Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2017), *31*, 286–297.
© Cambridge University Press 2017 0890-0604/17
doi:10.1017/S089006041700018X

How to compare performance of robust design optimization algorithms, including a novel method

JOHAN A. PERSSON AND JOHAN ÖLVANDER
Department of Management and Engineering, Linköping University, Linköping, Sweden

(Received October 19, 2016; Accepted January 19, 2017)

Abstract

This paper proposes a method to compare the performances of different methods for robust design optimization of computationally demanding models. Its intended usage is to help the engineer to choose the optimization approach when faced with a robust optimization problem. This paper demonstrates the usage of the method to find the most appropriate robust design optimization method to solve an engineering problem. Five robust design optimization methods, including a novel method, are compared in the demonstration of the comparison method. Four of the five compared methods involve surrogate models to reduce the computational cost of performing robust design optimization. The five methods are used to optimize several mathematical functions that should be similar to the engineering problem. The methods are then used to optimize the engineering problem to confirm that the most suitable optimization method was identified. The performance metrics used are the mean value and standard deviation of the robust optimum as well as an index that combines the required number of simulations of the original model with the accuracy of the obtained solution. These measures represent the accuracy, robustness, and efficiency of the compared methods. The results of the comparison show that sequential robust optimization is the method with the best balance between accuracy and number of function evaluations. This is confirmed by the optimizations of the engineering problem. The comparison also shows that the novel method is better than its predecessor is.

Keywords: Optimization; Robust Design Optimization; Surrogate Models; Surrogate-Based Optimization

1. INTRODUCTION

Modeling and simulation are used in the early stages of the product development process to analyze and compare different concepts. Most models are deterministic, which means that the model will yield the same results each time that it is simulated with the same settings. However, the performances of products are not deterministic in real life because they are affected by uncertainties and variations such as variable material properties, manufacturing tolerances, and varying environmental conditions.

Optimizations are commonly used to let the computer search for optimal products, but deterministic optimizations often lead to solutions that lie at the boundary of one or more constraints (Wiebinga et al., 2012). These solutions may lead to a high percentage of failure if the constraints are affected by uncertainties and variations. It is therefore desirable to perform optimizations where the statistics are taken into account, such as robust design optimization (RDO) or reliability-based design optimization.

The purpose of RDO is to find a robust optimal design that is insensitive to variations and uncertainties (Beyer & Sendhoff, 2007). The objective function of an RDO is therefore usually a linear combination of the mean value, μ, and standard deviation, σ, as shown in Eq. (1) (Aspenberg et al., 2013):

$$\min f(x) = \alpha\mu(x) + \beta\sigma(x). \tag{1}$$

The values of the coefficients, α and β, are used to determine how important the mean value and standard deviation are. This means that the mean value and standard deviation of the performance of the design needs to be estimated each time the value of the objective function is calculated.

Numerous methods for RDO exist (Beyer & Sendhoff, 2007), and it is important for the engineer or designer to be able to choose an appropriate method for the given problem. It is therefore desirable to enable benchmarking of RDO methods for a representative problem to guide the selection process. This paper addresses this problem by proposing a method that can be used to benchmark RDO methods.

Reprint requests to: Johan A. Persson, Department of Management and Engineering, Linköping University, Linköping SE-581 83, Sweden. E-mail: johan.persson@liu.se

The proposed benchmarking method is demonstrated through a comparison of five RDO methods, including a novel method.

1.1. Background

At least two numerical operations are needed to perform an RDO. One needs to estimate the robustness of a design, whereas the other handles the optimization.

Many methods can be used to estimate how uncertainties and variations affect the performance of a system, and in this paper Latin hypercube sampling (LHS) is used (McKay et al. 1979). It has been used by numerous authors (Beyer & Sendhoff, 2007) and can give reasonably accurate estimations of the mean value and standard deviation of the performance of a system by performing only a few tens of simulations (Persson & Ölvander, 2011). It is also easy to implement and use.

The most important properties of an optimization algorithm are the probability of finding the optimum, the wall clock time it takes to perform an optimization, and how easy the algorithm is to use (Krus & Ölvander, 2013). Different optimization algorithms have different advantages, and which one to choose depends on the characteristics of the problem and the preferences of the user. The two optimization algorithms that are used in this comparison are Complex-RF (Box, 1965; Krus & Ölvander, 2013), and a genetic algorithm (GA; Holland, 1975; Goldberg, 2006). These are described in Section 2.

Since an LHS is performed every time the optimization algorithm wants to calculate the value of the objective function, the wall clock time of an RDO of a computationally expensive model can be unrealistically long. One remedy is to use computationally efficient surrogate models (SMs) instead of the expensive model (Jin et al., 2003; Coelho, 2014).

Many comparisons of the efficiency of different types of SMs have been performed, but no common conclusion has been made (Wang & Shan, 2007). Anisotropic kriging is found to be the best one according to Jin et al. (2003) and Tarkian et al. (2012), and is therefore used in this paper. LHS has been used as sampling plan for SMs by several authors (Jin et al., 2003; Forrester et al., 2008; Wiebenga et al., 2012) and is used to create the initial SMs in this paper as well.

The remainder of the paper is structured as follows. The five compared methods are presented in Section 2, whereas the benchmarking method is presented in Section 3. A demonstration of the comparison method and a confirmation of its results are presented and discussed in Section 4. Finally, the conclusions summaries the paper in Section 5.

2. COMPARED METHODS

Five different RDO methods are compared in this paper, and they are presented in this section. The methods involve optimization algorithms and in this paper two different are used: Complex-RF and a GA.

The Complex-RF method is an extension of the original complex method developed by (Box, 1965) with inspiration from the Nelder–Mead simplex algorithm (Nelder & Mead, 1965), which has a good trade-off between accuracy and computational cost (Krus & Ölvander, 2013). It begins by spreading $k > n + 1$ points randomly in an n-dimensional design space to form a geometric shape referred to as a complex. The algorithm then progresses by reflecting the worst point in the complex through the centroid of the remaining points a reflection distance $\alpha > 1$. If this new point is still the worst, it is moved halfway toward the centroid. This process continues until convergence or the maximum number of function evaluations is reached. In Krus and Ölvander (2013), the complex-RF method is thoroughly explained and its performance optimized using meta optimization.

This paper uses the standard value of k, which is twice the number of optimization variables, and the maximum number of function evaluations is set to 500.

GAs mimic Darwin's theory of survival of the fittest; see, for example, Holland (1975) or Goldberg (2006). The algorithm is population based, where a new generation is obtained based on the best individuals in the previous generation. New individuals are created by combining genetic material from fit parents to create new children. It is also possible to include mutations to increase the robustness of the optimization. The main advantage of GAs are their generally high accuracy, or likelihood, of identifying the global optima in multimodal search spaces, whereas the drawback is that they usually require many objective function evaluations (Ölvander & Krus, 2006).

2.1. Brute force RDO (bfRDO)

The workflow of the first method is shown in Figure 1, and it is a method that performs robust design optimization without involving any SMs; hence, it is called the brute force method. As no SMs are used, the original model is simulated several times whenever the LHS estimates the mean value and standard deviations of a design. This means that the required number of simulations is the number of samples drawn by the LHS multiplied by the number of objective function evaluations of the optimization algorithm. Algorithms that require many objective function evaluations, such as GAs and particle swarm optimizations (Eberhart & Kennedy, 1995) are therefore unsuitable for this approach unless the model or function that is optimized is extremely fast to evaluate.

Complex-RF is chosen as an optimization algorithm for this method in this comparison due to its good trade-off between function evaluations and accuracy (Krus & Ölvander, 2013).

2.2. SM-based RDO (SMRDO)

A commonly used method for RDO is to fit an SM to the original model and then perform the RDO on the SM instead of the original model. The workflow is shown in Figure 2, and the benefit is that the only simulations of the original model that are needed are those that are used to create the SM. The SM can also be used for other purposes afterward.

A drawback with this method is that the SM needs to reanimate the original model accurately in the whole design space as the location of the optimum is unknown. Otherwise,

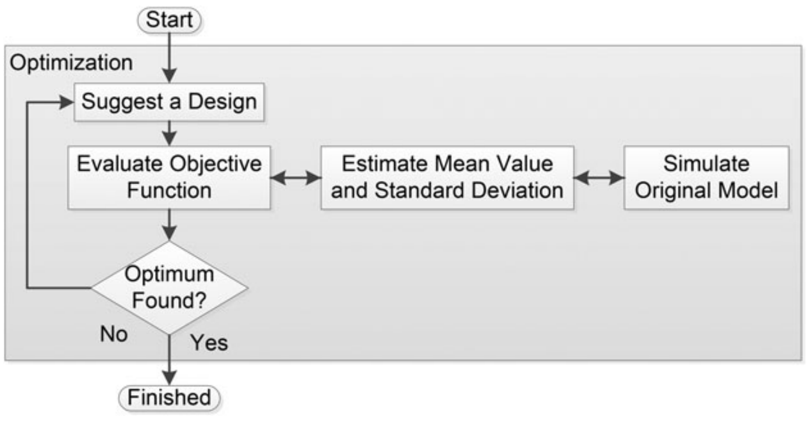

Fig. 1. A schematic workflow of a robust design optimization.

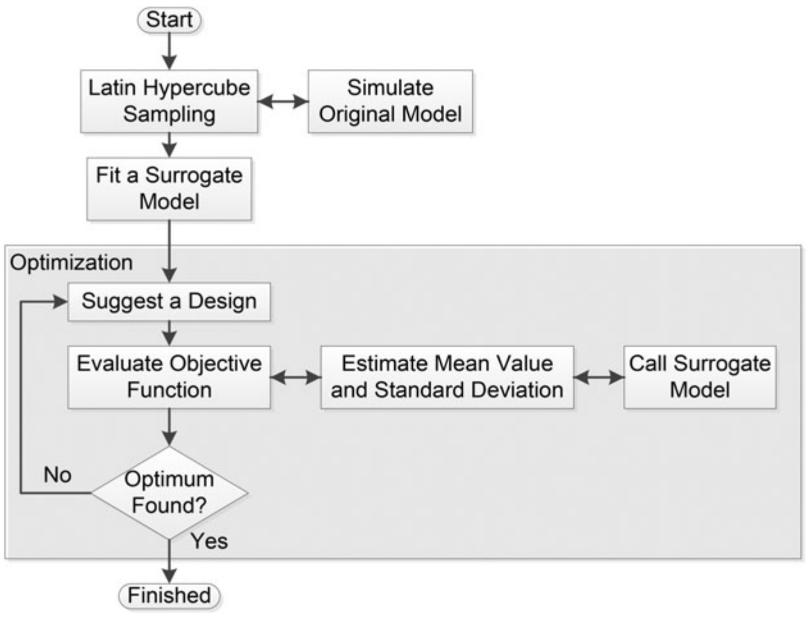

Fig. 2. A schematic workflow of a surrogate model-based robust design optimization.

there is always a risk that the SM is inaccurate in the vicinity of the optimum.

Because all evaluations during the optimization are made on the computationally effective SM, an optimization algorithm with a high accuracy should be chosen even if it requires many function evaluations. A GA is therefore chosen as the optimization algorithm for this problem.

2.3. Robust sequential optimization (RSO)

RSO is a newer approach that is presented in works by, for example, Wiebenga et al. (2012) and Rehman et al. (2014). A schematic of its workflow is shown in Figure 3. It begins by fitting an SM to the original model using LHS as sampling plan. An RDO is then performed on the SM to find the robust optimum. The original model is simulated once at the optimum, and the SM is updated with this new sample. A new RDO is then performed on the updated SM to find the new robust optimum. This iterative process continues until a termination criterion is reached.

The termination criteria that are used in this paper is that the algorithm stops when the same optimum has been found γ times in a row or the maximum allowed number of simulations of the original model has been reached. The allowed number of simulations of the original model is set to be either 10 times the number of variables or 50, depending on which one is the larger, and γ is set to 5. Because the LHSs that are performed in RSO are calling the computationally effective SM, GA is chosen as optimization algorithm for this method too.

2.4. Evolutionary algorithm for robustness optimization (EARO)

This method was proposed by Paenke et al. (2006) and uses a GA as optimization algorithm and polynomial response

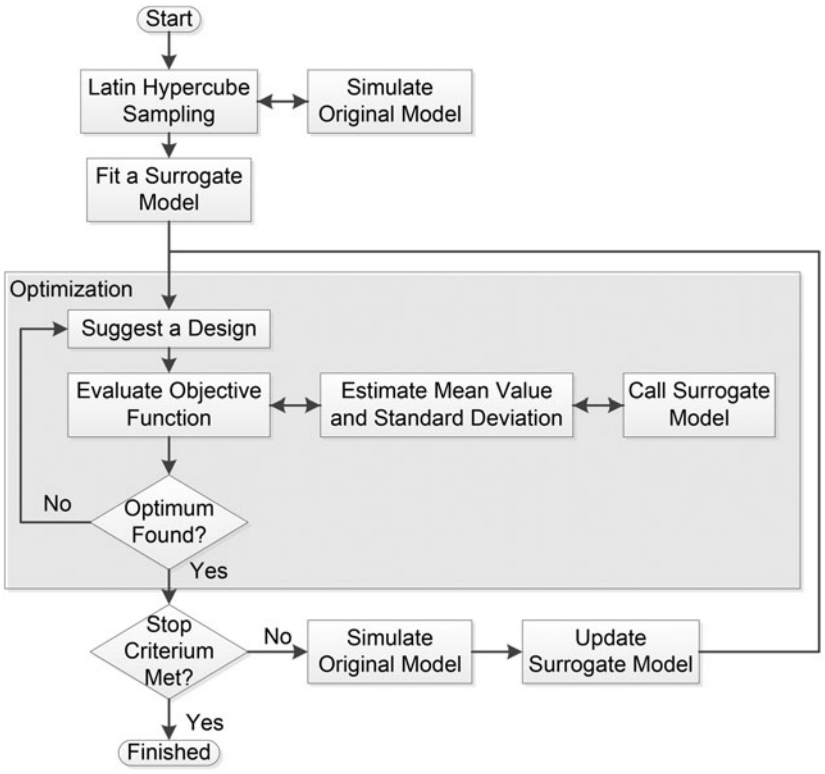

Fig. 3. A schematic workflow for a sequential robust optimization.

surfaces (PRSs) as surrogate models (Myers et al., 2009). Its workflow is shown in Figure 4.

Whenever the GA creates a new generation, the original model is simulated to make deterministic evaluations of the individuals. These values are stored and used to create SMs. A PRS is created from the closest samples when LHS should be used to estimate the mean value and standard deviation of an individual. The LHS is then calling the PRS instead of the original model.

This means that the required number of simulations of the original model for this RDO method equals the number of individuals that are evaluated by the GA. The computational cost is therefore similar to a deterministic optimization.

2.5. Complex-RDO (CRDO)

The proposed method is somewhat similar to EARO and an improved version of Approach 6 in Persson and Ölvander (2013) and its workflow is shown in Figure 5.

The method begins by fitting an SM to the original model according to an LHS sampling plan. The SM is then called by LHS to estimate the mean value and standard deviation at each of the initial points that were used to fit the SM.

The *k* points with the best objective function values are used as starting points for the Complex-RF optimization algorithm. Whenever Complex-RF wants to calculate the value of the objective function of a new point, a deterministic simulation of that design is performed, and the SM is updated with this value. LHS then calls the SM several times to estimate the mean value and standard deviation of the design to estimate its objective function value. This process continues until a stop criterion for Complex-RF is reached.

This means that the evaluation of the robustness of each design is preceded by a deterministic simulation of the original model. The total number of simulations of the original model is therefore the number of samples in the original sampling plan plus the number of points that are evaluated during the optimization.

It also means that there will always be at least one sample in the SM that is close to the evaluated design. This should increase the accuracy in the vicinity of the parts of the design space where the optimization algorithm currently operates.

The differences compared to the method proposed by Paenke et al. (2006) are that Complex-RF is used instead of GA as an optimization algorithm, and that kriging is used instead of PRS as the SM. The motive for using Complex-RF is that GA requires more function evaluations. A GA with 40 individuals and 40 generations means 1600 function evaluations. A complex optimization can often find a solution in 100–200 evaluations (Persson & Ölvander, 2015). Kriging usually performs better than polynomials as global SMs (Jin et al., 2003). Polynomials nevertheless show reasonable estimations for local estimations (Gobbi et al., 2013), and it can be argued that the estimations in this method are both global (in the beginning of the search) and local (at the convergence stage).

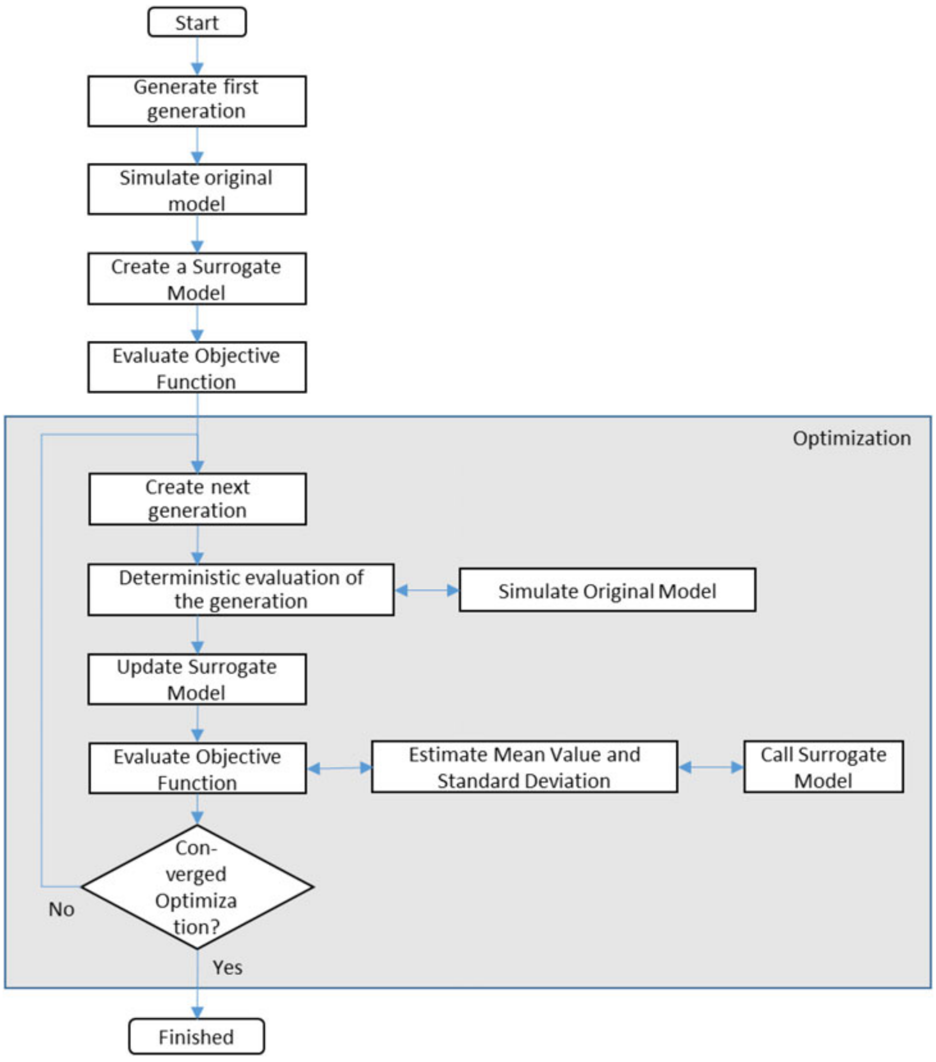
Start
Generate first generation
Simulate original model
Create a Surrogate Model
Evaluate Objective Function
Optimization
Create next generation
Deterministic evaluation of the generation
Simulate Original Model
Update Surrogate Model
Evaluate Objective Function
Estimate Mean Value and Standard Deviation
Call Surrogate Model
Con-verged Optimiza tion?
No
Yes
Finished

Fig. 4. The workflow for the evolutionary algorithm for robustness optimization.

3. PROPOSED COMPARISON METHOD

A schematic workflow for the proposed comparison method is shown in the bullet list below. A longer explanation of the steps follows below the list.

STEP 1. Decide which RDO methods should be benchmarked.

STEP 2. Decide on appropriate test models or functions to benchmark the methods on.

STEP 3. Decide on a robust objective function.

STEP 4. Choose performance measures

STEP 5. Perform numerous optimizations of the same problem with the same algorithm as shown in Figure 6.

STEP 6. Calculate performance metrics.

STEP 7. Compare the performance metrics for the RDO methods and test problems.

The first step is to identify the candidate RDO methods. This includes availability and ease of implementation and use. The next step is to identify and choose the test functions and models that the methods should be benchmarked on. It is important that the test problems are similar to the problems that are most desirable to solve. This includes the overall shape of the problem, number of optima, noisiness, and so on.

The third step is to choose the robust objective function that should be used. This means choosing the weights of Eq. (1) that balances the expected value versus the robustness of a solution against each other. The weights α and β are set to 1 and 0 for the comparison in this paper, to follow the definition by Branke (2001). This is shown in Eq. (2) and means that the optimization only strives to find the minimal mean value.

$$\min f(x) = \mu(x). \tag{2}$$

The fourth step is to choose which performance measures should be used to compare the RDO methods. This also en-

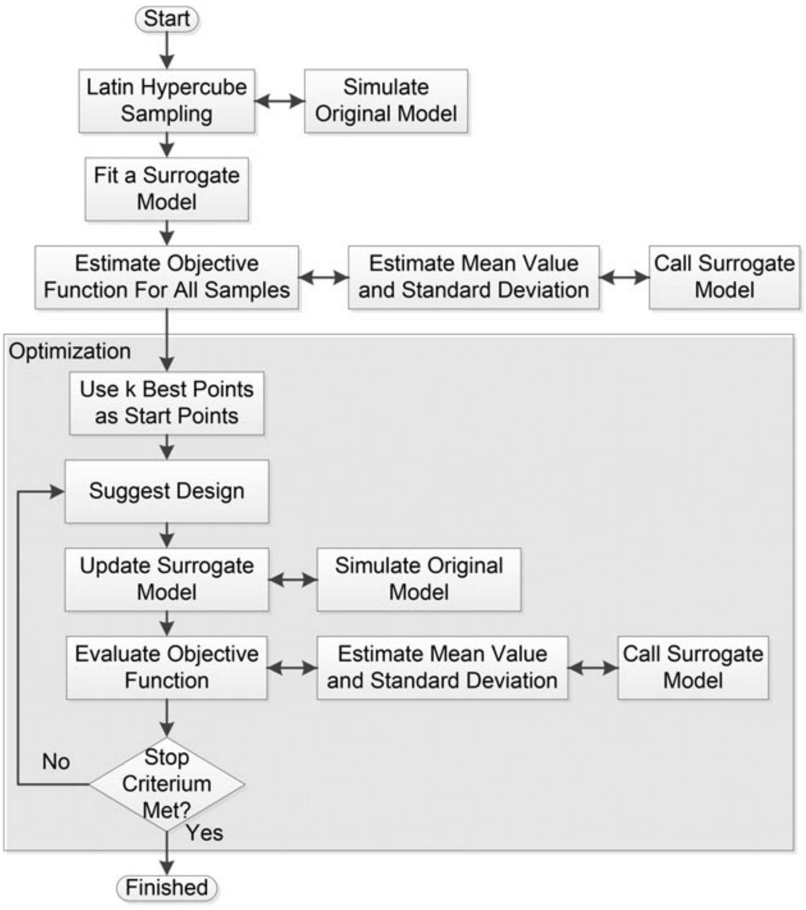

Fig. 5. A schematic workflow of a complex robust design optimization.

ables meta-optimization (Mercer & Sampson, 1978), where the parameters of the optimization algorithm can be optimized to optimize the performance of the algorithm itself.

The performance measures that are used to compare the RDO methods need to order the RDO methods according to several criteria:

- availability
- ease of use
- accuracy
- robustness
- efficiency

This means that both accuracy and required number of simulations of the original model need to be taken into account. Numerous optimizations with each method, as shown in Figure 6, are needed to test their performances as the optimizations estimate probabilistic phenomena and both the GA and complex algorithm are probabilistic methods. Popular measures of the accuracy and robustness are therefore mean values and standard deviations of the objective function values of the optimal points received from the optimizations (Tenne, 2015).

Several performance measures for the efficiency of an optimization algorithm have been proposed (Schutte & Haftka, 2005; Krus & Ölvander, 2013), but for this comparison the measure that is presented by Persson and Ölvander (2013), η, is used. The benefit with this measure is that it combines the accuracy of the optimization with the required number of simulations needed for it to converge. The metric is shown in Eq. (3) and can be interpreted as the probability of finding the optimum if 100 simulations of the original model are allowed. It can be noted that the value will be equal to 1 for all

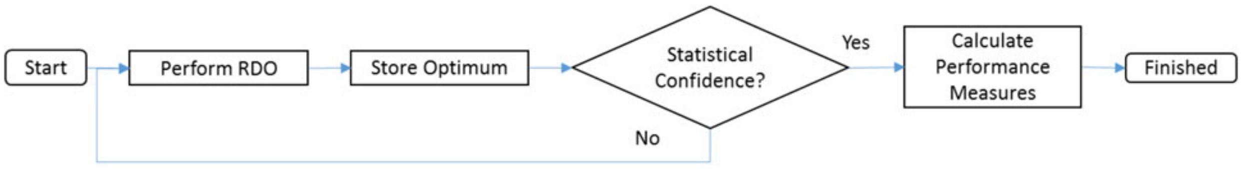

Fig. 6. A workflow for performing numerous robust design optimizations.

methods that have a 100% hit rate, but if two methods have the same accuracy, the one that requires the least number of simulations should naturally be chosen.

$$\eta = 1 - (1 - p_{\mathrm{opt}})^{100/n_{\mathrm{opt}}}. \tag{3}$$

The fifth step is to perform the numerous optimizations according to Figure 6. This step can be extremely time consuming if computationally expensive models were chosen in Step 2. The received optima are then used to calculate the performance metrics in Step 6. These metrics are then compared in Step 7 to investigate which RDO method performs best.

4. DEMONSTRATION OF THE COMPARISON METHOD

This demonstration of the comparison method strives to find the most appropriate method among the five in Section 2 for solving the problem in Section 4.1.

4.1. Engineering problem

The engineering problem that the best RDO method should be found for is the electrical motorcycle model presented by Persson and Ölvander (2015). It is implemented in MATLAB Simulink and consists of models of the battery, electrical motor, gear box, and the motorcycle itself. A screenshot of the model is shown in Figure 7. The objective is to optimize the velocity of the electrical motorcycle after 5 s. The optimization problem has three variables: the gear ratios of the first and second gear, and the speed at which the gear is shifted between the first and second gear. The randomness is introduced as a uniform noise of 1/8 of the variable ranges.

4.2. Test problems

Four mathematical functions are used to compare the five RDO methods. These are presented in Table 1 together with the electrical motorcycle problem. The table also contains information about the number of variables, variable limits, the randomness, and the criteria for a successful optimization for the different problems. The equations for the functions can be found in Appendix B together with descriptions of the properties of the functions.

The mathematical functions have been chosen to have approximately the same number of variables and the same behavior as the electrical motorcycle problem. They have also

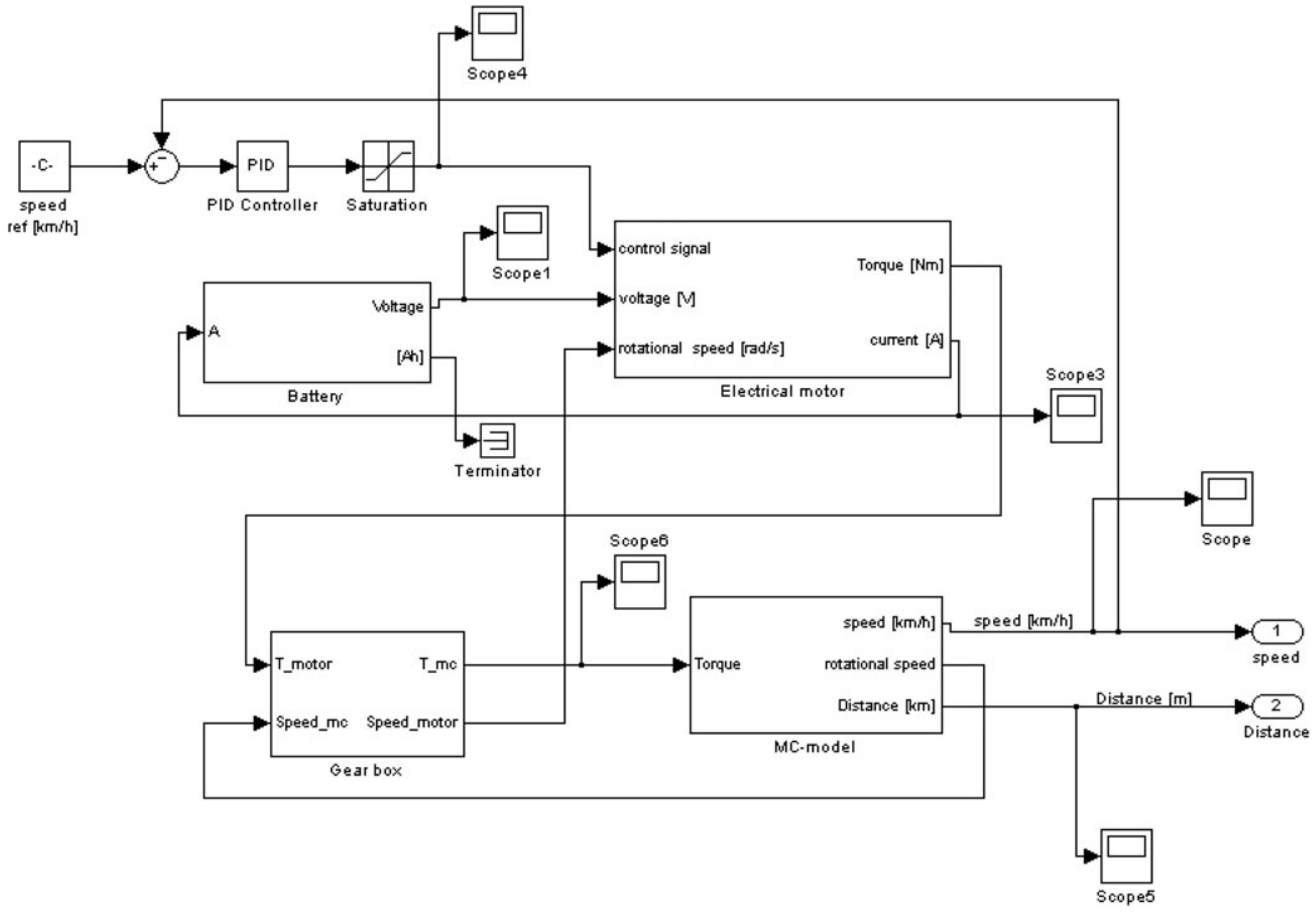

Fig. 7. Screenshot of a Simulink model of an electric motorcycle

Table 1. *Optimization problem settings*

Function	No. of Variables	Lower Bounds	Upper Bounds	δ	Successful if $f(x) <$
Branke3	2	[−1 −1]	[1 1]	±0.2	−1.4
Hartmann6	6	[0 0 0 0 0 0]	[1 1 1 1 1 1]	±0.2	−1.57
Peaks	2	[−3 −3]	[3 3]	±0.2	−5.1
Rehman	3	[−5 0 0]	[10 15 1]	±12.5%	12.2
Electrical MC	3	[1 1 10]	[10 10 60]	±12.5%	−66.6

been used in benchmarking of robust optimization by, for example, Branke (2001) and Rehman et al. (2014) or for deterministic optimization by for example Toal and Keane (2012).

The reason to use simple mathematical functions for this comparison is that they are computationally cheap to evaluate and can be used to improve the understanding of how the different methods operate (Beyer & Sendhoff, 2007). Furthermore, mathematical functions can be adapted to investigate which behaviors of the problems are easy or difficult for each method to solve (Neculai, 2008).

The randomness is introduced similar to Branke (2001) as a uniform noise, δ, on the variables as shown in Eq. (4). This means that if $\delta = \pm 0.2$ and a suggested design where $x = 1$ is analyzed, the values of $x + \delta$ can range from 0.8 to 1.2.

$$f = f(x+\delta), \\ \delta_{\min} \leq \delta \leq \delta_{\max}. \tag{4}$$

The functions used by Branke are tailored for investigating the performance of RDO for nonlinear problems and can be scalable to any number of variables. This means that it is possible to investigate how the number of variables affects the performance of the different methods. To enable scalability, each variable is independent from the others in the calculations. The total function value is the sum of the contributions from all variables as shown in Eq. (5).

$$f(x) = \sum_{i=1}^{n} f_i(x). \tag{5}$$

The robust function proposed by Rehman et al. (2014) has three variables and is not scalable. The randomness is introduced as a uniform noise on the design variables of ±1/8 (12.5%) of the variable ranges.

4.3. Comparison settings

Each robust optimization method needs to optimize each problem numerous times to collect the statistical data needed to estimate the performance measures. This number is set to 1000 optimizations in this comparison.

The performance measure in Eq. (3) needs a criterion for a successful optimization to estimate the accuracy, denoted hit rate. This comparison uses a similar criterion as the one used by Riesenthel and Lesieutre (2011): 100,000 points are spread in the design space and their objective function values calculated. The 1000th lowest value is then used as a limit, and any optimization that results in an objective function value that is lower is deemed successful. A drawback is that the 100,000 samples become more spread as the number of variables increases. This means that it is easier for an optimization to find a solution that is better than the limit for problems with many variables. It should be an adequate criterion of a successful optimization here as the test problems have approximately the same number of parameters.

4.4. Results from the comparison

The results from the 1000 optimization runs for each problem and method can be found in Appendix A. The mean value and standard deviations of the optimums found for each method and optimization problem are summarized in Table 2 together with the performance indices.

The RDO methods can be ordered according to their ranking for each of these problems and criteria to get a better overview. This is presented in Table 3, where the rankings of each method also are added together into overall scores. SMRDO, for example, places fifth, fifth, fifth, and fourth in the accuracy rankings for each problem and therefore receives an overall score of $5 + 5 + 5 + 4 = 19$.

The overall scores indicate that bfRDO, RSO, and CRDO are the three most accurate methods. They are also the methods with the highest robustness. RSO is the clear winner in terms of efficiency with the places 1, 2, 1, and 1. The overall scores therefore indicate that RSO should be the most suitable optimization method for our engineering problem if it is computationally expensive.

To investigate the appropriateness of choosing RSO, the electrical motorcycle problem is optimized by all RDO methods in the same manner as the test problems. These results can be found in the bottom of Table 2 and the rightmost column in Table 3.

Results show that bfRDO performs best on the electrical motorcycle problem in terms of both accuracy and robustness. It does however perform worse than all other methods in terms of efficiency. This means that it is unsuitable for the electrical motorcycle problem unless much computational power is accessible. It is expected that bfRDO has a low efficiency for many problems because it does not include surrogate models

Table 2. *Results from 1000 independent optimizations for the five methods for each robust optimization problem*

Function	No. of Variables	Metric	bfRDO	SMRDO	RSO	EARO	CRDO
Branke3	2	Mean	−1.24	−0.47	−1.01	−0.85	−1.05
		SD	0.44	1.64	0.44	0.38	0.54
		H	0.026	0.076	0.338	0.002	0.117
Hartmann6	6	Mean	−1.89	−0.49	−1.68	−1.08	−1.91
		SD	0.12	0.34	0.23	0.57	0.12
		η	1	0.010	0.861	0.010	0.712
Peaks	2	Mean	−4.86	−3.88	−6.08	−5.87	−5.55
		SD	2.00	2.88	0.67	1.29	1.20
		η	0.022	0.606	0.999	0.352	0.655
Rehman	3	Mean	10.73	11.56	10.09	17.82	10.52
		SD	1.32	1.73	1.13	11.61	1.38
		η	0.019	0.714	0.998	0.023	0.706
Electrical MC	3	Mean	−65.68	−62.56	−63.59	−59.97	−54.74
		SD	2.44	6.64	3.82	5.72	12.58
		η	0.005	0.110	0.503	0.015	0.044

and therefore needs to evaluate the original model many times every time the robustness of a design should be calculated.

The results indicate that RSO was an appropriate choice of optimization method. It places second behind bfRDO in terms of both accuracy and robustness, but it is the most efficient method. This means that the proposed benchmarking method succeeded in identifying the most suitable method for solving the engineering problem.

It is difficult to draw any general conclusions regarding the different RDO methods because the number of test problems are few and they all have few design variables. Some pointers can however be made.

SMRDO generally displays a bad accuracy. It places fifth, fifth, fifth, and fourth for the test problems and third for the engineering problem. The other methods improve their SMs during the optimization, and SMRDO does not. This means that the original SM needs to animate both the general appearance and the optimum of the function well. This is evidently a hard task, and hence, it is concluded that it is beneficial to improve the SM as the optimization evolves.

CRDO displays higher performance indices, η, than EARO for all problems. This shows that the proposed novel method, where the optimization algorithm is changed from a GA to the complex algorithm and the SMs from polynomial response surfaces to kriging models, is more computationally efficient. This is mainly because CRDO requires fewer simulations of the original model to converge than EARO. The Complex-RF optimization algorithm generally converges faster than a

Table 3. *Rankings of the five RDO methods for each optimization problem and performance criterion*

Accuracy Mean	Branke 3	Hartmann 6	Peaks	Rehman	Overall Score	Ranking	Electrical MC
bfRDO	1	2	4	3	10	3	1
SMRDO	5	5	5	4	19	5	3
RSO	3	3	1	1	8	1	2
EARO	4	4	2	5	15	4	4
CRDO	2	1	3	2	8	1	5
Robustness SD							
bfRDO	2	1	4	2	9	2	1
SMRDO	5	4	5	4	18	5	4
RSO	2	3	1	1	7	1	2
EARO	1	5	3	5	14	4	3
CRDO	4	1	2	3	10	3	5
Efficiency η							
bfRDO	4	1	5	5	15	4	5
SMRDO	3	4	2	2	11	3	2
RSO	1	2	1	1	5	1	1
EARO	5	4	4	4	17	5	4
CRDO	2	3	2	3	10	2	3

GA, and this is shown here as well. The drawback is that Complex-RF has a lower hit-rate than GA, but this is somewhat remedied by using kriging instead of PRS as SM. This is demonstrated by the similar or even higher hit rate for CRDO when compared with EARO. It can also be seen from the comparison of the mean values that CRDO has a better accuracy than EARO for the test problems but not for the engineering problem. However, the faster convergence yields an overall better performance measure for CRDO.

The accuracies of the SMs are crucial for the performance of the different methods. Kriging is generally a good SM, but best suited for stationary functions (Toal & Keane, 2012). It is possible that the methods would work better with other SM types or a nonstationary kriging.

The ease of use and availability of the RDO methods is not taken into account in this demonstration of the comparison method. The algorithms are quite fast to create from existing and freely available code and used with their standard settings.

5. CONCLUSIONS

This paper proposes a method that can be used to compare the performance of RDO methods in order to find the most suitable for a given problem. The comparison method is demonstrated for five RDO methods, where the fifth method (CRDO) is a novel method with a mechanism similar to one of the other methods (EARO). The proposed comparison method includes a performance index for efficiency, and it is shown that it can be used to compare different RDO methods.

This paper also proposes a novel RDO method that uses surrogate models to speed up the optimization. Every time the optimization algorithm wants to calculate the robustness of a suggested design, a deterministic simulation of the original model is performed and a surrogate model created. The robustness calculation is then performed by simulating the surrogate model instead of the original model. The proposed method is inspired by an existing method (EARO) proposed by Paenke et al. (2006) that uses a genetic algorithm as an optimization algorithm and polynomial response surfaces as surrogate models. The proposed method instead uses the Complex-RF an optimization algorithm and kriging as a surrogate model. The results show that the new method is more efficient than the method that it is a modification of.

The demonstration of the comparison method uses four mathematical functions to identify the most suitable algorithm for solving a low-dimensional engineering problem. The comparison suggests that robust sequential optimization is the most appropriate method to use. This selection is confirmed to be correct according to verification optimizations of the engineering problem with all RDO methods.

If the model that should be optimized is computationally efficient, no SMs are needed, and the problem can be solved without using SMs. The more computationally demanding the original model is to simulate, the more important it is that each simulation contributes to the solving of the optimization problem. This means that more calculations can be made by the optimization algorithm between each simulation to ensure that only necessary simulations are performed.

The algorithms that update the SMs during the optimizations are all dependent on accurate SMs in the beginning of the optimization process. The algorithms may otherwise search for solutions in the wrong region of the design space and never recover. This can be somewhat remedied by different updating schemes for the SMs (see, e.g., Jones 2001). It is possible that the algorithms can be improved further by incorporating these updating schemes, and this warrants further research.

The example comparison is made using an objective function where only the mean value of a design is considered, and hence the calculation of the standard deviation is not taken into account. If, however, objective functions are used where the standard deviation is included, other methods, for example, Taylor expansions of the standard deviation, should be considered as well.

REFERENCES

Aspenberg, D., Jergeus, J., & Nilsson, L. (2013). Robust optimization of front members in a full frontal car impact. *Engineering Optimization 45(3)*, 245–264.

Beyer, H.G., & Sendhoff, B. (2007). Robust optimization—a comprehensive survey. *Computer Methods in Applied Mechanics and Engineering 196(33)*, 3190–3218.

Box, M.J. (1965). A new method of constrained optimization and a comparison with other methods. *Computer Journal 8(1)*, 42–52.

Branke, J. (2001). *Evolutionary Optimization in Dynamic Environments*. Norwell, MA: Kluwer Academic.

Coelho, R.F. (2014). Metamodels for mixed variables based on moving least squares. *Optimization and Engineering 15(2)*, 311–329.

Eberhart, R.C., & Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proc. 6th Int. Symp. Micro Machine and Human Science*, Vol. 1, pp. 39–43, Nagoya, Japan, October 4–6.

Forrester, A., Sobester, A., & Keane, A. (2008). *Engineering Design Via Surrogate Modelling: A Practical Guide*. Hoboken, NJ: Wiley.

Gobbi, M., Guarneri, P., Scala, L., & Scotti, L. (2014). A local approximation based multi-objective optimization algorithm with applications. *Optimization and Engineering 15(3)*, 619–641.

Goldberg, D.E. (2006). *Genetic Algorithms*. Delhi: Pearson Education India.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI: University of Michigan Press.

Jin, R., Du, X., & Chen, W. (2003). The use of metamodeling techniques for optimization under uncertainty. *Structural and Multidisciplinary Optimization 25(2)*, 99–116.

Jones, D.R. (2001). A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization 21(4)*, 345–383.

Krus, P., & Ölvander, J. (2013). Performance index and meta-optimization of a direct search optimization method. *Engineering Optimization 45(10)*, 1167–1185.

McKay, M.D., Beckman, R.J., & Conover, W.J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics 21(2)*, 239–245.

Mercer, R.E., & Sampson, J.R. (1978). Adaptive search using a reproductive metaplan. *Kybernetes 7(3)*, 215–228. doi:10.1108/eb005486

Myers, R.H., Montgomery, D.C., & Anderson-Cook, C.M. (2009). *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. Hoboken, NJ: Wiley.

Neculai, A. (2008). An unconstrained optimization test functions collection. *Advanced Modeling and Optimization 10(1)*, 147–161.

Nelder, J.A., & Mead, R. (1965). A simplex method for function minimization. *Computer Journal 7(4)*, 308–313.

Ölvander, J., & Krus, P. (2006). Optimizing the optimization—a method for comparison of optimization algorithms. *Proc. AIAA Multidisciplinary*

Design Optimization Specialists Conf., Paper No. AIAA 2006-1915, Newport, RI, May 1–4. doi:10.2514/6.2006-1915

Paenke, I., Branke, J., & Jin, Y. (2006). Efficient search for robust solutions by means of evolutionary algorithms and fitness approximation. *IEEE Transactions on Evolutionary Computation 10(4)*, 405–420.

Persson, J., & Ölvander, J. (2011). Comparison of sampling methods for a dynamic pressure regulator. *Proc. 49th AIAA Aerospace Sciences Meeting*, Paper No. AIAA 2011-1205, Orlando, FL, January 4–7. doi:10.2514/6.2011-1205

Persson, J.A., & Ölvander, J. (2013). Comparison of different uses of metamodels for robust design optimization. *Proc. 51th AIAA Aerospace Sciences Meeting*, Paper No. AIAA 2013-1039, Grapevine, TX, January 7–10. doi:10.2514/6.2013-1039

Persson, J.A., & Ölvander, J. (2015). Optimization of the complex-RFM optimization algorithm. *Optimization and Engineering 16(1)*, 27–48.

Rehman, S., Langelaar, M., & van Keulen, F. (2014). Efficient kriging-based robust optimization of unconstrained problems. *Journal of Computational Science*. Advance online publication. doi:10.1016/j.jocs.2014.04.005

Reisenthel, P.H., & Lesieutre, D. J. (2011). A numerical experiment on allocating resources between design of experiment samples and surrogate-based optimization infills. *Proc. 52ndAIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conf.*, Paper No. AIAA 2011-2150, Denver, CO, April, 4–7.

Schutte, J.F., & Haftka, R.T. (2005). Improved global convergence probability using independent swarms. *Proc. 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conf., Structures, Structural Dynamics, and Materials and Co-located Conf.*, Austin, TX. doi:10.2514/6.2005-1896

Tarkian, M., Persson, J., Ölvander, J., & Feng, X. (2012). Multidisciplinary design optimization of modular industrial robots by utilizing high level CAD templates. *Journal of Mechanical Design 134(12)*, 124502.

Tenne, Y. (2015). An adaptive-topology ensemble algorithm for engineering optimization problems. *Optimization and Engineering 16(2)*, 303–334.

Toal, D.J.J., & Keane, A.J. (2012). Non-stationary kriging for design optimization. *Engineering Optimization 44(6)*, 741–765.

Wang, G.G., & Shan, S. (2007). Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical Design 129(4)*, 370–380.

Wiebenga, J.H., Van Den Boogaard, A.H., & Klaseboer, G. (2012). Sequential robust optimization of a V-bending process using numerical simulations. *Structural and Multidisciplinary Optimization 46(1)*, 137–153.

Johan A. Persson is a Senior Lecturer in the Department of Management and Engineering at Linköping University. His areas of interest are engineering optimization, especially surrogate modeling and robust optimization.

Johan Ölvander is a Professor of mechanical engineering at Linköping University, where he obtained his PhD. His research interests are multidisciplinary optimization, simulation-based engineering design, design automation, and product development. Dr. Ölvander has coauthored more than 100 papers, including 30 articles in archival journals and 80 in scientific conference proceedings.

APPENDIX A

Table A.1. *Data from the comparison*

	bfRDO			SMRDO		
Function	Calls	Hit-Rate	Perf Index	Calls	Hit-Rate	Perf Index
Branke3	3540	0.609	0.026	100	0.076	0.076
Hartmann6	10000	1	1	100	0.606	0.606
Peaks	4700	0.655	0.022	100	0.009	0.009
Rehman	9960	0.850	0.019	100	0.714	0.714
Electrical MC	9960	0.410	0.005	100	0.110	0.110

Table A.2. *Functions*

	RSO			EARO		
Function	Calls	Hit-Rate	Perf Index	Calls	Hit-Rate	Perf Index
Branke3	51	0.190	0.338	624	0.014	0.002
Hartmann6	61	0.700	0.861	2618	0.228	0.010
Peaks	43	0.960	0.999	633	0.936	0.352
Rehman	48	0.950	0.998	2444	0.428	0.023
Electrical MC	51	0.300	0.503	573	0.084	0.015

Table A.3. *Functions*

	CRDO		
Function	Calls	Hit-Rate	Perf Index
Branke	445	0.424	0.117
Hartmann	499	0.998	0.712
Peaks	182	0.856	0.655
Rehman	179	0.888	0.706
Electrical MC	214	0.091	0.044

APPENDIX B

Table B.1. *Functions*

Function	No. of Variables	Overall Behavior	No of Determin. Extreme Pts.
Branke	2	Discrete	1
Hartmann	6	Continuous	Multiple
Peaks	2	Continuous	Multiple
Rehman	3	Continuous	Multiple
Electrical MC	3	Continuous, noisy	1

Branke3

$$f(x_j) = \begin{cases} x_j + 0.8 & \text{if } -0.8 \le x_j < 0.2 \\ 0 & \text{otherwise} \end{cases}. \tag{A.1}$$

Hartmann6

$$f(x) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - P_{ij})^2\right), \tag{A.2}$$

where

$$\alpha = (1.0, 1.2, 3.0, 3.2)^T,$$

$$A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix},$$

$$P = 10^{-4}\begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}.$$

Peaks

$$\begin{aligned} f(x) = {} & 3(1-x_1)^2 \exp[-x_1^2 - (x_2+1)^2] \\ & - 10\left(\frac{{}^3x_1}{5} - x_1^3 - x_2^5\right)\exp[-x_1^2 - x_2^2] \\ & - \frac{1}{3}\exp[-(x_1+1)^2 - x_2^2]. \end{aligned} \tag{A.3}$$

Rehman

$$\begin{aligned} f(x) = {} & \left(x_2 - \frac{5.1}{4\pi}x_2 + \frac{5}{\pi}x_1 - 6\right)^2 \\ & + 10\left(\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 1\right) \\ & + (6x_3 - 2)^2 \sin(12x_3 - 4) + 8x_3. \end{aligned} \tag{A.4}$$