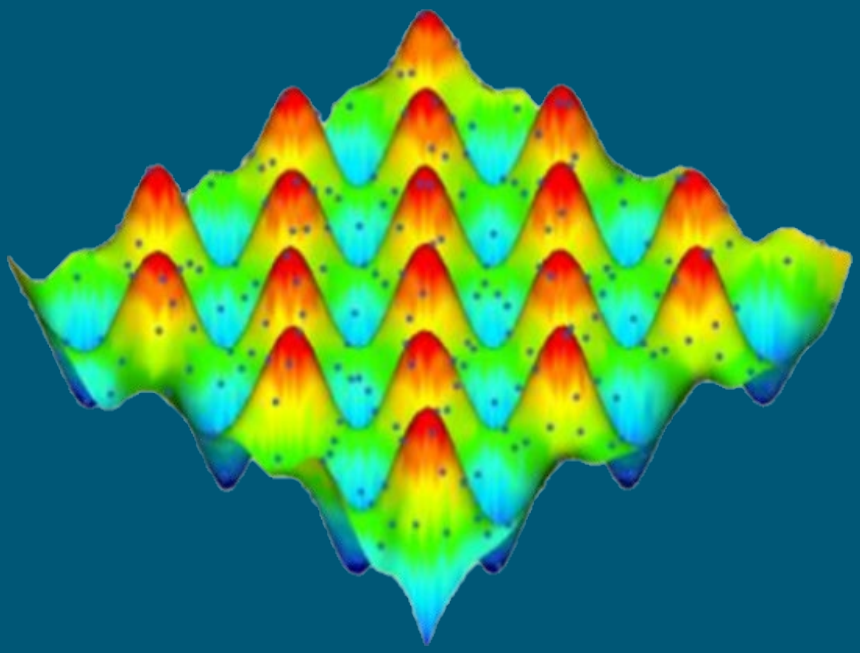
ALTAIR

EBOOK

Introduction into Fit Approximations with Altair HyperStudy™



CONTENTS

1 About This Book

A first eBook on DOE with HyperStudy has been released in the beginning of 2017. We hope you have appreciated it and learned useful knowledge helping you to improve your studies.

This second eBook deals with the Fit approximations' capabilities with HyperStudy. If you are beginner with HyperStudy we advise you to start by reading the DOE book first, at a minimum the introduction to HyperStudy.

We highly appreciate your contributions to further improve the content of our free eBooks!

Let's Fit!

In this eBook, we will describe Fit approximation methods in detail:

Fit Automatically Selected by Training (FAST)

Least Squares Regression (LSR)

Moving Least Squares Method (MLSM)

HyperKriging (HK)

Radial Basis Function (RBF)

Please note that a commercially released software is a living "thing" and so at every release (major or point release) new methods, new functions are added along with improvement to existing methods. This document is written using HyperWorks 2017.2.2 release. Any feedback helping to improve the quality of this book would be very much appreciated.

Thank you very much.

Dr. Matthias Goelke

On behalf of the Altair University Team

2 Acknowledgement

A very special Thank You goes to all the many colleagues who contributed in different ways:

The Editor, **Diana Mavrudieva** for creating, testing and organizing eBook chapters contained in this eBook. Matthias Goelke for the First Edition of DOE eBook.

Fatma Koçer and Joseph Pajot for reviewing and helpful discussion. For sure, your feedback and suggestions had a significant impact on the “shape” and content of this book.

Rahul Rajan for his work on the review of the manuscript.

Rahul Ponginan, Prakash Pagadala, George Johnson, Sanjay Nainani and Premanand Suryavanshi for their very valuable support and stimulating discussions.

Mike Heskitt, Sean Putman & Dev Anand for all the support.

The entire HyperWorks Documentation Team for putting together 1000’s of pages of documentation. Lastly, the entire HyperStudy development team led by Stephan Koerner deserves huge credit for their passion & dedication!

Thank you very much.

Your Altair University Team

3 Disclaimer

Every effort has been made to keep the book free from technical as well as other mistakes. However, publishers and authors will not be responsible for loss, damage in any form and consequences arising directly or indirectly from the use of this book.

4 Introduction

In order to introduce the purpose of this book let us briefly answer some questions you may ask.

1. What is Fit?

Fitting functions are approximations to simulations responses. They are also called response surface models, surrogate models, meta-models or approximations.

In HyperStudy we commonly use the term “to build a Fit” to represent the process of fitting a predictive mathematical model to the data.

2. Why do I need a Fit?

Fitting functions are used when:

- simulation or testing resources are scarce.
- the objective is to have a quick trade-off tool.
- noisy output responses need to be smoothened to increase the effectiveness of further studies;
- there is no physics-based model but there is enough data to create a fit (data-based model).

Let’s take a simple example.

Based on 4 design testing, you have identified that the glue bond strength is maximum when the temperature is 200 degrees and the pressure is 100 psi. You then realized that you cannot set the temperature to 200 degrees but to only 180 degrees. You now need to know the bond strength, but you don’t have the time and budget to do another test.

In this case, you can use the data collected from the 4 design testing and create a fit that will predict the bond strength at temperature 180 degrees without running another test.

- Design Data available:

	Temperature	Pressure	Strength
Experiment #1	100 degrees	50 psi	21 lbs
Experiment #2	100 degrees	100 psi	42 lbs
Experiment #3	200 degrees	50 psi	51 lbs
Experiment #4	200 degrees	100 psi	57 lbs

- Required Design Data:

	Temperature	Pressure	Strength
Reality	180 degrees	100 psi	????? lbs

- Fitting function assuming linear relations:

Strength = a + b (Temperature) +c (Pressure) + d (Temperature)(Pressure)

You need to determine the values of a, b, c and d. For that you need to solve 4 equations for the 4 unknowns by using the design data available. Then you evaluate the fitting function at Temperature = 180, Pressure = 100.

Next, we will explain how you can do this using HyperStudy.

3. How can I create a Fit with HyperStudy?

In HyperStudy there is a dedicated approach called “Fit”. This approach allows creating fitting functions by using approximation methods such as least squares, moving least squares, radial basis function, Hyperkriging and input design data. It also allows analyzing the Fit quality using Diagnostics, Residuals and Quality index. Once a good quality Fit has been created, it can be used in further approaches (DOE, Optimization and Stochastic).

4. What is the challenge when building a Fit?

When using approximations, the issue of a trade-off between accuracy and efficiency is ever present. The challenge is to capture sufficient details of the design space with a minimal amount of data. The solution depends on the nature of the design space as well as the number of design parameters, and how many runs are affordable given the computing resources.

5. What is the purpose of this book?

The purpose of this book is to provide you the foundation you need to leverage the Fit approach in HyperStudy to create predictive models. In the next chapters we will discuss Input and Testing matrices, the methods for fit building, quality analysis, and the steps to follow in HyperStudy.

5 Input and Testing Matrices

As discussed previously, in order to build fitting functions, we use approximation methods and design data. The design data set is also called Input matrix. It contains either simulation results and/or external (such as test, sensor) data. This data will be used to create the fitting function and tune its parameters. Hence, the fit accuracy is highly dependent on the amount of design data provided within the Input matrix and on the distribution of this data in the design space (space filling). Ideally, there should be enough data points equally spread out in the design space for high fitting accuracy. However, a trade-off between accuracy and efficiency is ever present for computationally demanding simulations.

Another “smaller” data set called Testing matrix could be used in addition to the Input matrix to assess the quality of the fitting function. The Testing matrix also contains simulation or external data, which are spread out in the design space. Contrary to the Input matrix data (used to build the fit), the Testing matrix data is used to evaluate the fit in order to compare the predicted values with the real ones.

An Input matrix is compulsory to build a fit, while a Testing matrix is recommended.

You can either create these matrices using HyperStudy DOE approaches or you can use external data.

Next, we will discuss the DOE approaches in HyperStudy for Input and Testing matrices creation.

6 DOE Introduction

In HyperStudy there are three types of DOE: Screening, Space filling (Fit) and Custom. An overview is presented in the chart below. In this chapter we will first introduce the difference between Screening and Space filling DOEs, then we will present the DOEs for space filling available in HyperStudy.

Method	Screening	Filling	Custom
Full Factorial	✓		
Fractional Factorial	✓		
Plackett Burman	✓		
Taguchi	✓		
Mels		✓	
D-optimal		✓	
Central Composite		✓	
Box-Behnken		✓	
Latin Hypercube		✓	
Hammersley		✓	
User-defined Design			✓
Run Matrix			✓

Table 1: DOEs available in HyperStudy.

6.1 Difference Between DOE for Screening and DOE for Space Filling

The purpose of DOE for screening is to get global understanding of the relations between input variables and output responses. For that, experiments are done setting input variables at certain levels only. For instance, in the case of 2-level DOEs (left scatter below) variables are set at their lower and upper bounds, testing thus the extremities of the design space. The experiment results are then analyzed to investigate the relations between input variables and output responses by means of important variables and interconnections between variables.

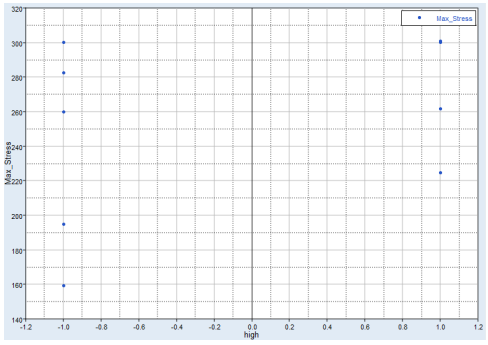


Figure 1. Scatter DOE for screening.

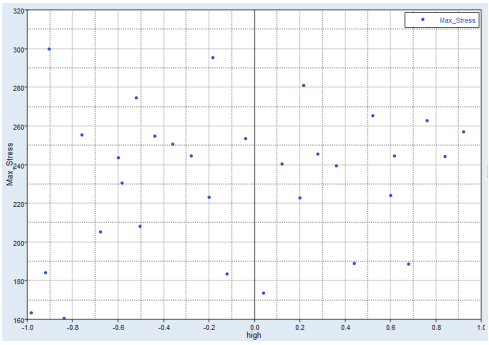


Figure 2. Scatter DOE for space filling.

On the other hand, the purpose of DOE for space filling is to provide relevant information about the model behavior within the entire design space. For that, experiment points are spread out all over the entire design space as shown on the scatter on the right. Such DOEs are appropriate to be used as Input matrix to build fitting functions and as Testing matrix to assess the fit quality.

In the next paragraphs we will discuss the DOEs for space filling available in HyperStudy: MELS; D-optimal; Central composite design; Box-Behnken; Latin Hypercube and Hammersley. We will also discuss how to use external data (such as field or test data) for creating fitting functions.

6.2 Modified Extensible Lattice Sequence (MELS)

MELS is the default DOE method in HyperStudy. It is automatically selected in the Specifications of any DOE approach added in the study.

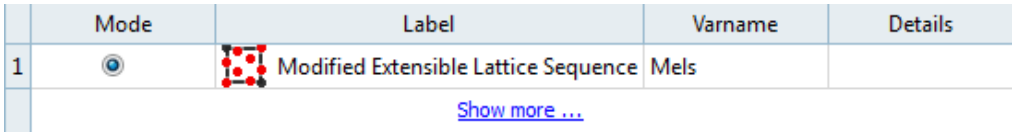


Figure 3: Specifications in DOE approach.

6.2.1 Definition

A lattice sequence is a quasi-random sequence, or low discrepancy sequence, designed to equally spread out points in a space by minimizing clumps and empty spaces. This property makes lattice sequences an excellent space filling DOE scheme. This DOE type also has the property of extensibility, which means the method can take an existing set of data in a space and add more data points to provide equal coverage.

6.2.2 Usability

- MELS is used to approximate highly nonlinear functions.
- MELS supports any variable levels and can be used with continuous, discrete and categorical variables.
- MELS is better space filler than Latin Hypercube.

6.2.3 Settings

The most common changes the user may want to make are increase the number of runs and/or add inclusion matrix.

Parameter	Default	Range	Description
Number of runs*	$\frac{1.1(N+1)(N+2)}{2}$ where N is the number of variables	> 0 integer	Number of new designs to be evaluated. It is recommended not to decrease the default number.
Random Seed	1	Integer 0 to 10000	Controls the repeatability of runs, depending on the way the sequence of random numbers is generated. 0 random (non-repeatable) > 0 triggers a new sequence of pseudo-random numbers, repeatable if the same number is specified
Use Inclusion Matrix**	false	true or false	The use of an inclusion matrix will trigger the DOE to be extensible as it tries to fill in the space already covered by the existing data set.

Table 2: MELS settings.

* To get a good quality fitting function, a minimum number of runs should be evaluated. (N+1)(N+2)/2 runs are needed to fit a second order polynomial, assuming that most responses are close to a second order polynomial within the commonly used design variable ranges of -+10%. An additional number of runs equal to 10% is recommended to provide redundancy, which results in more reliable post-processing. As a result, the recommended equation to calculate the minimum number of required runs is: 1.1*((N+1)(N+2))/2.

** While any data can be used as an inclusion, the best performance can be expected when the inclusion is an existing data set from a MELS DOE.

6.3 D-Optimal

6.3.1 Definition

This DOE type is primarily intended to be used as the input matrix for a least squares regression Fit. By identifying the type of regression that will be used, samples are selected to maximize the determinant of the information matrix that is inverted during the Fit's regression analysis, which in turn improves the numerical efficiency of the DOE. This optimal determinant is the source of the name D-Optimal. Unlike factorial designs, which also have high information matrix determinants, D-Optimal DOEs can have an arbitrary number of runs. The evaluation points are selected from a candidate pool that uses an advanced combination of factorial and space filling designs.

6.3.2 Usability

- D-optimal is used when the known goal is to build a regression;
- D-optimal is also useful when corner coverage is important, and you have problems with input variable constraints;
- The minimum number of allowable runs is equal to the number of unknown coefficients in the selected regression, which in turn is a function of the number of variables;
- The iterative search to improve the information matrix determinant can become computationally expensive. The search contains some internal logic to reduce excessive run times, but overall the time will scale with the number of variables, the number of terms in the regression, and the number of runs in the DOE;
- For linear regression, D-Optimal runs will tend to cluster near the corners and resemble a factorial design pattern. This is expected as edge points will have the largest impact on a linear function. This effect is less pronounced as the order of the regression is increased;
- Any data in the inclusion matrix is combined with the run data for post-processing. The optimal determinant is calculated using candidate and included points;
- Supports input variable constraints.

6.3.3 Settings

D-optimal supports any variable levels and can be used with continuous, discrete and categorical variables. In terms of settings, you can either accept the default number of runs or enter a different value. You can also select the appropriate regression model and/or use inclusion matrix.

Parameter	Default	Range	Description
Number of runs	Twice the number of runs required for a linear regression.	> 0 integer	Number of new designs to be evaluated.
Regression model	Linear	Linear Squared Cubic Interaction Full quadratic Full cubic	The target regression structure used to build the information matrix. Maybe limited by problem size.
Use Inclusion Matrix	false	true or false	Concatenation without duplication between the inclusion and the generated run matrix. The D-Optimal solution considers the locations of the existing inclusion runs when generating its data.

Table 3. D-Optimal settings.

6.4 Central Composite Design

6.4.1 Definition

Central Composite Design (CCD) is the most popular class of designs for fitting second-order model without needing to use a complete three-level factorial experiment. CCD suppresses selected runs from a full factorial matrix in an attempt to maintain the higher order surface definition. For example, for 3 three-level variables, the full factorial run size is 27 (=3^3). The central composite plan drops all of the middle edge nodes, resulting in only fifteen runs.

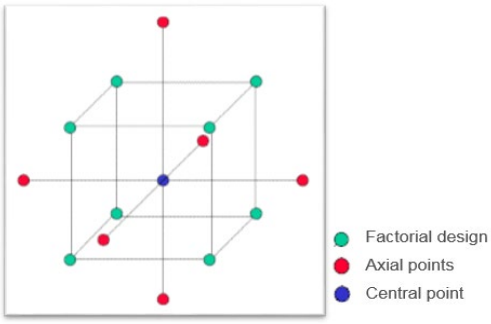


Figure 4. CCD for 3 design variables at 5 levels (15 points).

6.4.2 Usability

You can use this method when the output responses are known to be quadratic. But from a user experience (personal view of course) it is not used often, also because it is rather tricky to find out (or define) the axial points outside the design space.

6.4.3 Settings

CCD supports variable levels up to 5. It can be used with continuous variables only in the limit of 20 variables. The total number of runs is computed as: $2^k + 2k + N$ center points (k = design variables).

The only settings one can change are axial distance and number of center runs depending on the chosen preset:

Preset name	Axial distance	No of center runs
Rotatable	2	User defined
Orthogonal	1.41421	User defined
Rotatable & Orthogonal	2	12
On Face	1	User defined
User Defined	User defined	User defined

Table 4. CCD settings.

6.5 Box-Behnken

6.5.1 Definition

The Box-Behnken design and the Central Composite Design can be visualized as near compliments of each other. Let's consider the same example of 3 three-level variables, we know that the full factorial run size is 27 (=3^3). Contrary to CCD, which drops all of the middle edge nodes resulting thus in 15 runs, Box-Behnken uses the 12 middle edge nodes and the center node to fit a 2^{nd} order equation. Central Composite plus Box-Behnken becomes a full factorial with extra samples taken at the center.

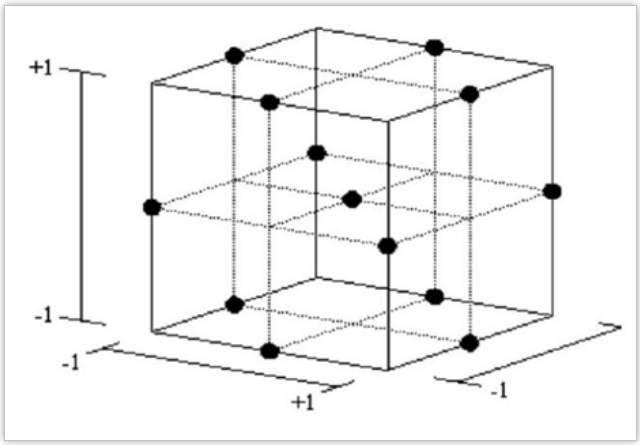


Figure 5. A Box Behnken Design for Three Factors (13 points): 12 middle edge nodes + 1 center node.

6.5.2 Usability

- Box-Behnken is generally used for fitting a second-order response surfaces using fewer required runs than a normal factorial and a CCD.
- Should not be used when accurate predictions at the extremes are important.
- Only defined when all design variables have 3 levels (continuous, discrete or categorical).
- Limited to 7 input variables.

6.5.3 Settings

Parameter	Default	Range	Description
Design	Auto Select	Auto Select, bbdgn13, bbdgn25, bbdgn41, bbdgn49, bbdgn57	Selecting Auto Select will pick bbdgn13 if N < 4, where N is the number of input variables; bbdgn25 if N = 4, bbdgn41 if N = 5, etc. bbdgn stands for Box Behnken design.
Number of runs (npt)	Dependant upon design selected.	13-57	Number of designs to be evaluated.

Use Inclusion Matrix	false	True or False	Concatenation without duplication between the inclusion and the generated run matrix.

Table 5. Box-Behnken settings.

6.6 Latin Hypercube

6.6.1 Definition

Latin Hypercube is a square grid containing sample positions is a Latin square if, and only if, there is only one sample in each row and each column. A Latin Hypercube DOE, categorized as a space filling DOE, is the generalization of this concept to an arbitrary number of dimensions.

When sampling a design space of N variables, the range of each variable is divided into M equally probable intervals. M sample points are then placed to satisfy the Latin Hypercube requirements. As a result, all experiments have unique levels for each input variable and the number of sample points, M, is not a function of the number of input variables. Of course, the more points we have, the better populated the design space will be.

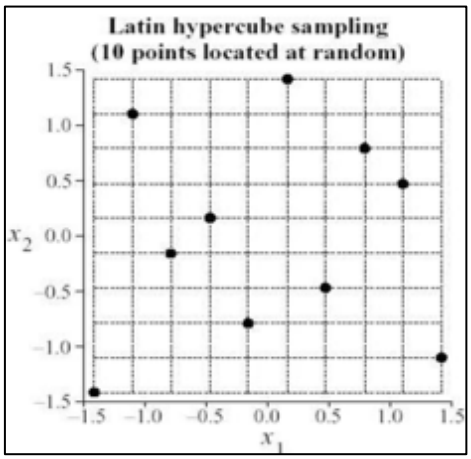


Figure 6. Latin Hypercube sampling 10 points.

6.6.2 Usability

- Use this method when the response surface is highly nonlinear.
- Used when the factors of interest have more than two levels and where there are no (or negligible) interactions between factors.

6.6.3 Settings

Parameter	Default	Range	Description
Number of runs	$\frac{1.1(N+1)(N+2)}{2}$ where N is the number of variables	> 0 integer	Number of designs to be evaluated. It is recommended not to decrease the default number.
Random Seed	1	Integer 0 to 10000	Controls the repeatability of runs, depending on the way the sequence of random numbers is generated. 0 random (non-repeatable) > 0 triggers a new sequence of pseudo-random numbers, repeatable if the same number is specified
Use Inclusion Matrix	false	true or false	The use of an inclusion matrix will trigger the DOE to be extensible as it tries to fill in the space already covered by the existing data set.

Table 6. Latin Hypercube settings.

6.7 Hammersley

6.7.1 Definition

Hammersley sampling belongs to the category of quasi-Monte Carlo methods. This technique uses a quasi-random number generator, based on the Hammersley points, to uniformly sample a unit hypercube, but also attempts to distribute the points evenly within the design space. It works in similar way to Latin Hypercube, but it has a better coverage of the design space. Look at the image below, in the Latin Hypercube you see more a kind of clustering of design points than in Hammersley; also there are quite some “white”, i.e. uncovered areas.

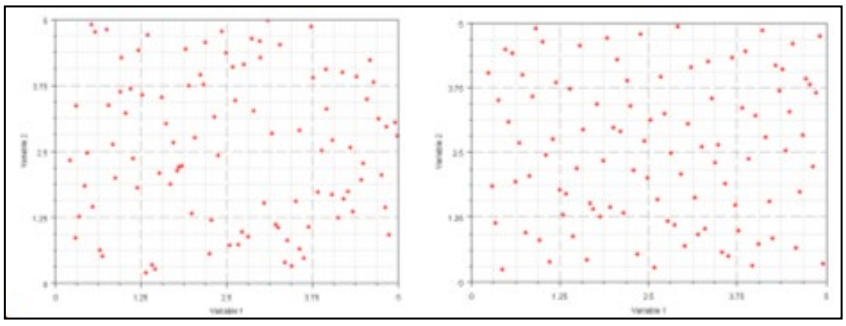
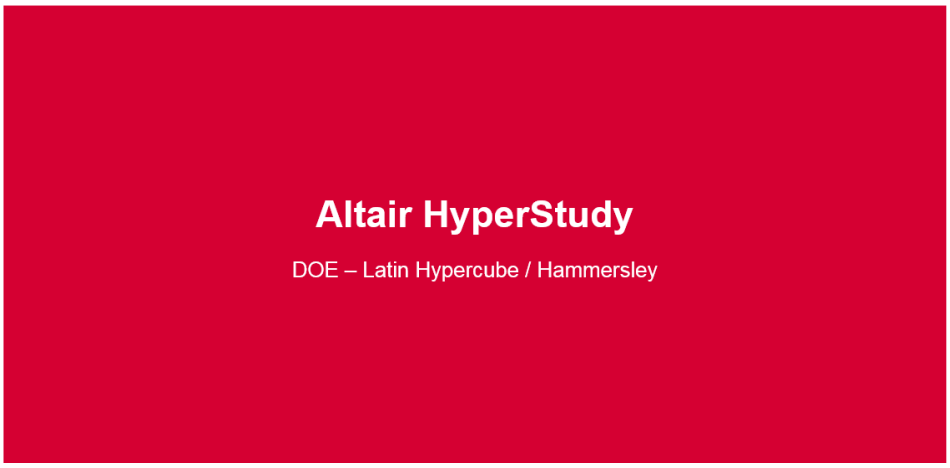


Figure 7. Latin Hypercube (left) and Hammersley (right) for 100 runs.

6.7.2 Usability

- Use this method when the response surface is highly nonlinear. This method is a better space filler than Latin Hypercube.

A summary on the last two methods is provided in the following recording:



https://altair-2.wistia.com/medias/r9ioggblnn

6.7.3 Settings

Parameter	Default	Range	Description
Number of runs	$\frac{1.1(N+1)(N+2)}{2}$ where N is the number of variables	> 0 integer	Number of designs to be evaluated. It is recommended not to decrease the default number.
Use Inclusion Matrix	false	true or false	The use of an inclusion matrix will trigger the DOE to be extensible as it tries to fill in the space already covered by the existing data set.

Table 7. Hammersley settings

6.8 User-Defined

The User-Defined method allows you to load your own design matrix. It is read in by HyperStudy and is used like any other design matrix. The advantage of using a User-Defined design is that you can create your own design based on individual requirements.

The user defines a perturb file (.csv) such as below and imports it in HyperStudy. In order to delimit the individual elements of the matrix you can use spaces, tabs, or commas.

9	3	
1	1	1
1	2	2
1	3	3
2	1	3
2	2	2
2	3	1
3	1	2
3	2	3
3	3	1

Figure 8. User-Defined matrix with 9 runs and 3 variables at different levels.

In the first row of this file, one must specify the number of runs (rows) and the number of columns (number of variables). Then one defines the matrix by using integers to represent the corresponding level of each variable. This is in contrast to the Run Matrix DOE (next paragraph), which contains exact values of the variables. Note that the number of levels specified in the file must be consistent with the number of variable levels specified in the HyperStudy user interface. As a result, imported values are mapped to the independent variables.

You can also choose to use inclusion matrix combined with user-defined matrix. Any run point which is already part of the inclusion data will not be rerun.

6.9 Run Matrix

Same as User-Defined, the Run Matrix method allows you to load your own design matrix. Contrary to User-Defined, the Run Matrix uses exact values of the variable.

The user defines a Matrix file (.csv) such as below and imports it in HyperStudy. In order to delimit the individual elements of the matrix you can use spaces, tabs, or commas.

1.0	2.0	3.0	4.0
4.1	4.3	4.5	4.6
6.7	8.1	10.0	11.0
17.2	1.0	1.0	3.0
.02	0.4	0.5	1.7
3.4	2.1	7.3	9.1

Figure 9. Run matrix with 6 runs (rows) and 4 variables (columns).

It is not necessary to utilize all designs in a study. Designs that are not desired can be turned off from the Write/Execute runs panel.

6.10 None

Again, it allows defining your own matrix, this time however, you are using Test Data consisting of variables (DV) and responses R1 (=results; measured values).

DV1	DV2	R1
150	52	47
132	98	54
198	42	67

Figure 10. Example of No method.

6.11 Lookup Tables

A Lookup model type is available in HyperStudy version 2017.2.2 or above. It allows connecting quickly and efficiently HyperStudy to a tabular data (csv file). Using a look-up model, you can point to your .csv file with your inputs and outputs. You then need to enter the number of inputs and HyperStudy will then import all the inputs and outputs with their labels and values. During evaluation the model returns output responses that match the given inputs, or it will fail if there are no valid matches. This model can be used to streamline studies that use pre-existing data or can be used as conditional logic in multi-model studies.

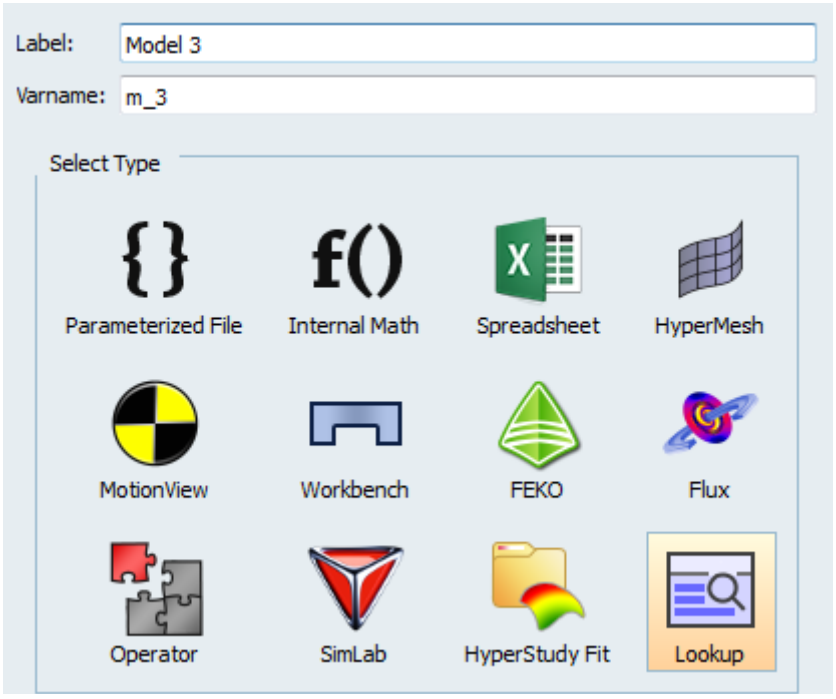


Figure 11. Model types in HyperStudy.

7 Fit Methods in HyperStudy

7.1 Introduction

In an attempt to create good-quality approximations to different problems, many methods have been developed.

In this chapter, we will cover the approximation methods that are implemented in HyperStudy, namely Least Squares Regression (LSR), Moving Least Squares Method (MLSM), HyperKriging (HK), Radial Basis Function (RBF) and Fit Automatically Selected by Training (FAST).

7.2 Fit Automatically Selected by Training (FAST)

FAST method is available in HyperStudy version 2018 or above. It is proposed as default choice in the Specifications of a Fit approach.

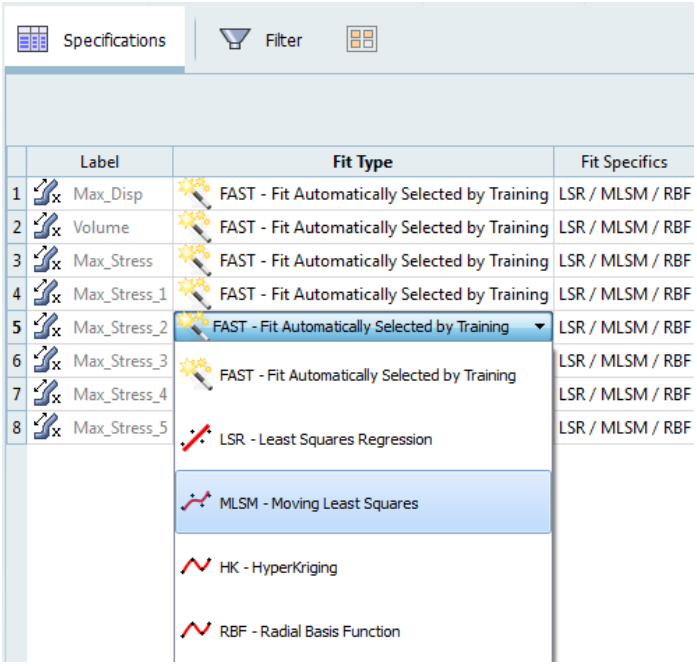


Figure 12. Specifications in Fit approach.

FAST allows building automatically best fitting functions. To do that, it is testing automatically all the methods available (Stepwise Regression Terms, LSR, MLSM, HK and RBF) and their settings to figure out which one is the most appropriate to obtain best quality fit for each approximated function.

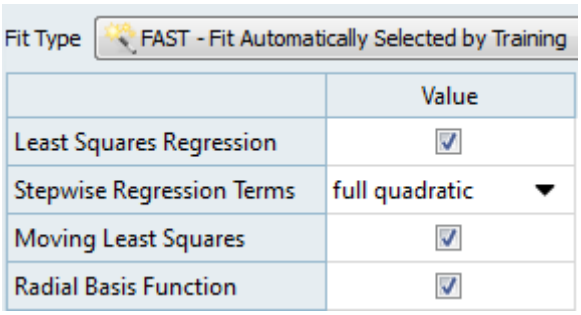


Figure 13. FAST settings.

It uses the cross-validation R^2 for comparison purposes. Different methods and settings could be applied in case of several approximated functions: for instance, RBF could be the best method for F1, and MLSM could be the best method for F2. As a result, using FAST considerably simplifies the fit building process.

7.3 Least Squares Regression (LSR)

7.3.1 Mathematical Basics

Least Squares Regression (LSR) attempts to minimize the sum of the squares of the differences (*residuals*) between responses generated by the approximation and the corresponding simulation results. This function is defined as:

$$\min\ E = \sum_{i=1}^{n} \left(f_i^{predicted} - f_i \right)^2$$

where n is the number of designs, $f_i^{predicted}$ is the output response value predicted by the regression model for the i^{th} design, and f_i is the output response value from the simulation of the i^{th} design. This is achieved by finding the regression model coefficient values that sets the derivative of E, with respect to each unknown coefficient, to zero.

In order to minimize E, it is necessary to take the derivative of summation with respect to each of the unknown coefficients and set each to zero:

$$\frac{\partial E}{\partial A_j} = 2\sum_{i=1}^{n} \left(f_i^{predicted} - f(f_i) \right) \left(\frac{\partial f}{\partial A_j} \right) = 0 \quad for\ all\ j=1 \ldots m.$$

For a function with *m* number of unknown coefficients, this will yield **m** equations. By solving this system of equations, an analytical solution can be found. This is one major advantage to the least squares method. There is no need for iteration or approximation with the linear least squares method.

7.3.2 Usability

- LSR is well suited for linear and noisy output responses.
- Quality of a LSR fit is a function of the number of runs, order of the polynomial, and the behavior of the application.

- If the quality is poor, then you can increase the order of the Fit if you have enough runs to fit that specific order.
- If increasing the order does not improve the fit quality, then you may want to inspect the input matrix collinearity and optionally add more runs. You should try the other available Fit methods as your application may have more non-linearity than polynomials can handle.
- Equation of the approximation is available.

7.3.3 Settings

In the Settings tab, you can access the following settings. By default, a linear regression model (1st order terms only) is used to build LSR fit. But you may need to change it as order is a critical setting for the quality of Fit. If the Fit quality is poor, then it is advised that you try the other regression models. Note that higher order models require a higher number of runs. Also, a higher order does not necessarily mean a better prediction quality.

Parameter	Default	Range	Description
Regression Model	Linear	Linear Squared Cubic Interaction Full Quadratic Full Cubic Custom	**Linear** First order terms only. $Y=A+Bx+Cy$ **Squared** Second order without cross terms. $Y=A+Bx+Cy+Dx^2+Ey^2$ **Cubic** Third order without cross terms. $Y=A+Bx+Cy+Dx^2+Ey^2+Fx^3+Gy^3$ **Interaction** Linear and the cross terms. $Y=A+Bx+Cy+Dxy$ **Full Quadratic** Complete second order polynomial. **Full Cubic** Complete third order polynomial. **Custom** User defined order and terms.

Table 8. LSR settings.

In the case of a Custom regression model HyperStudy allows the following additional settings:

- Creation of least squares regressions for any polynomial order (see the “Order” box below). This is a differentiator for HyperStudy, however in most cases polynomial of 4th order or higher do not bring more accuracy.

- Regression terms can be toggled on or off in the “Active” column below, allowing for suppression of terms known to be less significant.

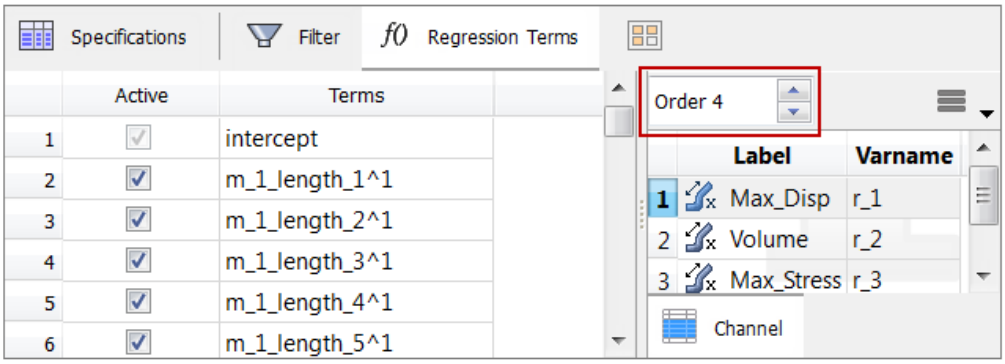


Figure 14. Custom Regression settings.

7.4 Moving Least Squares Method (MLSM)

7.4.1 Mathematical Basics

Moving least squares (MLSM) is a method of reconstructing continuous functions from a set of unorganized point samples via the calculation of a weighted least squares measure biased towards the region around the point at which the reconstructed value is requested.

MLSM is a generalization of a conventional weighted least squares model building. The main difference is that the weights associated with the individual DOE sampling points do not remain constant but are functions of the normalized distance from a sampling point to a point *x* where the surrogate model is evaluated: The weight, associated to a sampling point, decays as the evaluation point moves away from it. The decay is defined through a decay function (HyperStudy offers Gaussian, cubic, fourth, fifth or seventh order).

As weights are not constant in MLSM (they are dependent on the sampling point), there is no analytical form and so no equation is given in MLSM approximations.

HyperStudy allows the creation of moving least squares regressions for first, second, and third order. This is the order of the base polynomial.

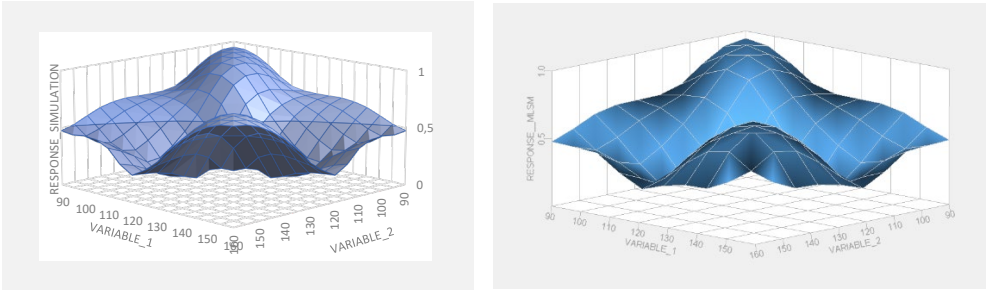


Figure 15. Comparison of Exact Simulation (left) with MLSM Approximation (right).

7.4.2 Usability

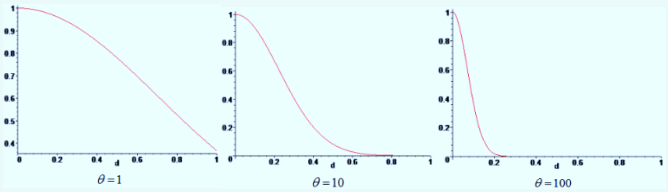
- MLSM is suitable for non-linear and noisy responses.
- It is recommended to use a Testing matrix in addition to an Input matrix for better diagnostics;
- Quality of an MLSM Fit is a function of the number of runs, order of the polynomial and the behavior of the application.
- If the quality is poor, then you can increase the order of the Fit if you have enough runs to fit that specific order.

Settings

A number of options are available for MLSM in HyperStudy. Note that for most applications the default settings work optimally, and you may only need to change the Order to improve the Fit quality.

Parameter	Default	Range	Description
Fit Parameter	5.0	>= 0.0 <= 10.0	The parameter controls the effect of screening out noise; the larger value, the less effect.
Minimum Weight	0.001	> 0.0	Minimum weight.
Number of Excess Points	3	>=0	Number of excessive points to build MLSM.
Regression Model	linear	Linear Squared Cubic Interaction Full quadratic Full cubic	The order of polynomial function. Linear : First order Squared : 2nd order without interactions Cubic: 3rd order without interactions Interaction: 1st order plus interactions Full quadratic: 2nd order with interactions Full cubic: 3rd order with all interaction terms
Weighting Function	Gaussian	Gaussian	The type of weighting function. Gaussian (recommended): $W_i = \exp\left(-\theta r_i^2\right)$ where r_i is the normalized distance from the i-th sampling point to a current point. The parameter θ defines the "closeness of fit", the case θ=0 is equivalent to the traditional least squares regression. When the parameter θ is large, it is possible to obtain a very close fit through the sampling points, if desired. The following figures

illustrate the change of the weight over the interval [0,1] where the sampling point is at r = 0:

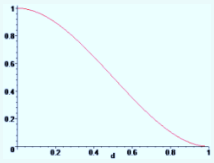


Cubic

Cubic

$$w_i = 1 - 3p_i^2 + 2p_i^3$$

where $p_i = r_i / R_{max}$, R_{max} is the normalized radius of the sphere of influence:

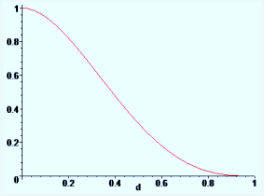


The normalized radius of the sphere of influence R_{max} inversely relates to the closeness of fit parameter, for example the smaller the value of R_{max}, the closer fit is obtained.

Fourth Order

Fourth Order

$$w_i = 1 - 6p_i^2 + 8p_i^3 + 3p_i^4$$



Fifth Order

Fifth Order

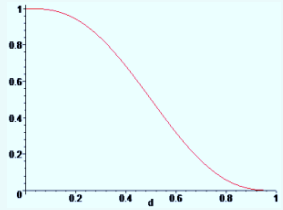
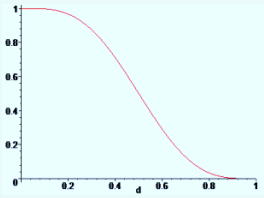
		Seventh Order	$w_i = 1 - 10p_i^3 + 15p_i^4 + 6p_i^5$  **Seventh order** $w_i = 1 - 35p_i^4 + 84p_i^5 + 70p_i^6 + 20p_i^7$ 

Table 9. MLSM settings.

7.5 HyperKriging (HK)

7.5.1 Mathematical Basics

HyperKriging is a scheme used to create predictive models with data sets coming from deterministic computer simulations, an area of application commonly known as the Design and Analysis of Computer Experiments (DACE). These experiments are unique because they do not require some concepts such as replication. This approximation method is designed to tightly pass through and smoothly interpolate between the known points.

As it can be seen on Figure 16, HyperKriging method passes through all design points and hence the error between predicted values and observed values is always equal to zero. As a result, one cannot obtain any diagnostics for the quality of the approximation. In order to overcome this critical limitation, it is suggested to have a set of designs as a testing matrix. We can then validate the quality of a HyperKriging approximation using designs that are not included during the creation of the approximation.

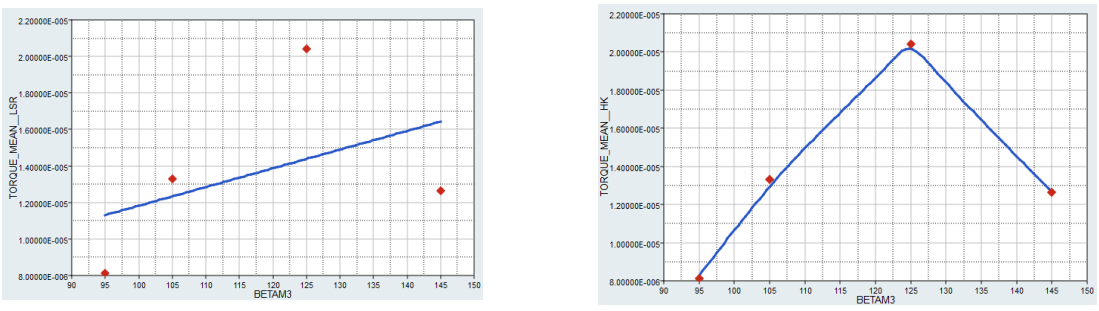


Figure 16. LSR (left) and HK (right) comparison.

7.5.2 Usability

- Suitable for modeling highly nonlinear output response data that does not contain numerical noise.
- It is suggested that you use HyperKriging for large studies that contain a large number of sampling points.

7.6 Radial Basis Function (RBF)

7.6.1 Mathematical Basics

Radial Basis Function method is a Fit method that uses linear combinations of basic functions. Typical basis functions are linear, cubic, thin-plate spline, Gaussian, multiquadric, and inverse-multiquadric. These basic functions are observed to be accurate for highly nonlinear output responses but not for linear output responses. To remedy this deficiency, in HyperStudy, an RBF model is augmented with a polynomial function.

7.6.2 Usability

RBF tries to go through the exact sampling points, and in general, the residuals are small, if not zero. As a result, diagnostic measures using only the complete input matrix do not produce meaningful values. Cross-validation results provide some diagnostics using a special scheme using only the input points. To get detailed diagnostics on the quality of a RBF fit, it is suggested that you use a testing matrix.

- Suitable for modeling highly nonlinear output response data that does not contain numerical noise;
- It is suggested that you use RBF for studies with a large number of variables.

Note: RBF Fits are evaluated faster than HyperKriging Fits when used in approaches. The reason is that each HK call back involves much more steps than RBF, which has already determined the weights/formulas in a manner similar to LSR.

7.6.3 Settings

Parameter	Default	Range	Description
Augmented Function	Constant	Constant or Linear	Type of augmented function.
Maximum Points	2000	>= 100	The maximum number of points for building RBF; if number of building points is larger than maxnpt, then the point reduction algorithm is activated and a warning message is shown; the purpose of introducing maxnpt is to reduce computational effort for large scale problems.

RBF Type	CS21	Multiquadric CS21 (formally known as Wu's Compactly Supported (2,1)) Gaussian	Type of RBF.
Relaxation Parameter	1.0	>= 0.0	Relaxation parameter d used in RBF; if RBF is CS21 or Gaussian, and d is set to 0.0 by users, then RBF will automatically set d = 1.0e-6.

Table 10. RBF settings.

Note: For most application default settings work optimally.

7.7 Overview

The overview below has the purpose to help you in choosing the right method to use in case that you would prefer not to use FAST.

Type	Equation Available	Highly Non Linear	Noisy Responses	Accuracy	Efficiency
LSR	✓	✗	✓	★	★★★
MLSM	✗	✓	✓	★★	★★
HK	✗	✓	✗	★★★	★★
RBF	✗	✓	✗	★★★	★★

Table 11. Approximation methods overview.

Accuracy reflects the method capability to build good quality fit. A “good quality fit” is a fit that predicts simulation responses best and for which the Testing matrix Residuals are the smallest.

Efficiency reflects the trade-off between accuracy and cost by means of computation time. Note that we can consider two computation times:

1. The time needed to build the Fit.

The time to build the fit can get larger depending on the # of DV versus the number of runs. For instance, HK is very impractical when the number of runs is large. In such cases you would prefer using RBF instead. RBF also slows down as the number of runs increases, but not nearly as slowly as HK. Both methods scale with the number of variables, but the scaling is similar.

2. The run time when the fit approximation is used in other approaches.

As discussed previously RBF fits are much faster to be evaluated than HK approximations.

8 Fit Approach in HyperStudy

8.1 Introduction

In order to create fitting functions in HyperStudy one uses the Fit approach.

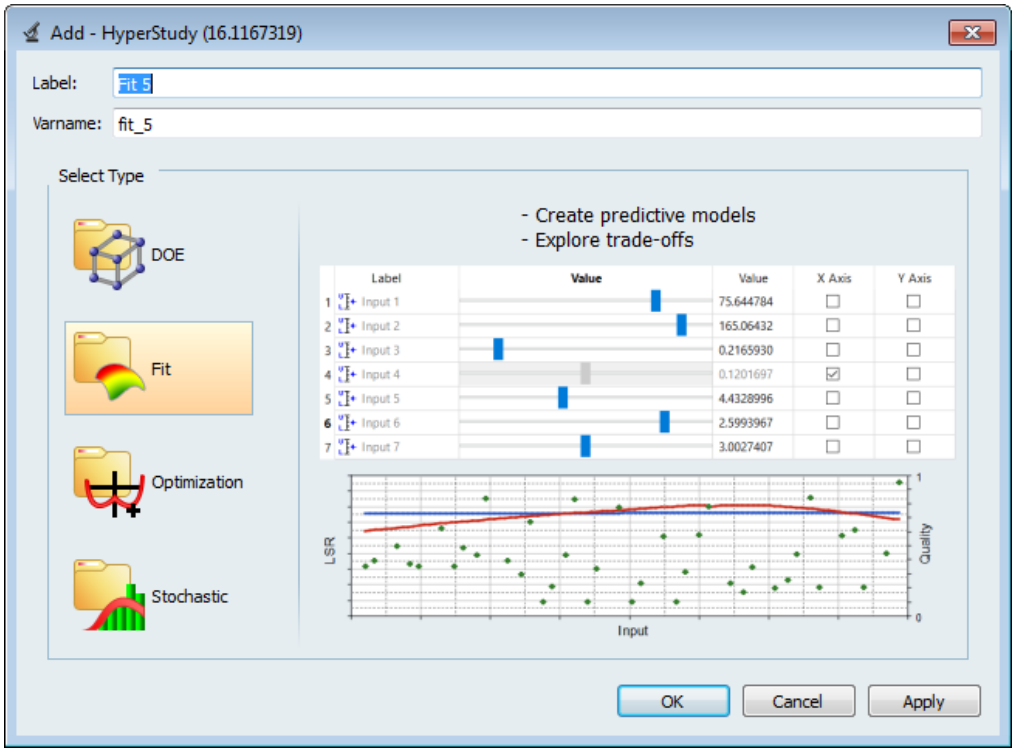
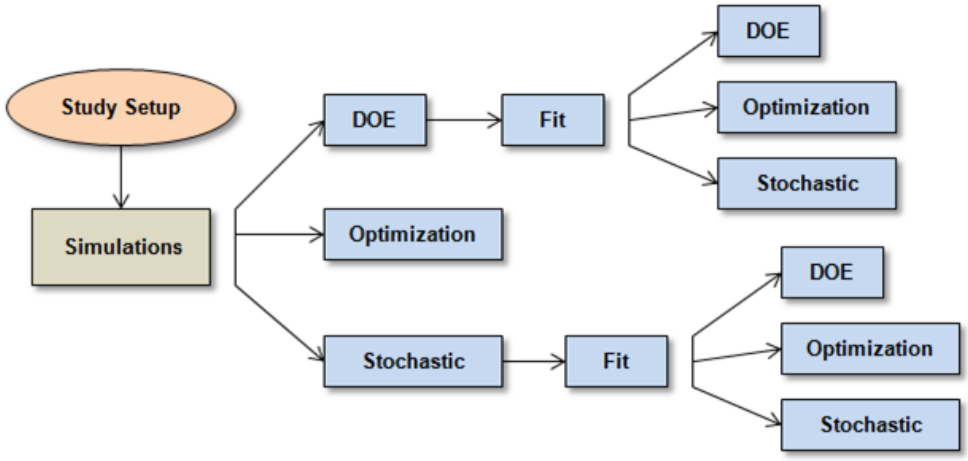


Figure 17. Approach selection in HyperStudy.

A Fit approach can be added in the study once the Setup has been complete, and more generally once a DOE or Stochastic studies have been solved.



8.2 Approach Steps

Once a Fit approach has been added in the study, a series of steps will display under the approach in the Explorer as shown on Figure 18. Using step-by-step definition and color codes, the user is guided to complete the definition. At the beginning you only have the possibility to define Select Matrices before going through the other steps. Each step must be validated by a green check mark before progressing on to the next step.

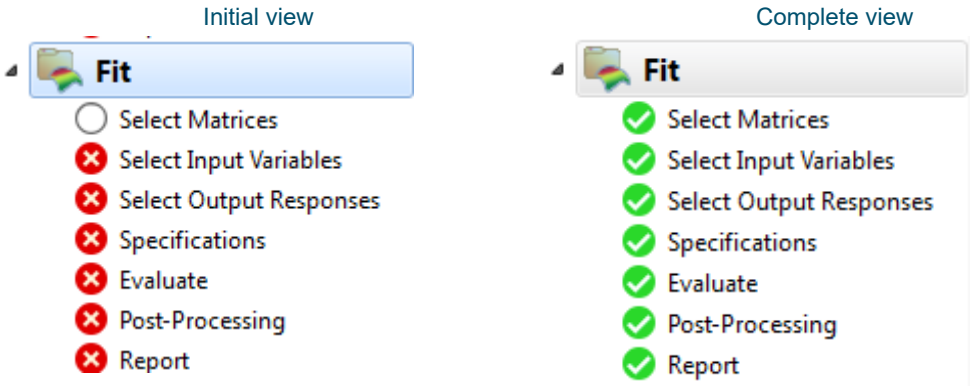
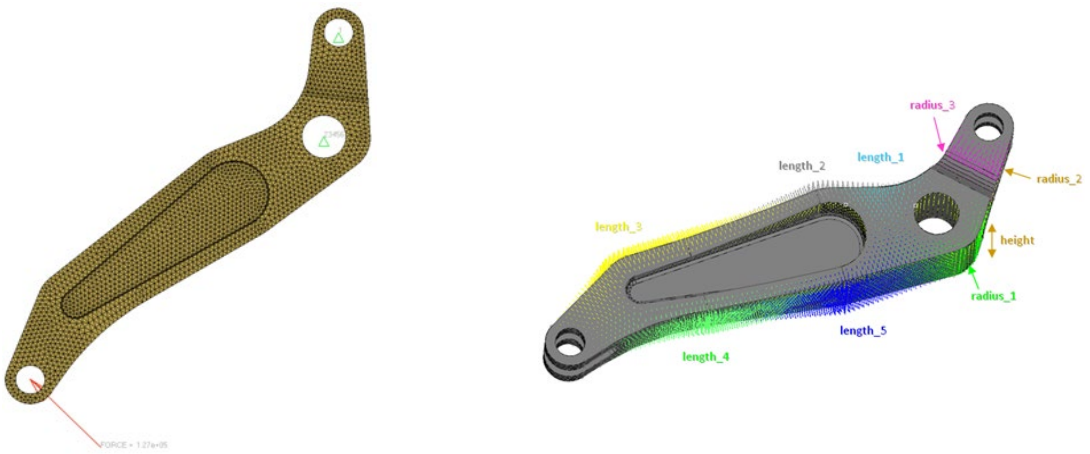


Figure 18. Fit approach definition.

The same steps are available for each approach except the **Select Matrices** step, which is only required in Fit approach.

In the next sections we will describe each of the Fit approach steps through an example.

The study case is presented on Figure 19. It is an arm model that has been meshed and modeled in HyperMesh and analyzed using OptiStruct. The arm is clamped at one end and under an axial loading on the other end.



Concentrated load and boundary conditions

Shapes defined on the arm model

Figure 19. Arm model study case.

The design can change shape in nine different regions (9 shapes defined), as shown on the image above, and there are three output responses: Volume; Max_Stress (Max Von Mises Stress) and Max_Disp (Nodal displacement of the node where the force is applied).

We are interested in creating fitting function on Max_Stress in order to use it in further approaches such as optimization and stochastic. We assume that we don't have prior knowledge on the function behavior: linear, nonlinear or noisy.

Starting from a study that already contains Setup and two DOE approaches (Figure 20), we will use Fit approach to create approximations.

	Label	Varname	Type
1	Setup	nom_1	Nominal
2	Mels	doe_5	DOE
3	Hammersley_12	doe_6	DOE

Figure 20. Starting point for the study.

In what follows, we will cover specifically the **Select Matrices**, **Specifications** and **Post-processing** steps.

For step-by-step instructions on the Setup and the DOE studies you can refer to the tutorial HS-2000. For step-by-step instructions on the Fit approximations you can refer to the tutorial HS-3000.

8.2.1 Select Matrices

As mentioned before, this step is only required in Fit approach. It consists on importing data matrices to build the fit and assess the quality.

There are two types of matrices that you can import:

- **Input Matrices** - the data will be used to create the fit and tune its parameters.
- **Testing Matrices** - the data will be used to assess the quality of the fit.

An Input matrix is compulsory to validate the step by a green check mark. A testing matrix is recommended, especially if HK and RBF methods are used.

In our example two DOEs have been added (Figure 21):

- **MELS** with 31 runs, which will be used as Input matrix;
- **Hammersley** with 12 runs, which will be used as Testing matrix.

Add Matrix Remove Matrix

	Active	Label	Varname	Type	Matrix Source	Matrix Origin	Status
1	☑	Fit Matrix 1	fitmatrix_1	Input	Mels (doe_5)	DoeMels	Import Successful
2	☑	Fit Matrix 2	fitmatrix_2	Testing	Hammersley_12 (doe_6)	DoeHammersley_12	Import Successful

Figure 21. Data matrices.

Once the matrices have been imported by clicking on Import Matrix, the step is complete, and one can progress on to the next step.

The matrices should be imported from an existing DOE or Stochastic approach and can be further edited on the fly. For instance, if you modify the number of runs in your DOE approach you can update the matrices in the Fit approach as well.

8.2.2 Select Input Variables

This step is complete simultaneously with Select Matrices and hence there is no needed action. The input variables table (Figure 22) is automatically populated using the information from the Input matrix, for instance the 3 radii are automatically disabled as it has been done in the DOE MELS approach. At this stage it is not possible to make them active, neither to change any variable bounds. It is only possible to disable additional variables.

Add Input Variable Remove Input Variable

	Active	Label	Varname	Lower Bound	Nominal	Upper Bound
1	☑	high	m_1_high	-1.0000000	0.0000000	1.0000000
2	☐	radius_3	m_1_radius_3	-0.5000000	0.0000000	1.0000000
3	☐	radius_2	m_1_radius_2	-0.5000000	0.0000000	1.0000000
4	☐	radius_1	m_1_radius_1	-2.0000000	0.0000000	2.0000000
5	☑	length_5	m_1_length_5	-1.0000000	0.0000000	1.0000000
6	☑	length_4	m_1_length_4	-1.0000000	0.0000000	1.0000000
7	☑	length_3	m_1_length_3	-1.0000000	0.0000000	1.0000000
8	☑	length_2	m_1_length_2	0.0000000	0.0000000	2.0000000
9	☑	length_1	m_1_length_1	-0.5000000	0.0000000	2.0000000

Figure 22. Input variables.

8.2.3 Select Output Responses

Same as for Select Input Variables, this step is complete simultaneously with Select Matrices and hence no specific action is needed. Nevertheless, one can disable some responses, which are not of interest for the fit creation. As in this example we disable Volume and Max_Disp. Then, we go on the next step.

Add Output Response Remove Output Response Evaluate from Fit Model

	Active	Label	Varname	Expression
1	☐	Max_Disp	m_1_r_1	max(readsim(getenv("HST_APPROACH_RUN_PATH") + "/m_1/arm.h3d", "Su...
2	☐	Volume	m_1_r_2	max(m_1_v_1)
3	☑	Max_Stress	m_1_r_3	max(readsim(getenv("HST_APPROACH_RUN_PATH") + "/m_1/arm.h3d", "Su...

Figure 23. Output responses.

8.2.4 Specifications

In Specifications we can select the approximation method, which will be used to build the approximation functions, or use the default option. As discussed in previous chapters, FAST is selected by default for all the responses. There are 4 other methods available. For the first test we select LSR from the Fit Type scroll down menu (Figure 24).

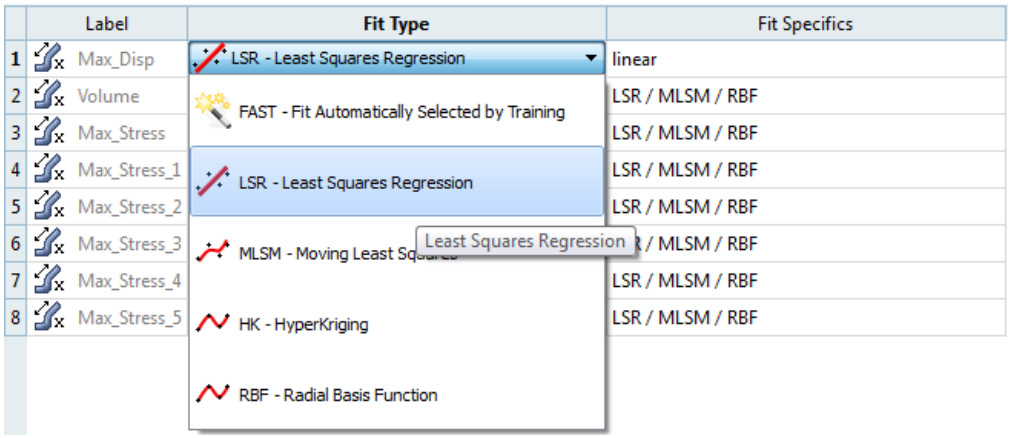


Figure 24. Approximation method selection.

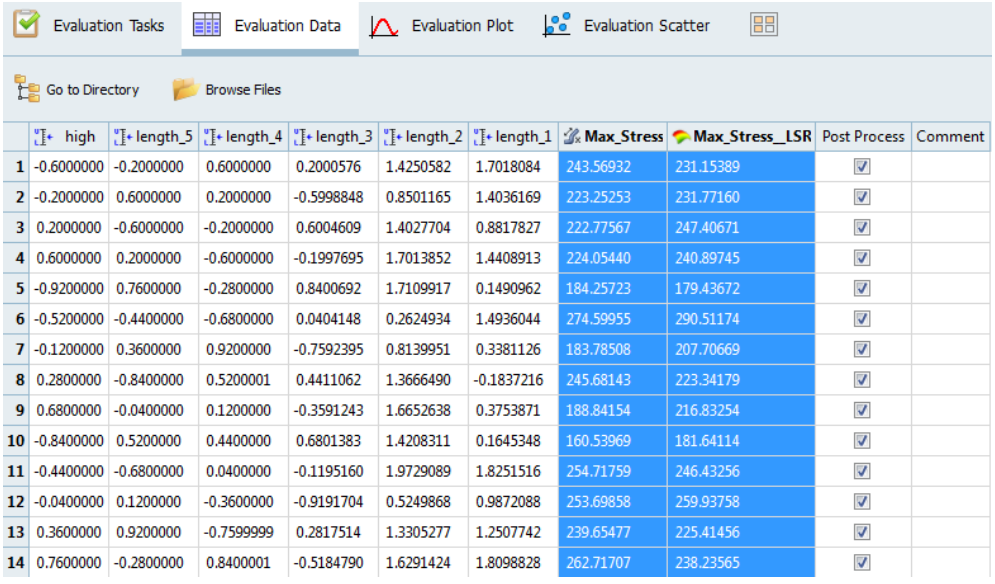
It is needed to click on Apply to validate the choice before progressing on to the next step.

8.2.5 Evaluate

In the Evaluate step, approaches are executed and monitored. In Fit approach (as in DOE and Stochastic) the Evaluate step consists in the following tabs:

- Evaluation Tasks.
- Evaluation Data;
- Evaluation Plot;
- Evaluation Scatter.

In the Evaluation Data tab, you can review a tabular summary of the Input matrix runs: variable and output response values for each run (Figure 24). You can also compare the output response value from the simulation (Max_Stress) with the predicted value from the method (Max_Stress_LSR).



	high	length_5	length_4	length_3	length_2	length_1	Max_Stress	Max_Stress_LSR	Post Process	Comment
1	-0.6000000	-0.2000000	0.6000000	0.2000576	1.4250582	1.7018084	243.56932	231.15389	☑	
2	-0.2000000	0.6000000	0.2000000	-0.5998848	0.8501165	1.4036169	223.25253	231.77160	☑	
3	0.2000000	-0.6000000	-0.2000000	0.6004609	1.4027704	0.8817827	222.77567	247.40671	☑	
4	0.6000000	0.2000000	-0.6000000	-0.1997695	1.7013852	1.4408913	224.05440	240.89745	☑	
5	-0.9200000	0.7600000	-0.2800000	0.8400692	1.7109917	0.1490962	184.25723	179.43672	☑	
6	-0.5200000	-0.4400000	-0.6800000	0.0404148	0.2624934	1.4936044	274.59955	290.51174	☑	
7	-0.1200000	0.3600000	0.9200000	-0.7592395	0.8139951	0.3381126	183.78508	207.70669	☑	
8	0.2800000	-0.8400000	0.5200001	0.4411062	1.3666490	-0.1837216	245.68143	223.34179	☑	
9	0.6800000	-0.0400000	0.1200000	-0.3591243	1.6652638	0.3753871	188.84154	216.83254	☑	
10	-0.8400000	0.5200000	0.4400000	0.6801383	1.4208311	0.1645348	160.53969	181.64114	☑	
11	-0.4400000	-0.6800000	0.0400000	-0.1195160	1.9729089	1.8251516	254.71759	246.43256	☑	
12	-0.0400000	0.1200000	-0.3600000	-0.9191704	0.5249868	0.9872088	253.69858	259.93758	☑	
13	0.3600000	0.9200000	-0.7599999	0.2817514	1.3305277	1.2507742	239.65477	225.41456	☑	
14	0.7600000	-0.2800000	0.8400001	-0.5184790	1.6291424	1.8098828	262.71707	238.23565	☑	

Figure 24. Summary of the Input matrix runs.

In the Evaluation Plot tab, you can compare Max_Stress and Max_Stress_LSR in function of the runs (Figure 25).

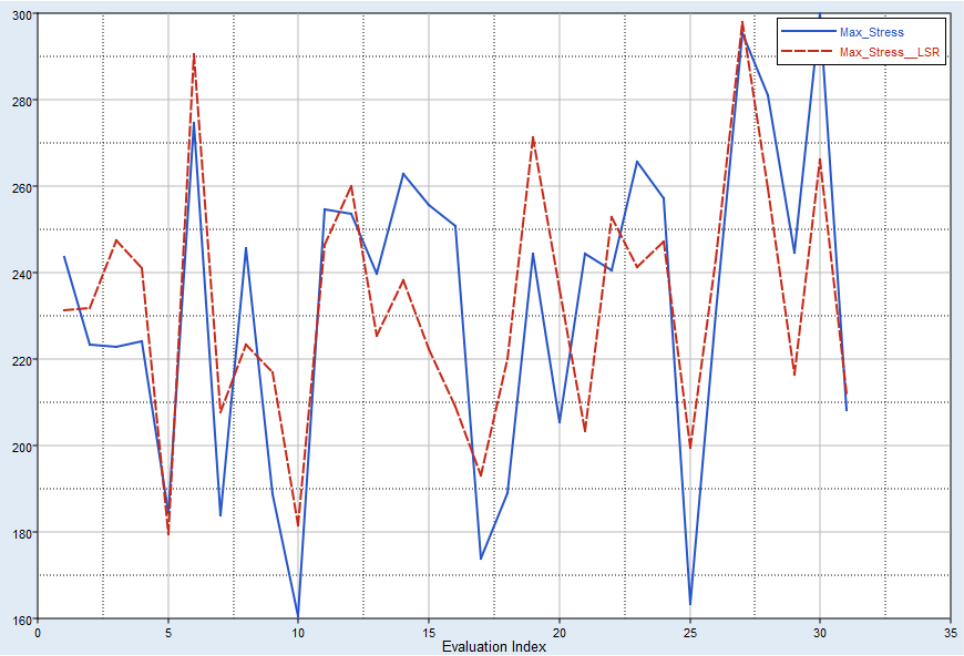
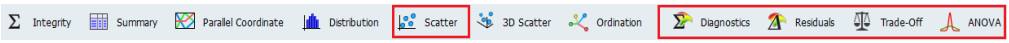


Figure 25. Simulated and predicted functions comparison.

In the Evaluation Scatter tab, you can visualize the distribution of the runs for each variable, or also display original function versus predicted one. We will discuss Scatter use in the next section.

8.3 Post-Processing

The post-processing will be particularly detailed as it is essential to be able to assess the fit quality and make decisions in function of the results. In Fit approach, the Post-processing step proposes the following tabs.



Tool	Description
Integrity	Review statistics of data sets.
Summary	Present raw data in tabular form.
Parallel Coordinate	Identify trends in large data sets.
Distribution	Visualize data trends and identify outliers.
Scatter	Visualize data sets in 2 dimensions.

3D Scatter	Visualize data sets in 3 dimensions.
Ordination	Identify trends in large data sets.
Diagnostics	Assess response surface quality in summary.
Residuals	Assess response surface quality in detail.
Trade-Off	Interactive output response surface tool to perform "what if" scenarios.
Anova	Identify the importance of LSR factors.

Table 12. Post-processing tools.

The most important tools for the fit quality analysis are: **Scatter, Diagnostics, Residuals and Trade-off**.

8.3.1 Scatter

The Scatter is the first thing you may want to check. It allows visualizing the fit accuracy in comparison with the original function. The plot on Figure 26 shows Max_Stress_LSR vs Max_Stress. The more the points follow the diagonal, better the fit is. On the below plot we can see lot of dispersed points, which are not on the diagonal. It means poor fit accuracy in this case. The next step will be to check the diagnostics.

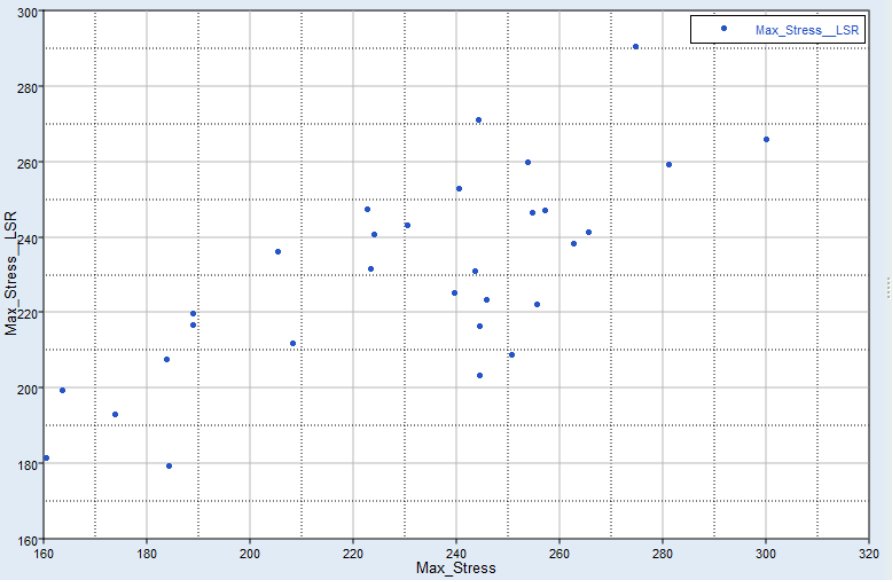


Figure 26. Original vs Predicted functions comparison.

8.3.2 Diagnostics

Diagnostics helps in accessing the accuracy of a Fit.

	Criterion	Input Matrix	Cross-Validation Matrix	Testing Matrix
1	R-Square	0.5923174	0.3172670	0.6509026
2	R-Square Adjusted	0.4903968	N/A	N/A
3	Multiple R	0.7696216	0.5632646	0.8067853
4	Relative Average Absolute Error	0.5659899	0.7182214	0.5123023
5	Maximum Absolute Error	41.737455	56.860553	39.718558
6	Root Mean Square Error	23.328788	30.189543	22.136466
7	Number of Samples	31	31	12

Figure 27. Diagnostics results for LSR method.

The quantities are displayed in three columns:

- The **Input Matrix** column shows the diagnostic information using only the input matrix. For methods which go through the data points, such as HyperKriging or Radial Basis Functions, input matrix diagnostics are not useful;
- The **Cross-Validation Matrix** column shows the diagnostic information using a k-fold scheme, which means input data is broken into k groups. For each group, the group's data is used as a validation set for a new approximate model using only the other k-1 group's data. This allows for diagnostic information without the need of a testing matrix.
- The **Testing Matrix** column compares the approximate model, which was built using the input matrix, against a separate set of user supplied points. Using a testing matrix is the best method to get accurate diagnostic information.

There are several diagnostics criteria available. They are:

- **R-Square,** commonly called the coefficient of determination, is a measure of how well the Fit can reproduce known data points. It is the square of R; the linear correlation coefficient between the observed and the predicted values. It is the proportion in the variability in the data explained by the analysis of variance model. For example, if R-Square = 0.84, then 84% of the variance in the data is predictable by the Fit. R-Square is also the measure of the amount of reduction in the variability obtained by using the regression variables $x_1, x_2, \ldots, x_k$ in the model.

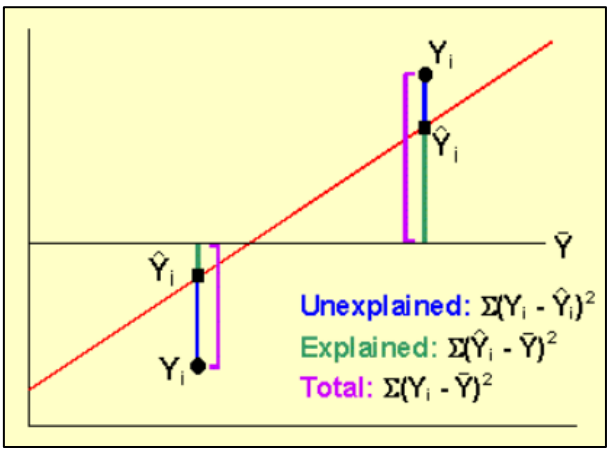


Figure 28. R-Square definition.

Using n to denote the number of samples, y_i the initial data, $\hat{y}_i$ the approximated function and $\overline{y}_i$ the mean value of the initial data, the R^2 is evaluated as follows:

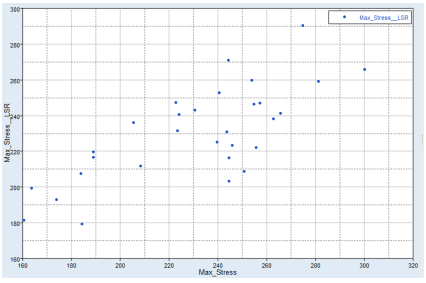
$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

R-Square is by definition between 0 and 1 and the closer to one, the better. However, if it is equal to 1.0, it should be used as an alert for further investigation since it is too good to be true for most practical applications. In practice a value above 0.92 is often pretty good. A value below 0.78 starts to get questionable and a value lower than 0.7 necessitate investigation using other metrics. There are some cases in which R-Square can be negative. A negative R-Square value indicates that the Fit quality is very poor and using the mean would be a better predictor than the Fit itself.

In the work area, R-Square values are presented with a spark line to indicate the relative value of the number (values typically vary between 0 and 1). Values are color coded based on the following:

- If R2 < 0.65 it is displayed **red**, which indicates the value is not good;
- If 0.8 < R2 < 0.995 it is displayed **green**, which indicates the value is good;
- Otherwise it is displayed **black**, which indicates that you should apply judgment when determining whether the value is or is not good.

Let's come back to our example. As a reminder, we have seen in the previous section that the Scatter was not good (Max_Stress_LSR vs. Max_Stress is not a diagonal line). In fact, it should alert you that R-Square will not be good neither as shown on the Figure 29 below.



	Criterion	Input Matrix	Cross-Validation Matrix
1	R-Square	0.5923174	0.3172670

Figure 29. Bad Scatter comes with low R-Square values.

A maximum possible value of 1.0 for R-Square could be obtained if the model perfectly predicts the known values. The Figure 30 presents an example of perfect Scatter, where the scatter points lie on a perfect diagonal line, and a perfect Plot where the simulation and fit functions are superimposed.

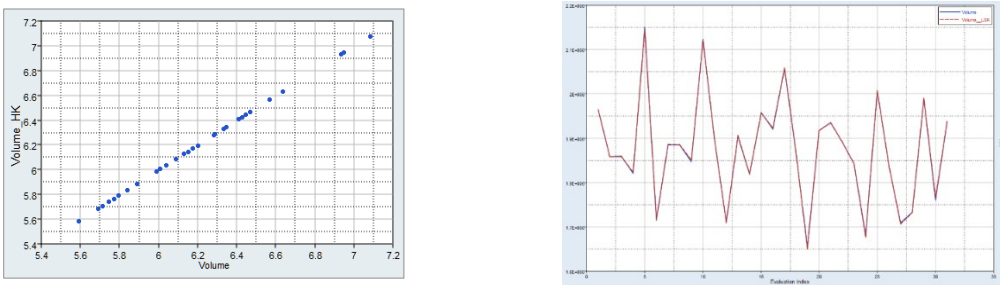


Figure 30. Example of perfect Scatter and Plot.

The value of R-Square decreases as errors increase and the scatter plot deviates more from a straight line.

So, R-Square is an important indication for the fit accuracy, but it doesn't tell us the entire story. Indeed, due to its formulation, adding a variable to the model will always increase R-Square. Hence you should not rely only on R-Square but also check the other metrics.

- **R-Square Adjusted** is a modification of R-Square that adjusts for the number of explanatory terms in a model. Unlike R-Square, the adjusted R-Square increases only if the new term improves the model more than would be expected by chance. If n is the number of points and p the number of variables, then R-Square Adjusted is defined as:

$$R^2_{adjusted} = 1 - \frac{n-1}{n-p-1}\left(1 - R^2\right)$$

The adjusted R^2 can be negative and will always be less than or equal to R^2. If both R^2 and R^2-adjusted are one, check again. When R^2 and R^2 adjusted differ dramatically, it indicates that non-significant terms may have been included in the model. R-Square adjusted is only available for LSR method, and it is not applicable on Cross-Validation and Testing matrices, N/A is displayed in the table.

- **Multiple R** is also available only for LSR. It is the multiple correlation coefficient between actual and predicted values, and in most cases, it is the square root of R-Square. It is an indication of the relationship between two variables.
- **Relative Average Absolute Error**

It represents the ratio of the average absolute error to the standard deviation: a comparison of the average predicted error and the spread in the data itself.

Relative Average Absolute Error is defined as:

$$\frac{\frac{1}{n}\sum_{i=1}^{n}\left[abs\left(y_i - \bar{y}_i\right)\right]}{\sqrt{\left(\frac{1}{n}\sum_{i=1}^{n}\left[y_i - \bar{y}\right]^2\right)}}$$

A low ratio is more desirable as it indicates that the variance in the Fit's predicted value are dominated by the actual variance in the data and not by modeling error.

- **Maximum Absolute Error**

The maximum difference, in absolute value, between the observed and predicted values. Maximum Absolute Error is defined as:

$$max\left(abs(y_i - \bar{y}_i)\right)$$

For the Input and Testing matrices, this value can also be observed in the **Residuals** tab.

	Criterion	Input Matrix	Cross-Validation Matrix
1	R-Square	0.5923174	0.3172670
2	R-Square Adjusted	0.4903968	N/A
3	Multiple R	0.7696216	0.5632646
4	Relative Average Absolute Error	0.5659899	0.7182214
5	Maximum Absolute Error	41.737455	56.860553
6	Root Mean Square Error	23.328788	30.189543
7	Number of Samples	31	31

Input Matrix Residuals

	Max_Stress	Max_Stress_LSR	Error
16	250.66675	208.92929	41.737455
21	244.48315	203.29786	41.185298
30	300.00272	266.03862	33.964093
15	255.62173	222.37753	33.244207

Figure 31. Maximum Absolute Error for Max_Stress fit.

- **Root Mean Square Error**

A measure of weighted average error. A higher quality Fit will have a lower value.

Root Mean Square Error is defined as:

$$\sqrt{\frac{\sum_{i=1}^{n}[y_i - \bar{y}_i]^2}{n}}$$

- **Number of Samples**

The number of data points used in the diagnostic computations.

Only for LSR, you will see in the Diagnostics tab an additional table as shown on Figure 32 providing the Regression Terms and the Regression Equation.

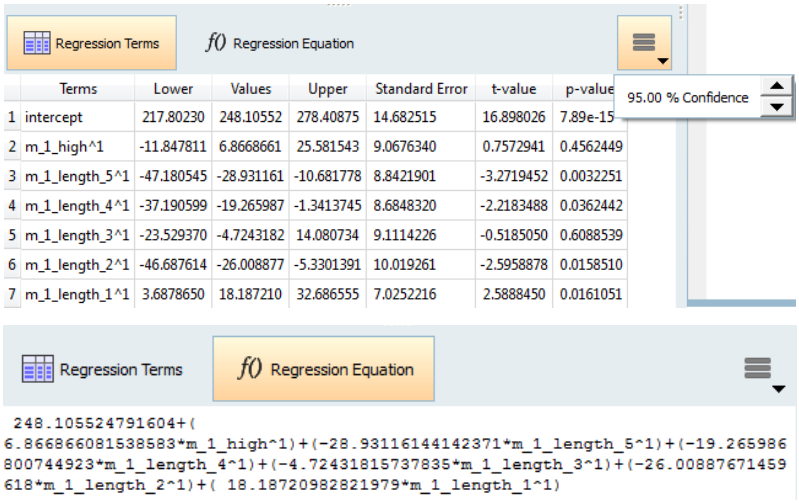


Figure 32. Regression Terms and Equation for LSR.

- **Confidence Intervals** (Least Squares Only)

Confidence intervals consist of an upper and lower bound on the coefficients of the regression equation. Bounds represent the confidence that the true value of the coefficient lies within the bounds, based on the given sample. A higher confidence value will result in wider bounds; a 95% confidence interval is typically used but you can change it as shown on the image above.

- **t-value** is a measure of the quality of a polynomial expression. t-value has the same sign as the regression coefficient. The larger the t-values, the more important the coefficient is for an accurate approximation. If the error is zero, its value in infinite. t-value is defined as:

$$t_j = \frac{\beta_j}{\sqrt{\sigma^2 c_{jj}}}$$

where *βj* is the corresponding regression coefficient (the **Values** column)

- **Standard error** is defined as: $SE = \sqrt{\sigma^2 c_{jj}}$ $\sigma^2 = \frac{\sum_{i=1}^{n}[y_i - \bar{y}_i]^2}{n-p}$ and where C_{jj} is the diagonal coefficient of the information matrix used during the regression calculation.

p-values are computed using the standard error and t-value to perform a student's t-test. The p-value indicates the statistical probability that the quantity in the **Value** column could have resulted from a random sample and that the real value of the coefficient is actually zero (the null hypothesis). A low value, typically less than 0.05, leads to a rejection of the null-hypothesis, meaning the term is statistically significant.

8.3.3 Residuals

Residuals help to identify errors for each design. In the Residuals tab, the error (and percentage) between the original response and the approximation is listed for each run of the Input, Cross-Validation, or Testing matrices.

The Figure 33 presents the Residuals results for Max_Stress fit. The Error column provides (Max_Stress–Max_Stress_LSR) values; and Percent error provides 100*(Max_Stress–Max_Stress_LSR)/Max_Stress error values. It can be seen that the difference between the simulated and the predicted values is quite high, which confirms the poor fit quality. You can use the Find and Sort options in the right-click context menu to order the values or to search for specific case

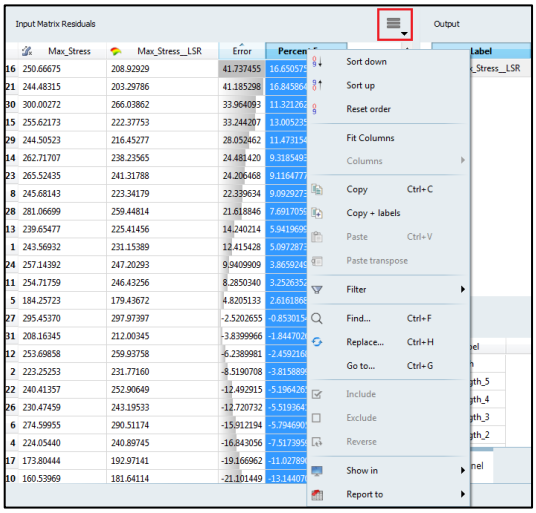


Figure 33. Residuals for Max_Stress fit with LSR.

In order to display the Residuals for the different matrices you have to click on the button as shown on Figure 34.

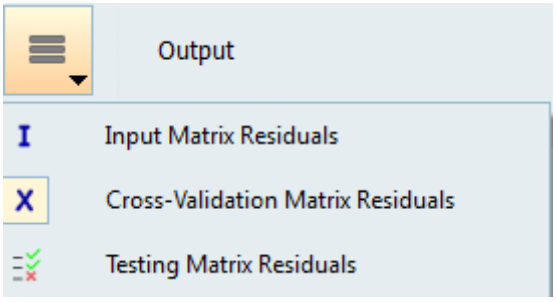


Figure 34. Matrix selection button.

The Residuals of Max_Stress fit evaluated on the Testing matrix (Figure 35) are also very high.

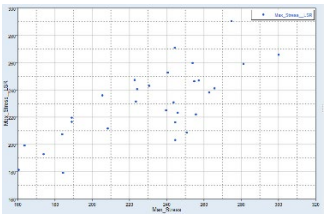
Validation Matrix Residuals

	Max_Stress	Max_Stress__LSR	Error	Percent Error
7	296.76755	257.04899	39.718558	13.383727
2	289.78741	260.45807	29.329348	10.120987
1	300.21008	272.47617	27.733909	9.2381670
3	254.96515	248.43996	6.5251893	2.5592475
11	243.85573	247.19009	-3.3343603	-1.3673496
10	213.13441	220.99467	-7.8602534	-3.6879325
4	225.74899	236.42185	-10.672859	-4.7277551

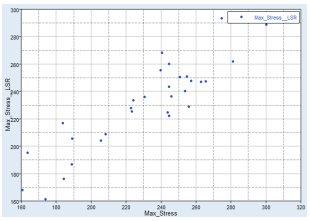
Figure 35. Max_Stress fit Residuals evaluated on Testing matrix.

Hence, this approximation could not be used efficiently in further studies, we need first to improve it.

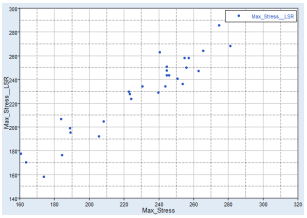
To do that, we start by trying another Regression model. The Figures below compare the Scatters, Diagnostics and Residuals for Max_Stress fit obtained with different LSR regression models.



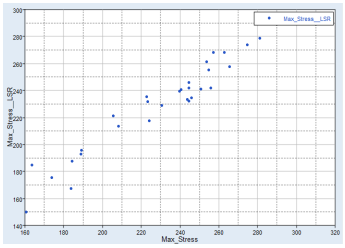
Linear



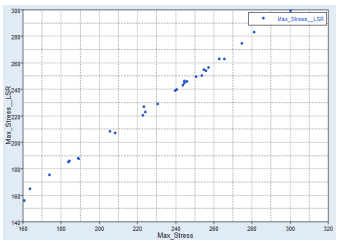
Squared



Cubic



Interaction



Full quadratic

Figure 36. Max_Stress_LSR vs Max_Stress scatters with different LSR regression models.

Regression model	R-Square
Linear	Criterion / Input Matrix / Cross-Validation Matrix / Validation Matrix 1 R-Square / 0.5923174 / 0.3172670 / 0.6509026
Squared	Criterion / **Input Matrix** / Cross-Validation Matrix / Validation Matrix 1 R-Square / 0.8191421 / 0.3024808 / 0.8318795
Cubic	Criterion / **Input Matrix** / Cross-Validation Matrix / Validation Matrix 1 R-Square / 0.9188500 / 0.1366401 / 0.6941274
Interaction	Criterion / **Input Matrix** / Cross-Validation Matrix / Validation Matrix 1 R-Square / 0.9406292 / -1.0505765 / 0.2235953
Full quadratic	Criterion / **Input Matrix** / Cross-Validation Matrix / Validation Matrix 1 R-Square / 0.9977848 / -0.2127554 / 0.4227944

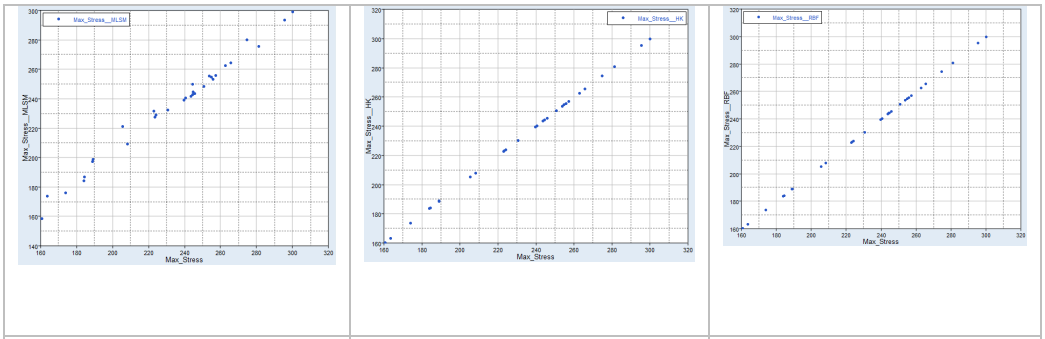
Figure 37. R-Square with different LSR regression models.

Regression model	Max error Input matrix (%)	Max error Testing matrix (%)
Linear	-22	-17
Squared	-18	-14
Cubic	-12	21
Interaction	-13	30
Full quadratic	2	31

Figure 38. Input and Testing matrices Residuals with different LSR regression models.

On Figure 36 one can see that Interaction and Full quadratic models provide better scatters, but the Diagnostics on the Cross-Validation and Testing matrices (Figure 37) and the Residuals on the Testing matrix (Figure 38) are not good with any of the regression models.

Hence, the next step is to try with another approximation method: MLSM, HK or RBF. The Figures below compare the Scatters, Diagnostics and Residuals for Max_Stress fit obtained with different approximation methods.



MLSM	HK	RBF

Figure 39. Max_Stress_LSR vs Max_Stress scatters with different approximation methods.

Approximation Method	R-Square
LSR (Linear)	Criterion / Input Matrix / Cross-Validation Matrix / Validation Matrix 1 R-Square: 0.5923174 / 0.3172670 / 0.6509026
MLSM	Criterion / Input Matrix / Cross-Validation Matrix / Validation Matrix 1 R-Square: 0.9804054 / 0.5464190 / 0.7491009
HK	Criterion / Input Matrix / Cross-Validation Matrix / Validation Matrix 1 R-Square: 0.9999998 / 0.5577041 / -1.5342734
RBF	Criterion / Input Matrix / Cross-Validation Matrix / Validation Matrix 1 R-Square: 1.0000000 / 0.7703622 / 0.8938733

Figure 40. R-Square with different approximation methods.

Approximation Method	Max error Input matrix (%)	Max error Testing matrix (%)
LSR (Linear)	-22	-17
MLSM	-7	-15
HK	-	55
RBF	-	12

Figure 41. Input and Testing matrices Residuals with different approximation methods.

The MLSM scatter looks better than the LSR one. HK and RBF scatters provide perfect diagonal lines, but remember that HK and RBF go through the exact points. It results in a perfect scatter and high R-Square (Input Matrix) values, but these results are misleading. In case of HK and RBF, you should check R-Square (Testing Matrix) and Residuals (Testing matrix). Following this statement, the most appropriate method seems to be RBF: R-Square (Testing matrix) = 0.89; Max error (Testing matrix) = 12%. Nevertheless, 12% of error is still high.

At this stage you may want to investigate the Max_Stress behavior in order to figure out what the reason of the poor fit quality is. For that, let's look at the simulation results for Max_Stress. The Figure 42 displays Max_Stress values on the arm model. We can see that maximum stress values are located in different parts of the model, on the top and on the bottom.

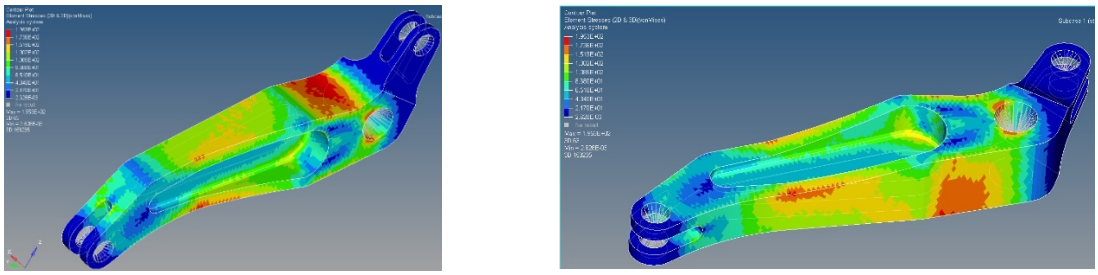


Figure 42. Arm model Max_Stress color-shades maps.

In fact, Max_Stress is a discontinuous function (a global envelope of localized effects). The nature of these envelope type of responses makes them difficult to capture accurately with a Fit. A recommendation to proceed in this situation would be to either increase the number of samples, which is not guaranteed to improve the accuracy, or to create a series of more localized responses that would be simpler functions of the variables – for example several responses that each capture the stress in specific regions. The locations of the high stresses are highlighted on Figure 43.

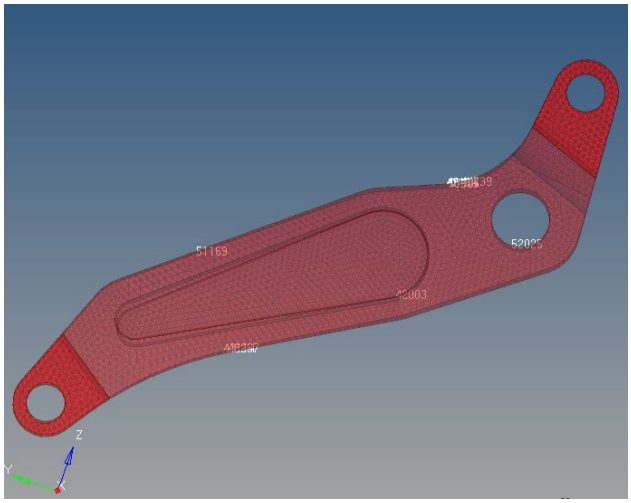


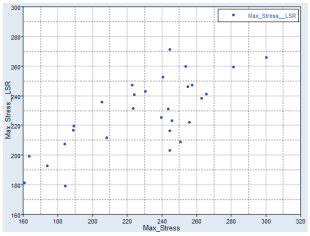
Figure 43. High stress locations in the studied arm model.

In HyperStudy, we create five additional stress responses as shown on Figure 44. Each of these responses correspond to one of the five areas of high stress. You can see in the table below that Max_Stress and Max_Stress_1 have the same values because for this design Max_Stress_1 is the maximum, but for other designs it could be Max_Stress_2, Max_Stress_3, …

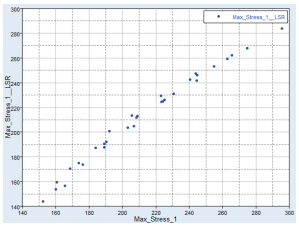
	Active	Label	Value
1	☑	Max_Disp	1.4108266
2	☑	Volume	1766760.0
3	☑	Max_Stress	195.29446
4	☑	Max_Stress_1	195.29446
5	☑	Max_Stress_2	178.35742
6	☑	Max_Stress_3	187.57291
7	☑	Max_Stress_4	110.87536
8	☑	Max_Stress_5	169.31192

Figure 44. Additional Max_Stress responses created in HyperStudy.

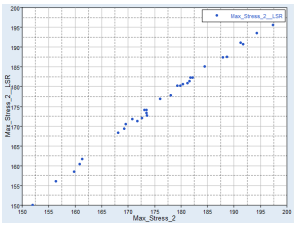
The goal is to build fitting functions on these additional responses and compare the results with those obtained on Max_Stress. The Figure 45 displays the scatters for all Max_Stress responses obtained with LSR (Linear Regression Model).



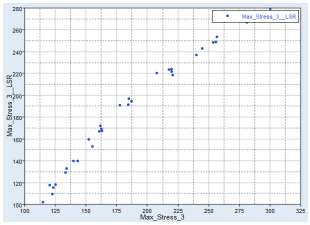
Max_Stress (LSR-Linear)



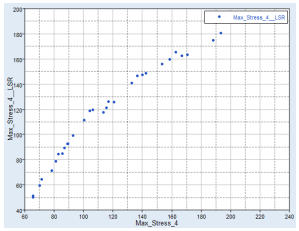
Max_Stress_1 (LSR-Linear)



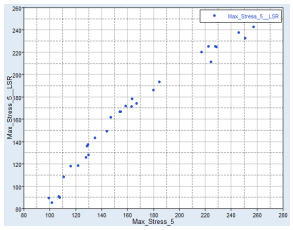
Max_Stress_2 (LSR-Linear)



Max_Stress_3 (LSR-Linear)



Max_Stress_4 (LSR-Linear)



Max_Stress_5 (LSR-Linear)

Figure 45. Max_Stress functions scatters for LSR (Linear).

It can be seen that scatters for Max_Stress_1 to 3 are pretty good diagonal lines; Max_Stress_4 and 5 scatters are slightly less good. But all Max_Stress_1 to 5 scatters are better than Max_Stress scatter.

The Figure 46 presents a comparison of the R-Square values for all Max_Stress responses.

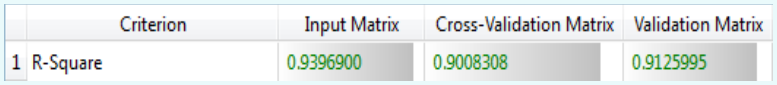
Function	R-Square
Max_Stress	
Max_Stress_1	
Max_Stress_2	
Max_Stress_3	
Max_Stress_4	
Max_Stress_5	

Figure 46. R-Square (LSR-Linear) comparison for all Max_Stress functions.

Contrary to Max_Stress, which have low R-Square values, the values for Max_Stress_1 to 5 are very good for both Input, Cross-Validation and Testing matrices. The Figure 47 presents a comparison of the percent errors for all Max_Stress responses.

Function	Max error Input matrix (%)	Max error Testing matrix (%)
Max_Stress	-22	-17
Max_Stress_1	5	-4
Max_Stress_2	1	1
Max_Stress_3	10	12
Max_Stress_4	23	16
Max_Stress_5	16	12

Figure 47. Residuals comparison for all Max_Stress functions.

It can be seen that Max_Stress_1, 2 and 3 errors are much smaller Max_Stress errors; Max_Stress_4 and Max_Stress_5 errors remain more important, which is obvious regarding the scatter plots.

So, using LSR with localized Max_Stress functions allows improving significantly the approximation quality for most of the responses. But the quality is not completely satisfactory for Max_Stress_4 and 5. Hence, it might be interesting to try the other approximation methods.

The Table 13 presents a global overview of the Residuals obtained with all approximation methods and all Max_Stress responses. MLSM and RBF allow reducing significantly the errors in comparison with LSR. In conclusion, one of these methods is the most suitable in this specific case.

Approximation method	Max error Input matrix (%)	Max error Testing matrix (%)
LSR (Linear)	Max_Stress: -22 Max_Stress_1: 5 Max_Stress_2: 1 Max_Stress_3: 10 Max_Stress_4: 23 Max_Stress_5: 16	Max_Stress: -17 Max_Stress_1: -4 Max_Stress_2: 1 Max_Stress_3: 12 Max_Stress_4: 16 Max_Stress_5: 12
MLSM	Max_Stress: -7 Max_Stress_1: **-2** Max_Stress_2: **-0.5** Max_Stress_3: **-3** Max_Stress_4: **-4** Max_Stress_5: **-5**	Max_Stress: -15 Max_Stress_1: **4** Max_Stress_2: **1.5** Max_Stress_3: **-9** Max_Stress_4: **13** Max_Stress_5: **-12**
HK	-	Max_Stress: 55 Max_Stress_1: -7.6 Max_Stress_2: 2.3 Max_Stress_3: 24 Max_Stress_4: -35 Max_Stress_5: -52
RBF	-	Max_Stress: 12 Max_Stress_1: **-3** Max_Stress_2: **1.8** Max_Stress_3: **10** Max_Stress_4: **14** Max_Stress_5: **11**

Table 13. Global Residuals overview.

We will cover now two additional post-processing tools, which are Trade-off and ANOVA.

8.3.4 Trade-Off

Trade-Off is a tool that allows performing “What if” studies in three different manners: Trade-off 1D; Trade-Off 2D and Trade-Off 3D. Note that Trade-off results can be exported to Excel.

- Trade-Off 1D

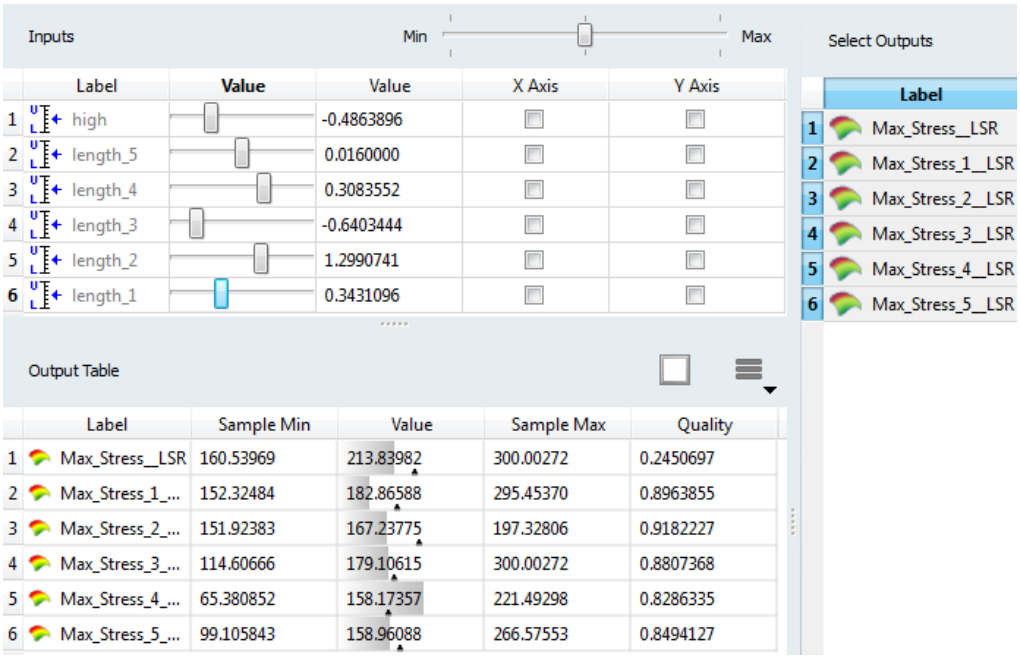


Figure 48. Trade-off 1D.

In the **Inputs** panel, you can modify the values of input variables to see their effect on the output response approximations in the **Output** panel. You can either enter manually a specific value or use the slider to modify each input variable. To set input variables to their initial, minimum, or maximum values, you can move the slider in the upper right-hand corner of the Inputs frame.

In the **Channel** panel, you can select the desired output responses to display.

For the given values of the input variables, the output responses’ predictions are calculated by the Fit, and displayed in the Output panel. Table shading is used to indicate the output response’s value between the minimum and maximum values contained in the input design matrix. When shading extends into either the **Sample Min** or **Sample Max** column, this indicates that the predicted value is beyond the bounds contained in the input matrix. If the shading extends significantly into these regions, it is suggested that you asses the validity of this value based on experience and knowledge of the modeled problem.

The **Quality** column is used as a normalized measure to assess the trust in the Fit at a specific point in the design space. The values in the Quality column run between 0 and 1. A value of 1 indicates that the predicted point lies in a portion of the variable space bounded by the known points and seems trustworthy. As this number decreases, the prediction at this point becomes less reliable, partially due to the values increasing based on an extrapolation of the data.

In the Trade-Off tab you can also plot the effect of an input variable on output response approximations. Select an input variable to plot by enabling its corresponding **X Axis** and/or **Y Axis** checkbox in the Inputs pane. Use the Channel selector to select output responses to plot.

The values for the input variables which are not plotted can be modified in the Inputs pane. Move the sliders in the Value column to modify the other input variables, while studying the output response throughout the design space.

- Trade-Off 2D

Enabling the X Axis checkbox creates a 2D trade-off in the Outputs panel.

In a 2D trade-off the metrics shown in the Quality column can be plotted alongside the output response curve by selecting ***Fit Quality*** from the menu that displays when you click as shown on the image below.

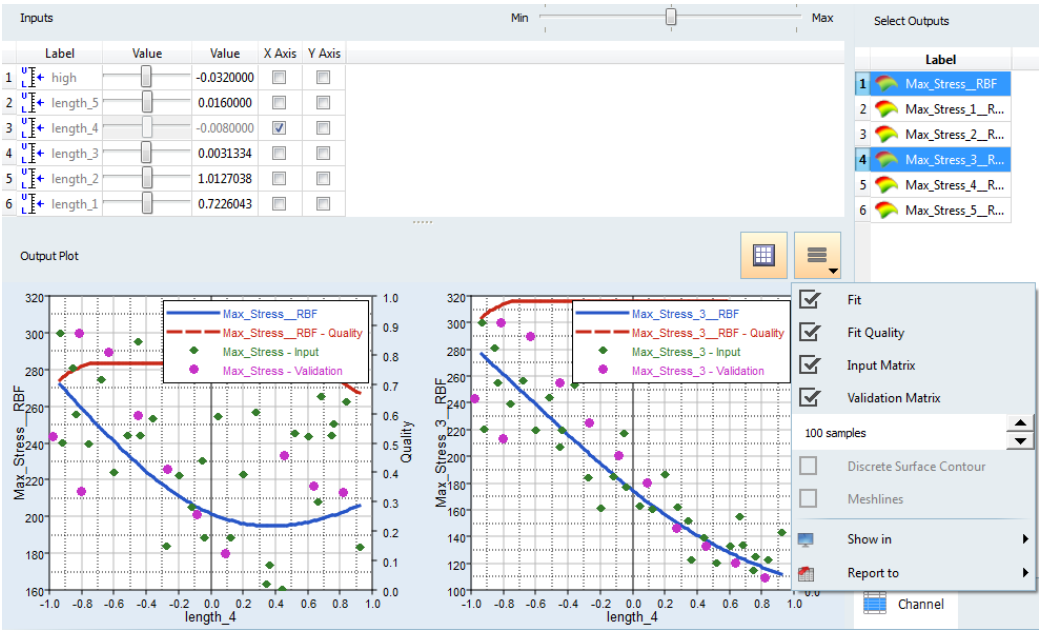


Figure 49. Trade-off 2D.

- Trade-Off 3D

Enabling both the X Axis and Y Axis checkboxes creates a 3D trade-off in the Outputs panel.

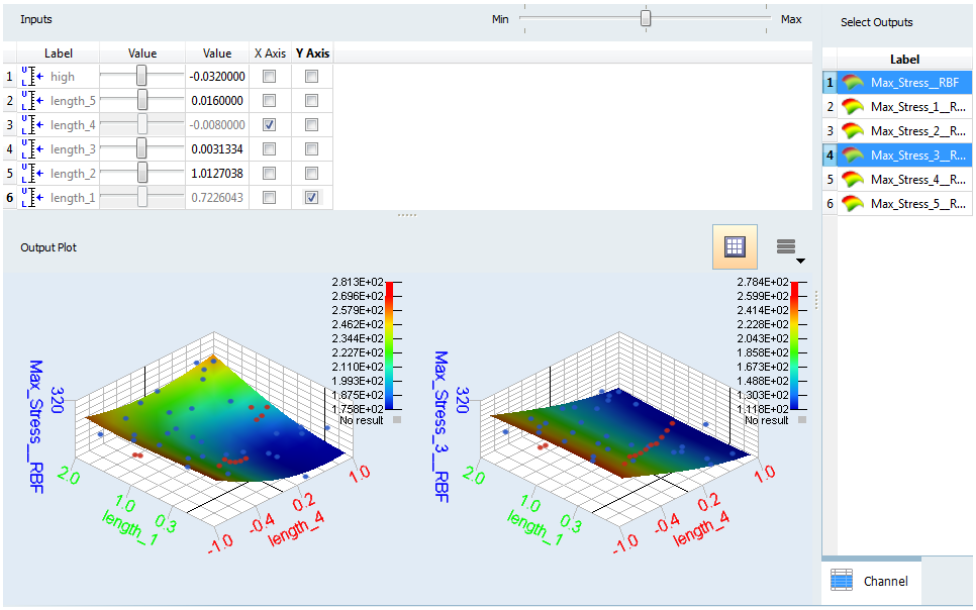


Figure 50. Trade-off 3D.

8.3.5 ANOVA

ANOVA (Analysis of Variance) is a technique used to estimate the error variance and determine the relative importance of various factors.

In HyperStudy, an ANOVA can only be performed on LSR approximations. ANOVA is used to identify which variables are explaining the variance in the data. This is done by examining the resulting increase in the unexplained error when variables are removed.

The following definitions are useful in describing ANOVA concepts. For a given set of n input values, denoted as y_i, the Fit predictions at the same points are denoted as $\bar{y}_i$. The mean of the input values is expressed $\bar{y}$. For a Least Squares Regression, *p* is the number of unknown coefficients in the regression.

The following three values are defined as follows:

Total sum of Squares:	$SS_{tot} = \sum_{i=1}^{n} [y_i - \bar{y}]^2$
Explained Sum of Square:	$SS_{exp} = \sum_{i=1}^{n} [\bar{y}_i - \bar{y}]^2$
Residual Sum of Squares:	$SS_{err} = \sum_{i=1}^{n} [y_i - \bar{y}_i]^2$
Average Absolute Error:	$\frac{1}{n}\sum_{i=1}^{n} [abs(y_i - \bar{y}_i)]$
Standard deviation:	$\sqrt{\left(\frac{1}{n}\sum_{i=1}^{n} [y_i - \bar{y}_i]^2\right)}$

In the ANOVA tab, the following quantities are computed in the process of performing the ANOVA.

	Variables	Degrees of Freedom	Sum of Squares	Mean Squares	Mean Squares Percent	F-value	p-value
1	high	1	576.27721	576.27721	1.5095147	20.456112	1.40e-04
2	length_5	1	7060.1029	7060.1029	18.493407	250.61247	3.34e-14
3	length_4	1	44.975074	44.975074	0.1178088	1.5964801	0.2185390
4	length_3	1	74.002496	74.002496	0.1938440	2.6268665	0.1181351
5	length_2	1	3364.0851	3364.0851	8.8119670	119.41492	8.46e-11
6	length_1	1	27028.710	27028.710	70.799666	959.43809	7.27e-21
7	Error	24	676.11349	28.171396	0.0737928	N/A	N/A
8	Total	30	41610.232	N/A	100.00000	N/A	N/A

	Label
1	Max_Stress__LSR
2	Max_Stress_1__LSR
3	Max_Stress_2__LSR
4	Max_Stress_3__LSR
5	Max_Stress_4__LSR
6	Max_Stress_5__LSR

Figure 51. Anova analysis.

- **Degrees of Freedom**

Number of terms in the regression associated with the variable. All degrees of freedom not associated with a variable are retained in the Error assessment. More degrees of freedom associated with the error increases the statistical certainty of the results: the p-values. Higher order terms have more degrees of freedom; for example, a second order polynomial will have two degrees of freedom for a variable: one for both the linear and quadratic terms.

- **Sum of Squares**

For each **Variable** row, the quantity shown is the increase in unexplained variance if the variable were to be removed from the regression. A variable which has a small value is less critical in explaining the data variance than a variable which has a larger value; The **Error** row entry represents the variance not explained by the model, which is SS_{err}.The entry in the **Total** row, which is SS_{tot}, will generally not equal to the sum of the other rows.

- **Mean Squares**

The ratio between unexplained error increase and degrees of freedom, computed as the Sum of Squares divided by the associated degrees of freedom.

- **Mean Squares Percent**

This quantity is interpreted as the relative contribution of the variables to the Fit quality, computed as the ratio of the Mean Square to the summed total of the Mean Squares. A variable with a higher percentage is more critical to explaining the variance in the given data than a variable with a lower percentage.

- **F-value**

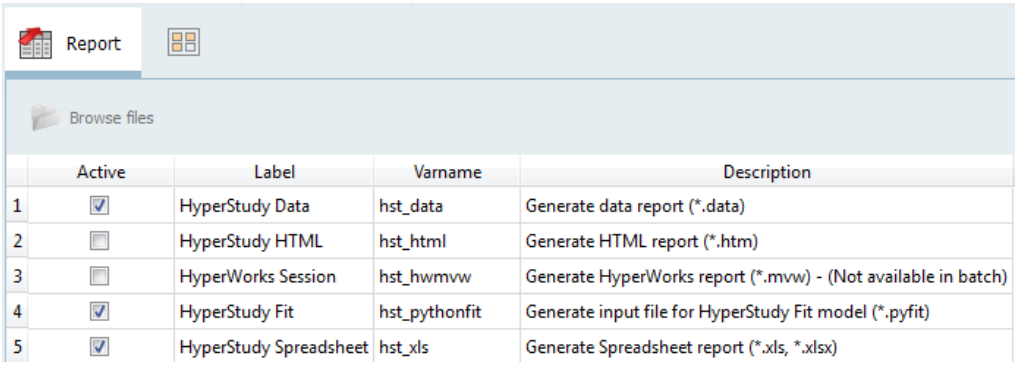
The quotient of the mean squares from the variable to the mean squares from the error. This is a relative measure of the variable's explanatory variance to overall unexplained variance.

- **p-value**

The result of an F-test on the corresponding F-value. The p-value indicates the statistical probability that the same pattern of relative variable importance could have resulted from a random sample and that the variable actually has no effect at all (the null hypothesis). A low value, typically less than 0.05, leads to a rejection of the null-hypothesis, meaning the variable is statistically significant.

8.3.6 Report

In the Report section you will find five different report types as shown on Figure 52.



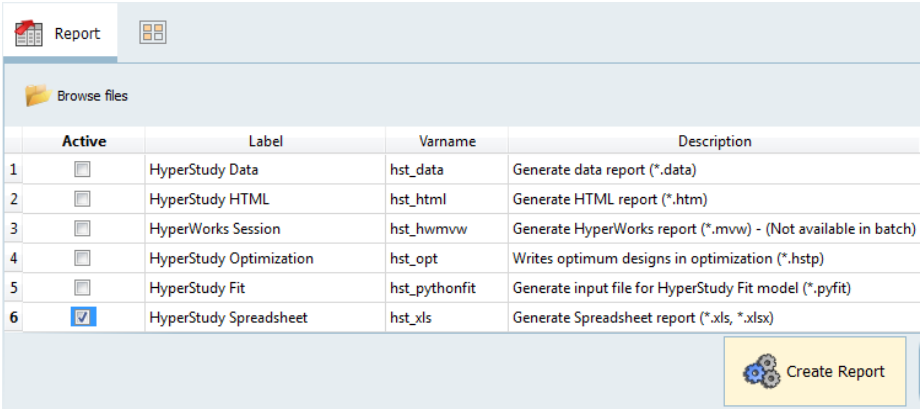
	Active	Label	Varname	Description
1	☑	HyperStudy Data	hst_data	Generate data report (*.data)
2	☐	HyperStudy HTML	hst_html	Generate HTML report (*.htm)
3	☐	HyperWorks Session	hst_hwmvw	Generate HyperWorks report (*.mvw) - (Not available in batch)
4	☑	HyperStudy Fit	hst_pythonfit	Generate input file for HyperStudy Fit model (*.pyfit)
5	☑	HyperStudy Spreadsheet	hst_xls	Generate Spreadsheet report (*.xls, *.xlsx)

Figure 52. Report types available.

In order to generate a desired report, you have to check the Active column and then click on Create Report. All the reports created are available in the report directory in the study working directory.

The most valuable report for Fit is the spreadsheet report shown on the image below. In this excel file you can find a Trade-off tab allowing to compute output values in function of input parameters according to the fit approximation built by HyperStudy. It's a kind of portable version of the response surfaces created by HyperStudy. It allows analysts to share results with non-HyperStudy such as designers or field personnel.

Once you have clicked on "Create Report" the excel report is generated and is automatically displayed on the screen. You can experiment to change the input values in order to see what the output values would be.



	Active	Label	Varname	Description
1	☐	HyperStudy Data	hst_data	Generate data report (*.data)
2	☐	HyperStudy HTML	hst_html	Generate HTML report (*.htm)
3	☐	HyperWorks Session	hst_hwmvw	Generate HyperWorks report (*.mvw) - (Not available in batch)
4	☐	HyperStudy Optimization	hst_opt	Writes optimum designs in optimization (*.hstp)
5	☐	HyperStudy Fit	hst_pythonfit	Generate input file for HyperStudy Fit model (*.pyfit)
6	☑	HyperStudy Spreadsheet	hst_xls	Generate Spreadsheet report (*.xls, *.xlsx)

Figure 53. Spreadsheet report generation.

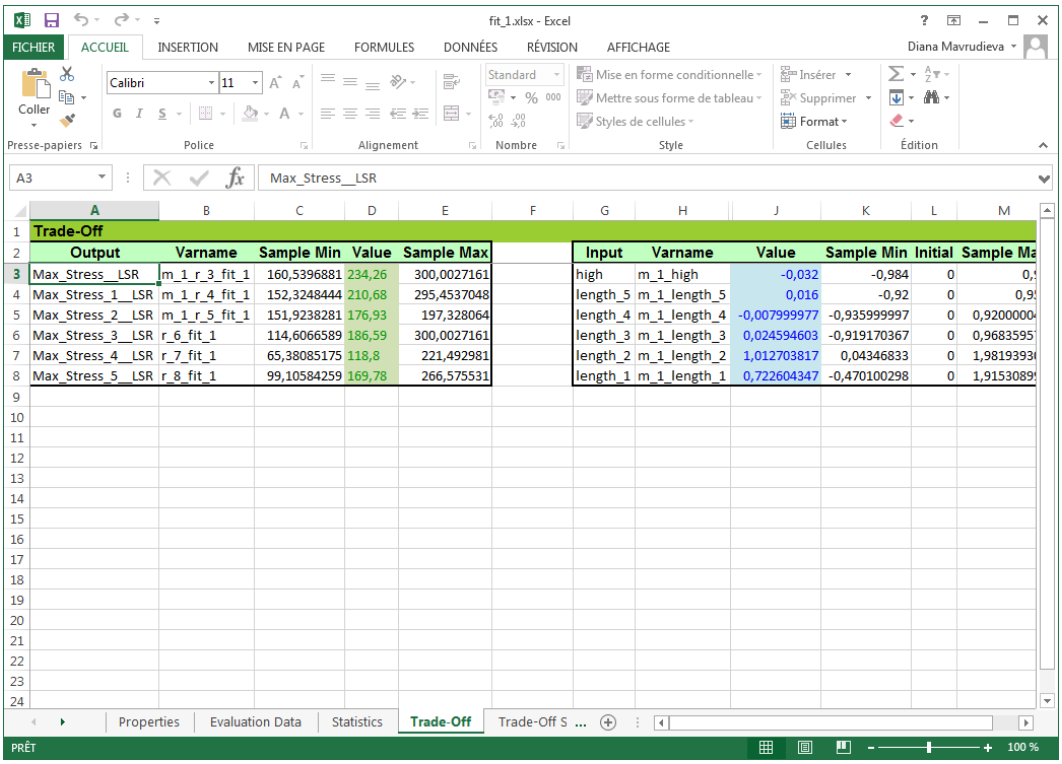


Figure 54. Excel sheet for trade-off studies.

Warning: You need to install a dedicated plug-in in order to access the HyperStudy fitting algorithms and makes the excel report usable.

The first time when you generate a spreadsheet report, you will see in the Messages window the following message:

44 Warning: The excel add-in for Fit is not installed. Install instructions are in file

(C:/Altair\hst\plugins\externals\hstfitaddin\hstfitaddin_readme.txt)

In order to enable the HyperStudy Fit addin for Microsoft windows MS/Excel 1), you need to do the following:

- Go in <ALTAIR_HOME>/hst/plugins/externals/hstfitaddin/
- Double click on hstfitaddin_install.vbs

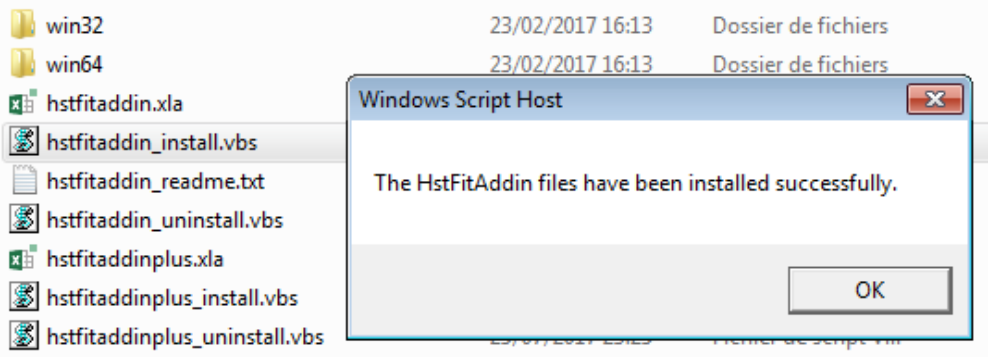
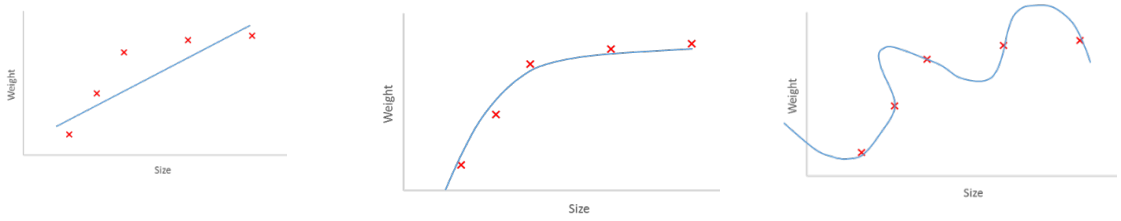


Figure 55. HstFitAddin installation.

9 Overfitting Introduction

Overfitting is an important and complex concept especially in Machine learning.

To introduce what's overfitting is let's look at the pictures below (Figure 56).



High bias (underfit) "Just right" High variance (overfit)

Figure 56. Bias/Variance trade-off.

These pictures illustrate the use of three different models to fit the same data set. The red marks represent the known dataset (Input matrix in HyperStudy Fit approach) used to build the fit, it is also called to train the model.

- The use of a simple regression model in this case (picture on the left) does not allow capturing well enough the pattern; the model is "underfitted";
- The use of "too complex" approximation model (picture on the right) could lead to overfitting. Concretely, with this "overfitted" model outputs for the known dataset will be predicted with high accuracy (small errors when checking Residuals on the Input matrix because the model goes through the exact points), but it may therefore fail to predict reliably unknown dataset (higher errors when checking Residuals on the Testing matrix). It emphasizes the utility to use Testing matrix to access the fit quality on dataset unused for the fit building and detect eventually a possible overfitting;
- The "right" model (picture in the middle) allows generalizing the trends from the input data without overfitting and hence be likely to predict accurately unknown dataset (small errors when checking Residuals on the Testing matrix).

Overfitting could occur in the following situations:

- Not enough relevant information (not enough or poorly distributed sampling points) ;
- Input data containing lot of noise. The noise refers to the irrelevant information or randomness of a dataset. Using a "too complex" model to build a fit from such input data could lead to overfitted model "memorizing more the noise" rather than the signal. Such a model could then make predictions based on that noise (Figure 57).

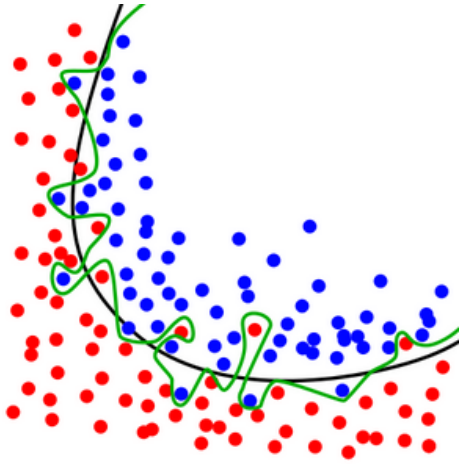


Figure 57. Red marks: relevant input data; Blue marks: Noise; Black line: “Right” model capturing the trends; Green line: “Overfitted” model.

In general, simplest the model, the better it is. Up until a certain number of degree of polynomial, new terms improve the model and both Testing and Input matrix errors decrease. If a model becomes « too complex », its ability to generalize can weaken as it begins to overfit the input data (Input matrix errors continue decreasing while Test errors increase significantly). By identifying that point it is possible to stop the building earlier and avoid overfitting (Figure 58).

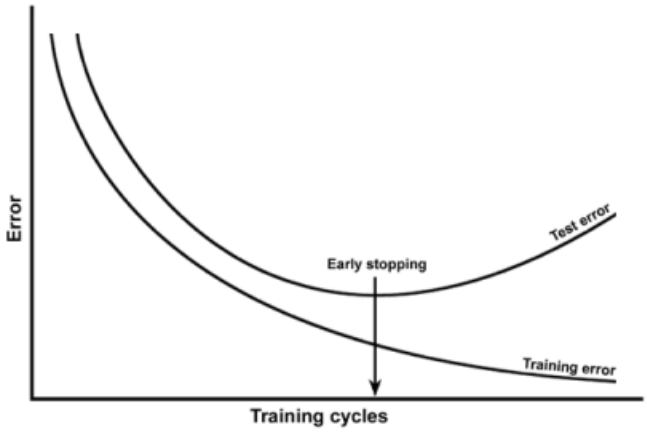


Figure 58. Early stopping.

10 Introduction to Machine Learning

Machine Learning (ML) is a very popular topic with many applications in all domains. There is tremendous material on it in the form of books, courses, videos, etc. The goal of this section is to introduce ML, relate it to engineering applications and demonstrate the use of machine learning on them using an engine life prediction example very briefly.

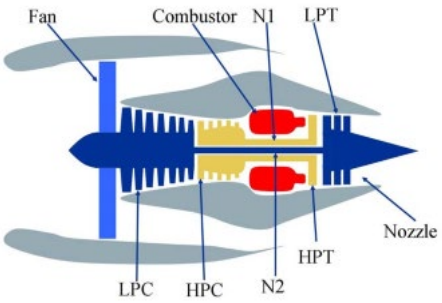
ML is first defined by computer scientist Arthur Samuel as “a field of study that gives computers the ability to learn without being explicitly programmed”. It differs from traditional, rule-based programming as its performance is entirely a function of the data it learns from as opposed to traditional programming where a set of discrete rules governing the logic of the application is programmed. As a result, predictive models from machine learning continuously improves with data and they are not degraded by simplifications done to be able to write rule-based codes. ML can use field data or simulation data. It can be supervised or unsupervised. Supervised machine learning can be of type regression or classification whereas unsupervised machine learning is of clustering type. Some of the most popular methods in these three types of machine learning are neural networks, decision trees and k-means clustering. HyperStudy Fit methods are of regression type ML.

Engineering design and operation decisions are largely dependent on engineers understanding of the application. This includes assumptions made to simplify the problem in order to solve it. However, these assumptions may introduce errors compared to actual behavior of the application. Increasing access to data; sensor or virtual; and computational resources combined with democratization of advanced machine learning algorithms; the use of ML for engineering applications is bringing field data and engineering knowledge together, thus leading to an increased level of accuracy in decision-making and improved performance.

The ML application example below deals with Remaining Useful Life (RUL) Prediction of an Aircraft Engine. This problem and the data set is from NASA Prognostics Center of Excellence Data Repository:

(https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/)

The dataset that is used has more than 20 thousand rows of 21 sensor data from 100 engines that have run until failure under the same operating conditions. The challenge in this problem is to use this historical dataset to predict the Remaining Useful Life (RUL) of the engines that are currently in operation.



Symbol	*Description*	*Units*
Parameters available to participants as sensor data		
T2	Total temperature at fan inlet	°R
T24	Total temperature at LPC outlet	°R
T30	Total temperature at HPC outlet	°R
T50	Total temperature at LPT outlet	°R
P2	Pressure at fan inlet	psia
P15	Total pressure in bypass-duct	psia
P30	Total pressure at HPC outlet	psia
Nf	Physical fan speed	rpm
Nc	Physical core speed	rpm
epr	Engine pressure ratio (P50/P2)	--
Ps30	Static pressure at HPC outlet	psia
phi	Ratio of fuel flow to Ps30	pps/psi
NRf	Corrected fan speed	rpm
NRc	Corrected core speed	rpm
BPR	Bypass Ratio	--
farB	Burner fuel-air ratio	--
htBleed	Bleed Enthalpy	--

Figure 59. Aircraft Engine and sensors definition.

Each engine starts with different degrees of initial wear and manufacturing variation, which is unknown to the user. This wear and variation are considered normal, i.e., it is not considered a fault condition. The engine is operating normally at the start of each time series and develops a fault at some point during the series. In the training set, the fault grows in magnitude until system failure. In the test set, the time series ends some time prior to system failure. The objective of the competition is to predict the number of

remaining operational cycles before failure in the test set, i.e., the number of operational cycles after the last cycle that the engine will continue to operate.

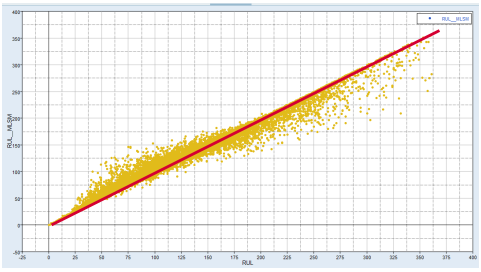
The data is provided in a csv file with 26 columns of numbers. Each row is a snapshot of data taken during a single operational cycle; each column is a different variable. The columns correspond to 1) unit number; 2) time, in cycles; 3) operational setting 1; 4) operational setting 2; 5) operational setting 3; 6) sensor measurement 1; 7) sensor measurement 2; ... 26) sensor measurement 26.

The machine learning steps are:

Step	Purpose
1 – Problem Formulation	Define and compute Remaining Useful Life (RUL)
2 – Data Analysis	Review the data Eliminate constant values, review outliers, review relations
3 - Predictive Modelling in HyperStudy	Start from a simple model; then move on to more complex models. Review diagnostics and pick the best predictive model.
4 - Predict	Use the predictive model to predict the next engine RUL. Approximate the prediction.

The table below presents 7 engine predictions. As you can see RUL predictions for those who have not went through enough cycles for data patterns to emerge are not accurate. However, once the engine goes through about half the cycles that they are designed for, patterns start to emerge, and predictions become very accurate.

Engine No.	Current Cycle	Actual RUL (cycles)	Predicted RUL (cycles)
101	31	112	171
102	49	98	139
103	126	69	65
104	106	82	81
124	186	20	19
125	48	145	160
134	395	7	11



Actual vs Predicted RUL

Figure 60. Prediction summary.

11 Best Practices

In order to help you using on the best manner Fit with HyperStudy, let's answer once again some questions.

1. Which DOE Method should I use as Input Matrix for Fit?

From the list of space filling DOE available in HyperStudy the recommended Top 3 chart is:

- 1st choice: **MELS**

You can either keep the default number of runs (not recommended to decrease it) or increase it. If you have runs from another approach it's recommended to leverage it as Inclusion matrix. The best performance can be expected when the inclusion is an existing data set from a MELS DOE

- 2nd choice: **Hammersley**

It allows distributing the points evenly within the design space and it is well suited to fit highly nonlinear response surface. Same as for MELS, you can leverage available runs as inclusion.

- 3rd choice: Latin Hypercube.

It is also well suited for fitting highly nonlinear response surface but it is a less good space filler. You can also leverage in data from another DOE if available.

Method	Space filling quality	Suited for highly nonlinear response surface	Inclusion matrix
MELS	★★★	✓	✓
Hammersley	★★	✓	✓
Latin Hypercube	★	✓	✓

2. Which DOE Method should I use as Testing Matrix for Fit?

The recommended Top 3 chart is the same as for Input matrix. You can use MELS, Hammersley or Latin Hypercube as Testing matrices, but with smaller number of runs.

Note that if you have used MELS as Input matrix, and you would like to use another MELS as Testing matrix, you should be aware that the resulting Testing matrix could be a subset of the MELS based Input matrix due to the extensible property of MELS.

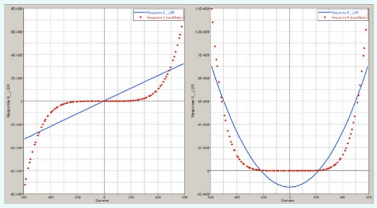
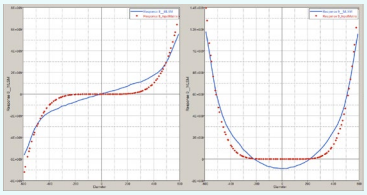
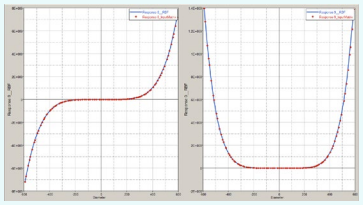
Ex. For instance if you create a MELS_1 with 40 runs and create another one MELS_2 with 4 runs; the 4 runs of MELS_2 would be the same designs as the first four designs of MELS_1. As a result, if you use MELS_1 as an Input matrix and MELS_2 as a Testing matrix to Fit, you are essentially validating against the same runs as you are creating the fit and hence not really validating it.

To prevent this from happening, you have to change the Random Seed setting of the Testing matrix to be a number larger than the number of runs in the Input data before building it.

Ex. MELS_1 (Input matrix): 40 runs, Random seed=1; MELS_2 (Testing matrix): 4 runs, Random seed=41.

3. Which Approximation Method should I use?

FAST is the recommended choice as it allows building automatically best fitting functions by testing all the methods below and their settings:

Method	Equation	Application	Behavior
Least Squares Regression (LSR)	Available	Linear or low nonlinear functions	
Moving Least Squares Method (MLSM)	Not available	Noisy and/or nonlinear functions	
HyperKriging (HK)	Not available	Nonlinear functions without noise	
Radial Basis Functions (RBF)	Not available	Nonlinear functions without noise	

4. What should I do if the quality of the Fit obtained with FAST is not satisfactory?

FAST will test all the methods and hence will provide the best fit on the studied function. If the quality is still not satisfactory, you have to inspect the function behavior (ex. Discontinuity).

12 Useful Tutorials

There are some tutorials related to Fit, which are available in the HyperStudy documentation:

HS-3000

HS-3005

HS-1705

HS-1810

You can refer to them in order to practice fit approach.