Robust optimization of a mathematical function with sequential improvement and automatic metamodel update

The Branin function is generally used as a test function in optimization. The Branin function has two variables. To use this function in a robust optimization context, one variable is designated as design parameter and the other one is noise variable[1].

$$r(x,z) = (z - \frac{5.1x^2}{4\pi^2} + \frac{5x}{\pi} - 6)^2$$
$$+ 10\left(1 - \frac{1}{8\pi}\right)\cos(x) + 10 \quad (1)$$

Where x is the design parameter and z is the noise variable that follows a normal distribution described by a given mean and standard deviation. It is assumed that no constraints are present. The optimization problem is expressed by:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) = \mu_r(x) + 3\sigma_r(x) \end{array}$$

$$\begin{array}{ll} \text{Subject to} & -5 \leq x \leq 10 \\ & z \sim \mathcal{N}(7.5, 2.5) \end{array}$$

Table 1 and

Table 2 show the input variables of the Branin function

Table 1. Lower and upper bounds for design variable

Variable name	Lower bound	Upper bound
x	-5	10

Table 2. Mean and standard deviation for noise variable

Variable name	Mean	Standard deviation
z	7.5	2.5

As there are only 1 design and 1 noise variables we can directly proceed to the *Initializing problem* tab without performing the sensitivity analysis. In the I*nitializing problem* tab, the values from Table 1 and

Table 2 are entered as shown in Figure 1.

Figure 1. Design and noise data

A DOE of 10 points including the LHS design is created in the next tab.

Figure 2. Design and noise data

In this example the use of scripts to import the results to Robustimizer is explained. This option is activated by selecting the option of *Import using a script* in the *Import results method* panel as shown in Figure 3.

There are a few requirements to be able to use the automatic import of results:

➢ The script must be a standalone executable file.
➢ Script should read a file named in.txt which includes the DOE point sorted in columns and return a file named out.txt

---

[1] O. Nejadseyfi, H.J.M. Geijselaers, A.H. Boogaard, (2019). Robust optimization based on analytical evaluation of uncertainty propagation. Engineering Optimization, 51(9), 1581-1603.

➢ All files (Robustimizer, script executable file, in.txt and out.txt must be located in the one folder).

➢ If there are existing out.txt files in the folder, the script file must overwrite that file.

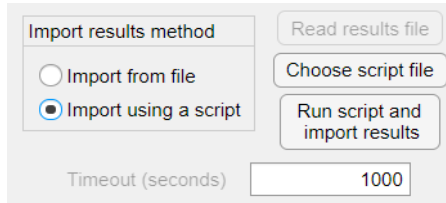➢ out.txt must be generated at once at the end of the calculations for all DOE point.

Figure 3. Import results method panel

In this example we use a script file compiled in Phyton that reads the file named *in.txt* and generates the output file *out.txt.* Therefore, the DOE saved in the previous stage with another name must be renamed to *in.txt.* (Figure 4)
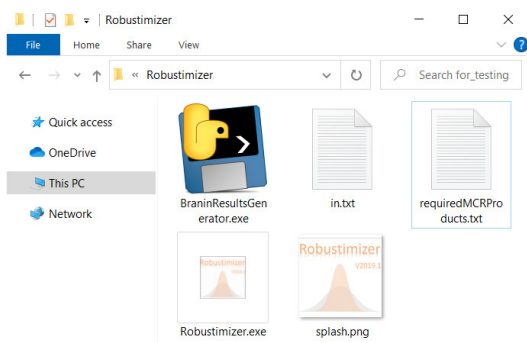

Figure 4. Executable script in the same folder as Robustimizer

Figure 5 shows the selection of the script file inside Robustimizer.
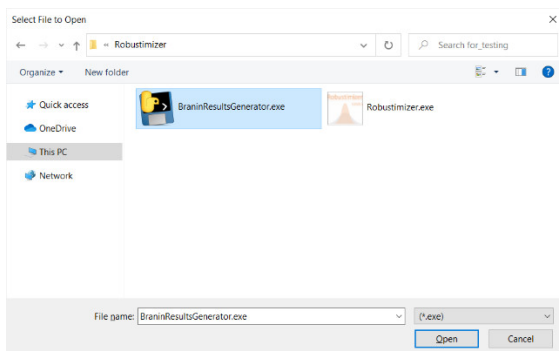

Figure 5. Selecting the script

After selecting the script file, by clicking the *Run script and import results* button the results on DOE points will be calculated by script and the output file generated by the script will be automatically imported into Robustimizer. In this case Robustimizer will wait for the generation of out.txt. If such a file in not generated it not possible to proceed with the optimization process. The evaluation of black-box function might be time consuming and depending on the script it might take up to several hours or days. It is recommended to use the automatic option for scripts that require shorter calculation times. After running the script successfully the results will appear in the table as show in Figure 6.

Figure 6. Auto-generated output

In the next tab a metamodel is fitted, cross validated and plotted as shown in Figure 7.
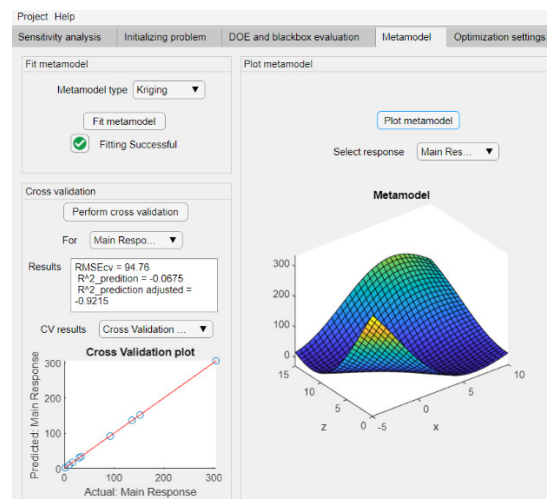

Figure 7. Generating metamodel

Robust optimization of a mathematical function with sequential improvement and automatic metamodel update

Desired optimization settings can be chosen in *Optimization settings* tab as shown in Figure 8. There are no constraints defined in this tab.

The optimization results are obtained by clicking the *Perform optimization* button in the *Robust optimization* tab and the scatter around the optimum can be visualized. (Figure 9 & Figure 10)

Figure 8. Optimization settings and options

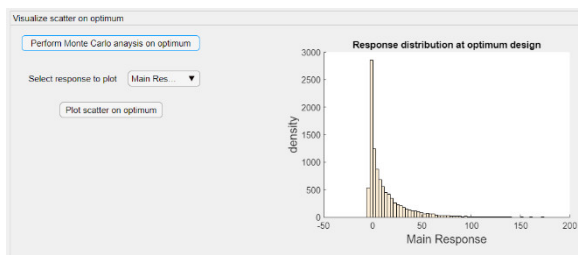*Figure 9. Optimization results*

Figure 10. Visualizing the scatter around the optimum

The biggest advantage of using a script file to generate the results of black-box function evaluation is the automatic execution of sequential improvement by Robustimizer. In this case a candidate DOE point is automatically added to the existing DOE. Subsequently, the results of black-box function evaluation is automatically added to Robustimizer and the metamodel is updated. Robust optimization is also automatically performed and the results of each iteration is shown in the *sequential improvement* tab. In this example 2 steps of sequential improvement is performed using jones criterion with a weighting factor of 0.5.

By clicking on the *Perform sequential metamodel improvement automatically and perform optimization* button Robustimizer will start a sequence of finding new infill point based on given criterion, update the metamodel and find robust optimum on new metamodel. The infill points are then displayed in *infill DOE* textbox and the results of optimization are displayed in *Optimization results* textbox. This step can be repeated multiple times and there is a possibility of visualizing the updated metamodel in the *Metamodel* tab. In addition, it is possible to combine multiple options of infilling in different stages of the optimization.

Figure 11. The results of sequential improvement of metamodel are shown in sequential improvement tab