White Paper

Benchmark of HyperStudy Optimization Algorithms

Zhifan Luo – Technical Specialist, Altair
Altair Engineering 3303 Monte Villa Parkway, Suite 320 Bothell, WA 98021

1. Introduction

Automated design optimization technology has been applied to many engineering domains and many well-known algorithms and commercial software options are available. For a given optimization problem, different optimization methods will perform distinctively. Choosing the appropriate optimization method for a given problem depends on the type of optimization problem formulation and characteristics of the design space that are determined by the set of variables, objectives, and constraints. However, the exact nature of the optimization design space is still generally unknown until after it has been explored, if at all. Selecting the most appropriate algorithm and then tuning its parameters for peak performance on a specific problem is a time consuming process. Furthermore, a set of parameters tuned for one problem is not likely to perform optimally on a different optimization problem. Therefore, a desirable characteristic of an optimization algorithm is the ability to perform well on a wide range of problems which avoids a time consuming manual tuning process.

The efficiency of an optimization method is also very important. Engineering applications often involve expensive model evaluations. It is attractive if an algorithm can find an optimized design with fewer model evaluations.

The objective of this paper is to assess several optimization algorithms in HyperStudy for their effectiveness and efficiency. The following sections of this paper present an overview of the optimization algorithms frequently used in HyperStudy. This is followed by benchmarking of both single objective and multi-objective optimization problems, respectively. Conclusions are made in the final section. The data shown in this study was collected using HyperStudy

2. Overview of the HyperStudy Optimization Methods

The HyperStudy optimization methods used within this benchmark are briefly described in this section. All algorithm parameters are left as default unless otherwise specified.

2.1. GRSM

GRSM (Global Response Surface Method) is a proprietary optimization method available in HyperStudy [1]. In each iteration, the response surface based optimization generates several designs. Additional designs are generated globally to ensure good balance on local search efficiency and global search capability. All the designs generated in one iteration can be solved in parallel. In each iteration, the response surface is adaptively updated with the newly generated designs to have a better approximation of the original model.

GRSM can be applied to a wide range of optimization problems. It can work with either single objective or multiple objective optimization problems, and it supports continuous, discrete, and mixed variables. The inclusion of global search features places GRSM into the category of exploratory optimization algorithms. This means that the method does not show the numerical convergence characteristics typically observed in other algorithms, such as gradient based schemes. Similar to some other exploratory schemes it terminates only after a fixed number of evaluations.

In this study, the algorithmic parameter “Random Seed” is varied to test the robustness of the algorithm.

2.2. ARSM

ARSM (Adaptive Response Surface Method) works by internally building response surfaces and continually updating them as new evaluation data becomes available [1]. The first response surface built is a linear regression, then it finds the optimum on this surface and validates the prediction with an exact simulation. If the response values from the response surface and the exact simulation are not sufficiently close, ARSM updates the surface with the new evaluation, increasing the regression complexity, and then finds the optimum on this updated surface. ARSM repeats this loop until it meets one of the convergence criteria.

In contrast to GRSM, ARSM exhibits numerical convergence to a local optimum and does not contain global search capability. ARSM can be used for single objective optimization problems only. It supports continuous, discrete and mixed variables.

2.3. SQP

SQP (Sequential Quadratic Programming) is a gradient-based iterative optimization method [2]. It is generally an efficient algorithm for solving local optimization problems in which the objective function and all constraints are smooth.
However, it is often not effective for multi-modal or non-smooth problems. SQP can be used for single objective optimization problem only, and it cannot accommodate discrete variables.

2.4. GA

GA (Genetic Algorithm) is implemented based on the evolutionary process theory [3]. GA starts with the creation of a population of designs, known as a generation. These designs are then ranked with respect to their fitness, which is a measure of design's goodness, and is calculated as a function of both constraint violation and objective function values. Selected designs are then reproduced through the application of genetic operators, typically crossover and mutation. The individuals that result from this process, known as the children, become members of the next generation. This process is repeated for many generations until the evolution of a population converges to the optimal solution.

GA's are very good at exploring multi-modal design spaces. Yet they are generally not efficient in local search, which slows the convergence rate considerably. The GA in HyperStudy has a hybrid scheme which greatly increases the local search capability. GA can be used for single objective optimization problems only. It supports continuous, discrete and mixed variables. In this study, the algorithmic parameter "Random Seed" is varied to test the robustness of the algorithm.

2.5. MOGA

MOGA (Multi-Objective Genetic Algorithm) is an extension of Genetic Algorithm that solves multi-objective optimization (MOO) problems. MOGA is only applicable to multi-objective optimization problem. It supports continuous, discrete and mixed variables.
In this study, the algorithmic parameter "Random Seed" is varied to test the robustness of the algorithm.

3. Benchmarking of Single Objective Problems

In this section, five single objective optimization examples from a published reference are utilized to benchmark the performance of the optimization algorithms including GRSM, ARSM, SQP and GA [4]. These benchmark problems exhibit particular sets of features that are representative of challenging optimization problems that could exhibit difficulty in converging to the optimal design. As a result, they may require a larger number of evaluations than a typical engineering design optimization application. These examples are re-presented in this section for convenience.

For each problem, ten runs are carried out for each algorithm with the starting designs varied over a wide range within the design space. This variation is done to assess the robustness of the results obtained by each algorithm, as well as to ensure that the results are not biased based by a given set of starting designs. The solutions of these multiple runs are then averaged. The standard deviation of the differing solutions is also presented to better understand the robustness of an algorithm. To be more easily interpreted, the average solution and the standard deviation to each problem is transferred using formulas (1) and (2), shown below. Hence, the ideal transferred average solution and transferred standard deviation to each problem should approach the value 0.

$f\_{transfer}=\ln(|f'-f|+1)$
$s\_{transfer}=\ln(s+1)$

where, f' is the best solution found by optimization algorithm, f is the known optimal solution, and s is the standard deviation.

3.1 Goldstein-Price's Function

The Goldstein-Price Function is slightly multi-modal and continuous. This function is defined as:

$$f(x_1,x_2)=[1+(x_1+x_2+1)^2\times(19-14x_1+3x_1^2 -14x_2+6x_1 x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2 \times(18-32x_1+12x_1^2+48x_2-36x_1 x_2+27x_2^2)]$$

where $-2\leq x_i\leq 2$. The global minimum occurs at f(0,-1)=3. Figure 1 shows the global minimum location and the multi-modality around the global minimum.

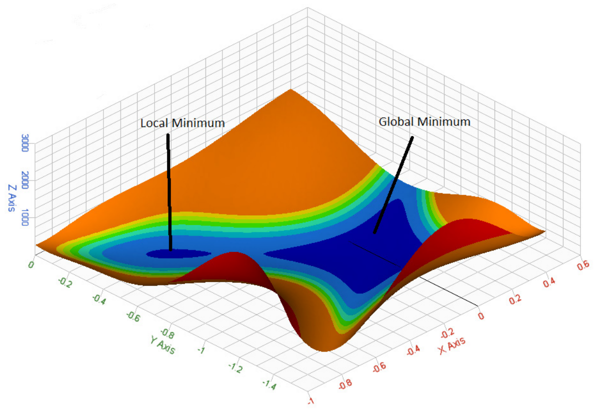


Figure 1: Three-dimensional contour plot of the local region surrounding the global minimum location of the Goldstein-Price Function.

The maximum number of evaluations is set to 200 in this study. Optimization results of GRSM, ARSM, SQP and GA are plotted in figures 2 and 3. SQP did not perform well on this example. It cannot make additional progress beyond about 120 evaluations, because it has already converged. Gradient-based methods are not able to explore more than one local minimum at a time, and they tend to be trapped by the nearest local minimum. ARSM did not perform well in this example since it also tends to be stuck at the local minimum. GA shows relatively slow convergence in this example. GRSM is nearly converged to the global optimal for all of the ten runs within 150 evaluations, and exhibits little variation in the found optimum.

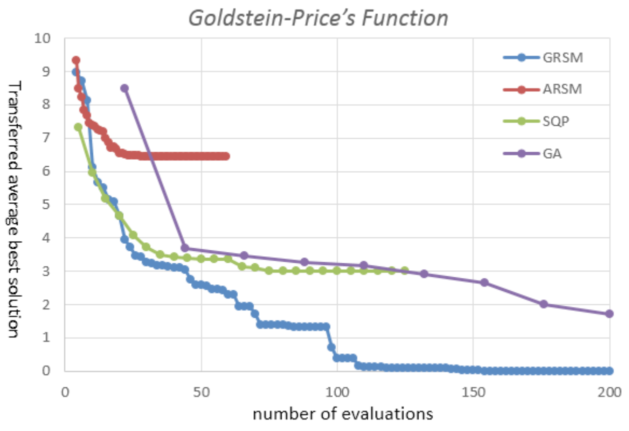


Figure 2: Transferred average best solution vs. number of evaluations for the Goldstein-Price function..

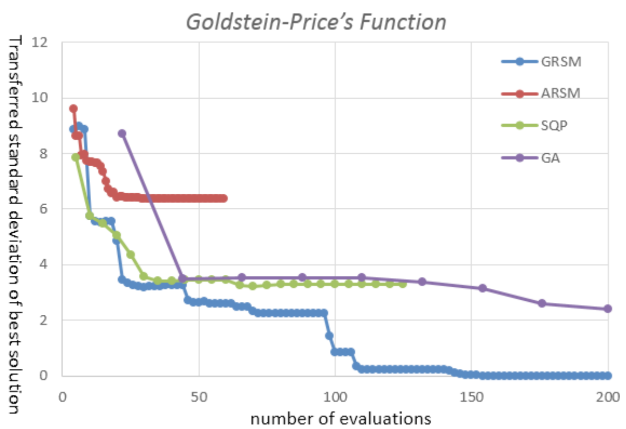


Figure 3: Transferred standard deviation of the best solution vs. number of evaluations for the Goldstein-Price function.

3.2. Rosenbrock's Valley

Rosenbrock's Valley is also known as a banana function due the shape of the objective function. The global optimum is in a long, narrow valley that curves parabolically along its axis, like a banana. Finding the valley itself is simple, but identifying the global minimum is difficult because the valley is non-linear with strong interactions among the variables. Many algorithms converge slowly because they require repeated changes in their search direction.

This function is defined as:

$$f(x_i)=\sum_{i=1}^{n-1}\left[100(x_{i+1} - x_i^2)^2+(1 - x_i)^2\right]$$

where $-2.048 \leq x_i \leq 2.048$. Figure 4 shows this function for two design variables. In this paper, five design variables are chosen (x_1, x_2, x_3, x_4, x_5). The global optimum value is known to be f = 0.0, which is located at (1.0, 1.0, 1.0, 1.0, 1.0).

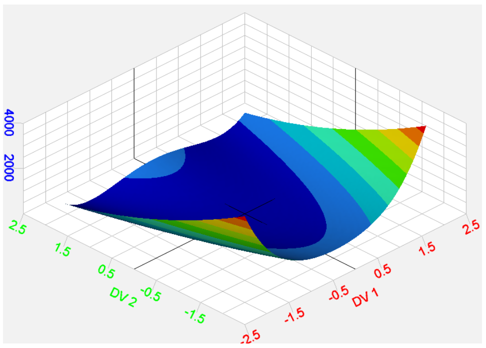


Figure 4:Three-dimensional plot of Rosenbrock's Valley for two design variables.

The maximum number of evaluations is set to 400 in this study. Optimization results of GRSM, ARSM, SQP and GA are plotted in figures 5 and 6. For this example, GRSM is the only method that finds the optimal solution in all ten of the runs within 400 evaluations. By 340 evaluations, GRSM is nearly converged to the optimum for all the runs, and there is little variation in the standard deviation. SQP performs relatively well in the early search. However it cannot make additional progress beyond 360 evaluations, as it struggles to traverse the valley. ARSM reaches its convergence condition very early, at approximately 65 evaluation but the solution is far from the optimal design. GA shows limited progress after 300 evaluations.

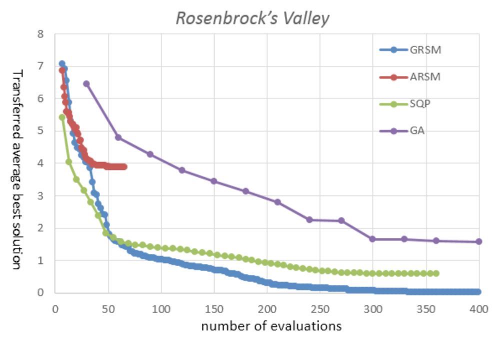


Figure 5: Transferred average best solution vs. number of evaluations for the Rosenbrock's function.

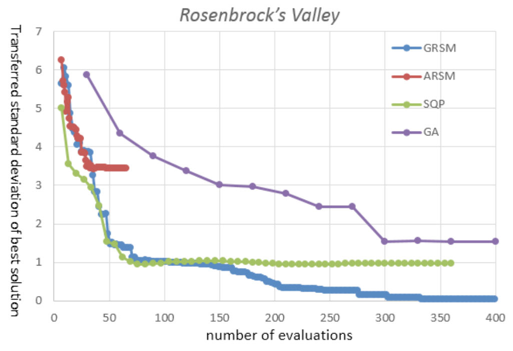


Figure 6: Transferred standard deviation of the best solution vs. number of evaluations for the Rosenbrock's function.

3.3. Cantilevered Beam Problem with Continuous Variables

A cantilevered I-beam subjected to a tip load is shown in Figure 7. The goal is to design the cross-sectional shape of the I-beam such that a minimum volume solution that also satisfies constraints on the stress and deflection is found. The design variables are identified in the diagram.

The objective function to be minimized is the beam volume

$$f(H,h_1,b_1,b_2)=L(2h_1\, b_1+b_2\, (H-2h_1))$$

The constraint functions are defined as:

$$\sigma : \frac{PLH}{(2I)} \le 5000$$

$$\delta : \frac{PL^3}{(3EI)} \le 0.1$$

σ is the maximum bending stress at the fixed end of the beam, and δ is the maximum deflection at the tip of the beam. P=1000 is the applied load, $E=10^7$ is the material modulus, L=36 is the length of the beam, and the moment of inertia of the beam cross

section is $I = \frac{b_2\,(H-2h_1)^3}{12} + 2\left(\frac{b_1 h_1^3 + b_1 h_1 (H-h_1)^2/4}{12}\right)$

The variables are allowed to vary within the ranges listed as below.

$3.0 \le H \le 7.$

$0.1 \le h_1 \le 1.0$

$2.0 \le b_1 \le 12.0$

$0.1 \le b_2 \le 2.0$

The global minimum has a value f(7.0 , 0.1, 9.48482, 0.1) = 92.77.

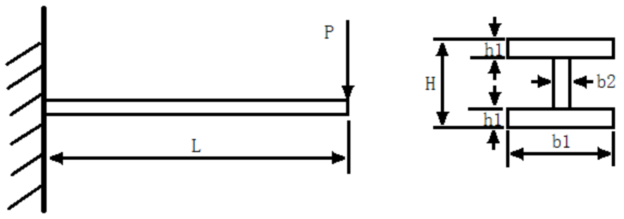


Figure 7: Cross-sectional shape variables in the cantilevered I-beam with a tip load.

In this study, maximum number of evaluations is set to 200. Optimization results of GRSM, ARSM, SQP and GA are plotted in figures 8 and 9. ARSM performs well in this example. It approaches quickly to the optimum and is converged at approximately 65 evaluations. GRSM and SQP also exhibit good performance. They are nearly converged to the optimum for all of the ten runs within 80 evaluations.

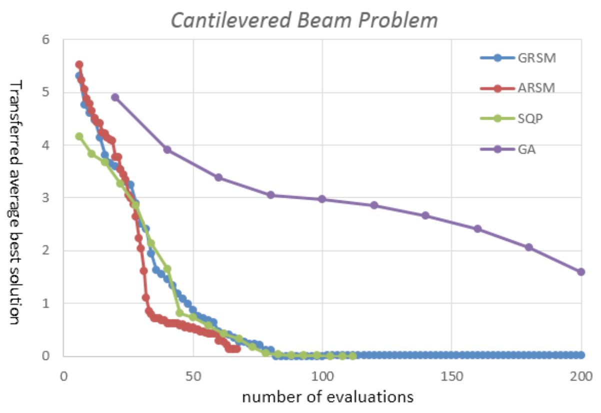


Figure 8: Transferred average best solution vs. number of evaluations for the cantilevered beam problem.

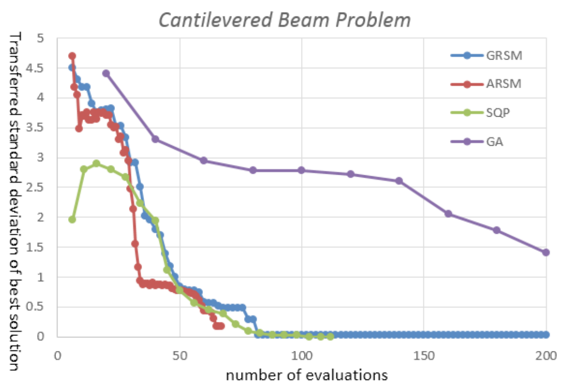


Figure 9: Transferred standard deviation of the best solution vs. number of evaluations for the cantilevered beam problem.

3.4. Cantilevered Beam Problem with Mixed Variables

This problem is identical to the cantilevered beam problem described in the above example, with the exception that one of the variables, the flange thicknessh_1, is now discrete. Engineering problems generally have some continuous and some discrete variables. When this is the case, the problem is said to contain mixed variables. SQP cannot be applied to this example. This variable here may only take eight specific values from the predefined set {0.1, 0.25, 0.35, 0.5, 0.65, 0.75, 0.9, 1.0}.

In this study, maximum number of evaluations is set to 200. Optimization results of GRSM, ARSM and GA are plotted in figures 10 and 11.

For this example, GRSM is the only method to find the optimal solutions for all the ten runs within 200 evaluations. GRSM is nearly converged to the optimal design for all the runs and with little variation within approximately 100 evaluations.

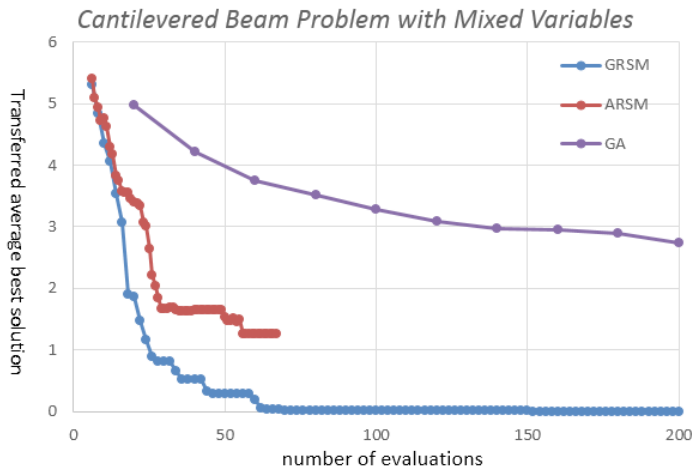


Figure 10: Transferred average best solution vs. number of evaluations for the cantilevered beam problem with mixed variables.

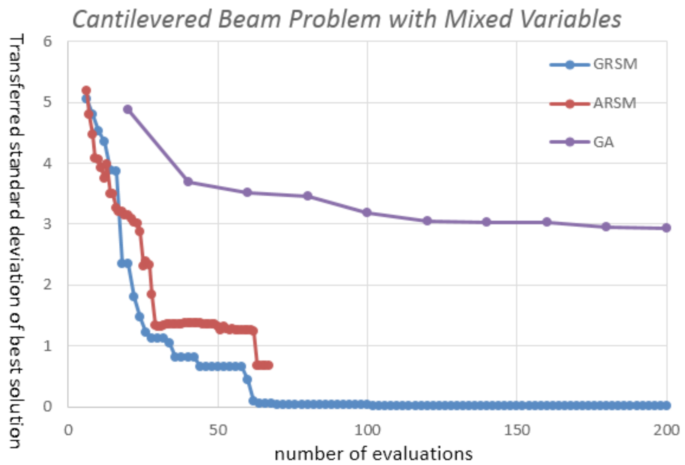


Figure 11: Transferred standard deviation of best solution vs. number of evaluations for the cantilevered beam problem with mixed variables.

3.5. Six Hump Camel Back Function

The Six Hump Camel Back Function is multi-modal and has two variables. It has six local minima, two of which are global minima. This function is defined as:

$$f(x_1 , x_2) = x_1^2\ (4 - 2.1x_1^2 + \frac{x_1^4}{3} + x_1 x_2 + x_2^2 (4x^2_2 - 4)$$

$$-3.0 \leq x_1 \leq 3.0$$

$$-2.0 \leq x_2 \leq 2.0$$

Figure 12 illustrates the multi-modality of the problem around the optima. The two global minima have a value -1.03164 at the locations (x_1, x_2) = (0.0898, -0.7126) and (-0.0898, 0.7126).

In this study, maximum number of evaluations is set to 50. Optimization results of GRSM, ARSM, SQP and GA are plotted in figures 13 and 14. For this example, GRSM is the only method to find the optimal solutions for all the ten runs within 50 evaluations. ARSM and SQP do not perform well on this example. They are trapped by local minimums in some runs.

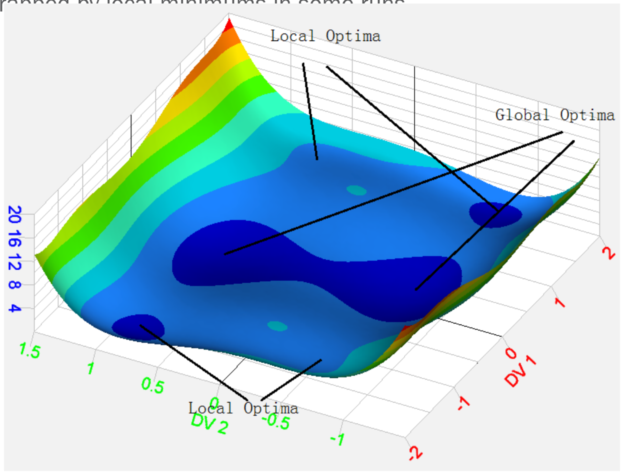


Figure 12: Three-dimensional contour plot of the local region surrounding the global minima locations of the Six Hump Camel Back Function.

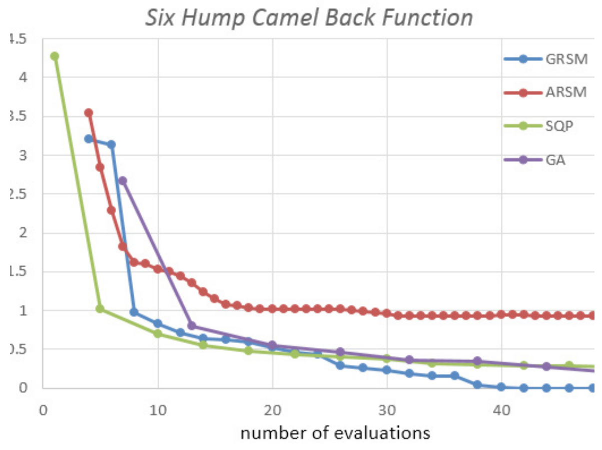


Figure 13: Transferred average best solution vs. number of evaluations for the six hump camel back function.

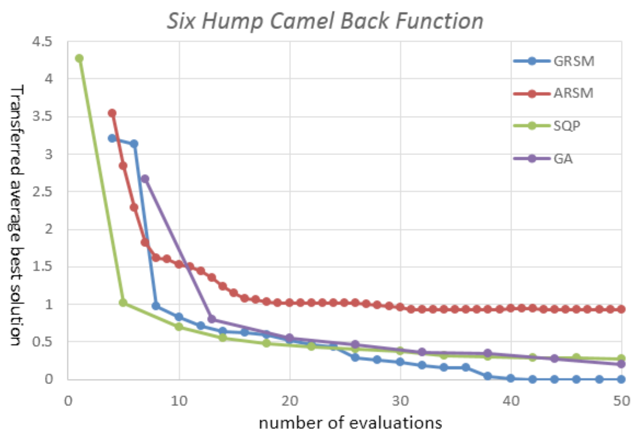


Figure 14: Transferred standard deviation of best solution vs. number of evaluations for the six hump camel back function.

4. Benchmark Testing on Multi-Objective Problem

In this section, five examples from published references are utilized to benchmark the performance of the multi-objective optimization methods GRSM and MOGA [5,6]. These test functions contain particular features that are representative of a real world optimization problem that could cause difficulty in converging to the Pareto-front. These examples will be re-presented in this section for convenience.
For each problem, five runs are carried out for each algorithm with the starting designs varying over a wide range of the design space. These five runs are to assess the robustness of the results obtained by each algorithm, as well as to ensure that the results are not biased based on a given set of starting designs.

There is no universally accepted way to compare the performance of multi-objective optimization approaches since the optimization results are a set of points distributed on the Pareto-front. In this section, the results of all five runs for each method are superimposed in each plot so that it is possible to see the variation of the results among the different runs.

The performance of each method is evaluated mainly by the placement of the non-dominated points relative to the Pareto-front. The closer the algorithm is to the Pareto-front, the better performing the algorithm. The robustness of an optimization method can be measured using the breadth of the band formed from its five runs. The narrower the band of the algorithm, the more robust it is in finding a particular front, and the less dependent on the given starting design and the stochastic variation introduced by the random number sequence [5].

4.1. ZDT1 function

ZDT1 function has 30 design variables and multiple local Pareto-fronts. The optimization problem is formulated as below.

$$\min \ f_1(x_1) = x_1$$

$$\min \ f_2(x_i) = g(x_1)\left(1 - \sqrt{\frac{f_1(x_1)}{g(x_i)}}\right)$$

$$g(x_i) = 1 + \frac{9}{29}\,\Sigma_{i=2}^{30}$$

$$0 \leq x_i \leq 1 \text{ for } i=1,\ldots, 30$$

Figure 15 plots the true Pareto-front of this example.

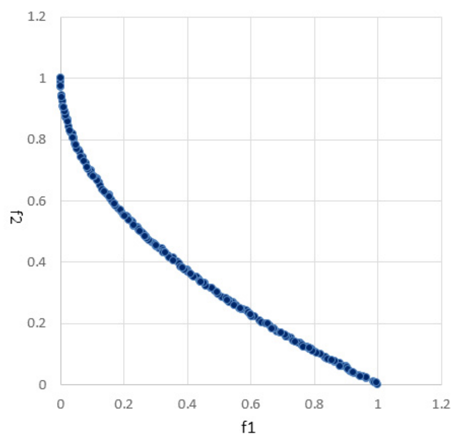


Figure 15: The Pareto-front of ZDT1 function.

Four different values of the maximum number of evaluations including 200, 300, 400 and 500 are tried in this benchmark test. Superimposed results are plotted in Figure 16. For this example, GRSM dramatically outperforms MOGA. GRSM approaches to the true Pareto-front within 200 evaluations, while M

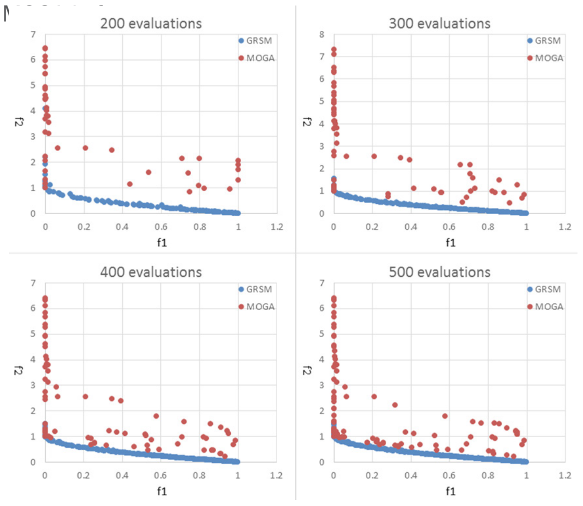


Figure 16 : Superimposed results of 5 runs each of GRSM and MOGA on ZDT1 function.

4.2. ZDT2 function

ZDT2 function has 30 design variables and multiple local Pareto-fronts. The optimization problem is formulated as below.

$$\min\ f_1(x_1) = x_1$$

$$\min\ f_2(x_i) = g(x_i)\left(1 - \left(\frac{f_1(x_1)}{g(x_i)}\right)^2\right)$$

$$g(x_i) = 1 + \frac{9}{29}\,\Sigma_{i=2}^{30}\ x_i,$$

$$0 \leq x_i \leq 1 \text{ for } i=1,\ldots,30$$

Figure 17 shows the true Pareto-front of this example.

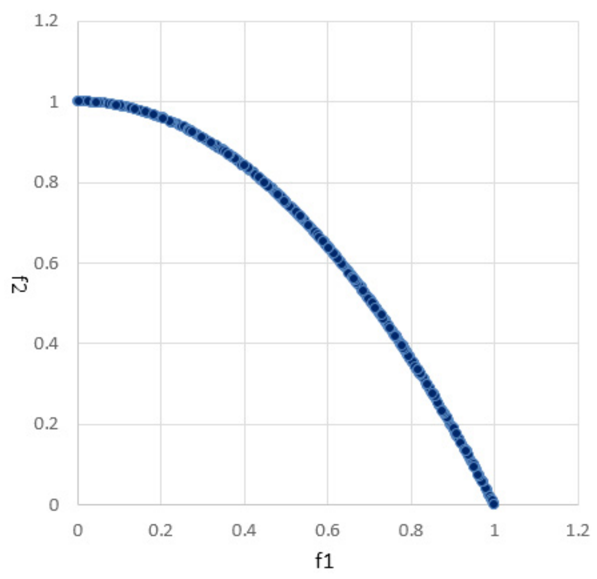


Figure 17: The Pareto-front of ZDT2 function

This paper tries four different values of the maximum number of evaluations including 200, 300, 400 and 500. Superimposed results are plotted in Figure 18. GRSM is nearly converged to the true Pareto-front within 200 evaluations, while MOGA is still unconverged. GRSM outperforms than MOGA in this example.

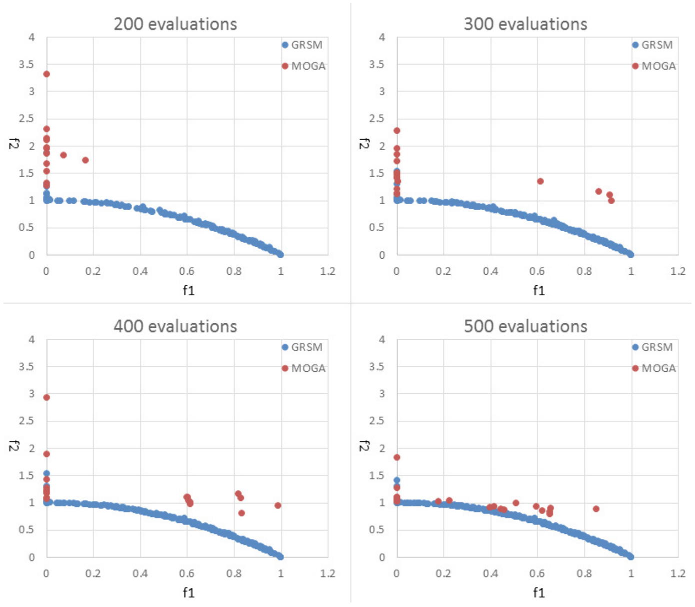


Figure 18: Superimposed results of 5 runs each of GRSM and MOGA on ZDT2 function.

4.3. ZDT3 function

ZDT3 function has 30 design variables and multiple discontinuous Pareto fronts. The optimization problem is formulated as below.

$$\min \; f_1(x_1) = x_1$$

$$\min \; f_2(x_i) = g(x_i)\left(1 - \sqrt{\frac{f_1(x_1)}{g(x_i)}} - \sqrt{\frac{f_1(x_1)}{g(x_i)}} \sin(10\pi f_1(x_1))\right)$$

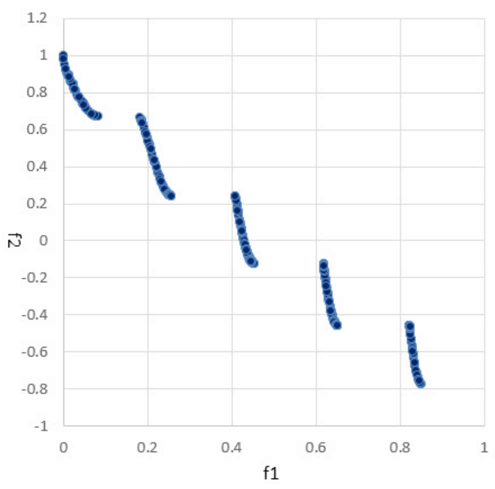
$$g(x_i) = 1 + \frac{9}{29}\Sigma_{i=2}^{30}$$

$$0 \le x_i \le 1 \text{ for } i=1,\dots,30$$

The true Pareto front for this problem is discontinuous, which can make the solution challenging. Figure 19 illustrate the Pareto-front of this example.

Four different values of the maximum number of evaluations including 250, 500, 750 and 1000 are tried in this paper. Superimposed results are plotted in Figure 20. For this example, GRSM approaches to the true Pareto-front within 250 evaluations, while MOGA has not. GRSM is nearly converged to the true Pareto-front within 750 evaluations, yet still MOGA does not converge to the true Pareto-front within 1000 evaluations. It can be seen from the results that GRSM dramatically outperforms MOGA in this example.

Figure 19: The Pareto-front of ZDT3 function



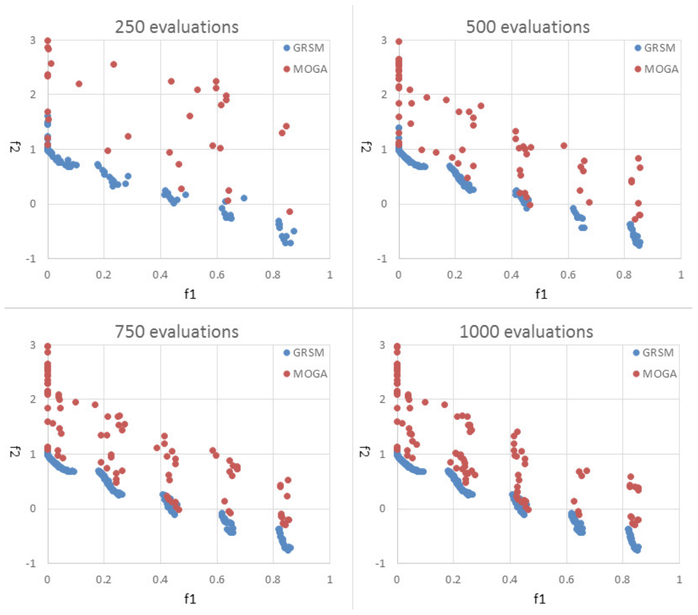


Figure 20 : Superimposed results of 5 runs each of GRSM and MOGA on ZDT3 function

4.4. MOP-C4 function

The MOP-C4 function is formulated as below.

$$\min \ f_1(x_1) = x_1$$

$$\min \ f_2(x_1) = x_2$$

Such that

$$x_1^2 + x_2^2 - 1 - 0.1\cos\left(16\arctan\left(\frac{x_1}{x_2}\right)\right) \geq 0$$

$$0 \leq x_1, x_2 \leq \pi$$

The Pareto-front of MOP-C4 function is discontinuous. The shape of the Pareto-front is highly determined by the constraint. Figure 21 shows the true Pareto-front of this example.

This paper tries two different values of the maximum number of evaluations including 100 and 200. Superimposed results are plotted in Figure 22. For this example, GRSM is nearly converged to the true Pareto-front within 100 evaluations. MOGA only find a few points close to the true Pareto-front within 200 evaluations. It can be seen from the results that GRSM performs better than MOGA in this example.

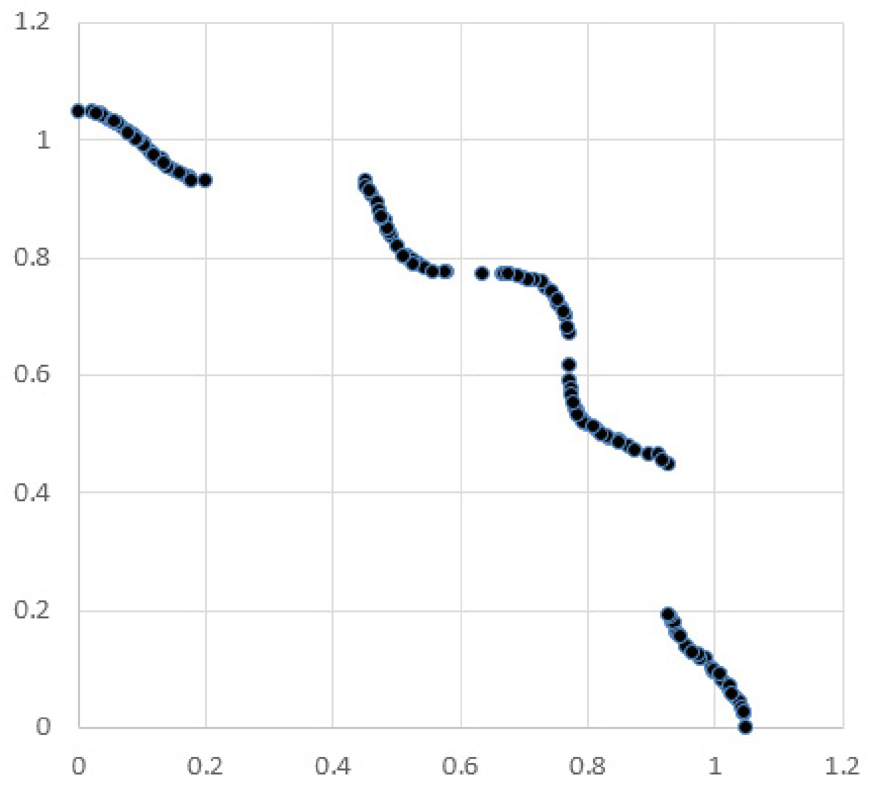


Figure 21: The Pareto-front of MOP-C4 function.

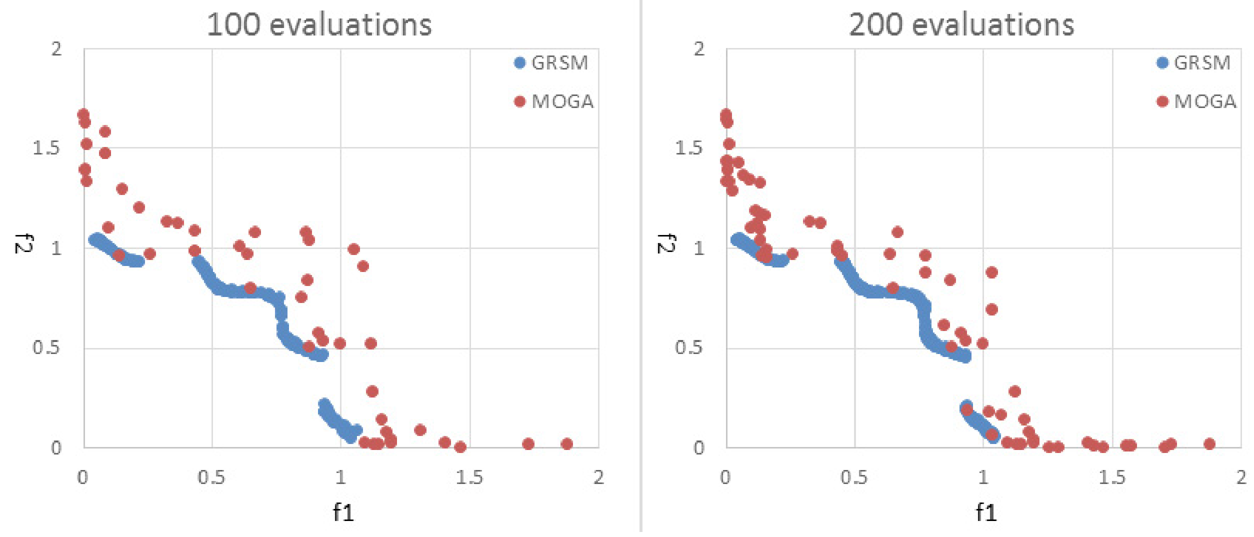


Figure 22: Superimposed results of 5 runs each of GRSM and MOGA on MOP-C4 function.

4.5. OSY function

The OSY function is formulated as:

$$\min f_1(x_1) = -[25(x_1-2)^2 + (x_2-2)^2 + (x_3-1)^2 + (x_4-4)^2 + (x_5-1)^2]$$

$$\min f_2(x_1) = \sum_{i=1}^{6} x_i^2$$

such that

$x_1 + x_2 - 2 \geq 0$

$6 - x_1 - x_2 \geq 0$

$2 + x_1 - x_2 \geq 0$

$2 - x_1 + 3x_2 \geq 0$

$4 - (x_3 - 3)^2 - x_4 \geq 0$

$(x_5 - 3)^2 + x_6 - 4 \geq 0$

$0 \leq x_1, x_2, x_6 \leq 10$

$1 \leq x_3, x_5 \leq 5$

$0 \leq x_4 \leq 6$

In the OSY function, the constraints divide the Pareto front into five regions which creates difficulties for optimization algorithms to find all parts of the Pareto front. Figure 23 plots the Pareto-front of the OSY function.

Two different values of the maximum number of evaluations including 100 and 200 are tried in this paper. Superimposed results are plotted in Figure 24., GRSM dramatically outperforms MOGA as GRSM is nearly converged to the true Pareto-front within 200 evaluations, while MOGA yet far away.

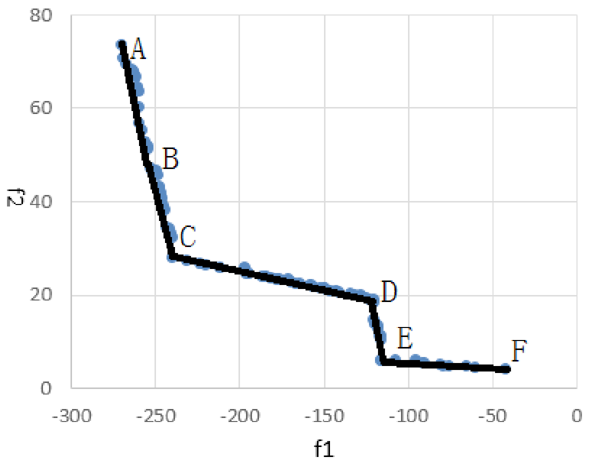


Figure 23: The Pareto-front of OSY function.

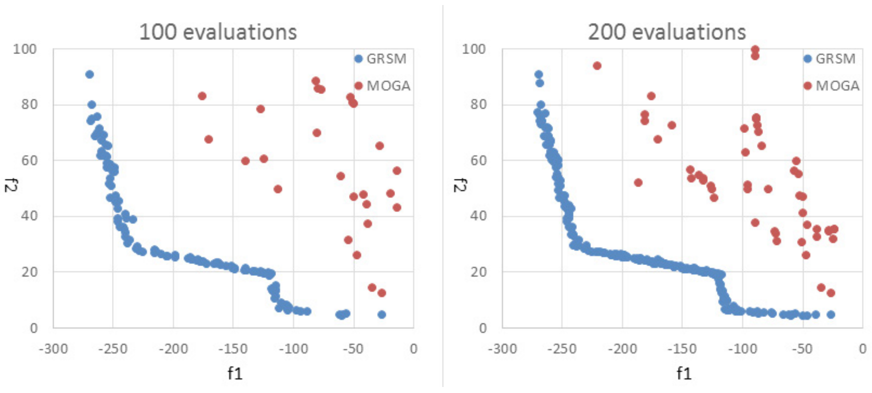


Figure 24: Superimposed results of 5 runs each of GRSM and MOGA on OSY function.

5. Conclusions

Benchmark studies have been conducted to compare the performance of GRSM, ARSM, SQP and GA on a set of single objective optimization examples. Similarly, GRSM and MOGA have been compared on a set of multi-objective optimization examples. These examples contain several features common to engineering optimization problems, and can be used to evaluate the suitability of handling issues such as multi-modality, constraints, mixed variables and strongly coupled variables.

This paper is focused on comparing the quality of the solutions at a given number of evaluations. This measure is important because engineers seldom have the luxury of allowing expensive optimization studies to converge completely. Therefore, a rapid approach toward optimal solutions early in a study is highly desired. Overall, GRSM exhibits better performance than other methods in both single objective optimization and multi-objective optimization.

6. References

[1] HyperStudy Help, 2014. http://www.altairhyperworks.com/hwhelp/Altair/hw13.0/help/hst/hst.htm

[2] Paul T. Boggs and Jon W. Tolle, Sequential Quadratic Programming, Acta Numerica, 1996

[3] Goldberg DE. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley; 1989

[4] N. Chase, M. Rademacher, E. Goodman, etc. A Benchmark Study of Optimization Search Algorithms, Red Cedar Technology.

[5] N. Chase, M. Rademacher, E. Goodman, etc. A Benchmark Study of Multi-Objective Optimization Methods, Red Cedar Technology.

[6] Carlos A. Coello Coello, Gary B. Lamont and David A. Van Veldhuizen. Evolutionary Algorithms for Solving Multi-Objective Problems, Second Edition.