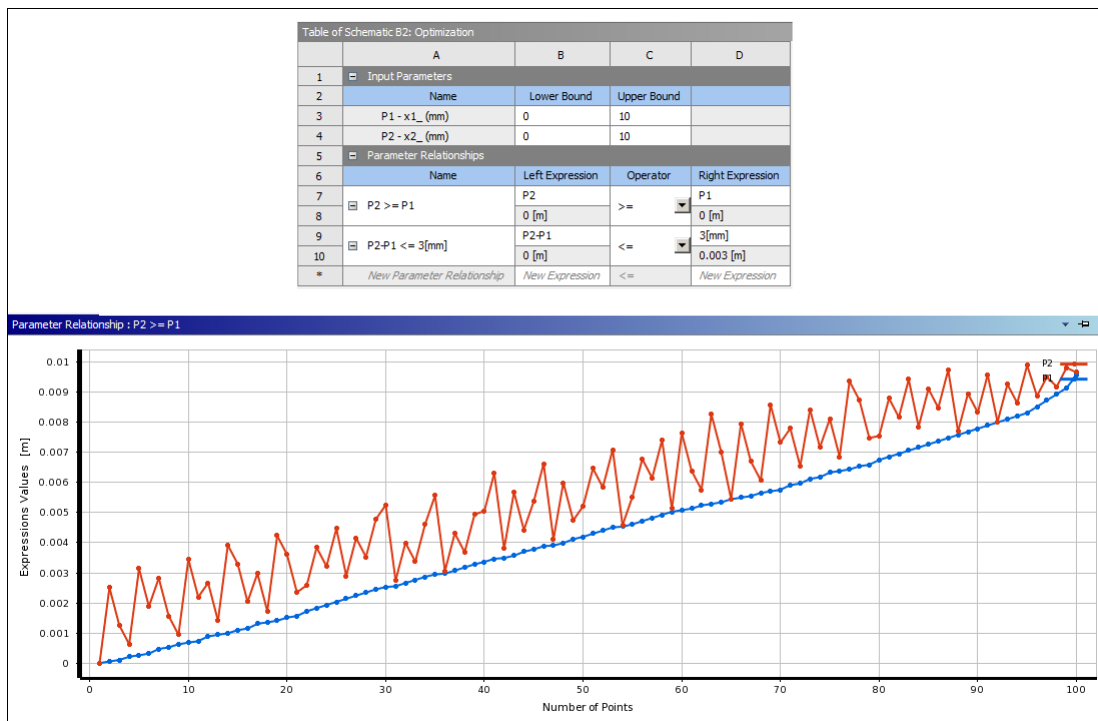


Using the Parameter Relationship History Chart

To generate the History chart for an enabled input parameter relationship, update the **Optimization** cell. Then, in **Outline** pane under **Domain**, select the parameter relationship. If you change your selection during the update, the chart refreshes automatically, allowing you to monitor the update process.

For a parameter relationship, the History chart displays two lines to represent the evolution of the left expression and right expression of the relationship. The number of points are along the X axis. The expression values are along the Y axis.

In the following History chart for a Screening optimization, the parameter relationship **P2 > P1** is plotted.



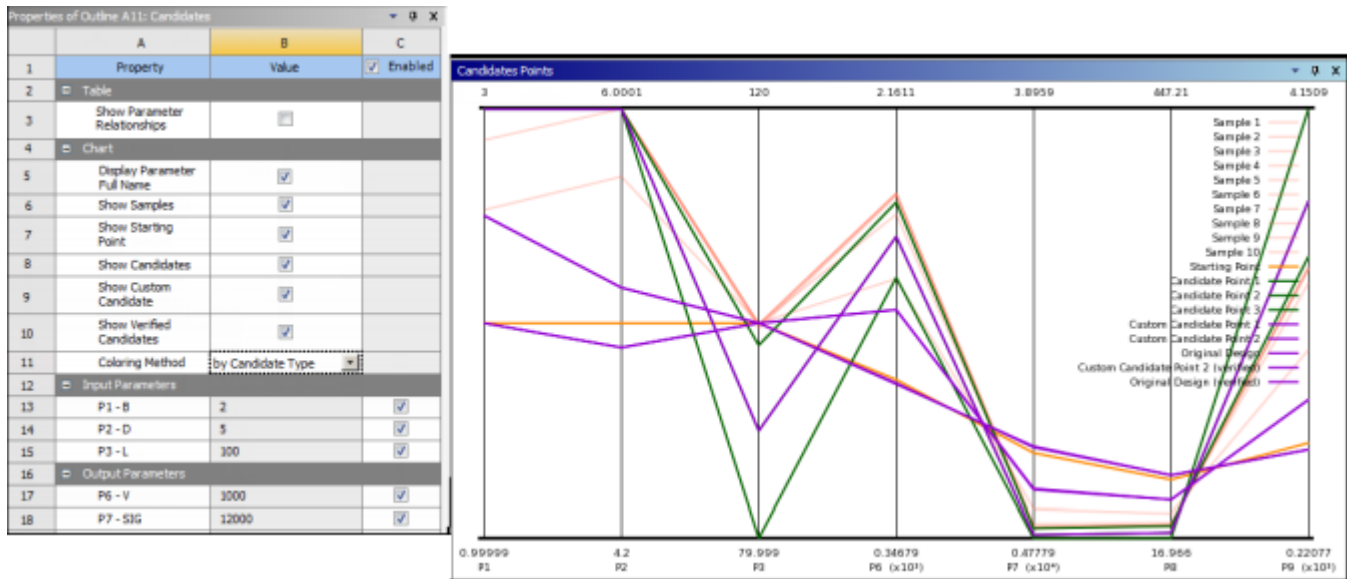
Using Candidate Point Results

Candidate point results are displayed in both the **Table** and **Chart** panes. You can see different kinds of information about candidate points and specify one or more parameters for which you want to display candidate point data. In the **Chart** pane, the legend's color-coding allows you to view and interpret the samples, candidate points identified by the optimization, candidates inserted manually, and candidates for which output values have been verified by a design point update. You can specify the chart's properties to control the visibility of each axis, feasible samples, candidates you've inserted manually, and candidates with verified output values. For information on results in the **Table** pane, see [Viewing and Editing Candidate Points in the Table Pane](#) (p. 212).

To generate candidate points results, update the **Optimization** cell. Then, in the **Outline** pane under **Results**, select **Candidate Points**.

Understanding the Display of Candidate Point Results

The rendering of candidate point results in the **Chart** pane depends on your selections in the **Properties** pane. Once results are generated, you can adjust the properties to change what is displayed. In the following results for an NLPQL optimization, **Coloring Method** is set to **by Candidate Type**.



In this chart, both samples and candidate points are displayed:

- Pale orange lines represent feasible samples.
- The orange line represents the starting point.
- Green lines represent each the candidate points generated by the optimization.
- Purple lines represent each of the custom candidate points.

If you set **Coloring Method** to **by Source Type**, the samples are colored according to the source from which they were calculated, following the color convention used for data in the **Table** pane. Samples calculated from a simulation are represented by black lines. Samples calculated from a response surface are represented by a line in the custom color specified in **Tools** → **Options** → **Design Exploration** → **Response Surface**. For more information, see [Response Surface Options](#) (p. 38).

When you move your mouse over the results, you can pick out individual objects, which become highlighted in orange. When you select a point, the parameter values for the point are displayed in the **Value** column of the **Properties** pane.

Across the bottom of the results, a vertical line displays for each parameter. When you mouse over a vertical line, two handles appear at the top and bottom of the line. Drag the handles up or down to narrow the focus down to the parameters ranges that interest you.

When you select a point on the results, the right-click context menu provides options for exporting data and saving candidate points as design, refinement, verification, or custom candidate points.

Candidate Point Results: Properties

To specify the properties of the candidate point results in the **Table** and **Chart** panes, in the **Outline** pane under **Results**, select **Candidate Points**. Then, in the **Properties** pane, edit the properties. The properties available depend on the type of optimization selected.

Table

Determines the properties of the results displayed in the **Table** pane. For the **Show Parameter Relationships** property, select the **Value** check box to display parameter relationships in the **Table** pane for the results.

Chart

Determines the properties of the results displayed in the **Chart** pane. Select the **Value** check box to enable the property.

- **Display Parameter Full Name:** Select to display the full parameter name rather than the short parameter name.
- **Show Candidates:** Select to show candidates in the results.
- **Show Samples:** Select to show samples in the results.
- **Show Starting Point:** Select to show the starting point in the results (NLPQL and MISQP only).
- **Show Verified Candidates:** Select to show verified candidates in the results. This option is available for a **Response Surface Optimization** system only. Candidate verification is not necessary for a **Direct Optimization** system because the points result from a real solve, rather than an estimation.
- **Coloring Method:** Select whether the results should be colored by candidate type or source type:
 - **by Candidate Type:** Different colors are used for different types of candidate points. This is the default value.
 - **by Source Type:** Output parameter values calculated from simulations are displayed in black. Output parameter values calculated from a response surface are displayed in the custom color selected on the **Response Surface** tab in the **Options** window. For more information, see [Response Surface Options \(p. 38\)](#).

Input Parameters

Each of the input parameters is listed in this section. In the **Enabled** column, you can select or clear check boxes to enable or disable input parameters. Only enabled input parameters are shown on the chart.

Output Parameters

Each of the output parameters is listed in this section. In the **Enabled** column, you can select or clear check boxes to enable or disable output parameters. Only enabled output parameters are shown on the chart.

Generic Chart Properties

You can modify various generic chart properties for the results. For more information, see [Setting Chart Properties](#).

Using the Sensitivities Chart (GDO)

The Sensitivities chart for a goal-driven optimization (GDO) shows the global sensitivities of the output parameters with respect to the input parameters. You can display this chart as a bar chart or a pie chart by changing the setting for **Mode** in the chart's **Properties** pane. You can also select which parameters you want to display on the chart from the **Properties** pane by selecting or clearing the check boxes next to the parameters.

You can change various generic [chart properties](#) can be changed for this chart.

Note:

- The Sensitivities chart is available only for the Screening and MOGA optimization methods.
- If the p-Value calculated for a particular input parameter is above the **Significance Level** specified in **Tools** → **Options** → **Design Exploration**, the bar for that parameter is shown as a flat line on the chart. For more information, see [Viewing Significance and Correlation Values \(p. 72\)](#).

Using the Tradeoff Chart

When an **Optimization** cell is solved, the Tradeoff chart is created by default under **Results** in the **Outline** pane. The Tradeoff chart is a scatter chart representing the generated samples. The colors applied to the points represent the Pareto front to which they belong. Red points are the worst. Blue points are the best.

You can change the properties for the chart in the **Properties** pane.

- You can set **Mode** to **2D** or **3D**.
- You can select which parameter to display on each axis of the chart by selecting the parameter from the list next to the axis name.
- You can limit the Pareto fronts shown by moving the slider or entering a value in the field above the slider.

You can change various generic [chart properties](#) for this chart.

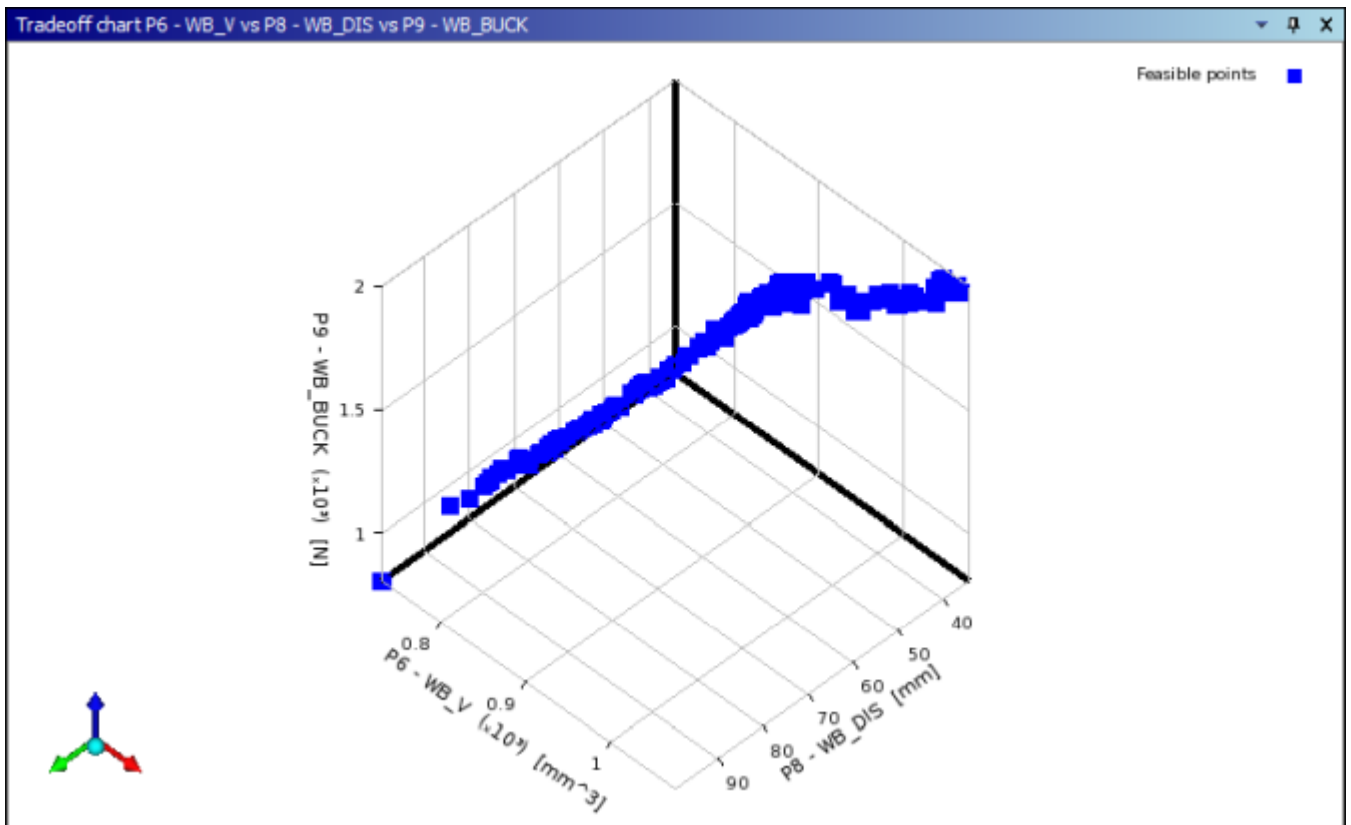
Using Tradeoff Studies

In the Tradeoff chart, the samples are ranked by non-dominated Pareto fronts. You can view the tradeoffs that most interest you. To make sense of the true nature of the tradeoffs, you must view the plots with the output parameters as the axes. This approach shows which goals can be achieved and whether this entails sacrificing the goal attainment of other outputs. Typically, a Tradeoff chart shows you a choice of possible, non-dominated solutions from which to choose.

When an optimization is updated, you can view the best candidates (up to the requested number) from the sample set based on the stated objectives. However, these results are not truly representative of the solution set, as this approach obtains results by ranking the solution by an aggregated weighted

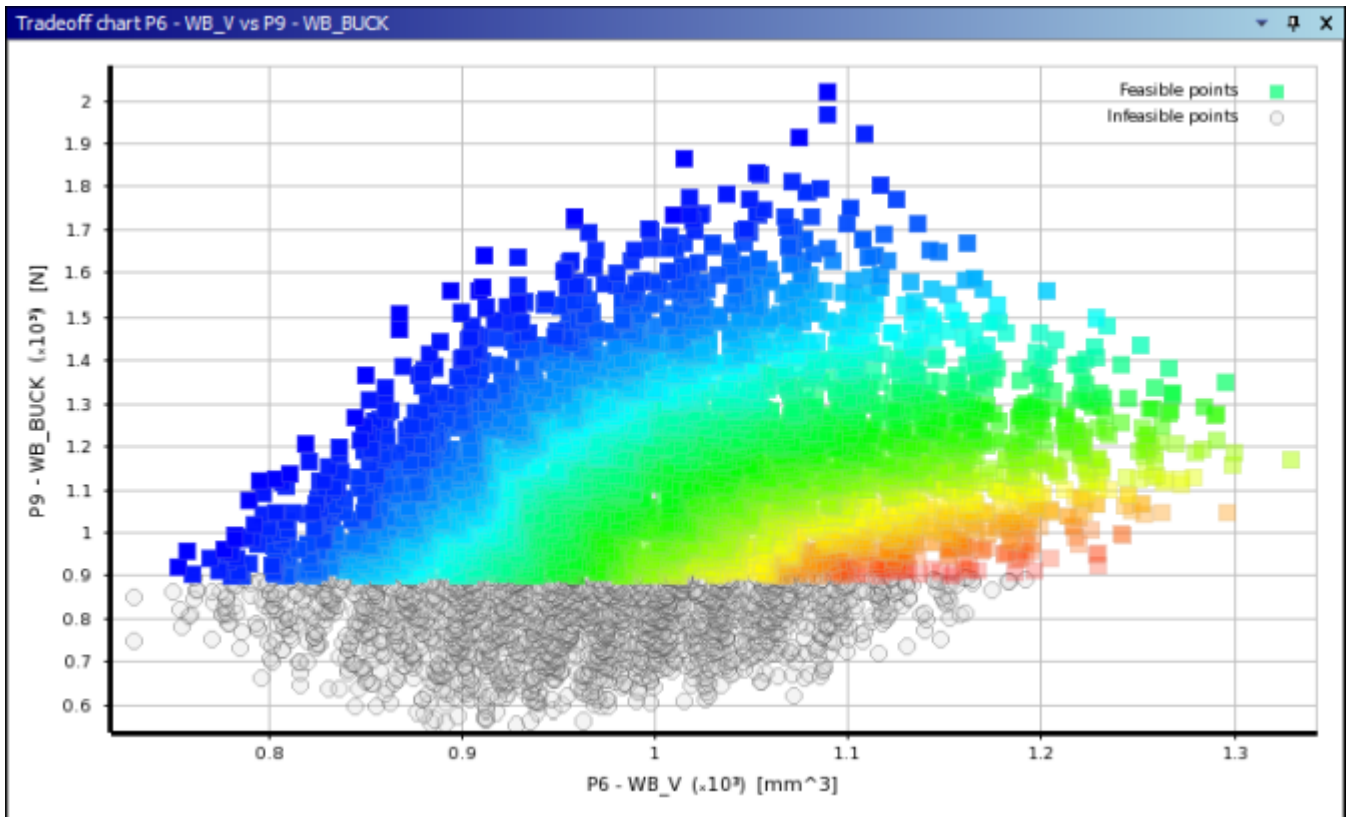
method. Schematically, this represents only a section of the available Pareto fronts. To display different sections, you change the weights for the **Objective Importance** or **Constraint Importance** property in the **Properties** pane. This postprocessing step helps in selecting solutions if you are sure of your preferences for each parameter.

The following figures shows the results of the tradeoff study performed on a MOGA sample set. The first Pareto front (non-dominated solutions) is represented by blue points on the output-axis plot. You can move the slider in the **Properties** pane to the right to add more fronts, effectively adding more points to the Tradeoff chart. Additional points added in this way are inferior to the points in the first Pareto front in terms of the objectives or constraints that you specified. However, in some cases where there are not enough first Pareto front points, these additional points can be necessary to obtain the final design. You can right-click individual points and save them as design points or response points.



In 2D and 3D Tradeoff charts, MOGA always ensures that feasible points are shown as being of better quality than the infeasible points. It uses different markers to indicate them in the chart. Colored rectangles represent feasible points. Gray circles represent infeasible points. Infeasible points are available if any of the objectives are defined as constraints. You can enable or disable the display of infeasible points in the **Properties** pane.

Also, in both 2D and 3D Tradeoff charts, the best Pareto front is blue. The fronts gradually transition to red for the worst Pareto front. The following figure is a typical 2D Tradeoff chart with feasible and infeasible points.



For more information on Pareto fronts and Pareto-dominant solutions, see [GDO Principles \(p. 351\)](#).

Using the Samples Chart

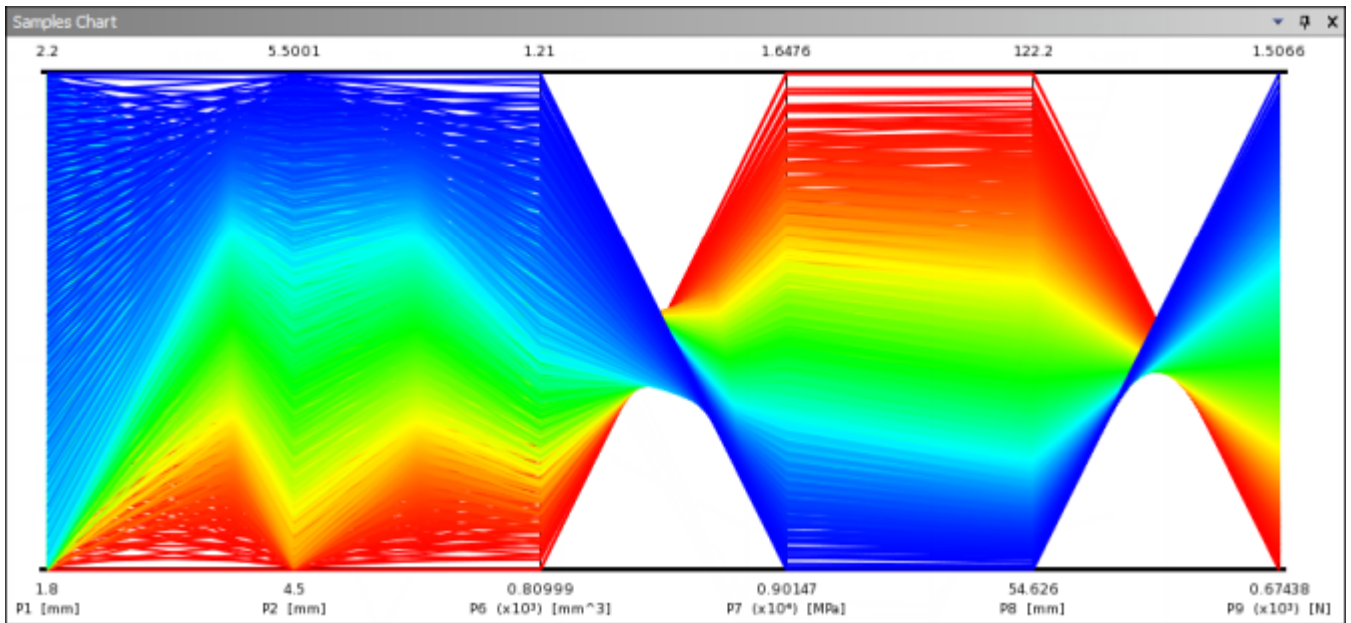
The Samples chart is a postprocessing feature that allows you to explore a sample set given defined goals. After solving an **Optimization** cell, a **Samples** chart appears in the **Outline** pane under **Results**.

This chart provides a multidimensional representation of the parameter space that you are studying. It uses the parallel Y axes to represent all of the inputs and outputs. Each sample is displayed as a group of lines, where each point is the value of one input or output parameter. The color of the line identifies the Pareto front to which the sample belongs. You can also set the chart so that the lines display the best candidates and all other samples.

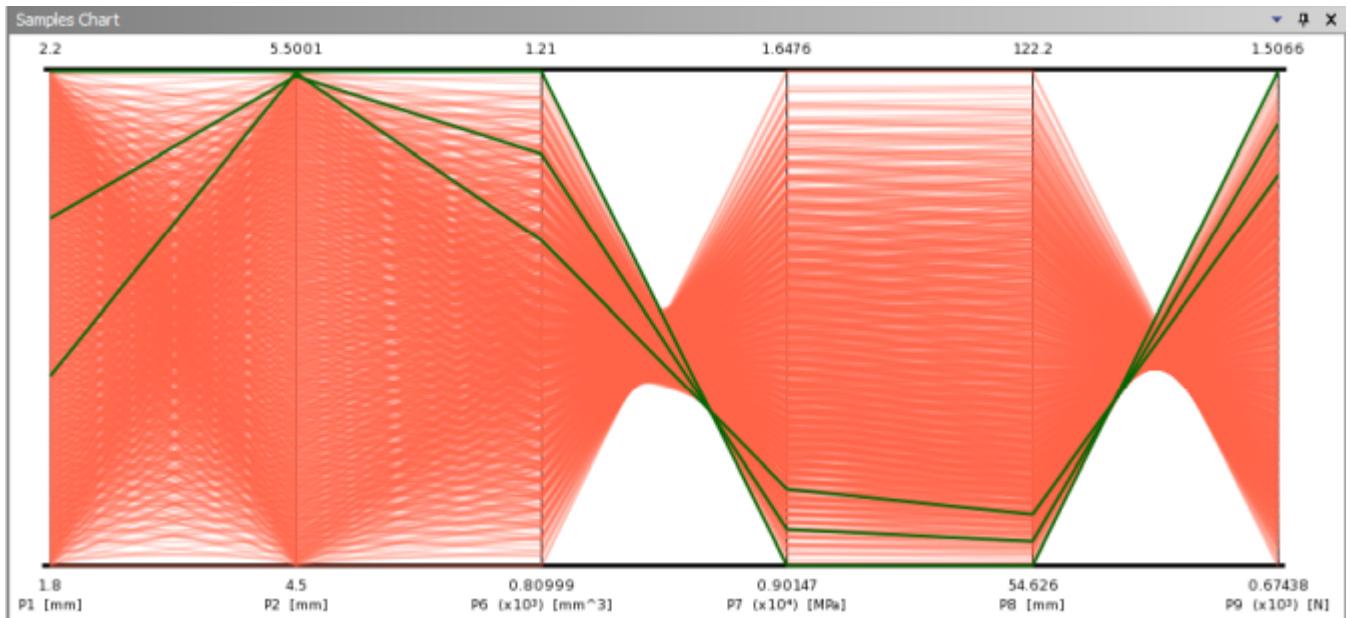
While the Tradeoff chart can show only three parameters at a time, the Samples chart can show all parameters at once, making it a better option for exploring the parameter space. Because of its interactivity, the Samples chart is a powerful exploration tool. Using the axis sliders in the **Properties** pane to easily filter each parameter provides you with an intuitive way to explore alternative designs. The Samples chart dynamically hides the samples that fall outside of the bounds. Repeating this operation with each axis allows you to manually explore and find trade-offs.

The **Properties** pane for the Samples chart has two choices for **Mode**: **Candidates** and **Pareto Fronts**. You can display the candidates with the full sample set or display the samples by Pareto front (same as the Tradeoff chart). If you set **Mode** to **Pareto Fronts**, the **Coloring method** property becomes available, allowing you to specify **by Pareto Front** or **by Samples** for coloring the chart.

In the following Samples chart, **Mode** is set to **Pareto Fronts** and **Coloring method** is set to **by Pareto Front**. The gradient ranges are from blue for the best to red for the worst.



When **Mode** is set to **Candidates**, **Coloring method** is not shown because coloring is always by samples.



Samples Chart: Properties

To specify the properties for a Samples chart, in the **Outline** pane under **Results**, select **Samples**. Then, in the **Properties** pane, edit the chart properties. The properties available depend on the type of optimization selected.

Chart Properties

Specifies properties for the chart.

- **Display Parameter Full Name:** Select to show the full parameter name rather than the short parameter name.
- **Mode:** Specifies whether to display the chart by candidates or Pareto fronts. Choices are **by Pareto Fronts** and **by Samples**. If **Pareto Fronts** is selected, **Coloring method** becomes available as the last option so that you can specify how to color the chart.
- **Number of Pareto Fronts to Show:** Specifies the number of Pareto fronts to display on the chart.

Input Parameters

Lists the input parameters. In the **Enabled** column, you can select or clear check boxes to enable or disable input parameters. Only enabled input parameters are shown on the chart.

Output Parameters

Lists the output parameters. In the **Enabled** column, you can select or clear check boxes to enable or disable output parameters. Only enabled output parameters are shown on the chart.

Generic Chart Properties

Specifies various generic chart properties. For more information, see [Setting Chart Properties](#).

Using ROMs

The following sections explain how to produce parametric ROMs (reduced order models) from Fluent steady analyses and then use them to evaluate models in 2D or 3D to rapidly explore the variation of results:

[ROM Overview](#)

[ROM Workflow](#)

[ROM Production Example for Fluent](#)

[Exporting the ROM](#)

[Consuming an FMU 2.0 File in Twin Builder](#)

[Consuming a ROMZ File in the ROM Viewer](#)

[Consuming a ROMZ File in Fluent](#)

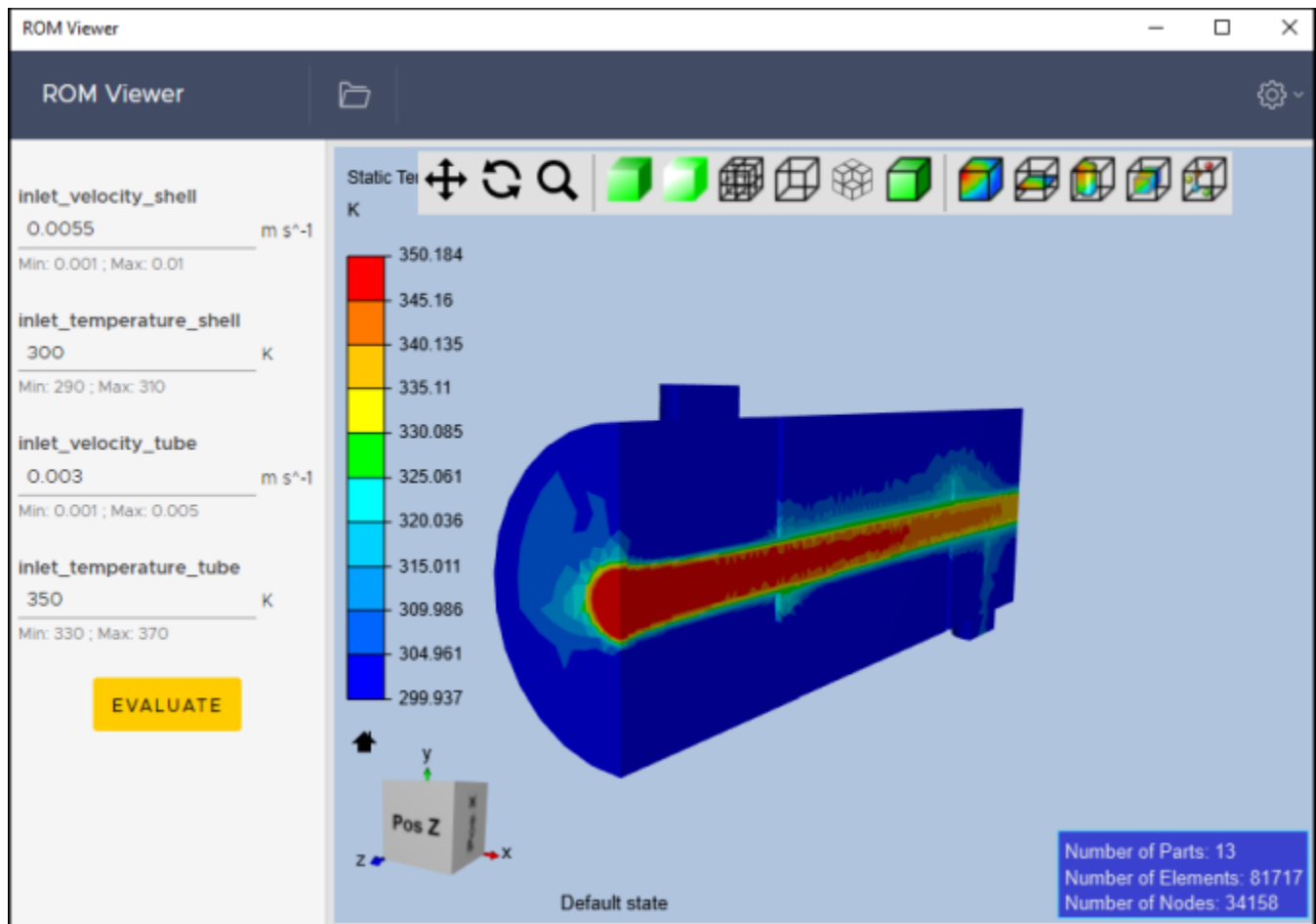
[Analyzing and Troubleshooting ROM Production](#)

[Quality Metrics for ROMs](#)

[ROM Limitations and Known Issues](#)

ROM Overview

You can produce a ROM by learning the physics of a given model and extracting its global behavior from offline simulations. As a standalone digital object, a ROM can be consumed outside of its production environment for computationally inexpensive, near real-time analysis. The following figure shows a ROM for a steady fluids simulation of a heat exchanger.



In a parametric ROM, you can evaluate the model and rapidly explore the variation of the results, depending on input parameter values. In this particular example, to explore the variation of the result, you move the sliders in the left pane to change input parameter values and then click **Evaluate** to display updated results. You can also use tools to show the mesh and set the mesh translucency. Above the list of regions, **Show Result** provides for showing and hiding result values.

Because calculating a ROM result requires only simple algebraic operations (such as vectors summation and response surfaces evaluations), this step is computationally inexpensive compared to the FOM (full order model) processing step. Not only is the ROM processing step several orders of magnitude cheaper, but also the ROM can easily be delivered to and consumed by any number of users, yielding significant returns on your initial investment in its production.

ROM Workflow

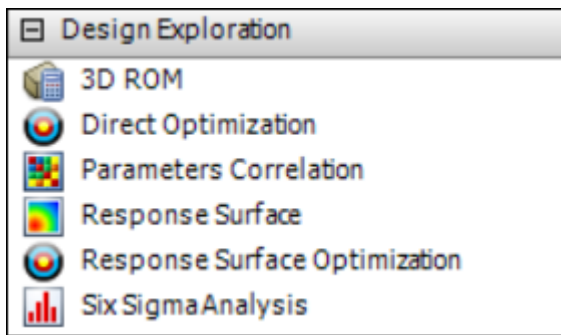
The ROM workflow consists of two distinct stages:

[ROM Production](#)

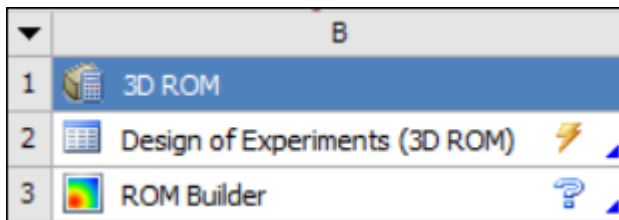
[ROM Consumption](#)

ROM Production

In Workbench, you use a DesignXplorer **3D ROM** system to drive ROM production from either a 2D or 3D simulation. In the Workbench **Toolbox**, the **3D ROM** system is visible under **Design Exploration**.



A **3D ROM** system is based on a Design of Experiments (DOE) and its design points, which automate the production of solution snapshots and the ROM itself.



You define the input parameters and content of the ROM in the simulation environment, which is currently limited to Fluent. A medium-sized ROM generally has 3 to 6 input parameters, while a very large ROM might have more than 15.

When lots of input parameters are enabled, you might need to increase the number of ROM snapshot files to maintain ROM accuracy. If you decide that you no longer want to vary an input parameter, you can disable it.

A ROM must always have a least one output parameter. While output parameters have no impact on ROM production, you can use them to monitor results while DesignXplorer updates the design points.

While ROM setup is specific to the Ansys product, the ROM production workflow is generic. This means that as ROM support is extended to additional Ansys products in future releases, the steps that you take to produce a ROM will be the same in all simulation environments.

Note:

- It is not possible to update DesignXplorer systems when the project includes a **3D ROM** system and **Update Option** is set to **Submit to Remote Solve Manager**. You must change **Update Option** to **Run in Foreground** and might want to set **Submit to Remote Solve Manager** at the solution component level.
 - If a Workbench project with a ROM was created in a version earlier than 2019 R3, the previously existing ROM system name (**ROM Builder**) and DOE cell name (**Design of Experiments (RB)**) are shown.
-

Because ROM production requires several simulations, this stage can be computationally expensive. However, once the ROM is built, it can be consumed at negligible cost.

ROM Consumption

The workflows for consuming ROMs can differ from one application to another. Currently, to consume a ROM, you [export the ROM \(p. 251\)](#) as either an FMU 2.0 file or a ROMZ file, depending on which consumption environment is targeted.

- An FMU 2.0 file is a compressed package containing the data and libraries that are needed for ROM consumption. You can import this file into Ansys Twin Builder, which provides for displaying the fields in the **ROM Viewer** while the simulation is running and after the simulation is complete. For more information, see [Consuming an FMU 2.0 File in Twin Builder \(p. 251\)](#).
- A ROMZ file is also a compressed package containing the data and libraries that are needed for ROM consumption. If you launch the **ROM Viewer** separately from Twin Builder, you can consume the ROMZ file in the viewer. For more information, see [Consuming a ROMZ File in the ROM Viewer \(p. 263\)](#). You can also import the ROMZ file into Fluent in Workbench so that you can evaluate results directly in Fluent. For more information, see [Reduced Order Model \(ROM\) Evaluation in Fluent](#) in the *Fluent User's Guide*.

Because an exported ROM is a standalone digital object, deployment for consumption by many users is quick and easy.

ROM Production Example for Fluent

This section describes a steady fluids simulation of a heat exchanger and provides prerequisites for downloading and extracting sample files and then enabling advanced DesignXplorer options. It then describes how to use the standard workflow for automatically running a set of design points in a Workbench project to produce a ROM for this heat exchanger:

Note:

For advanced Fluent users, a beta workflow exists for manually defining or generating in standalone Fluent all of the appropriate files for finally producing the Fluent ROM in Workbench. For more information about this advanced workflow, see the *DesignXplorer Beta Features Manual*.

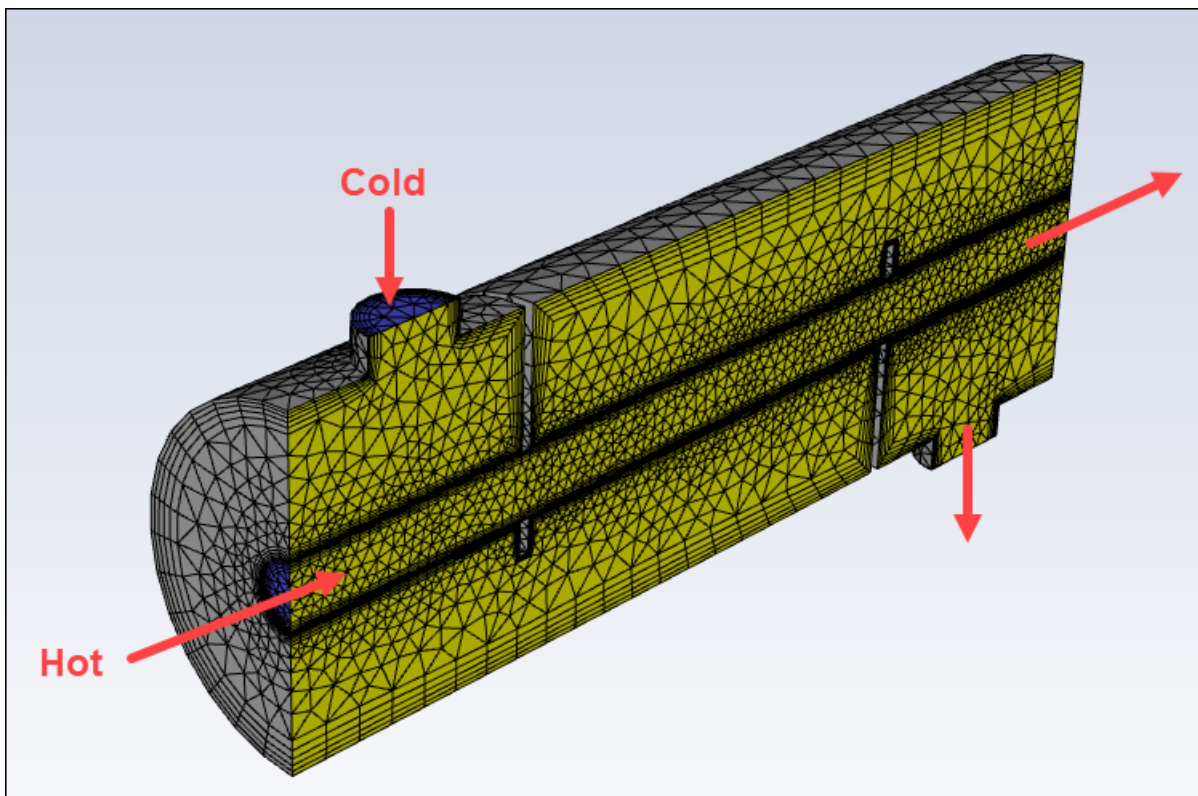
[Heat Exchanger Model](#)

[Prerequisites](#)

[Producing the ROM](#)

Heat Exchanger Model

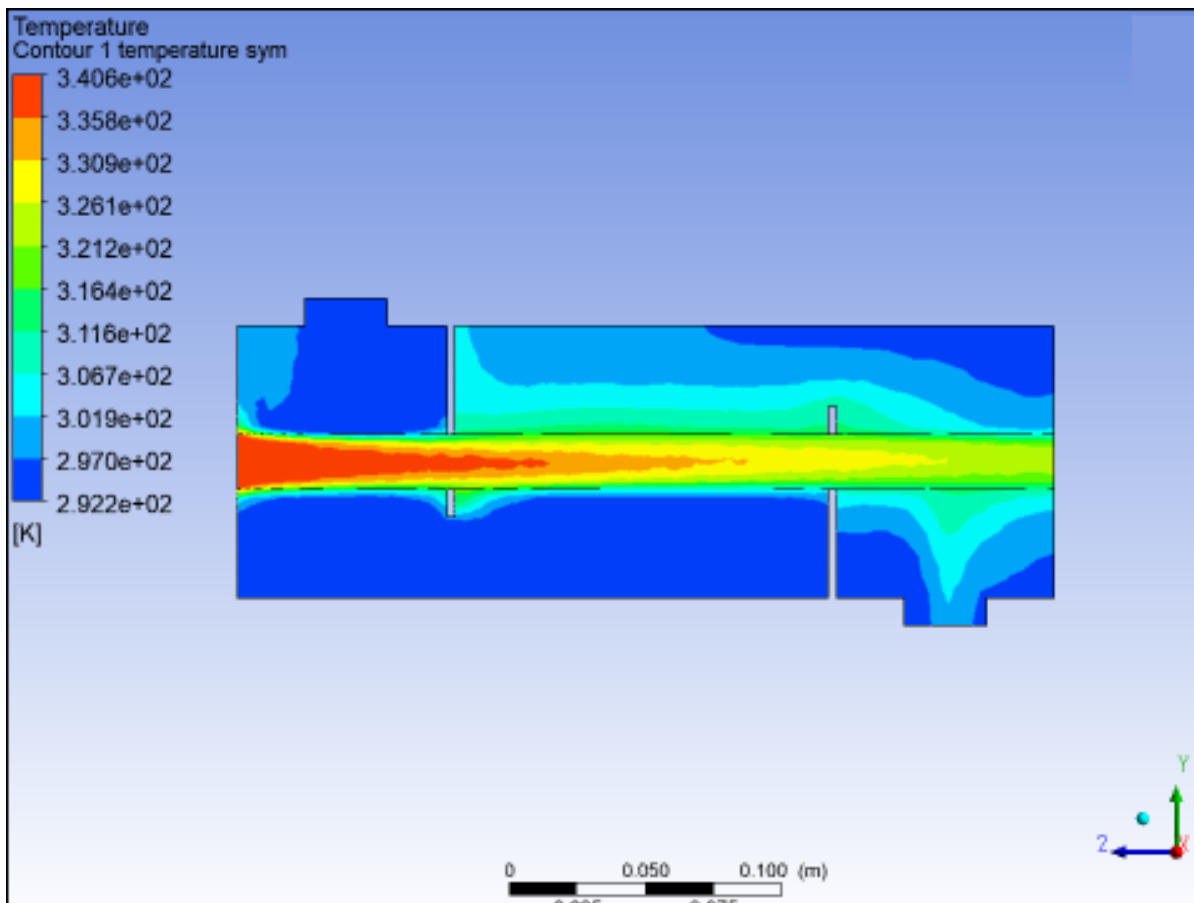
In the heat exchanger, hot water flows into a straight tube that is made of thin aluminium. This tube is cooled by the cold water around it, which flows at a different velocity.



The model has four input parameters:

- Inlet velocity in the shell
- Inlet temperature in the shell
- Inlet velocity in the tube
- Inlet temperature in the tube

An example contour plot for temperature might look like this:



To assess ROM accuracy, you can view quality metrics and run both verification points and refinement points. For more information, see [Quality Metrics for ROMs \(p. 276\)](#).

Once ROM accuracy is verified, you can export the ROM for consumption in Twin Builder, the **ROM Viewer** that you launch separately from Twin Builder, or Fluent in Workbench. For more information, see [Exporting the ROM \(p. 251\)](#).

Prerequisites

The following steps explain how to download and extract sample files and then how to enable DesignXplorer advanced options:

1. Click [here](#) to download a ZIP file with the sample files.
2. Extract the files to a directory that you can access.
3. Start Workbench.
4. Enable DesignXplorer advanced options:
 - a. Select **Tools** → **Options** to open the **Options** window.
 - b. On the **Design Exploration** page, select the **Show Advanced Options** check box.
 - c. Click **OK**.

DesignXplorer now displays advanced options in italic type in various panes. Advanced ROM options provide for opening the ROM Builder log file and setting a user-generated ROM mesh file.

Producing the ROM

This section describes how to use the standard workflow for automatically running a set of design points in a Workbench project to produce a ROM for a steady fluids simulation of the heat exchanger:

[Setting Up ROM Building in Fluent](#)

[Creating the ROM in the ROM Builder](#)

To estimate minimum production time for a ROM, you can multiply the time it takes to update one design point by the number of learning points. For this example, producing the ROM takes approximately 40 minutes on 10 cores.

For your convenience, project archive files without solved design points (**HeatExchanger.wbpz**) and with solved design points (**HeatExchanger_DOE_Solved.wbpz**) are included in the directory where you extracted the sample files.

Note:

If you want to learn how to use a ROM rather than how to create a ROM, you can skip to [Consuming a ROMZ File in the ROM Viewer \(p. 263\)](#). You can then consume the supplied ROMZ file (**HeatExchanger.romz**). This file is included in the directory where you extracted the sample files.

Setting Up ROM Building in Fluent

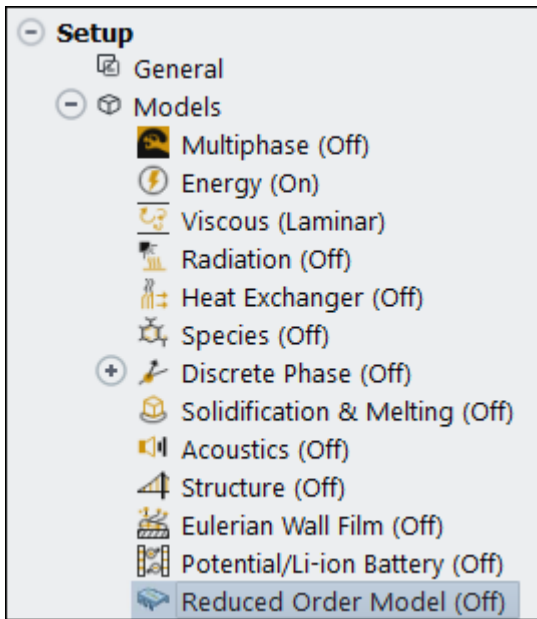
This topic explains how to enable and load the ROM feature in Fluent and then select flow variables and cell zones. When you close Fluent, your changes are pushed to your Workbench project.

1. In Workbench, select **File** → **Open**, navigate to the directory where you extracted the sample files, and open the project archive file **HeatExchanger.wbpz**.
2. Save the project to a new file name.
3. To view the parameters for this project:
 - a. In the **Project Schematic**, double-click the **Parameter Set** bar to open it.
 - b. In the **Outline** pane, look at the input and output parameters, which have been well defined.

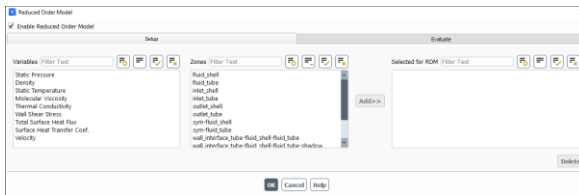
ID	Parameter Name	Value	Unit
[-] Input Parameters			
[-] Fluid Flow (Fluent) (B1)			
P1	inlet_velocity_shell	0.01	m s ⁻¹ ▾
P2	inlet_temperature_shell	300	K ▾
P3	inlet_velocity_tube	0.005	m s ⁻¹ ▾
P4	inlet_temperature_tube	350	K ▾
New input parameter	New name	New expression	
[-] Output Parameters			
[-] Fluid Flow (Fluent) (B1)			
P5	outlet_temperature_shell	302.74	K
P6	outlet_temperature_tube	331.69	K
New output parameter		New expression	
Charts			

- c. Close the **Parameter Set** bar.
4. In the **Project Schematic**, right-click the **Mesh** cell and select **Update**.
 5. When the update finishes, double-click the **Setup** cell to open Fluent, clicking **Yes** when asked whether to load the new mesh.
 6. When the **Fluent Launcher** window opens, if necessary, adjust the number of processors based on your local license and hardware configurations and then click **Start**.
 7. Once Fluent has started, do the following:
 - a. To make **Reduced Order Model (Off)** visible in the tree under **Setup** → **Models**, enable and load the ROM addon module by executing this command in the Fluent console:

```
define models addon 11
```
 - b. On the ribbon's **Solution** tab, click the button in the **Initialization** group for initializing the entire flow field and then click **Calculate** in the **Run Calculation** group.
 - c. When the calculation completes, click **OK** to close the dialog box. Then, in the tree under **Setup** → **Models**, double-click **Reduced Order Model (Off)**.



- d. In the **Reduced Order Model** window that opens, select the **Enable Reduced Order Model** check box to expand the window so that you can set up the ROM.



On the **Setup** tab, each pane displays a filtering option and buttons for toggling which values to show, how to show these values, and selecting and deselecting all shown values. Any Fluent custom function fields that you have defined are included in the list of variables available for selection.

Note:

The **Evaluate** tab is available for postprocessing and evaluating the solved ROM in Fluent only after the ROMZ file is imported. For more information, see [Reduced Order Model \(ROM\) Evaluation in Fluent](#) in the *Fluent User's Guide*.












- e. Select the variables and zones to include in the ROM and then click **Add** to move them to the **Selected for ROM** list.

Because you cannot add selections to the ROM later, ensure that you add all variables and zones of interest. If you want to delete a particular selection from the **Selected for ROM** list, click it and then click **Delete** beneath the list.

For this example, select:

- **Static Temperature:** all zones

- **Velocity:** all zones except wall zones
- f. Once you have finalized your selections, click **OK**.
Everything in the **Selected for ROM** list will be included in the ROM.
 - g. In the Fluent toolbar, click the button with the Ansys logo.
This button synchronizes the Workbench cell, pushing your changes to the **Design of Experiments (3D ROM)** cell.
 - h. If you are asked to save your changes, select the option for saving changes for current and future calculations.
 - i. Select **File** → **Close Fluent**.
8. In the Workbench project, do the following:
 - a. On the **Project Schematic**, right-click the **Fluid Flow (Fluent)** system and select **Update**.
 - b. When the update finishes, select **View** → **Files** and locate the ROM snapshot file (ROMSNP) that has been created.

Files				
	A	B	C	D
1	Name	Ce...	Size	Type
2	 Geom.agdb	A2,B2	2 MB	Geometry File
3	 FFF.set	B4	283 KB	FLUENT Model File
4	 HeatExchanger.wbpj		44 KB	Workbench Project File
5	 FFF.mshdb	B3	4 MB	Mesh Database File
6	 FFF-Setup-Output.cas.gz	B4	2 MB	FLUENT Case File
7	 Fluid Flow Fluent.cst	B6	48 KB	CFD-Post State File
8	 FFF.msh	B3,B4	4 MB	Fluent Mesh File
9	 FFF.ip	B5	5 MB	FLUENT Interpolation Data File
10	 cache.romcch	Global	860 B	ROM Snapshot Cache File
11	 260d1fe4-2c6f-4332-8442-4f2fddd3e56	Global	3 MB	ROM Snapshot File
12	 FFF-6-00100.dat.gz	B1	5 MB	FLUENT Data File

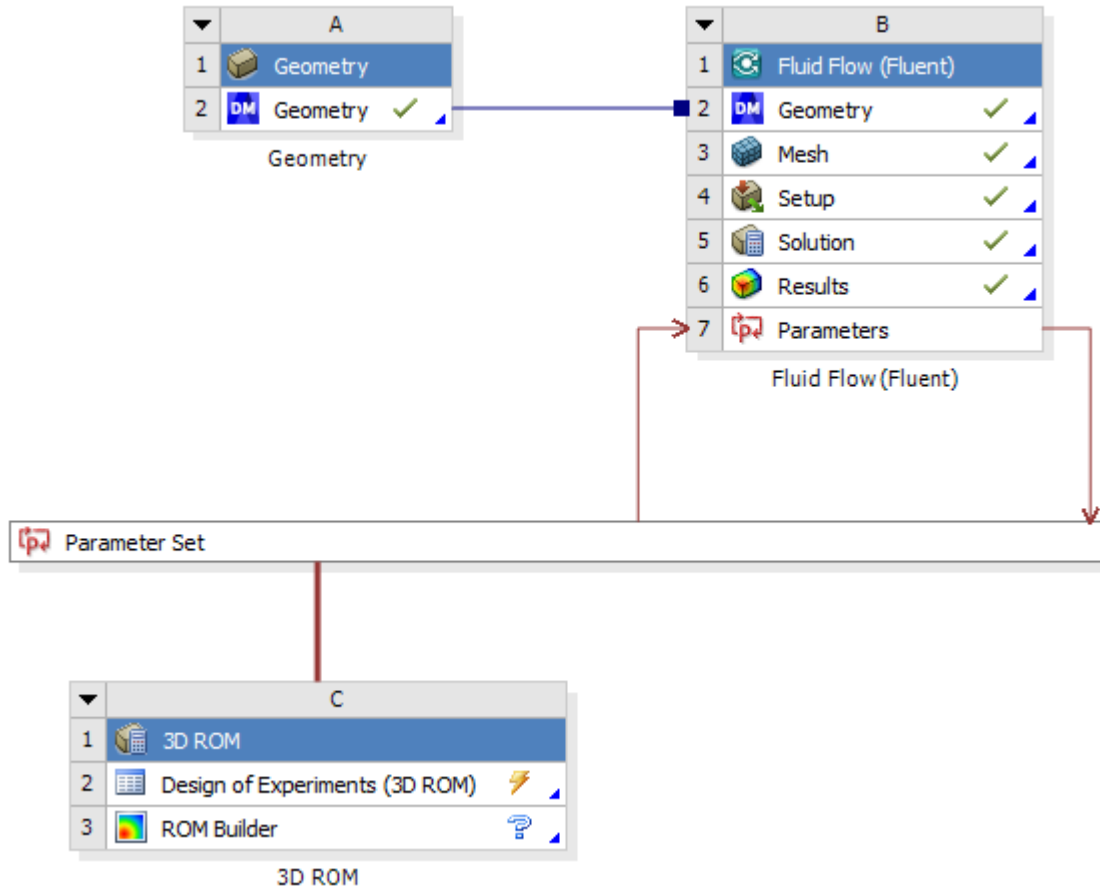
This global file captures the results for the current parameters configured for **DP 0**. It is one of a set of snapshot files that is needed to build the ROM.

Creating the ROM in the ROM Builder

This topic explains how to add, configure, and update the **3D ROM** system.

1. In the Workbench **Toolbox** under **Design Exploration**, double-click **3D ROM** to add a system of this type to the **Project Schematic**.

A **3D ROM** system is inserted under the **Parameter Set** bar. In the **Design of Experiments (3D ROM)** cell, (**3D ROM**) indicates that the design points in this cell are used to build the ROM. The DOE for a **3D ROM** system can share data only with the DOE for another **3D ROM** system.



2. Configure the **Design of Experiments (3D ROM)** cell:

Tip:

You can import data from an external CSV file into the **Design of Experiments (3D ROM)** cell. For more information, see [Importing Data from a CSV File](#) (p. 316) and [Exporting and Importing ROM Snapshot Archive Files](#) (p. 268).

- a. Double-click the **Design of Experiments (3D ROM)** cell to open it.
- b. In the **Outline** pane, select **Design of Experiments (3D ROM)**.
- c. In the **Properties** pane, review and edit DOE properties as necessary.

Accepting the default values is generally recommended.

Properties of Outline A2: Design of Experiments (3D ROM)		
	A	B
1	Property	Value
2	[-] Design Points	
3	Preserve Design Points After DX Run	<input type="checkbox"/>
4	[-] Failed Design Points Management	
5	Number of Retries	0
6	[-] ROM	
7	ROM Mesh File	Choose a custom mesh file. <input type="button" value="Browse..."/>
8	[-] Design of Experiments	
9	Design of Experiments Type	Optimal Space-Filling Design
10	Design Type	Max-Min Distance
11	Maximum Number Of Cycles	10
12	Samples Type	User-Defined Samples
13	Random Generator Seed	0
14	Number of Samples	32

The default value for **Design of Experiments Type** is **Optimal Space-Filling Design**. The default value for **Number of Samples** depends on the number of input parameters. DesignXplorer calculates this value by multiplying the number of enabled input parameters by eight. In the example, four input parameters are enabled. Consequently, **Number of Samples** is set to **32**.

- d. For each input parameter in the **Outline** pane, select it and edit the lower and upper bounds in the **Properties** pane. For this example, use the following values.
 - For **P1 - inlet_velocity_shell**, set the bounds from **0.001** to **0.01** m/s.
 - For **P2 - inlet_temperature_shell**, set the bounds from **290** to **310** K.
 - For **P3 - inlet_velocity_tube**, set the bounds from **0.001** to **0.005** m/s.
 - For **P4 - inlet_temperature_tube**, set the bounds from **330** to **370** K.
- e. In the **Outline** pane, select **Design of Experiments (3D ROM)**.
- f. In the **Properties** pane, select the **Preserve Design Points After DX Run** check box and set **Report Image** to **FFF-Results:Figure001.png**.

In the **Table** pane, the **Report Image** column is added. The image for each design point will display temperature results on the symmetry face. These figures are defined in the **Results** cell in CFD-Post. For more information, see [Figure Command](#) in the *CFD-Post User's Guide* and [Viewing Design Point Images in Tables and Charts](#) (p. 304).

3. In the toolbar, click **Preview** to generate the 32 design points without updating them.

To update the 32 design points takes about 40 minutes on 10 cores.

Tip:

To avoid the wait, you can save and close this project and then open the project archive file `HeatExchanger_DOE_Solved.wbpz` and save it to a new file name. Because the 32 design points are already updated in this file, you can skip to step 7.

4. In the toolbar, click **Update**.

For each design point, a report image and a ROM snapshot file (ROMSNP) are produced. For more information, see [ROM Snapshot Files \(p. 265\)](#).

5. When the update finishes, close the **Design of Experiments (3D ROM)** cell.
6. Save the project.
7. Configure the **ROM Builder** cell:
 - a. On the **Project Schematic**, double-click the **ROM Builder** cell to open it.
 - b. In the **Outline** pane, select **ROM Builder**.

The **Table** pane displays a summary of all the variables that are included in the ROM.

- c. In the **Properties** pane, verify that **Solver System** is set to the correct Ansys product.

The default is **Fluid Flow (Fluent) (FFF)** because this is only the system in which a ROM is set up.

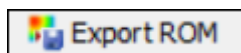
- d. For **Construction Type**, accept the default value, which is **Fixed Number of Modes**.
- e. For **Number of Modes**, accept the default value of **10**.

Note:

If the design points for the **Design of Experiments (3D ROM)** cell was smaller than **10**, you would set **Number of Modes** to the number of design points.

- f. In the toolbar, click **Update** to build the ROM.

When the update completes, a toolbar button is enabled for exporting the ROM to either an FMU 2.0 file or ROMZ file.



For more information, see [Exporting the ROM \(p. 251\)](#).

8. To assess the ROM accuracy, in the **Outline** pane, select **Goodness Of Fit**. Then, in the chart, check the error metrics at the DOE points. Learning points include DOE points plus refinement points. For more information, see [Quality Metrics for ROMs \(p. 276\)](#).
9. To better assess the ROM accuracy, run some verification points:
 - a. In the **Outline** pane, select **ROM Builder**.
 - b. In the **Properties** pane, select the **Preserve Design Points After DX Run** check box and the **Generate Verification Points** check box.

Preserving design points is necessary for producing report images.
 - c. Set **Number of Verification Points** to **5**.
 - d. For **Report Image**, select a figure.
 - e. In the toolbar, click **Update** to run the five verification points.
 - f. When the update finishes, in the **Outline** pane, select **Verification Points**; then in the **Table** pane, check the verification point values and report images.
 - g. In the **Outline** pane, select **Goodness Of Fit**.
 - h. In in the chart, check the error metrics at the verification points.

To improve ROM accuracy, you can choose to run some additional design points. You can do so in either the **Design of Experiments** cell or the **ROM Builder** cell.

- In the **Design of Experiments** cell:
 1. Set **Design of Experiments Type** to **Custom + Sampling**.
 2. Set **Number of Samples** to the total number of points desired.
 3. In the toolbar, click **Preview** to generate the design points without updating them.

The additional points are added where the distances between existing design points are the highest.

- In the **ROM Builder** cell, manually add refinement points.

In the **Table** pane for the **Design of Experiments (3D ROM)** cell, you can see the statuses of ROM snapshot files and perform many other operations. For more information, see [ROM Snapshot Files \(p. 265\)](#). ROM snapshot files and the ROM Builder log file provide for [analyzing and troubleshooting ROM production \(p. 264\)](#).

If you want to edit output values for a design point or add, import, or copy design points, you must change the **Design of Experiments Type** property to **Custom** or **Custom + Sampling**. To edit one or more output values for a design point, you right-click the output value and then select either the option for setting it or all output values as editable. For more information, see [Editable Output Parameter Values \(p. 314\)](#). For any design points that you add manually, you must set ROM snapshot files. For more information, see [Setting Specific ROM Snapshot Files for Design Points \(p. 267\)](#).

When you edit a design point, you must update the ROM.

Exporting the ROM

Once a ROM is verified, you can export it to either an FMU 2.0 file or ROMZ file:

1. To open the **Export ROM** dialog box, do one of the following:
 - In the toolbar for the **ROM Builder** cell, click **Export ROM**.
 - In the **Project Schematic**, right-click the **ROM Builder** cell and select **Export ROM**.
2. In the **Export ROM** dialog box, navigate to the location where you want to save the file.
3. For **File name**, enter the name to assign the file.
4. For **Save as type**, select the file type to which to export the file. Choices are:
 - **FMU Version 2 Files (*.fmu)**. An FMU (Functional Mock-up) 2.0 file can be consumed by anyone who has access to Ansys Twin Builder or any other tool that can read this file type. When you import the FMU 2.0 file into Twin Builder, you can display the fields in the **ROM Viewer** while the simulation is running and after the simulation is complete. For more information, see [Consuming an FMU 2.0 File in Twin Builder \(p. 251\)](#).
 - **ROM Files (*.romz)**. A ROMZ file can be consumed by anyone who can launch the **ROM Viewer** separately from Twin Builder. For more information, see [Consuming a ROMZ File in the ROM Viewer \(p. 263\)](#). The ROMZ file for a ROM can also be imported into Fluent in Workbench, where results can be evaluated directly in Fluent. For more information, see [Consuming a ROMZ File in Fluent \(p. 264\)](#).
5. Click **Save** to export the ROM to the selected file type.

Note:

Using an FMU 2.0 file exported from DesignXplorer implies the approval of the terms of use supplied in the `License.txt` file. To access `License.txt`, use a zip utility to manually extract all files from the `.fmu` package.

Consumption of the exported ROM file can require a lot of memory, depending on the ROM setup and the ROM mode count.

Consuming an FMU 2.0 File in Twin Builder

Twin Builder is the Ansys go-to product for 3D ROMs. You can consume an FMU 2.0 file that was exported from a DesignXplorer 3D ROM by adding it in Twin Builder as an FMU component. When this ROM

opens in the **ROM Viewer** in Twin Builder, you gain Predictive Engineering Insight (PEI) by watching multiphysics simulations run in real time.

Note:

To experiment, you can consume the supplied FMU 2.0 file for the supplied [ROM production example \(p. 240\)](#) (`HeatExchanger.fmu`) in Twin Builder. This file resides in the directory where you extracted the sample files.

An FMU 2.0 file that is created by exporting a 3D ROM has all the enabled input parameters from the DOE and works only within the ranges for these inputs. The FMU outputs are the minimum, maximum, and average values for each selected variable on each selected zone. Existing scalar outputs from Workbench are ignored. Currently, there is no way to add custom scalar outputs.

Otherwise, you can use an FMU 2.0 file that is created by exporting a 3D ROM in the same manner as any other FMU file. For more information on these files and using 3D ROMs in Twin Builder, search the Twin Builder help for these topics:

- FMU Components
- Static ROM Components

Twin Builder Setup

Twin Builder setup consists of adding the FMU 2.0 file as an FMU component and then connecting all FMU inputs before starting the analysis:

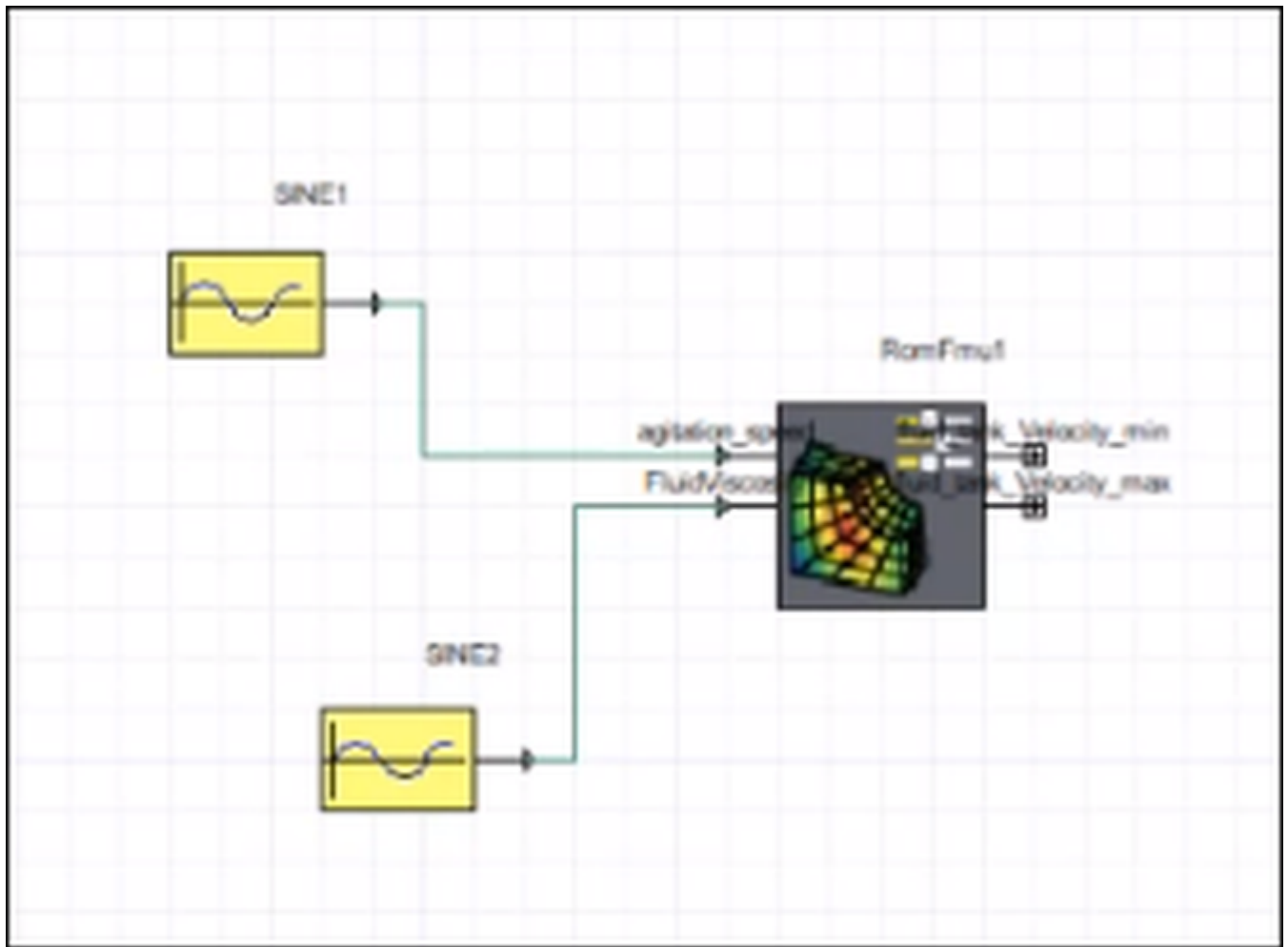
1. In Twin Builder, select **Twin Builder** → **SubCircuit** → **Add FMU Component**.

The **Select FMU model file** dialog box opens.

2. Navigate to and select the FMU 2.0 file for the 3D ROM.
3. Click **Open**.

The **Import FMU Model** dialog box opens.

4. Select the results in which you are interested.
5. Click **Import** and place the model.



6. To have the **ROM Viewer** launch during the simulation, set these properties for the FMU component:

- **TwinBuilder_LaunchViewer:** Set to **2** to launch the latest viewer or to **1** to launch the initial viewer. If set to **0**, the viewer does not launch.
- **TwinBuilder_SendingPeriod:** Set the time period in seconds at which the Twin Builder is to send the data to the viewer. The default value is **0**, which means data is sent at each simulation time instance.

Name	Value	Units	Description
UseStartValueInitialization	"true"		Use start values for input provided by <inputName>_start parameters at initialization
inlet_velocity_shell_start	0.0055		Start value of (inlet_velocity_shell) : 0.001m s ⁻¹ -0.01m s ⁻¹
inlet_temperature_shell_start	300		Start value of (inlet_temperature_shell) : 290K-310K
inlet_velocity_tube_start	0.003		Start value of (inlet_velocity_tube) : 0.001m s ⁻¹ -0.005m s ⁻¹
inlet_temperature_tube_start	350		Start value of (inlet_temperature_tube) : 330K-370K
TwinBuilder_SendingPeriod	0	s	Time period at which TwinBuilder to send the data to Static ROM Viewer for visualization purpose
TwinBuilder_LaunchViewer	2		Option for TwinBuilder to launch Static ROM Viewer for visualization purpose
inlet_velocity_shell	SINE1.VAL		0.001m s ⁻¹ -0.01m s ⁻¹
inlet_temperature_shell	SINE2.VAL		290K-310K
inlet_velocity_tube	SINE3.VAL		0.001m s ⁻¹ -0.005m s ⁻¹
inlet_temperature_tube	SINE4.VAL		330K-370K

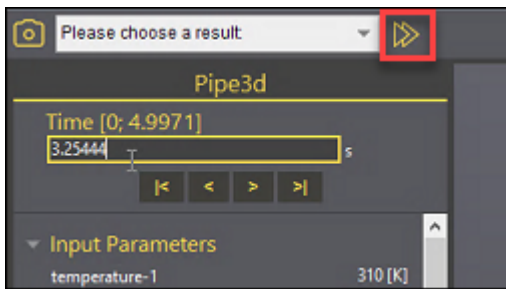
7. To have the **ROM Viewer** stay open once the simulation is finished, click **Edit Active Setup** in the ribbon and select the **Enable continue to solve** check box.
8. Connect all FMU inputs with the remainder of the system model.
9. Click **Analyze**.

Twin Builder launches the **ROM Viewer** that is set up in the properties for the FMU component.

Note:

Closing the **ROM Viewer** will not end the Twin Builder simulation. In Twin Builder's **Message Manager**, a warning indicates that the viewer has been closed or disconnected. After the simulation finishes, the Twin Builder simulation is kept in the **Continue** mode. You can extend the simulation by defining a new **Tend**. Stopping the Twin Builder simulation will close all open instances of the viewer.

If you set up the FMU component to launch the original viewer, the **Time** property and replay buttons appear below the title bar (project name). You must click the toolbar button to update the 3D scene to the most recent Twin Builder simulation data.



For **Time**, the unit is always seconds. You can enter the time instance from which you want to start replaying simulations. If you directly enter a value for **Time**, results for the closest time instance display. Using the replay buttons below this property, you can step through time instances. Unlike the latest viewer, the original viewer does not update automatically.

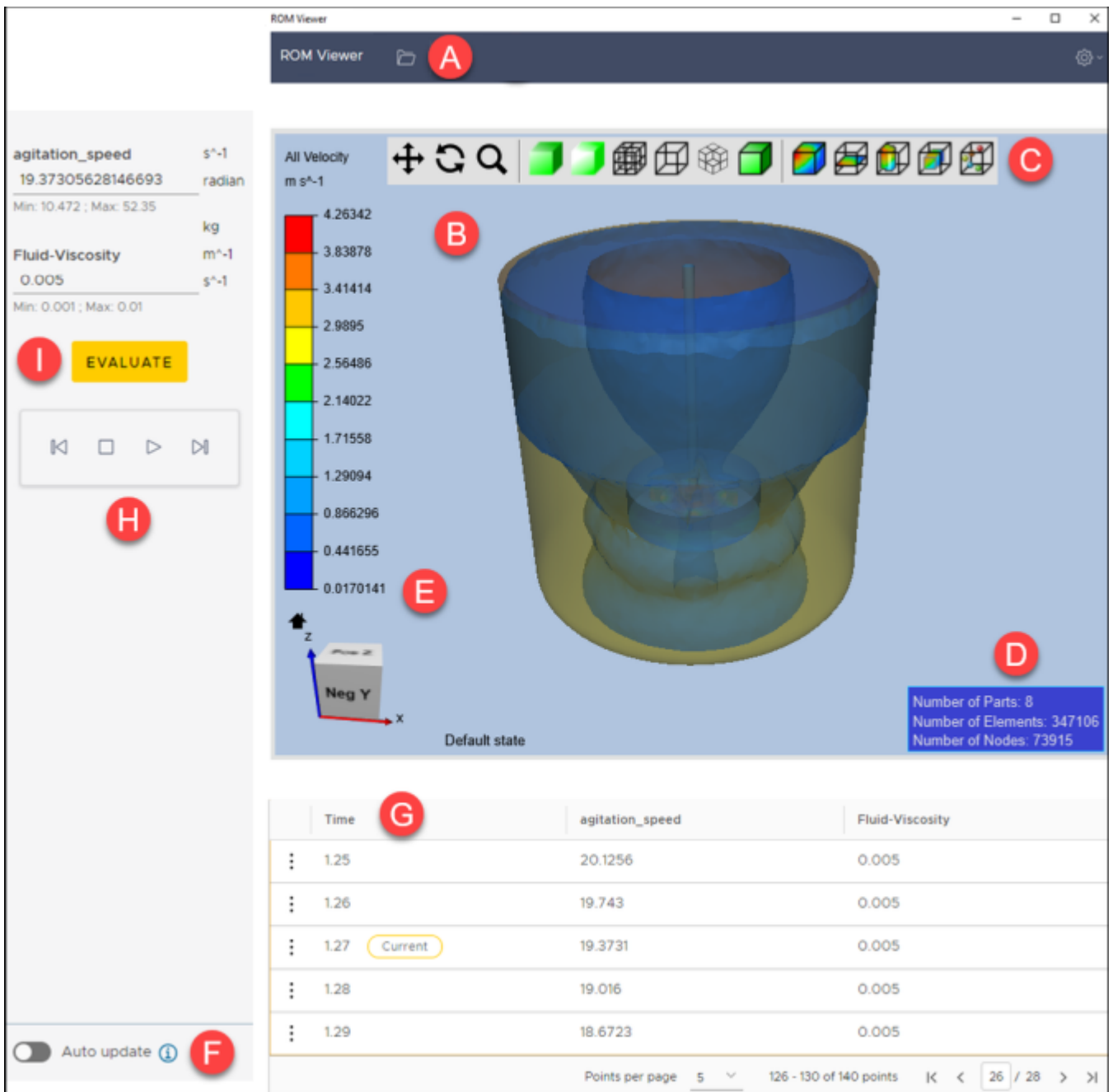
The next several sections describe the latest viewer.

Live Updating of the 3D Scene

The latest **ROM Viewer** includes live updating and replay capabilities, 3D scene interactions, geometry rendering controls, and postprocessing tools. The viewer starts in **Auto update** mode, which means that as Twin Builder sends new time instances, the viewer updates the 3D scene in real time. The table below the 3D scene displays the time instances that Twin Builder has sent.

If instances are sent at a speed faster than the viewer can evaluate them, the viewer will skip instances and stay synchronized with the Twin Builder simulation. All instances, including those skipped during the live update, are available for replay.

The following image shows an exploded view of the **ROM Viewer**. The subsequent table explains each area.



Area	
A	Button for opening ROM files. While you must not use this button in the ROM Viewer in Twin Builder, when the viewer is launched separately from Twin Builder, you can use it to open the following file types: ROMZ, ROMPJT, and ROMMSH.
B	3D scene provides these interactions: <ul style="list-style-type: none"> • Left-click an element to open the Element Info window. • To hide a region, press and hold the Shift key and then left-click the region. • To display all regions, right-click the 3D scene and select Show all parts. The context menu also has options for choosing how to render all parts.

Area	
	<ul style="list-style-type: none"> • To rotate an object, left-click it and move the mouse. • To set a rotation point, in the 3D scene, click where you want to set it and select Set a Rotation Point. • To pan, press and hold the Control key, left-click in the 3D scene, and then move the mouse • To zoom, click in the 3D scene and scroll the mouse wheel.
C	<p>Toolbar with three sections:</p> <ul style="list-style-type: none"> • Display controls for panning, rotating, and zooming in the 3D scene. • Geometry rendering controls for setting the draw style of all parts. Draw style choices are Surface, Transparent Surface, Lines, Outline, Surface Mesh, and Outline Surface. The current selection applies to all regions of the 3D scene, except for postprocessing regions, where opacity can be controlled on individual regions. • Postprocessing tools for configuring results, cutting planes, isosurfaces, isovolumes, and particle traces. Later sections describe how to use these tools.
D	Status box displays the number of points, elements, and nodes in the ROM.
E	Color bar for displaying results. For more information, see Configuring ROM Results (p. 258) .
F	Control for disabling and enabling automatic updates. For more information, see Disabling and Enabling Automatic Updates (p. 256) .
G	Table displaying the time instances that Twin Builder has sent.
H	Controls for replaying the parametric trajectory. For more information, see Replaying the Parametric Trajectory (p. 257) .
I	Analysis area for updating results in the 3D scene. After providing input parameter values, you click EVALUATE to update ROM results in the 3D scene.

Disabling and Enabling Automatic Updates

To pause automatic updates, you can disable **Auto update**:

- In the lower left corner of the window, clicking **Auto update** switches between disabling and enabling automatic updates.
- Clicking the play or next button in the left pane disables automatic updates.

When **Auto update** is disabled, the 3D scene no longer updates in real time. However, any new instances that Twin Builder sends to the viewer are still written to the table.

You might disable **Auto update** to:

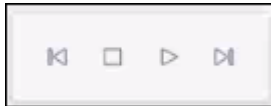
- Use postprocessing tools to change the setup of the 3D scene.

- Enter input parameters values in the left pane and then click **EVALUATE** to update the 3D scene.
- Replay the trajectory from a specific point. When you place the mouse cursor over a table row, clicking the three dots that appear to the left displays a menu with options for replaying the trajectory from this point or displaying the results for this point.

When you enable **Auto update** again, the 3D scene returns to updating in real time.

Replaying the Parametric Trajectory

The **ROM Viewer** has buttons for replaying the parametric trajectory.



In the table, the **Current** label marks the point that is currently shown in the viewer.

	Time	agitation_speed	Fluid-Viscosity
⋮	1.25	20.1256	0.005
⋮	1.26	19.743	0.005
⋮	1.27	19.3731	0.005
⋮	1.28	19.016	0.005
⋮	1.29	18.6723	0.005

During replay, the play button becomes a pause button, and clicking the next or previous button automatically pauses the replay. When paused, the play button allows you to resume the replay.

Clicking the stop button and then the play button restarts the replay from the beginning.

Note:

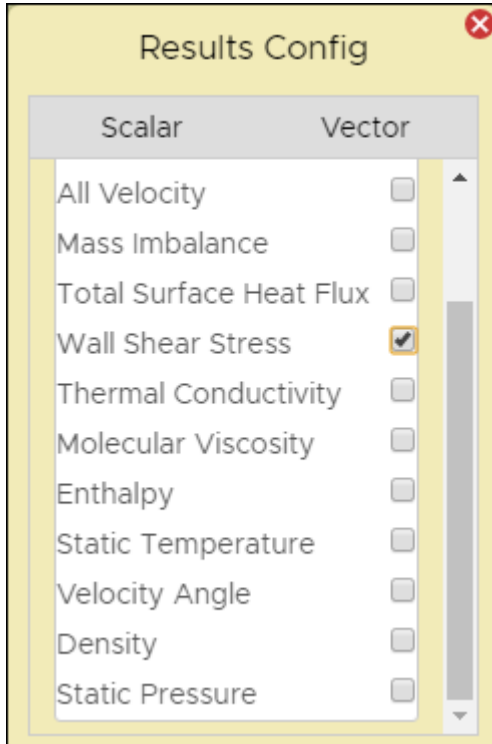
Enabling **Auto update** during replay automatically pauses the replay.

Additionally, when you place the mouse cursor over any table row, three dots appear on the left. When you click the three dots, you can select **Evaluate** or **Play from here** from the menu:

- Selecting **Evaluate** displays the results for this point.
- Selecting **Play from here** moves through the trajectory, starting from this point.

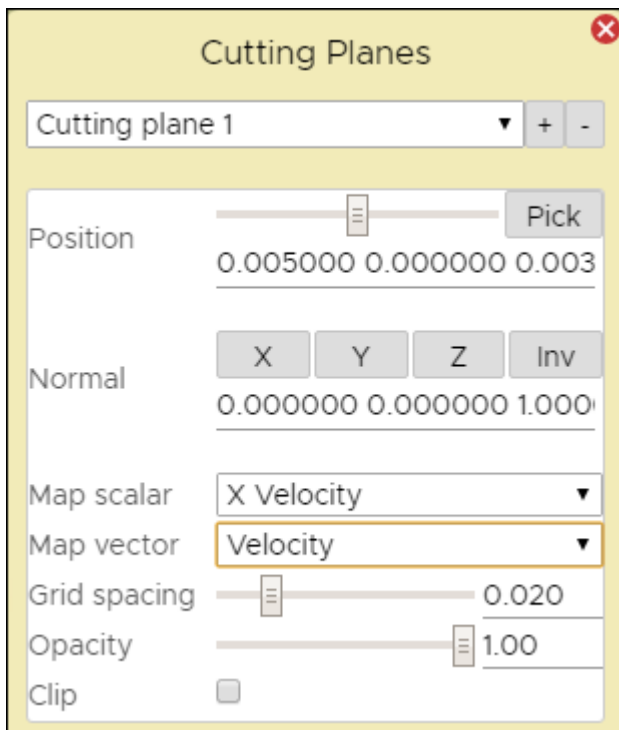
Configuring ROM Results

Clicking the **Results** button (🌈) in the postprocessing tools opens the **Results Config** window, where you can configure the results to display on all regions that are currently shown in the 3D scene. Scalar results are displayed using a color scale. Vector results are displayed using 3D arrows.



Configuring Cutting Planes

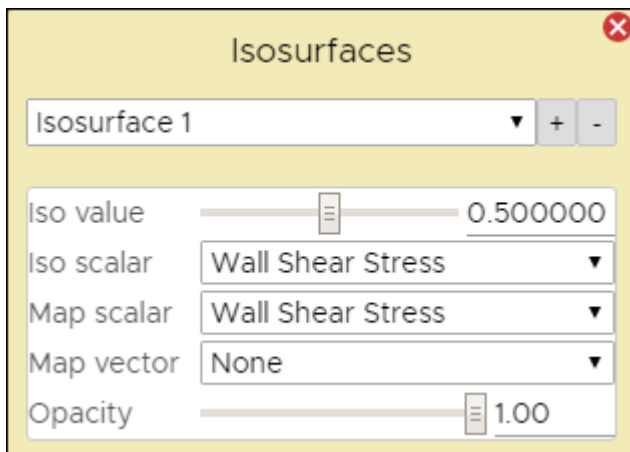
Clicking the **Cutting planes** button (✂️) in the postprocessing tools opens the **Cutting Planes** window, where you can add, delete, and configure cuttings planes in the 3D scene.



- The first set of controls is used to select, add, or delete a cutting plane in the 3D scene.
- **Position** provides both a slider and entry fields for moving the cutting plane position in the 3D scene. You can click **Pick** to select the cutting plane position in the 3D scene.
- **Normal** specifies the cutting plane orientation. You can use the preselected X, Y, and Z axes or enter axis values manually. You can click the **Inv** button to invert the cutting plane.
- **Map scalar** specifies the result to render on the cutting plane.
- **Map vector** specifies the vector result to render on the cutting plane. It is displayed using 3D arrows. To be able to select a vector, a field of vectors must be available in the ROM setup.
- **Grid spacing**, which is visible only when a vector result is selected, provides both a slider and an entry field for specifying the arrow density.
- **Opacity** provides both a slider and an entry field for specifying the opacity for the cutting plane.
- **Clip** indicates whether to hide one side of the geometry. This check box is cleared by default. The side of the geometry that is hidden can be controlled using the **Normal Inv** button.

Configuring Isosurfaces

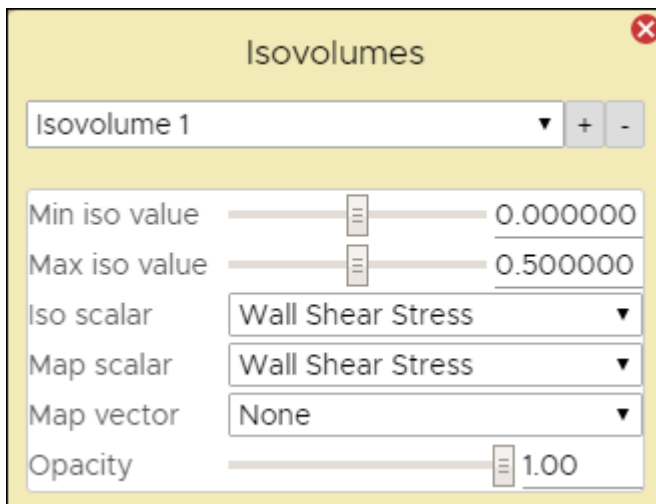
Clicking the **Isosurfaces** button (📐) in the postprocessing tools opens the **Isosurfaces** window, where you can configure isosurfaces. An isosurface is a three-dimension surface that represents points of a constant value (such as pressure, temperature, velocity, or density). To view the isosurface, some regions must be hidden, or the global draw style must be set to **Transparent Surface**, **Lines**, or **Outline**.



- The first set of controls is used to select, add, or delete an isosurface in the 3D scene.
- **Iso value** provides both a slider and an entry field for specifying the constant value used to generate the isosurface.
- **Iso scalar** specifies the result used for generating the isosurface.
- **Map scalar** specifies the result to render on the isosurface.
- **Map vector** specifies the vector result for displaying arrows on the isosurface. To be able to select a vector, a field of vectors must be available in the ROM setup.
- **Opacity** provides both a slider and an entry field for specifying the opacity for the isosurface.

Configuring Isovolumes

Clicking the Isovolumes button (📐) in the postprocessing tools opens the **Isovolumes** window, where you can configure isovolumes. An isovolume is a three-dimensional region whose nodes and elements are constrained to a constant interval range in a scalar field. To view the isovolume, some regions must be hidden, or the global draw style must be set to **Transparent Surface**, **Lines**, or **Outline**.

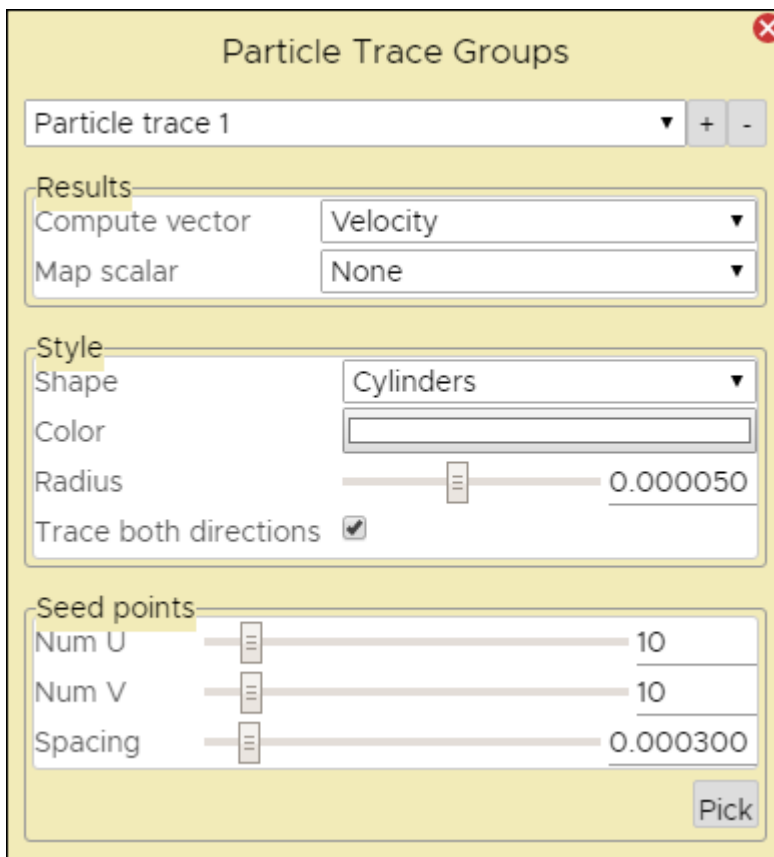


- The first set of controls is used to select, add, or delete an isovolume in the 3D scene.

- **Min iso value** provides both a slider and an entry field for specifying the minimum value for the interval range.
- **Max iso value** provides both a slider and an entry field for specifying the maximum value for the interval range.
- **Iso scalar** specifies the result used for generating the Isovolume.
- **Map scalar** specifies the result to render on the isovolume.
- **Map vector** specifies the vector result for displaying arrows in the isovolume. To be able to select a vector, a field of vectors must be available in the ROM setup.
- **Opacity** provides both a slider and entry field for specifying the opacity for the isovolume.

Configuring Trace Particles

Clicking the **Particle trances** button (📍) in the postprocessing tools opens the **Particle Trace Groups** window, where you can configure particle traces. A particle trace is a streamline for visualizing the path that particles without mass take when they are released into the flow.

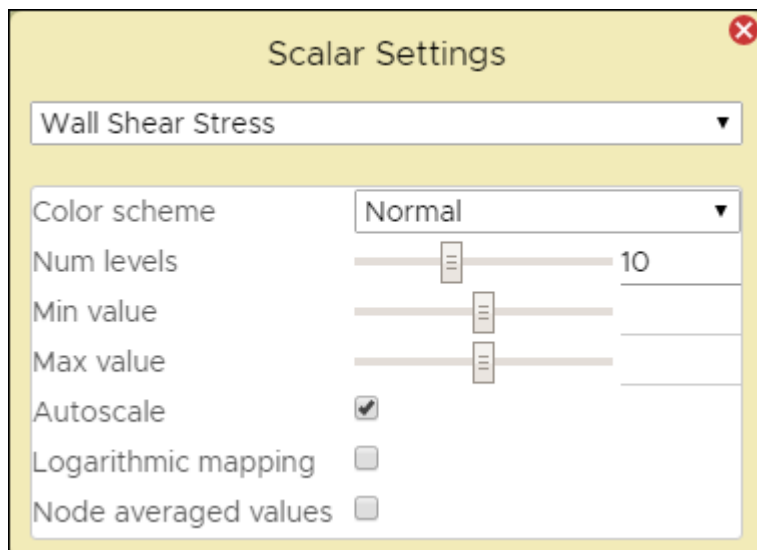


- The first set of controls is used to select, add, or delete a particle trace in the 3D scene.
- Under **Results**:
 - **Compute vector** specifies the vector result used to compute the particle trace.

- **Map scalar** specifies the result to render on the particle trace.
- Under **Style**:
 - **Shape** specifies the shape of the particle trace.
 - **Color** specifies the color of the particle trace.
 - **Radius** provides both a slider and an entry field for specifying the thickness of the particle trace.
 - **Trace both directions** indicates whether to draw the particle trace both upstream (backward) and downstream (forward) of the seed points. This check box is selected by default. If you clear it, the particle trace is drawn only downstream (forward).
- Under **Seed points**, which are created in a grid-like pattern:
 - **Num U** provides both a slider and an entry field for specifying the number of seed points in the U direction of the grid.
 - **Num V** provides both a slider and an entry field for specifying the number of seed points in the V direction of the grid.
 - **Spacing** provides both a slider and an entry field for specifying the particle trace density.
 - **Pick** allows you to select the center of the grid by clicking any surface of the model.

Configuring Scalar Settings

In the 3D scene, clicking the color bar opens the **Scalar Settings** window, where you can configure settings for the result to display.



- The first control is used to select the result to configure.
- **Color scheme** specifies the colors to display in the color bar.

- **Num levels** provides both a slider and an entry field for specifying the number of colors used in the color bar.
- **Min value** provides both a slider and an entry field for specifying the minimum value. While this value and lower values are still shown, they appear in gray in the color bar.
- **Max value** provides both a slider and an entry field for specifying the maximum value. While this value and higher values are still shown, they appear in gray in the color bar.
- **Autoscale** indicates whether to automatically scale the minimum and maximum values for the current result. This check box is selected by default. If you clear this check box, the minimum and maximum values will be the same for each evaluation.
- **Logarithmic mapping** indicates whether to use a logarithmic scale. This check box is cleared by default, which means a logarithmic mapping is not used.
- **Node averaged values** indicates whether to average elemental results to nodes or not. This setting has no effect on nodal results.

Configuring the Minimum Display Time for a Set of Results

Because the time required to make an update depends on your hardware, the ROM size, and what the 3D scene is to display, the minimum time between updates is not compressible. However, if the 3D scene is updating too fast for you to process a set of results before the next set displays, you can increase the minimum time before refreshing the display:

1. In the upper right corner of the **ROM Viewer**, click the gear icon.

A window opens for configuring settings.

2. For **Refresh Display**, increase the number of milliseconds between two result displays.

The default is 100 milliseconds.

Consuming a ROMZ File in the ROM Viewer

To consume a ROMZ file in the **ROM Viewer**, you must launch the viewer separately from Twin Builder.

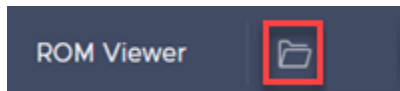
Note:

- If you do not have Twin Builder, you must request an installation package for the **ROM Viewer** from Ansys Technical Support. You can then launch the viewer from the root directory in which you install this package.
 - While this topic is about consuming a ROMZ file, when the **ROM Viewer** is launched separately from Twin Builder, you can also open a ROM project file (ROMPJT) or a ROM mesh file (ROMMSH) in the viewer. To open a ROM project file, the ROM mesh file must reside in the project directory. If the ROM mesh file is not found, an error displays, indicating that you must move or copy the ROM mesh file to this directory.
-

1. To start the **ROM Viewer** separately from Twin Builder, locate and double-click the file `ROMViewer.exe`.
 - If Ansys Electronics Desktop is installed, this file is in `C:\Program Files\AnsysEM\AnsysEMVersion\Win64\common\ROMViewer`.
 - If the **ROM Viewer** package is installed, this file is in the root directory.

The **ROM Viewer** starts. When launched separately from Twin Builder, the viewer does not display the live updating and replay capabilities that are available in Twin Builder.

2. In the header of the **ROM Viewer**, click the button for opening ROM files.



3. Navigate to and select the ROMZ file that you want to consume.
4. Click **Open**.

The **ROM Viewer** opens the ROMZ file, providing these features:

- 3D scene interactions
- Three-section toolbar
- Status bar
- Color bar for displaying results
- Analysis area

To update ROM results in the 3D scene, you provide input parameter values and click **EVALUATE**. For additional information, review the relevant viewer topics in [Consuming an FMU 2.0 File in Twin Builder](#) (p. 251).

Consuming a ROMZ File in Fluent

You can consume the ROMZ file in Fluent in Workbench by importing it. For more information, see [Reduced Order Model \(ROM\) Evaluation in Fluent](#) in the *Fluent User's Guide*.

Note:

You cannot import a ROMZ file into the Fluent system used to produce the 3D ROM in the same instance of Workbench.

Analyzing and Troubleshooting ROM Production

You use ROM snapshot files and the ROM Builder log file to analyze and troubleshoot ROM production:
[ROM Snapshot Files](#)

[ROM Mesh Files](#)

[ROM Builder Log File](#)

[ROM Builder Log File Metrics](#)

ROM Snapshot Files

A ROM snapshot file (ROMSNP) contains multiple simulation results, such as **temperature** and **velocity**, for a single design point. These results are distributed between different regions, such as **inlet** and **fluid**.

When creating a ROM in the ROM Builder, you update the **Design of Experiments (3D ROM)** cell in the **3D ROM** system, which generates a ROM snapshot file for each design point.

You can perform many different operations on snapshot files, including:

[Viewing Statuses of ROM Snapshot Files](#)

[Displaying ROM Snapshot File Information](#)

[Setting Specific ROM Snapshot Files for Design Points](#)

[Clearing Unused ROM Snapshot Files](#)

Viewing Statuses of ROM Snapshot Files

In the **Table** pane for the **Design of Experiments (3D ROM)** cell, the last column displays statuses of ROM snapshot files. The column name indicates the solver system. For the supplied example, the column name is **Fluid Flow (Fluent) ROM Snapshot**.

In the **ROM Builder** cell, this same column is also available in refinement and verification point tables. Placing the mouse cursor over a cell in the column displays the name of the snapshot file.

Descriptions follow for each status icon that can display in the snapshot column:

- Up-to-date (✓): The design point has a snapshot file associated with it, and no issues with this file were detected.
- Update required (⚡): The design point is not yet solved and must be updated. Updating the design point will generate a new snapshot file.
- Update failed (✖): The design point has been calculated or has an imported snapshot file associated with it but the file is not useable.
- Attention required (🔔): The design point has editable outputs but does not have a snapshot file associated with it. To continue, you must select a snapshot file for this design point.

When an information icon (i) appears to the right of the status icon, you can click it to view remarks, errors, or guidance.

When a snapshot file is invalid or missing, the DOE cell cannot be updated. To continue, you must do one of the following:

- Set a valid snapshot file manually, which is described in [Manually Adding Design Points and Setting Snapshot Files](#) (p. 268)

- Calculate the design point to generate a new snapshot file
- Remove the design point from the DOE

The same is true if there is an invalid or missing snapshot file for a refinement point or validation point in the **ROM Builder** cell.

- Invalid DOE design points and refinement points prevent building the ROM.
- Invalid verification points prevent building the [goodness of fit](#) (p. 277).

Displaying ROM Snapshot File Information

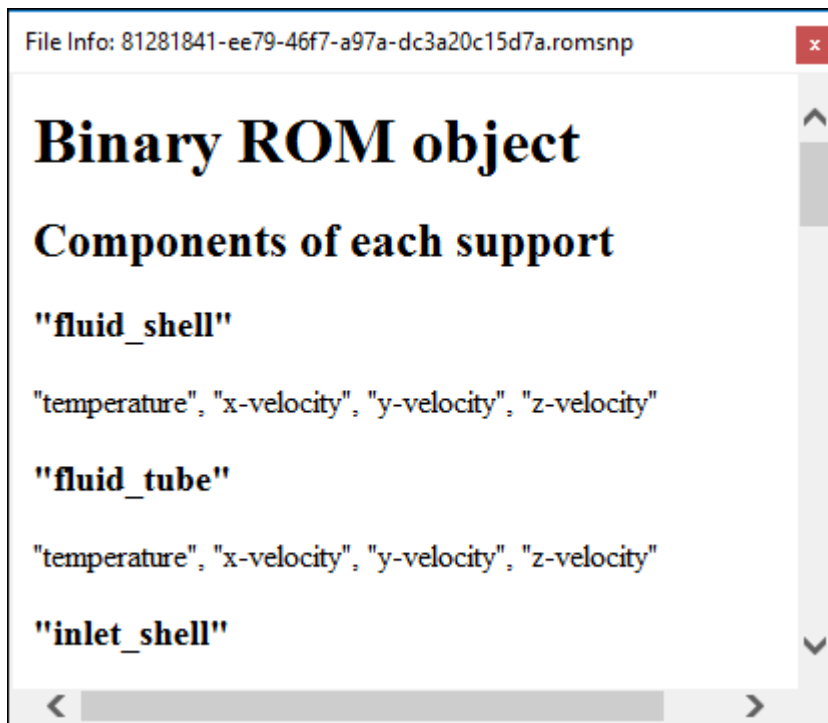
In the **Table** pane for the **ROM Builder** cell, you can display information about a ROM snapshot file by right-clicking the cell in the ROM snapshot column and selecting **Display File Info**.

Additionally, you can display information about a ROM snapshot file from the **Files** pane of the **Project Schematic**:

1. If the **Files** pane is not visible, select **View** → **Files**.
2. In the **Name** column, right-click the cell with the snapshot file in which you are interested and select **Display File Info**.

22		Report.htm	B6	9 KB	Default File
23	ROM	52deacc9-9ef6-4994-9fbf-b42cbb7e0519.romsnp	Global	3 MB	ROM Snapshot File
24	ROM	81281841-ee79-46f7-a97a-dc3a20c15d7a.romsnp	Global	3 MB	ROM Snapshot File
25	F	Solution.trn			JENT Solution Transc
26	F	Solution.trn			JENT Solution Transc
27	F	SolutionMonitor.gz			JENT Residual File
28	F	Solution.trn			JENT Solution Transc
29	ROM	4fcc8525-2ee1-4897-a4d3-46aa3d5703d9.romsnp			ROM Snapshot File
30	F	SolutionMonitor.gz	B5	8 KB	FLUENT Residual File
31	F	Solution.trn		18 KB	FLUENT Solution Transc
32	ROM	26adecee-d5d0-4ced-88d3-fa4abd9d64f7.romsnp	Global	3 MB	ROM Snapshot File

The file information displays results for each variable and zone that was included in the ROM.



When you finish reviewing this information, you can close the file.

Setting Specific ROM Snapshot Files for Design Points

In a **3D ROM** system, the **Design of Experiments (3D ROM)** cell is like the DOE cells in other DesignXplorer systems. If you set the **Design of Experiments Type** property for the DOE to **Custom** or **Custom + Sampling**, you can edit existing input parameter values for design points. You can also manually add design points and [import \(p. 98\)](#) and [copy \(p. 99\)](#) design points.

However, for added design points, ROM snapshot files must be set. Multiple **3D ROM** systems can use the same snapshot files.

To add design points and set snapshot files, you can use these methods:

[Manually Adding Design Points and Setting Snapshot Files](#)

[Exporting and Importing ROM Snapshot Archive Files](#)

[Importing Design Points and Snapshot File Settings from CSV Files](#)

In certain situations, these methods ask how you would like to proceed:

- Before copying or importing design points into a DOE, DesignXplorer parses and validates the data. For more information, see the parsing and validation information in [Copying Design Points \(p. 99\)](#).
- If you set a snapshot file that is in the ROM production folder for a different project and this same file does not already exist in the ROM production folder for the current project, a dialog box opens, asking whether you want to move or copy the file.
- In a case where a snapshot file with the same name already exists in the ROM production folder, a dialog box opens, asking whether you want to use the existing file or rename the

new file. Before deciding which action to take, you can click **Show Details** and compare information for the two files. If you select rename, the imported file is copied and assigned a new name that corresponds to the current name plus a suffix with time stamp information.

Any time that you edit design points or refinement points, you must update the ROM.

Caution:

If you edit verification points without defining output parameter values and setting snapshot files, you must update these points to get the new goodness of fit. If you add verification points that are up-to-date, the goodness of fit is updated automatically.

Manually Adding Design Points and Setting Snapshot Files

You can always manually add design points to a custom DOE. However, for these design points, you must also manually set snapshot files.

For each design point that you manually add:

1. Right-click the row and select **Set Output Values as Editable**.

If you select multiple rows, output parameter values for all selected rows are set as editable. If you want to make output parameter values for all rows editable, you would select **Set All Output Values as Editable**. For more information, see [Editable Output Parameter Values](#) (p. 314).

2. In the **Table** pane, right-click any editable output parameter value in the row and select **Set Snapshot File**.
3. In the dialog box that opens, select the appropriate snapshot file and click **Open**.

Exporting and Importing ROM Snapshot Archive Files

A ROM snapshot archive file (SNPZ) is a compressed package containing all data in the DOE cell of a **3D ROM** system. This package includes not only a CSV file with design point data but also all snapshot files in the ROM production folder. You can export the data in the DOE cell of one **3D ROM** system to a snapshot archive file and then import this data into the DOE cell of another **3D ROM** system.

To export DOE data and all snapshots to a snapshot archive file:

1. In the **Table** pane for the **Design of Experiments (3D ROM)** cell, right-click and select **Export as Snapshot Archive**.
2. In the dialog box that opens, specify a directory location and file name and then click **Save**.

To import DOE data and snapshots from a snapshot archive file:

1. In the **Table** pane for the **Design of Experiments (3D ROM)** cell, right-click and select **Import Design Points and Snapshot Files** → **Browse**.

2. In the dialog box that opens, navigate to the desired snapshot archive file and click **Open**.

A check for consistency is made against the first valid snapshot that is imported.

Note:

You can use a software tool like 7-Zip to open and modify a snapshot archive file. If you want, you can save the modified file as a ZIP file. DesignXplorer supports importing snapshot archive files with either SNPZ or ZIP extensions.

Importing Design Points and Snapshot File Settings from CSV Files

In CSV files that contain design points to import, for each design point, you can include the name of the snapshot file to set to avoid having to manually set it. The snapshot files referenced must be in the ROM production folder of the project into which you are importing the design points.

1. Ensure that the snapshot files referenced in the CSV file are copied into the ROM production folder of the project into which to import design points.

To easily find this directory, in the **Table** pane for the DOE cell, you can right-click a cell in the ROM snapshot column and select **Open Containing Folder**.

2. Proceed with the import of the CSV file. For more information, see [Importing Data from a CSV File \(p. 316\)](#).

Clearing Unused ROM Snapshot Files

During ROM production, you can clear unused ROM snapshot files to free disk space. When you right-click in the background of the **Project Schematic** and select **Clear All Unused ROM Snapshots** from the context menu, DesignXplorer removes all snapshot files that are not associated with any of the points in the following tables:

- Design points in the **Design of Experiments (3D ROM)** cell
- Refinement points in the **ROM Builder** cell
- Verification points in the **ROM Builder** cell
- Design points in the **Parameter Set** bar

DesignXplorer keeps any snapshot file that was generated with the same inputs as a point in any of these tables. The tables listed in the first three bullets have ROM snapshot columns, so it's clear which snapshot files are kept. While the table listed in the last bullet does not have a ROM snapshot column, DesignXplorer still keeps snapshot files associated with its points.

The **Clear All Unused ROM Snapshots** option is available on the context menu after you set up a solver for ROM production. It is disabled if there are no unused snapshot files to remove.

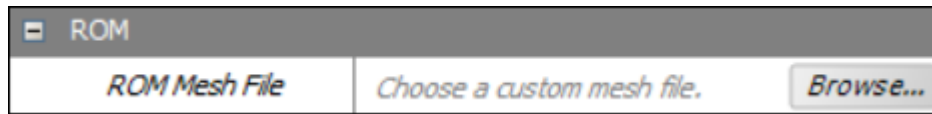
Note:

The **Retain Data** option does not modify the behavior of this functionality.

ROM Mesh Files

Rather than having DesignXplorer automatically generate a ROM mesh file (ROMMSH), you can set a user-generated ROM mesh file for when you export a ROM as an FMU 2.0 file or ROMZ file.

When DesignXplorer advanced options are shown, **ROM Mesh File** is visible in the **Properties** pane for the **Design of Experiments (3D ROM)** cell.



The mesh file must reside in the ROM production folder for the current project. If you select a mesh file in a different project and this same file does not already exist in the current project, a dialog box opens, asking whether you want to move or copy the file. In a case where a mesh file with the same name already exists in the ROM production folder, the dialog box indicates that the existing file will be overwritten.

Caution:

Ensure that the custom mesh file that you select is compatible with the ROM.

The mesh file must be in a "processed" state. If it is not, it is immediately processed.

To stop using the custom mesh file, you would clear the **ROM Mesh File** property. While DesignXplorer does not remove the file from the project, it unregisters the file so that you can remove it manually.

Tip:

If the mesh file already exists in the ROM production folder for the current project, you can directly enter the file name in **ROM Mesh File**. If the mesh file is in the ROM production folder for a different project, you must enter the full path.

ROM Builder Log File

After you configure and update the **ROM Builder** cell, you can open the ROM Builder log file to analyze and troubleshoot ROM production.

1. In the **Outline** pane for the **ROM Builder** cell, select **ROM Builder**.

When DesignXplorer advanced options are shown, **ROM Builder.log** is visible in the **Properties** pane under **Log File**.

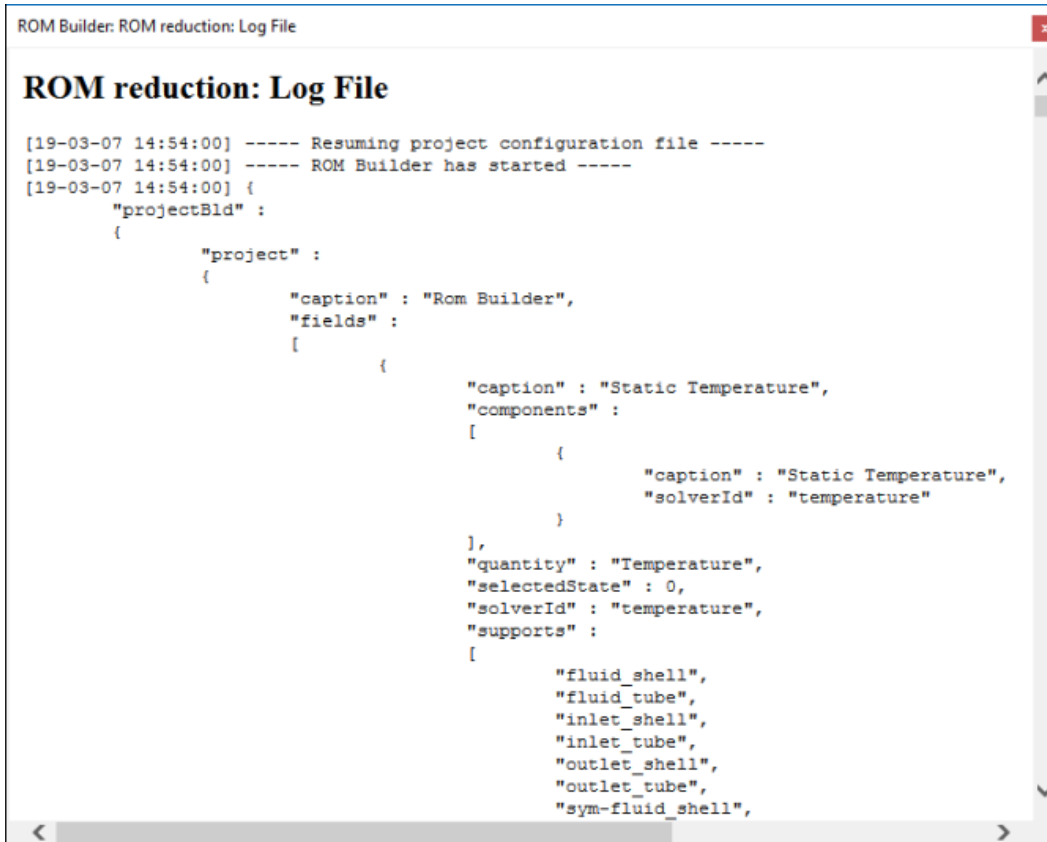
2. Click **ROM Builder.log**.

Tip:

When advanced options are not shown, in the **Files** pane for the **Project Schematic**, you can right-click `Rom Builder.log` and select **Open Containing Folder** to go

to the location where this log file is stored. You can then double-click the file to open it.

For explanations of this file's content, see the next topic.



ROM Builder Log File Metrics

The ROM Builder uses Singular Vector Decomposition (SVD) to build the ROM.

When **Display Level Log File** is set to **Medium**, the ROM Builder writes metric errors about SVD.

- When **Construction Type** is set to **Global**, the ROM Builder computes one SVD per field.
- When **Construction Type** is set to **Local**, the ROM Builder computes one SVD per field and per support.

A table of SVD errors follows.

Nb Modes	Rel. Proj. Err.	Abs. Proj. Err.	Rel. Proj. Err. RMS	Abs. Proj. Err. RMS	LOO Rel. Proj. Err.	LOO Abs. Proj. Err.	LOO Rel. Proj. Err. RMS	LOO Abs. Proj. Err. RMS
1								
2								
...								

For a given row, the errors shown in the table correspond to errors obtained by using the M first modes of the SVD, with M equal to the number of modes. See the **Nb Modes** column.

The following notation is used in the mathematical representations in descriptions for these criteria:

- Y_i corresponds to the vector of reference values of the i -th snapshot.
- Y_{proj_i} corresponds to the vector of values of the projection of the i -th snapshot onto the subspace based on the first modes of the SVD.
- $\hat{Y}_i(j)$ corresponds to the approximated value of the ROM on the j -th entities of the i -th snapshot.
- N is the number of snapshots.
- n is the number of entities (size of the support) of the snapshot.

Rel. Proj. Err

Computes the ratio between the average of projection error on snapshots and the average of the norm of snapshots:

$$\frac{\sum_{i=1}^N \|Y_i - Y_{proj_i}\|_2}{\sum_{i=1}^N \|Y_i\|_2}$$

Abs. Proj. Err

Computes the average of projection error on snapshots:

$$\frac{1}{N} \sum_{i=1}^N \|Y_i - Y_{proj_i}\|_2$$

Rel. Proj. Err. RMS

Computes the ratio between root mean square of the projection error on snapshots and the root mean square of the norm of snapshots:

$$\sqrt{\frac{\sum_{i=1}^N \|Y_i - Y_{proj_i}\|_2^2}{\sum_{i=1}^N \|Y_i\|_2^2}}$$

NB: This metric corresponds to the one exposed in the DesignXplorer interface to control the level of ROM accuracy when **Construction Type** is set to **Fixed Accuracy**. The number of modes for building the ROM is selected such that the associated **Rel. Proj. Err. RMS** is less than the defined **Maximum Relative Error**.

Abs. Proj. Err. RMS

Computes the root mean square of the projection error on snapshots:

$$\sqrt{\frac{1}{N} \sum_{i=1}^N \|Y_i - Y_{proj_i}\|_2^2}$$

The *LOO* (Leave-one-out) computation metrics are similar to the previous metrics except for one difference: the projection of the *i*-th snapshot is computed by using SVD based only on all snapshots except the *i*-th snapshot. This technique allows you to see the stability of the SVD and the accuracy of the projection on other points than those used by the SVD.

The number of selected modes used to build the ROM is specified below the table of SVD errors. For example, it might indicate: **Number of selected modes = 3**.

A result example follows:

```

Component = temperature; Support = wall; Support size = 438
Component = temperature; Support = symmetry; Support size = 274
Component = temperature; Support = solid; Support size = 2846
Component = temperature; Support = outlet; Support size = 20
Component = temperature; Support = interior-solid; Support size = 5240
Component = temperature; Support = inlet-2; Support size = 32
Component = temperature; Support = inlet-1; Support size = 32
No Modes | Rel. Proj. Err. | Abs. Proj. Err. | Rel. Proj. Err. RMS | Abs. Proj. Err. RMS | LOO Rel. Proj. Err. | LOO Abs. Proj. Err. | LOO Rel. Proj. Err. RMS | LOO Abs. Proj. Err. RMS
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
1 | 5.127e-02 | 1.569e+03 | 4.015e-02 | 1.846e+03 | 5.231e-02 | 1.603e+03 | 6.133e-02 | 1.883e+03
2 | 1.427e-02 | 4.365e+02 | 1.708e-02 | 5.246e+02 | 1.601e-02 | 4.595e+02 | 1.907e-02 | 5.551e+02
3 | 8.128e+03 | 2.481e+02 | 9.422e+03 | 2.955e+02 | 8.876e+03 | 2.710e+02 | 1.061e+02 | 3.259e+02
4 | 5.301e-03 | 1.422e+02 | 6.324e-03 | 1.942e+02 | 6.036e-03 | 1.847e+02 | 7.379e-03 | 2.266e+02
    
```

When **Display Level Log File** is set to **High**, the ROM Builder writes additional metrics about ROM approximation errors.

One table of ROM errors is shown per support and per field. Each row displays metric errors per snapshot on its entities.

Snapshot	Err. Nrm2	Rel. Err. Nrm2	Err. NrmInf	Rel. Err. NrmInf	DOF error 1st Quartile	DOF error Median	DOF error 3rd Quartile	DOF error 95th Percentile
1								
2								
...								

Err. Nrm2

Computes the root of the sum of squared errors on entities of the snapshot:

$$\sqrt{\sum_{j=1}^n (Y_i(j) - \hat{Y}_i(j))^2}$$

Rel. Err. Nrm2

Computes the root of the sum of squared errors on entities of the snapshot divided by the norm of the snapshot:

$$\frac{\sqrt{\sum_{j=1}^n (Y_i(j) - \hat{Y}_i(j))^2}}{\|Y_i\|_2}$$

Err. NrmInf

Computes the maximum absolute error measured on the entities of the snapshot:

$$\max_{j=1..n} |Y_i(j) - \hat{Y}_i(j)|$$

Rel. Err. NrmInf

Computes the maximum absolute error measured on the entities of the snapshot divided by the maximum absolute value of the entities of the snapshot:

$$\frac{\max_{j=1..n} |Y_i(j) - \hat{Y}_i(j)|}{\max_{j=1..n} |Y_i(j)|}$$

DOF error 1st Quartile

Upper Bound Error of 25% of the entities of the snapshot

DOF error Median

Upper Bound Error of 50% of the entities of the snapshot

DOF error 3rd Quartile

Upper Bound Error of 75% of the entities of the snapshot

DOF error 95th Percentile

Upper Bound Error of 95% of the entities of the snapshot

A result example follows:

Component = temperature; Support = wall; Support size = 498									
ROM errors on snapshots used to build the ROM:									
snapshot	Err. Nrm2	Rel. Err. Nrm2	Err. NrmInf	Rel. Err. NrmInf	DOF error 1st Quartile	DOF error Median	DOF error 3rd Quartile	DOF error 95th Percentile	
1	4.898e+01	4.989e-03	1.086e+01	3.170e-02	1.217e+00	1.442e+00	1.586e+00	4.469e+00	
2	2.530e+01	3.225e-03	7.927e+00	2.127e-02	1.443e-01	4.877e-01	1.082e+00	2.342e+00	
3	2.883e+01	4.279e-03	6.174e+00	1.882e-02	7.997e-01	8.013e-01	1.313e+00	2.362e+00	

To complete this table, DesignXplorer computes other statistics:

	Err. Nrm2	Rel. Err. Nrm2	Err. NrmInf	Rel. Err. NrmInf
Minimum				
Maximum				
Average				
1st quart				
median				
3rd quart				

Minimum

Corresponds, for a given metric, to the minimum value obtained on all snapshots

Maximum

Corresponds, for a given metric, to the maximum value obtained on all snapshots

Average

Corresponds, for a given metric, to the average value obtained on all snapshots

1st quart

Corresponds, for a given metric, to the upper bound of 25% of all snapshots

median

Corresponds, for a given metric, to the upper bound of 50% of all snapshots

3rd quart

Corresponds, for a given metric, to the upper bound of 75% of all snapshots

A result example follows:

minimum	2.190e+00	3.242e-04	1.110e+00	3.302e-03
maximum	1.186e+02	1.622e-02	2.069e+01	6.156e-02
average	3.302e+01	4.585e-03	8.025e+00	2.327e-02
1st quart	1.371e+01	2.009e-03	3.488e+00	9.477e-03
median	2.827e+01	3.613e-03	6.955e+00	2.127e-02
3rd quart	4.601e+01	6.187e-03	1.188e+01	3.317e-02

Reference values on DOFs of snapshots are used to build the ROM. The following table allows you to see quickly statistics associated to reference values of entities of each snapshot.

Snapshot	Min. Value	Max. Value	1st Quartile	Median	3rd Quartile	Lower Limit	Upper Limit	Outliers
1								
2								
...								

Min. Value

Minimum value of entities of the snapshot

Max. Value

Maximum value of entities of the snapshot

Mean Value

Mean value of entities of the snapshot

1st Quartile

Upper bound of 25% of entities of the snapshot (=Q1)

Median

Upper bound of 50% of entities of the snapshot

3rd Quartile

Upper bound of 75% of entities of the snapshot (=Q3)

Lower Limit

Equal to $Q1 - 1.5 * IQR$ with $IQR = Q3 - Q1$

Upper Limit

Equal to $Q3 + 1.5 * IQR$

Lower and upper limits correspond to the "inner fences" that mark off the "reasonable" values from the outlier values. An outlier is a value that is distant from other values. In some cases, an outlier can correspond to an erratic point and reveal a real problem on the snapshot.

Outliers

Indicates when the **Min. Value** is less than the **Lower Limit** or when the **Max. Value** is greater than the **Upper Limit**.

A result example follows:

Component = temperature; Support = wall; Support size = 455										
ROM errors on snapshots used to build the ROM:										
snapshot	Err. Max2	Rel. Err. Max2	Err. MinInf	Rel. Err. MinInf	DOF error 1st Quartile	DOF error Median	DOF error 3rd Quartile	DOF error 95th Percentile		
1	4.838e+01	6.989e-03	1.086e+01	3.170e-02	1.217e+00	1.442e+00	1.986e+00			4.469e+00
2	2.530e+01	3.225e-03	7.927e+00	2.127e-02	1.443e+01	4.877e-01	1.082e+00			2.342e+00
3	2.885e+01	4.279e-03	4.174e+00	1.882e-02	7.537e-01	8.013e-01	1.313e+00			2.342e+00

Quality Metrics for ROMs

Quality metrics for ROMs include [goodness of fit \(p. 277\)](#) and [verification points \(p. 279\)](#). Used together, they enable you to assess and improve upon the quality of your ROM.

If goodness of fit is poor, you can enrich your ROM and improve its accuracy by manually adding [refinement points \(p. 126\)](#). Adding refinement points to a **ROM Builder** cell is equivalent to adding design points to a **Design of Experiments** cell. Both design points and refinement points are taken into account when you update the ROM.

Goodness of Fit for ROMs

The design points and any refinement points that are used to build the ROM are collectively called *learning points*. Goodness of fit is calculated for both design points and refinement points. Calculations for the goodness of fit compare predicted ROM values with observed values for ROM snapshot files.

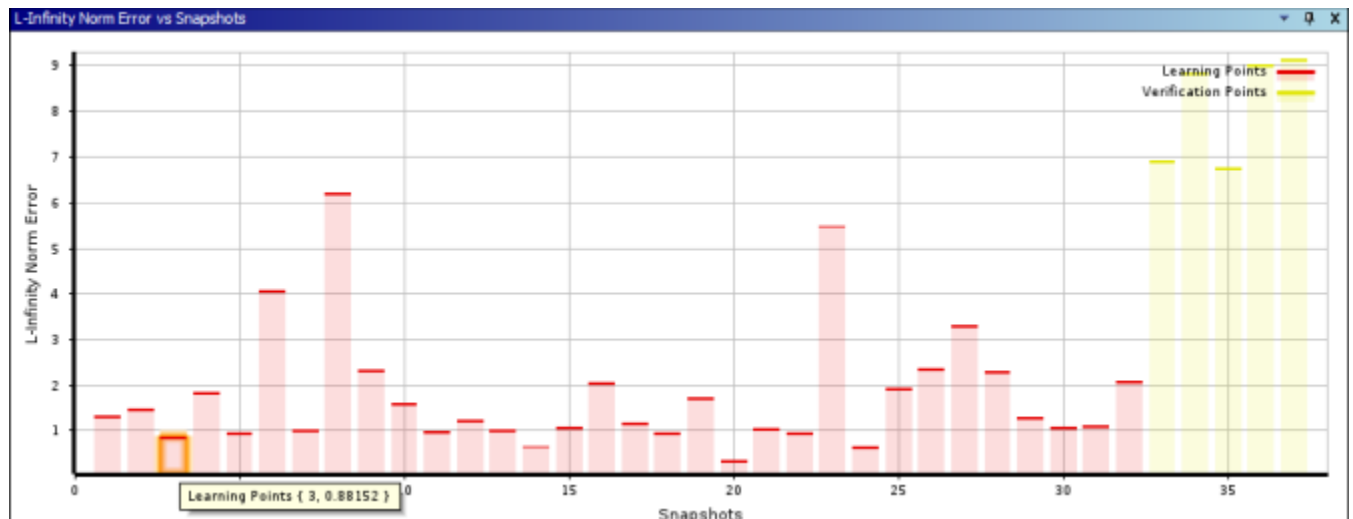
To view goodness of fit for the output parameters in a ROM, in the **Outline** pane for the **ROM Builder** cell, select **Goodness Of Fit**.

In both the **Table** and **Chart** panes, the design points and refinement points are grouped together and labeled as **Learning Points**.

- The **Table** pane displays the differences (errors) between the real solve solutions and the ROM solutions, calculated at all nodes and all design points (average, maximum, and so on), for each field (variable) for the selected region (zone). Each parameter is rated on how close it comes to the ideal value for each goodness of fit metric. The rating is indicated by the number of gold stars or red crosses next to the parameter. The worst rating is three red crosses. The best rating is three gold stars.

Table of Schematic C3: ROM Builder			
	A	B	C
1		Static Temperature	Velocity
2	Global Relative Norm2 Error (Best Value = 0%)		
3	Learning Points	★★★ 0.044312	★★★ 0.041437
4	Verification Points	★★ 0.49871	★★★ 0.1023
5	Average Relative Norm2 Error (Best Value = 0%)		
6	Learning Points	★★★ 0.042645	★★★ 0.042392
7	Verification Points	★★ 0.46457	★★★ 0.10783
8	Maximum Relative Norm2 Error (Best Value = 0%)		
9	Learning Points	★★★ 0.069061	★★★ 0.060002
10	Verification Points	★★ 0.71993	★★★ 0.14143
11	Average NormInf Error (Best Value = 0)		
12	Learning Points	1.7765	2.3107E-05
13	Verification Points	8.1367	4.0771E-05
14	Maximum NormInf Error (Best Value = 0)		
15	Learning Points	6.2163	5.3588E-05
16	Verification Points	9.143	5.9895E-05

- The **Chart** pane displays the error that is measured for each snapshot. You can choose to turn on and off the display of learning points and verification points in the chart properties. You can display the chart as either a bar chart of the error for each snapshot or a cumulative distribution. Depending on the mode, placing the mouse cursor over a particular point displays the error value or cumulative error percentage.



Goodness of fit differs for each field. In the **Properties** pane, **Chart** and **General** categories indicate what to display in the chart and table.

Under **Chart**, you set properties for the chart:

- **Show Verification Points:** Indicates whether to show or hide verification points.
- **Show Learning Points:** Indicates whether to show or hide learning points.
- **Mode:** Indicates the type of error to graph. Choices are:
 - **Error per Snapshot:** Allows you to quickly see if the level of error is uniform for all snapshots and which snapshots have the highest and lowest errors.
 - **Cumulative Distribution Error:** Allows you to quickly see what percentage of snapshots have an error smaller than a given value.
- **Error Type:** Indicates the type of error to display. Choices are **L-Infinity Norm Error** and **Relative L2-Norm Error (%)**. The first choice displays the absolute error. The other choice displays the normalized error. For more information, see [ROM Goodness of Fit Criteria \(p. 279\)](#).
- **Field to Show:** Indicates the field to display.

Under **General**, **Region** indicates the region of interest to display in the table. You can select **All Regions** or a particular region. In the table, you see error metrics for each field included in the ROM.

If you want to add a new goodness-of-fit object, right-click **Quality** and select **Insert Goodness of Fit**. You can also right-click an existing goodness-of-fit object and then copy and paste, duplicate, or delete it. After duplicating a goodness-of-fit object, you can modify its properties, such as the region to display in the table and the field to show in the chart.

Note:

During computation of the goodness of fit, you can interrupt the update of the **ROM Builder** cell if the computation is taking too long. You are still able to [export the](#)

ROM (p. 251). You can then update the **ROM Builder** cell later to compute the goodness of fit.

ROM Goodness of Fit Criteria

The **Table** and **Chart** panes display goodness-of-fit metrics for each output parameter.

Two types of errors are calculated for the points taken into account in the construction of the response surface: **L-Infinity Norm Error** and **Relative L2-Norm Error (%)**. The mathematical representations for these criteria use the following notation:

Y_i : i-th snapshot

$Y_{i,j}$: j-th entity value of the i-th snapshot

$\widehat{Y}_{i,j}$: ROM approximation of j-th entity value of the i-th snapshot

S: number of snapshots

N: size of snapshot

Metric	Formula	Definition	Metric Type
$NormInfError(Y_i)$	$\max_{j=1..N} Y_{i,j} - \widehat{Y}_{i,j} $	Local	Absolute
$Norm2(Y_i)$	$\sqrt{\sum_{j=1}^N Y_{i,j}^2}$	Global	Absolute
$Norm2Error(Y_i)$	$\sqrt{\sum_{j=1}^N (Y_{i,j} - \widehat{Y}_{i,j})^2}$	Global	Absolute
$RelativeNorm2Error(Y_i)$	$\frac{Norm2Error(Y_i)}{\frac{1}{S} \sum_{k=1}^S Norm2(Y_k)}$	Global	Relative
Global Relative Norm2 Error(Y)	$\frac{\sqrt{\sum_{i=1}^S Norm2Error(Y_i)^2}}{\sqrt{\sum_{i=1}^S Norm2(Y_i)^2}}$	Global	Relative
Average Relative Norm2 Error(Y)	$\frac{1}{S} \sum_{i=1}^S RelativeNorm2Error(Y_i)$	Global	Relative
Maximum Relative Norm2 Error(Y)	$\max_{i=1..S} RelativeNorm2Error(Y_i)$	Global	Relative
Average NormInf Error(Y)	$\frac{1}{S} \sum_{i=1}^S NormInfError(Y_i)$	Local	Absolute
Maximum NormInf Error(Y)	$\max_{i=1..S} NormInfError(Y_i)$	Local	Absolute

Verification Points for ROMs

Verification points are not used to build the ROM but rather to verify its accuracy. When **Goodness of Fit** is selected in the **Outline** pane for the **ROM Builder** cell, you can compare the solutions from the "real solve" with the solutions from the ROM.

To add verification points in the verification points table:

1. In the **Outline** pane for the **ROM Builder** cell, select **Verification Points**.
2. In the **Table** pane, for each verification point to add, enter values for its input parameters in the **New Verification Point** row.
3. On the toolbar, click **Update** to perform a real solve of each verification point.

Verification point results are then compared with the ROM predictions and the difference is calculated and displayed in the refinement points table.

Refinement Points for ROMs

Manual refinement is a way to force the ROM to take into account points of your choice, in addition to the points already in the DOE. You insert refinement points in the refinement points table. You do not need to do an initial solve of the ROM (without refinement points) before updating your ROM with refinement points.

To manually add refinement points to the refinement points table:

1. In the **Outline** pane, select **Refinement Points**.
2. In the **Table** pane, for each refinement point to add, enter values for its input parameters in the **New Refinement Point** row.
3. On the toolbar, click **Update** to update each out-of-date refinement point and then rebuild the ROM from both the design points and refinement points.

In the [ROM Builder log file \(p. 270\)](#), metrics take into account refinement points.

ROM Limitations and Known Issues

The following topics summarize ROM limitations and known issues:

[ROM General Limitations](#)

[ROM Viewer Limitations](#)

Note:

For ROM limitations specific to Fluent, see [ROM Limitations](#) in the *Fluent User's Guide*.

ROM General Limitations

- During ROM production, it is not possible to use several Fluent systems along with a single **3D ROM** system.
- Input parameters coming from a non-Fluent cell must be disabled (for instance a **Geometry** cell in a **Fluid Flow** system). All enabled input parameters must be defined in Fluent.
- A **3D ROM** system does not support geometric parameter updates. All geometric parameters enabled in the **Design of Experiments (3D ROM)** cell must be disabled.
- After importing a Fluent case file with the ROM already created, the **Design of Experiments (ROM)** cell may appear as though it is undefined. If this occurs for a case that already has the DOE defined, open the Fluent **Setup** cell and then close Fluent. The **Design of Experiments (3D ROM)** cell will change to **Update Required**.
- Clearing the **Geometry** or **Mesh** cell in a **Fluent Fluid Flow** system created in a previous release may prevent the ROM from being created. Create the Fluent mesh again using the latest release to allow for ROM creation.

- You cannot import a ROMZ file into the Fluent system used to produce the 3D ROM in the same instance of Workbench.

ROM Viewer Limitations

Known limitations follow for the **ROM Viewer** that is launched from Twin Builder. If you do not have access to Twin Builder, you can request an installation package for the **ROM Viewer** from Ansys Technical Support.

- The viewer does not produce logs.
- The button for opening ROM files is accessible in the viewer in Twin Builder, but it must be used only when the viewer is launched separately from Twin Builder. For more information, see [Consuming a ROMZ File in the ROM Viewer \(p. 263\)](#).
- When Twin Builder opens the viewer, it controls the parameter values, not you. While the parameter values should be read-only, this is not currently the case.
- When increasing the minimum display time for a set of results, clicking the pause or stop button might not have an immediate effect.
- When clicking the play button, if the time instance to be replayed is already shown, nothing happens. The workaround is to first display another time instance point by clicking the previous or next button and then clicking the play button.

Using Six Sigma Analysis

This section contains information about running a Six Sigma Analysis. For more information, see [Six Sigma Analysis Component Reference](#) (p. 60) and [Six Sigma Analysis \(SSA\) Theory](#) (p. 385).

[Performing a Six Sigma Analysis](#)

[Using Statistical Postprocessing](#)

[Statistical Measures](#)

Performing a Six Sigma Analysis

The procedure for performing a Six Sigma Analysis includes the following steps:

[Adding a Six Sigma Analysis System](#)

[Defining Uncertainty Variables](#)

[Solving the Six Sigma Analysis System](#)

Adding a Six Sigma Analysis System

To perform a Six Sigma Analysis, you first drag a **Six Sigma Analysis** system from under **Design Exploration** in the **Toolbox** and drop it in your **Project Schematic**. For the **Design of Experiments** cell in this system, you then specify which [input parameters](#) (p. 292) to evaluate, assign distribution functions to them as described in [Defining Uncertainty Variables](#) (p. 283), and specify distribution attributes, which generates your samples for each input variable.

Defining Uncertainty Variables

For each uncertainty variable used in Six Sigma Analysis, you must specify the statistical distribution function that describes its randomness as well as the parameters of this distribution function. The following distribution functions are supported:

- Uniform
- Triangular
- Normal
- Truncated Normal
- Lognormal
- Exponential
- Beta

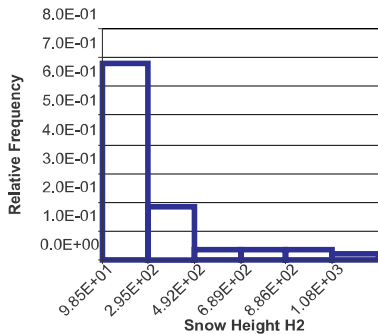
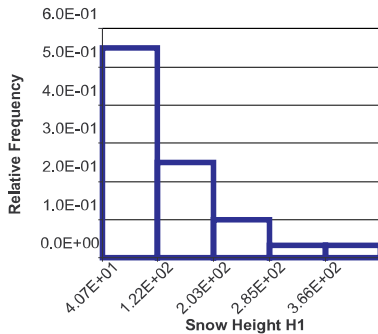
- Weibull

For more information, see [Distribution Functions](#) (p. 391).

Distribution Histogram Example

This example assumes the following:

- You have a beam supporting a roof with a snow load.
- You measured the snow height on both ends of the beam 30 different times.
- You created the following histograms from these measurements.



From these histograms, you can conclude that an exponential distribution is suitable to describe the scatter of the snow height data for **H1** and **H2**. From the measured data, you determine that the average snow height of **H1** is **100 mm** and the average snow height of **H2** is **200 mm**. You can directly derive the parameter λ by dividing 1 by the mean value. This leads to $\lambda_1 = 1/100 = 0.01$ for **H1**, and $\lambda_1 = 1/200 = 0.005$ for **H2**.

Consistency Validations of the Distribution Definition

When you specify input parameters for a Six Sigma Analysis, DesignXplorer validates the distribution properties to ensure that there are no inconsistencies in the distribution definition. However, in some cases, invalid attribute combinations can already exist. For example, a project created with a previous version of DesignXplorer could have invalid attribute combinations. When you open the

project, DesignXplorer runs checks to catch any inconsistencies in the distribution definition. The following table shows the checks performed for each distribution type:

Distribution Type	Validation Check
Uniform	Lower bound is smaller than upper bound.
Triangular	Maximum likely value is between the lower bound and upper bound, and the lower bound is smaller than the upper bound.
Normal	Standard deviation is positive but not too small when compared to the mean: $\sigma_x > \bar{x} \times 1.e-14$
Truncated Normal	Lower bound is smaller than upper bound, and these bounds are not too far away from the mean in terms of standard deviation: $\frac{ \bar{x} - x_{lower} }{\sigma_x} < 20$ and $\frac{ x_{upper} - \bar{x} }{\sigma_x} < 20$
Log Normal	Standard deviation is positive.
Exponential	Decay value is positive.
Beta	Beta shape R and Beta shape T values are positive, and their sum is not too high (Shape R + Shape T < 1000). The lower bound is smaller than the upper bound.
Weibull	Weibull exponent is positive, and Weibull characteristic value is positive and greater than the lower bound.

If inconsistencies are found, a warning dialog box opens. The **Messages** pane provides additional information.

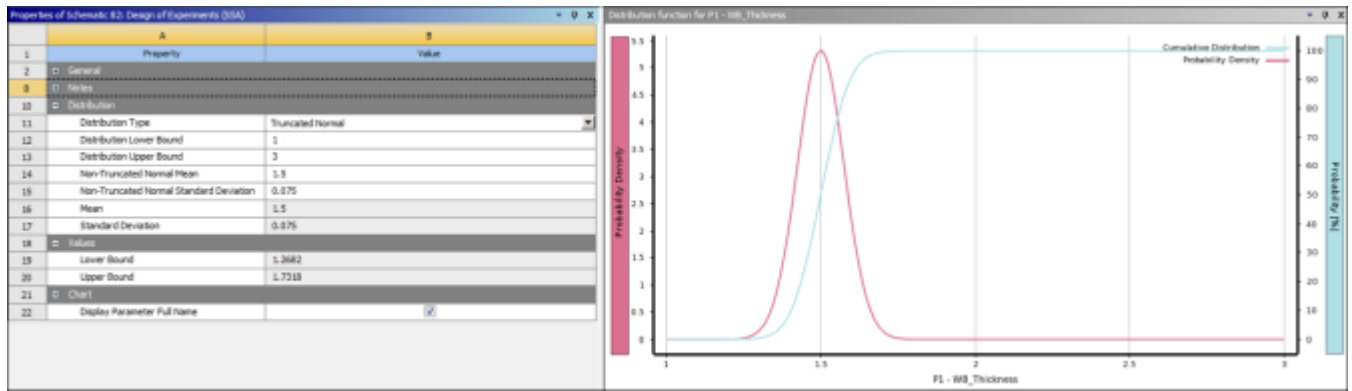
In the **Project Schematic**, the **Design of Experiments** cell has a state of **Attention Required**, indicating that you must edit the distribution definition to resolve any inconsistencies.

Example:

The truncated normal distribution is used in this example. It is defined by the following attributes:

- Mean
- Standard deviation
- Distribution lower bound
- Distribution upper bound

In the following figure, you can see these attributes in the **Properties** and **Chart** panes for the **Design of Experiments** cell in the **Six Sigma Analysis** system:



In this example, **Distribution Upper Bound (3)** is far away from the **Mean (1.5)** when compared to the **Standard Deviation (0.075)**. The validation check fails because $(\text{Upper Bound} - \text{Mean}) / \text{Standard Deviation} = 20$. To fix the inconsistency, you must either reduce the bounds, modify the mean, or modify the standard deviation.

Solving the Six Sigma Analysis System

Once your **Six Sigma Analysis** system is defined, you can solve it in any of the following ways:

- To update each cell in the analysis separately, right-click the cell and select **Update**.
- To update the entire analysis at once, right-click the system header in the **Project Schematic** and select **Update**.
- To update the entire project at once, in the **Project Schematic**, click **Update Project** on the toolbar.

Using Statistical Postprocessing

On the **Six Sigma Analysis** component tab, for the parameter selected in the **Outline** pane, you can see statistics in the **Properties** pane, probability tables in the **Table** pane, and statistics charts in the **Chart** pane. To view a Sensitivities chart, in the **Outline** pane under **Charts**, select this chart.

Tables (SSA)

In the **Six Sigma Analysis** component tab, you can view probability tables for any input or output parameter selected in the **Outline** pane. In the **Properties** pane for the parameter, select **Quantile-Percentile** or **Percentile-Quantile** for **Probability Table**.

You can modify both types of tables by adding or deleting values.

- To add a value to the Quantile-Percentile table, type the desired value into the **New Parameter Value** cell at the end of the table. A row with the value that you entered is added to the table in the appropriate location.
- To add a new value to the Percentile-Quantile table, type the desired value into the appropriate cell (**New Probability Value** or **New Sigma Level**) at the end of the table.
- To delete a row from either table, right-click the row and select **Remove Level**.

You can also overwrite any value in an editable column. Corresponding values are then displayed in the other columns in this row.

Using Parameter Charts (SSA)

You can review the statistical results of the analysis by selecting an input or output parameter to view its **Chart**. The results of a Six Sigma Analysis are visualized using histogram plots and cumulative distribution function plots.

You can change various generic [chart properties](#) for this chart.

Using the Sensitivities Chart (SSA)

If you select **Sensitivities** in the **Outline** pane for the **Six Sigma Analysis** cell, you can review the sensitivities derived from the samples generated for the analysis. Sensitivities for Six Sigma Analysis (SSA) are [global sensitivities](#) (p. 313), not [local sensitivities](#) (p. 157). In the **Properties** pane for the Sensitivities chart, you can choose the output parameters for which you want to view sensitivities. You can also choose the input parameters that you would like to evaluate for the output parameters.

You can change various generic [chart properties](#) for this chart.

Note:

If the p-Value calculated for a particular input parameter is above the value specified for **Significance Level** in **Tools** → **Options** → **Design Exploration**, the bar for that parameter is shown as a flat line on the chart. For more information, see [Viewing Significance and Correlation Values](#) (p. 72).

Statistical Measures

When the **Six Sigma Analysis** cell is updated, the following statistical measures display in the **Properties** pane for each parameter. For descriptions of these measures, see [SSA Theory](#) (p. 400).

- Mean
- Standard deviation
- Skewness
- Kurtosis
- Shannon entropy
- Signal-to-noise ratios
- Sigma minimum and maximum

Working with DesignXplorer

The following section describe the type of work you do in DesignXplorer:

[Working with Parameters](#)

[Working with Design Points](#)

[Working with Sensitivities](#)

[Working with Tables](#)

[Working with Remote Solve Manager and DesignXplorer](#)

[Working with Design Point Service and DesignXplorer](#)


[Working with DesignXplorer Extensions](#)

[Working with Design Exploration Results in Workbench Project Reports](#)

Working with Parameters

Parameters are exposed from the individual systems being analyzed. You can edit any cell of a design exploration system to see the available input and output parameters in the **Outline** pane. For the parameter selected in the **Outline** pane, you can set properties and view additional data in the **Properties** pane.

Note:

If you modify your analysis after it is solved, parameters can change. DesignXplorer displays the refresh required icon () on cells with changed data.

Parameters as Design Variables for Optimization

When you insert a design exploration system in the **Project Schematic**, DesignXplorer captures the current value of each input parameter. This initial value is the default value and is used to determine the default parameter range. The upper and lower bounds default to plus and minus 10% of the initial value, respectively. The default values might not be valid in certain situations. Be sure to verify that ranges are valid with respect to the input geometry and analysis system scenario.

Effects of Parameter Changes on Design Exploration Systems

When changes are made to input parameters, cells in design exploration systems must be refreshed. On a **Refresh** operation, design exploration results can be partially or completely invalidated, depending on the nature of the change. This includes design point results and the cache of design point results, which means an **Update** operation is required to recalculate them.

- When a direct input or direct output parameter is added to or deleted from the project, or when the unit of a parameter is changed, all results, including design point results and the cache of

design point results, are invalidated on a **Refresh** operation. An **Update** operation is required to recalculate all design points.

- When a derived output parameter is added to or deleted from the project, or when the expression of a derived output parameter is modified, all results but the cache of design point results are invalidated on a **Refresh** operation. So, on an **Update** operation, all design point results are retrieved without any new calculation. Other design exploration results are recalculated.
- Because DesignXplorer is primarily concerned with the range of variation in a parameter, changes to a parameter value in the model or the **Parameter Set** bar are not updated to existing design exploration systems with a **Refresh** or **Update** operation. The parameter values that were used to initialize a new design exploration system remain fixed within DesignXplorer unless you change them manually.

Tip:

You can change the way that units are displayed in your design exploration systems from the **Units** menu. Changing the units display in this manner causes the existing data in each system to be shown in the new units system. It does not require an update of the design exploration systems.

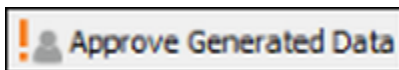
Effects of Non-Parametric Changes on Design Exploration Systems

Parametric changes occur when you edit an input parameter value or add or delete a parameter. All other changes are non-parametric.

When you make non-parametric changes, the first cell in the design exploration system indicates that a refresh is required. However, the **Refresh** operation invalidates design point results and the cache of design points. An **Update** operation is then required to recalculate design points.

Because generating new design points can be time-consuming and costly, when you know that recalculating design points is unnecessary, you can approve generated data instead. Rather than invalidating design point results, the **Approve Generated Data** operation retains the already generated design points as up-to-date and retains the design point results in the cache of design points.

If non-parametric changes exist, the **Approve Generated Data** option is available in the right-click context menu for the first cell in a design exploration system. Additionally, when the cell is being edited, a button is available on the toolbar:



Note:

A response surface or sensitivity study is based on mathematically analyzing variations of the outputs and trying to relate them to variations of the inputs. It works best if all variation within the data set come from variation of the parametric inputs. If other inputs are also varying but are not being tracked, the resulting input "noise" can potentially reduce the accuracy of any downstream analysis. You should only choose to approve non-parametric model variations if you are confident that they have minimal influence on the results.

DesignXplorer flags results based on user-approved data so that others who view them are aware of the potential inaccuracy.

When this operation runs, all cells that were up-to-date before the non-parametric changes are once again up-to-date.

- In the **Outline** pane for each cell with user-approved data, an alert icon (⚠) displays in the **Message** column for the root node. If you click this icon to view warnings, you see a cautionary message that indicates data contains user-approved generated data that may include non-parametric changes.
- In the **Table** pane for the cell, the user-approved icon (⚠) displays in the **Name** column of all design points with user-approved data.

Outline of Schematic B2: Design of Experiments				Table of Outline A2: Design Points of Design of Experiments				
	A	B	C		A	B	C	
1		Enabled	Message	1	Name	P1 - WB_Thickness	P2 - WB_Radius	P3
2	✓ Design of Experiments		1 ⚠	2	⚠ 1	2	132	1.0
3	Input Parameters			3	⚠ 2	1	132	1.0
4	Microsoft Office Excel (A1)			4	⚠ 3	3	132	1.0
5	P1 - WB_Thickness	✓		5	⚠ 4	1.5	132	1.0
6	P2 - WB_Radius	✓		6	⚠ 5	2.5	132	1.0
7	Output Parameters			7	⚠ 6	2	108	8.8
8	Microsoft Office Excel (A1)			8	⚠ 7	1	108	8.7
9	P3 - WB_Mass			9	⚠ 8	3	108	8.9
10	P4 - WB_Deformation			10	⚠ 9	1.5	108	8.8
11	P5 - WB_Stress			11	⚠ 10	2.5	108	8.9
12	P6 - WB_Sinus			12	⚠ 11 DP 0	2	120	9.8
13	P7 - Output Parameter			13	⚠ 12	1	120	9.7
14	Charts			14	⚠ 13	3	120	9.9
15	Parameters Parallel			15	⚠ 14	1.5	120	9.8
16	Design Points vs Parameter			16	⚠ 15	2.5	120	9.9
17	Parameters Parallel							

A **Design of Experiments** cell with user-approved design points is once again marked as up-to-date. A **Response Surface** cell can be marked as up-to-date if it depends on a **Design of Experiments** cell with user-approved design points or if it contains user-approved points, such as refinement points.

When a cell in a design exploration system contains user-approved data, any cell that depends on it also displays the user-approved icon. For example, if a **Design of Experiments** cell contains user-approved data, any **Response Surface** cell that depends on this DOE displays this icon. Any new design points that you might insert in a cell's table do not display icons because their results are to be based on a real solve.

Assume that you switch to another DOE type and generate new design points, without keeping the previous design points. The DOE then contains only new design points, so no user-approved icons display. However, cells that depend on this DOE still display user-approved icons.

- The **Response Surface** cell displays the icon because it still contains user-approved refinement points. If you deleted these refinement points and updated the response surface, the **Response Surface** cell would no longer display the icon.
- The **Optimization** cell displays the icon because it still contains user-approved verified candidate points. If you deleted these candidate points and updated the optimization, the **Optimization** cell would no longer display the icon.

All reports that DesignXplorer automatically generates include the same cautionary message that is displayed when you click the alert icon (⚠) in the **Message** column for the root node of a cell with user-approved data. For example, the Workbench project report contains this cautionary message in corresponding component sections. Additionally, in table images, the project report displays the user-approved icon (👍) in the **Name** column of user-approved points.

If you export table or chart data, the first row of the CSV file also includes the same cautionary message.

Note:

- In some situations, non-parametric changes are not detected. For example, if you edit the input file of a Mechanical APDL system outside of Workbench, this non-parametric change is not detected. To synchronize design exploration systems with the new state of the project, you must perform a **Clear Generated Data** operation followed by an **Update** operation. In a few rare cases, inserting, deleting, duplicating or replacing systems in a project is not reliably detected.
 - The **Clear Generated Data** operation does not clear the design point cache. To clear the design point cache, right-click in an empty area of the **Project Schematic** and select **Clear Design Points Cache for All Design Exploration Systems**.
-

Input Parameters

By defining and adjusting input parameters, you specify the analysis of the model under investigation. This section describes how to define and change input parameters.

Defining Discrete Input Parameters

A discrete parameter physically represents a configuration or state of the model, such as the number of holes or number of weld points in a geometry. The integer values possible for a discrete parameter are known as levels, which are editable.

To define an input parameter as discrete:

1. In the **Outline** pane for the **Design of Experiments** cell, select the parameter.
2. In the **Properties** pane, set **Classification** to **Discrete**.

Both the **Properties** pane and **Table** pane display *level* information for the discrete parameter selected in the **Outline** pane. In the **Properties** pane, you see the number of levels (discrete values). In the **Table** pane, you see the integer values for each level. You can add, delete, and edit levels for the discrete parameter, even if it is disabled.

The image shows three panels from the ANSYS DesignXplorer interface:

- Outline of Schematic B2: Design of Experiment:** A tree view showing a hierarchy of parameters. P2 - H is highlighted in yellow.
- Table of Outline A6: P2 - H:** A table with columns 'A' (Name) and 'B' (Discrete Value). It lists Level 1 (7) and Level 2 (3), with a red box around these two rows.
- Properties of Outline A6: P2 - H:** A table with columns 'A' (Property) and 'B' (Value). The 'Classification' is set to 'Discrete' and 'Number Of Levels' is set to 2, with a red box around these two rows.

Note:

When a parametrized property is constrained to integer values, DesignXplorer automatically initializes the parameter as discrete with a single level. The value for this single level is equal to the current value of the parameter. In this case, **Classification** is set to **Discrete** and is not editable.

- In the **Table** pane, define the levels for the discrete parameter:
 - To add a level, select the empty cell in the bottom row of the **Discrete Value** column, type an integer value, and press **Enter**. In the **Properties** pane, **Number of Levels** is updated automatically.
 - To delete a level, right-click any part of the row containing the level to remove and select **Delete**.

- To edit a level, select the cell with the value to change, type an integer value, and press **Enter**.

Note:

In the **Table** pane, the **Discrete Value** column is not sorted as you add, delete, and edit levels. To sort it manually, click the down-arrow on the right of the header cell and select a sorting option. Once you sort the column, integer values are auto-sorted as you add, delete, and edit levels.

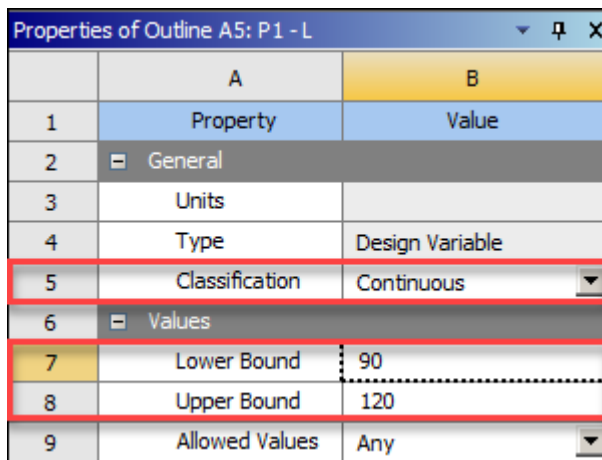
Defining Continuous Input Parameters

A continuous parameter is one that physically varies in a continuous manner within a specific range of analysis. The range is defined by a lower bound and an upper lower bound, which default respectively to -10% and $+10\%$ of the initial value of the parameter. You can edit the default bounds to adjust the range.

To define an input parameter as continuous:

1. In the **Outline** pane for the **Design of Experiments** cell, select the parameter.
2. In the **Properties** pane, set **Classification** to **Continuous**.

The values for **Lower Bound** and **Upper Bound** define the range of the analysis.



	A	B
1	Property	Value
2	General	
3	Units	
4	Type	Design Variable
5	Classification	Continuous
6	Values	
7	Lower Bound	90
8	Upper Bound	120
9	Allowed Values	Any

DesignXplorer initializes the range based on the current value for the parameter, using -10% for the lower bound and $+10\%$ for the upper bound. If a parameter has a current value of 0.0, the initial range is computed as $0.0 \rightarrow 10.0$.

Because DesignXplorer is not aware of the physical limits of parameters, you must check that the assigned range is compatible with the physical limits of the parameter. Ideally, the current value of the parameter is at the midpoint of the range between the upper and lower bounds. However, this is not a requirement.

3. If you need to change the range, select the bound to change, type in a number, and press **Enter**. You are not limited to entering integer values. However, the relative variation must

be equal to or greater than 1e-10 in the same units as the parameter. If the relative variation is less than 1e-10, you can either adjust the range or disable the parameter.

- To adjust the range, select the parameter in the **Outline** pane and then edit the values for the bounds in the **Properties** pane.
- To disable the parameter, clear the **Enable** check box in the **Outline** pane.

In the **Properties** pane, **Allowed Values** specifies whether you want to impose a limitation beyond the range defined by the lower and upper bounds. When this property is editable, the default setting is **Any**, which means any value within the range is allowed.

4. If you want to further limit values, change the setting for **Allowed Values**. Descriptions follow for the other choices that are possible. The subsequent table summarizes when a choice is shown.
 - **Snap to Grid**. When selected, **Grid Interval** displays, specifying the distance that must exist between adjacent design points when generating new design points. The units are the same as those for the parameter. The default value is calculated as follows: $(\text{Upper Bound} - \text{Lower Bound}) / 1000$. This value is then rounded to the closest power of 10. For more information, see [Defining a Grid Interval \(p. 296\)](#).
 - **Manufacturable Values**. When selected, levels display in the **Table** pane so that only the real-world manufacturing or production values that you specify are taken into account during postprocessing. For more information, see [Defining Levels for Manufacturable Values \(p. 297\)](#).

The following table describes possible states for **Allowed Values** and choice availability based on its state.

System	Cell	State of "Allowed Values"	Choices for "Allowed Values"
Direct Optimization	Optimization	Editable	<ul style="list-style-type: none"> • Any • Snap to Grid • Manufacturable Values
Response Surface Optimization	Design of Experiments	Editable	<ul style="list-style-type: none"> • Any • Manufacturable Values
	Response Surface	Read-only	Same value as selected in the DOE
	Optimization	Editable if Any is selected in the DOE	<ul style="list-style-type: none"> • Any • Snap to Grid
		Read-only if Manufacturable Values is selected in the DOE	Manufacturable Values

Defining a Grid Interval

When you set **Allowed Values** to **Snap to Grid**, you eliminate spending too much time progressing towards a better design with increments that you know are insignificant. This is because the value for **Grid Interval** determines the distance that must exist between adjacent design points when generating new design points. The optimizer uses the grid interval to produce a series of values between the lower and upper bounds, selectively picking new design points from the grid.

Note:

- Use of this method does not affect existing design points. They are reused without adjustment, even when they do not match the grid.
- Not all optimization methods support setting **Allowed Values** to **Snap to Grid**. For example, NLPQL does not support this setting.

The default value for **Grid Interval** is calculated as follows: $(\text{Upper Bound} - \text{Lower Bound})/1000$. The resulting value is then rounded to the closest power of 10. For example, assume a lower bound of 3.52 and an upper bound of 4.08. After subtracting 3.52 from 4.08, the calculation $(0.56/1000)$ yields 0.00056, which is rounded to 0.001.

You can specify a different value for **Grid Interval**. However, the value cannot be negative or zero. It must also be less than or equal to this value: $(\text{Upper Bound} - \text{Lower Bound})$.

When you edit the grid interval, the lower bound, upper bound, or starting point can become invalid. In this case, DesignXplorer highlights in yellow the values which need your attention. Edit each highlighted property to enter a value matching the grid interval. If you enter an invalid value, DesignXplorer automatically snaps it to the grid.

When **Snap to Grid** is set, DesignXplorer excludes optimization methods that do not support this setting from the list of methods available. However, you might have already selected such a method before you set **Snap to Grid**. In this case, DesignXplorer cannot update the **Optimization** cell. When you place the mouse cursor over **Optimization** in the **Outline** pane, a tooltip indicates that the selected optimization method does not support continuous input parameters with **Allowed Values** set to **Snap to Grid**. It then indicates that you must either select another optimization method or change **Allowed Values** to some choice other than **Snap to Grid**.

DesignXplorer also cannot update the **Optimization** cell in other situations where at least one continuous input variable has **Allowed Values** set to **Snap to Grid**. Tooltips for the **Optimization** cell indicate how to resolve these additional problems:

- A lower or upper bound is not on the grid defined by the grid interval.
- A starting value is not on the grid defined by the grid interval.
- A grid interval is too small.
- A grid interval is not defined.

Defining Levels for Manufacturable Values

When you set **Allowed Values** to **Manufacturable Values**, you limit analysis of the sample set to only values representing real-world manufacturing or production constraints. You specify the manufacturable values to use as levels in the **Table** pane. The **Optimization** cell then returns designs based on only the manufacturable values specified.

In the **Properties** pane, **Number of Levels** is initially set to **2** because this is the minimum number of levels allowed. The **Table** pane displays all levels specified for manufacturable values. The values for the two initial levels default to the lower and upper bounds. You can add, delete, and edit levels, even if this continuous input parameter is currently disabled.

The screenshot displays the ANSYS Design of Experiments interface. The **Outline of Schematic B2: Design of Experiments** pane shows a tree view with parameter **P1 - B** selected. The **Table of Outline A5: P1 - B** pane shows the following data:

	A	B
1	Name	Manufacturable Values
2	Level 1	1.8
3	Level 2	2.2
*	New Level	

The **Properties of Outline A5: P1 - B** pane shows the following configuration:

	A	B
1	Property	Value
2	General	
3	Units	
4	Type	Design Variable
5	Classification	Continuous
6	Values	
7	Lower Bound	1.8
8	Upper Bound	2.2
9	Allowed Values	Manufacturable Values
10	Number Of Levels	2

A red arrow points from the **Number Of Levels** value of 2 in the Properties pane to the **New Level** row in the Table pane.

In the **Table** pane, define the manufacturable values:

- To add a level, select the empty cell in the bottom row of the **Manufacturable Values** column, type a numeric value, and press **Enter**. In the **Properties** pane, **Number of Levels** is updated automatically.
- To delete a level, right-click any part of the row containing the level to remove and select **Delete**. If you delete the level representing either the upper bound or lower bound, the range is not narrowed.
- To edit a level, select the manufacturable value to change, type a numeric value, and press **Enter**.

Note:

- In the **Properties** pane, **Value** is populated with the parameter value defined in the **Parameters** table. This value cannot be edited in the DOE.
- If you enter a numeric value for a level that is outside of the range defined in the **Properties** pane, you can opt to either automatically extend the range to encompass the new value or cancel the commit of the new value. If you opt to extend the range, all existing DOE results and design points are deleted.
- If you adjust the range when manufacturable values are defined, manufacturable values falling outside the new range are automatically removed and all results are invalidated.

If you decide that you no longer want to limit analysis of the sample set to only manufacturable values, you can set **Allowed Values** to **Any**. This removes the display of levels from the **Tables** pane. If you ever set **Allowed Values** to **Manufacturable Values** again, the **Table** pane once again displays the levels that you previously defined.

Once results are generated, you can change the setting for **Allowed Values** without invalidating the DOE or response surface. You can also add, delete, or edit levels without needing to regenerate the entire DOE and response surface, provided that you do not alter the range. As long as the range remains the same, DesignXplorer reuses the information from the previous updates.

Because the following types of results are based on manufacturable values, if you make any edits to manufacturable values, you must regenerate them:

- Response points and related charts
- Min/Max objects
- Optimization candidates and charts

Changing Input Parameters

You can make changes to an input parameter in a **Design of Experiments**, **Design of Experiments (SSA)**, or **Parameters Correlation** cell. In the **Outline** pane for the cell, select the input parameter that you want to edit. You make most parameter changes in the **Properties** pane. The changes that you make affect only the parameter information for the current system and any systems that share a **Design of Experiments** or **Response Surface** cell from this system.

- From the **Outline** pane for the **Design of Experiments** cell, you enable or disable an input parameter by selecting or clearing the check box to the right of the parameter.
- In the **Properties** pane for the parameter selected in the **Outline** pane, you specify further attributes, such as the levels (integer values) for a discrete input parameter or the range (upper and lower bounds) and allowed values for a continuous input parameter.

Making any of the following changes to an input parameter in a **Design of Experiments**, **Design of Experiments (SSA)**, or **Parameters Correlation** cell would require clearing all generated data associated with this system:

- Enable or disable an input parameter.
- Change **Classification** of an input parameter from **Continuous** to **Discrete** or vice versa.
- Add or remove a level for a discrete parameter.
- Change the range for a continuous input parameter.

For example, assume that you make one of these changes in a **Design of Experiments** cell for a goal-driven optimization system. Because the change would clear all generated data in all cells of the system, a dialog box displays, asking you to confirm the change. The change is committed only if you click **Yes**. If you click **No**, the change is discarded. If desired, you can duplicate the system and then make the parameter change in the new system. Alternatively, you can change the DOE type to **Custom** before making a parameter change to retain the design points falling within the new range.

Note:

Using the [DOE type \(p. 83\)](#), the number of generated design points is directly related to the number of selected input parameters. The design and analysis [workflow \(p. 29\)](#) shows that specifying many input parameters makes heavy demands on computer time and resources, including system analysis, DesignModeler geometry generation, and CAD system generation. Also, large ranges for input parameters can lead to inaccurate results.

Design Variables and Uncertainty Variables

A range (upper and lower bounds) of values is required for all continuous input parameters. Also, values are required for discrete input parameters or continuous input parameters with manufacturable values.

Six Sigma Analysis refers to input variables as *uncertainty variables*. For more information, see [Defining Uncertainty Variables \(p. 283\)](#).

Output Parameters

Each output parameter corresponds to a response surface, which is expressed as a function of the input parameters. Some typical output parameters are equivalent stress, displacement, and maximum shear stress.

When you select an output parameter in the **Outline** pane for a cell that displays parameters, you can see maximum and minimum values for this output parameter in the **Properties** pane. The maximum and minimum values shown depend on the state of the design exploration system.

They are the "best" minimum and maximum values available in the context of the current cell. This means that they are the best values between what the current cell eventually produced and what the parent cell provided. A **Design of Experiment** cell produces design points, and a best minimum value and maximum value are extracted from these design points. A **Response Surface** cell produces Min-Max search results if this option is enabled. If a refinement is run, new points are generated, and a best minimum value and maximum value are extracted from these points. An **Optimization** cell produces a sample set and again, a better minimum value and maximum value than what is provided by the parent response surface is found in these samples. Consequently, it is important to remember the state of the design exploration system when viewing the minimum and maximum values in the parameter properties.

Working with Design Points

When you update a design exploration system, DesignXplorer creates design points in the **Parameter Set** bar and requests that Workbench updates them so that corresponding output parameter values can be obtained. These solved design points are the input data that DesignXplorer uses to calculate its own parametric results, such as response surface, sensitivities, optimum designs, and more.

The number and the definition of the design points created depend on the number of input parameters and the properties of the DOE. For more information, see [Using a Central Composite Design DOE \(p. 94\)](#).

You can preview the generated design points by clicking **Preview** on the toolbar before updating a DOE, a response surface (if performing a refinement), or another design exploration feature that generates design points during an update.

Some design exploration features provide the ability to edit the list of design points and the output parameter values of these points. For more information, see [Working with Tables \(p. 314\)](#).

Before starting the update of design points, you can set the design point update option, change the design point update order, and specify initialization conditions for a design point update. For more information, see [Specifying Design Point Update Options \(p. 301\)](#).

You can update a design exploration cell or system in several ways:

- From a component tab, click **Update** on the toolbar.
- From a component tab, right-click the root node in the **Outline** pane and select **Update**.
- In the **Project Schematic**, right-click the cell and select **Update**.
- In the **Project Schematic**, click **Update Project** on the toolbar to update all systems in the project.

Note:

- The **Update All Design Points** operation updates the design points for the **Parameter Set** bar. It does not update the design points for design exploration systems.

- If an update of a design exploration system reuses design points that are partially updated, the update required icon (🔧) displays beside the output parameters that are partially updated. DesignXplorer always updates partially updated design points to completion and publishes an informational message.

During the update, design points are updated simultaneously if the analysis system is configured to perform simultaneous solutions. Otherwise, they are updated sequentially.

When you update a **Design of Experiments**, **Response Surface**, or **Parameters Correlation** cell, its design points table is updated dynamically. As the points are solved, generated design points appear and their results display.

As each design point is updated, parameter values are written to a CSV log file. If you want to use this data to continue your work with the response surface, you can import it back into this cell's design points table.

Specifying Design Point Update Options

Before updating multiple design points, you can set the design point update option, change the design point update order, and specify initialization conditions for a design point update.

Setting the Design Point Update Option

In the properties for the **Parameter Set** bar, **Update Option** specifies where design point updates are processed. For more information, see [Setting the Design Point Update Option](#) in the *Workbench User's Guide*. The update location for design points can differ from the update location of solution cells, which is described in [Solution Process](#).

Changing the Design Point Update Order

DesignXplorer already optimizes the design point update order for the DOEs that it creates. However, you might want to change the design point update order yourself to further improve its efficiency. For instance, if several design points use the same geometry parameter values, it is more efficient to process them together so that the geometry is updated only once.

You can change the update order using options in the design points table. For more information, see [Changing the Design Point Update Order](#) in the *Workbench User's Guide*.

Specifying the Initialization Conditions for a Design Point Update

In the properties for the **Parameter Set** bar, **Design Point Initiation** is available when **Update Option** is set to **Run in Foreground** or **Submit to Remote Solve Manager**.

- If **Design Point Initiation** is set to **From Current** (default), when a design point is updated, it is initialized with the data of the design point that is designated as current.
- If **Design Point Initiation** is set to **From Previously Updated**, when a design point is updated, it is initialized with the data of the previously updated design point. In some cases, it can be

more efficient to update each design point starting from the data of the previously updated design point, rather than restarting from the current design point each time.

Note:

Retained design points with valid retained data do not require initialization data.

For more information, see [Specifying the Initialization Conditions for a Design Point Update](#) in the *Workbench User's Guide*.

Preserving Generated Design Points to the Parameter Set

During the update of a design exploration system, the generated design points are temporarily created in the project and listed in the design points table for the **Parameter Set** bar. Once the operation is completed, each generated design point is removed from the project unless it has failed to update. In this case, it is preserved to facilitate further investigation.

DesignXplorer allows you to preserve generated design points so that they are automatically saved to the **Parameter Set** bar for later exploration or reuse. The preservation of design points must be enabled first at the project level. You can then configure the preservation of design points for individual components.

- You enable this functionality at the project level in **Tools** → **Options** → **Design Exploration**. Under **Design Points**, select the **Preserve Design Points After DX Run** check box.
- You enable this functionality at the component level in the cell properties. Right-click the cell and select **Edit**. In the **Properties** pane, under **Design Points**, select the **Preserve Design Points After DX Run** check box.

When design points are preserved, they are included in the design points table for the **Parameter Set** bar. A design points table for a cell indicates if design points here correspond to design points for the **Parameter Set** bar. Design points include DOE points, refinement points, direct correlation points, and candidate points. Design points correspond when they share the same input parameter values.

When a correspondence exists, the point's name specifies the design point to which it is related. If the source design point is deleted from the **Parameter Set** bar or the definition of either design point is changed, the indicator is removed from the point's name and the link between the two points is broken, without invalidating your model or results.

Retaining Data for Generated Design Points

In addition to preserving the design points in the project, you can retain the data for each preserved design point. The design point data is saved within the project, enabling you to switch back and forth between different designs within the same project. You can also configure retained design point functionality at the component level.

- You enable this functionality at the project level in **Tools** → **Options** → **Design Exploration**. Under **Design Points**, select both the **Preserve Design Points After DX Run** and **Retain Data for Each Preserved Design Point** check boxes.

- You enable this functionality at the component level in the cell properties. Right-click the cell and selecting **Edit**. In the **Properties** pane, under **Design Points**, select both the **Preserve Design Points After DX Run** and **Retain Data for Each Preserved Design Point** check boxes.

Note:

The behavior of a newly inserted cell follows the project-level settings. The behavior of existing cells is not affected. Existing cells follow their configuration at the component level.

When a DesignXplorer cell is updated, preserved design points are added to the project's design points table and the calculated data for each of these design points is retained.

Once design point data has been retained, you have the option of using the data within the project or exporting design point data to a separate project:

- To switch to another design within the project, right-click a design point in the design points table and select **Set as Current**. This allows you to review and explore the associated design.
- To export retained design point data to a separate project, go to the design points table, right-click one or more design points with retained data, and select **Export Selected Design Points**.

For more information about using retained design point data, see [Preserving Design Points and Retaining Data \(p. 311\)](#) in the *Workbench User's Guide*.

Inserting Design Points

From a design points table for a DesignXplorer system, you can insert selected design points into the project by selecting **Insert as Design Point** from the context menu. Some tables that support this operation are the Candidate Points table for an **Optimization** cell and the Min-Max Search table for a **Response Surface** cell.

In a chart, you can right-click a design point and select **Insert as Design Point**. Some charts that support this operation are the Samples chart, Tradeoff chart, Response chart, and Correlation Scatter chart.

When inserting new design points into the project, the input parameter values of the selected candidate design points are copied. The output parameter values are not copied because they are approximated values provided by the response surface. To view the existing design points in the project, edit the **Parameter Set** bar in the **Project Schematic**.

The availability of the **Insert as Design Point** option from an optimization chart depends on the context. For instance, you can right-click a point in a Tradeoff chart to insert it as a design point in the project. The same operation is available from a Samples chart, providing it is not being displayed as a Spider chart.

Note:

If your cell is out-of-date, the charts and tables that you see are out-of-date. However, you can still explore and manipulate the existing table and chart data and insert design points.

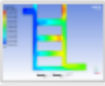



Viewing Design Point Images in Tables and Charts

You can display image files for solved design points in DesignXplorer tables and charts if the following conditions exist:

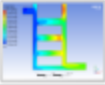



- Design points exist in the **Parameter Set** bar. Because this is not the case by default, you can easily accomplish this by selecting the **Preserve Design Points After DX Run** check box for the DesignXplorer component and then updating the component. If you want to enable this property for all new DesignXplorer components, you do so in **Tools** → **Options** → **Design Exploration**. For more information, see [Design Exploration Options \(p. 34\)](#).
- A Workbench project report exists. For information about Workbench project reports, see [Working with Project Reports](#) in the *Workbench User's Guide*.
- The image to display in DesignXplorer tables and charts is selected for the **Report Image** property for a DesignXplorer component that uses design points. In the following figure, you can see that **Preserve Design Points After DX Run** is selected and **Report Image** has a PNG file selected. You can select from all PNG files that were generated for the Workbench project report.

Property	Value
[-] Design Points	
Preserve Design Points After DX Run	<input checked="" type="checkbox"/>
Retain Data for Each Preserved Design Point	<input type="checkbox"/>
[-] Failed Design Points Management	
Number of Retries	0
[-] Optimization	
Method Name	NLPQL ▼
Finite Difference Approximation	Forward ▼
Allowable Convergence (%)	0.1
Maximum Number of Iterations	20
Maximum Number of Candidates	3
[-] Optimization Status	
Converged	No
Number of Iterations	0
Number of Evaluations	0
Number of Failures	0
Size of Generated Sample Set	0
Number of Candidates	0
[-] Design Point Report	
Report Image	Post-Results:Figure001.png ▼

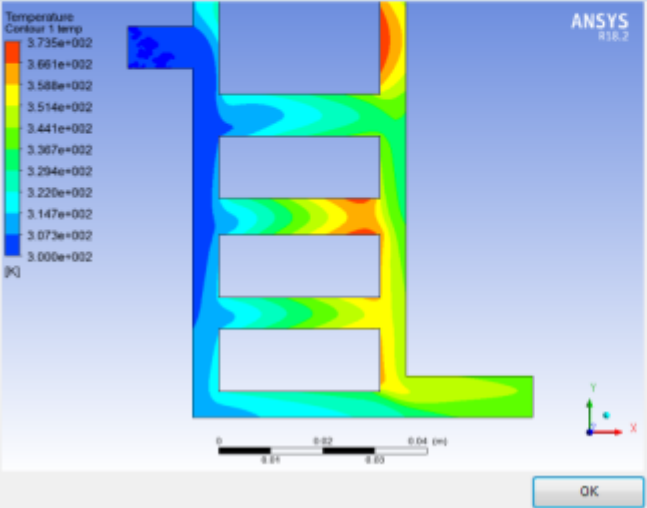
Initially, **Report Image** is set to **None**, which means tables using design points do not display the **Report Image** column. However, once you select a PNG file for **Report Image**, all tables using design points display this column. If the selected image exists for a design point, the column displays a thumbnail.

Reference	Name	P2 - inlet_velocity (m s ⁻¹)	P4 - heat_convection (W m ⁻² K ⁻¹)	P9 - maxtemp (K)		P10 - outtemp (K)	Report Image
				Parameter Value	Variation from Reference		
Starting Point	DP 0	0.06	1	377.04	0.00 %	342.91	
Candidate Point 1	DP 20	0.060818	1.1	373.34	-0.98 %	341.63	
Candidate Point 2	DP 17	0.060177	1.0245	376.15	-0.24 %	342.59	
Candidate Point 3	DP 14	0.060022	1.0039	376.91	-0.03 %	342.85	
New Custom Candidate Point		0.06	1				

To open the image, you click the thumbnail.

Reference	Name	P2 - inlet_velocity (m s ⁻¹)	P4 - heat_convection (W m ⁻² K ⁻¹)	P9 - maxtemp (K)		P10 - outtemp (K)	Report Image
				Parameter Value	Variation from Reference		
Starting Point	DP 0	0.06	1	377.04	0.00 %	342.91	
Candidate Point 1	DP 20	0.060818	1.1	373.34	-0.98 %	341.63	
Candidate Point 2	DP 17	0.060177	1.0245	376.15	-0.24 %	342.59	
Candidate Point 3	DP 14	0.060022	1.0039	376.91	-0.03 %	342.85	

DP 20 Post-Results:Figure001.png



ANSYS 18.2

Temperature Contour 1 temp

3.735e+002

3.661e+002

3.588e+002

3.514e+002

3.441e+002

3.367e+002

3.294e+002

3.220e+002

3.147e+002

3.073e+002

3.000e+002

[K]

OK

From a chart, you open the image for a point by right-clicking the point and selecting **Show Report Image**. If this context menu option is not available, the selected point is not linked to a design point.

During a design point update, the **Report Image** column displays new thumbnails as images for design points are generated. While the update is running, you can open an image.

Cache of Design Point Results

To reduce the time required to perform parametric analyses in a project, DesignXplorer stores the design points that it has updated successfully in its design points cache.

As a consequence, if the same design points are reused when previewing or updating a design exploration system, they immediately show up as up-to-date and the cached output parameter values display.

The cached data is invalidated automatically when relevant data changes occur in the project. You can also force the cache to be cleared by right-clicking in an empty area of the **Project Schematic** and selecting **Clear Design Points Cache for All Design Exploration systems**.

Raw Optimization Data

DesignXplorer allows you to monitor design point data for **Direct Optimization** systems, both while the optimization is in process and after it is completed.

While a **Direct Optimization** system is updating, DesignXplorer refreshes the results in the design points table as it is calculated. Design points pulled from the cache also display. The **Table** pane generally refreshes dynamically as design points are submitted for update and as they are updated. However, if design points are updated via Remote Solve Manager, the RSM job must be completed before results in the **Table** pane are refreshed.

Once the optimization is completed, the raw design point data is saved. When you select **Raw Optimization Data** in the **Outline** pane, the **Table** pane displays the raw data. While you cannot edit the raw data, you can export it to a CSV file by right-clicking in the table and selecting **Export Table Data as CSV**. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) in the *Workbench User's Guide*. Once the raw data is exported, you can import it as a custom DOE.

You can also select one or more rows in the table and right-click to select one of the following options from the context menu:

- **Insert as Design Point:** Creates new design points in the project by copying the input parameter values of the selected candidate points to the design points table for the **Parameter Set** bar. The output parameter values are not copied because they are approximated values provided by the response surface.
- **Insert as Custom Candidate Point:** Creates new custom candidate points in the candidate points table by copying the input parameter values of the selected candidate points.

Note:

The design point data is in raw format, which means it is displayed without analysis or optimization results. Consequently, it does not show feasibility, ratings, Pareto fronts, and so on.

Design Point Log Files

DesignXplorer saves design point update data. As each design point is solved, DesignXplorer writes its full definition, which includes both input and output parameter values, to a design point log file. This occurs when you take any of the following actions:

- Update the current design point by right-clicking it and selecting **Update Selected Design Point**.

Note:

The **Update Project** operation is processed differently than a design point update. When you update the whole project, design point data is not logged. You must use the right-click context menu to log data on the current design point.

- Update either selected design points or all design points using the **Update Selected Design Point** option on the right-click context menu.
- Update a design exploration cell such as **Design of Experiments**.
- Open an existing project. For all up-to-date design points in the project data is immediately logged.

Formatting

The generated log file is in the extended CSV file format used to export table and chart data and to import data from external CSV files to create new design, refinement, and verification points. While this file is primarily formatted according to CSV standards, it supports some non-standard formatting conventions. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) and [Exporting Design Point Parameter Values to a Comma-Separated Values File](#).

File Location

The log file is named `DesignPointLog.csv` and is written to the directory `user_files` for the Workbench project. You can locate the file in the **Files** pane of Workbench by selecting **View** → **Files**.

Importing the File

Because the design point log file is in the extended CSV file format used elsewhere by DesignXplorer, you can import the design point data back into the design points table for the **Design of Experiments** cell of any design exploration system.

To import data from the design point log file, you set **Design of Experiments Type** to **Custom**. The list of parameters in the file must exactly match the order and parameter names in DesignXplorer. For example, the order and names might be **P1**, **P7**, and **P3**.

To import the log file, you might need to manually extract a portion.

Manually Extracting Part of the Log File

1. Identify the column order and parameter names in use by DesignXplorer. You can do this by either of the following methods:
 - Review the column order and parameter names in the header row of the **Table** pane.
 - Export the first row of your custom DOE to create a file with the correct order and parameter names for the header row.
2. Find the file `DesignPointLog.csv` in the directory `user_files` for the project and then compare it to your exported DOE file. Verify that the column order and parameter names exactly match those in DesignXplorer.
3. If necessary, update the column order and parameter names in the design point log file.

If parameters were added or removed from the project, the file has several blocks of data to reflect this that are distinguished by header lines.

4. Manually remove any unnecessary sections from the file, keeping only the block of data that is consistent with your current parameters.

Note:

The header line is produced when the log file is initially created and reproduced when a parameter is added or removed from the project. If parameters have been added or removed, you must verify the match between DesignXplorer and the log file header row again.

Importing the Log File into the Design Points Table

1. In the **Properties** pane for the **Design of Experiments** cell, verify that **Design of Experiments Type** is set to **Custom**.
2. Right-click any cell in the design points table and select **Import Design Points** and then **Browse**.
3. Browse to the directory `user_files` for the project and select the file `DesignPointLog.csv`.

The design point data is loaded into the design points table.

4. Update the DOE.

Failed Design Points

This section provides recommendations on how to prevent design point failures from occurring, instructions on preserving design point data for future troubleshooting, and various methods that you can use to deal with design points once they have failed:

[Preventing Design Point Update Failures](#)

[Preserving Design Points and Retaining Data](#)

[Handling Failed Design Points](#)

Preventing Design Point Update Failures

Common causes of failed design points include licensing issues, IT issues (such as networking or hardware issues), a lack of robustness in the geometry, mesh, or solver, and parametric problems or conflicts. By taking a proactive approach, you can possibly prevent failures from occurring during design point updates. This section offers recommendations to help you increase the probability of successful design point updates.

Minimize potential licensing and IT issues

Before doing a design point update, you can minimize design point failures by addressing issues that are not directly related to your project or its contents, but that could cause problems during the update.

Check your network connections

Check your network connections to verify that everything is in order.

Set power options for your computer to always be on

In the power options for your computer, configure your computer to always remain turned on. Set options for the time after which to turn off the hard disk, sleep, and hibernate to **Never**. This can avoid update issues that are caused by the computer timing out or shutting down during an update.

Verify that licenses are available

If you're using CAD, select **Tools** → **Options** → **Geometry Import** and set **CAD Licensing** to **Hold**. This places a hold on the license currently being used so that it is available for the update. When the update finishes, you can set **CAD Licensing** back to **Release**, which is the default.

Leave the modeling application open

Don't close your modeling application while you are running DesignXplorer. For example, using the "Reader" mode in CAD can create update issues because the editors are closed by default.

Restart Ansys Mechanical or Ansys Meshing

By default, Ansys Mechanical and Ansys Meshing restart after each design point. This default value lengthens the overall processing time but improves overall system performance (memory and CPU) when the generation steps of each design point (geometry, mesh, solve, and postprocessing) are lengthy.

If overall system performance is not a concern, you can reduce the overall processing time by directing Ansys Mechanical and Ansys Meshing to restart less frequently or not at all. Select **Tools** → **Options** and then in the **Mechanical** and **Meshing** tabs, under **Design Points**, do one of the following:

- To restart less frequently, set the number of design points to update before restarting to a higher value, such as 10.
- To prevent restarts completely, clear the check box for periodically restarting during a design point update.

Create a robust model

To create a robust model, review both the parameterization of your model (for instance, CAD or Ansys DesignModeler) and your DesignXplorer settings.

Keep only what is being used

Only expose the parameters that you are actively optimizing. Additionally, turn off surfaces that aren't being used, coordinate system imports, and so on.

Identify and correct potential parametric conflicts

While design exploration provides some safeguards against parametric conflicts and poorly defined problems, it cannot identify and eliminate all potential issues. Give careful consideration to your parametric setup, including factors such as ranges of variation, to make sure that the setup is reasonable and well-defined.

Check your mesh

If you have geometry parameters, double-check your mesh to avoid any simulation errors that can be caused by meshing. Pay particular attention to local refinement and global quality. If you are using Ansys Mechanical or Fluent, use the **Morphing** option when possible.

Use the Box-Behnken Design DOE type

Consider setting **Design of Experiments Type** to **Box-Behnken Design** if your project has parametric extremes (such as parameter values in corners that are difficult to build) and has 12 or fewer continuous input parameters. Because this DOE type doesn't have corners and does not combine parametric extremes, it can reduce the risk of update failures.

Test the robustness of the model

Before you submit multiple design points or all design points for update, test the model and verify the robustness of your design.

Verify model parameters by running a test project

Design points often fail because the model parameters cannot be regenerated. To test the model parameters, try creating a project with a test load and coarse mesh that runs quickly. Solve the test project to verify the validity of the parameters.

Verify model parameters by submitting a Geometry-Only update to Remote Solve Manager

You have the ability to submit a **Geometry-Only** update for all design points to DesignXplorer. In the **Properties** pane for the **Parameter Set** bar, set **Update Option** to **Submit to Remote Solve Manager** and set **Pre-RSM Foreground Update** to **Geometry**. When you submit the next update, the geometry is updated first, so any geometry failures are found sooner.

If necessary, re-parameterize the model

Occasionally, all of your parameters appear to be feasible, but the model still fails to update due to issues in the history-based CAD tool. In this case, you can re-parameterize the model in Ansys SpaceClaim or a similar direct modeling application.

Verify that the current design point updates successfully

Try to update the current design point before submitting a full design point update. If the update fails, you can open the project in an Ansys product to further investigate this design point. For more information, see [Handling Failed Design Points \(p. 311\)](#).

Attempt to update a few "extreme" design points

Select a few design points with extreme values and try to update them. For example, select the design points with the smallest gap and largest radius and try to update them. In this way, you can assess the robustness of your design before committing to a full correlation.

Preserving Design Points and Retaining Data

By default, when a design point other than the current design point fails, DesignXplorer saves the design point but does not retain any of its files. Calculated data is saved on disk only for the current design point. Before performing a full update, you might want to configure DesignXplorer to preserve design points and retain their data.

Caution:

Because every design point is retained within the project, this approach can affect performance and disk resources when used on projects with 100 or more design points.

The preservation of design points and design point data can be defined as the default behavior at the project level or can be configured for individual cells.

- To set this as the default behavior at the project level, select **Tools** → **Options** → **Design Exploration**. Under **Design Points**, select both the **Preserve Design Points After DX Run** and **Retain Data for Each Preserved Design Point** check boxes.
 - When you opt to preserve the design points, the design points are added to the design points table for the **Parameter Set** bar.
 - When you opt to also retain the design point data, the calculated data for each design point is saved in the project.
- To configure this functionality at the component level, right-click the cell and select **Edit**. In the **Properties** pane, under **Design Points**, select both the **Preserve Design Points After DX Run** and **Retain Data for Each Preserved Design Point** check boxes.

Once data has been retained for a design point, you can set it as the current design point. This enables you to review the associated design within the project and further investigate any update problems that might have occurred.

Handling Failed Design Points

This section contains information and suggestions on investigating and handling design points once they've failed. The strategy to use depends on how many design points failed and the nature of the failure.

Perform Preliminary Troubleshooting Steps

Review error messages.

The first step in gathering information about the update issue is to review error messages. In DesignXplorer, a failed design point is considered to be completely failed. In reality, though, it might be a partial failure, where the design point failed to update for only a single output. As such, you need to know which output is related to the failure.

You can find this information by hovering your mouse over a failed design point in the design points table. A primary error message tells you what output parameters have failed for the design point and specifies the name of the first failed component. Additional information might also be available in the **Messages** pane.

Attempt to update all design points once again.

Sometimes a design point update fails because of a short-term issue, such as a license or CPU resource being temporarily unavailable. The issue could actually have been remedied in the time between your first update attempt and a second attempt. Try again to update all design points before proceeding further.

In **Tools** → **Options** → **Design Exploration**, the **Retry Failed Design Points** check box globally sets whether DesignXplorer is to make additional attempts to solve all design points that failed during the previous run. This check box is applicable to all DesignXplorer systems except **Six Sigma Analysis** systems and **Parameters Correlation** systems that are linked to response surfaces.

When **Retry Failed Design Points** is selected, **Number of Retries** and **Retry Delay** become available so that you can specify the number of times that the update should be retried and the delay in seconds between each attempt.

You can override this global option in the **Properties** pane for a cell. In the **Properties** pane, under **Failed Design Points Management**, setting **Number of Retries** to **0** disables automatically retrying the update for failed design points. When any other integer value is set, **Retry Delay** is available so that you can specify the delay in seconds between each attempt.

Address licensing and IT issues.

Error messages can indicate that the failure was caused by factors external to DesignXplorer, such as network, licensing, or hardware issues. For example, the update can fail because a license was not available when it was needed or because a problem existed with your network connectivity. If the issue isn't remedied by your second attempt to update, address the issue in question and then retry the update.

Create a Duplicate Test Project

When preparing to investigate one or more failed design points, remember that editing the current project during your investigation could invalidate other design points and your design exploration results.

To avoid potentially invalidating the original project, you should create a duplicate test project using the **Save As** menu option. You can either duplicate the entire project or narrow your focus by creating separate projects for one or more failed design points. To create separate projects, select

the failed design points in the table. Then, right-click any of these selections and select **Export Selected Design Points**.

Retain the Failed Design Point

By default, with a design point update, Workbench retains calculated data for only the current design point. To investigate a failed design point, you can mark it as **Retained** in the design points table for the **Parameter Set** bar. Once a design point has retained data, you can use it in several ways to explore update failures.

Set the failed design point as current

Once a design point has retained data, you can right-click it and select **Set as Current**. You can then open the project in an editor to troubleshoot the update issue. For more information, see [Retaining Data for Generated Design Points \(p. 302\)](#)

Export a failed design point

To further investigate one or more failed design points without invalidating other design points or design exploration results, you can create a duplicate test project for the failed design points. Right-click the one or more failed design points with retained data and select **Export Selected Design Points**. With the next update, each failed design point that you've selected is exported to a separate project. For more information, see [Exporting Design Points to New Projects](#) in the *Workbench User's Guide*.

Working with Sensitivities

The sensitivity charts available in design exploration allow you to understand how sensitive the output parameters are to the input parameters. This understanding can help you innovate toward a more reliable and better quality design or to save money in the manufacturing process while maintaining the reliability and quality of your product. You can request a sensitivity plot for any output parameter in your model.

The sensitivities available for Six Sigma Analysis and goal-driven optimizations are statistical sensitivities, which are global sensitivities. The single parameter sensitivities available for response surfaces are local sensitivities.

Global, statistical sensitivities are based on a correlation analysis using the generated sample points, which are located throughout the entire space of input parameters.

Local parameter sensitivities are based on the difference between the minimum and maximum value obtained by varying one input parameter while holding all other input parameters constant. As such, the values obtained for local parameter sensitivities depend on the values of the input parameters that are held constant.

Global, statistical sensitivities do not depend on the values of the input parameters because all possible values for the input parameters are already taken into account when determining the sensitivities.

Global Sensitivity Chart Limitations

- Sensitivities are calculated only for continuous parameters.

- Sensitivity charts are not available if all input parameters are discrete.

For more information, see:

[Using Local Sensitivity Charts \(p. 157\)](#)

[Using the Sensitivities Chart \(GDO\) \(p. 231\)](#)

[Statistical Sensitivities in a SSA \(p. 398\)](#)

Working with Tables

The **Table** pane displays tabular data in one or several tables. Some tables are read-only, populated with data that DesignXplorer generates. Other tables are partially or completely editable.

The background color of a cell indicates if it is editable or not:

- A gray background indicates a read-only cell.
- A white background indicates an editable cell.

You can add a new row to a table by entering values in the * row. You enter values in columns for input parameters. Once you enter an input value in the * row, the row is added to the table and the remaining input parameters are set to their initial values. You can then edit this row in the table, changing input parameter values as needed. Output parameter values are calculated when the cell is updated.

Output parameter values calculated from a design point update are displayed in black text. Output parameter values calculated from a response surface are displayed in the custom color specified in **Tools** → **Options** → **Design Exploration** → **Response Surface**. For more information, see [Response Surface Options \(p. 38\)](#).

Viewing Points in the Table Pane

Points shown in the **Table** pane for a cell can correspond to design points in the **Table** pane for the **Parameter Set** bar. DesignXplorer points include DOE points, refinement points, direct correlation points, candidate points, and more. Points correspond when they share the same input parameter values. When a correspondence exists, the name of the point indicates the design point to which it is related.

If the source design point is deleted from the **Parameter Set** bar or the definition of either design point is changed, the link between the two points is broken without invalidating your model or results. Additionally, the indicator is removed from the name of the point.

Editable Output Parameter Values

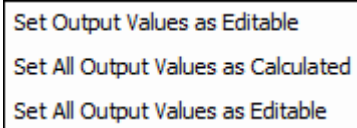
For the following tables, you can make output parameter values editable:

- Design points table for a **Design of Experiments** cell or **Design of Experiments (3D ROM)** cell when the **Design of Experiments Type** property is set to either **Custom** or **Custom + Sampling**
- Refinement points table for a **Response Surface** cell

- Verification points table for a **Response Surface** cell

Because output values are provided by a real solve, editing output values is not normally necessary. However, you might want to insert existing results from external sources such as experimental data or known designs. Rows with editable output values are not calculated during an update of the DOE.

By default, output values are calculated during an update. However, you can make output values in the previously specified tables editable using options on the context menu:



- **Set Output Values as Editable** makes the output values editable in selected rows. If you right-click one or more rows with editable output values, you can select **Set Output Values as Calculated** if you want output values to be set as calculated once again. However, this invalidates the DOE, requiring you to update it.
- **Set All Output Values as Calculated** sets output values for all rows as calculated.
- **Set All Output Values as Editable** makes the output values editable in all rows.
 - If you enter an input value in the * row, the row is added to the table and the remaining input parameters are set to their initial values. The output parameter values are blank. You must enter values in all columns before updating the DOE.
 - If you enter an output value in the * row, the row is added to the table and the values for all input parameters are set to their initial values. The remaining output parameter values are blank. You must enter values in all columns before updating the DOE.

Note:

- A table can contain derived parameters. Derived parameters are always calculated, even if **Set All Output Values as Editable** is selected.
 - Editing output values for a row changes the cell's state to indicate that an update is required. The cell must be updated, even though no calculations are done.
 - If the points are solved and you select **Set All Output Values as Editable** and then select **Set All Output Values as Calculated** without making any changes, the outputs are marked out-of-date. You must update the cell to recalculate the points.
-

Copying and Pasting Data

You can copy data from a spreadsheet, text file, or table and paste it into any DesignXplorer table that is editable. You can also copy data from any DesignXplorer table and paste it into a spreadsheet, text file, or table.

To manipulate the data in a large number of table rows, you should use the export and import capabilities that are described in the next two sections.

Exporting Table Data

You can export all data in a table to a CSV file. Simply right-click the table and select **Export Table Data as CSV**, enter a filename, and export. For more information, see [Exporting Design Point Parameter Values to a Comma-Separated Values File](#) in the *Workbench User's Guide*.

Importing Data from a CSV File

You can import data from an external CSV file to set up a specific DOE type from an external tool or to use existing results as verification points. The imported data can contain input parameter values and, optionally, output parameter values.

The import option is available from the context menu when you right-click a cell in the **Project Schematic**. It is also available when you right-click in the **Table** pane for a cell where the import feature is implemented. For example, you can right-click a **Response Surface** cell in the **Project Schematic** and select **Import Verification Points**. Or, you can right-click in the **Table** pane for a **Design of Experiments** cell or a **Design of Experiments (3D ROM)** cell and select **Import Design Points from CSV**.

Note:

For the **Import Design Points from CSV** option to be available for a DOE cell, the **Design of Experiments Type** property must be set to **Custom** or **Custom + Sampling**.

For example, assuming that the DOE cell is set to a custom DOE type, to import design points from an external CSV file into the DOE cell, do the following:

1. In the **Project Schematic**, right-click the DOE cell into which to import design points and select **Import Design Points from CSV**.
2. Select the CSV file and click **Open**.

During the import, the CSV file is parsed and validated. If the format is invalid or the described parameters are not consistent with the current project, a list of errors is displayed and the import operation is terminated. If the file validates, the data is imported.

The main rules for imports follow.

- The file must conform to the extended CSV file format. In particular, a header line identifying each parameter by its ID (P1, P2, ..., P_n) is mandatory to describe each column. For more information, see [Extended CSV File Format](#) in the *Workbench User's Guide*.
- The order of the parameters in the file might differ from the order of the parameters in the project.
- Disabled input parameters must not be included.
- Values must be provided in the units defined, which you can see by selecting **Units** → **Display Values As Defined**.

- If values for output parameter are provided, they must be provided for all output parameters except derived output parameters. Any values provided for derived output parameters are ignored.
- Even if the header line states that output parameter values are provided, it is possible to omit them on a data line.
- If parameter values for imported design points are out of range or do not fill the range, a dialog box opens, giving you options for automatically expanding and shrinking the parameter ranges in the DOE to better fit the imported values. This same dialog box can also appear when you are copying design points from the **Parameter Set** bar into a DOE. For more information, see [Parsing and Validation of Design Point Data \(p. 99\)](#).
- Once design points are imported, any provided output parameter values are editable. For design points imported without output parameter values, values are read-only. You must either update the DOE cell or [set output parameter values as editable \(p. 314\)](#) and then enter values to continue.

Working with Remote Solve Manager and DesignXplorer

Remote Solve Manager (RSM) is a job queuing and resource management system that distributes tasks to available local and remote computing resources. When you submit a job to RSM, it can either be queued to run in the background of the local machine or to one or more remote machines. For more information, see the [Remote Solve Manager User's Guide](#).

To send updates via RSM, you must first install and configure RSM. For more information, see the [Installation and Licensing Help and Tutorials](#) page on the Ansys customer site. From the customer site menu, select **Downloads** → **Installation and Licensing Help and Tutorials**. For general information about materials and services available to our customers, go to the [main page](#) of the customer site.

Submitting Design Point Updates to RSM from DesignXplorer

You can configure DesignXplorer to send design point updates to RSM. This configuration can be different than the configuration for solution cell updates, so it is possible to send design point updates and solution cell updates to different locations for processing. For example, design point updates might be submitted to RSM for remote processing while the solution cell update is run in the background of the local machine. Or, the solution cell update might be submitted to RSM while the design point updates are run in the foreground of the local machine.

For more information on submitting design points for remote update, see [Updating Design Points in Remote Solve Manager](#) in the *Workbench User's Guide*.

For more information on configuring the solution cell update location, see [Solution Process](#) in the *Workbench User's Guide*.

For more information on configuring the design point update option, see [Setting the Design Point Update Option](#) in the *Workbench User's Guide*.

Previewing Updates

When a **Design of Experiments**, **Response Surface**, or **Parameters Correlation** cell requires an update, in some cases, you can preview the results of the update. A preview prepares the data and displays it without updating the design points. This allows you to experiment with different settings and options before actually solving a DOE or generating a response surface or parameters correlation.

- In the **Project Schematic**, either right-click the cell to update and select **Preview** or select the cell and click **Preview** on the toolbar.
- In the **Outline** pane for the cell, either right-click the root node and select **Preview** or select this node and click **Preview** on the toolbar.

Pending State

Once a remote update for a **Design of Experiments**, **Response Surface**, or **Parameters Correlation** cell begins, the cell enters a pending state. However, some degree of interaction with the project is still available. For example, you can:

- Open the **Options** window, access the component tab for the **Parameter Set** bar, and archive the project.
- Follow the overall process of an update in the **Progress** pane by clicking **Show Progress** in the lower right corner of the window.
- Interrupt, abort, or cancel the update by clicking the red stop button to the right of the progress bar in the **Progress** pane and, in the dialog box that then opens, clicking the button for the desired operation.
- Follow the process of individual design point updates in the **Table** pane for the cell. Each time a design point is updated, the **Table** pane displays the results and status.
- Exit the project and either create a new project or open another existing project. If you exit the project while it is in a pending state due to a remote design point update, you can later reopen it. The update will then resume automatically.

Exit a project while a design point update or a solution cell update via RSM is in progress. For information on behavior when exiting during an update, see [Updating Design Points in Remote Solve Manager](#) or [Exiting a Project During a Remote Solve Manager Solution Cell Update](#) in the *Workbench User's Guide*.

Note:

- For iterative update processes such as Kriging refinement, Sparse Grid response surface updates, and Correlation with Auto-Stop, design point updates are sent to RSM iteratively. The calculations of each iteration are based on the convergence results of the previous iteration. Consequently, if you exit Workbench during the pending state for an iterative process, when you reopen the project, only the current iteration is completed.
- The pending state is not available for updates that contain verification points or candidate points. You can still submit these updates to RSM. However, you do not receive intermediate results during the update process, and you cannot exit the project. Once the update

is submitted via RSM, you must wait for the update to complete and the results to be returned from the remote server.

For more information, see [Working with Ansys Remote Solve Manager](#) and [User Interface Overview](#) in the *Workbench User's Guide*.

Working with Design Point Service and DesignXplorer

Design Point Service (DPS) is part of Distributed Compute Services (DCS). DPS quickly and efficiently evaluates tens of thousands of design points using multiple compute resources. It supports submission of design point updates on local machines and on HPC clusters using Remote Solve Manager (RSM).

To use DPS, in the properties for either the **Parameter Set** bar or a DesignXplorer system cell, you set **Update Option** to **Submit to Design Point Service (DPS)**. Any update of design points then sends the Workbench project definition and unsolved design points to DPS for evaluation.

For more information, see the [DCS for Design Points Guide](#) and the following topics:

[Sending an Optimization Study to DPS](#)

[Importing DPS Design Points into DesignXplorer](#)

Sending an Optimization Study to DPS

A goal-driven optimization is a study of design goals for generating optimized designs. In the study, you define design goals in the form of [objectives and constraints \(p. 205\)](#). To successfully submit an update for an **Optimization** cell to DPS, in the properties for either the **Parameter Set** bar or the **Optimization** cell, **Update Option** must be set to **Submit to Design Point Service (DPS)**. Additionally, the **Optimization** cell must have:

- Tolerance values for objectives of the **Seek Target** type
- Tolerance values for constraints
- Initial values for objectives

If these values are set and are consistent, when the update is submitted to DPS, DesignXplorer automatically sends the objectives and constraints for the study to the DPS project as *fitness terms*. If these values are either not set or are inconsistent, DesignXplorer cannot send the objectives and constraints, which means DPS cannot start the update.

If the **Optimization** cell is already up-to-date, you can manually send objectives and constraints for the study to the DPS project. In the **Outline** pane of the **Optimization** cell, right-click **Optimization** and select **Send Study to DPS Project**. The **Messages** pane indicates if the study has been sent successfully.

In the DPS web app, the project's **Configuration** tab displays a **Fitness** heading, under which you can see any fitness terms sent from DesignXplorer, along with any fitness terms that are entered directly in the DPS project. On the project's **Design Points** tab, you can easily sort design points by their

calculated fitness values to find the best candidates. For more information, see [Defining Fitness](#) in the *DCS for Design Points Guide*.

Note:

Changes to fitness terms do not invalidate evaluated DPS design points but rather trigger the re-evaluation of fitness values for all DPS design points.

Importing DPS Design Points into DesignXplorer

From a Workbench project, you can import design points that have been updated in DPS into these editable DesignXplorer tables for reuse:

- Custom parameters correlations
- Custom DOEs
- Refinement points tables for response surfaces
- Verification points tables for response surfaces
- Custom candidate points tables for optimizations

You can also import design points updated in DPS into the design points table for the Workbench **Parameter Set** bar. For more information, see [Importing DPS Design Points into the Workbench Project](#) in the *DCS for Design Points Guide*.

Working with DesignXplorer Extensions

In addition to using the features provided by DesignXplorer, you can customize and extend DesignXplorer functionality via extensions created with Ansys ACT. By installing and loading extensions for external sampling or optimization algorithms, you integrate their features into your design exploration workflow.

[Locating and Downloading Available Extensions](#)

[Installing a DesignXplorer Extension](#)

[Loading a DesignXplorer Extension](#)

[Selecting an External Optimizer or DOE](#)

Note:

For information on using ACT, see the *Ansys ACT Developer's Guide*, the *Ansys ACT XML Reference Guide*, and the *Ansys ACT Reference Guide*.

Locating and Downloading Available Extensions

You can use custom extensions that are developed by your company and also download extensions from the [Ansys Store](#).

Installing a DesignXplorer Extension

To use a DesignXplorer extension, you must first install it to make it available to Ansys Workbench.

To install an extension:

1. From the Workbench **Project** tab, select **Extensions** → **Install Extension**.
2. Browse to the location of the extension that you want to install.

In most cases, you select a binary extension in which a defined optimizer is compiled into a WBEX file. WBEX files cannot be modified.

3. Select the extension and click **Open**.

The extension is installed in your `App Data` directory. Once installed, it is shown in the **Extensions Manager** and can be loaded to your projects.

Loading a DesignXplorer Extension

Once a DesignXplorer extension is installed, you can load it to your Ansys Workbench project so that the functionality defined by the extension becomes available in DesignXplorer. To load an extension to your project:

1. From the Workbench **Project** tab, select **Extensions** → **Manage Extensions**. The **Extensions Manager** opens, showing all installed extensions.
2. Select the check box for the extension to load and then close the **Extensions Manager**.

The extension should now be loaded to the project, which means that it is available to be selected as an optimization method. You can select **Extensions** → **View Log File** to verify that the extension loaded successfully.

Note:

The extension must be loaded separately to each project unless you have specified it as a default extension in the **Options** window. You can also specify whether loaded extensions are saved to your project. For more information, see [Extensions](#) in the *Workbench User's Guide*.

Selecting an External Optimizer or DOE

Once an extension is loaded to a project, the optimizers or DOEs defined in the extension are available in a drop-down menu in the **Properties** pane. DesignXplorer filters the menu options applicable to the current project, displaying only those options that can be applied to the problem as it is currently defined. For more information, see [External Design of Experiments \(p. 89\)](#) and [Performing an Optimization with an External Optimizer \(p. 200\)](#).

In some cases, a necessary extension might not be available when a project is reopened. For example, the extension could have been unloaded or the extension might not have been saved to the project upon exit. When this happens, the option in the drop-down menu is replaced with an alternate label,

<extensionname>@<optimizername> or <extensionname>@<doename> instead of the actual name of the external optimizer or DOE. You can still do postprocessing with the project for data obtained when the problem was solved previously. However, the properties are read-only and further calculations cannot be performed unless you reload the extension or select a different optimizer or DOE type.

- If you reload the extension, the properties become editable and calculations can be performed. You have the option of saving the extension to the project so it does not have to be reloaded again the next time the project is opened.
- When you select a different optimization method or sampling type (one not defined in the missing extension), the alternate label disappears from the list of options.

Working with Design Exploration Results in Workbench Project Reports

The Workbench project report includes results from design exploration systems. The contents and organization of the project report reflect the layout of the **Project Schematic** and the current state of the project.

Selecting **File** → **Export Report** generates the project report, which you can edit and save. For more information, see [Working with Project Reports](#) in the *Workbench User's Guide*.

The project report begins with a summary, followed by separate sections for global, system, and component information. The project report also has appendices.

Project Summary

The project summary includes the project name, date and time created, and product version.

Global Information

This section includes an image of the schematic and tables corresponding to the **Files**, **Outline of All Parameters**, **Design Points**, and **Properties** panes.

System Information

A system information section exists in the project report for each design exploration system in the project. For example, a project with **Response Surface** and **Response Surface Optimization** systems has two system information sections.

Component Information

Each system information section contains subsections for the components (cells) in the system. For example, because a **Response Surface Optimization** system has three cells, its system information section has three subsections. Because a **Parameters Correlation** system has only one cell, its system information section has only one subsection. Each component information subsection contains such data as parameter or model properties and charts.

Appendices

The final section of the project report consists of matrices, tables, and additional information related to the project.

DesignXplorer Theory

When performing a design exploration, a theoretical understanding of the methods available is beneficial. The underlying theory of the methods is categorized as follows:

[Parameters Correlation Filtering Theory](#)

[Response Surface Theory](#)

[Goal-Driven Optimization Theory](#)

[Six Sigma Analysis \(SSA\) Theory](#)

[Theory References](#)

Parameters Correlation Filtering Theory

To filter the major and the minor input parameters, DesignXplorer takes into accounts three criteria:

- The relevance of the correlation value between input and output parameters
- The R^2 contribution of input parameters in prediction of output parameter values (gain in prediction)
- The maximum number of major inputs

Relevance of the Correlation Value

For each pair of input-output parameters, DesignXplorer computes different kinds of correlation:

- Pearson Correlation (linear relationship)
- Spearman Correlation (monotonic relationship)
- Quadratic Correlation (quadratic relationship)

For each correlation value, DesignXplorer computes the p-value of this correlation.

The p-value of a correlation value allows DesignXplorer to quantify the relevance of the correlation.

The following notation is used in equations:

- n = number of sample points
- r_{XY} = observed correlation between X (an input parameter) and Y (an output parameter)
- ρ_{XY} = real correlation value between X and Y
- H_0 = null hypothesis for $r_{XY}; \rho_{XY} = 0$

The t test of the null hypothesis is:

$$t = \frac{r_{XY}\sqrt{n-2}}{\sqrt{1-r_{XY}^2}}$$

The variable t is approximately distributed like a Student's Distribution $A(t,df)$ where $df=n-2$ degrees of freedom.

The p-value corresponds to the probability of obtaining the same r_{XY} assuming that the null hypothesis is true. The p-value is given by $1-A(t,df)$.

When trying to find a relationship between X and Y , two types of errors are possible:

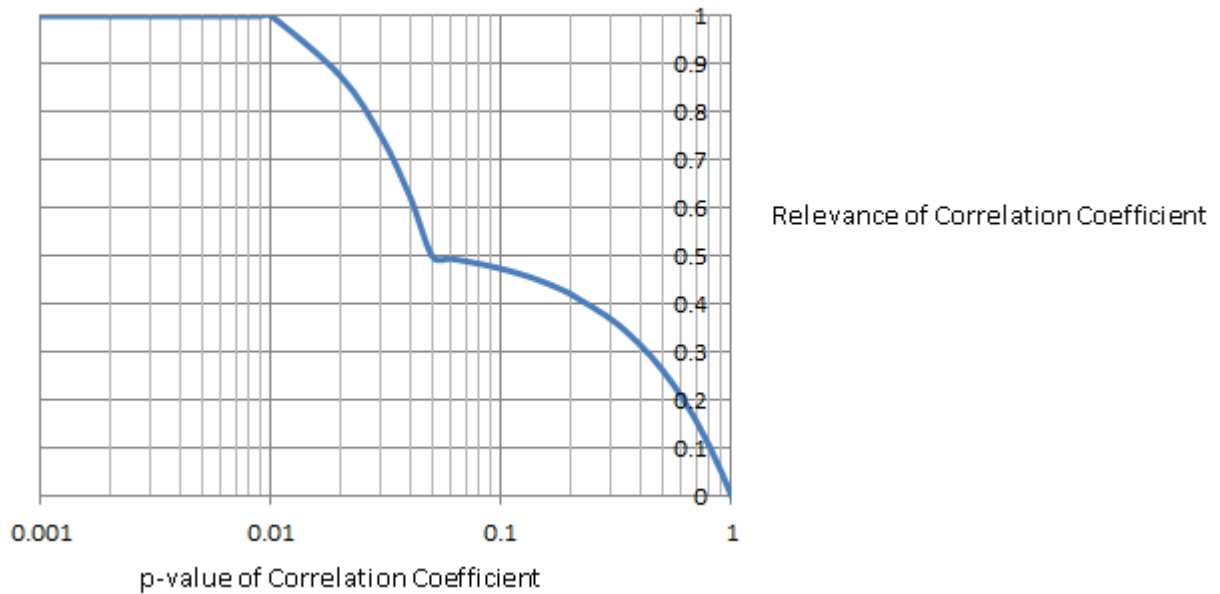
- **Type I Error** (false positive): You think that there is a relationship between X and Y , when there is in fact no relation between X and Y .
- **Type II Error** (false negative): You think that there is no relationship between X and Y , when there is in fact a relationship between X and Y .

H0: There is no relationship between X & Y		Reality	
		H0 True Y does not depend on X	H0 False Y depends on X
Decision	H0 True Y does not depend on X	RIGHT DECISION	Error β = H0 is accepted when H0 is false (false negative)
	H0 False Y depends on X	Error α = H0 is rejected when H0 is true (false positive)	RIGHT DECISION

The p-value corresponds to the α value.

The larger the correlation value r_{XY} , the less likely it is that the null hypothesis is true.

The relevance value of the correlation value exposed at the DesignXplorer level is a transformation of the p-value from [0; 1] to [0; 1]:



When running a filtering method with a relevance threshold equal to 0.5 on the correlation value only, the major input parameters selected must be at least a p-value of the correlation coefficient with any filtering output parameter equals to or less than 0.05.

R² Contribution

For each filtering output parameter, DesignXplorer builds an internal meta-model (polynomial response surface) based on the sample points generated during the correlation update.

A first response surface is built based on major input parameter returned by the filtering by using the correlation values.

Once a first response surface has been built, DesignXplorer tests the contribution of each input parameter in the prediction of the output parameter, and refines the response surface by removing the insignificant input parameters and by adding new significant input parameters.

The following notation is used in equations:

- n = number of sample points
- R^2 = R-squared of the response surface M based on current major input parameters
- R_{-i}^2 = R-squared of the response surface M_{-i} based on current major input parameters without the i -th major input parameter
- R_{+i}^2 = R-squared of the response surface M_{+i} based on current major input parameters and the i -th minor input parameter
- k = number of terms used in the regression of the response surface M

- k_{-i} = number of terms used in the regression of the response surface M_{-i}
- k_{+i} = number of terms used in the regression of the response surface M_{+i}

DesignXplorer computes the Fisher Test to test the significance of major input parameter and the gain in prediction:

$$F = \frac{R^2 - R_{-i}^2}{1 - R^2} \times \frac{n - k}{k - k_{-i}}$$

Under the null hypothesis that response surface M does not provide a significantly better R-squared than the response surface M_{-i} , F has an F distribution, with $(k - k_{-i}, n - k)$ degrees of freedom.

The null hypothesis is rejected if the F calculated from the data is greater than the critical value of the F distribution for the false-rejection probability equals to 0.05. This means that if the p-value of the Fisher Test is greater than 0.05, the major input parameter becomes a minor input parameter.

DesignXplorer unselects all major input parameters when $R^2 - R_{-i}^2 < 0.01$.

DesignXplorer computes the Fisher Test to test the significance of major input parameter and the gain in prediction:

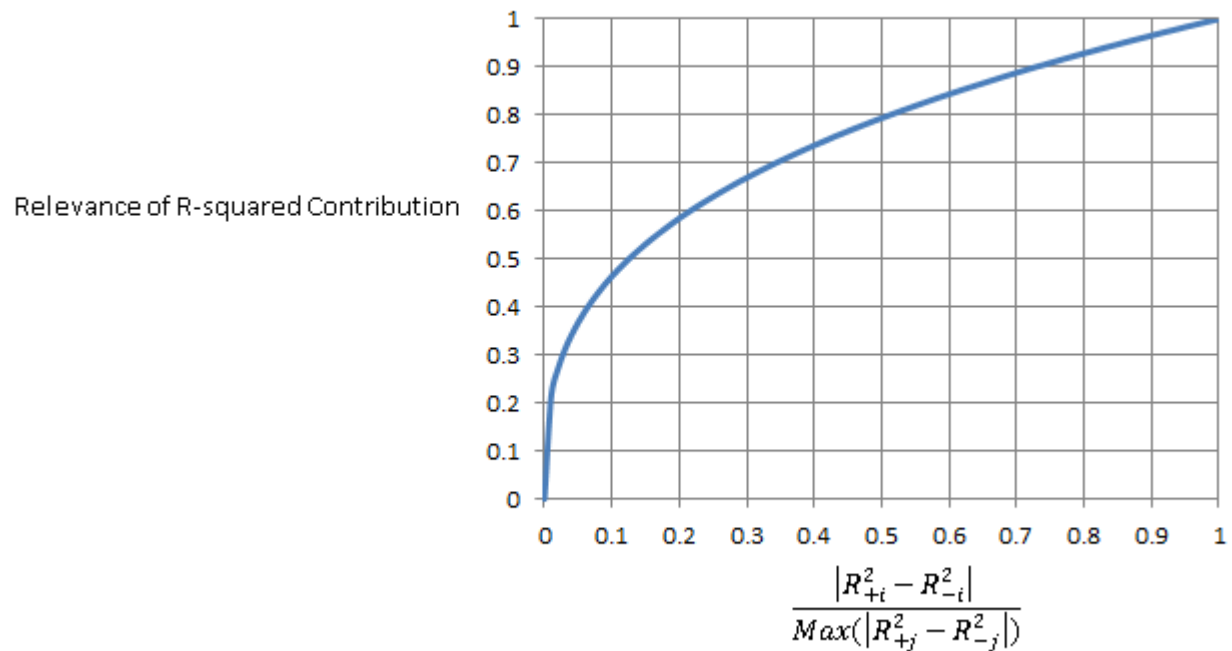
$$F = \frac{R_{+i}^2 - R^2}{1 - R_{+i}^2} \times \frac{n - k_{+i}}{k_{+i} - k}$$

Under the null hypothesis that response surface M_{+i} does not provide a significantly better R-squared than the response surface M , F has an F distribution, with $(k_{+i} - k, n - k_{+i})$ degrees of freedom.

DesignXplorer promotes a minor input parameter as a major input parameter when $R_{+i}^2 - R^2 > 0.01$ (1%) and when the p-value of the Fisher Test is less than 0.05 (5%).

For each output parameter, DesignXplorer computes the R^2 contribution of each input parameter (R_{-i}^2 or R_{+i}^2).

For a given output, to compute the relevance of R^2 contribution, DesignXplorer scales R^2 contribution by the best known R^2 contribution for this output, and transforms this value with the following function:



Maximum Number of Major Inputs

When several filtering output parameters are selected, the list of major and minor input parameters for each filtering output can be different. This means that you can control the maximum number of major inputs.

The following notation is used in equations:

- $Rel_{i,j}$ = relevance of the i -th input parameter on the j -th filtering output parameter
- $Rel_i = \max_j (Rel_{i,j})$ = best relevance of the i -th input parameter for any filtering output parameter
- N = maximum number of major inputs

The list of major inputs corresponds to the input parameters with its relevance $Rel_i > \text{Relevance Threshold}$.

If the size of the list exceeds N input parameters, DesignXplorer selects only the N input parameters with the best relevance.

Response Surface Theory

In a process of engineering design, it is very important to understand what and/or how many input variables are contributing factors to the output variables of interest. It is a lengthy process before a conclusion can be made as to which input variables play a role in influencing, and how, the output variables. Designed experiments help to revolutionize the lengthy process of costly and time-consuming

trial-and-error search to a powerful and cost-effective (in terms of computational time) statistical method.

A very simple designed experiment is the screening design. In this design, a permutation of lower and upper limits (two levels) of each input variable (factor) is considered to study their effect to the output variable of interest. While this design is simple and popular in industrial experimentations, it only provides a linear effect, if any, between the input variables and output variables. Furthermore, effect of interaction of any two input variables, if any, to the output variables is not characterizable.

To compensate for the insufficiency of the screening design, it is enhanced to include the center point of each input variable in experimentations. The center point of each input variable allows a quadratic effect, minimum or maximum inside the explored space, between input variables and output variables to be identifiable, if one exists. The enhancement is commonly known as response surface design to provide quadratic response model of responses.

The quadratic response model can be calibrated using full-factorial design (all combinations of each level of input variable) with three or more levels. However, the full-factorial designs generally require more samples than necessary to accurately estimate model parameters. In light of the deficiency, a statistical procedure is developed to devise much more efficient experiment designs using three or five levels of each factor but not all combinations of levels, known as fractional factorial designs. Among these fractional factorial designs, the two most popular DOE types (p. 330) are [Central Composite Designs \(CCDs\)](#) (p. 330) and [Box-Behnken designs](#) (p. 332).

Response surfaces are created using:

- [Genetic Aggregation](#) (p. 333)
- [Standard Response Surface - Full 2nd-Order Polynomials](#) (p. 339)
- [Kriging](#) (p. 340)
- [Non-Parametric Regression](#) (p. 344)
- [Sparse Grid](#) (p. 347)

Design of Experiments Types

This section provides the theory for DOE types:

[Central Composite Design \(CCD\)](#)

[Box-Behnken Design](#)

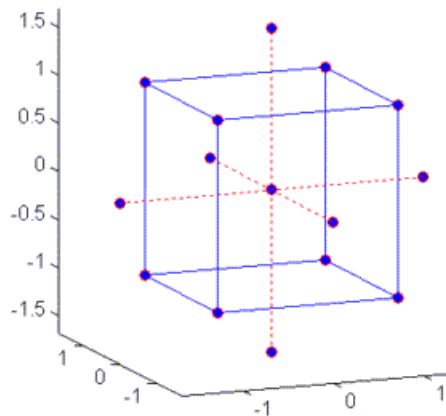
Central Composite Design (CCD)

Central Composite Designs (CCDs), also known as Box-Wilson Designs, are a five-level fractional factorial design that is suitable for calibrating the quadratic response model. There are three types of CCDs that are commonly used in experiment designs: circumscribed, inscribed, and face-centered CCDs.

The five-level coded values of each factor are represented by $[-\alpha, -1, 0, +1, +\alpha]$, where $[-1, +1]$ corresponds to the physical lower and upper limit of the explored factor space. It is obvious that $[-\alpha, +\alpha]$ establishes new "extreme" physical lower and upper limits for all factors. The value of α varies depending on design property and number of factors in the study. For the circumscribed

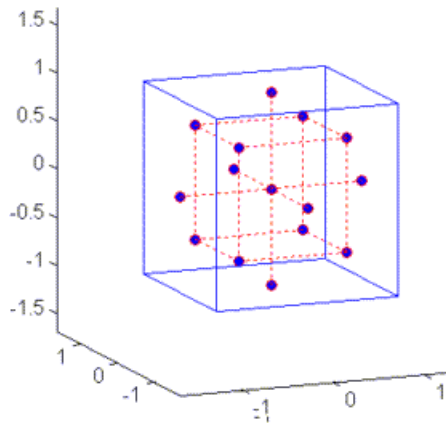
CCDs, considered to be the original form of CCDs, the value of α is greater than 1. The following is a geometrical representation of a circumscribed CCD of three factors:

Example 1: Circumscribed



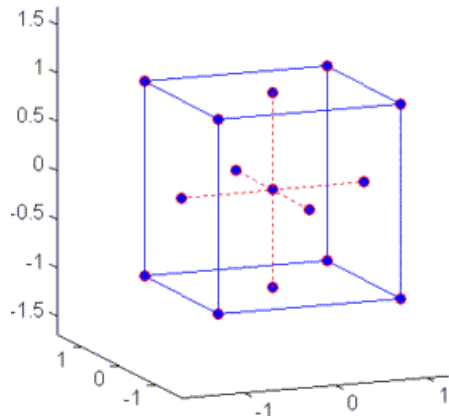
Inscribed CCDs, on the contrary, are designed using $[-1, +1]$ as "true" physical lower and upper limits for experiments. The five-level coded values of inscribed CCDs are evaluated by scaling down CCDs by the value of evaluated from circumscribed CCDs. For the inscribed CCDs, the five-level coded values are labeled as $[-1, -1/\alpha, 0, +1/\alpha, +1]$. The following is a geometrical representation of an inscribed CCD of three factors:

Example 2: Inscribed



Face-centered CCDs are a special case of CCDs in which $\alpha=1$. As a result, the face-centered CCDs become a three-level design that is located at the center of each face formed by any two factors. The following is a geometrical representation of a face-centered CCD of three factors:

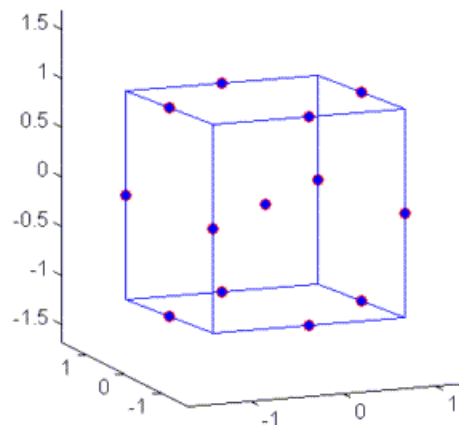
Example 3: Face-Centered



Box-Behnken Design

Unlike the CCD, the Box-Behnken Design is quadratic and does not contain embedded factorial or fractional factorial design. As a result, the Box-Behnken Design has a limited capability of orthogonal blocking, compared to CCD. The main difference of Box-Behnken Design from CCD is that Box-Behnken is a three-level quadratic design in which the explored space of factors is represented by $[-1, 0, +1]$. The "true" physical lower and upper limits corresponding to $[-1, +1]$. In this design, however, the sample combinations are treated such that they are located at midpoints of edges formed by any two factors. The following is a geometry representation of Box-Behnken designs of three factors:

Example 4: Box-Behnken Designs of Three Factors



The circumscribed CCD generally provides a high quality of response prediction. However, the circumscribed CCD requires factor level setting outside the physical range. Due to physical and economic constraints, some industrial studies prohibit the use of corner points, where all factors are at an extreme. In such cases, factor spacing/range can possibly be planned out in advance to ensure $[-\alpha, +\alpha]$, so each coded factor falls within feasible levels.

The inscribed CCD uses points only within the specified factor levels. This means that it does not have the "corner point constraint" issue that the circumscribed CCD has. However, the improvement compromises the accuracy of the response prediction near the lower and upper limits of each

factor. While the inscribed CCD provides a good response prediction over the central subset of factor space, the quality is not as high as the response prediction provided by the inscribed CCD.

Like the inscribed CCD, the face-centered CCD does not require points outside the original factor range. Compared to the inscribed CCD, the face-centered CCD provides a relatively high quality of response prediction over the entire explored factor space. The drawback of the face-centered CCD is that it gives a poor estimate of the pure quadratic coefficient, which is the coefficient of square term of a factor.

For relatively the same accuracy, Box-Behnken Design is more efficient than CCD in cases involving three or four factors because it requires fewer treatments of factor level combinations. However, like the inscribed CCD, the prediction at the extremes (corner points) is poor. The property of "missing corners" can be useful if these should be avoided due to physical or economic constraints, because potential of data loss in those cases can be prevented.

Response Surface Types

This section provides the theory for the following response surface types:

[Genetic Aggregation](#)

[Standard Response Surface - Full 2nd-Order Polynomials](#)

[Kriging](#)

[Non-Parametric Regression](#)

[Sparse Grid](#)

For information on the Neural Network response surface, see [Neural Network \(p. 116\)](#).

Genetic Aggregation

The Genetic Aggregation response surface can be written as an ensemble using a weighted average of different metamodels:

$$\hat{y}_{ens}(x) = \sum_{i=1}^{N_M} w_i \cdot \hat{y}_i(x)$$

where:

\hat{y}_{ens} = prediction of the ensemble

\hat{y}_i = prediction of the i -th response surface

N_M = number of metamodels used, $N_M \geq 1$

w_i = weight factor of the i -th response surface

The weight factors satisfy:

$$\sum_{i=1}^{N_M} w_i = 1 \text{ and } w_i \geq 0, \quad 1 \leq i \leq N_M$$

To estimate the best weight factor values, DesignXplorer minimizes the Root Mean Square Error (RMSE) of the Design of Experiments points (or design points) on \hat{y}_{ens} , and the RMSE of the same design points based on the cross-validation of \hat{y}_{ens} ($PRESS_{RMSE}$).

$$RMSE(\hat{y}_{ens}) = \sqrt{\frac{1}{N} \sum_{j=1}^N (y(x_j) - \hat{y}_{ens}(x_j))^2}$$

$$PRESS_{RMSE}(\hat{y}_{ens}) = \sqrt{\frac{1}{N} \sum_{j=1}^N (y(x_j) - \hat{y}_{ens-j}(x_j))^2}$$

With:

$$\hat{y}_{ens-j}(x) = \sum_{i=1}^{N_M} w_i \cdot \hat{y}_{i-j}(x)$$

where:

$x_j = j$ -th design point

$y(x_j) =$ output parameter value at x_j

$\hat{y}_{i-j} =$ prediction of the i -th response surface built without the j -th design point

$N =$ number of design points

Cross-Validation

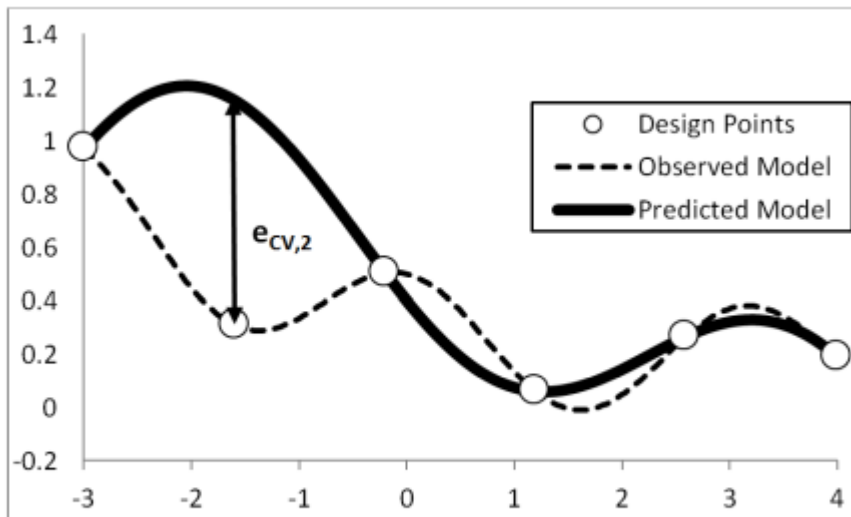
DesignXplorer uses the *Leave-One-Out* and *K-Fold* cross-validation methods.

Leave-One-Out Method:

- Solves as many sub-metamodels as there are design points.
- For a given i -th response surface, DesignXplorer computes N sub-metamodels, where each sub-response surface corresponds to the i -th response surface fitted to $N - 1$ design points.
- The cross-validation error of the j -th design point is the error at this point of the sub-response surface built without the j -th design point.

K-Fold Method:

- Builds k sub-metamodels of the i -th response surface, where each sub-response surface corresponds to the i -th response surface fitted to $N - N/k$ design points.
- The cross-validation error at the j -th design point is the error at this point of the sub-response surface built without the subset of N/k design points containing the j -th design point.
- To improve the relevance of the k-fold strategy, the N/k design points used as validation points of each fold are selected by using the maximum of minimum distance.



Cross-validation error at the second sample point: $e_{cv,2}$

The Leave-One-Out method can be expensive, so DesignXplorer uses 10-fold cross-validation by default and switches to the Leave-One-Out method when the number of design points is too small to obtain a relevant 10-fold cross-validation. To reduce the computation cost, the cross-validation is done in parallel.

If you note \tilde{e}_{i-j} , the cross-validation error of i -th response surface built without the j -th design point, then:

$$\tilde{e}_{ens-j}(x) = \sum_{i=1}^{N_M} w_i \cdot \tilde{e}_{i-j}(x)$$

DesignXplorer computes analytically the weight factor values. This computation is based on a similar approach proposed by Viana *et al.* (2009).

$$w = \frac{C^{-1} \mathbf{1}}{\mathbf{1}^T C^{-1} \mathbf{1}}$$

where $\mathbf{1}$ is the identity matrix and C the matrix of the mean square errors:

$$C_{ij} = \frac{1}{N} E_i E_j$$

To get positive weights, DesignXplorer uses only the diagonal elements of C .

With:

E_i and E_j are respectively the errors (combination of $RMSE$, $PRESS_{RMSE}$, and the overfitting penalization) of the i -th and j -th metamodels.

Genetic Algorithm

There are many types of metamodels, including Polynomial Regression, Kriging, Support Vector Regression, and Moving Least Squares. For each response surface, there are different settings. For example, on the Kriging response surface, you can control the type of kernel (Gaussian, exponential, and so on), the type of kernel variation (anisotropic or isotropic), and the type of polynomial regression (linear, quadratic, and so on).

To increase the chance of getting the most effective response surface, DesignXplorer generates a population of metamodels with different types and settings. This population corresponds to the first population of the genetic algorithm run by DesignXplorer. The next populations are obtained by cross-over and mutation of previous population.

Cross-Over Operator

There are two types of cross-over. The first one is the cross-over between two response surfaces of the same type (for example, two Kriging), and the second one is the cross-over between two response surfaces of different types (for example, a Kriging and a polynomial regression).

In the first case, DesignXplorer exchanges a part of settings from the first parent to the second parent (for example, the exchange of kernel type between two Kriging response surfaces).

In the second case, DesignXplorer creates a new response surface (an ensemble), which is a combination of the two parents (for example, the combination of a Kriging and a polynomial regression response surface).

Mutation Operator

DesignXplorer mutates one or several settings of the response surface (or of the response surfaces, in the case of a combination of response surfaces).

To keep a diversity of response surfaces, the genetic algorithm removes a part of the response surface type that is too present in the population, while it retains other response surfaces less present.

In the best case, the population contains similar metamodels in terms of prediction accuracy (*RMSE*) when the predicted values are different ($\hat{y}(x)$): this increases the chance of error cancellation on the ensemble.

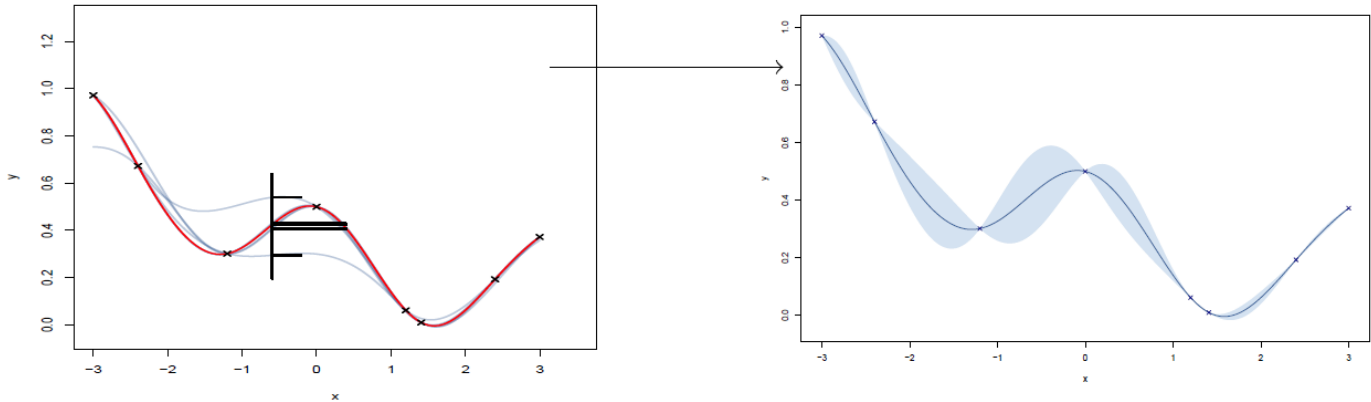
For external references, see [Genetic Aggregation Response Surface References \(p. 405\)](#).

Automatic Refinement

Gaussian Process regression methods, such as Kriging, provide a prediction distribution. The genetic aggregation can contain several surrogate models that do not afford a local prediction distribution.

Ben Salem *et al.* [1] introduced a universal prediction (UP) for all surrogate models. This distribution relies on cross-validation sub-model predictions. Based on this empirical distribution, they define an adaptive sampling technique for global refinement (UP-SMART).

The predicted error is the criterion \mathbf{Y} .



The Genetic Aggregation refinement is a hybrid variant of UP-SMART consisting in adding at step n a point:

$$x_{n+1} \in \underset{x \in \mathbb{X}}{\operatorname{argmax}} (\gamma_n(x))$$

With:

- $\gamma_n(x) = \hat{\sigma}_n^2(x) + \delta d_{X_n}(x)$
- $\hat{\sigma}_n^2(x)$ = variant of the local UP variance at x point at step n
- $d_{X_n}(x) = \inf\{d(x, x') : x' \in X_n\}$ and $d(\cdot, \cdot)$ denote a given distance on \mathbb{R}^p
- X_n = sample set used to build the response surface
- \mathbb{X} = parameter space defined by the user
- δ = distance penalization

The predicted error exposed in DesignXplorer corresponds to $3\hat{\sigma}_n(x)$.

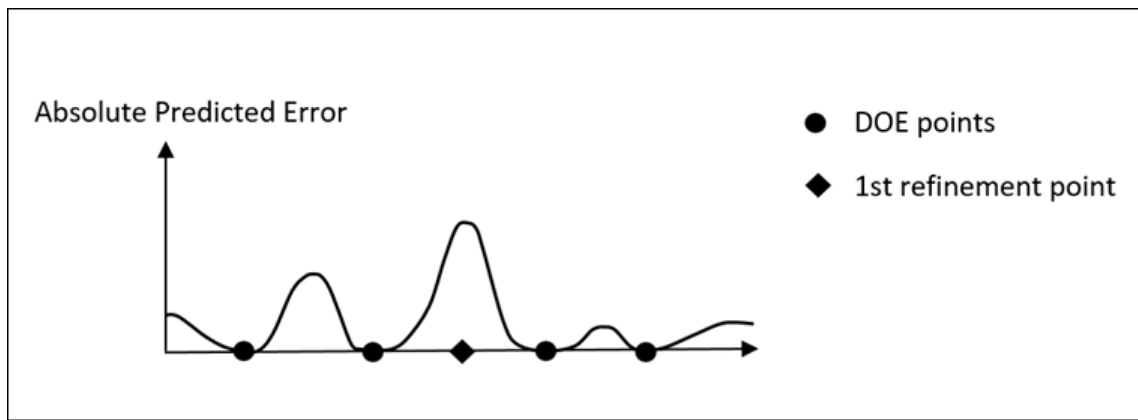
Bibliography

[1] M. Ben Salem, O. Roustant, F. Gamboa, & L. Tomaso (2017). "Universal prediction distribution for surrogate models." *SIAM/ASA Journal on Uncertainty Quantification*, 5(1), 1086-1109.

Genetic Aggregation with Multiple Refinement Points

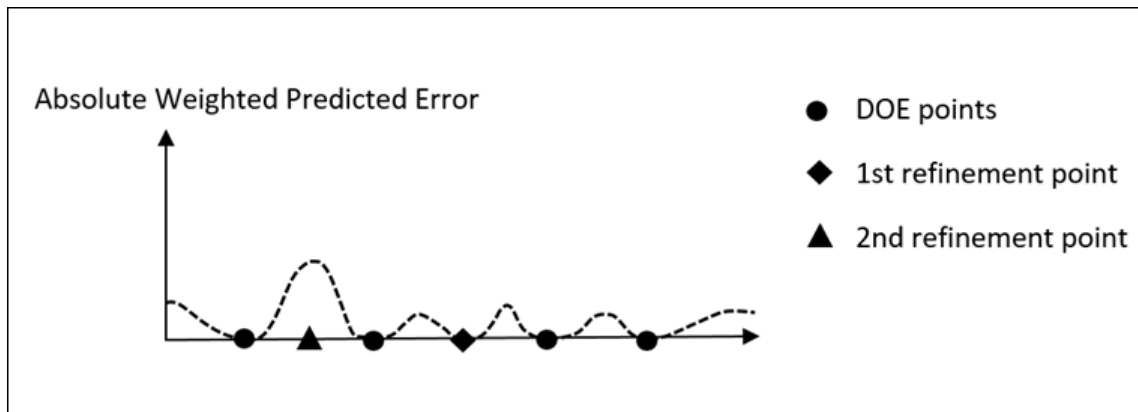
The first refinement point selected is the worst point, having the maximum absolute predicted error (APE) in regard to the expected tolerance. This point corresponds to the refinement point generated by DesignXplorer if **Number of Refinement Points per Iteration** is equal to 1.

The following figure represents the absolute predicted error of a response surface with only one input parameter:



For N refinement points to submit simultaneously, the generation of the i -th refinement point depends on the $(i-1)$ first pending refinement points, with $i > 2$. The response surface is updated only when all N refinement points have been generated.

While the previous figure shows that the first refinement point is based on the APE, in the next figure, refinement points are based on the absolute weighted predicted error (AWPE), which takes into account the influence of the pending refinement points on the response surface. AWPE is a transformation of APE that reduces the predicted error around the pending refinement point to favor the domain with a high predicted error and a low density of pending refinement points. The i -th refinement point, with $i > 1$, is selected as the worst point with the maximum AWPE.



Algorithm Summary:

1. $\{X;Y\}$ = Design of Experiments (DOE)
2. $RS(X,Y)$ = Response Surface
3. $j = 0$
4. While $\text{Max}(APE) < \text{Tolerance}$ AND $j < N_{\text{max}}$
 - a. $X^{(1)}$ = New Refinement Point
 - b. $AWPE = APE * W(X^{(1)})$
 - c. $i = 2$

- d. While $i < N$ AND $\text{Max}(\text{AWPE}) < \text{Tolerance}$
 - i. $X^{(i)} = \text{New Refinement Point} = \text{Max}(\text{AWPE})$
 - ii. $\text{AWPE} = \text{AWPE} * W(X^{(i)})$
 - iii. $i = i + 1$
- e. $\{Y^{(1)}, \dots, Y^{(N)}\} = \text{Update of Refinement Points } \{X^{(1)}, \dots, X^{(N)}\}$
- f. $\{X; Y\} = \{X; Y\} \cup \{X^{(1)}, \dots, X^{(N)}; Y^{(1)}, \dots, Y^{(N)}\}$
- g. $\text{RS}(X, Y) = \text{Update of the Response Surface}$
- h. $j = j + N$

where:

- X and Y correspond respectively on input and output parameter values
- APE is the Absolute Predicted Error of the Response Surface (RS)
- Tolerance is the expected accuracy of RS given by the user
- N_{max} is the Maximum Number of Refinement Points
- $W(X')$ is equal to 1 in the full input parameter space except close to point X' , where W decreases to be equal to 0 at X'
- AWPE is the Absolute Weighted Predicted Error of RS

Standard Response Surface - Full 2nd-Order Polynomials

The meta-modeling algorithms are categorized as:

[General Definitions \(p. 339\)](#)

[Linear Regression Analysis \(p. 340\)](#)

General Definitions

The error sum of squares SSE is:

$$SSE = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = (\{y\} - \{\hat{y}\})^T (\{y\} - \{\hat{y}\}) \quad (1)$$

where:

y_i = value of the output parameter at the i -th sampling point

\hat{y}_i = value of the regression model at the i -th sampling point

The regression sum of squares SSR is:

$$SSR = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2$$

where:

$$\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$$

The total sum of squares SST is:

$$SST = \sum_{i=1}^n (y_i - \bar{y})^2$$

For linear regression analysis the relationship between these sums of squares is:

$$SST = SSR + SSE$$

Linear Regression Analysis

For a linear regression analysis, the regression model at any sampled location $\{x_i\}$, with n in the m -dimensional space of the input variables can be written as:

$$y_i = [t]_i \{c\} + \varepsilon$$

where:

$[t]_i$ = row vector of regression terms of the response surface model at the i -th sampled location

$\{c\} = [c_1 \ c_2 \ \dots \ c_p]^T$ = vector of the regression parameters of the regression model

p = total number of regression parameters. For linear regression analysis, the number of regression parameters is identical to the number of regression terms.

The regression coefficients $\{c\}$ can be calculated from:

$$\{c\} = ([d]^T [d])^{-1} [d]^T \{y\}$$

Kriging

Kriging postulates a combination of a polynomial model plus departures of the form given by:

$$y(x) = f(x) + Z(x)$$

where $y(x)$ is the unknown function of interest, $f(x)$ is a polynomial function of x , and $Z(x)$ is the realization of a normally distributed Gaussian random process with mean zero, variance σ^2 , and non-zero covariance. The $f(x)$ term in is similar to the polynomial model in a response surface and provides a "global" model of the design space.

While $f(x)$ "globally" approximates the design space, $Z(x)$ creates "localized" deviations so that the Kriging model interpolates the N sample data points. The covariance matrix of $Z(x)$ is given by:

$$\text{Cov}[Z(x^i), Z(x^j)] = \sigma^2 R([r(x^i, x^j)])$$

where R is the correlation matrix and $r(x^i, x^j)$ is the spatial correlation of the function between any two of the N sample points x^i and x^j . R is an $N \times N$ symmetric, positive definite matrix with ones along the diagonal. The correlation function $r(x^i, x^j)$ is a Gaussian correlation function:

$$r(x^i, x^j) = \exp\left(-\sum_{k=1}^M \theta_k |x_k^i - x_k^j|^2\right)$$

The θ_k in are the unknown parameters used to fit the model, M is the number of design variables, and x_k^i and x_k^j are the k^{th} components of sample points x^i and x^j . In some cases, using a single correlation parameter gives sufficiently good results. You can specify the use of a single correlation parameter or one correlation parameter for each design variable in the **Options** dialog box. Select **Tools** → **Options** → **Design Exploration** → **Response Surface** and then, under **Kriging Options**, make the desired section for **Kernel Variation Type**.

$Z(x)$ can be written:

$$Z(x) = \sum_{i=1}^N \lambda_i r(x^i, x)$$

Predicted Error

Kriging or Gaussian process regression (GPR) is widely popular, especially in spatial statistics. It is based on the early works of Krige [1]. The mathematical framework can be found in [2,3]. Kriging models predict the outputs of a function f : based on a set of n observations. Within the GP framework, the posterior distribution is given by the conditional distribution of \mathbf{Y} given the observations:

$$\mathbf{y}_n = (y_1, \dots, y_n)^T \text{ where } y_i = f(\mathbf{x}^{(i)}) \text{ for } 1 \leq i \leq n.$$

In the Gaussian process framework, Kriging or Gaussian process regression provides a mean function (prediction) and a predicted variance that DesignXplorer uses to assess the local prediction.

The general form assumes that the true model response is a realization of a Gaussian process described by the following equation:

$$m_Z(\mathbf{x}) = \underbrace{\mu(\mathbf{x})}_{\text{trend function}} + \underbrace{\sigma^2 Y(\mathbf{x})}_{\text{Zero mean Gaussian Process Regression}}$$

Focus here is on the Universal Kriging:

- **Universal Kriging:** when $\mu(\mathbf{x}) = \beta^\top \mathbf{h}(\mathbf{x})$ where (h_i) are known trend functions and β is unknown.

In this case, the Kriging predictions (m) and variance ($\hat{\sigma}^2$) are given by:

$$\hat{m}_{\mathbf{Z}_n}(\mathbf{x}) = \mathbf{h}(\mathbf{x})^\top \hat{\beta} + k(\mathbf{x}, \mathbf{X}_n)^\top K^{-1} (Y - H^\top \hat{\beta})$$

$$\begin{aligned} \hat{\sigma}_{\mathbf{Z}_n}^2(\mathbf{x}) &= k(\mathbf{x}, \mathbf{x}) - k(\mathbf{x}, \mathbf{X}_n)^\top K^{-1} k(\mathbf{x}, \mathbf{X}_n) \\ &\quad + \left(\mathbf{h}(\mathbf{x})^\top + k(\mathbf{x}, \mathbf{X}_n)^\top K^{-1} H \right)^\top \left(H^\top K^{-1} H \right)^{-1} \left(\mathbf{h}(\mathbf{x})^\top + k(\mathbf{x}, \mathbf{X}_n)^\top K^{-1} H \right) \end{aligned}$$

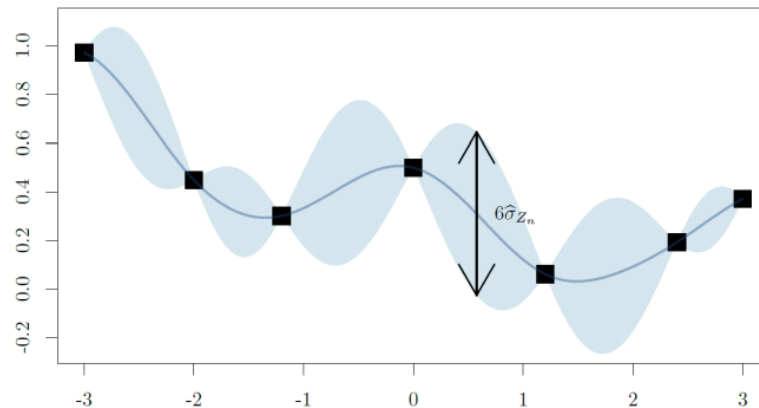
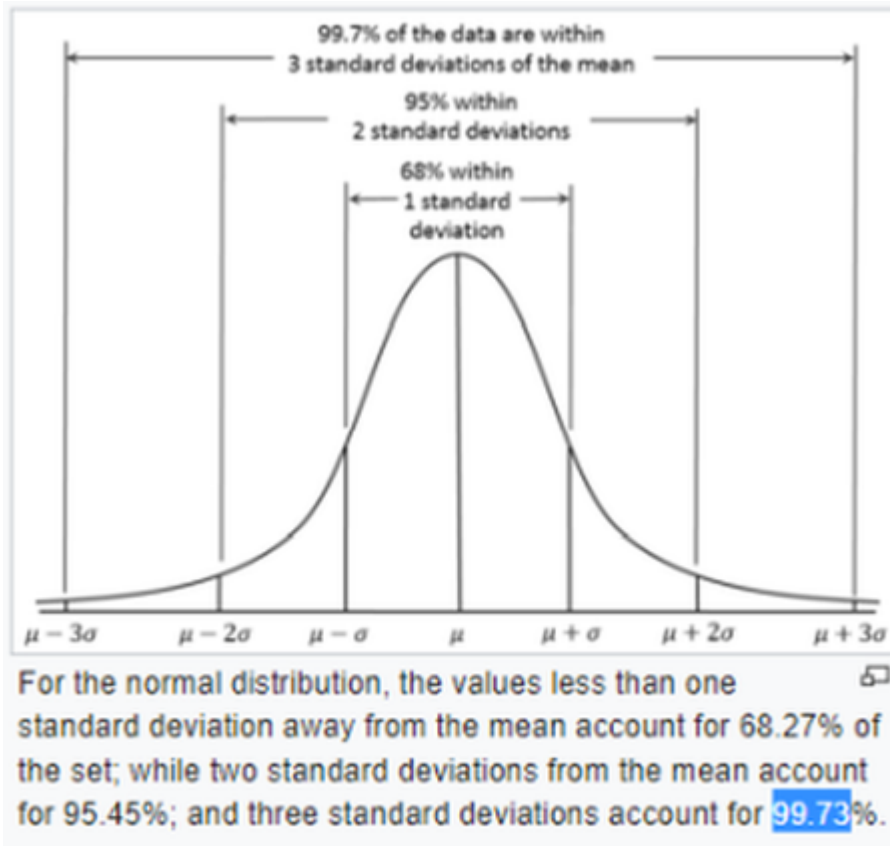
where:

$$\hat{\beta} = (H^\top K^{-1} H)^{-1} H^\top K^{-1} Y.$$

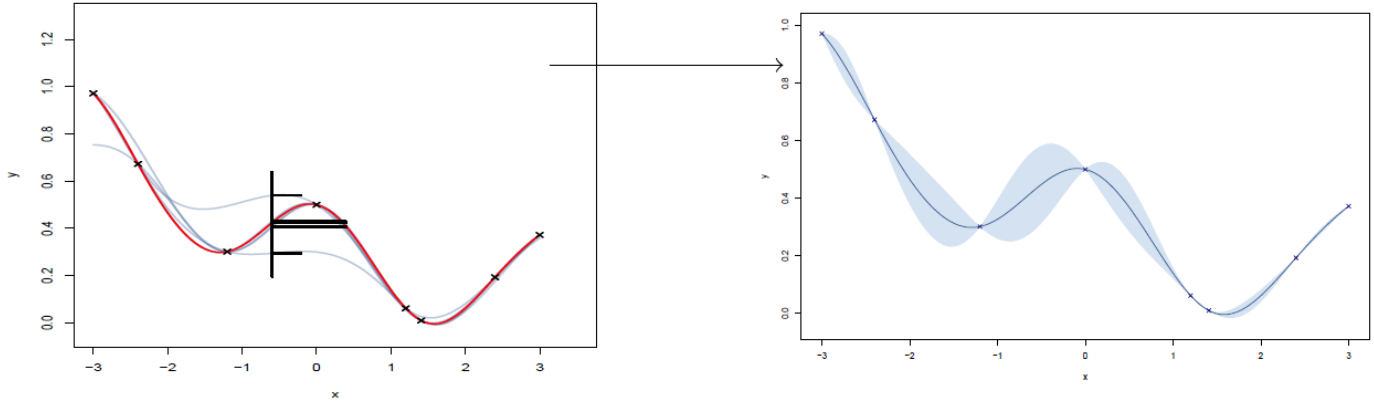
Here, $k(\mathbf{x}, \mathbf{X}_n)$ is the vector $(k(\mathbf{x}, \mathbf{x}^{(1)}), \dots, k(\mathbf{x}, \mathbf{x}^{(n)}))^\top$ and $K_\theta = k(X, X)$ is the invertible matrix with entries $\left(k(\mathbf{X}_n, \mathbf{X}_n) \right)_{ii} = k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$, for $1 \leq i, j \leq n$.

$k_\theta(\mathbf{x}, \mathbf{x}') = \text{Cov}[Y(\mathbf{x}), Y(\mathbf{x}')]$ the covariance function

The standard deviation $\hat{\sigma}$ is used as a tool to assess the errors. In fact, for a Gaussian distribution, $3\hat{\sigma}$ cover 99.7% of the possible predictions. The predicted error displayed in DesignXplorer is $3\hat{\sigma}$.



For a Genetic Aggregation response surface, predicted error is calculated using the universal prediction (UP) distribution [4] to quantify the predicted error. The UP distribution can be applied for any surrogate model. It is based on a weighted empirical probability measure supported by the CV submodel prediction.



As indicated in [Automatic Refinement \(p. 336\)](#) in the genetic aggregation theory section, the predicted error is the criterion Y .

Bibliography

- [1] D. G. Krige. "A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand." *Journal of the Chemical, Metallurgical and Mining Society of South Africa*, 52:119–139, 1951
- [2] G. Matheron. "Principles of geostatistics." *Economic Geology*, 58(8):1246–1266, 1963.
- [3] M. L. Stein. "Interpolation of Spatial Data: Some Theory for Kriging." Springer Science & Business Media, New York, 2012.
- [4] M. Ben Salem, O. Roustant, F. Gamboa, & L. Tomaso (2017). "Universal prediction distribution for surrogate models." *SIAM/ASA Journal on Uncertainty Quantification*, 5(1), 1086-1109.

Non-Parametric Regression

Let the input sample (as generated from a DOE method) be $X = \{x_1, x_2, x_3, \dots, x_M\}$, where each x_i is an N -dimensional vector and represents an input variable. The objective is to determine the equation of the form:

$$Y = \langle W, X \rangle + b \tag{2}$$

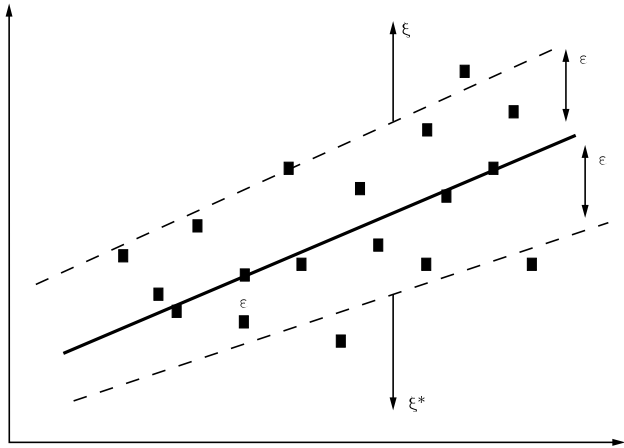
where W is a weighting vector.

In case of generic non-parametric case, [Equation 2 \(p. 344\)](#) is rewritten as:

$$Y = \sum_{i=1}^N (A_i - A_i^*) K(\vec{X}_i, \vec{X}) + b \tag{3}$$

where: $K(\vec{X}_i, \vec{X})$ is the kernel map and the quantities A_i and A_i^* are Lagrange multipliers whose derivation are shown in later sections.

To determine the Lagrange multipliers, you start with the assumption that the weight vector W must be minimized such that all (or most) of the sample points lie within an error zone around the fitted surface. The following figure provides a simple demonstration. It fits a regression line for a group of sample points with a tolerance of ϵ , which is characterized by slack variables ξ^* and ξ .



Thus, a primal optimization formulation is written as follows:

$$\begin{aligned} \text{Minimize } L &= 0.5\|W\|^2 + \frac{C}{N} \sum_{i=1}^N (\xi_i + \xi_i^*) \\ y_i - (\langle W, X_i \rangle + b) &\leq \varepsilon + \xi_i \\ \text{S.T.} \\ (\langle W, X_i \rangle + b) - y_i &\leq \varepsilon + \xi_i^* \end{aligned} \quad (4)$$

where C is an arbitrary (>0) constant. To characterize the tolerance properly, you define a loss function in the interval $(-\varepsilon, \varepsilon)$, which de facto becomes the residual of the solution. In the present implementation, you use the ε -insensitive loss function, which is given by the equation:

$$\begin{aligned} L_\varepsilon &= 0 \forall |f(x) - y| < \varepsilon \\ L_\varepsilon &= |f(x) - y| - \varepsilon \end{aligned} \quad (5)$$

The primal problem in [Equation 4 \(p. 345\)](#) can be rewritten as the following using generalized loss functions:

$$\begin{aligned} L &= 0.5\|W\|^2 + C \sum_{i=1}^N (\bar{L}(\xi_i) + \bar{L}(\xi_i^*)) \\ \text{S.T.} \\ \xi_i, \xi_i^* &\geq 0 \\ \langle W, X_i \rangle + b - y_i &\leq \varepsilon + \xi_i \\ y_i - \langle W, X_i \rangle - b &\leq \varepsilon + \xi_i^* \end{aligned} \quad (6)$$

To solve this efficiently, a Lagrangian dual formulation is done on this to yield the following expression:

$$\begin{aligned}
L = & 0.5^* \|W\|^2 + C \sum_{i=1}^N (l(\xi_i) + l(\xi_i^*)) + \sum_{i=1}^N A_i [\langle W, X_i \rangle + b - y_i (\varepsilon + \xi_i)] \\
& + \sum_{i=1}^N A_i^* [y_i - \langle W, X_i \rangle - b - (\varepsilon + \xi_i^*)] \\
& + \sum_{i=1}^N (\eta_i \xi_i + \eta_i^* \xi_i^*)
\end{aligned} \tag{7}$$

After some simplification, the dual Lagrangian can be written as the following [Equation 8 \(p. 346\)](#):

$$\begin{aligned}
\min L = & 0.5 \sum_{i=1}^N \sum_{j=1}^N (A_i^* - A_i)(A_j^* - A_j) \langle X_i, X_j \rangle \\
& + \sum_{i=1}^N [\varepsilon (A_i^* - A_i) - y_i (A_i^* - A_i)] \\
& + C \sum_{i=1}^N \left[\xi_i \frac{\partial l(\xi_i)}{\partial \xi_i} - l(\xi_i) \right] + C \sum_{i=1}^N \left[\xi_i^* \frac{\partial l(\xi_i^*)}{\partial \xi_i^*} - l(\xi_i^*) \right] \\
\text{S.T.} \\
& \sum_{i=1}^N (A_i^{(*)} - A_i) = 0 \\
& 0 \leq A_i^{(*)} \leq C \frac{\partial l(\xi_i^{(*)})}{\partial \xi_i^{(*)}} \\
& \xi_i^{(*)} = \inf \left\{ \xi \mid C \frac{\partial l(\xi_i^{(*)})}{\partial \xi_i^{(*)}} \geq A_i^{(*)} \right\}
\end{aligned} \tag{8}$$

is a quadratic constrained optimization problem, and the design variables are the vector A . Once this is computed, the constant b in can be obtained by the application of Karush-Kuhn-Tucker (KKT) conditions. Once the ε -insensitive loss functions are used instead of the generic loss functions $l(\xi)$ in, this can be rewritten in a much simpler form as the following:

$$\begin{aligned}
\text{MIN}_{A,A^*} L = & 0.5 \sum_{i=1}^N \sum_{j=1}^N (A_i^* - A_i)(A_j^* - A_j) K(X_i, X_j) \\
& + \sum_{i=1}^N [\varepsilon (A_i^* + A_i) - y_i (A_i^* - A_i)] \\
\text{S.T.} \\
& 0 \leq A_i^{(*)} \leq C \\
& \sum_{i=1}^N (A_i^* - A_i) = 0
\end{aligned} \tag{9}$$

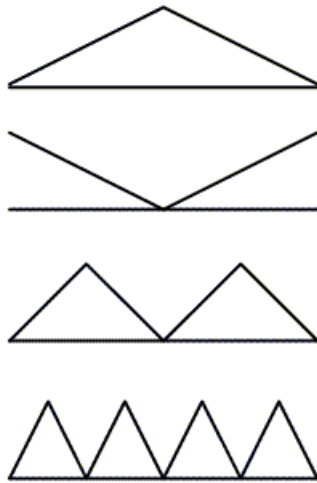
which is to be solved by a QP optimizer to yield the Lagrange multipliers $A_i^{(*)}$ and the constant b is obtained by applying the KKT conditions.

Sparse Grid

The Sparse Grid metamodeling is a hierarchical Sparse Grid interpolation algorithm based on piecewise multilinear basis functions.

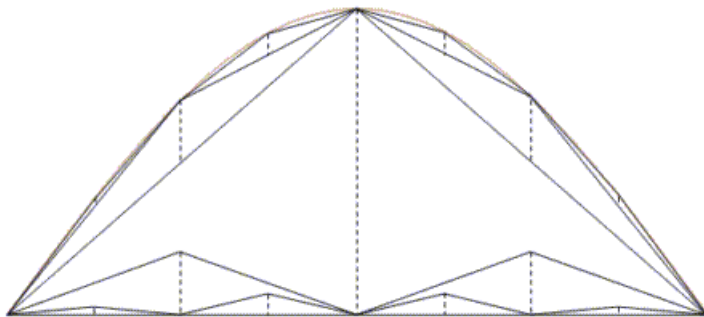
The first ingredient of a Sparse Grid method is a one-dimensional multilevel basis.

Piecewise linear hierarchical basis (from the level 0 to the level 3):



The calculation of coefficients values associated to a piecewise linear basis is hierarchical and obtained by the differences between the values of the objective function and the evaluation of the current Sparse Grid interpolation.

Example 5: Interpolation with the Hierarchical Basis



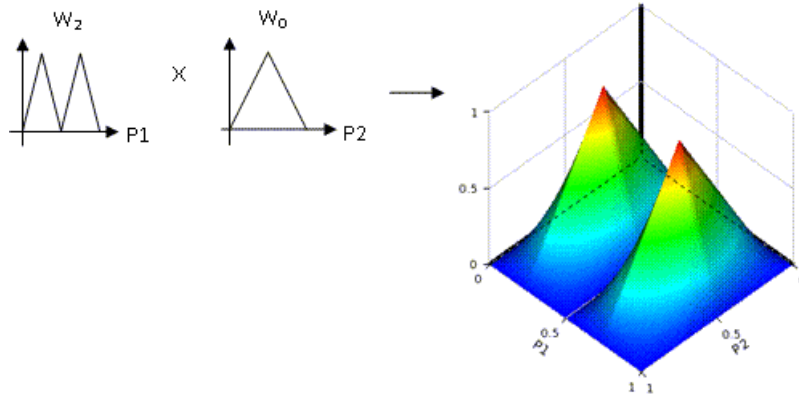
For a multi-dimensional problem, the Sparse Grid metamodeling is based on piecewise multilinear basis functions that are obtained by a sparse tensor product construction of one-dimensional multilevel basis.

Let $\underline{l} = (l_1, l_2, \dots, l_d) \in \mathbb{N}^d$ denote a multi-index, where d is the problem dimension (number of input parameters) and l_i corresponds to the level in the i -th direction.

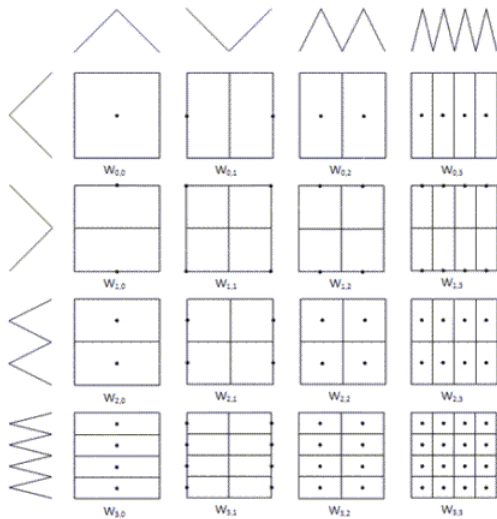
The generation of the Sparse Grid W_1 is obtained by the following tensor product:

$$W_l = W_{l_1} \oplus W_{l_2} \oplus \dots \oplus W_{l_d} = \bigoplus_{i=1}^d W_{l_i}$$

Example 6: Tensor Product Approach to Generate the Piecewise Bilinear Basis Functions $W_{2,0}$



Tensor product of linear basis functions for a two-dimensional problem:



To generate a new Sparse Grid W_l , any Sparse Grid $W_{l'}$ that meets the order relation $W_{l'} < W_l$ must be generated before:

$$W_{l'} < W_l$$

means:

$$|l|_1 \leq |l'|_1$$

with

$$|I|_1 = \sum_{i=1}^d l_i$$

and

$$\exists i < d \mid l_i < l'_i$$

For example, in a two-dimensional problem, the generation of the grid $W_{2,1}$ requires several steps:

1. Generation of $W_{0,0}$
2. Generation of $W_{0,1}$ and $W_{1,0}$
3. Generation of n and $W_{1,1}$

The calculation of coefficients values associated to a piecewise multilinear basis is similar to the calculation of coefficients of linear basis: the coefficients are obtained by the differences between the values of the objective function on the new grid and the evaluation (of the same grid) with the current Sparse Grid interpolation (based on old grids).

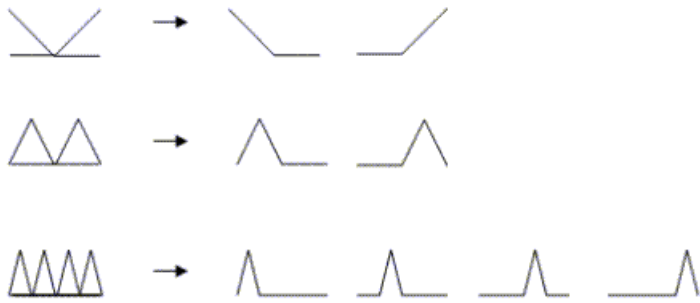
You can observe for a higher-dimensional problem that not all input variables carry equal weight. A regular Sparse Grid refinement can lead to too many support nodes. This is why the Sparse Grid metamodeling uses a dimension-adaptive algorithm to automatically detect separability and which dimensions are the more or the less important ones to reduce computational effort for the objectives functions.

The hierarchical structure is used to obtain an estimate of the current approximation error. This current approximation error is used to choose the relevant direction to refine the Sparse Grids. If the approximation error has been found with Sparse Grid W_l , the next iteration consists in the generation of new Sparse Grids obtained by incrementing of each dimension level of W_l (one by one) as far as possible: the refinement of $W_{2,1}$ can generate two new Sparse Grids $W_{3,1}$ and $W_{2,2}$ (if the $W_{3,0}$ and $W_{1,2}$ already exist).

W_3				
W_2	$W_{0,2}$			
W_1	$W_{0,1}$	$W_{1,1}$	$W_{2,1}$	
W_0	$W_{0,0}$	$W_{1,0}$	$W_{2,0}$	$W_{3,0}$
x	W_0	W_1	W_1	W_3

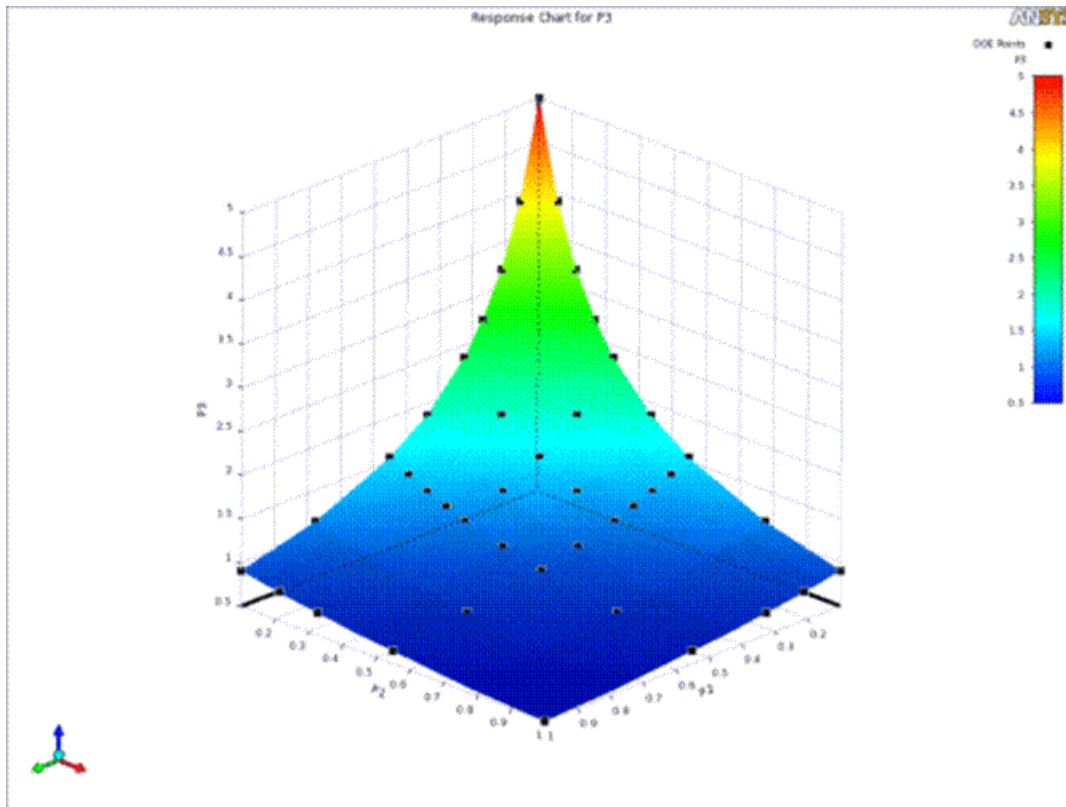
The Sparse Grid metamodeling stops automatically when the desired accuracy is reached or when the maximum depth is met in all directions (the maximum depth corresponds to the maximum number of hierarchical interpolation levels to compute: if the maximum depth is reached in one direction, the direction is not refined further).

The new generation of the Sparse Grid allows as many linear basis functions as there are points of discretization.



All Sparse Grids generated by the tensor product contain only one-point that allows refinement more locally.

Sparse Grid metamodeling is more efficient with a more local refinement process that uses less design points, as well as, reaching the requested accuracy faster.



Goal-Driven Optimization Theory

Goal-driven optimization (GDO) is a set of constrained, multi-objective optimization techniques in which the best possible designs are obtained from a sample set given the objectives you set for parameters. The available optimization methods are:

- Screening
- MOGA
- NLPQL

- MISQP
- Adaptive Single-Objective
- Adaptive Multiple-Objective

The Screening, MISQP, and MOGA methods can be used with discrete parameters. The Screening, MISQP, MOGA, Adaptive Multiple-Objective, and Adaptive Single-Objective methods can be used with continuous parameters with manufacturable values.

The GDO process allows you to determine the effect on input parameters with certain objectives applied to the output parameters. For example, in a structural engineering design problem, you can determine the combination of design parameters that best satisfy minimum mass, maximum natural frequency, maximum buckling and shear strengths, and minimum cost, with maximum value constraints on the von Mises stress and maximum displacement.

This section describes GDO and its use in performing single- and multiple-objective optimization.

[GDO Principles](#)

[GDO Guidelines and Best Practices](#)

[Goal-Driven Optimization Theory](#)

GDO Principles

You can apply goal-driven optimization to design optimization by using any of the following methods:

- **Screening.** This is a non-iterative direct sampling method that uses a quasi-random number generator based on the Hammersley algorithm.
- **MOGA.** This is an iterative Multi-Objective Genetic Algorithm that can optimize problems with continuous input parameters.
- **NLPQL.** This is a gradient-based, single-objective optimizer based on quasi-Newton methods.
- **MISQP.** This is a gradient-based, single-objective optimizer that solves mixed-integer non-linear programming problems by a modified sequential quadratic programming (SQP) method.
- **Adaptive Single-Objective.** This is a gradient-based, single-objective optimizer that employs an OSF (Optimal Space-Filling) DOE, a Kriging response surface, and MISQP.
- **Adaptive Multiple-Objective.** This is an iterative, multi-objective optimizer that employs a Kriging response surface and MOGA. In this method, the use of a Kriging response surface allows for a more rapid optimization process because:
 - All design points are not evaluated except when necessary.
 - Part of the population is simulated by evaluations of the Kriging response surface, which is constructed of all design points submitted by MOGA.

MOGA is better for calculating the global optima, while NLPQL and MISQP are gradient-based algorithms ideally suited for local optimization. So you can start with Screening or MOGA to locate the multiple tentative optima and then refine with NLPQL or MISQP to zoom in on the individual local maximum or minimum value.

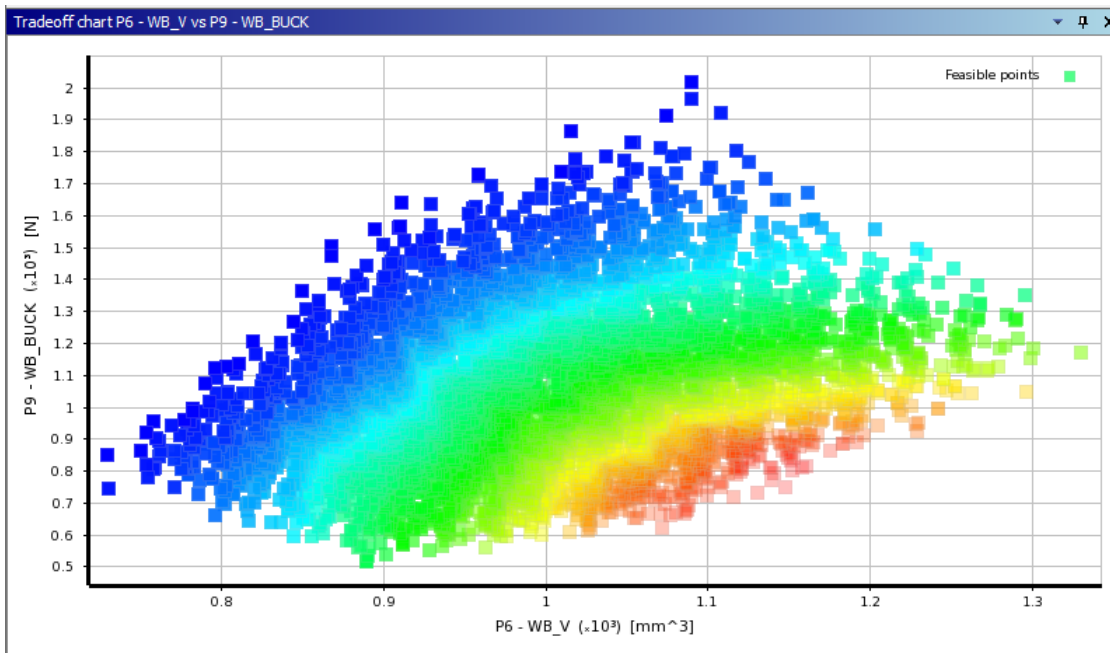
The GDO framework uses a Decision Support Process (DSP) based on satisfying criteria as applied to the parameter attributes using a weighted aggregate method. In effect, the DSP can be viewed as a postprocessing action on the Pareto fronts as generated from the results of the various optimization methods.

Usually Screening is used for preliminary design, which can lead you to apply one of the other approaches for more refined optimization results. Note that running a new optimization causes a new sample set to be generated.

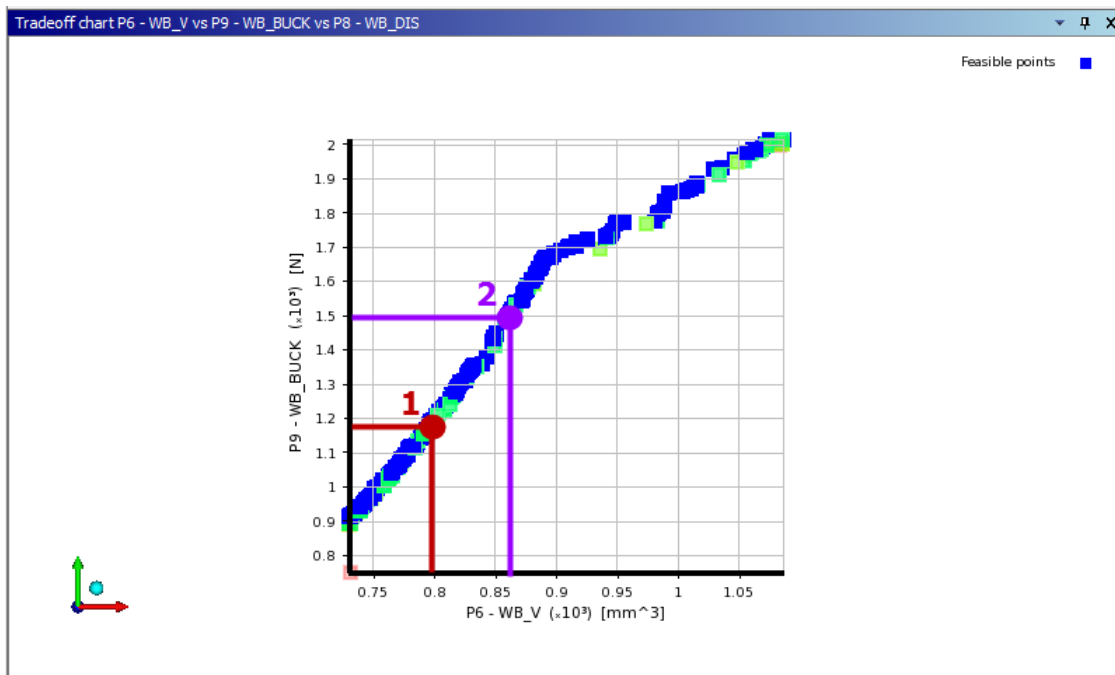
In either approach, the Tradeoff chart, as applied to the resulting sample set, shows the Pareto-dominant solutions. However, in MOGA, the Pareto fronts are better articulated and most of the feasible solutions lie on the first front, as opposed to the usual results of Screening, where the solutions are distributed across all the Pareto fronts. This is illustrated in the following two figures.

This first figure shows the 6,000 sample points generated by the Screening method.

This second figure shows a final sample set for the same problem after 5,400 evaluations by MOGA.



The following figure demonstrates the necessity of generating Pareto fronts. The optimal Pareto front shows two non-dominated solutions. The first Pareto front or Pareto frontier is the *list of non-dominated points* for the optimization.



A *dominated point* is a point that, when considered in regard to another point, is not the best solution for any of the optimization objectives. For example, if point **A** and point **B** are both defined, point **B** is a dominated point when point **A** is the better solution for all objectives.

In this example, the two axes represent two output parameters with conflicting objectives: the X axis represents **Minimize P6 (WB_V)**, and the Y axis represents **Maximize P9 (WB_BUCK)**. The chart shows two optimal solutions, point **1** and point **2**. These solutions are *non-dominated*, which means that both points are equally good in terms of Pareto optimality, but for different objectives. Point **1** is the better solution for **Minimize P6 (WB_V)**, and point **2** is the better solution for **Maximize P9 (WB_BUCK)**. Neither point is strictly dominated by any other point, so both are included on the first Pareto front.

GDO Guidelines and Best Practices

Guidelines and best practices follow for using DesignXplorer's goal-driven optimization (GDO) functionality:

[Convergence Rate % and Initial Finite Difference Delta % in NLPQL and MISQP](#)

[Objectives vs. Constraint Satisfaction in Goal-Driven Optimization](#)

Convergence Rate % and Initial Finite Difference Delta % in NLPQL and MISQP

Typically, the use of NLPQL or MISQP is suggested for continuous problems when there is only one objective function. The problem might or might not be constrained and must be analytic. This means that the problem must be defined only by continuous input parameters and that the objective functions and constraints should not exhibit sudden "jumps" in their domain.

The main difference between these algorithms and MOGA is that MOGA is designed to work with multiple objectives and does not require full continuity of the output parameters. However, for continuous single objective problems, the use of NLPQL or MISQP gives greater accuracy of the solution as gradient information and line search methods are used in the optimization iterations.

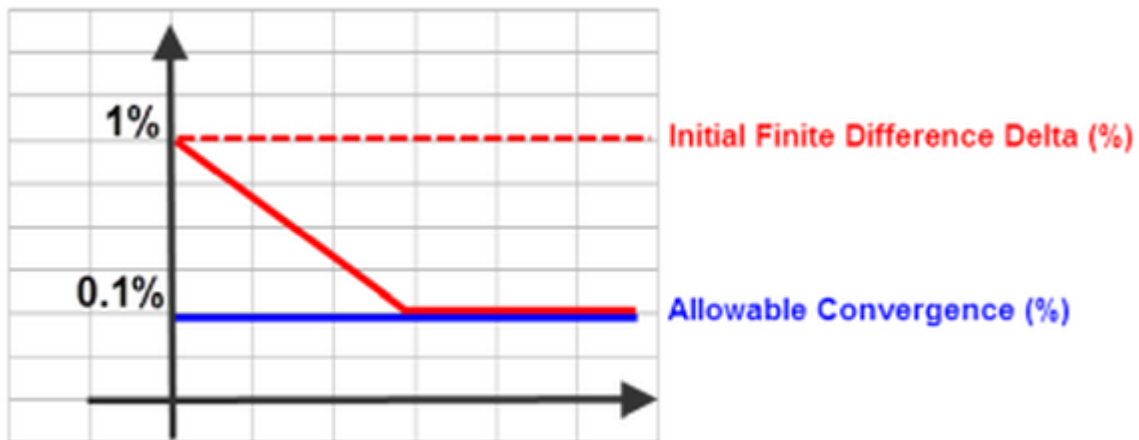
MOGA is a global optimizer designed to avoid local optima traps, while NLPQL and MISQP are local optimizers designed for accuracy.

For NLPQL and MISQP, the default convergence rate, which is specified by the **Allowable Convergence (%)** property, is set to **0.1%** for a **Direct Optimization** system and **0.0001 %** for a **Response Surface Optimization** system. The maximum value for this property is **100%**. This is computed based on the (normalized) Karush-Kuhn-Tucker (KKT) condition. This implies that the fastest convergence rate of the gradients or the functions (objective function and constraint) determine the termination of the algorithm.

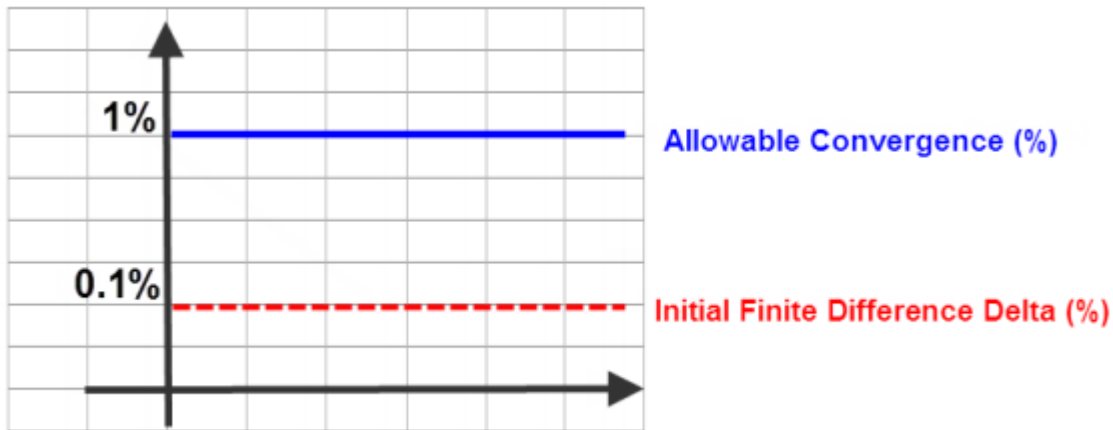
The default convergence rate is used in conjunction with the initial finite difference delta percentage value, which is specified by the **Initial Finite Difference Delta (%)** advanced property. This property defaults to **1%** for a **Direct Optimization** system and **0.001 %** for a **Response Surface Optimization** system. You use this property to specify a percentage of the variation between design points to ensure that the Delta use in the calculation of finite differences is large enough to be seen over simulation noise. The specified percentage is defined as a relative gradient perturbation between design points.

The advantage of this approach is that for large problems, it is possible to get a near-optimal feasible solution quickly without being trapped into a series of iterations involving small solution steps near the optima. To work most effectively with NLPQL and MISQP, keep the following guidelines in mind:

- **If the Initial Finite Difference Delta (%) is greater than the Allowable Convergence (%)**, the relative gradient perturbation gets iteratively smaller, until it matches the allowable convergence rate. At this point, the relative gradient value stays the same through the rest of the analysis.



- **If the Initial Finite Difference Delta (%) is less than or equal to the Allowable Convergence (%)**, the current relative gradient step remains constant through the rest of the analysis.



- **Both the Initial Finite Difference Delta (%) and Allowable Convergence (%) should be higher than the magnitude of the noise in your simulation.**

When setting the values for these properties, you have the usual trade-offs between speed and accuracy. Smaller values result in more convergence iterations and a more accurate (but slower) solution, while larger values result in fewer convergence iterations and a less accurate (but faster) solution. At the same time, however, you must be aware of the amount of noise in your model. For the input variable variations to be visible in the output variables, both values must be greater than the magnitude of the simulation's noise.

In general, DesignXplorer's default values for **Initial Finite Difference Delta (%)** and **Allowable Convergence (%)** cover the majority of optimization problems. For example, if you know that the noise magnitude in your direct optimization problem is 0.0001, then the default values (**Allowable Convergence (%) = 0.001** and **Initial Finite Difference Delta (%) = 0.01**) are good.

When the defaults are not a good match for your problem, of course, you can adjust the values to better suit your model and your simulation needs. If you require a more numerically accurate solution, you can set the convergence rate to as low as **1.0E-10%** and then set the **Initial Finite Difference Delta (%)** accordingly.

Objectives vs. Constraint Satisfaction in Goal-Driven Optimization

Screening is the only optimization method that does not require an objective for at least one output parameter. To use any other optimization method, you must define an objective for at least one output parameter.

Note:

For NLPQL, MISQP, or Adaptive Single-Objective, an objective can be defined for only one output parameter.

An optimization problem is undefined when none of its parameters have an objective defined. An optimization problem can also be unable to satisfy a constraint.

Undefined Optimization Problem

The following figure shows an optimization where none of the parameters have objectives defined. This optimization cannot be run.

Table of Schematic B4: Optimization							
	A	B	C	D	E	F	G
1	Input Parameters						
2	Name	Parameter	Objective		Constraint		
3			Type	Target	Type	Lower Bound	Upper Bound
4	P1	P1 - B	No Objective		No Constraint		
5	P2	P2 - D	No Objective		No Constraint		
6	P4	P4 - P	No Objective		No Constraint		
7	P7	P7 - SIG	No Objective		No Constraint		
8	P8	P8 - DIS	No Objective		No Constraint		
9	P9	P9 - BUCK	No Objective		No Constraint		
*		Select a Parameter					

In the **Project Schematic**, both the state and the Quick Help for the **Optimization** cell would indicate that input is required. Likewise, in the **Outline** view for the cell, both the state and Quick Help for the root node would indicate that input is required. If you tried to update the optimization with no objective set, an error message would display in the **Message** view.

Constraint Satisfaction Problem

If the constraints defined for the optimization cannot be satisfied, the single Pareto fronts displayed from the Tradeoff chart represent the feasible points that meet the constraints. The infeasible points that do not satisfy the constraints can also be displayed on the chart. This is a good way to demonstrate the feasibility boundaries of the problem in the output space, letting you determine the boundaries of the design envelope that reflects your preferences. Infeasible points can also be displayed on any chart where the optimization contains at least one constraint.

As demonstrated by the following figure, some combinations of **Constraint Type**, **Lower Bound**, and possibly **Upper Bound** settings specify constraints that cannot be satisfied. The results obtained from these settings indicate only the boundaries of the feasible region in the design space. Currently, only the Screening method can solve a pure constraint satisfaction problem.

Table of Schematic B4: Optimization							
	A	B	C	D	E	F	G
1	Input Parameters						
2	Name	Parameter	Objective		Constraint		
3			Type	Target	Type	Lower Bound	Upper Bound
4	P7 >= 1005	P7 - SIG	No Objective		Values >= Lower Bound	1005	
5	P8 <= 1610	P8 - DIS	No Objective		Values <= Upper Bound		1610
6	1300 <= P9 <= 1650	P9 - BUCK	No Objective		Lower Bound <= Values <= Upper Bound	1300	1650
7	P1	P1 - B	No Objective		No Constraint		
8	P2	P2 - D	No Objective		No Constraint		
9	P3	P3 - L	No Objective		No Constraint		
*		Select a Parameter					

In a Screening optimization, sample generation is driven by the domain definition, which is the lower and upper bounds for the parameter. Sample generation is not affected by parameter objective and constraint settings, unlike non-Screening optimization methods. However, parameter objective and constraint settings do affect the generation of candidate points.

When a constraint is defined, it is treated as a constraint if **Constraint Handling** in the **Properties** view for the **Optimization** cell is set to **Strict**. If **Constraint Handling** is set to **Relaxed**, those constraints are actually treated as objectives. If only constraints are defined, Screening is your only optimization method. If at least one objective is defined, the other optimization methods are also available.

Typically, the Screening method is best suited for conducting a preliminary design study. It is a low-resolution, fast, and exhaustive study that you can use to quickly locate approximate solutions. Because the Screening method does not depend on any parameter objectives, you can change the objectives after performing the analysis to view the candidates that meet different objective sets, allowing you to quickly perform preliminary design studies. It's easy to keep changing the objectives and constraints to view the different corresponding candidates, which are drawn from the original sample set.

For example, you can run a Screening optimization for 3,000 samples and then use Tradeoff or Samples charts to view the Pareto fronts. You can use the solutions slider in the **Properties** view for the chart to display only the prominent points in which you are interested, which are usually the first few fronts. When you run MOGA or Adaptive Multiple-Objective, you can limit the number of Pareto fronts that are computed in the analysis.

Once all of your objectives are defined, update the **Optimization** cell to generate up to the requested number of candidate points. You can save any of the candidates by right-clicking and selecting **Explore Response Surface at Points**, **Insert as Design Points**, or **Insert as Refinement Points**.

When working with a **Response Surface Optimization** system, you should validate the best obtained candidate results by saving the corresponding design points, solving them at the project level, and comparing the results.

Goal-Driven Optimization Theory

This section presents theoretical aspects of goal-driven optimization and the different optimization methods available in DesignXplorer.

[Sampling for Constrained Design Spaces](#)

[Shifted Hammersley Sampling \(Screening\)](#)

[Single-Objective Optimization Methods](#)

[Multiple-Objective Optimization Methods](#)

[Decision Support Process](#)

Sampling for Constrained Design Spaces

Common samplings created in a constrained design space might produce infeasible sample points if the underlying formulation disregards the constraints. DesignXplorer provides a sampling type that takes into account the constraints defined for input parameters.

DesignXplorer generates sampling:

$$s(X, Y)$$

subject to:

$$g_j(X, Y) \leq 0 \quad j=1, \dots, m$$

$$X \in \mathbb{R}^{N_c}$$

$$Y \in \mathbb{N}^{N_i}$$

$$X_L \leq X \leq X_U$$

$$Y_L \leq Y \leq Y_U$$

The symbols X and Y denote the vectors of the continuous and the integer variables, respectively. DesignXplorer allows you to define a constrained design space with linear or non-linear constraints.

Examples of linear constraints:

$$x_1 + x_2 \leq 1$$

$$x_1 \leq x_2$$

Examples of non-linear constraints:

$$x_1^2 + x_2^2 \leq 1$$

$$\log(x_1) \leq 10$$

The constraint sampling is a heuristic method based on Shifted-Hammersley (Screening) and MISQP sampling methods.

For a given screening of N sample points generated in the hypercube of input parameters, only a part of this sampling (M) is within the constrained design space (feasible domain).

To obtain a constraint sampling that is close to a uniform sampling of Nf points, DesignXplorer solves this problem first:

$$\min_S (\|Card(E) - Nf\|) \quad \text{where } E = \{x \text{ in } S | x \text{ is feasible}\}$$

If $Card(E) = Nf$, the sampling is completed.

If $Card(E) > Nf$, only the Nf furthestmost sample points are kept.

Otherwise, DesignXplorer needs to create additional points to reach the Nf sample points.

The Screening method is not guaranteed to find either enough sample points or at least one feasible point. For this reason, DesignXplorer solves an MISQP problem for each constraint on input parameters:

$$C_j = \min_{X, Y} g_j(X, Y)$$

subject to:

$$g_j(X, Y) \leq 0 \quad j=1, \dots, m$$

In the best case, you obtain m feasible points.

If A is the center of mass of the feasible points, then:

$$A = \frac{1}{m} \sum_{j=1}^m C_j$$

DesignXplorer then solves the new MISQP problem:

$$\tilde{A} = \min_X \|X - A\|_2$$

subject to:

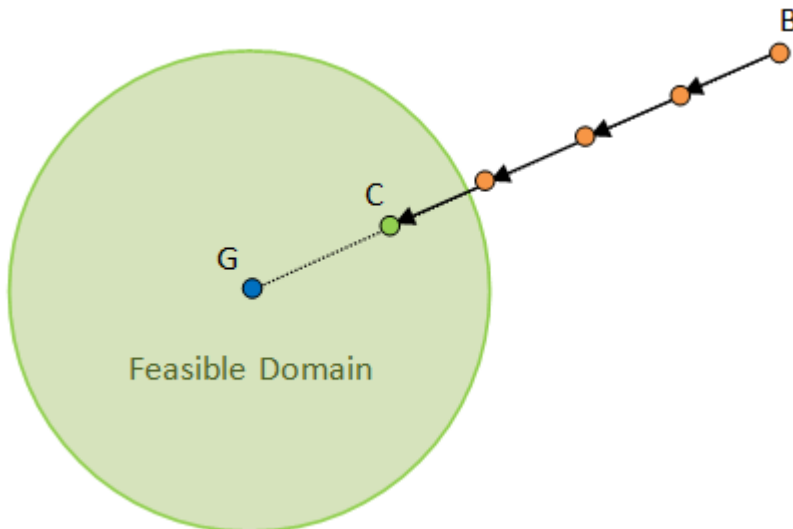
$$g_j(X, Y) \leq 0 \quad j=1, \dots, m$$

Once the point closest to the center mass of the feasible points has been found, DesignXplorer projects part of the infeasible points onto the skin of the feasible domain.

Given an infeasible point B and feasible point G , you can build a new feasible point C :

$$C = G + \alpha \cdot \vec{GB}$$

To find C close to the skin of the feasible domain, the algorithm starts with α equals 1, and decreases α value until C is feasible.



To optimize the distribution of point on the skin of the feasible domain, the algorithm chooses the furthestmost infeasible points in terms of angles:

If B is the first infeasible point processed, the next one is:

$$\min_x \frac{\vec{GB} \cdot \vec{GX}}{\|\vec{GB}\|_2 \|\vec{GX}\|_2}$$

Once enough points have been generated on the skin of the feasible domain, the algorithm generates internal points. The internal points are obtained by combination of internal points and points generated on the skin of the feasible domain.

Shifted Hammersley Sampling (Screening)

The Shifted Hammersley Sampling algorithm is used for sample generation by all methods except NLPQL. The conventional Hammersley sampling algorithm is a quasi-random number generator, which has very low discrepancy and is used for quasi-Monte Carlo simulations.

A low-discrepancy sequence is defined as a sequence of points that approximate the equidistribution in a multi-dimensional cube in an optimal way. This means that the design space is populated almost uniformly by these sequences and that dimensionality is not a problem because of the inherent properties of Monte Carlo sampling. The number of points does not increase exponentially with an increase in the number of input parameters.

The conventional Hammersley sampling algorithm is constructed by using the radical inverse function. Any integer n can be represented as a sequence of digits $n_0, n_1, n_2, \dots, n_m$ by the following equation:

$$n = n_0 n_1 n_2 n_3 \cdots n_m \quad (10)$$

For example, consider the integer 687459, which can be represented this way as $n_0=6, n_1=8$, and so on. Because this integer is represented with radix 10, you can write it as $687459 = 9 + 5 * 10 + 4 * 100$ and so on. In general, for a radix R representation, the equation is:

$$n = n_m + n_{m-1} * R + \cdots + n_0 \quad (11)$$

The inverse radical function is defined as the function that generates a fraction in (0, 1) by reversing the order of the digits in [Equation 11 \(p. 360\)](#) about the decimal point, as shown.

$$\begin{aligned} \Phi_R(n) &= 0 n_m n_{m-1} n_{m-2} \cdots n_0 \\ &= n_m * R^{-1} + n_{m-1} * R^{-2} + \cdots + n_0 * R^{-(m-1)} \end{aligned} \quad (12)$$

Thus, for a k -dimensional search space, the Hammersley points are given by the following expression:

$$H_k(i) = [i/N, \Phi_{R1}(i), \Phi_{R2}(i), \dots, \Phi_{Rk-1}(i)] \quad (13)$$

where $i=0, \dots, N$ indicates the sample points. Now, from the plot of these points, it is seen that the first row (corresponding to the first sample point) of the Hammersley matrix is zero and the last row is not 1. This implies that, for the k -dimensional hypercube, the Hammersley sampler generates a block of points that are skewed more toward the origin of the cube and away from the far edges and faces. To compensate for this bias, a point-shifting process is proposed that shifts all Hammersley points by the amount that follows:

$$\Delta = \frac{1}{2} N \quad (14)$$

This moves the point set more toward the center of the search space and avoids unnecessary bias. Thus, the initial population always provides unbiased, low-discrepancy coverage of the search space.

Single-Objective Optimization Methods

Descriptions follow for the different types of single-objective optimization methods:

Nonlinear Programming by Quadratic Lagrangian (NLPQL)

Mixed-Integer Sequential Quadratic Programming (MISQP)

Adaptive Single-Objective Optimization (ASO)

Nonlinear Programming by Quadratic Lagrangian (NLPQL)

NLPQL (Nonlinear Programming by Quadratic Lagrangian) is a mathematical optimization algorithm developed by Klaus Schittkowski. It solves constrained nonlinear programming problems of the form

minimize:

$$f = f(\{x\})$$

subject to:

$$g_k(\{x\}) \leq 0, \forall k=1, 2, \dots, K$$

$$h_l(\{x\}) = 0, \forall l=1, 2, \dots, L$$

where:

$$\{x_L\} \leq \{x\} \leq \{x_U\}$$

It is assumed that objective function and constraints are continuously differentiable. The idea is to generate a sequence of quadratic programming subproblems obtained by a quadratic approximation of the Lagrangian function and a linearization of the constraints. Second order information is updated by a quasi-Newton formula and the method is stabilized by an additional (Armijo) line search.

The method presupposes that the problem size is not too large and that it is well-scaled. Also, the accuracy of the methods depends on the accuracy of the gradients. Because analytical gradients are unavailable for most practical problems, it is imperative that the numerical (finite difference based) gradients are as accurate as possible.

Newton's Iterative Method

Before the actual derivation of the NLPQL equations, Newton's iterative method for the solution of nonlinear equation sets is reviewed. Let $f(x)$ be a multivariable function such that it can be expanded about the point x in a Taylor's series.

$$f(x+\Delta x) \approx f(x) + \{\Delta x\}^T \{f'(x)\} + \left(\frac{1}{2}\right) \{\Delta x\}^T [f''(x)] \{\Delta x\} \quad (15)$$

where it is assumed that the Taylor series actually models a local area of the function by a quadratic approximation. The objective is to devise an iterative scheme by linearizing the vector [Equation 15 \(p. 361\)](#). To this end, it is assumed that at the end of the iterative cycle, the [Equation 15 \(p. 361\)](#) would be exactly valid. This implies that the first variation of the following expression with respect to Δx must be zero.

$$\phi(\Delta x) = f(x + \Delta x) - \left(f(x) + \{\Delta x\}^T \{f'(x)\} + \left(\frac{1}{2}\right) \{\Delta x\}^T [f''(x)] \{\Delta x\} \right) \quad (16)$$

Which implies that

$$f(x + \Delta x)_{,\Delta x} - (\{f'(x)\} + [f''(x)] \{\Delta x\}) = 0 \quad (17)$$

The first expression indicates the first variation of the converged solution with respect to the increment in the independent variable vector. This gradient is necessarily zero because the converged solution clearly does not depend on the step-length. Therefore, [Equation 17 \(p. 362\)](#) can be written as the following:

$$\{x_{j+1}\} = \{x_j\} - [f''(x_j)]^{-1} \{f'(x_j)\} \quad (18)$$

where the index j indicates the iteration. [Equation 18 \(p. 362\)](#) is therefore used in the main quadratic programming scheme.

NLPQL Derivation:

Consider the following single-objective nonlinear optimization problem. It is assumed that the problem is smooth and analytic throughout and is a problem of N decision variables.

minimize:

$$f = f(\{x\})$$

subject to:

$$g_k(\{x\}) \leq 0, \forall k = 1, 2, \dots, K$$

$$h_l(\{x\}) = 0, \forall l = 1, 2, \dots, L$$

where:

$$\{x_L\} \leq \{x\} \leq \{x_U\} \quad (19)$$

where K and L are the numbers of inequality and equality constraints. In many cases the inequality constraints are bound above and below, in such cases, it is customary to split the constraint into two inequality constraints. To approximate the quadratic sub-problem assuming the presence of only equality constraints, the Lagrangian for [Equation 19 \(p. 362\)](#) as:

$$A(\{x\}, \{\lambda\}) = f(\{x\}) + \{\lambda\}^T \{h(\{x\})\} \quad (20)$$

where $\{\lambda\}$ is a L dimensional vector (non-zero), which are used as Lagrange multipliers. Thus, [Equation 20 \(p. 362\)](#) becomes a functional, which depends on two sets of independent vectors. To minimize this expression, you seek the stationarity of this functional with respect to the two sets of vectors. These expressions give rise to two sets of vector expressions as the following:

$$\begin{aligned} A(\{x\}, \{\lambda\})_{,x} &= \{\nabla A\} = \{\nabla f\} + [H] \{\lambda\} = \{0\} \\ A(\{x\}, \{\lambda\})_{,\lambda} &= \{h(\{x\})\} = \{0\} \end{aligned} \quad (21)$$

[Equation 21 \(p. 362\)](#) defines the Karush-Kuhn-Tucker (KKT) conditions that are the necessary conditions for the existence of the optimal point. The first equation is composed of N nonlinear al-

gebraic equations and the second one is made up of L nonlinear algebraic equations. The matrix $[H]$ is a (N^*L) matrix defined as the following:

$$[H]_{(N^*L)} = [\{\nabla h_1(\{x\})\}, \{\nabla h_2(\{x\})\}, \{\nabla h_3(\{x\})\}, \dots] \quad (22)$$

Thus, in Equation 22 (p. 363), each column is a gradient of the corresponding equality constraint. For convenience, the nonlinear equations of Equation 21 (p. 362) can be written as the following:

$$\{F(\{Y\})\} = \{0\} \quad (23)$$

where Equation 23 (p. 363) is a $(N+L)$ system of nonlinear equations. The independent variable set $\{Y\}$ can be written as:

$$\{Y\} = \begin{Bmatrix} \{x\} \\ \{\lambda\} \end{Bmatrix} \quad (24)$$

While the functional set $\{F\}$ is written as the following:

$$\{F\} = \begin{Bmatrix} \{\nabla \Lambda\} \\ \{h\} \end{Bmatrix} \quad (25)$$

Referring to the section on Newton based methods, the vector $\{Y\}$ in Equation 24 (p. 363) is updated as the following:

$$\{Y_{j+1}\} = \{Y_j\} + \{\Delta Y_j\} \quad (26)$$

The increment of the vector is given by the iterative scheme in Equation 18 (p. 362). Referring to Equation 23 (p. 363), Equation 24 (p. 363), and Equation 25 (p. 363), the iterative equation is expressed as the following:

$$[\nabla F_j] \{\Delta Y_j\} = -\{F(\{Y_j\})\} \quad (27)$$

This is only a first order approximation of the Taylor expansion of Equation 23 (p. 363). This is in contrast to Equation 18 (p. 362) where a quadratic approximation is done. This is because in Equation 24 (p. 363), a first order approximation has already been done. The matrices and the vectors in Equation 27 (p. 363) can be expanded as the following:

$$\{\Delta Y\} = \begin{Bmatrix} \{\Delta x\} \\ \{\Delta \lambda\} \end{Bmatrix} \quad (28)$$

and

$$[\nabla F] = \begin{bmatrix} [\nabla^2 \Lambda]_{(N^*N)} & [H]_{(N^*L)} \\ [H]^T_{(L^*N)} & [0]_{(L^*L)} \end{bmatrix}_{((N+L)^*(N+L))} \quad (29)$$

This is obtained by taking the gradients of Equation 25 (p. 363) with respect to the two variable sets. The sub-matrix $[\nabla^2 \Lambda]_{(N^*N)}$ is the (N^*N) Hessian of the Lagrange function in implicit form.

To demonstrate how Equation 29 (p. 363) is formed, let us consider the following simple case. Given a vector of two variables x and y , you write:

$$\{V\} = \begin{cases} \{a(x,y)\} \\ \{b(x,y)\} \end{cases} = \begin{pmatrix} a1(x,y) \\ a2(x,y) \\ b1(x,y) \\ b2(x,y) \end{pmatrix}$$

It is required to find the gradient or Jacobian of the vector function V). To this effect, the derivation of the Jacobian is evident because in the present context, the vector V indicates a set of nonlinear (algebraic) equations and the Jacobian is the coefficient matrix that "links" the increment of the independent variable vector to the dependent variable vector. Thus, you can write:

$$\{\Delta V\} = \begin{cases} \{\Delta a(x,y)\} \\ \{\Delta b(x,y)\} \end{cases} = \begin{bmatrix} \frac{\partial a1}{\partial x} & \frac{\partial a1}{\partial y} \\ \frac{\partial a2}{\partial x} & \frac{\partial a2}{\partial y} \\ \frac{\partial b1}{\partial x} & \frac{\partial b1}{\partial y} \\ \frac{\partial b2}{\partial x} & \frac{\partial b2}{\partial y} \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

Thus, the Jacobian matrix is formed. In some applications, the equation is written in the following form:

$$\{\Delta V\} = \begin{cases} \{\Delta a(x,y)\} \\ \{\Delta b(x,y)\} \end{cases} = \begin{bmatrix} \frac{\partial a1}{\partial x} & \frac{\partial a2}{\partial x} & \frac{\partial b1}{\partial x} & \frac{\partial b2}{\partial x} \\ \frac{\partial a1}{\partial y} & \frac{\partial a2}{\partial y} & \frac{\partial b1}{\partial y} & \frac{\partial b2}{\partial y} \end{bmatrix}^T \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

where the column i indicates the gradient of the i -th component of the dependent variable vector with respect to the independent vector. This is the formalism you use in determining [Equation 29 \(p. 363\)](#).

[Equation 27 \(p. 363\)](#) can be rewritten as the following:

$$\begin{bmatrix} [\nabla^2 \Lambda]_{(N*N)} & [H]_{(N*L)} \\ [H]^T_{(L*N)} & [0]_{(L*L)} \end{bmatrix}^{(j)} \begin{pmatrix} \{\Delta x\} \\ \{\Delta \lambda\} \end{pmatrix}^{(j)} = - \begin{pmatrix} \{\nabla \Lambda\} \\ \{h\} \end{pmatrix}^{(j)} \quad (30)$$

Solution of [Equation 30 \(p. 364\)](#) iteratively solves [Equation 31 \(p. 364\)](#) in a series of linear steps till the point when the increment is negligible. The update schemes for the independent variable and Lagrange multiplier vectors x and λ are written as the following:

$$\begin{aligned} \{x_{j+1}\} &= \{x_j\} + \{\Delta x_j\} \\ \{\lambda_{j+1}\} &= \{\lambda_j\} + \{\Delta \lambda_j\} \end{aligned} \quad (31)$$

The individual equations of the [Equation 30 \(p. 364\)](#) are written separately now. The first equation (corresponding to minimization with respect to x) can be written as:

$$\begin{aligned} [\nabla^2 \Lambda]_j \{\Delta x\}_j + [H]_j \{\Delta \lambda\}_j &= -\{\nabla \Lambda\}_j \\ [\nabla^2 \Lambda]_j \{\Delta x\}_j + [H]_j \{\Delta \lambda\}_j &= -\{\nabla f\}_j - [H]_j \{\lambda\}_j \\ [\nabla^2 \Lambda]_j \{\Delta x\}_j + [H]_j \{\lambda\}_{j+1} &= -\{\nabla f\}_j \end{aligned} \quad (32)$$

The last step in Equation 32 (p. 364) is done by using Equation 31 (p. 364). Thus, using Equation 32 (p. 364) and Equation 30 (p. 364), the iterative scheme can be rewritten in a simplified form as:

$$\begin{bmatrix} [\nabla^2 \Lambda]_{(N^*N)} & [H]_{(N^*L)} \\ [H]^T_{(L^*N)} & [0]_{(L^*L)} \end{bmatrix}^{(j)} \begin{Bmatrix} \{\Delta x\}^{(j)} \\ \{\lambda\}^{(j+1)} \end{Bmatrix} = - \begin{Bmatrix} \{\nabla f\} \\ \{h\} \end{Bmatrix}^{(j)} \quad (33)$$

Thus, Equation 33 (p. 365) can be used directly to compute the $(j+1)$ th value of the Lagrange multiplier vector. Note that by using Equation 33 (p. 365), it is possible to compute the update of x and the new value of the Lagrange multiplier vector λ in the same iterative step. Equation 33 (p. 365) shows the general scheme by which the KKT optimality condition can be solved iteratively for a generalized optimization problem.

Now, consider the following quadratic approximation problem as given by:

$$Q = \{\nabla f\}^T \{\Delta x\} + \left(\frac{1}{2}\right) \{\Delta x\}^T [\nabla^2 \Lambda] \{\Delta x\} \quad (34)$$

subject to the equality constraints given by the following:

$$\{h\}_{(L^*1)} + [H]^T_{(L^*N)} \{\Delta x\} = \{0\} \quad (35)$$

where the definition of the matrices in Equation 34 (p. 365) and Equation 35 (p. 365) are given earlier. To solve the quadratic minimization problem let us form the Lagrangian as:

$$\Gamma = \{\nabla f\}^T \{\Delta x\} + \left(\frac{1}{2}\right) \{\Delta x\}^T [\nabla^2 \Lambda] \{\Delta x\} + \{\lambda\}^T \{h\} + \{\lambda\}^T [H]^T \{\Delta x\} \quad (36)$$

Now, the KKT conditions can be derived (as done earlier) by taking gradients of the Lagrangian in Equation 36 (p. 365) as the following:

$$\begin{aligned} \Gamma_{,\{\Delta x\}} &= \{\nabla f\} + [\nabla^2 \Lambda] \{\Delta x\} + [H] \{\lambda\} = \{0\} \\ \Gamma_{,\{\lambda\}} &= \{h\} + [H]^T \{\Delta x\} = \{0\} \end{aligned} \quad (37)$$

In a condensed matrix form Equation 37 (p. 365) can be written as the following:

$$\begin{bmatrix} [\nabla^2 \Lambda] & [H] \\ [H]^T & [0] \end{bmatrix} \begin{Bmatrix} \{\Delta x\} \\ \{\lambda\} \end{Bmatrix} = - \begin{Bmatrix} \{\nabla f\} \\ \{h\} \end{Bmatrix} \quad (38)$$

Equation 38 (p. 365) is the same as Equation 33 (p. 365). This implies that the iterative scheme in Equation 38 (p. 365), which actually solves a quadratic subproblem (Equation 34 (p. 365) and Equation 35 (p. 365)) in the domain Δx . In case the real problem is quadratic, then this iterative scheme solves the exact problem.

On addition of inequality constraints, the Lagrangian of the actual problem can be written as the following:

$$\Lambda(\{x\}, \{\lambda\}, \{\mu\}, \{y\}) = f(\{x\}) + \{\lambda\}^T \{h(\{x\})\} + \{\mu\}^T \{g(\{x\})\} + \{\mu\}^T \{y^2\} \quad (39)$$

The inequality constraints have been converted to equality constraints by using a set of slack variables y as the following:

$$g_k(\{x\}) + (y_k)^2 = 0, \forall k = 1, 2, \dots, M \quad (40)$$

The squared term is used to ensure that the slack variable remains positive, which is required to satisfy Equation 40 (p. 365). The Lagrangian in Equation 39 (p. 365) acts as an enhanced objective function. It is seen that the only case where the additional terms might be active is when the constraints are not satisfied.

The KKT conditions as derived from Equation 39 (p. 365) (by taking first variations with respect to the independent variable vectors) are:

$$\begin{aligned}
 \Lambda_{\{x\}} &= \{\nabla \Lambda\} = \{\nabla f\} + [H]\{\lambda\} + [G]\{\mu\} = \{0\}_{(N^*1)} \\
 \Lambda_{\{\lambda\}} &= \{h(\{x\})\} = \{0\}_{(L^*1)} \\
 \Lambda_{\{\mu\}} &= \{g(\{x\})\} = \{y^2\} = \{0\}_{(M^*1)} \\
 \Lambda_{\{y\}} &= \{2y\mu\} = \{0\}_{(M^*1)}
 \end{aligned} \tag{41}$$

From the KKT conditions in Equation 41 (p. 366), it is evident that there are $(N+L+2^*M)$ equations for a similar number of unknowns. Therefore, this equation set possesses a unique solution. Let this (optimal) solution be marked as x . At this point, a certain number of constraints are active while others are inactive. Let the number of active inequality constraints be p and the total number of active equality constraints be q . By an active constraint, it is assumed that the constraint is at its threshold value of zero. Therefore, let J_1 and J_2 be the sets of active and inactive equality constraints (respectively) and K_1 and K_2 be the sets of active and inactive inequality constraints respectively. Therefore, you can write the following relations:

$$\begin{aligned}
 J_1 \cap J_2 &= \emptyset; \text{meas}(J_1 \cup J_2) = L; \text{meas}(J_1) = q; \text{meas}(J_2) = (L - q) \\
 K_1 \cap K_2 &= \emptyset; \text{meas}(K_1 \cup K_2) = M; \text{meas}(K_1) = p; \text{meas}(K_2) = (M - p)
 \end{aligned} \tag{42}$$

where $\text{meas}()$ indicates the count of the elements of the set under consideration. These sets partition the constraints into active and inactive sets. Therefore, you can write:

$$\begin{aligned}
 h_k(\{x\}) &= 0, \forall k \in J_2 \\
 g_j(\{x\}) &= 0, y_j = 0; \mu_j \neq 0; \forall j \in K_1 \\
 g_j(\{x\}) &< 0, y_j \neq 0; \mu_j = 0; \forall j \in K_2
 \end{aligned} \tag{43}$$

Thus, the last three equations in Equation 41 (p. 366) can be represented by the Equation 43 (p. 366). These are the optimality conditions for constraint satisfaction. From these equations, you can now eliminate y such that the Lagrangian in Equation 39 (p. 365) depends on only three independent variable vectors. From the last two conditions in Equation 43 (p. 366), you can write the following condition, which is always valid for an optimal point:

$$\mu_j g_j(\{x\}) = 0 \tag{44}$$

Using Equation 44 (p. 366) in the set Equation 41 (p. 366), the KKT optimality conditions can be written as the following:

$$\begin{aligned}
 \{\nabla \Lambda\} &= \{\Delta f\} + [H]\{\lambda\} + [G]\{\mu\} = \{0\}_{(N^*1)} \\
 \{h(\{x\})\} &= \{0\}_{(L^*1)} \\
 \{\mu g(\{x\})\} &= \{0\}_{(M^*1)}
 \end{aligned} \tag{45}$$

Thus, the new set contains only $(N+L+M)$ unknowns. Now, following the same logic as in Equation 24 (p. 363) as done earlier—let us express Equation 45 (p. 366) in the same form as in

Equation 23 (p. 363). This represents a $(N+L+M)$ system of nonlinear equations. The independent variable set can be written in vector form as the following:

$$\{Y\} = \begin{Bmatrix} \{x\} \\ \{\lambda\} \\ \{\mu\} \end{Bmatrix} \quad (46)$$

Newton's iterative scheme is also used here. Therefore, the same equations as in Equation 26 (p. 363) and Equation 27 (p. 363) also apply here. Following Equation 27 (p. 363), you can write:

$$\{\Delta Y\} = \begin{Bmatrix} \{\Delta x\} \\ \{\Delta \lambda\} \\ \{\Delta \mu\} \end{Bmatrix} \quad (47)$$

Taking the first variation of the KKT equations in Equation 45 (p. 366) and equating to zero, the sub-quadratic equation is formulated as the following:

$$\begin{bmatrix} [\nabla^2 \Lambda] & [H]^T & [G]^T \\ [H] & 0 & 0 \\ [\mu][G] & 0 & [g] \end{bmatrix} \begin{Bmatrix} \delta x \\ \delta \lambda \\ \delta \mu \end{Bmatrix} = - \begin{Bmatrix} \{\nabla \Lambda\} \\ \{h\} \\ \{\mu g\} \end{Bmatrix} \quad (48)$$

where: $[\mu]$ indicates a diagonal matrix.

At the j -th step, the first equation can be written (by linearization) as:

$$[\nabla^2 \Lambda]_j \{\Delta x\}_j + [H]_j \{\Delta \lambda\}_j + [G]_j \{\Delta \mu\}_j = -\{\nabla f\}_j - [H]_j \{\lambda\}_j - [G]_j \{\mu\}_j \quad (49)$$

which simplifies to:

$$[\nabla^2 \Lambda]_j \{\Delta x\}_j + [H]_j \{\lambda\}_{j+1} + [G]_j \{\mu\}_{j+1} = -\{\nabla f\}_j \quad (50)$$

Thus, the linearized set of equations for Newton's method to be applied can be written in an explicit form as:

$$\begin{bmatrix} [\nabla^2 \Lambda] & [H]_j^T & [G]_j^T \\ [H]_j & 0 & 0 \\ [\mu]_j [G]_j & 0 & [g]_j \end{bmatrix} \begin{Bmatrix} \{\Delta x\}_j \\ \{\lambda\}_{j+1} \\ \{\mu\}_{j+1} \end{Bmatrix} = - \begin{Bmatrix} \{\nabla f\}_j \\ \{h\}_j \\ 2\{\mu g\}_j \end{Bmatrix} \quad (51)$$

So, in presence of both equality and inequality constraints, Equation 51 (p. 367) can be used in a quasi-Newtonian framework to determine the increments $\{\Delta x\}_j$ and Lagrange multipliers $\{\lambda\}_{j+1}$ and $\{\mu\}_{j+1}$ when stepping from iteration j to $j+1$.

The Hessian matrix $[\nabla^2 \Lambda]$ is not computed directly but is estimated and updated in a BFGS type line search.

Mixed-Integer Sequential Quadratic Programming (MISQP)

MISQP (Mixed-Integer Sequential Quadratic Programming) is a mathematical optimization algorithm as developed by Oliver Exler, Thomas Lehmann and Klaus Schittkowski (NLPQL). This method solves Mixed-Integer Non-Linear Programming (MINLP) of the form:

minimize:

$$f(x,y)$$

subject to:

$$g_j(x,y)=0, \quad j=1,\dots,m_e,$$

$$g_j(x,y)\geq 0, \quad j=m_e+1,\dots,m$$

where:

$$x \in \mathbb{R}^{n_c}, y \in \mathbb{N}^{n_i}$$

$$x_l \leq x \leq x_u$$

$$y_l \leq y \leq y_u$$

The symbols x and y denote the vectors of the continuous and integer variables, respectively. It is assumed that problem functions $f(x,y)$ and $g_j(x,y)$, $j=1,\dots,m$ are continuously differentiable subject to all $x \in \mathbb{R}^{n_c}$. It is not assumed that integer variables can be relaxed. In other words, problem functions are evaluated only at integer points and never at any fractional values in between.

MISQP solves MINLP by a modified sequential quadratic programming (SQP) method. After linearizing constraints and constructing a quadratic approximation of the Lagrangian function, mixed-integer quadratic programs are successively generated and solved by an efficient branch-and-cut method. The algorithm is stabilized by a trust region method as originally proposed by Yuan for continuous programs. Second order corrections are retained. The Hessian of the Lagrangian function is approximated by BFGS updates subject to the continuous and integer variables. MISQP is able to solve also non-convex nonlinear mixed-integer programs.

For external references, see [MISQP Optimization Algorithm References](#) (p. 406).

Adaptive Single-Objective Optimization (ASO)

Adaptive Single-Objective (ASO) is a mathematical optimization method that combines an OSF (Optimal Space-Filling) DOE, a Kriging response surface, and MISQP. It is a gradient-based algorithm based on a response surface, which provides a refined, global, optimized result.

ASO supports a single objective and multiple constraints. It is available for continuous parameters, including those with manufacturable values. It does not support the use of parameter relationships in the optimization domain and is available only for a **Direct Optimization** system.

Like MISQP, ASO solves constrained nonlinear programming problems of the form:

minimize:

$$f = f(\{x\})$$

subject to:

$$g_k(\{x\}) \leq 0, \forall k=1,2,\dots,K$$

$$h_l(\{x\}) = 0, \forall l=1,2,\dots,L$$

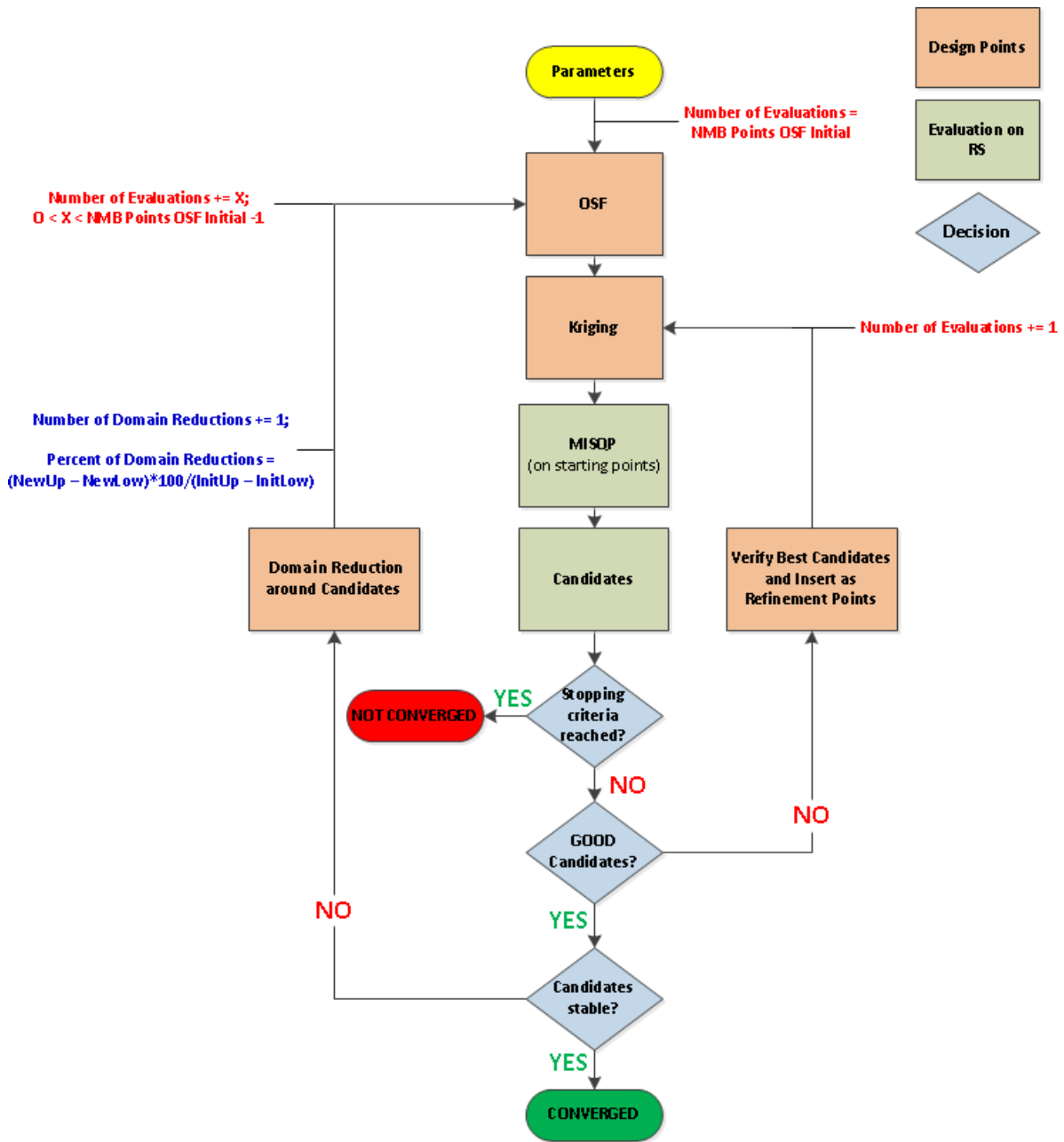
where:

$$\{x_L\} \leq \{x\} \leq \{x_U\}$$

The purpose is to refine and reduce the domain intelligently and automatically to provide the global extrema.

ASO Workflow

The workflow of ASO is as follows:



ASO Steps

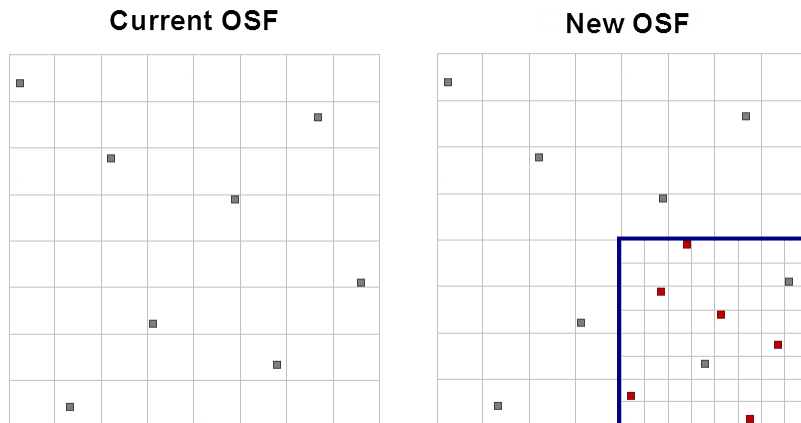
OSF Sampling

OSF (Optimal Space-Filling Design) is used for the Kriging construction. In the original OSF, the number of samples equals the number of divisions per axis and there is one sample in each division.

When a new OSF is generated after a domain reduction, the reduced OSF has the same number of divisions as the original and keeps the existing design points within the new bounds. New design points are added until there is a point in each division of the reduced domain.

In the following example, the original domain has eight divisions per axis and contains eight design points. The reduced domain also has eight divisions per axis and includes two of the ori-

ginal design points. To have a design point in each division, six new design points need to be added.



Note:

The total number of design points in the reduced domain can exceed the number in the original domain if multiple existing points wind up in the same division. In the previous example above, if two existing points wound up in the same division of the new domain, seven new design points (rather than six) would have been added to have a point in each of the remaining divisions.

Kriging Generation

A response surface is created for each output, based on the current OSF and consequently on the current domain bounds.

For more information on Kriging algorithms, see [Kriging \(p. 340\)](#).

MISQP

MISQP is run on the current Kriging response surface to find potential candidates. A few MISQP processes are run at the same time, beginning with different starting points, and consequently, giving different candidates.

Candidate Point Validation

All the obtained candidates are either validated or not, based on the Kriging error predictor. The candidate point is checked to see if further refinement of the Kriging surface changes the selection of this point. A candidate is considered as acceptable if there aren't any points, according to this error prediction, that call it into question. If the quality of the candidate is not called into question, the domain bounds are reduced. Otherwise, the candidate is calculated as a verification point.

- **Refinement Point Creation** (If the selection is *not* to be changed)

When a new verification point is calculated, it is inserted in the current Kriging response surface as a refinement point and the MISQP process is restarted.

- **Domain Reduction** (If the selection is to be changed)

When candidates are validated, new domain bounds must be calculated. If all of the candidates are in the same zone, the bounds are reduced, centered on the candidates. Otherwise, the bounds are reduced as an inclusive box of all candidates. At each domain reduction, a new OSF is generated (conserving design points between the new bounds) and a new Kriging response surface is generated based on this new OSF.

Multiple-Objective Optimization Methods

This section describes both theoretical aspects and the different optimization methods available for multi-objective optimization:

[Pareto Dominance in Multi-Objective Optimization](#)

[Convergence Criteria in MOGA-Based Multi-Objective Optimization](#)

[Multi-Objective Genetic Algorithm \(MOGA\)](#)

[Adaptive Multiple-Objective Optimization](#)

Pareto Dominance in Multi-Objective Optimization

The concept of Pareto dominance is of extreme importance in multi-objective optimization, especially where some or all of the objectives and constraints are mutually conflicting. In such a case, there is no Utopia point, which is a single point that yields the "best" value for all objectives and constraints. Instead, the best solutions, often called a Pareto or non-dominated set, are a group of solutions such that selecting any one of them in place of another always sacrifices quality for at least one objective or constraint, while improving at least one other. Formally, the description of such Pareto optimality for generic optimization problems can be formulated as in the following equations.

Taking a closer, more formal look at the multi-objective optimization problem, let the following denote the set of all feasible solutions, which are those that do not violate constraints:

$$X := \{x \in \mathcal{X}'' : g(x) \geq 0, h(x) = 0, x_l \leq x \leq x_u\} \quad (52)$$

The problem can then be simplified to:

$$\min_{x \in X} \vec{f}(x) \quad (53)$$

If there exists $x^* \in X$ such that for all objective functions x^* is optimal. This, for $i=1, \dots, k$, is expressed:

$$f_i(x^*) \leq f_i(x) \quad \forall x \in X \quad (54)$$

This indicates that x^* is certainly a desirable solution. Unfortunately, this is a utopian situation that rarely exists, as it is unlikely that all $f_1(x)$ has minimum values for X at a common point (x^*). The question is left: What solution should be used? That is, how should an "optimal" solution be defined? First, consider the so-called ideal (utopian) solution. To define this solution, separately attainable minima must be found for all objective functions. Assuming there is one, let x^{*1} be the solution of the scalar optimization problem:

$$\min_{x \in X} f_i(x) = f_i^* \quad (55)$$

Here f_1^* is called the individual minimum for the scalar problem i . The vector $f^* = (f_1^*, \dots, f_k^*)^i$ is called ideal for a multi-objective optimization problem, and the points in X , which determined this vector is the ideal solution.

It is usually not true that Equation 56 (p. 373) holds, although it would be useful, as the multi-objective problem would have been solved by considering a sequence for scalar problems. It is necessary to define a new form of optimality, which leads to the concept of Pareto Optimality. Introduced by V. Pareto in 1896, it is still the most important part of multi-objective optimization.

$$f^* = (f_1^*, \dots, f_k^*)^i \quad (56)$$

A point $x' \in X$ is said to be Pareto Optimal for the problem if there is no other vector $x \in X$ such that for all $i=1, \dots, k$,

$$f_i(x) \leq f_i(x^*) \quad (57)$$

And, for at least one $i \in \{1, \dots, k\}$:

$$f_i(x) < f_i(x^*) \quad (58)$$

This definition is based on the intuitive conviction that the point $x^* \in X$ is chosen as the optimal if no criterion can be improved without worsening at least one other criterion. Unfortunately, the Pareto optimum almost always gives not a single solution, but a set of solutions. Usually Pareto optimality is spoken of as being global or local depending on the neighborhood of the solutions X , and in this case, almost all traditional algorithms can at best guarantee a local Pareto optimality. However, this MOGA-based system, which incorporates global Pareto filters, yields the global Pareto front.

Convergence Criteria in MOGA-Based Multi-Objective Optimization

Convergence criteria are the conditions that indicate when the optimization has converged. In the MOGA-based multi-objective optimization methods, the following convergence criteria are available:

Maximum Allowable Pareto Percentage

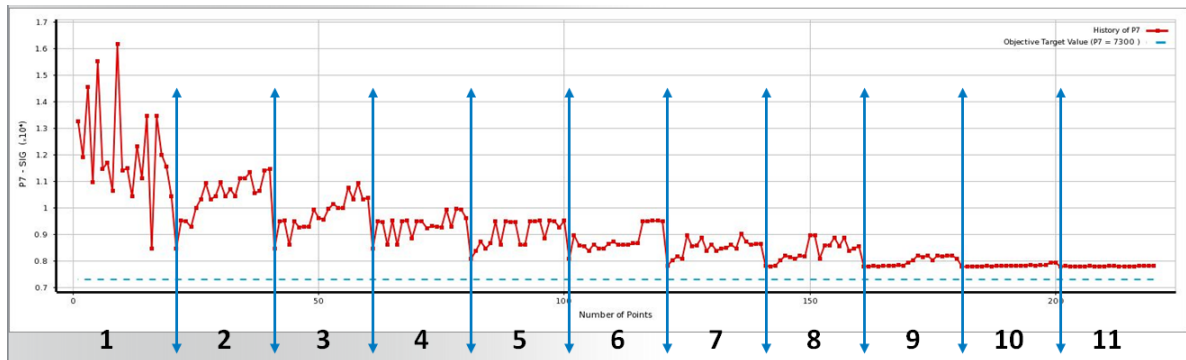
The **Maximum Allowable Pareto Percentage** criterion looks for a percentage that represents a specified ratio of Pareto points per Number of Samples Per Iteration. When this percentage is reached, the optimization is converged.

Convergence Stability Percentage

The **Convergence Stability Percentage** criterion looks for population stability, based on mean and standard deviation of the output parameters. When a population is stable with regards to the previous one, the optimization is converged. The criterion functions in the following sequence:

- **Population 1:** When the optimization is run, the first population is not taken into account. Because this population was not generated by the MOGA algorithm, it is not used as a range reference for the output range (for scaling values).
- **Population 2:** The second population is used to set the range reference. The minimum, maximum, range, mean, and standard deviation are calculated for this population.

- **Populations 3 – 11:** Starting from the third population, the minimum and maximum output values are used in the next steps to scale the values (on a scale of 0 to 100). The mean variations and standard deviation variations are checked. If both of these are smaller than the value for the **Convergence Stability Percentage** property, the algorithm is converged.



At each iteration and for each active output, convergence occurs if:

$$\frac{|Mean_i - Mean_{i-1}|}{Max - Min} < \frac{S}{100}$$

and

$$\frac{|StdDev_i - StdDev_{i-1}|}{Max - Min} < \frac{S}{100}$$

where:

S = Stability Percentage

$Mean_i$ = Mean of the i -th Population

$StdDev_i$ = Standard Deviation of the i -th Population

Max = Maximum Output Value calculated on the first generated population of MOGA

Min = Minimum Output Value calculated on the first generated population of MOGA

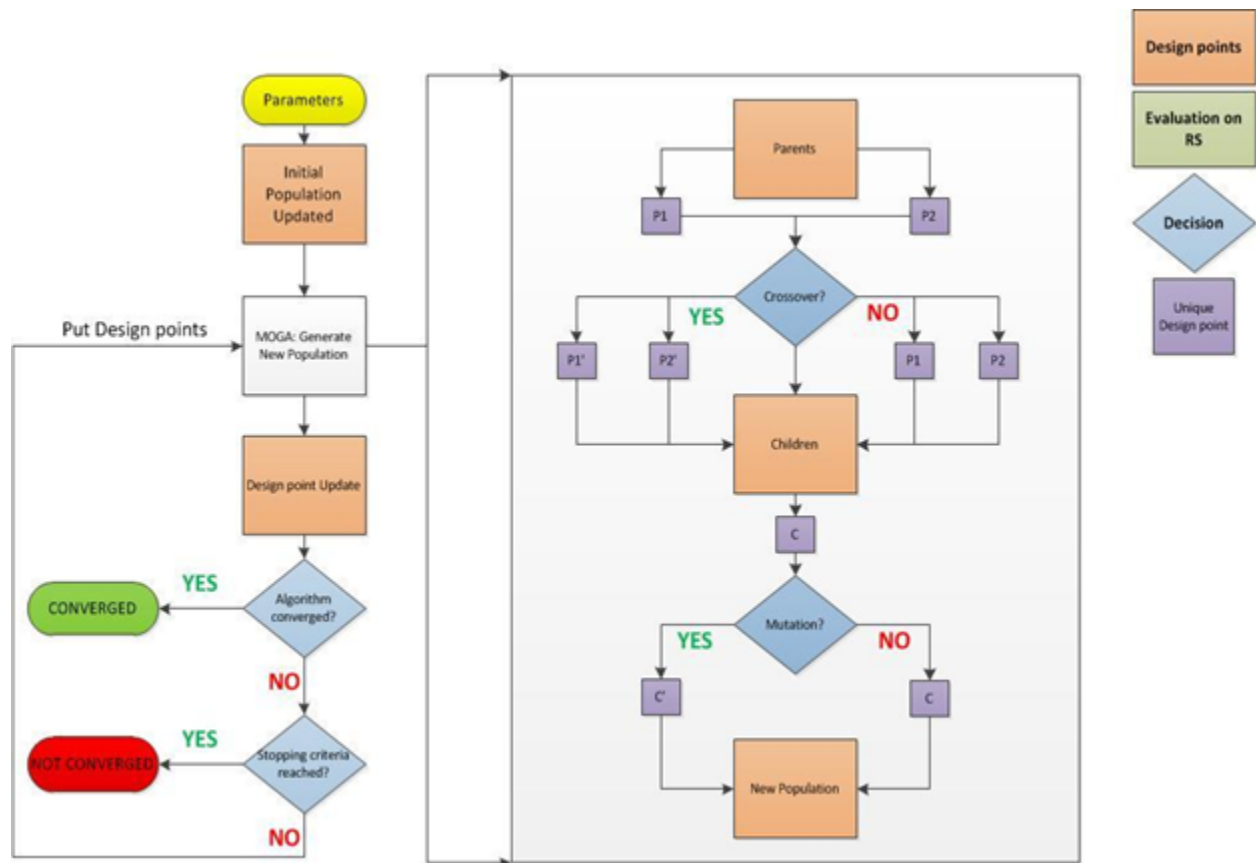
Multi-Objective Genetic Algorithm (MOGA)

The Multi-Objective Genetic Algorithm (MOGA) used in GDO is a hybrid variant of the popular NSGA-II (Non-dominated Sorted Genetic Algorithm-II) based on controlled elitism concepts. It supports all types of input parameters. The Pareto ranking scheme is done by a fast, non-dominated sorting method that is an order of magnitude faster than traditional Pareto ranking methods. The constraint handling uses the same non-dominance principle as the objectives. Therefore, penalty functions and Lagrange multipliers are not needed. This also ensures that the feasible solutions are always ranked higher than the infeasible solutions.

The first Pareto front solutions are archived in a separate sample set internally and are distinct from the evolving sample set. This ensures minimal disruption of Pareto front patterns already available from earlier iterations. You can control the selection pressure (and, consequently, the elitism of the process) to avoid premature convergence by altering the **Maximum Allowable Pareto Percentage** property. For more information about this and other MOGA properties, see [Performing a MOGA Optimization \(p. 183\)](#).

MOGA Workflow

The MOGO workflow follows:



MOGA Steps

1. First Population of MOGA

The initial population is used to run MOGA.

2. MOGA Generates a New Population

MOGA is run and generates a new population via cross-over and mutation. After the first iteration, each population is run when it reaches the number of samples defined by the **Number of Samples Per Iteration** property. For details, see [MOGA Steps to Generate a New Population](#) (p. 376).

3. Design Point Update

The design points in the new population are updated.

4. Convergence Validation

The optimization is validated for convergence.

- **Yes: Optimization Converged**

MOGA converges when the **Maximum Allowable Pareto Percentage** or the **Convergence Stability Percentage** has been reached.

- **No: Optimization Not Converged**

If the optimization is not converged, the process continues to the next step.

5. Stopping Criteria Validation

If the optimization has not converged, it is validated for fulfillment of stopping criteria.

- **Yes: Stopping Criteria Met**

When the **Maximum Number of Iterations** criterion is met, the process is stopped without having reached convergence.

- **No: Stopping Criteria Not Met**

If the stopping criteria have not been met, MOGA is run again to generate a new population (return to Step 2).

6. Conclusion

Steps 2 through 5 are repeated in sequence until the optimization has converged or the stopping criteria have been met. When either of these things occurs, the optimization concludes.

MOGA Steps to Generate a New Population

The process MOGA uses to generate a new population has two main steps: **Cross-over** and **Mutation**.

1. Cross-over

Cross-over combines (mates) two chromosomes (parents) to produce a new chromosome (offspring). The idea behind cross-over is that the new chromosome can be better than both of the parents if it takes the best characteristics from each of the parents. Cross-over occurs during evolution according to a user-definable cross-over probability.

- **Cross-over for Continuous Parameters**

A cross-over operator that linearly combines two parent chromosome vectors to produce two new offspring according to the following equations:

$$\text{Offspring1} = a * \text{Parent1} + (1 - a) * \text{Parent2}$$

$$\text{Offspring2} = (1 - a) * \text{Parent1} + a * \text{Parent2}$$

Consider the following two parents (each consisting of four floating genes), which have been selected for cross-over:

$$\text{Parent 1: } (0.3)(1.4)(0.2)(7.4)$$

$$\text{Parent 2: } (0.5)(4.5)(0.1)(5.6)$$

If $\mathbf{a} = \mathbf{0.7}$, the following two offspring would be produced:

Offspring1: (0.36)(2.33)(0.17)(6.86)

Offspring2: (0.402)(2.981)(0.149)(6.842)

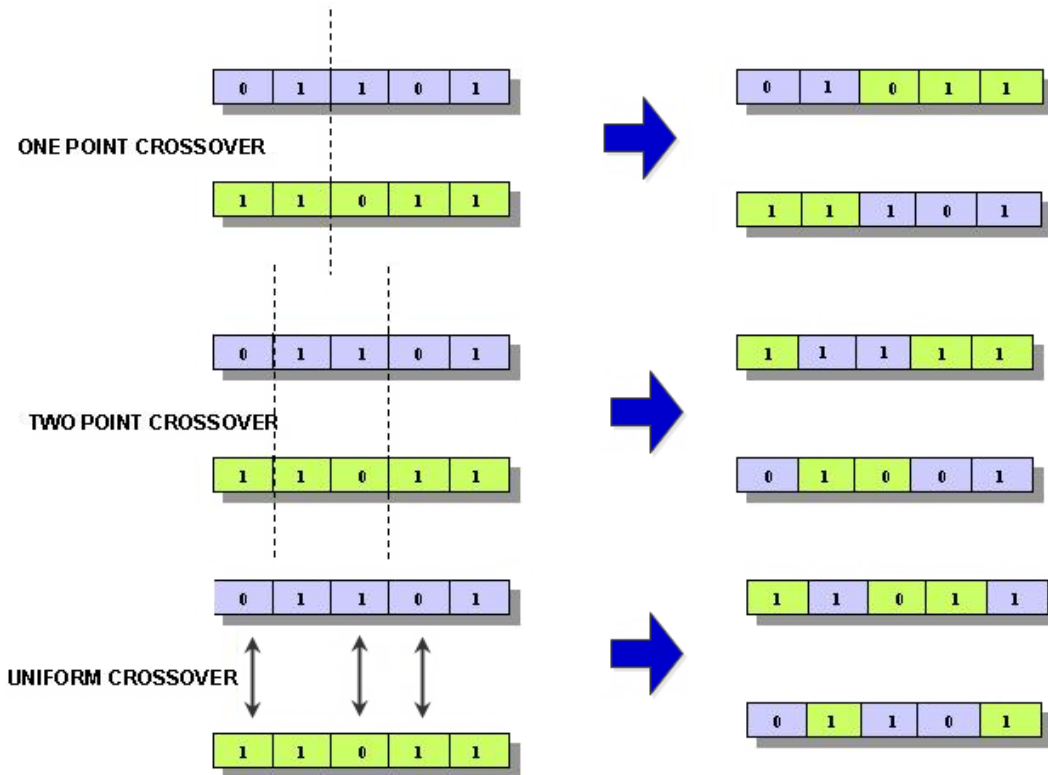
- **Cross-over for Discrete Parameters and Continuous Parameters with Manufacturable Values**

Each discrete parameter or continuous parameter with manufacturable values is represented by a binary chain corresponding to the number of levels. For example, a parameter with two values (levels) is encoded to one bit, a parameter with seven values is encoded to three bits, and an n -bits chain represents a parameter with $2(n-1)$ values.

The concatenation of these chains forms the chromosome, which crosses over with another chromosome.

Three different kinds of cross-over are available:

- **One-Point:** A one-point cross-over operator that randomly selects a cross-over point within a chromosome then interchanges the two parent chromosomes at this point to produce two new offspring.
- **Two-Point:** A two-point cross-over operator randomly selects two cross-over points within a chromosome then interchanges the two parent chromosomes between these points to produce two new offspring.
- **Uniform:** A uniform cross-over operator decides (with some probability, which is known as the "mixing ratio") which parent contributes each of the gene values in the offspring chromosomes. This allows the parent chromosomes to be mixed at the gene level rather than the segment level (as with one and two-point cross-over). For some problems, this additional flexibility outweighs the disadvantage of destroying building blocks.



2. Mutation

Mutation alters one or more gene values in a chromosome from its initial state. This can result in entirely new gene values being added to the gene pool. With these new gene values, the genetic algorithm might be able to arrive at a better solution than was previously possible. Mutation is an important part of the genetic search, as it helps to prevent the population from stagnating at any local optima. Mutation occurs during evolution according to a user-defined mutation probability.

- **Mutation for Continuous Parameters**

For continuous parameters, a polynomial mutation operator is applied to implement mutation.

$$C = P + (\text{UpperBound} - \text{LowerBound})\delta$$

where C is the child, P is the parent, and δ is a small variation calculated from a polynomial distribution.

- **Mutation for Discrete Parameters and Continuous Parameters with Manufacturable Values**

For discrete parameters or continuous parameters with manufacturable values, a mutation operator simply inverts the value of the chosen gene (0 goes to 1 and 1 goes to 0) with a

probability of 0.5. This mutation operator can only be used for binary genes. The concatenation of these chains forms the chromosome, which crosses over with another chromosome.

Adaptive Multiple-Objective Optimization

Adaptive Multiple-Objective (AMO) is a mathematical optimization that combines a Kriging response surface and MOGA. It allows you to either generate a new sample set or use an existing set, providing a more refined approach than the Screening method. Except when necessary, the optimizer does not evaluate all design points. The general optimization approach is the same as MOGA, but a Kriging response surface is used. Part of the population is "simulated" by evaluations of the Kriging response surface. The Kriging error predictor reduces the number of evaluations used in finding the first Pareto front solutions.

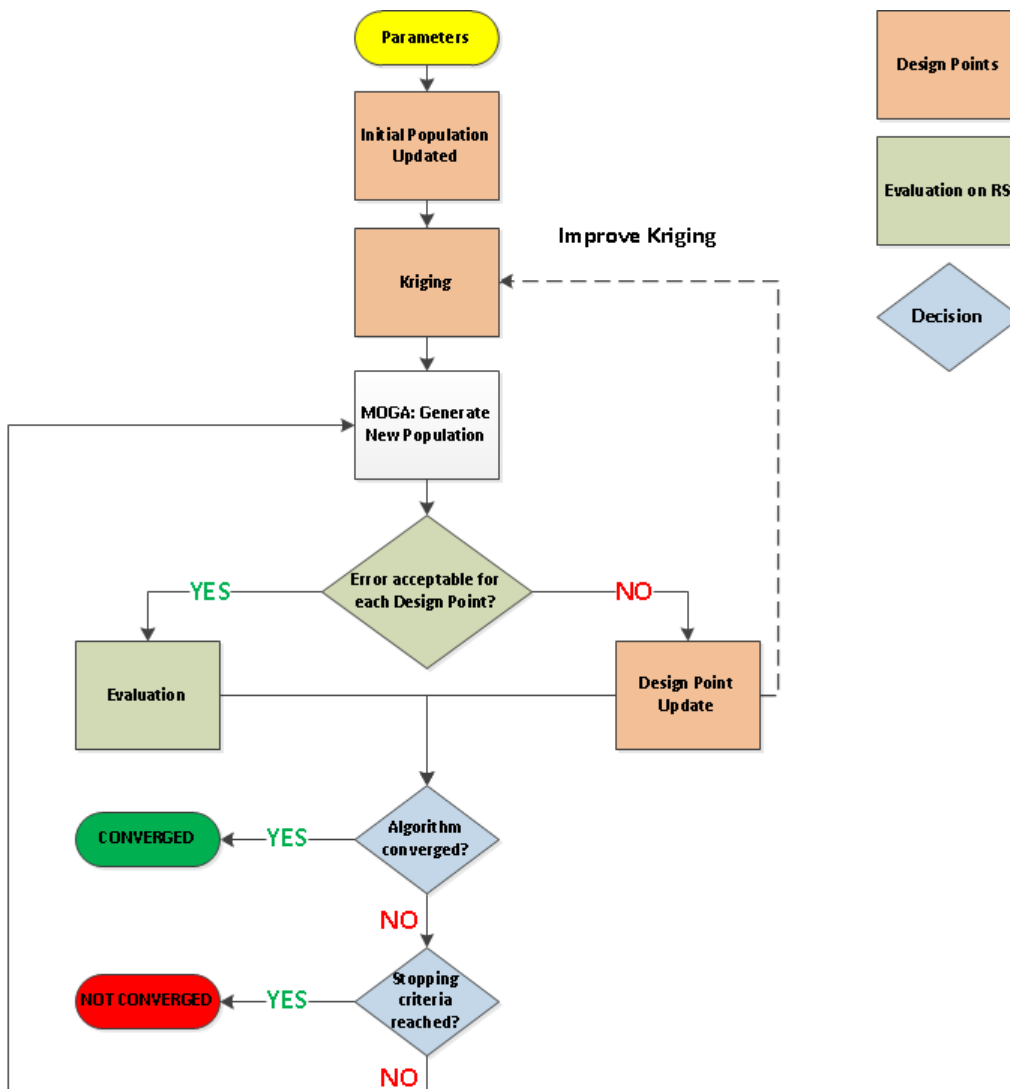
AMO supports multiple objectives and multiple constraints. It is limited to continuous parameters, including those with manufacturable values. It is available only for a **Direct Optimization** system.

Note:

AMO does not support discrete parameters because with discrete parameters, it is necessary to construct a separate response surface for each discrete combination. When discrete parameters are used, MOGA is the more efficient optimization method. For more information, see [Multi-Objective Genetic Algorithm \(MOGA\) \(p. 374\)](#).

AMO Workflow

The workflow of AMO follows:



AMO Steps

1. First Population of MOGA

The initial population of MOGA is used for the constructing Kriging response surfaces.

2. Kriging Generation

A Kriging response surface is created for each output, based on the first population and then improved during simulation with the addition of new design points.

For more information, see [Kriging \(p. 111\)](#) or [Kriging \(p. 340\)](#).

3. MOGA

MOGA is run, using the Kriging response surface as an evaluator. After the first iteration, each population is run when it reaches the number of samples defined by the **Number of Samples Per Iteration** property.

4. Evaluate the Population

5. Error Check

The Kriging error predictor is checked for each point.

- **Yes: Error Acceptable**

Each point is validated for error. If the error for a given point is acceptable, the approximated point is included in the next population to be run through MOGA (return to Step 3).

- **No: Error Not Acceptable**

If the error is not acceptable, the points are promoted as design points. The new design points are used to improve the Kriging response surface (return to Step 2) and are included in the next population to be run through MOGA (return to Step 3).

6. Convergence Validation

The optimization is validated for convergence.

- **Yes: Optimization Converged**

MOGA converges when the maximum allowable Pareto percentage has been reached. When this happens, the process is stopped.

- **No: Optimization Not Converged**

If the optimization is not converged, the process continues to the next step.

7. Stopping Criteria Validation

If the optimization has not converged, it is validated for fulfillment of the stopping criteria.

- **Yes: Stopping Criteria Met**

When the maximum number of iterations has been reached, the process is stopped without having reached convergence.

- **No: Stopping Criteria Not Met**

If the stopping criteria have not been met, the MOGA algorithm is run again (return to Step 3).

8. Conclusion

Steps 2 through 7 are repeated in sequence until the optimization has converged or the stopping criteria have been met. When either of these things occurs, the optimization concludes.

Decision Support Process

The Decision Support Process is a goal-based, weighted, aggregation-based design ranking technique. It is the final step in the optimization, where the optimization is post-processed.

During an optimization, the DesignXplorer optimizer generates a sample set as follows:

- Screening: Sample set corresponds to the number of Screening points plus the Min-Max search results. Search results that duplicate existing points are omitted.
- Single-objective optimization (NLPQL, MISQP, ASO). Sample set corresponds to the iteration points.
- Multiple-objective optimization (MOGA, AMO) Sample set corresponds to the final population.

The Decision Support Process sorts the sample set using the cost function to extract the best candidates. The cost function takes into account both the **Importance** level of objectives and constraints and the feasibility of points. (The feasibility of a point depends on how constraints are handled. When the **Constraint Handling** property is set to **Relaxed**, all infeasible points are included in the sort. When the property is set to **Strict**, all infeasible points are removed from the sort.) Once the sample set has been sorted, you can change the **Importance** level and **Constraint Handling** properties for one or more constraints or objectives without causing DesignXplorer to create more design points. The Decision Support Process sorts the existing sample set again.

Given n input parameters, m output parameters, and their individual targets, the collection of objectives is combined into a single, weighted objective function, Φ , which is sampled by means of a direct Monte Carlo method using uniform distribution. The candidate points are subsequently ranked by ascending magnitudes of the values of Φ . The function for Φ (where all continuous input parameters have usable values of type "Continuous") is given by the following:

$$\Phi \equiv \sum_{i=1}^n w_i N_i + \sum_{j=1}^m w_j M_j \quad (59)$$

where:

w_i and w_j = weights defined in [Equation 62 \(p. 383\)](#)

N_i and M_j = normalized objectives for input and output parameters, respectively

The normalized objectives (metrics) are:

$$N_i = \left(\frac{|x_t - x|}{x_u - x_l} \right)_i \quad (60)$$

$$M_j = \left(\frac{|y_t - y|}{y_{\max} - y_{\min}} \right)_j \quad (61)$$

where:

x = current value for input parameter i

x_t, y_t = corresponding "target value"

y = current value for output parameter j

x_l and x_u = lower and upper values, respectively, for input parameter i

y_{\min} and y_{\max} = corresponding lower and upper bounds, respectively, for output parameter j

The fuzziness of the combined objective function derives from the weights w , which are simply defined as follows:

$$w_i = w_j = \begin{cases} 1.000, & \text{if the Importance is "Higher"} \\ 0.666, & \text{if the Importance is "Default"} \\ 0.333, & \text{if the Importance is "Lower"} \end{cases} \quad (62)$$

The labels used are defined in [Defining Optimization Objectives and Constraints \(p. 205\)](#).

The targets represent the desired values of the parameters, and are defined for the continuous input parameters as follows:

$$x_t = \begin{cases} x, & \text{if Objective Type is "No Objective"} \\ x_l, & \text{if Objective Type is "Minimize"} \\ \frac{1}{2}(x_l + x_u), & \text{if Objective Type is "Seek Target"} \\ x_u, & \text{if Objective Type is "Maximize"} \end{cases} \quad (63)$$

And, for the output parameters, you have the following desired values:

$$y_t = \begin{cases} y, & \text{if Objective Type is "No Objective"} \\ y_{\min}', & \text{if Objective Type is "Minimize"} \\ y_{t2}', & \text{if Constraint Type is "Values } \leq \text{Upper Bound" and "Upper Bound" is defined and } y \geq y_{t2} \\ y, & \text{if Constraint Type is "Values } \leq \text{Upper Bound" and "Upper Bound" is defined and } y \leq y_{t2} \\ y_t^*, & \text{if Objective Type is "Seek Target"} \\ y, & \text{if Constraint Type is "Values } \geq \text{Lower Bound" and "Lower Bound" is defined and } y \geq y_{t1} \\ y_{t1}', & \text{if Constraint Type is "Values } \geq \text{Lower Bound" and "Lower Bound" is defined and } y \leq y_{t1} \\ y_{\max}', & \text{if Objective is "Maximize"} \\ y, & \text{if Constraint Type is "Lower Bound } \leq \text{Values } \leq \text{Upper Bound" and } y_{t1} \leq y \leq y_{t2} \\ y_{t1}', & \text{if Constraint Type is "Lower Bound } \leq \text{Values } \leq \text{Upper Bound" and } y < y_{t1} \\ y_{t2}', & \text{if Constraint Type is "Lower Bound } \leq \text{Values } \leq \text{Upper Bound" and } y > y_{t2} \end{cases} \quad (64)$$

where:

y_t^* = user-specified target

y_{t1} = constraint lower bound

y_{t2} = constraint upper bound

Thus, [Equation 62 \(p. 383\)](#) and [Equation 63 \(p. 383\)](#) constitute the input parameter objectives for the continuous input parameters and [Equation 62 \(p. 383\)](#) and [Equation 64 \(p. 383\)](#) constitute the output parameter objectives and constraints.

The following section considers the case where discrete input parameters and continuous input parameters with manufacturable values are possible. Assume that a continuous input parameter with manufacturable values is defined as follows:

$$\{X\} = \{x_0, x_1, \dots, x_{l-1}\} \quad (65)$$

With one usable value, the following metric is defined:

$$P_k = \frac{|x_t - x|}{|x_{MAX} - x_{MIN}|} \quad (66)$$

where, as before:

x_{max} = upper bound of the usable values

x_{min} = lower bound of the usable values

The target value x_t is given by the following:

$$x_t = \begin{cases} x, & \text{if Constraint Type is "No Constraint"} \\ x_t^*, & \text{if Constraint Type is "Values } \leq \text{Upper Bound" and "Upper Bound" is defined and } x \geq x_t^* \\ x, & \text{if Constraint Type is "Values } \leq \text{Upper Bound" and "Upper Bound" is defined and } x \leq x_t^* \\ x_t^*, & \text{if Objective Type is "Seek Target"} \\ x, & \text{if Constraint Type is "Values } \geq \text{Lower Bound" and "Lower Bound" is defined and } x \geq x_t^* \\ x_t^*, & \text{if Constraint Type is "Values } \geq \text{Lower Bound" and "Lower Bound" is defined and } x \leq x_t^* \end{cases} \quad (67)$$

Thus, the GDO objective equation becomes the following (for parameters with discrete usable values).

$$\Phi \equiv \sum_{i=1}^n w_i N_i + \sum_{j=1}^m w_j M_j + \sum_{l=1}^p w_l P_l \quad (68)$$

Therefore, [Equation 61 \(p. 382\)](#), [Equation 62 \(p. 383\)](#), and [Equation 66 \(p. 384\)](#) constitute the input parameter objectives for parameters that might be continuous or possess discrete usable values.

The norms, objectives, and constraints as in equations [Equation 66 \(p. 384\)](#) and [Equation 67 \(p. 384\)](#) are also adopted to define the input goals for input parameters of the type **Discrete**, which are those parameters whose usable alternatives indicate a whole number of some particular design feature (number of holes in a plate, number of stiffeners, and so on).

Thus, equations [Equation 61 \(p. 382\)](#), [Equation 62 \(p. 383\)](#), and [Equation 66 \(p. 384\)](#) constitute the input parameter goals for discrete parameters.

Therefore, the GDO objective function equation for the most general case (where there are continuous and discrete parameters) can be written as the following:

$$\Phi \equiv \underbrace{\sum_{i=1}^n w_i N_i}_{\text{(Continuous Input)}} + \underbrace{\sum_{j=1}^m w_j M_j}_{\text{(Continuous Output)}} + \underbrace{\sum_{f=1}^l w_f P_f}_{\text{(Manufacturable Values + Discrete)}} \quad (69)$$

where:

n = number of continuous input parameters

m = number of continuous output parameters

l = number of continuous input parameters with manufacturable values and number of discrete input parameters

From the normed values it is obvious that the lower the value of Φ , the better the design with respect to the desired values and importance. Thus, a quasi-random uniform sampling of design points is done by a Hammersley algorithm and the samples are sorted in ascending order of Φ . The desired number of designs are then drawn from the top of the sorted list. A crowding technique is employed to ensure that any two sampled design points are not very close to each other in the space of the input parameters.

Rating Candidate Design Points

Each parameter range is divided into six zones, or rating scales. The location of a design candidate value in the range is measured according to the rating scales. For example, for parameter X with a range of 0.9 to 1.1, the rating scale for a design candidate value of 1.0333 is calculated:

$$\left(\left(\text{Absolute}(1.0333-1.1) \right) / (1.1-0.9) \right) * 6 - (6/2) = 2.01 - 3 = -1 \text{ [one star]} \text{ (as 0 indicating neutral, negative values indicating closer to the target, up to -3. Positive value indicating farther away from the target, up to +3)}$$

Following the same procedures, you get rating scale for design candidate value of 0.9333 as $5.001 - 3 = +2$ [two crosses] (away from target). Therefore, the extreme cases are as follows:

1. Design Candidate value of 0.9 (the worst), the rating scale is $6 - 3 = +3$ [three crosses]
2. Design Candidate value of 1.1 (the best), the rating scale is $0 - 3 = -3$ [three stars]
3. Design Candidate value of 1.0 (neutral), the rating scale is $3 - 3 = 0$ [dash]

Note:

Objective-driven parameter values with inequality constraints receive either three stars (the constraint is met) or three red crosses (the constraint is violated).

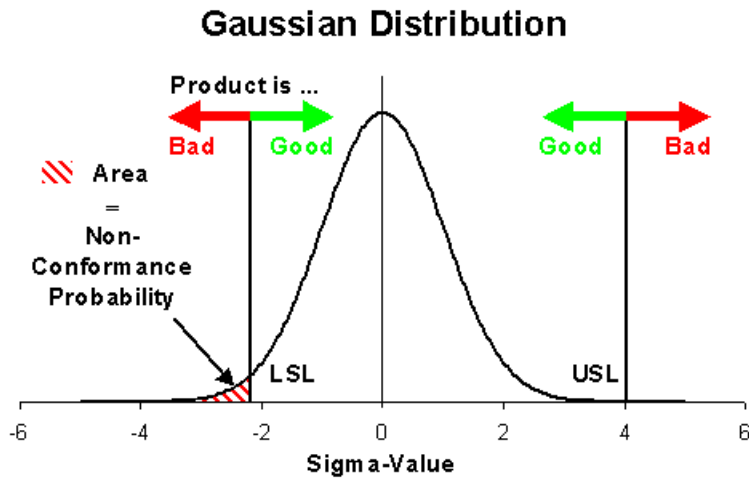
Six Sigma Analysis (SSA) Theory

A Six Sigma Analysis (SSA) allows you to determine the extent to which uncertainties in the model affect the results of an analysis. An uncertainty (or random quantity) is a parameter whose value is impossible to determine at a given point in time (if it is time-dependent) or at a given location (if it is location-dependent). An example is ambient temperature. You cannot know precisely what the temperature will be one week from now in a given city.

SSA uses statistical distribution functions (such as Gaussian, normal, uniform, and so on) to describe uncertain parameters.

SSA allows you to determine whether your product satisfies Six Sigma quality criteria. A product has Six Sigma quality if only 3.4 parts out of every 1 million manufactured fail. This quality definition is based on the assumption that an output parameter relevant to the quality and performance assessment follows a Gaussian distribution, as shown.

An output parameter that characterizes product performance is typically used to determine whether a product's performance is satisfactory. The parameter must fall within the interval bounded by the lower specification limit (LSL) and upper specification limit (USL). Sometimes only one of these limits exists.



An example of this is a case when the maximum von Mises stress in a component must not exceed the yield strength. The relevant output parameter is the maximum von Mises stress and the USL is the yield strength. The lower specification limit is not relevant. The area below the probability density function falling outside the specification interval is a direct measure of the probability that the product does not conform to the quality criteria, as shown above. If the output parameter does follow a Gaussian distribution, then the product satisfies a Six Sigma quality criterion if both specification limits are at least six standard deviations away from the mean value.

In reality, an output parameter rarely follows a Gaussian distribution exactly. However, the definition of Six Sigma quality is inherently probabilistic. It represents an admissible probability that parts do not conform to the quality criteria defined by the specified limits. The nonconformance probability can be calculated no matter which distribution the output parameter actually follows. For distributions other than Gaussian, the Six Sigma level is not really six standard deviations away from the mean value, but it does represent a probability of 3.4 parts per million, which is consistent with the definition of Six Sigma quality.

This section describes a **Six Sigma Analysis** system in DesignXplorer and how to use it to perform a SSA.

[SSA Principles](#)

[Guidelines for Selecting SSA Variables](#)

[Sample Generation](#)

[Weighted Latin Hypercube Sampling \(WLHS\)](#)

[Postprocessing SSA Results](#)

[SSA Theory](#)

SSA Principles

Computer models are described with specific numerical and deterministic values. For example, material properties are entered using certain values, and the geometry of the component is assigned a certain length or width. An analysis based on a given set of specific numbers and values is called a deterministic analysis. The accuracy of a deterministic analysis depends upon the assumptions and input values used for the analysis.

While scatter and uncertainty naturally occur in every aspect of an analysis, deterministic analyses do not take them into account. To deal with uncertainties and scatter, use SSA to answer the following questions:

- If the input variables of a finite element model are subject to scatter, how large is the scatter of the output parameters? How robust are the output parameters? Here, output parameters can be any parameter that Ansys Workbench can calculate. Examples are the temperature, stress, strain, or deflection at a node, the maximum temperature, stress, strain, or deflection of the model, and so on.
- If the output is subject to scatter due to the variation of the input variables, then what is the probability that a design criterion given for the output parameters is no longer met? How large is the probability that an unexpected and unwanted event, such as failure, takes place?
- Which input variables contribute the most to the scatter of an output parameter and to the failure probability? What are the sensitivities of the output parameter with respect to the input variables?

SSA can be used to determine the effect of one or more variables on the outcome of the analysis. In addition to SSA techniques available, Ansys Workbench offers a set of strategic tools to enhance the efficiency of the SSA process. For example, you can graph the effects of one input parameter versus an output parameter, and you can easily add more samples and additional analysis loops to refine your analysis.

In traditional deterministic analyses, uncertainties are either ignored or accounted for by applying conservative assumptions. You would typically ignore uncertainties if you know for certain that the input parameter has no effect on the behavior of the component under investigation. In this case, only the mean values or some nominal values are used in the analysis. However, in some situations, the influences of uncertainties exist but are still neglected, as for the thermal expansion coefficient, for which the scatter is usually ignored.

Example 7: Accounting for Uncertainties

If you are performing a thermal analysis and want to evaluate the thermal stresses, the equation is:

$$\sigma_{therm} = E\alpha\Delta T$$

because the thermal stresses are directly proportional to the Young's modulus as well as to the thermal expansion coefficient of the material.

The following table shows the probability that the thermal stresses will be higher than expected, taking uncertainty variables into account.

Uncertainty variables taken into account	Probability that the thermal stresses are more than 5% higher than expected	Probability that the thermal stresses are more than 10% higher than expected
Young's modulus (Gaussian distribution with 5% standard deviation)	~16%	~2.3%
Young's modulus and thermal expansion coefficient (each with Gaussian distribution with 5% standard deviation)	~22%	~8%

Reliability, Quality, and Safety Issues

Use SSA when issues of reliability, quality, and safety are paramount.

Reliability is typically a concern when product or component failures have significant financial consequences (costs of repair, replacement, warranty, or penalties) or worse, can result in injury or loss of life.

If you use a conservative assumption, the difference in thermal stresses shown above tells you that uncertainty or randomness is involved. Conservative assumptions are usually expressed in terms of safety factors. Sometimes regulatory bodies demand safety factors in certain procedural codes. If you are not faced with such restrictions or demands, then using conservative assumptions and safety factors can lead to inefficient and costly over-design. By using SSA methods, you can avoid over-design while still ensuring the safety of the component.

SSA methods even enable you to quantify the safety of the component by providing a probability that the component will survive operating conditions. Quantifying a goal is the necessary first step toward achieving it.

Guidelines for Selecting SSA Variables

When choosing and defining uncertainty variables, you should:

- Specify a reasonable range of values for each uncertainty variable.
- Set reasonable limits on the variability for each uncertainty variable.

More information about choosing and defining uncertainty variables can be found in the following sections.

[Uncertainty Variables for Response Surface Analyses](#)

Uncertainty Variables for Response Surface Analyses

The number of simulation loops that are required for a response surface analysis depends on the number of uncertainty variables. Therefore, you want to select the most important input variables, the ones you know have a significant effect on the result parameters. If you are unsure which uncertainty variables are important, include all of the random variables you can think of and then perform a Monte Carlo analysis. After you learn which uncertainty variables are important and should be included in your response surface analysis, you can eliminate those that are unnecessary.

Choosing a Distribution for a Random Variable

The type and source of the data you have determines which distribution functions can be used or are best suited to your needs.

[Measured Data](#)

[Mean Values, Standard Deviation, Exceedance Values](#)

[No Data](#)

Measured Data

If you have measured data, then you must first know how reliable that data is. Data scatter is not just an inherent physical effect but also includes inaccuracy in the measurement itself. You must consider that the person taking the measurement might have applied a "tuning" to the data. For example, if the data measured represents a load, the person measuring the load could have rounded the measurement values. This means that the data you receive is not truly the measured values. The amount of this tuning could provide a deterministic bias in the data that you need to address separately. If possible, you should discuss any bias that might have been built into the data with the person who provided the data to you.

If you are confident about the quality of the data, then how you proceed depends on how much data you have. In a single production field, the amount of data is typically sparse. If you have only a small amount of data, use it only to evaluate a rough figure for the mean value and the standard deviation. In these cases, you could model the uncertainty variable as a Gaussian distribution if the physical effect you model has no lower and upper limit, or use the data and estimate the minimum and maximum limit for a uniform distribution.

In a mass production field, you probably have a lot of data. In these cases you could use a commercial statistical package that allows you to actually fit a statistical distribution function that best describes the scatter of the data.

Mean Values, Standard Deviation, Exceedance Values

The mean value and the standard deviation are most commonly used to describe the scatter of data. Frequently, information about a physical quantity is given as a value such as 100 ± 5.5 . Often, this form means that the value 100 is the mean value and 5.5 is the standard deviation. Data in this form implies a Gaussian distribution, but you must verify this (a mean value and standard deviation can be provided for any collection of data regardless of the true distribution type). If you have more information, for example, you know that the data is lognormal distributed, then SSA allows you to use the mean value and standard deviation for a lognormal distribution.

Sometimes the scatter of data is also specified by a mean value and an exceedance confidence limit. The yield strength of a material is sometimes given in this way. For example, a 99% exceedance limit based on a 95% confidence level is provided. This means that, from the measured data, you can be sure by 95% that in 99% of all cases the property values exceed the specified limit and only in 1% of all cases do they drop below the specified limit. The supplier of this information is using the mean value, the standard deviation, and the number of samples of the measured data to derive this kind of information. If the scatter of the data is provided in this way, the best way to pursue this further is to ask for more details from the data supplier. Because the given exceedance limit is based on the measured data and its statistical assessment, the supplier might be able to provide you with the details that were used.

If the data supplier does not give you any further information, then you could consider assuming that the number of measured samples was large. If the given exceedance limit is denoted with $x_{1-\alpha/2}$ and the given mean value is denoted with x_μ , the standard deviation can be derived from the equation:

$$\sigma = \frac{|x_{1-\alpha/2} - x_\mu|}{C}$$

where the values for the coefficient C are:

Exceedance Probability	C
99.5%	2.5758
99.0%	2.3263
97.5%	1.9600
95.0%	1.6449
90.0%	1.2816

No Data

In situations where no information is available, there is never just one right answer. Following are hints about which physical quantities are usually described in terms of which distribution functions. This information might help you with the particular physical quantity that you have in mind. Additionally, a list follows of which distribution functions are usually used for which kind of phenomena. You might need to choose from multiple options.

Geometric Tolerances

- If you are designing a prototype, you could assume that the actual dimensions of the manufactured parts would be somewhere within the manufacturing tolerances. In this case it is reasonable to use a uniform distribution, where the tolerance bounds provide the lower and upper limits of the distribution function.
- If the manufacturing process generates a part that is outside the tolerance band, one of two things can happen: the part must either be fixed (reworked) or scrapped. These two cases are usually on opposite ends of the tolerance band. An example of this is drilling a hole. If the hole is outside the tolerance band, but it is too small, the hole can just be drilled larger (reworked). If, however, the hole is larger than the tolerance band, then the problem is either expensive or impossible to fix. In such a situation, the parameters of the manufacturing process are typically tuned to hit the tolerance band closer to the rework side, steering clear of the side where parts need to be scrapped. In this case, a Beta distribution is more appropriate.
- Often a Gaussian (normal) distribution is used. The fact that this distribution has no bounds (it spans minus infinity to infinity), is theoretically a severe violation of the fact that geometrical extensions are described by finite positive numbers only. However, in practice, this lack of bounds is irrelevant if the standard deviation is very small compared to the value of the geometric extension, which is typically true for geometric tolerances.

Material Data

- Very often the scatter of material data is described by a Gaussian distribution.
- In some cases the material strength of a part is governed by the weakest-link theory. This theory assumes that the entire part fails whenever its weakest spot fails. For material properties where the weakest-link assumptions are valid, the Weibull distribution might be applicable.
- For some cases, it is acceptable to use the scatter information from a similar material type. For example, if you know that a material type very similar to the one you are using has a certain material property with a Gaussian distribution and a standard deviation of $\pm 5\%$ around

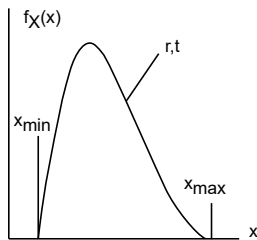
the measured mean value, then you can assume that for the material type you are using, you only know its mean value. In this case, you could consider using a Gaussian distribution with a standard deviation of $\pm 5\%$ around the given mean value.

Load Data

For loads, you usually only have a nominal or average value. You could ask the person who provided the nominal value the following questions: Out of 1000 components operated under real life conditions, what is the lowest load value any one of the components sees? What is the most likely load value? That is, what is the value that most of these 1000 components are subject to? What is the highest load value any one component would be subject to? To be safe, you should ask these questions not only of the person who provided the nominal value but also of one or more experts who are familiar with how your products are operated under real-life conditions. From all the answers you get, you can then consolidate what the minimum, the most likely, and the maximum value probably is. As verification, compare this picture with the nominal value that you would use for a deterministic analysis. The nominal value should be close to the most likely value unless using a conservative assumption. If the nominal value includes a conservative assumption (is biased), its value is probably close to the maximum value. Finally, you can use a triangular distribution using the minimum, most likely, and maximum values obtained.

Distribution Functions

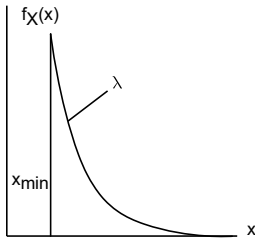
Beta Distribution



You provide the shape parameters r and t and the distribution lower bound and upper bound x_{\min} and x_{\max} of the random variable x .

The Beta distribution is very useful for random variables that are bounded at both sides. If linear operations are applied to random variables that are all subjected to a uniform distribution, then the results can usually be described by a Beta distribution. For example, if you are dealing with tolerances and assemblies where the components are assembled and the individual tolerances of the components follow a uniform distribution (a special case of the Beta distribution), the overall tolerances of the assembly are a function of adding or subtracting the geometrical extension of the individual components (a linear operation). Hence, the overall tolerances of the assembly can be described by a Beta distribution. Also, as previously mentioned, the Beta distribution can be useful for describing the scatter of individual geometrical extensions of components as well.

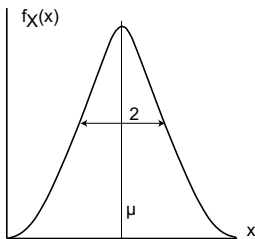
Exponential Distribution



You provide the decay parameter λ and the shift (or distribution lower bound) x_{\min} of the random variable x .

The exponential distribution is useful in cases where there is a physical reason that the probability density function is strictly decreasing as the uncertainty variable value increases. The distribution is mostly used to describe time-related effects. For example, it describes the time between independent events occurring at a constant rate. It is therefore very popular in the area of systems reliability and lifetime-related systems reliability, and it can be used for the life distribution of non-redundant systems. Typically, it is used if the lifetime is not subjected to wear-out and the failure rate is constant with time. Wear-out is usually a dominant life-limiting factor for mechanical components that would preclude the use of the exponential distribution for mechanical parts. However, where preventive maintenance exchanges parts before wear-out can occur, then the exponential distribution is still useful to describe the distribution of the time until exchanging the part is necessary.

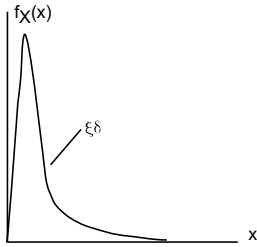
Gaussian (Normal) Distribution



You provide values for the mean value μ and the standard deviation σ of the random variable x .

The Gaussian, or normal, distribution is a fundamental and commonly-used distribution for statistical matters. It is typically used to describe the scatter of the measurement data of many physical phenomena. Strictly speaking, every random variable follows a normal distribution if it is generated by a linear combination of a very large number of other random effects, regardless which distribution these random effects originally follow. The Gaussian distribution is also valid if the random variable is a linear combination of two or more other effects if those effects also follow a Gaussian distribution.

Lognormal Distribution

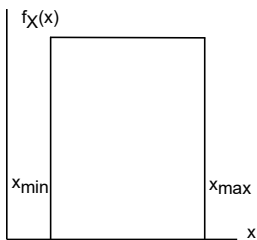


You provide values for the logarithmic mean value ξ and the logarithmic deviation δ . The parameters ξ and δ are the mean value and standard deviation of $\ln(x)$:

$$f(x, \xi, \delta) = \frac{1}{\sqrt{2\pi} \cdot x \cdot \sigma} \cdot \exp\left(-\frac{1}{2} \left(\frac{\ln x - \xi}{\delta}\right)^2\right)$$

The lognormal distribution is another basic and commonly-used distribution, typically used to describe the scatter of the measurement data of physical phenomena, where the logarithm of the data would follow a normal distribution. The lognormal distribution is suitable for phenomena that arise from the multiplication of a large number of error effects. It is also used for random variables that are the result of multiplying two or more random effects (if the effects that get multiplied are also lognormally distributed). It is often used for lifetime distributions such as the scatter of the strain amplitude of a cyclic loading that a material can endure until low-cycle-fatigue occurs.

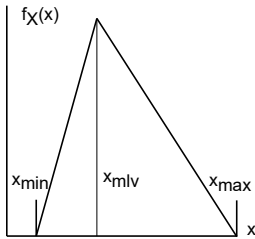
Uniform Distribution



You provide the distribution lower bound and upper bound x_{\min} and x_{\max} of the random variable x .

The uniform distribution is a fundamental distribution for cases where the only information available is a lower and an upper bound. It is also useful to describe geometric tolerances. It can also be used in cases where any value of the random variable is as likely as any other within a certain interval. In this sense, it can be used for cases where "lack of engineering knowledge" plays a role.

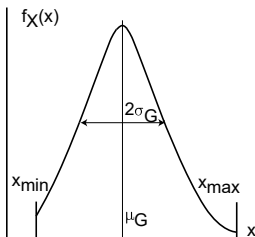
Triangular Distribution



You provide the minimum value (or distribution lower bound) x_{\min} , the most likely value limit x_{mlv} and the maximum value (or distribution upper bound) x_{\max} .

The triangular distribution is most helpful to model a random variable when actual data is not available. It is very often used to capture expert opinions, as in cases where the only data you have are the well-founded opinions of experts. However, regardless of the physical nature of the random variable you want to model, you can always ask experts questions like "Out of 1000 components, what are the lowest and highest load values for this random variable?" and other similar questions. You should also include an estimate for the random variable value derived from a computer program, as described above. For more details, see [Choosing a Distribution for a Random Variable \(p. 388\)](#).

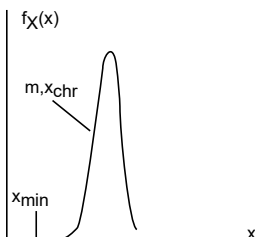
Truncated Gaussian (Normal) Distribution



You provide the mean value μ and the standard deviation σ of the non-truncated Gaussian distribution and the truncation limits x_{\min} and x_{\max} (or distribution lower bound and upper bound).

The truncated Gaussian distribution typically appears where the physical phenomenon follows a Gaussian distribution, but the extreme ends are cut off or are eliminated from the sample population by quality control measures. As such, it is useful to describe the material properties or geometric tolerances.

Weibull Distribution



You provide the Weibull characteristic value x_{chr} , the Weibull exponent m , and the minimum value x_{\min} (or distribution lower bound). There are several special cases. For $x_{\min}=0$ the distribu-

tion coincides with a two-parameter Weibull distribution. The Rayleigh distribution is a special case of the Weibull distribution with $\alpha=x_{chr}-x_{min}$ and $m=2$.

In engineering, the Weibull distribution is most often used for strength or strength-related lifetime parameters, and is the standard distribution for material strength and lifetime parameters for very brittle materials (for these very brittle materials, the "weakest-link theory" is applicable). For more details, see [Choosing a Distribution for a Random Variable \(p. 388\)](#).

Sample Generation

For SSA, the sample generation is based on the Latin Hypercube Sampling (LHS) technique by default. In the **Properties** view for a **Six Sigma Analysis** cell, **Sampling Type** can be set to either **LHS** or **WLHS** (Weighted Latin Hypercube Sampling).

LHS is a more advanced and efficient form of Monte Carlo analysis methods. With LHS, the points are randomly generated in a square grid across the design space, but no two points share input parameters of the same value. This means that no point shares a row or a column of the grid with any other point. Generally, LHS requires 20% to 40% fewer simulations loops than the Direct Monte Carlo technique to deliver the same results with the same accuracy. However, that number is largely problem-dependent.

Weighted Latin Hypercube Sampling (WLHS)

In some cases, a very small probability of failure (Pf), such as in the order of 10^{-6} , is of interest in engineering design. It would require $100/Pf$ simulations/runs of regular LHS. To reduce the number of runs, WLHS can be used to generate biased/more samples in the region.

In WLHS, the input variables are discretized unevenly/unequally in their design space. The cell (or hypercube, in multiple dimensions) size of probability of occurrence is evaluated according to topology of output/response. The cell size of probability of occurrence is discretized such that it is smaller (relatively) around minimum and maximum of output/response. As evaluation of cell size is output/response oriented, WLHS is somewhat unsymmetrical/biased.

In general, WLHS is intended to stretch distribution farther out in the tails (lower and upper) with less number of runs than LHS. This means that given same number of runs, WLHS is expected to reach a smaller Pf than LHS. Due to biasness, however, the evaluated Pf can be subject to some difference compared to LHS.

Postprocessing SSA Results

The following postprocessing results are available for SSA:

[Histogram](#)

[Cumulative Distribution Function](#)

[Probability Table](#)

[Statistical Sensitivities in a SSA](#)

Histogram

A histogram plot is most commonly used to visualize the scatter of a SSA variable. A histogram is derived by dividing the range between the minimum value and the maximum value into intervals of equal size. Then SSA determines how many samples fall within each interval, that is, how many "hits" landed in each interval.

SSA also allows you to plot histograms of your uncertainty variables so you can double-check that the sampling process generated the samples according to the distribution function you specified. For uncertainty variables, SSA not only plots the histogram bars, but also a curve for values derived from the distribution function you specified. Visualizing histograms of the uncertainty variables is another way to verify that enough simulation loops have been performed. If the number of simulation loops is sufficient, the histogram bars have the following characteristics:

- They are close to the curve that is derived from the distribution function.
- They are smooth, meaning that no large steps are present.
- They have no major gaps. While they have no hits in an interval, neighboring intervals have many hits.

However, if the probability density function is flattening out at the far ends of a distribution (for example, the exponential distribution flattens out for large values of the uncertainty variable) then there might logically be gaps. Hits are counted only as positive integer numbers and as these numbers gradually get smaller, a zero hit can happen in an interval.

Cumulative Distribution Function

The cumulative distribution function is a primary review tool if you want to assess the reliability or the failure probability of your component or product. Reliability is defined as the probability that no failure occurs. Hence, in a mathematical sense, reliability and failure probability are different ways of examining the same problem and numerically they complement each other (they sum to 1.0). The cumulative distribution function value at any given point expresses the probability that the respective parameter value remains below that point.

The value of the cumulative distribution function at the location x_0 is the probability that the values of X stay below x_0 . Whether this probability represents the failure probability or the reliability of your component depends on how you define failure.

For example, if you design a component such that a certain deflection should not exceed a certain admissible limit, then a failure event occurs if the critical deflection exceeds this limit. Thus, for this example, the cumulative distribution function is interpreted as the reliability curve of the component. On the other hand, if you design a component such that the eigenfrequencies are beyond a certain admissible limit, then a failure event occurs if an eigenfrequency drops below this limit. So for this example, the cumulative distribution function is interpreted as the failure probability curve of the component.

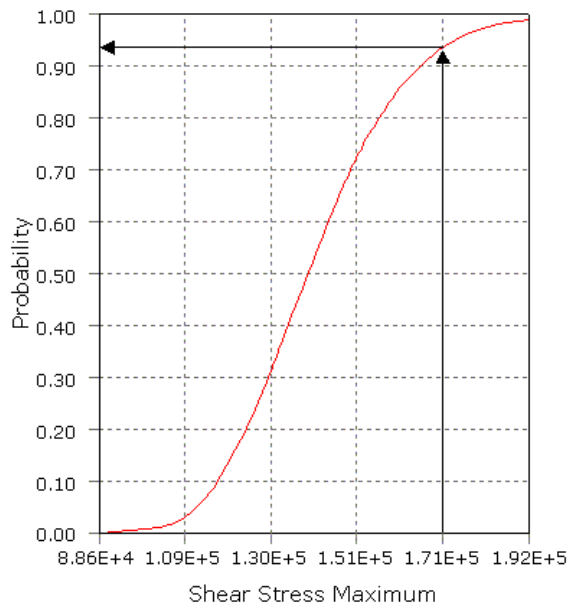
The cumulative distribution function also lets you visualize what the reliability or failure probability would be if you chose to change the admissible limits of your design.

A cumulative distribution function plot is an important tool to quantify the probability that the design of your product does or does not satisfy quality and reliability requirements. The value of

a cumulative distribution function of a particular output parameter represents the probability that the output parameter remains below a certain level as indicated by the values on the X axis of the plot.

Example 8: Illustration of Cumulative Distribution Function

The probability that Shear Stress Maximum remains less than a limit value of $1.71E+5$ is about 93%, which means that there is a 7% probability that Shear Stress Maximum exceeds the limit value of $1.71E+5$.



For more information, see [SSA Theory \(p. 400\)](#).

Probability Table

Instead of reading data from the cumulative distribution chart, you can also obtain important information about the cumulative distribution function in tabular form. A probability table is available that is designed to provide probability values for an even spread of levels of an input or output parameter. You can view the table in either Quantile-Percentile (Probability) mode or Percentile-Quantile (Inverse Probability) mode. The probability table lets you find out the parameter levels corresponding to probability levels that are typically used for the design of reliable products. If you want to see the probability of a value that is not listed, you can add it to the table. Likewise, you can add a probability or sigma-level and see the corresponding values. You can also delete values from the table. For more information, see [Using Statistical Postprocessing \(p. 286\)](#).

	A	B	C
1	P1 - DXLENGTH	Probability	Sigma Level
2	43.97	0.0069075	-2.462
3	44.8	0.022859	-1.998
4	45.631	0.043931	-1.7068
5	46.461	0.077687	-1.4208
6	47.292	0.13922	-1.0838
7	48.122	0.2242	-0.7581
8	48.952	0.33403	-0.42881
9	49.783	0.46503	-0.087763
10	50.613	0.59335	0.23617
11	51.443	0.71737	0.57505
12	52.274	0.81438	0.89414
13	53.104	0.89377	1.2468
14	53.934	0.94418	1.5909
15	54.765	0.96858	1.8604
16	55.595	0.98111	2.0771
17	56.425	0.98845	2.2717
18	57.256	0.99309	2.462
*	New Parameter Value		

	A	B	C
1	Probability	Sigma Level	P1 - DXLENGTH
2	0.0069075	-2.462	43.97
3	0.01	-2.3263	44.229
4	0.02275	-2	44.797
5	0.025	-1.96	44.857
6	0.05	-1.6449	45.854
7	0.1	-1.2816	46.793
8	0.15866	-1	47.487
9	0.3	-0.5244	48.718
10	0.5	0	50.025
11	0.7	0.5244	51.318
12	0.84134	1	52.512
13	0.9	1.2816	53.228
14	0.95	1.6449	54.185
15	0.975	1.96	55.127
16	0.97725	2	55.287
17	0.99	2.3263	56.664
18	0.99309	2.462	57.256
*	New Probability Value	New Sigma Level	

Note:

Both tables have more rows if the number of samples is increased. If you are designing for high product reliability, which is a low probability that the product does not conform to quality or performance requirements, then the sample size must be adequately large to address those low probabilities. Typically, if your product does not conform to the requirements denoted with "Preq," then the minimum number of samples should be determined by $N_{\text{samp}} = 10.0 / \text{Preq}$. For example, if your product has a probability of $\text{Preq} = 1.0 \times 10^{-4}$ that it does not conform to the requirements, then the minimum number of samples should be $N_{\text{samp}} = 10.0 / 1.0 \times 10^{-4} = 10 \times 10^4 = 10^5$.

Statistical Sensitivities in a SSA

The available sensitivities plots allow you to efficiently improve your design toward a more reliable and better quality design, or to save money in the manufacturing process while maintaining the reliability and quality of your product. You can view a sensitivities plot for any output parameter in your model. For more information, see [Sensitivities Chart \(SSA\)](#) (p. 63).

The sensitivities available under the **Six Sigma Analysis** and **Goal Driven Optimization** views are statistical sensitivities. Statistical sensitivities are global sensitivities, whereas the parameter sensitivities available under the **Responses** view are local sensitivities. The global, statistical sensitivities are based on a correlation analysis using the generated sample points, which are located throughout the entire space of input parameters. The local parameter sensitivities are based on the difference between the minimum and maximum value obtained by varying one input parameter while holding all other input parameters constant. As such, the values obtained for local parameter sensitivities depend on the values of the input parameters that are held constant. Global, statistical sensitivities do not depend on the values of the input parameters, because all possible values for the input parameters are already taken into account when determining the sensitivities.

Design exploration displays sensitivities as both a bar chart and pie chart. The charts describe the sensitivities in an absolute fashion (taking the signs into account). A positive sensitivity indicates that increasing the value of the uncertainty variable increases the value of the result parameter for which the sensitivities are plotted. Conversely, a negative sensitivity indicates that increasing the uncertainty variable value reduces the result parameter value.

Using a sensitivity plot, you can answer the following important questions.

How can I make the component more reliable or improve its quality?

If the results for the reliability or failure probability of the component do not reach the expected levels, or if the scatter of an output parameter is too wide and therefore not robust enough for a quality product, then you should make changes to the important input variables first. Modifying an input variable that is insignificant would be a waste of time.

Of course, you are not in control of all uncertainty parameters. A typical example where you have very limited means of control involves material properties. For example, if it turns out that the environmental temperature (outdoor) is the most important input parameter, then there is probably nothing you can do. However, even if you find out that the reliability or quality of your product is driven by parameters that you cannot control, this data has importance—it is likely that you have a fundamental flaw in your product design! You should watch for influential parameters like these.

If the input variable you want to tackle is a geometry-related parameter or a geometric tolerance, then improving the reliability and quality of your product means that it might be necessary to change to a more accurate manufacturing process or use a more accurate manufacturing machine. If it is a material property, then there might be nothing you can do about it. However, if you only had a few measurements for a material property and consequently used only a rough guess about its scatter, and the material property turns out to be an important driver of product reliability and quality, then it makes sense to collect more raw data.

How can I save money without sacrificing the reliability or the quality of the product?

If the results for the reliability or failure probability of the component are acceptable or if the scatter of an output parameter is small and therefore robust enough for a quality product, then there is usually the question of how to save money without reducing the reliability or quality. In this case, you should first make changes to the input variables that turned out to be insignificant because they do not affect the reliability or quality of your product. If it is the geometrical properties or tolerances that are insignificant, you can consider applying a less expensive manufacturing process. If a material property turns out to be insignificant, then this is not typically a good way to save money, because you are usually not in control of individual material properties. However, the loads

or boundary conditions can be a potential for saving money, but in which sense this can be exploited is highly problem-dependent.

SSA Theory

The purpose of a SSA is to gain an understanding of the effects of uncertainties associated with the input parameter of your design. This goal is achieved using a variety of statistical measures and postprocessing tools.

Statistical Postprocessing

Convention: Set of data x_i .

1. Mean Value

Mean is a measure of average for a set of observations. The mean of a set of n observations is defined as follows:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^n y_i \quad (70)$$

2. Standard Deviation

Standard deviation is a measure of dispersion from the mean for a set of observations. The standard deviation of a set of n observations is defined as follows:

$$\hat{\sigma} = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (y_i - \hat{\mu})^2} \quad (71)$$

3. Sigma Level

Sigma level is calculated as the inverse cumulative distribution function of a standard Gaussian distribution at a given percentile. Sigma level is used in conjunction with standard deviation to measure data dispersion from the mean. For example, for a pair of quantile X and sigma level n_α , it means that value X is about n_α standard deviations away from the sample mean.

4. Skewness

Skewness is a measure of degree of asymmetry around the mean for a set of observations. The observations are symmetric if distribution of the observations looks the same to the left and right of the mean. Negative skewness indicates the distribution of the observations being left-skewed. Positive skewness indicates the distribution of the observations being right-skewed. The skewness of a set of n observations is defined as follows:

$$\hat{\gamma} = \frac{n}{(n-1)(n-2)} \sum_{i=1}^n \left(\frac{y_i - \hat{\mu}}{\hat{\sigma}} \right)^3 \quad (72)$$

5. Kurtosis

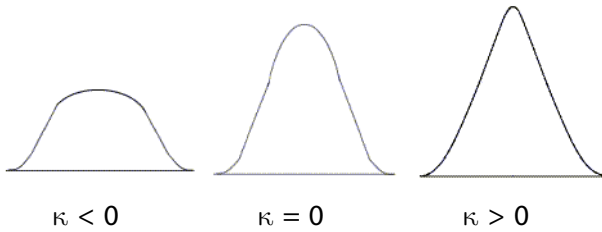
Kurtosis is a measure of relative peakedness/flatness of distribution for a set of observations. It is generally a relative comparison with the normal distribution. Negative kurtosis indicates a relatively flat distribution of the observations compared to the normal distribution, while positive kurtosis

indicates a relatively peaked distribution of the observations. As such, the kurtosis of a set of n observations is defined with calibration to the normal distribution as follows:

$$\hat{\kappa} = \left\{ \frac{n(n+1)}{(n-1)(n-2)(n-3)} \sum_{i=1}^n \left(\frac{y_i - \hat{\mu}}{\hat{\sigma}} \right)^4 \right\} - \frac{3(n-1)^2}{(n-2)(n-3)} \quad (73)$$

where $\hat{\mu}$ and $\hat{\sigma}$ represent mean and standard deviation, respectively.

Table 1: Different Types of Kurtosis



$\kappa < 0$ for flat distribution
 $\kappa = 0$ for normal distribution
 $\kappa > 0$ for peaked distribution

6. Shannon Entropy

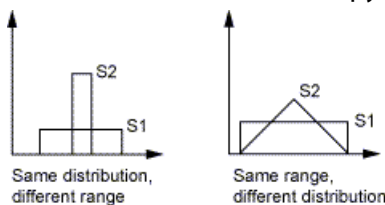
Entropy is a concept first introduced in classical thermodynamics. It is a measure of disorder in a system. The concept of entropy was later extended and recognized by Claude Shannon in the domain of information/communication as a measure of uncertainty over the true content of a message. In statistics, the entropy is further reformulated as a function of mass density, as follows:

$$H = - \sum P_i \ln P_i \quad (74)$$

where P_i represents mass density of a parameter. In the context of statistics/probability, entropy becomes a measure of complexity and predictability of a parameter. A more complex, and less predictable, parameter carries higher entropy, and vice versa. For example, a parameter characterized with uniform distribution has higher entropy than that with truncated triangular distribution (within the same bounds). Hence, the parameter characterized with uniform distribution is less predictable compared to the triangular distribution.

$$P_i = \frac{N_i}{N \cdot \Delta w} \quad (75)$$

The probability mass is used in a normalized fashion, such that not only the shape, but the range of variability, or the distribution is accounted for. This is shown in Equation 75 (p. 401), where N_i/N is the relative frequency of the parameter falling into a certain interval, and Δw is the width of the interval. As a result, Shannon entropy can have a negative value. Following are some comparisons of the Shannon entropy, where S2 is smaller than S1.



7. Taguchi Signal-to-Noise Ratios

Three signal-to-noise ratios are provided in the statistics of each output in your SSA. These ratios are as follows:

- Nominal is Best
- Smaller is Better
- Larger is Better

Signal-to-noise (S/N) ratios are measures used to optimize control parameters to achieve a robust design. These measures were first proposed by Dr. Taguchi of Nippon Telephone and Telegraph Company, Japan, to reduce design noises in manufacturing processes. These design noises are normally expressed in statistical terms such as mean and standard deviation (or variance). In computer aided engineering (CAE), these ratios have been widely used to achieve a robust design in computer simulations. For a design to be robust, the simulations are carried out with an objective to minimize the variances. The minimum variance of designs/simulations can be done with or without targeting a certain mean. In design exploration, minimum variance targeted at a certain mean (called Nominal is Best) is provided, and is given as follows:

$$\eta = 20 \log \frac{\hat{\mu}}{\hat{\sigma}} \quad (76)$$

where $\hat{\mu}$ and $\hat{\sigma}$ represent mean and standard deviation, respectively.

Nominal is Best is a measure used for characterizing design parameters such as model dimension in a tolerance design, in which a specific dimension is required, with an acceptable standard deviation.

In some designs, however, the objectives are to seek a minimum or a maximum possible at the price of any variance.

- For the cases of minimum possible (Smaller is Better), which is a measure used for characterizing output parameters such as model deformation, the S/N is expressed as follows:

$$\eta = -10 \log \left[\frac{1}{n} \sum_{i=1}^n y_i^2 \right] \quad (77)$$

- For the cases of maximum possible (Larger is Better), which is a measure used for characterizing output parameters such as material yield, the S/N is formulated as follows:

$$\eta = -10 \log \left[\frac{1}{n} \sum_{i=1}^n \frac{1}{y_i^2} \right] \quad (78)$$

These three S/N ratios are mutually exclusive. Only one of the ratios can be optimized for any given parameter. For a better design, these ratios should be maximized.

8. Sigma Minimum and Maximum Values

The minimum and maximum values of a set of n observations are:

$$x_{\min} = \min(x_1, x_2, \dots, x_n) \quad (79)$$

$$x_{\max} = \max(x_1, x_2, \dots, x_n) \quad (80)$$

Note:

The minimum and maximum values strongly depend on the number of samples. If you generate a new sample set with more samples, then chances are that the minimum value is lower in the larger sample set. Likewise, the maximum value of the larger sample set is most likely higher than for the original sample set. Hence, the minimum and maximum values should not be interpreted as absolute physical bounds.

9. Statistical Sensitivity Measures

The sensitivity charts displayed under **Six Sigma Analysis** are global sensitivities based on statistical measures. For more information, see [Single Parameter Sensitivities \(p. 157\)](#).

Generally, the effect of an input parameter on an output parameter is driven by:

- The amount by which the output parameter varies across the variation range of an input parameter.
- The variation range of an input parameter. Typically, the wider the variation range, the larger the effect of the input parameter.

The statistical sensitivities displayed under **Six Sigma Analysis** are based on the Spearman-Rank Order Correlation coefficients that take both those aspects into account at the same time.

Basing sensitivities on correlation coefficients follows the concept that the more strongly an output parameter is correlated with a particular input parameter, the more sensitive it is with respect to changes of that input parameter.

10. Spearman Rank-Order Correlation Coefficient

The Spearman rank-order correlation coefficient is:

$$r_s = \frac{\sum_i^n (R_i - \bar{R})(S_i - \bar{S})}{\sqrt{\sum_i^n (R_i - \bar{R})^2} \sqrt{\sum_i^n (S_i - \bar{S})^2}} \quad (81)$$

where:

R_i = rank of x_i within the set of observations $[x_1 x_2 \dots x_n]^T$

S_i = rank of y_i within the set of observations $[y_1 y_2 \dots y_n]^T$

\bar{R}, \bar{S} = average ranks of a R_i and S_i respectively

11. Significance of Correlation Coefficients

Because the sample size n is finite, the correlation coefficient r_p is a random variable. Hence, the correlation coefficient between two random variables X and Y usually yields a small, but nonzero value, even if X and Y are not correlated at all in reality. In this case, the correlation coefficient would be insignificant. Therefore, you need to find out if a correlation coefficient is significant or not. To determine the significance of the correlation coefficient, assume the hypothesis that the correlation between X and Y is not significant at all, meaning that they are not correlated, and $r_p=0$ (null hypothesis). In this case the variable:

$$t=r_p\sqrt{\frac{n-2}{1-r_p^2}} \quad (82)$$

is approximately distributed like the student's t -distribution with $\nu=n-2$ degrees of freedom. The cumulative distribution function student's t -distribution is:

$$A(t|\nu)=\frac{1}{\sqrt{\nu} B\left(\frac{1}{2}, \frac{\nu}{2}\right)} \int_{-t}^t \left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}} dx \quad (83)$$

where:

$B(\dots)$ = complete Beta function

There is no closed-form solution available for [Equation 83 \(p. 404\)](#). See Abramowitz and Stegun (*Pocketbook of Mathematical Functions*, abridged version of the *Handbook of Mathematical Functions*, Harry Deutsch, 1984) for more details.

The larger the correlation coefficient r_p , the less likely it is that the null hypothesis is true. Also, the larger the correlation coefficient r_p , the larger is the value of t from [Equation 82 \(p. 404\)](#) and consequently also the probability $A(t|\nu)$ is increased. Therefore, if the probability that the null hypothesis is true is given by $1-A(t|\nu)$. If $1-A(t|\nu)$ exceeds a certain significance level, for example 2.5%, you can assume that the null hypothesis is true. However, if $1-A(t|\nu)$ is below the significance level, then it can be assumed that the null hypothesis is not true and that consequently the correlation coefficient r_p is significant. This limit can be changed in [Design Exploration Options \(p. 34\)](#).

12. Cumulative Distribution Function

The cumulative distribution function of sampled data is also called the empirical distribution function. To determine the cumulative distribution function of sampled data, you need to order the sample values in ascending order. Let x_i be the sampled value of the random variable X having a rank of i , which makes i -th smallest out of all n sampled values. The cumulative distribution function F_i that corresponds to x_i is the probability that the random variable X has values below or equal to x_i . Because you have only a limited amount of samples, the estimate for this probability is itself a random variable. According to Kececioglu (*Reliability Engineering Handbook*, Vol. 1, 1991, Prentice-Hall, Inc.), the cumulative distribution function F_i associated with x_i is:

$$\sum_{k=i}^n \frac{n!}{(n-k)!k!} F_i^k (1-F_i)^{n-k} = 50\% \quad (84)$$

[Equation 84 \(p. 404\)](#) must be solved numerically.

13. Probability and Inverse Probability

The cumulative distribution function of sampled data can only be given at the individual sampled values $x_1, x_2, \dots, x_i, x_{i+1}, \dots, x_n$ using [Equation 84 \(p. 404\)](#). Hence, the evaluation of the probability that the random variable is less than or equal to an arbitrary value x requires an interpolation between the available data points.

If x is, for example, is between x_i and x_{i+1} , the probability that the random variable X is less or equal to x is:

$$P(X \leq x) = F_i + (F_{i+1} - F_i) \frac{x - x_i}{x_{i+1} - x_i} \quad (85)$$

The cumulative distribution function of sampled data can only be given at the individual sampled values $x_1, x_2, \dots, x_i, x_{i+1}, \dots, x_n$ using [Equation 84 \(p. 404\)](#). Hence, the evaluation of the inverse cumulative distribution function for any arbitrary probability value requires an interpolation between the available data points.

The evaluation of the inverse of the empirical distribution function is most important in the tails of the distribution, where the slope of the empirical distribution function is flat. In this case, a direct interpolation between the points of the empirical distribution function similar to [Equation 85 \(p. 405\)](#) can lead to inaccurate results. Therefore, the inverse standard normal distribution function Φ^{-1} is applied for all probabilities involved in the interpolation. If p is the requested probability for which you are looking for the inverse cumulative distribution function value, and p is between F_i and F_{i+1} , the inverse cumulative distribution function value can be calculated using:

$$x = x_i + (x_{i+1} - x_i) \frac{\Phi^{-1}(p) - \Phi^{-1}(F_i)}{\Phi^{-1}(F_{i+1}) - \Phi^{-1}(F_i)} \quad (86)$$

Theory References

This section provides external references for further reading and study on the theory underpinning DesignXplorer's design exploration capabilities:

[Genetic Aggregation Response Surface References](#)

[MISQP Optimization Algorithm References](#)

[Reference Books](#)

[Reference Articles](#)

Note:

These references are not available through Ansys.

Genetic Aggregation Response Surface References

GA reference 1

Ben Salem, M., & Tomaso, L., "Automatic selection for general surrogate models, *Structural and Multidisciplinary Optimization*, Vol. 58, No. 2, 2018, pp. 719-734.

GA reference 2

Ben Salem, M., Roustant, O., Gamboa, F., & Tomaso, L., "Universal prediction distribution for surrogate models," *SIAM/ASA Journal on Uncertainty Quantification*, Vol. 5, No. 1, 2017, pp. 1086-1109.

GA reference 3

Acar, E., "Various Approaches for Constructing an Ensemble of Metamodels Using Local Measures," *Structural and Multidisciplinary Optimization*, Vol. 42, No. 6, 2010, pp. 879-896.

GA reference 4

Viana, F.A.C., Haftka, R.T., and Steffen V., "Multiple Surrogates: How Cross-Validation Errors Can Help Us to Obtain the Best Predictor," *Structural and Multidisciplinary Optimization*, Vol. 39, No. 4, 2009, pp. 439-457.

MISQP Optimization Algorithm References

A list follows of publications relevant to the MISQP optimization method:

MISQP reference 1

O. Exler, K. Schittkowski, T. Lehmann, "A comparative study of numerical algorithms for nonlinear and nonconvex mixed-integer optimization," *Mathematical Programming Computation*, Vol. 1, 2012, pp.383-412.

MISQP reference 2

O. Exler, T. Lehmann, K. Schittkowski, *MISQP: A Fortran Subroutine of a Trust Region SQP Algorithm for Mixed-Integer Nonlinear Programming - User's Guide Report*, Department of Computer Science, University of Bayreuth, 2012.

MISQP reference 3

O. Exler, K. Schittkowski, "A trust region SQP algorithm for mixed-integer nonlinear programming," *Optimization Letters*, Vol. 1, 2007, p.269-280.

Reference Books

A list follows of reference books—from primer to advanced users.

***Probabilistic Structural Mechanics Handbook – Theory and Industrial Applications* Edited by Raj Sundararajan**

This book provides a very good introduction into probabilistic analysis and design of all kinds of industrial applications. The coverage is quite balanced and includes the theory of probabilistic methods, structural reliability methods, and industrial application examples. For more information on probabilistic methods (sampling-based), see "[Simulation and the Monte Carlo Method](#)" (p. 407) by Reuven Rubinstein. For more information on structural reliability methods (such as FORM/SORM), see "[Probability, Reliability, and Statistical Methods in Engineering Design](#)" (p. 407) by Achintya Haldar and Sankaran Mahadevan.

Simulation and the Monte Carlo Method by Reuven Rubinstein

This book provides detailed descriptions of all kinds of sampling methods, including Monte Carlo (brute force), Variance Reduction Monte Carlo (of which Latin Hypercube Sampling is one), Importance Sampling, Markov Chain Monte Carlo (widely known as MCMC), and many more.

Probability, Reliability, and Statistical Methods in Engineering Design by Achintya Haldar and Sankaran Mahadevan

This book presents a good introduction on risk assessments and structural reliability designs. It clearly addresses the methods widely used to conduct risk-based and reliability-based designs, such as FORM/SORM.

Applied Linear Statistical Models by John Neter, Michael Kutner, William Wasserman and Christopher Nachtsheim

This book is basically the backbone of the 2nd-order polynomials response surface in DesignXplorer. It comprehensively covers regression analysis (stepwise regression in particular), input-output transformation for better regression, and statistical goodness-of-fit measures. It also covers sampling schemes for DOEs. For more on DOEs, CCD, and Box-Behnken design, see *Design and Analysis of Experiment* by Douglas Montgomery. CCD and Box-Behnken design are traditional DOE sampling schemes. They differ from *Design and Analysis of Computer Experiments (DACE)*, which is used for an interpolation-based response surface, such as Kriging.

Design and Analysis of Experiments by Douglas Montgomery

This book is a good resource on sampling schemes for a least-square fit based response surface. It provides a detailed explanation on establishing fraction factorial design with a certain resolution.

Design and Analysis of Computer Experiments by Thomas Santner, Brian Williams and William Notz

This book describes how to create a space filling design for an interpolation-based response surface. This includes non-parametric regression and Kriging/Radial Basis, which are supported by DesignXplorer, and Pursuit Projection Regression, which is not supported by DesignXplorer.

Reference Articles

A list follows of reference articles on design exploration topics such as probability, reliability, and statistics.

Articles reference 1

Ang, A. H-S., and Tang, W. H.

Probability Concepts in Engineering Planning and Design; Volume II: Decision, Risk, and Reliability

John Wiley, 1984.

Articles reference 2

Ayyub, B. M. (editor)

Uncertainty Modeling and Analysis in Civil Engineering

CRC Press, 1997, 528 pp.

Articles reference 3

Ayyub, B. M., and McCuen, R. H.

Probability, Statistics, and Reliability for Engineers

CRC Press, Boca Raton, 1997, 514 p.

Articles reference 4

Ayyub, B. M., and McCuen, R. H.

Solution Manual for Probability, Statistics, and Reliability for Engineers

1997.

Articles reference 5

Barlow, R. E., Fussell, J. B., and Singpurwalla, N. D., (editors)

Reliability and Fault Tree Analysis

SIAM, 1975.

Articles reference 6

Barnes, J.W.

Statistical Analysis for Engineers and Scientists

McGraw-Hill, New York, 1994. ISBN 0-07-839608-5.

Articles reference 7

Beasley, M.

Reliability for Engineers: An Introduction

Macmillan, London, 1981.

Articles reference 8

Benjamin, J. R., and Cornell, C. A.

Probability, Statistics, and Decision for Civil Engineers

Prentice Hall, New York, 1970.

Articles reference 9

Billinton, R.

Reliability Evaluation of Engineering Systems: Concepts and Techniques

Plenum Press, New York, 1983 and 1992..

Articles reference 10

Calabro, S. R.

Reliability Principles and Practices

McGraw Hill, New York, 1962.

Articles reference 11

Casciati, F.

Mathematical Models for Structural Reliability Analysis

CRC Press, Boca Raton, July 1996, 384 p.

Articles reference 12

Catuneanu, V. M.

Reliability Fundamentals

Elsevier, New York, 1989.

Articles reference 13

Chorafas, D. N.

Statistical Processes and Reliability Engineering

Van Nostrand, Princeton, 1960.

Articles reference 14

Cox, S. J.

Reliability, Safety, and Risk Management: An Integrated Approach

Butterworth-Heinemann, 1991.

Articles reference 15

Dai, S-H.

Reliability Analysis in Engineering Applications

Van Nostrand Reinhold, New York, June 1992, 448 p.

Articles reference 16

Ditlevson, O.

Structural Reliability Methods

John Wiley & Sons, New York, June 1996.

Articles reference 17

Gumbel, E.

Statistics of Extremes

Columbia University Press, New York, 1958.

Articles reference 18

Haldar, A. and Mahadevan, S.

Probability, Reliability, and Statistical Methods in Engineering Design

John Wiley & Sons, New York 2000.

Articles reference 19

Haldar, A. and Mahadevan, S.

Reliability Assessment Using Stochastic Finite Element Analysis

John Wiley & Sons, New York 2000.

Articles reference 20

Harr, M. E.

Reliability-Based Design in Civil Engineering

McGraw-Hill, New York, 1987.

Articles reference 21

Hart, Gary C.

Uncertainty Analysis of Loads and Safety in Structural Engineering

Prentice Hall, Englewood Cliffs, March 1982, 240 p.

Articles reference 22

Henley, E. J.

Reliability Engineering and Risk Assessment

Prentice-Hall, Englewood Cliffs, 1981.

Articles reference 23

Hoel, P. G., Sidney, C. P., and Stone, C. J.

Introduction to Probability Theory

Houghton Mifflin Company, Boston, 1971.

Articles reference 24

Kapur, K. C.

Reliability in Engineering Design

John Wiley, New York, 1977.

Articles reference 25

Leemis, L. M.

Reliability: Probabilistic Models and Statistical Methods

Prentice Hall, Englewood Cliffs, 1995.

Articles reference 26

Leitch, R. D.

Reliability Analysis for Engineers, an Introduction

Oxford University Press, New York, July 1995, 248 p.

Articles reference 27

Litle, W. A.

Reliability of Shell Buckling Predictions

MIT Press, 1964.

Articles reference 28

Lloyd, M., Lipow, M.

Reliability, Management, Methods, and Mathematics

1982.

Articles reference 29

Lucia, A. C.

Advances in Structural Reliability

Kluwer Academic Publishers, Norwell, March 1987.

Articles reference 30

Lutes, L. D., Sarkani, S.

Stochastic Analysis of Structural and Mechanical Vibrations

Prentice Hall, New Jersey, 1997.

Articles reference 31

Madsen, H. O.

Methods of Structural Safety

Prentice Hall, Englewood Cliffs, October 1985, 336 p.

Articles reference 32

Madsen, H. O., Krenk, S., and Lind, N. C.

Methods of Structural Safety

Prentice Hall, New York, 1986.

Articles reference 33

Marek, P.

Simulation-Based Reliability Assessment for Structural Engineers

CRC Press, Boca Raton, October 1995, 372 p.

Articles reference 34

McCormick, N. J.

Reliability and Risk Analysis: Methods and Nuclear Power Applications

Academic Press, New York, 1981.

Articles reference 35

Melchers, R. E.

Structural Reliability Analysis and Prediction

Prentice Hall, Englewood Cliffs, NJ, 1987.

Articles reference 36

Misra, K. B.

Reliability Analysis and Prediction, A Methodology Oriented Treatment

Elsevier Science, New York, July 1992, 890 p.

Articles reference 37

Modarres, M.

What Every Engineer Should Know about Reliability and Risk Analysis

M. Dekker, New York, 1993.

Articles reference 38

Nowak, A.S., (editor)

Modeling Human Error in Structural Design and Construction

ASCE, 1986.

Articles reference 39

Nowak, A. S., and Collins, K. R.

Reliability of Structures

McGraw-Hill, New York, 2000.

Articles reference 40

Papoulis, A.

Probability, Random Variables, and Stochastic Processes

McGraw-Hill, New York, 1965.

Articles reference 41

Pugsley, A. G.

The Safety of Structures

E. Arnold Publ. Ltd., London, 1966.

Articles reference 42

Rackwitz, R.

Reliability and Optimization of Structural Systems

Chapman and Hall, New York, June 1995, 324 p.

Articles reference 43

Rao, S. S.

Reliability-Based Design

McGraw-Hill, New York, 1992.

Articles reference 44

Rubinstein, R. Y.

Simulation and the Monte Carlo Method

John Wiley, New York, 1981.

Articles reference 45

Schlaifer, R.

Analysis of Decisions Under Uncertainty

McGraw-Hill, New York, 1969.

Articles reference 46

Schneider, J.

Introduction to Safety and Reliability of Structures

International Association of Bridge and Structural Engineering, Structural Engineering Document #5.

Articles reference 47

Shooman, M. L.

Probabilistic Reliability: An Engineering Approach

McGraw-Hill, New York, 1968.

Articles reference 48

Sinha, S. K.

Reliability and Life Testing

New York, Wiley, 1986.

Articles reference 49

Smith, D. J.

Reliability, Maintainability, and Risk: Practical Methods for Engineers

4th edition, Butterworth-Heinemann, Boston, 1993.

Articles reference 50

Spencer, B. F.

Reliability of Randomly Excited Hysteretic Structures

Springer-Verlag, New York, 1986.

Articles reference 51

Thoft-Christensen, P.

Application of Structural Systems Reliability Theory

Springer-Verlag, New York, June 1986, 350 p.

Articles reference 52

Thoft-Christensen, P.

Reliability & Optimization of Structural Systems

Springer-Verlag, New York, December 1987, 474 p.

Articles reference 53

Thoft-Christensen, P.

Reliability Theory and Its Application in Structures and Soil Mechanics

Kluwer Academic Publishers, Norwell, June 1983.

Articles reference 54

Thoft-Christensen, P., and Baker, M. J.

Structural Reliability Theory and Its Applications

Springer-Verlag, July 1982, 267 p.

Articles reference 55

Tichy, M.

Applied Methods of Structural Reliability

Kluwer Academic Publishers, Norwell, July 1993. 416 p.

Chart Interpretation reference 56

Wittmann, F. H.

Structural Reliability and Probabilistic Safety Assessment

Ashgate Publishing Co., Brookfield, January 1987, 498 p.

Troubleshooting

An Ansys DesignXplorer Update operation returns an error message saying that one or several design points have failed to update.

Click **Show Details** in the message dialog box to see the list of design points that failed to update and the associated error messages. Each failed design point is automatically preserved for you at the project level, so you can return to the project and edit the **Parameter Set** bar to see the failed design points and investigate the reason for the failures.

This error means that the project failed to update correctly for the parameter values defined in the listed design points. There are a variety of failure reasons, from a lack of a license to a CAD model generation failure. If you try to update the system again, an update attempts to update only the failed design points, and if the update is successful, the operation completes. If there are persistent failures with a design point, to investigate, copy its values to the current design point, attempt a project update, and edit the cells in the project that are not correctly updated.

When you open a project, one or more DesignXplorer cells are marked with a black X icon in the Project Schematic.

The black X icon next to a DesignXplorer cell indicates a disabled status. This status is given to DesignXplorer cells that fail to load properly because the project has been corrupted by the absence of necessary files (most often the `parameters.dxd` and `parameters.params` files). The **Messages** pane displays a warning message indicating the reason for the failure and lists each of the disabled cells.

When a DesignXplorer cell is disabled, you cannot update, edit, or preview it. Try one of the following methods to address the issue.

Method 1: (recommended) Navigate to the **Files** pane, locate the missing files if possible, and copy them into the correct project folder. The `parameters.dxd` and `parameters.params` files are normally located in the subdirectory `<project name>_files\dpall\global\DX`. With the necessary files restored, the disabled cells should load successfully.

Method 2: Delete all DesignXplorer systems containing disabled cells. Once the project is free of corrupted cells, you can save the project and set up new DesignXplorer systems.

Workbench Interacts with Microsoft Office Excel

When an Excel file is added to a Microsoft Office Excel component in a project, an instance of the Excel application is launched in the background. If you double-click any other Excel workbook in the Windows Explorer window or open any other Excel workbook from the **Recent items** option on the **Start** menu, the Excel application attempts to use the instance of Excel that is already running. As a result, all of the background workbooks are displayed in foreground and any update of the Excel components in Workbench prevents you from modifying any opened Excel workbooks. Therefore, to open a different workbook, do not use the same instance of Excel that Workbench is using.

To open or view a different Excel workbook (other than the ones added to the project):

1. Start a new instance of Excel.
2. From this new instance, select **File** → **Open** and then the workbook.

The workbook opens in the new instance of Excel.

Note:

You can still view the Excel workbooks added to the project by selecting **Open File in Excel** from the right-click context menu.

Fatal error causes project to get stuck in a pending state.

If a fatal error occurs while a cell of DesignXplorer is either in a pending state or resuming from a pending update, the project could be stuck in the pending state even if the pending updates are no longer running.

To recover from this situation, you can abandon the pending updates by selecting **Tools** → **Abandon Pending Updates**.

Incorrect derived parameter definition causes a DesignXplorer cell to be in an Unfulfilled state.

To update a DesignXplorer cell or a project, derived parameters must be correctly defined. If the definition for a derived parameter is invalid, which can occur when the parameter is defined without units or the value of the **Expression** property has an incorrect syntax. DesignXplorer cells remain in an unfulfilled state until the error is fixed.

To verify derived parameter definition, right-click the **Parameter Set** bar and select **Edit** to open the **Parameter Set** tab. In the **Outline** pane, derived parameters that are incorrectly defined have a red cell containing an error message in the **Value** column. In the **Properties** pane for such a derived parameter, the same error message displays in the **Expression** property cell.

Based on the information in the error message, correct derived parameter definitions as follows:

1. In the **Outline** pane, select the parameter.
2. In the **Parameters** pane, enter a valid expression for the **Expression** property.
3. Refresh the project or the unfulfilled DesignXplorer cell to apply the changes.