Robustimizer Technical Guide

Contents

1 Initializing problem

The vector of design variables is referred to as **x** and the vector of noise variables is referred to as **z**. The vector **v** = (**x**,**z**) is input for the process model. The process model is evaluated for a selected number of different values of **v**, the Design of Experiments (DOE) made using Fractional or full factorial design (FFD).

The robust optimization starts by deciding on the number of design and noise variables with the help of sensitivity analysis step. Robustimizer supports not only independent noise variables but also performs principal component analysis (PCA) on correlated noise data imported from measurements.

The correlations among noise variables are evaluated for a better description of the input noise. For this purpose, the covariance principal component analysis is used to find

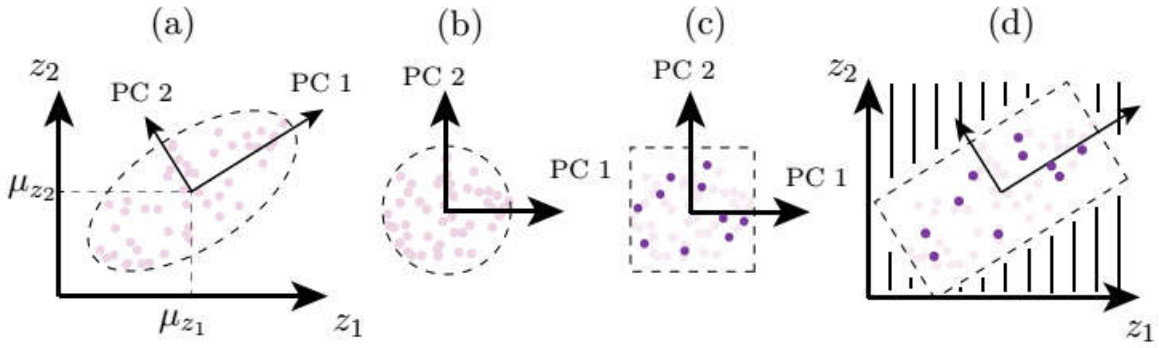

Figure 1: Schematic of sampling from correlated parameters [1]

the principal components (PCs) of the noise variables as described by [2]. The purpose of PCA analysis is to get uncorrelated subspace, to possibly reduce the parameter space and to avoid sampling of physically-unlikely parameter combinations when the DOE is created.

The orthogonal transformation from the correlated data to linearly uncorrelated PCs is shown schematically in Figures 1(a) and 1(b). The DOE is then created in the PC space (Figure 1(c)) avoiding physically-unlikely parameters sets, the hashed area shown in Figure 1(d). A set of physically-unlikely parameters can lead to an overestimation of the response scatter during the robust optimization [2]. For evaluation of the process model, a back transformation from the PC domain to the physical parameter domain is performed. The mutual linear correlation coefficient for various material parameters are calculated by:

$$\rho_{\mathrm{X,Y}} = \frac{\mathrm{Cov}(\mathbf{X}, \mathbf{Y})}{\sigma_{\mathrm{X}} \sigma_{\mathrm{Y}}} \qquad (1)$$

where **X** and **Y** are two random variables centered about their mean, σ_X and σ_Y are their standard deviations, and Cov(**X**,**Y**) is the covariance, the degree of linear relationship between **X** and **Y**. The value for the correlation coefficient ranges from -1 to 1 and it indicates the amount of correlation between two parameters.

2 Making a DOE and model evaluation

In robust optimization, The DOE can be made using various schemes such as factorial design, central composite, random sampling, Latin hypercube sampling (LHS), and orthogonal sampling [3, 4]. It is desirable to have a small number of DOE points since the process model is often expensive to evaluate. In addition, the goal is to sample uniformly to get an accurate approximation of the underlying relationship between input and output in the whole design-noise domain.

Figure 2 provides a schematic representation of various sampling techniques for two input variables. In a full factorial or fractional factorial sampling (Figure 2-(a,b)), extreme values of each parameter are used. For the design variables, the lower and

the upper bound are selected as extreme values. The extreme values for a noise parameter are generally $\mu_z \pm 3\sigma_z$ in which μ_z is the mean and σ_z is the standard deviation of that noise parameter.

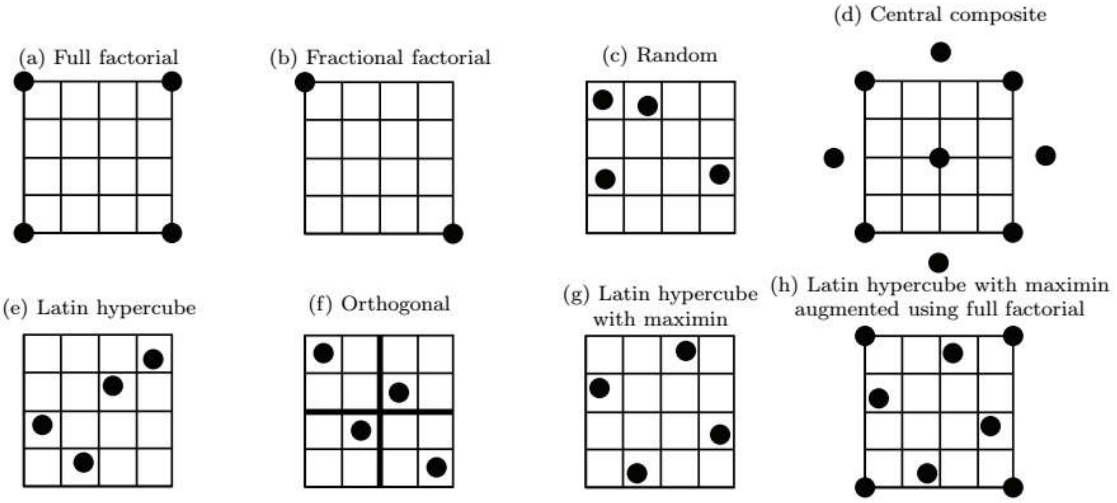

Figure 2: Various methods for generating a DOE

In random sampling the DOE points are generated without considering the previously generated points (Figure 2(c)). Therefore, it is very probable that the points are not evenly distributed. A central composite design (Figure 2(d)) consists of three types of points including FFD, the center point, and axial (star) points. In LHS the range of each parameter is divided into n_{DOE} equiprobable bins and each bin is sampled once (Figure 2(e)). Orthogonal sampling is an extension to LHS in which the sampling domain is divided into sub-domains and, similarly to LHS, the domain is sampled such that each subdomain has the same density of points (Figure 2(f)). The LHS method can be implemented in such a way that the minimum distance between the points is maximized (Maximin). In that case, a uniform and disperse sample can be obtained (Figure 2(g)). The Maximin approach can also be used for combination of LHS and FFD (Figure 2(h)).

The choice of the size of the sample to build the DOE directly influences the accuracy of the surrogate model and the computational effort required to build the surrogate model. As a rule of thumb, it is proposed to select the number of sampling points equal to 10 times the number of input variables for moderately complex functions. This number can be altered if a highly nonlinear process response is expected. Nevertheless, it is feasible to start using a small number of sample points and add infill points to the initial DOE at later stages of the robust optimization to improve the accuracy of the surrogate model where necessary. Robustimizer not only supports many standard methods such as LHS or factorial design, but also offers the flexibility to import any custom-made DOE in a text file.

The process model - sometimes referred to as black-box function - is evaluated in each DOE point to obtain the output. The response can be the result of a computer simulation, evaluation of the analytical model, or experimental results. The models

and simulations are an approximation of a real process and therefore are not exact. They must be accurate enough to capture the influence of variation in the input variables. If the difference between the predicted response and the real response is large, the process model must be improved. Furthermore, it is recommended to perform a study on the numerical noise of the model before running the optimization procedure, to make sure that the order of variation due to numerical noise is lower than the order of variation caused by noise variables [2].

Process response can be imported via GUI using two different methods. The response can be evaluated in DOE points using any simulation tools and then imported via GUI in a text file. Alternatively, an executable script file can be selected which is capable of reading the DOE points as a text file and write the output in another text file. This feature offers a significant advantage as it enables the iterative addition of new DOE points in the later stages of optimization. Additionally, it allows for the automatic import of process model simulation results into the software. This will be covered in next sections.

3 Surrogate modeling and validation

The surrogate model describes the relationship between the response of the process model r (or constraint response, r_c) and the input vector $\mathbf{v} = (\mathbf{x},\mathbf{z})$. The surrogate model is an approximate representation of the output obtained from process model evaluation. It is possible to assess the prediction error at each data point, enabling the iterative enhancement of the surrogate model's prediction by utilizing this error evaluation.

A surrogate model built on the discrete responses obtained from evaluations of a process model is required to search for the robust optimum design. One can choose from several methods of surrogate modeling, such as Gaussian process (GP) models, radial basis function networks, neural networks or regression models [5]. The choice of the surrogate model depends on the complexity of the response with respect to input variables. GPs are a widely used method for surrogate modeling and therefore it is implemented in Robustimizer by describing the response of the process model using:

$$r(\mathbf{v}) = \boldsymbol{\phi}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y} \qquad (2)$$

In this equation, the vector $\boldsymbol{\phi}$ contains the correlations between the point $\mathbf{x}$ and the DOE points, $\mathbf{R}$ is an $n_{\mathrm{DOE}} \times n_{\mathrm{DOE}}$ matrix that contains the spatial correlations between all DOE points, and $\mathbf{y}$ is the vector containing the responses of the process model on the DOE points. The error estimate, $\hat{s}_r$ at every point can be calculated by:

$$\hat{s}_{\mathrm{r}}(\mathbf{v}) = \hat{\sigma}^2[1 - \boldsymbol{\phi}^{\mathrm{T}}\mathbf{R}^{-1}\boldsymbol{\phi}]$$
$$\hat{\sigma}^2 = \frac{\mathbf{y}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{y}}{n_{\mathrm{DOE}}} \qquad (3)$$

The main assumption is that the data is a realization of a Gaussian random field which means that the responses are spatially correlated. The distance between the DOE points is used to determine the correlation between them.
In surrogate modeling, validation techniques play a crucial role in assessing the accuracy and reliability of the surrogate models. These techniques help ensure that the surrogate models accurately represent the underlying system or process being modeled. One commonly used validation technique is cross-validation. Cross-validation involves dividing the available data into multiple subsets or folds. The surrogate model is then trained on a subset of the data and tested on the remaining subset. This process is repeated multiple times, with different subsets used for training and testing in each iteration. The results from each iteration are then averaged to provide an overall estimate of the surrogate model performance. One of the most commonly used cross-validation techniques is leave-one-out cross-validation (LOOCV). LOOCV is a special case of k-fold cross-validation where k is equal to the number of data points in the dataset. In each iteration of LOOCV, one data point is withheld as the test set, and the surrogate model is trained on the remaining data points. This process is repeated for each data point in the dataset, and the results are averaged to obtain the final performance estimate. Using the Surrogate model tab in Robustimizer, the user is able to fit, plot and validate the surrogate models based on GPs [6].

4 Optimization settings

In the optimization settings tab the solution methods, objective function and constraints and the uncertainty propagation methods can be chosen using various techniques that are discussed in this section.

4.1 Solution methods

Generally, a robust optimization problem in the presence of constraints can be expressed as:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{h}(\mathbf{x}) = \mathbf{0} \\ & \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\ & \mathbf{lb} < \mathbf{x} < \mathbf{ub} \end{aligned} \tag{4}$$

where $f(\mathbf{x})$ is the objective function, $\mathbf{h}(\mathbf{x})$ are the equality constraints, $\mathbf{g}(\mathbf{x})$ are the inequality constraints, **lb** are the lower bounds of **x** and **ub** are the upper bounds of **x**. This formulation can be used for both deterministic and probabilistic optimization. The difference is in the definition of the objective function and constraints. The definition of the objective function and constraints for a robust optimization

approach will be presented in the next section. The focus in this section is to introduce the methods that are used to solve such an optimization problem specifically when the objective function, the constraints, or both are nonlinear. In that case, Equation (4) is referred to as a constrained nonlinear problem. To solve such a problem, constrained nonlinear optimization algorithms can be employed [7]. There are two main classes of algorithms namely derivative-free or stochastic techniques such as genetic algorithm (GA), and iterative methods which require the derivatives such as iterative quadratic programming (SQP). A genetic algorithm maintains a large population of candidate solutions [8] in contrast to iterative search methods in which a single potential solution is generated at each iteration. GA is on the basis of bio-inspired operators (e.g. mutation, crossover and selection) and a population of candidate solutions evolves toward better solutions. Both methods are available to use with various classes of problems in Robustimizer.

4.2 Objective function and constraints

One of the earliest methods for robust process design originates from the work of Taguchi [9]. The Taguchi method is generally used to classify robust design problems. The objective in robust design is one of the following:

- The smaller the better
- The larger the better
- On target is best

When the focus is mainly on the *performance* of the process (mean of the response), the first two approaches are employed. In the third approach, the Taguchi method typically involves a two-step procedure. In the first step, certain design variables are identified to reduce the variability of the response. In the second step, different design variables are utilized to shift the mean towards the target value.

A common approach, employed in Robustimizer is to simultaneously minimize the response variability while setting the mean at the target value. This approach must also satisfy certain constraints under the influence of uncertainties. Various methods can be employed to quantify the response variability and handle the constraints. In robust optimization, the satisfaction of constraints is directly associated with the probability of failure, often referred to as chance constraints in the literature. The main challenge in analyzing the constraints lies in evaluating the probability of failure since the response probability density function is unknown. To address this, a *moment matching* technique is frequently used, which approximates the reliability based on the statistical moments of the response. More specifically, the mean and standard deviation (or variance) are used to estimate the reliability of the satisfaction of a constraint using:

$$\mu_{\mathrm{r_c}}(\mathbf{x}) + n\sigma_{\mathrm{r_c}}(\mathbf{x}) \leq 0 \qquad (5)$$

where $\mu_{r_c}(\mathbf{x})$ and $\sigma_{r_c}(\mathbf{x})$ are the mean and standard deviation of the response of a constraint. The mean and standard deviation are simple to compute using limited stochastic data and therefore this method is widely used in the literature [10, 11]. The choice of n is related to the probability of constraint satisfaction assuming a normal distribution for the response of the constraint.

For a lower bound constraint:

$$\mu_{r_c}(\mathbf{x}) - n\sigma_{r_c}(\mathbf{x}) \geq 0 \tag{6}$$

There are several approaches to define the objective function, and one common measure for assessing the variability of the response is the standard deviation. The standard deviation provides an indication of the data spread around the mean value. For a robustness measure that minimizes the variation of response in addition to the difference between mean and target value, C_r, several expressions are proposed in such as:

$$\text{minimize} \ \left((\mu_r(\mathbf{x}) - C_r)^2 + w\sigma_r^2(\mathbf{x})\right) \tag{7}$$

Or:

$$\text{minimize} \ \left(|\mu_r(\mathbf{x}) - C_r| + w\sigma_r(\mathbf{x})\right) \tag{8}$$

In Equations (7) and (8), w is a weighting factor to adjust the optimization objective between mean on target and response variation. By varying w, this weighted sum formulation can lead to a set of optimal solutions. Then a set of Pareto optimal solutions is obtained which indicates the trade-off between the deviation of the mean from the target and the variation of the response.

4.3 Uncertainty quantification

The criteria used for evaluation of objective function and constraints require the statistical moments of the response (mean and standard deviation). Finding the statistical moments of the response from the noise variables is referred to as noise propagation. Several methods for noise propagation have been developed over past decades. Monte Carlo (MC) and its variations, perturbation methods, Gaussian Quadrature (GQ), polynomial chaos, Bayesian statistical modeling, method of moments (Taylor-series expansion) and stochastic collocation have been widely used [12]. Dimensionality of the problem [13] and fidelity of the model [14] determine the efficiency and effectiveness of these methods. The mean and variance of response are expressed by the following integrals in the noise space:

$$\mu_r(\mathbf{x}) = \int_{\mathbf{z}} r(\mathbf{x}, \mathbf{z}) p(\mathbf{z}) d\mathbf{z} \tag{9}$$

$$\sigma_r^2(\mathbf{x}) = \int_{\mathbf{z}} [r(\mathbf{x}, \mathbf{z}) - \mu_r(\mathbf{x})]^2 p(\mathbf{z}) d\mathbf{z} \tag{10}$$

Monte Carlo (MC) analysis is one of the most widely used methods in the literature for propagation of noise and for approximating the above-mentioned moments [15, 16]. Sampling from the noise parameter involves selecting values from its probability distribution. Several methods can be used to perform this sampling, and the concept is akin to the sampling techniques described in Section 3.2. The primary distinction lies in the objective, while Section 3.2 focuses on uniform sampling, here the samples are drawn from a probability distribution. To perform the sampling, random numbers between 0 and 1 are chosen from the cumulative probability distribution (CDF) of a normal distribution and then mapped to the noise domain. With random sampling, the previously sampled points are not taken into account, and the points can belong to any subset of the sampling domain. In Latin Hypercube Sampling (LHS), the CDF is divided into sub-domains of equal size, and sampling is conducted by adding new points while ensuring that more than one point is not selected in each domain.

After choosing the samples, the approximate mean and standard deviation are evaluated by:

$$\begin{aligned} \mu_{\mathrm{r}}(\mathbf{x}) &\simeq \frac{1}{N_{\mathrm{mc}}} \sum_{s=1}^{N_{\mathrm{mc}}} r(\mathbf{x}, \mathbf{z}_s) \\ \sigma_{\mathrm{r}}^2(\mathbf{x}) &\simeq \frac{1}{N_{\mathrm{mc}}} \sum_{s=1}^{N_{\mathrm{mc}}} \left(r(\mathbf{x}, \mathbf{z}_s) - \mu_{\mathrm{r}}(\mathbf{x})\right)^2 \end{aligned} \tag{11}$$

where N_{mc} is the number of sample points drawn from the noise probability distribution, $p(\mathbf{z})$. It has been shown by Nejadseyfi et al [17-19] that mean, standard deviation, higher order moments and their derivatives can be efficiently obtained by using analytical formulas for GPs and correlated/uncorrelated noise variables by:

$$\mu_r(\mathbf{x}) = a_0 + \sum_{i=1}^{N} \left\{ a_i \prod_{p=1}^{m} b_{ip}(x_p) \prod_{q=1}^{n} C1_{iq} \right\} \tag{12}$$

and:

$$\sigma_r^2(\mathbf{x}) = \sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j \prod_{p=1}^{m} b_{ip}(x_p) b_{jp}(x_p) \left(\prod_{q=1}^{n} C2_{ijq} - \prod_{q=1}^{n} C1_{iq} C1_{jq}\right) \tag{13}$$

where $C1$ and $C2$ depend on the choice of surrogate model and probability distribution:

$$C1_{iq} = \int_{z_q} b_{iq}(z_q) p(z_q) dz_q \tag{14}$$

$$C2_{ijq} = \int_{z_q} b_{iq}(z_q) b_{jq}(z_q) p(z_q) dz_q \tag{15}$$

where b_{iP} (X_P) are basis functions for that surrogate model and $p(\mathbf{z})$ is the input probability distribution. Once $C1$ and $C2$ are known for combination of surrogate model and probability distribution, mean and standard deviation of output can be evaluated at each design point. This method is programmed using vectorized operations which is computationally efficient compared to numerical integration e.g. by MC method and it is one of the main features that contribute to efficiency and speed of Robustimizer. Moreover, in gradient-based optimization, the gradients can also be calculated efficiently using:

$$\frac{\mathrm{d}\mu_{\mathrm{r}}(\mathbf{x})}{\mathrm{d}x_u} = \sum_{i=1}^{N} \left\{ a_i \times \frac{\frac{\mathrm{d}b_{iu}(x_u)}{\mathrm{d}x_u}}{b_{iu}(x_u)} \prod_{p=1}^{n_{\mathrm{x}}} b_{ip}(x_p) \prod_{q=1}^{n_{\mathrm{z}}} C1_{iq} \right\} \tag{17}$$

$$\frac{\mathrm{d}\sigma_{\mathrm{r}}(\mathbf{x})}{\mathrm{d}x_u} = \frac{1}{2\sigma_{\mathrm{r}}(\mathbf{x})} \sum_{i=1}^{N} \sum_{j=1}^{N} \left\{ a_i a_j \times \frac{\frac{\mathrm{d}b_{iu}(x_u)}{\mathrm{d}x_u} b_{ju}(x_u) + b_{iu}(x_u) \frac{\mathrm{d}b_{ju}(x_u)}{\mathrm{d}x_u}}{b_{iu}(x_u) b_{ju}(x_u)} \right.$$

$$\left. \prod_{p=1}^{n_{\mathrm{x}}} b_{ip}(x_p) b_{jp}(x_p) \times \left(\prod_{q=1}^{n_{\mathrm{z}}} C2_{ijq} - \prod_{q=1}^{n_{\mathrm{z}}} C1_{iq} C1_{jq} \right) \right\} \tag{18}$$

5 Evaluating the robust optimum and improving surrogate model accuracy

An optimum set of design variables found using the surrogate model is not always equal to the optimum of the underlying process model. This occurs when the prediction behaviour of the surrogate model around the predicted optimum is poor when there are no sampling points around the optimum. Therefore, it is necessary to use an update procedure to improve the surrogate model and subsequently obtain an accurate robust optimum. Moreover, the optimum might be in the areas that are less explored due to the initial DOE design.

Gaussian processes provide a flexible framework for modelling the uncertainty associated with data points. They define a prior distribution over functions and, after observing data, produce a posterior distribution that incorporates the data information. When using Gaussian processes for Bayesian optimization or active learning, the goal is to find the optimal points in a high-dimensional search space with limited observations or evaluations. Exploration involves gathering more information about the unknown regions of the search space by adding more points into the design of experiments, while exploitation focuses on exploiting the regions that appear to be optimal based on the current knowledge. The exploration-exploitation trade-off is often formalized using acquisition functions in the context of Gaussian processes. An acquisition function provides a measure of how valuable it is to evaluate a candidate point in the search space. The selection of the

acquisition function depends on the specific problem and the trade-off between exploration and exploitation.

Unlike space-filling methods that the new points are added to the initial DOE in the sparsely-sampled regions, adaptive sampling techniques require a criterion to add an infill point where it is needed most. In some cases, the infill point is added at the predicted optimum. However, in surrogate-model-based robust optimization uncertainties the infill criterion is based on the surrogate model estimation error, $\hat{s}$. Using this potential error, various methods can be developed to add infill points. A simple approach is to add an infill point where $\hat{s}$ has the biggest value.

One of the most used adaptive methods is based on *expected improvement* [20] that has been proposed to take into account both local and global search for new infill points in deterministic optimization. The expected improvement method has a closed-form of the acquisition function, which effectively uses the existing sampling points to improve the optimization accuracy. The expected improvement is defined by:

$$EI(\mathbf{x}) = (r^*_{\min} - \hat{r})\Phi\left(\frac{r^*_{\min} - \hat{r}}{\hat{s}_{\mathrm{r}}}\right) + \hat{s}_{\mathrm{r}}\phi\left(\frac{r^*_{\min} - \hat{r}}{\hat{s}_{\mathrm{r}}}\right) \tag{16}$$

In this equation, ϕ is the standardized normal distribution, Φ is the cumulative distribution of a standardized normal distribution and $r^*_{\min}$ is the minimum value of the response at the DOE points examined so far. $\hat{r}$ and $\hat{s}_{\mathrm{r}}$ are the predicted value and uncertainty of the predicted response. The procedure is to search for a point that has the highest *EI* value to add it to the initial DOE.

Equation (16) is used to optimize the response of a process (deterministic optimization).

It can be altered to be used in a robust optimization procedure [21]. A new infill point $(\mathbf{x}', \mathbf{z}')$ must be selected in the combined design and noise space. The objective function value, $f(\mathbf{x})$, replaces the response, $r(\mathbf{x})$. Moreover, to evaluate $f(\mathbf{x})$ and get the minimum value of the objective function at the DOE points, $f^*_{\min}$, one needs to calculate $\mu_{\mathrm{r}}(\mathbf{x})$ and $\sigma_{\mathrm{r}}(\mathbf{x})$. Therefore, the objective function values are not a result of evaluations of the process model, but of a prediction using a surrogate model. Thus, there is a prediction uncertainty at the current best point. The minimum objective function value at the DOE points has an uncertainty of $\hat{s}^*$. A suitable estimation for the prediction error at any design point, $\hat{s}(\mathbf{x})$, is also required. The uncertainty measure on surrogate model ($\hat{s}_r$) is dependent on both design variables and noise variables. To obtain the uncertainty of the objective function value ($\hat{s}_f$) an integral over noise space is usually evaluated (calculating mean value of mean square error (MSE)) :

$$\hat{s}^2_{\mathrm{f}}(\mathbf{x}) = \int_{\mathbf{z}} \hat{s}^2(\mathbf{x}, \mathbf{z}) p(\mathbf{z}) d\mathbf{z} \tag{17}$$

The influence of uncertainty of the best point, $\hat{s}^*$, is ignored and expected improvement is evaluated using [20]:

$$EI(\mathbf{x}) = \bar{\omega}(f^*_{\min} - f)\Phi\left(\frac{f^*_{\min} - f}{\hat{s}_{\mathrm{f}}}\right) + (1 - \bar{\omega})\hat{s}_{\mathrm{f}}\phi\left(\frac{f^*_{\min} - f}{\hat{s}_{\mathrm{f}}}\right) \tag{18}$$

More details about including the influence of $\hat{s}^*$ can be found elsewhere [22]. The first term in Equation (18) is related to local search (Exploitation near the predicted optimum) and the second term is related to global search (Exploration). One can adjust the search in the global and local domain by choosing a proper weight factor such that $0 < \omega < 1$. Maximizing expected improvement using (18) leads to an infill point in the design space, $\mathbf{x}'$. At that design setting, a point in noise space must be selected to be able to evaluate the process model. For this purpose $\mathbf{z}' = \text{argmax}_{\mathbf{z}} (\hat{s}^2(\mathbf{x}', \mathbf{z})p(\mathbf{z}))$ is employed. The point $(\mathbf{x}', \mathbf{z}')$ is then added to the initial DOE, a new surrogate model is fitted and the robust optimum is evaluated again using the updated surrogate model.

Since exploration-exploitation is an adaptive method, Robustimizer supports external scripts which provide the function evaluation on a newly added infill point. Using this method it is convenient to update the DOE automatically, evaluate the function output in the new DOE, fit a new surrogate model and repeat robust optimization sequentially. This feature and its settings are shown in he next section.

References:

1. Nejadseyfi, O., *Robust Optimization and Tailoring of Scatter in Metal Forming Processes*. 2019: Gildeprint.
2. Wiebenga, J.H., et al., *Effect of material scatter on the plastic behavior and stretchability in sheet metal forming.* Journal of Materials Processing Technology, 2014. **214**(2): p. 238-252.
3. Uy, M. and J.K. Telford. *Optimization by Design of Experiment techniques*. in *2009 IEEE Aerospace conference*. 2009.
4. Tang, B., *Orthogonal Array-Based Latin Hypercubes.* Journal of the American Statistical Association, 1993. **88**(424): p. 1392-1397.
5. Luo, J. and W. Lu, *Comparison of surrogate models with different methods in groundwater remediation process.* Journal of Earth System Science, 2014. **123**(7): p. 1579-1589.
6. Lophaven, S.N., et al., *A matlab kriging toolbox.* Technical University of Denmark, Kongens Lyngby, Technical Report No. IMM-TR-2002-12, 2002.
7. Baginski, M.E., D.L. Faircloth, and M.D. Deshpande, *Comparison of two optimization techniques for the estimation of complex permittivities of multilayered structures using waveguide measurements.* IEEE Transactions on Microwave Theory and Techniques, 2005. **53**(10): p. 3251-3259.
8. Homaifar, A., C.X. Qi, and S.H. Lai, *Constrained Optimization Via Genetic Algorithms.* SIMULATION, 1994. **62**(4): p. 242-253.
9. Taguchi, G., *System of experimental design: engineering methods to optimize quality and minimize costs*. Vol. 1. 1987: UNIPUB/Kraus International Publications.
10. Scholar, V., X. Du, and W. Chen, *A Methodology for Managing the Effect of Uncertainty in Simulation-Based Design.* AIAA Journal, 2000. **38**.
11. Du, X. and W. Chen, *Efficient uncertainty analysis methods for multidisciplinary robust design.* AIAA Journal, 2002. **40**: p. 545-552.

12. Lee, S.H. and W. Chen, *A comparative study of uncertainty propagation methods for black-box-type problems.* Structural and Multidisciplinary Optimization, 2009. **37**(3): p. 239-253.
13. Fuchs, M. and A. Neumaier. *Handling Uncertainty in Higher Dimensions with Potential Clouds towards Robust Design Optimization*. in *Soft Methods for Handling Variability and Imprecision*. 2008. Berlin, Heidelberg: Springer Berlin Heidelberg.
14. Ng, L.W.T. and K.E. Willcox, *Multifidelity approaches for optimization under uncertainty.* International Journal for Numerical Methods in Engineering, 2014. **100**(10): p. 746-772.
15. Zhou, Q., et al., *Robust optimization for reducing welding-induced angular distortion in fiber laser keyhole welding under process parameter uncertainty.* Applied Thermal Engineering, 2018. **129**: p. 893-906.
16. Putko, M., et al., *Approach For Uncertainty Propagation And Robust Design In Cfd Using Sensitivity Derivatives.* Journal of Fluids Engineering, 2001. **124**.
17. Nejadseyfi, O., H. Geijselaers, and T. van den Boogaard, *Robust optimization based on analytical evaluation of uncertainty propagation.* Engineering Optimization, 2019.
18. Nejadseyfi, O., H.J. Geijselaers, and A.H. van den Boogaard, *Evaluation and assessment of non-normal output during robust optimization.* Structural and multidisciplinary optimization, 2019. **59**: p. 2063-2076.
19. Nejadseyfi, O., H.J.M. Geijselaers, and A.H. van den Boogaard, *Evaluation and assessment of non-normal output during robust optimization.* Structural and Multidisciplinary Optimization, 2019. **59**(6): p. 2063-2076.
20. Jones, D.R., M. Schonlau, and W.J. Welch, *Efficient Global Optimization of Expensive Black-Box Functions.* Journal of Global Optimization, 1998. **13**(4): p. 455-492.
21. Ur Rehman, S. and M. Langelaar, *Efficient infill sampling for unconstrained robust optimization problems.* Engineering Optimization, 2016. **48**(8): p. 1313-1332.
22. Sóbester, A., S.J. Leary, and A.J. Keane, *A parallel updating scheme for approximating and optimizing high fidelity computer simulations.* Structural and Multidisciplinary Optimization, 2004. **27**(5): p. 371-383.