Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

A MODEL-BASED APPROACH TO MULTI-OBJECTIVE OPTIMIZATION

Joshua Q Hale
Enlu Zhou

H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology
755 Ferst Drive, NW
Atlanta, GA 30332, USA

ABSTRACT

We develop a model-based algorithm for the optimization of multiple objective functions that can only be assessed through black-box evaluation. The algorithm iteratively generates candidate solutions from a mixture distribution over the solution space and updates the mixture distribution based on the sampled solutions' domination count such that the future search is biased towards the set of Pareto optimal solutions. The proposed algorithm seeks to find a mixture distribution on the solution space so that 1) each component of the mixture distribution is a degenerate distribution centered at a Pareto optimal solution and 2) each estimated Pareto optimal solution is uniformly spread across the Pareto optimal set by a threshold distance. We demonstrate the performance of the proposed algorithm on several benchmark problems.

1 INTRODUCTION

Multi-objective optimization problems consist of several objectives that are required to be optimized simultaneously. Naturally, such problems arise in many real world situations, usually with conflicting objectives, see e.g., Le Riche, Saouab, and Bréard (2003) , Park et al. (2005), Ghiasi, Pasini, and Lessard (2008). Multi-objective optimization differs from single-objective optimization in that the goal is to produce multiple solutions, where the quality of solutions is indifferent without any prior knowledge. These solutions are known as Pareto optimal solutions, which represent the trade-offs among multiple objective functions. Since there is not one optimal solution, comparing results of one method to another is more difficult than in the single-objective case. The goal in multi-objective optimization is to obtain uniformly distributed solutions close to the true Pareto front.

An extensive amount of research has been applied to develop methods that solve multi-objective optimization problems. A survey of these methods is included in Ruzika and Wiecek (2005). In particular, evolutionary algorithms are a popular approach, since they have the ability to synchronously search for multiple Pareto optimal solutions, see e.g., Coello, Van Veldhuizen, and Lamont (2002), Deb (2001), Deb et al. (2002), Zitzler and Thiele (1999), Fonseca and Fleming (1995). The search is executed by a selection step followed by a generation step. In the selection step, the search selects the best solutions of the existing population; whereas, in the generation step, mutation and crossover operations are carried out in order to generate new and better solutions from the solutions obtained by the selection step. Mutation is accomplished by taking small randomized steps near the selected solutions, and the crossover operator combines parts of selected solutions to produce new solutions as seen in Fonseca and Fleming (1995). Similarly, particle swarm optimization methods are another commonly used population-based method that searches for multiple Pareto optimal solutions in parallel, see e.g., Lee et al. (2014), Mete and Zabinsky (2014), Parsopoulos and Vrahatis (2008). The particle swarm optimization method keeps a population of potential solutions. The candidate solutions are known as particles and the population of solutions is known as a swarm. The particles are manipulated by a velocity vector, which changes the position of the particle at each iteration. Parsopoulos and Vrahatis (2008) explained that the particles' movement is based not only on the history of their individual performance, but also on the performance history of the swarm. A primary drawback of these types of methods is that they require specific parameters to be optimally tuned for each problem.

Some approaches use a weighted-sum method to evaluate the quality of a solution. Weighted-sum methods avoid the complexity of optimizing multiple objectives by converting the multi-objective problem into a single-objective problem via assigning different weights to the objective functions, see e.g., Jin, Olhofer, and Sendhoff (2001), Athan and Papalambros (1996) , Das and Dennis (1998). The optimization of the single-objective problem, given a weight vector, corresponds to finding one point of the Pareto front. Therefore, optimizing the single-objective with different weight values can correspond to many different points on the Pareto front. Marler and Arora (2010) pointed out that the major problems with this approach are that a uniform spread of the weighting coefficients does not necessarily produce a uniform spread of Pareto optimal solutions, and a considerable computational burden is caused by solving the single optimization problem repeatedly .

The methods mentioned above are instance-based methods, as they maintain a population of candidate solutions, and new candidate solutions are generated by only considering previous generated solutions. Conversely, in model-based search algorithms, candidate solutions are generated using a parameterized probability model that is updated using previous candidate solutions. The update process ensures that the probability model will focus the search in regions containing the top performing samples, see e.g., Zlochin et al. (2004), Hu et al. (2012). Model-based algorithms are effective at optimizing poorly-structured single-objective functions. Since the majority of model-based algorithms are designed to only generate one optimal solution, they mostly have been used to solve single-objective functions. Considering the fact that model-based algorithms are well suited to solve ill-structured single-objective functions, it seems rational to extend the model-based approach to multi-objective problems. We accomplish this by using mixture probability distributions to generate the candidate solutions. The advantage of using mixture probability distributions is that a combination of multiple simple probability distributions can capture an accurate assessment of the set of Pareto optimal solutions in a complex decision space.

Therefore, we propose a model-based multi-objective algorithm that generates candidate solutions from a mixture probability distribution model and updates the parameters of the model with the goal of obtaining degenerate simple probability models that are concentrated on Pareto optimal solutions. During an iteration of our algorithm, we determine the top performing samples based on their domination counts, and we cluster the samples in the decision space according to a threshold distance, where each cluster is used to describe a simple probability distribution. We terminate the algorithm when each simple probability model is concentrated on a solution that has a relative domination count of zero. The advantages of our algorithm are that (1) the final solutions are approximately uniformly spread along or close to the Pareto optimal set by a threshold distance that can be set by the user; and (2) our algorithm is not sensitive to the structure of the Pareto-optimal front.

We evaluate the performance of our model-based multi-objective algorithm on test problems based on convergence and diversity metrics. We solved problems SCH from Schaffer (1985), FON from Fonseca and Fleming (1998), and ZDT4 and ZDT6 from Zitzler, Deb, and Thiele (2000). All of the mentioned problems are on continuous domains and we compared our results with the well-known Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II) proposed by Deb et al. (2002).

In section 2, we provide more insight behind our method by reviewing model-based methods for single-objective optimization problems, and introduce our model-based multi-objective algorithm. We provide the results of our computational experiments in section 3. In the last section, we conclude with a discussion of some future research directions.

2 Model-Based Method for Multi-Objective Optimization

We present a review of the current model-based methods for solving single-objective optimization problems in order to describe the central idea of model-based methods. The model-based multi-objective algorithm extends the general framework of model-based methods for single-objective problems such that it can be used for optimization problems that have multiple objectives. Since our proposed algorithm deals with multiple objectives and produces multiple solutions, our model-based approach differs greatly from model-based methods used to solve single-objective optimization problems.

2.1 Review of Model-Based Methods for Single-Objective Optimization

The goal in single-objective optimization is to find a feasible solution x^* that achieves the best value of an objective function. More formally, the problem is mathematically described as:

$$x^* \in \arg\max_{x\in X} h(x), \tag{1}$$

where x is a vector of decision variables, X is the feasible solution space, and $h(\cdot)$ is a bounded function. The model-based search approach uses the general framework below to solve the single-objective optimization problem:

1. Generate candidate solutions from a parameterized probability distribution over the decision space.
2. Update the sampling distribution based on the evaluation of generated candidate solutions.

The fundamental idea is to iteratively construct a probability distribution such that the limiting distribution will allocate most of its mass on the optimal solution x^*. Essentially, the hope is that the resulting probability distribution is concentrated on the optimal solution. Typically, the top performing samples are used to update the sampling distribution. This ensures that after each iteration, more weight is assigned to the area that contains the best samples from the previous iteration. Examples of model-based algorithms include the cross entropy method (Rubinstein (2001)), model reference adaptive search (Hu, Fu, and Marcus (2007)), and gradient-based adaptive stochastic search (Zhou and Hu (2014)). The main difference among the various model-based methods is the way they update the sampling distribution.

2.2 Model-Based Multi-Objective Algorithm

A multi-objective optimization problem is defined as

$$\min_{x\in S} \; h(x) \quad = \quad [h_1(x), h_2(x), \ldots, h_n(x)]$$

where $x = [x(1), \ldots, x(m)]$ is the vector of decision variables and each h_i maps from the feasible set S to a continuous real number for $i = 1, \ldots, n$ and $n \geq 2$. We assume the feasible set S is bounded. Since we have a multiple number of objectives that we seek to optimize simultaneously, non-dominated solutions are considered to be Pareto optimal. A vector x^1 dominates x^2 if and only if $h_i(x^1) \leq h_i(x^2)$ for all $i = 1, 2, \ldots, n$ and there exists a j such that $h_j(x^1) < h_j(x^2)$. A solution x^* is said to be Pareto optimal if and only if there does not exist a solution $x' \in S$ that dominates x^*. The set of all Pareto optimal solutions is denoted as the Pareto optimal set P^* and the Pareto front is defined as $PF^* = \{h(x) : x \in P^*\}$.

The goal of our algorithm is to find a diverse set of Pareto optimal solutions such that the decision maker can choose the solution that fits best to her preferences. This is accomplished by finding a mixture distribution on the decision space so that each mixture component of the mixture distribution is a degenerate distribution concentrated at a Pareto optimal solution x^* and each x^* is distributed evenly across the Pareto optimal set by a threshold distance Δ. During each iteration, a mixture component of the mixture distribution corresponds to a cluster in the decision space, and the parameters of each mixture component are updated separately based on the solutions with the lowest domination count in the current iteration. The threshold distance for the next iteration is updated by the candidate solutions in the current iteration. The domination count of x^i is defined as the number of candidate solutions in the current iteration's population of solutions that dominates x^i. The algorithm terminates when the threshold distance is below the desirable threshold bound $\bar{\Delta}$. The full description of the algorithm is below.

Algorithm 1: Model-Based Multi-Objective Algorithm

```
1 Initialization: Determine values for $N, p, C > 1$ and $\bar{\Delta}$. Let $n_0 = 1$ and $t = 0$.
2 Sampling: Generate $N$ candidate solutions $x_t = [x_t^1, \dots, x_t^N]$ corresponding to the mixture distribution
      $$f(x) = \sum_{k=1}^{n_t} \frac{1}{n_t} f(x; \theta_k^t),$$
  where $\theta_k^t$ is the parameter of the $k$th Normal distribution at iteration $t$.
3 Evaluation: Evaluate the function value $h_j(x_t^i)$ for all $i$ and $j$. Calculate the domination count of
  each candidate solution, and rank the candidate solutions by their domination counts in ascending
  order.
4 Selection: Select the top $\lfloor pN \rfloor$ ranked candidate solutions as elite solutions and add any additional
  solution that has a domination count equal zero.
5 Clustering: If $t = 0$, calculate the threshold distance
      $$\Delta_0 = \sum_{i=1}^{m} \sigma_1^t(i),$$
  where $\sigma_k^t(i)$ is the standard deviation of the candidate solutions of the $i$th dimension for cluster $k$.
  Cluster elite solutions based on the threshold-based clustering algorithm with threshold distance $\Delta_t$
  detailed in Algorithm 2 below. Let $n_{t+1}$ be the resulting number of clusters produced by the
  clustering algorithm.
6 Updating: For each cluster $C_{t,k}$, calculate the sample standard deviation $\sigma_k^t$ and sample mean $\mu_k^t$.
  Perform a smoothing update on the standard deviation $\sigma_k^t$ of each cluster $C_{t,k}$
      $$\sigma_k^t = \alpha \sigma_k^t + (1-\alpha) \max_{l=1,\dots,n_{t-1}} \sigma_l^{t-1}.$$
  Update the parameters of the mixture distribution $\theta_k^{t+1} = [\mu_k^t, \sigma_k^t]$ for all $k$. Update the threshold
  distance
      $$\Delta_{t+1} = \frac{1}{C} \sum_{i=1}^{m} \left[ \frac{1}{n_t} \sum_{k=1}^{n_t} \sigma_k^t(i) \right]$$
7 Stopping: If $\Delta_{t+1} > \bar{\Delta}$ , let $t \leftarrow t+1$ and go back to step 2.
```

In Step 2 of our algorithm, we generate candidate solutions according to a mixture distribution. Each simple probability distribution $f(x;\theta_k)$ in the mixture probability distribution is denoted as a mixture component, and the probability λ_k associated with each component is referred to as the mixture weight. A mixture probability distribution is defined as

$$f(x) = \sum_{k=1}^{n} \lambda_k f(x;\theta_k),$$

where $\lambda_k \geq 0$ for all k and $\sum_{k=0}^{n} \lambda_k = 1$. The mixture distribution allows us to use each mixture component to describe a specific promising area of the decision space. Moreover, the different mixture components aid in providing diversity among the set of estimated Pareto optimal solutions.

In Step 3, for each candidate solution x_i we evaluate the function value $h(x^i) = [h_1(x^i), \dots, h_n(x^i)]$. We determine the domination count with respect to the solutions in the current iteration and rank the candidate solutions from smallest to largest according to their domination count. This step is key to the incorporation of the model-based approach. In essence, the domination count of the solutions corresponds to the objective value in the single-objective case, where the performance of a solution depends on its objective value. Conversely, in the multi-objective case the solutions that have low domination counts are preferred. Instead of finding a single global optimal solution, we seek to find multiple solutions that have a domination count of zero.

In step 4 of the model-based multi-objective algorithm, we select the top $\lfloor pN \rfloor$ samples with respect to their domination counts of the current iteration. All solutions with a domination count of zero are also selected as elite solutions. This ensures that all solutions that are currently considered to belong to the Pareto optimal set are used to update the mixture distribution for the next iteration. Essentially, this step enables the mixture components of the distribution to move toward the region of the decision space where solutions have low domination counts.

In step 5 of our algorithm, we cluster the solutions based on the threshold-based clustering algorithm Bhatia (2004). The implementation of the algorithm is presented below. Although the number of resulting clusters is not determined a priori, the distance between the median of each cluster is at least as far as Δ, which aids in providing an approximate uniform distribution of the location of the clusters in the decision space. The use of clustering allows us to efficiently break up the elite solutions and group them in order to describe each group by their individual mixture component. We denote $N_{t,k}$ by the number of candidate solutions in cluster $C_{t,k}$. Each cluster k is represented by a $N_{t,k}$ x m matrix

$$C_{t,k} = \begin{bmatrix} x^1_{t,k}(1) & x^1_{t,k}(2) & x^1_{t,k}(3) & \cdots & x^1_{t,k}(m) \\ x^2_{t,k}(1) & x^2_{t,k}(2) & x^2_{t,k}(3) & \cdots & x^2_{t,k}(m) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x^N_{t,k}(1) & x^N_{t,k}(2) & x^N_{t,k}(3) & \cdots & x^N_{t,k}(m) \end{bmatrix}, \tag{2}$$

where $x^i_{t,k}(j)$ represents the jth dimension of the ith candidate solution of cluster $C_{t,k}$. This step allows the algorithm to produce multiple estimated Pareto optimal solutions. Note that the number of clusters generated will heavily depend on the size of the Pareto optimal set.

Algorithm 2: Threshold-Based Clustering Algorithm

```
1 Initialization: Select the first solution from the elite set. This solution is defined as the median of
  cluster $C^t_1$.
2 Select the next solution from the elite set that is not assigned to a cluster, calculate the distance to
  the median of each existing cluster. If the distance is less than the threshold distance $\Delta$, assign the
  solution to this cluster and update the median of the cluster as the average of the solutions in the
  cluster. Otherwise, create a new cluster where the solution is the median of the new cluster.
3 Merge clusters if the distance between the median of two clusters is less than the threshold.
4 If there exists an unclassified solution, go to step 2; otherwise, terminate the algorithm.
```

In the final step of the algorithm, we calculate the sample mean and standard deviation of the elite solutions in each cluster. We perform a smoothing update on the standard deviation for each cluster, which takes into account the current standard deviation of the cluster and the maximum standard deviation of all clusters from the previous iteration. This step ensures global exploration of the decision space and prevents clusters from concentrating at a local optimum point. We update the parameters of the mixture distribution based on the sample mean and the adjusted standard deviation. Since each cluster corresponds to a mixture component, we can accurately fit the mixture distribution over the Pareto optimal set. The mixing weight is chosen to be $\lambda_k^t = 1/n_t$. Giving each mixture component an equal probability of generating a candidate solution ensures that the solutions are distributed as evenly as possible in the decision space. Drawing new solutions from the resulting mixture probability distribution results in robust exploration in the decision space. We point out that since this is a search algorithm, the domination count for each candidate solution is with respect to the solutions generated in the current iteration.

The threshold distance for the next iteration is calculated by the sum over all dimensions of the average standard deviation of the clusters divided by some constant C greater than one. Essentially, the threshold distance plays a similar role as the variance parameter in the single-objective case. The hope is that the threshold distance converges to the threshold bound, which is set to a value close to zero, where each cluster is only made up of one point. Since the standard deviation is bounded by the threshold distance, dividing by the constant C ensures that the standard deviation of the clusters are non-increasing. The algorithm is terminated with the threshold distance is below a specified threshold bound $\bar{\Delta}$. Since each cluster converges to one point, the final estimated Pareto front consists of the means of the final clusters.

3 Numerical Results and Discussion

To evaluate the performance of our model-based multi-objective algorithm, we considered the following test functions:

- SCH (Schaffer (1985))

$$\begin{aligned} h_1(x) &= x^2 \\ h_2(x) &= (x-2)^2 \\ x &\in [-10^3, 10^3] \\ \text{Optimal Solutions: } x^* &\in [0,2] \end{aligned}$$

- FON (Fonseca and Fleming (1998))

$$\begin{aligned} h_1(x) &= 1-\exp(-\sum_{i=1}^{3}(x_i - \frac{1}{\sqrt{3}})^2) \\ h_2(x) &= 1-\exp(-\sum_{i=1}^{3}(x_i + \frac{1}{\sqrt{3}})^2) \\ x_i &\in [-4,4], \quad i = 1,2,3 \\ \text{Optimal Solutions: } x_i^* &\in [\frac{-1}{\sqrt{3}}, \frac{1}{\sqrt{3}}],\ i = 1,2,3 \end{aligned}$$

- ZDT4 (Zitzler, Deb, and Thiele (2000))

$$\begin{aligned} h_1(x) &= x_1 \\ h_2(x) &= g(x)(1-\sqrt{x_1/g(x)}) \\ g(x) &= 1+10(n-1)+\sum_{i=2}^{10}(x_i^2 - 10\cos(4\pi x_i)) \\ x_1 &\in [0,1] \\ x_i &\in [-5,5], \quad i = 2,\dots,10 \\ \text{Optimal Solutions: } x_1^* &\in [0,1] \\ x_i^* &= 0, \quad i = 2,\dots,10 \end{aligned}$$

- ZDT6 (Zitzler, Deb, and Thiele (2000))

$$
\begin{aligned}
h_1(x) &= 1-\exp(-4x_1)\sin^6(6\pi x_1) \\
h_2(x) &= g(x)(1-(h_1(x)/g(x))^2) \\
g(x) &= 1+9((\sum_{i=2}^{10} x_i)/(n-1))^{\cdot}25 \\
x_i &\in [0,1], \quad i=2,\ldots,10 \\
\text{Optimal Solutions: } x_1^* &\in [0,1] \\
x_i^* &= 0, \;\; i=2,\ldots,10
\end{aligned}
$$

We compared our method with the Elitist Non-Dominated Sorting Genetic Algorithm NSGA-II Deb et al. (2002). In NSGA-II an initial population of solutions is generated. Then the population of solutions is divided based on the following rule: the first group of solutions is all non-dominated and the second group of solutions is only dominated by the solutions in the first group. This grouping is continued in this manner until all solutions are classified. Once the solutions are divided, the crowding distance is calculated, which measures how close a solution is to its neighbors. Solutions are selected based on their group classification and crowding distance, and new solutions are generated from crossover and mutation operators.

Similarly to Deb et al. (2002), we use the convergence and the diversity metrics to evaluate the effectiveness of our algorithm. The convergence metric Γ measures the closeness of the solutions generated by our algorithm to the known set of Pareto-optimal solutions. We created a reference set R by generating 500 uniformly spaced solutions from the true Pareto-optimal front. For each solution obtained by our algorithm, we calculated the minimum Euclidean distance to the reference set. The convergence metric is the average of the calculated distances, as given by

$$\Gamma = \frac{1}{|R|}\sum_{y:h(y)\in R}\left\{\min_{x\in Z}(h_i(y)-h_i(x))\right\}, \tag{3}$$

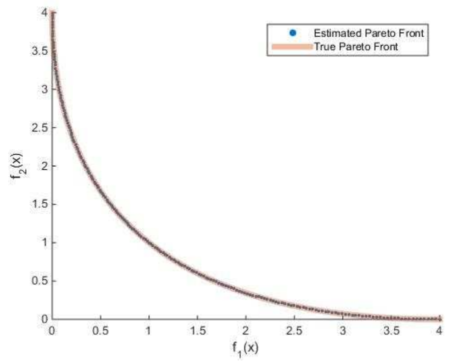
where Z is the set of the estimated Pareto optimal solutions produced by our algorithm, and $|R|$ denotes the cardinality of the set R. The smaller the value is for this metric, the closer our estimated Pareto optimal solutions are to the true Pareto optimal solutions.

The diversity metric δ measures how well the solutions are evenly spread in the decision space. We denote the Euclidean distance between two consecutive solutions in the estimated Pareto optimal solution set by d_i, and the average of these distances by $\bar{d}$. We determined consecutive solutions by starting from the solution whose value is closest to the left boundary point of the domain and using the nearest neighbor heuristic to identify the adjacent solutions. We obtain the two boundary points on the true Pareto front by choosing the rightmost point y_r and the leftmost point y_l on the Pareto front. We denote $d_l = \min_{x\in Z}(d(x,y_r))$ and $d_r = \min_{x\in Z}(d(x,y_l))$, as the distance between the left and the right boundary points, respectively. Note that $d(x,y)$ is the Euclidean distance between point x and y. The diversity metric is calculated by

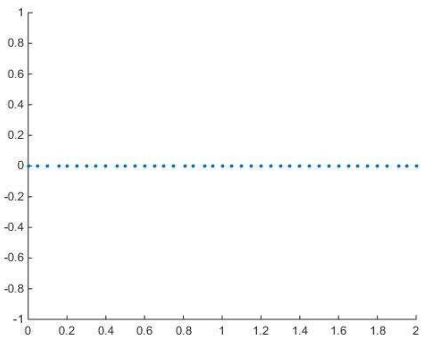
$$\delta = \frac{d_r + d_l + \sum_{i=1}^{N-1}|d_i-\bar{d}|}{d_r+d_l+(N-1)\bar{d}}, \tag{4}$$

where N is the number of solutions generated by our algorithm. The goal of our algorithm is to have the solutions evenly spread across the entire Pareto optimal set. Therefore, similar to the convergence metric, the target value for this metric is zero. A value of zero would imply that the distance between every two consecutive solutions is the same and the two boundary solutions were obtained by our algorithm.

For each problem instance, we performed 20 independent replications of our algorithm implemented in MATLAB. We reported the average convergence metric (Γ), the average diversity metric (δ), and the standard error of the convergence metric $SE(\Gamma)$ and diversity metric $SE(\delta)$ obtained out of the 20 trials.



(a) Pareto Front



(b) Estimated Pareto Optimal Set

Figure 1: SCH results.

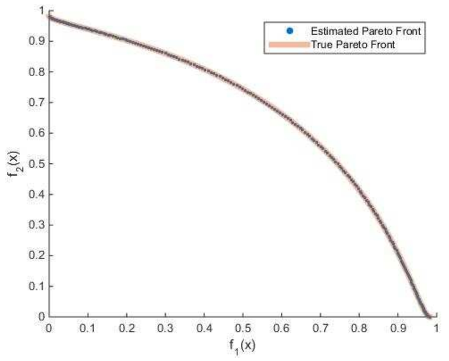
We chose the following initial parameters: sample size $N = 3000$, percent quantile $p = .10$, smoothing constant $\sigma = .7$, threshold bound $\bar{\Delta} = .0001$, constant $C = 1.5$, initial mean $\mu_0 = 0$ and initial standard deviation $\sigma_0 = 1000$. The values for the convergence and diversity metric from the NSGA-II algorithm were taken from Deb et al. (2002). The results from the computational experiments are presented in Table 1 and the estimated Pareto front and the Pareto optimal set of each problem is illustrated in Figures 1-4. Each figure (a) shows the true Pareto front and the estimated Pareto front produced by the proposed algorithm, where each point $(f_1(x), f_2(x))$ on the graph corresponds to solution x. In figure (b) we ordered the solutions and then graphed every other 20th solution in order to better view how well the solutions are spread along the optimal set. To illustrate the Pareto estimated set for problems with $m > 3$, we used the Principal Component Analysis method to perform a linear dimensionality reduction. The model-based multi-objective algorithm outperformed both NSGA-II methods with respect to the convergence metric three out of the four functions tested. Our algorithm received the lowest values for the diversity metric for all four functions. This is most likely because the median of each cluster represents an estimated Pareto optimal solution and the median of each cluster is separated by a threshold distance. As a result, the distance between each estimated Pareto optimal solution is approximately equal to the threshold bound. The average number of function evaluations for SCH, FON, ZDT4, and ZDT6 is 16521, 18342, 27541, and 29342, respectively.

Table 1: Convergence and diversity metrics.

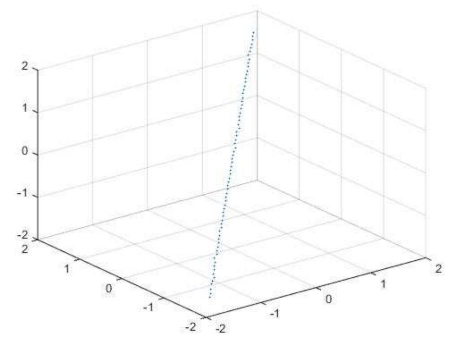
Problem	Model-Based Multi-Objective				NSGA-II (Real Coded)		NSGA-II (Binary-Coded)	
	Γ	$SE(\Gamma)$	δ	$SE(\delta)$	Γ	δ	Γ	δ
SCH	.002145	.000482	.015623	0.000321	.003391	.477899	.002833	.449265
FON	.00072	0	.018921	.000978	.001931	.378065	.002571	.395131
ZDT4	.432178	.0143	.03678	.00568	.513053	.702612	3.227636	.479475
ZDT6	.32145	.00843	.03256	.00732	.296564	.668025	7.806798	.644477

4 Conclusion

We developed a model-based multi-objective algorithm that uses a model-based approach in solving multi-objective optimization problems. The goal is that each simple probability distribution that forms the mixture distribution is concentrated on a Pareto optimal solution. The resulting estimated Pareto optimal solutions from our algorithm are separated by the threshold distance.

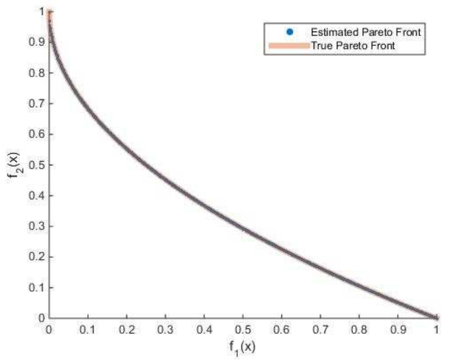


(a) Pareto Front

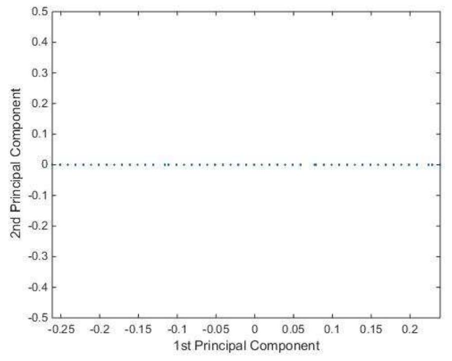


(b) Estimated Pareto Optimal Set

Figure 2: FON results.

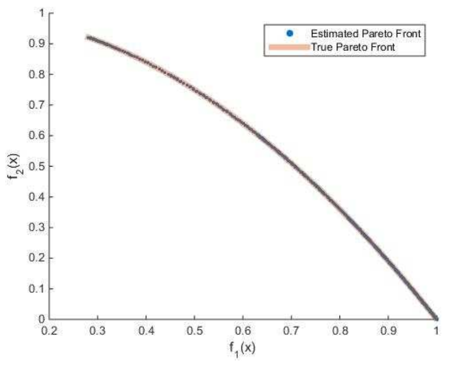


(a) Pareto Front

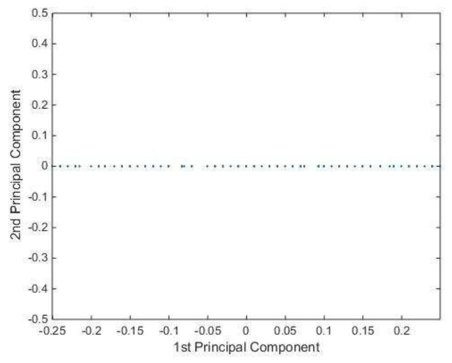


(b) Estimated Pareto Optimal Set

Figure 3: ZDT4 results.



(a) Pareto Front



(b) Estimated Pareto Optimal Set

Figure 4: ZDT6 results.

The numerical results illustrate that our proposed algorithm performs better than or similarly to the NSGA-II method with respect to both convergence and diversity metrics on the test functions presented in this paper. We plan to expand this method to handle the stochastic version of this problem. Moreover, we can establish theoretical analysis of the convergence of our algorithm.

REFERENCES

Athan, T. W., and P. Y. Papalambros. 1996. "A Note on Weighted Criteria Methods for Compromise Solutions in Multi-Objective Optimization". *Engineering Optimization* 27 (2): 155–176.

Bhatia, S. K. 2004. "Adaptive K-Means Clustering.". In *FLAIRS Conference*, 695–699.

Coello, C. A. C., D. A. Van Veldhuizen, and G. B. Lamont. 2002. *Evolutionary Algorithms for Solving Multi-Objective Problems*, Volume 242. Springer.

Das, I., and J. E. Dennis. 1998. "Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems". *SIAM Journal on Optimization* 8 (3): 631–657.

Deb, K. 2001. *Multi-Objective Optimization Using Evolutionary Algorithms*, Volume 16. John Wiley & Sons.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan. 2002. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II". *Evolutionary Computation, IEEE Transactions on* 6 (2): 182–197.

Fonseca, C. M., and P. J. Fleming. 1995. "An Overview of Evolutionary Algorithms in Multiobjective Optimization". *Evolutionary Computation* 3 (1): 1–16.

Fonseca, C. M., and P. J. Fleming. 1998. "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms. I. A Unified Formulation". *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 28 (1): 26–37.

Ghiasi, H., D. Pasini, and L. Lessard. 2008. "Constrained Globalized NelderMead Method for Simultaneous Structural and Manufacturing Optimization of a Composite Bracket". *Journal of Composite Materials* 42 (7): 717–736.

Hu, J., M. C. Fu, and S. I. Marcus. 2007. "A Model Reference Adaptive Search Method for Global Optimization". *Operations Research* 55 (3): 549–568.

Hu, J., Y. Wang, E. Zhou, M. C. Fu, and S. I. Marcus. 2012. "A Survey of Some Model-Based Methods for Global Optimization". In *Optimization, Control, and Applications of Stochastic Systems*, 157–179. Springer.

Jin, Y., M. Olhofer, and B. Sendhoff. 2001. "Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does It Work and How?". *In Proc: GECCO 2001 Conf*.

Le Riche, R., A. Saouab, and J. Bréard. 2003. "Coupled Compression RTM and Composite Layup Optimization". *Composites Science and Technology* 63 (15): 2277–2287.

Lee, L. H., E. P. Chew, Q. Yu, H. Li, and Y. Liu. 2014. "A Study on Multi-Objective Particle Swarm Optimization with Weighted Scalarizing Functions". In *Proceedings of the 2014 Winter Simulation Conference*, 3718–3729. Piscattaway, New Jersey: IEEE Press: Institute of Electrical and Electronics Engineers, INC.

Marler, R. T., and J. S. Arora. 2010. "The Weighted Sum Method for Multi-Objective Optimization: New Insights". *Structural and Multidisciplinary Optimization* 41 (6): 853–862.

Mete, H. O., and Z. B. Zabinsky. 2014. "Multiobjective Interacting Particle Algorithm for Global Optimization". *INFORMS Journal on Computing* 26 (3): 500–513.

Park, C. H., W. I. Lee, W. S. Han, and A. Vautrin. 2005. "Multiconstraint Optimization of Composite Structures Manufactured by Resin Transfer Molding Process". *Journal of Composite Materials* 39 (4): 347–374.

Parsopoulos, K. E., and M. N. Vrahatis. 2008. "Multi-Objective Particles Swarm Optimization Approaches". *Multi-Objective Optimization in Computational Intelligence: Theory and Practice*:20–42.

Rubinstein, R. Y. 2001. “Combinatorial Optimization, Cross-Entropy, Ants and Rare Events”. In *Stochastic Optimization: Algorithms and Applications*, 303–363. Springer.

Ruzika, S., and M. M. Wiecek. 2005. “Approximation Methods in Multiobjective Programming”. *Journal of Optimization Theory and Applications* 126 (3): 473–501.

Schaffer, J. D. 1985. “Multiple Objective Optimization with Vector Evaluated Genetic Algorithms.”. In *Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, July 1985*, 93–100.

Zhou, E., and J. Hu. 2014. “Gradient-Based Adaptive Stochastic Search for Non-Differentiable Optimization”. *IEEE Transactions on Automatic Control* 59:1818–1832.

Zitzler, E., K. Deb, and L. Thiele. 2000. “Comparison of Multiobjective Evolutionary Algorithms: Empirical Results”. *Evolutionary computation* 8 (2): 173–195.

Zitzler, E., and L. Thiele. 1999. “Multiobjective Evolutionary Algorithms: A Comparative Case Study and The Strength Pareto Approach”. *Evolutionary Computation, IEEE transactions on* 3 (4): 257–271.

Zlochin, M., M. Birattari, N. Meuleau, and M. Dorigo. 2004. “Model-Based Search for Combinatorial Optimization: A Critical Survey”. *Annals of Operations Research* 131 (1-4): 373–395.

AUTHOR BIOGRAPHIES

Joshua Q Hale is a Ph.D student in the H. Milton Stewart School of Industrial and Systems Engineering at Georgia Institute of Technology. His research interests include simulation optimization. His email address is jhale32@gatech.edu.

Enlu Zhou is an Assistant Professor in the H. Milton Stewart School of Industrial and Systems Engineering at Georgia Institute of Technology. She received the B.S. degree with highest honors in electrical engineering from Zhejiang University, China, in 2004, and received the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 2009. Her research interests include stochastic control and simulation optimization, with applications towards financial engineering. Her e-mail address is enlu.zhou@isye.gatech.edu and her web page is http://enluzhou.gatech.edu/.